AIDA

Communicator

Description

Revision: see Revision Index

Contents

0 Revision Index

Date	Author	Rev.	Ref.	Type	Description
2010-03-30	T. Grewe	1.11.00	4	cont.	Changed connecting to PI from core.run to core.listenfor commands.
2006-01-17	H. Schmidts	1.10.00			Hyperlinks updated.
2004-09-15	K.-H. Damm	1.09.00	2.1 2.2.1.5	cont. cont.	Stacker screenshots updated for English version. Section on “Extensions” menu added.
2004-09-15	R. Dzionsko	1.08.00	2.2.8 4.1.2 to 4.1.6	cont.	Added network management sections
2004-02-19	K.-H. Damm	1.07.00	2.2.1 2.2.3 2.2.6	cont. cont. cont.	Screenshots updated. Descriptions for new menu items added. Section on tx nodes panel added. Section on timeout panels added.
2004-01-29	K.-H. Damm	1.06.00	2.2.2 2.2.4 2.2.5 2.2.7.2	cont. cont. cont. cont.	Screenshots updated. Usage of marker images for descriptions. Several explanations added. Screenshots updated. Usage of marker images for descriptions. Several explanations added. Usage of marker images in screenshots for descriptions. Usage of marker images in screenshots for descriptions.
2004-01-21	K.-H. Damm	1.05.00	2.2.6.2	cont.	Note concerning *“_bN”*-signals added.
2004-01-15	K.-H. Damm	1.04.00	2.2.6 2.2.1.1	cont. cont.	Section on VolCANo extensions added. Description for MRU files menu item extended.
2003-12-17	K.-H. Damm	1.03.00	4.3	cont.	Section “How to select a message for …” added.
2003-08-27	K.-H. Damm	1.02.00	2.2.1 2.2.4 4 var.	cont. cont. cont. cont.	Descriptions for menu items and tool buttons added. Descriptions of user panels added. FAQ chapter added. Illustrations updated for current Communicator version.
2003-04-30	K.-H. Damm	1.01.00	var.	edit.	Translation from German into English.
2003-03-20	K.-H. Damm	1.00.00	var.	edit., cont.	First version as a printable document.

1 Introduction

The AIDA Communicator is a tool to generate (in particular: cyclic) messages respectively signals of CAN bus participants (network nodes or control units). In addition, it also serves to monitor received messages resp. signals.

Via the CMCTR API (see the CMCTR and NMOSEK sections in the POOL Standard Modules documentation) its quite easy for any POOL programmer to extend the Communicator's functionality by instantiating and programming it within a POOL application.

2 Operating Manual

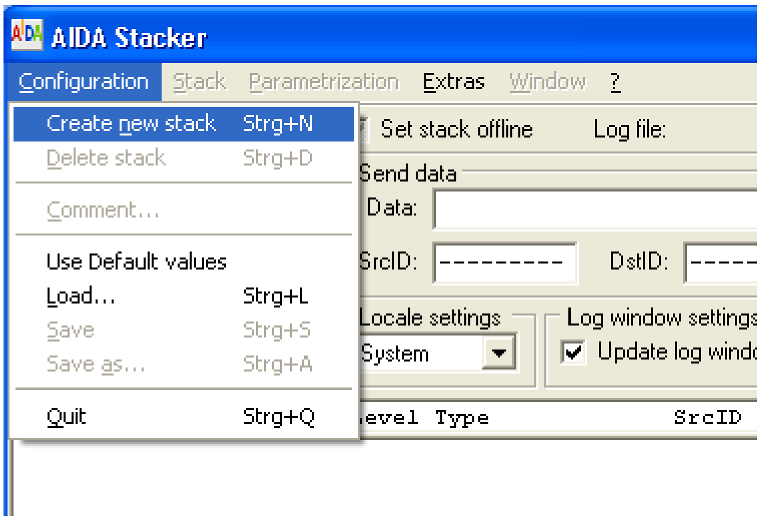
2.1 AIDA Driver Stacks

All communication is based on a pre-configured AIDA Driver Stack in the form of a **.aida-cfg* file, which first has to be created using the AIDA Stacker.

As an example we will now build a simple driver stack for a CAN connection (11-bit standard identifier length, 100 kbps bit rate):

2.1.1 Adding Stack Components

Run *AIDA_Stacker.exe* and select menu item `Create new stack`:

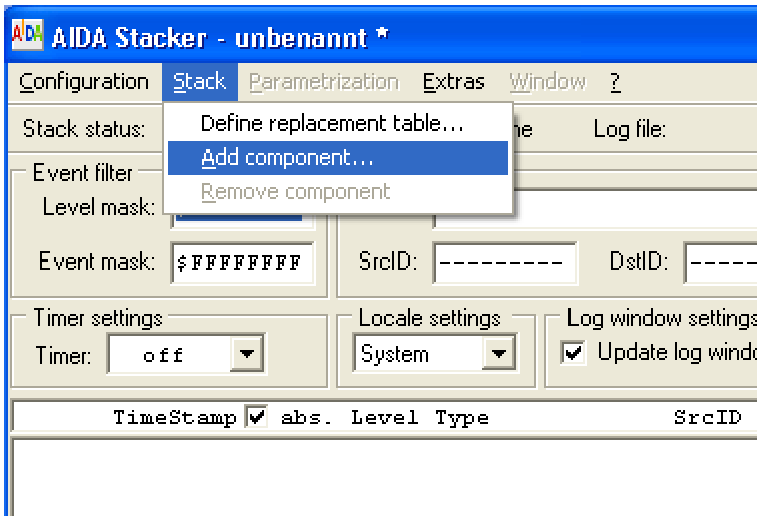


Picture 1: Creating a new driver stack configuration

AIDA stacks consist of communication layers represented by components and are built from the top level component down to a hardware driver.

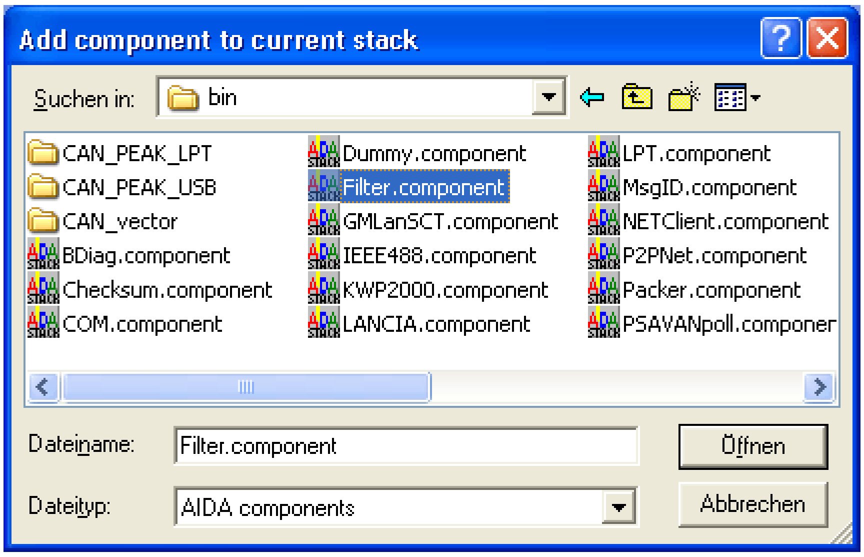
At first, the higher level protocol components are added. The last component represents the physical layer (in our case CAN).

The AIDA Communicator absolutely requires a filter component *Filter.component*, which is added with menu item `Stack - Add component ...` .



Stack status:
Event filter
Level mask:
Event mask: $FFFFFFFF
SrcID: ---------
DstID: -----
Timer: off
System

Picture 2: Adding a stack component to our driver stack



Suchen in: bin

CAN_PEAK_LPT
CAN_PEAK_USB
CAN_vector
BDiag.component
Checksum.component
COM.component
Dummy.component
Filter.component
GMLanSCT.component
IEEE488.component
KWP2000.component
LANCIA.component
LPT.component
MsgID.component
NETClient.component
P2PNet.component
Packer.component
PSAVANpoll.componer

Dateiname: Filter.component
Dateityp: AIDA components

Picture 3: Selecting the filter component

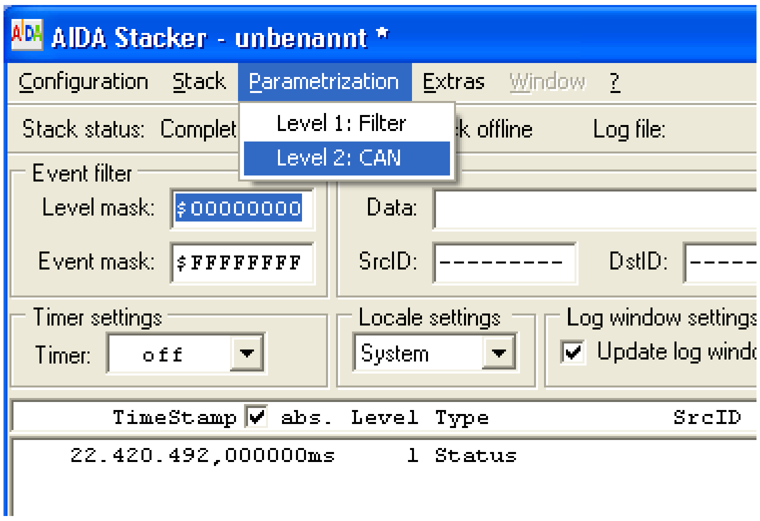
As next and last component, we will add a CAN component. The corresponding file is located in one of the CAN_* subdirectories, depending on your installed driver and hardware.

2.1.2 Configuring Stack Components

Before we can use our new stack within AIDA Communicator, we first have to parameterize the single stack components (apart from the filter component, which will later be reprogrammed by the Communicator anyway).

Note: The filter component will be completely reprogrammed by the AIDA Communicator using the object database (one or several DBC files) assigned to the same bus and a possibly previously saved Communicator configuration (*.*cmctr-cfg*).

To change the parameters of the CAN component select menu item `Parametrization - Level 2: CAN`.



Stack status: Complet ... k offline

Level mask: $00000000

Event mask: $FFFFFFFF

SrcID: --------- DstID: -----

Timer: off

Locale settings: System

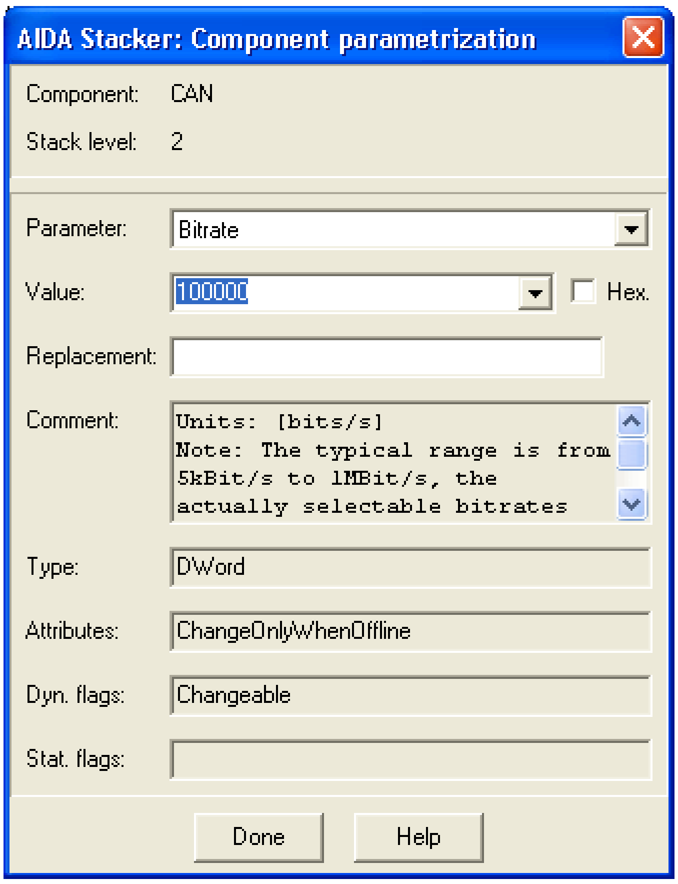
[✓] Update log windo

```
TimeStamp ✓ abs.  Level Type          SrcID
  22.420.492,000000ms     1 Status
```

Picture 4: Configuring the CAN component

Note: Most parameters of the CAN component can only be changed when the stack is offline (see attribute ChangeOnlyWhenOffline in the Stacker). Therefore the component must be disconnected first by setting the parameter `ChannelMask` to $0. This is the default setting for a newly added CAN component.

To begin with, we set the parameter `Bitrate` to „100000“ bps.



AIDA Stacker: Component parametrization

Component: CAN

Stack level: 2

Parameter: Bitrate

Value: 100000 ☐ Hex.

Replacement:

Comment:
```
Units: [bits/s]
Note: The typical range is from
5kBit/s to 1MBit/s, the
actually selectable bitrates
```

Type: DWord

Attributes: ChangeOnlyWhenOffline

Dyn. flags: Changeable

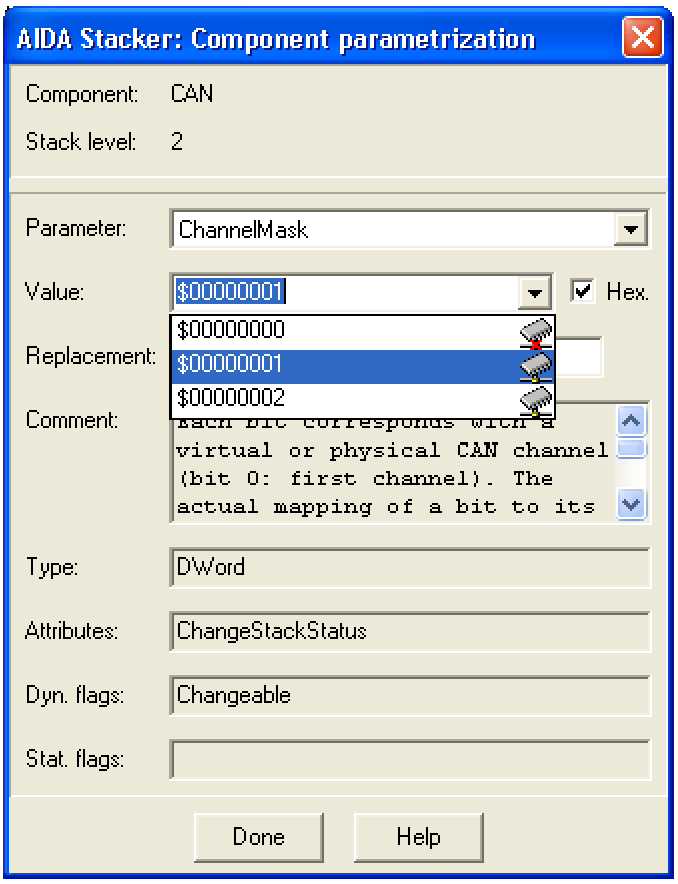
Stat. flags:

Picture 5: Setting the Bitrate parameter of the CAN component

„11bit CAN Identifier Length“ is set by parameters **`AcceptanceCode`** and **`AcceptanceMask`** equal $0 (see the comment field in the corresponding stack dialog box).

Note: The CAN protocol differentiates between 11-bit Standard Identifiers and 29-bit Extended Identifiers. As default, the CAN component is parameterized for the usage of standard identifiers. To use 29-bit extended identifiers you must change the values of the 3 following parameters from $0 to $80000000 (i.e. bit 31 = 1): **`AcceptanceMask`**, **`AcceptanceCode`** and **`SendID`**.

The last step needs configuration of the parameter **`ChannelMask`** corresponding to the existing CAN driver and CAN bus. As soon as the channel mask is set to the right value, the Stacker immediately starts displaying any incoming communication data.



AIDA Stacker: Component parametrization

Component: CAN

Stack level: 2

Parameter: ChannelMask

Value: $00000001 ☑ Hex.

$00000000
$00000001
$00000002

Replacement:

Comment:
```
Each bit corresponds with a
virtual or physical CAN channel
(bit 0: first channel). The
actual mapping of a bit to its
```

Type: DWord

Attributes: ChangeStackStatus

Dyn. flags: Changeable

Stat. flags:

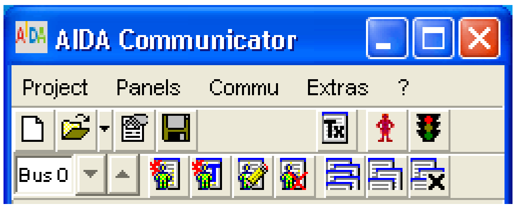
Picture 6: Setting the ChannelMask parameter of the CAN component

Now, we are finished with the stack configuration. Please save the example stack using `Configuration - Save as ...` as file *C:\test.aida-cfg* .

Note: To learn more, you should read the following sections in the AIDA Stack Components documentation: CAN component, Filter component and MsgID component.

2.2 AIDA Communicator

2.2.1 Menus and Tool Bars



Picture 7: the upper part of the main window with menu bar and tool bars

2.2.1.1 Project

New Configuration ...

Opens the main configuration dialog window, where you can assign an **.aida-cfg* file and several **.dbc* files to any one of the 4 logical buses as well as make further bus-specific and configuration-global settings.

Open Configuration ...

Opens the file selector dialog window. Choose an existing **.cmctr-cfg* file to load.

As shortcut you should use the `Most Recently Used Files >` menu.

Save Configuration

Saves a previously loaded **.cmctr-cfg* to disk or opens the file selector dialog window for a newly arranged configuration. (see `Save Configuration As ...` for details)

Save Configuration As ...

Opens the file selector dialog window. Choose an existing directory and type a filename to save your current configuration. Be aware that all absolute paths to the assigned files are automatically converted into relative paths (relative to the **.cmctr-cfg* file's location) during the saving process.

Edit Configuration ...

Opens the main configuration dialog window. Here you can change the current configuration settings (e.g. assigned **.aida-cfg* file and **.dbc* files, and further bus-specific settings) for any one of the 4 logical buses.

Most Recently Used Files >

The list of the recently used **.cmctr-cfg* files (not exceeding 10) for quick access/opening. Note: the full path of the selected file is shown in the status bar.

Exit

Stops the communication if running and terminates your current Communicator session. If you made some major changes at your configuration settings (i.e. assigned files and settings in the main configuration dialog or created, changed or deleted any user-defined panels) you will be prompted to save your configuration or cancel the exit process.

2.2.1.2 Panels

Bus 0
Bus 1
Bus 2
Bus 3

Toggles the visible bus control. Affects the following menu commands: `New Standard User-Panel ...`, `New Timeout User-Panel ...`, `Edit User-Panel ...`, `Delete User-Panel`, `Create All Panels`, `Show All Panels`, `Hide All Panels`, `Execute AIDA Stacker ...`, `Execute Dbc-Editor ...` , which all refer to the selected bus control.

New Standard User-Panel ...

Opens the user panel configuration dialog window with the DBC base for the currently visible bus control. Allows you to select messages and signals for display in a user-defined panel and to arrange them freely.

New Timeout User-Panel ...

Opens the user panel configuration dialog window with the DBC base for the currently visible bus control. Allows you to select messages for display in a user-defined timeout panel and to arrange them freely.

Edit User-Panel ...

Opens the user panel configuration dialog window with the DBC base for the currently visible bus control and the selected standard or timeout user panel (i.e. the selected entry in the user panel list, not the actual panel). Change the arrangement of an existing user-defined panel.

Delete User-Panel

Deletes the selected (i.e. the entry in the user panel list, not the actual panel) user-defined panel.

Create All Panels

Generates and opens all panels for the currently selected bus. On slow machines and for large databases this may take several minutes. Therefore the user is asked for confirmation to proceed first.

Show All Panels

Shows all open panels for the currently selected bus.

Hide All Panels

Hides all open panels for the currently selected bus.

Show TxNodes Panel

Shows the configuration-global panel which lists all nodes. Nodes with assigned tx messages may be selected for transmission, nodes which have only rx messages assigned are disabled.

2.2.1.3 Commu

Start/Stop Communication

Toggles communication for all 4 buses (if valid stack and DBC configurations have been assigned).

Activate/Deactivate Recv. Display

Toggles update of received messages within panels for all 4 buses. In the background the received messages are still processed, so that external applications instantiating Communicator via CMCTR API (see the POOL Standard Modules documentation) will still receive `COBJ_nMSFRxInd_Msk` and `COBJ_nMSFChanged_Msk` events for their registered messages and signals.

2.2.1.4 Extras

Global Settings ...

Opens a dialog window which allows the user to edit the global settings.

Execute PI/POOL Task ...

Allows to load one additional PI task. Before a new task is loaded, any previously possibly loaded task will be terminated and unloaded before. (See remarks for `LoadTask()`, `StartTask()` resp. `TerminateThread()`, `UnloadTask()` in the POOL Standard Modules documentation).

The following menu commands are only available, if you are connected via localhost (i.e. Commander/Visual Objects and PI are running on the same machine).

Execute AIDA Tracer ...

Executes/Runs a new instance of *AIDA_Tracer.exe*. If there is a **.tracer-cfg* with the same base name as the current **.cmctr-cfg*, it will be opened. Note: Internally, the environment variable `$(AIDABIN)` is used to locate the executable.

Execute AIDA Stacker ...

Executes/Runs a new instance of *AIDA_Stacker.exe* and opens the **.aida-cfg* for the currently selected bus (i.e. the currently visible bus control). Changes within this **.aida-cfg* are only used after a complete reload of the corresponding **.cmctr-cfg*. Note: Internally, the environment variable `$(AIDABIN)` resp. the registry entry for file extension *.aida-cfg* are used to locate the executable.

Execute Dbc-Editor ...

Executes/Runs a new instance of the optional DBC-Editor (e.g. CANdb++) and opens the first **.dbc* for the currently selected bus (i.e. the currently visible bus control). If there are several **.dbc* files assigned to the bus, it will only open the first in line. Changes are only used after a complete reload of the configuration. Note: Internally, the entry for `DbcEditorFullPath` in *aida-cmctr.ini* is used to locate the executable. See section 4.2 for a detailed explanation of the structure of *aida-cmctr.ini*.

2.2.1.5 Extensions

The `Extensions` menu is only available, if Communicator is started via an external application, e.g. the Communicator with Network Management (2.2.8) .

NM-Panels

A sub menu that lists only menu items specific to NMOSEK (2.2.8.2) .

2.2.1.6 ?

Help ...

Opens this documentation in your default web browser.

About ...

Opens the "Info About" dialog window, which contains the Communicator's version and build date info.

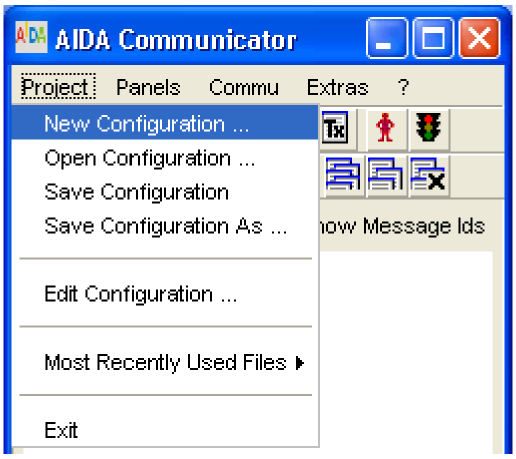
Problems Log ...

Lists all problems (warnings and errors) that occurred during your current Communicator session.

2.2.2 Basic Configuration

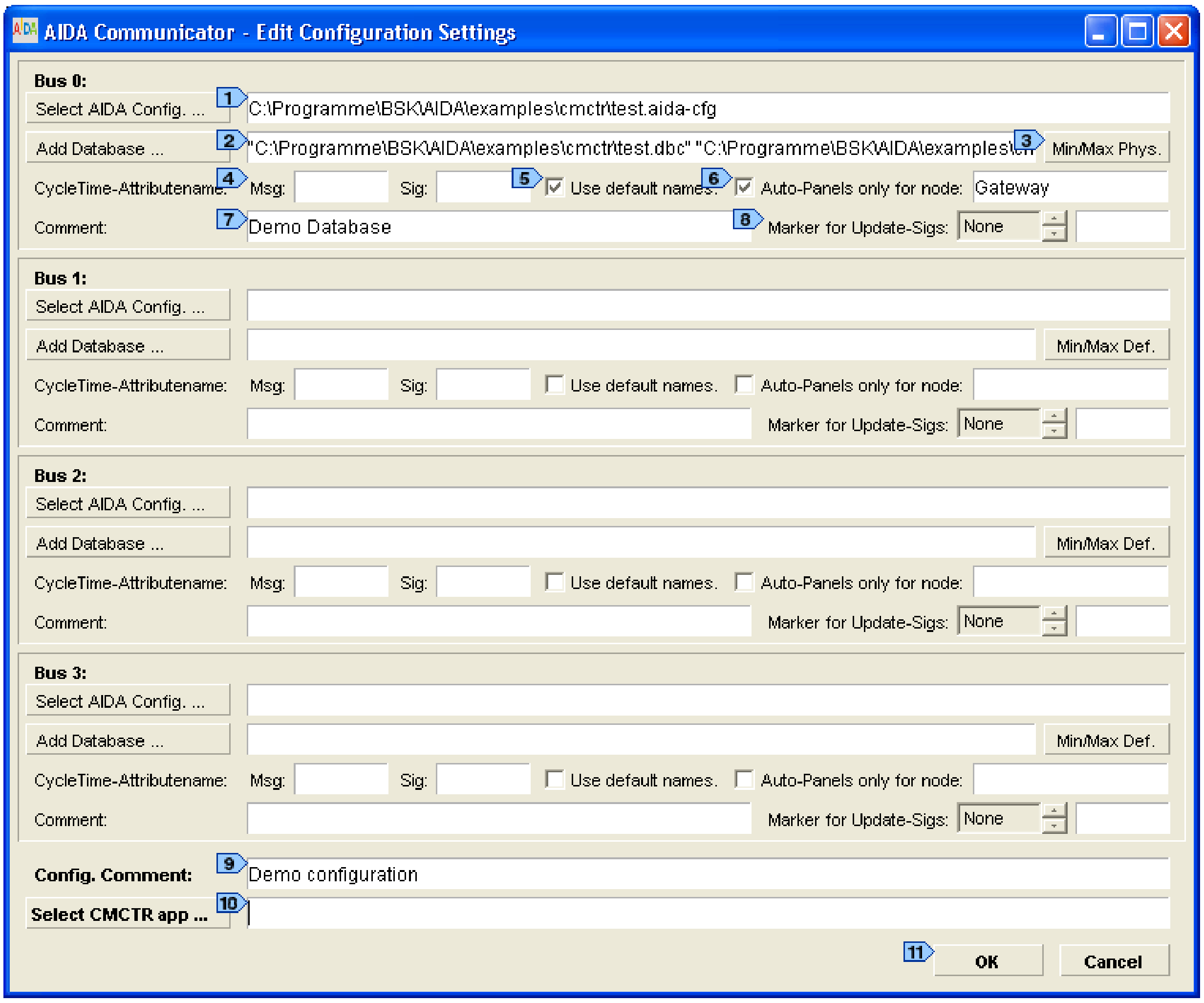
To start with a new configuration, you first need a pre-configured AIDA driver stack in the form of an **.aida-cfg* file as well as a CANdb network description in form of (at least) one **.dbc* file.

By selecting menu item `Project - New Configuration` you open the configuration dialog box.



Picture 8: Create a new Communicator configuration

The driver stack *test.aida-cfg,* which we've just built with AIDA Stacker in section 2.1, will now be assigned to the (logical) bus no. 0. Either type in the file's full path directly 1 or use the file selector dialog box which pops up when you click the `Select *.aida-cfg` button.



Bus 0:
Select AIDA Config. ... C:\Programme\BSK\AIDA\examples\cmctr\test.aida-cfg
Add Database ... "C:\Programme\BSK\AIDA\examples\cmctr\test.dbc" "C:\Programme\BSK\AIDA\examples\cm Min/Max Phys.
CycleTime-Attributename: Msg: Sig: ☑ Use default names. ☑ Auto-Panels only for node: Gateway
Comment: Demo Database Marker for Update-Sigs: None

Bus 1:
Select AIDA Config. ...
Add Database ... Min/Max Def.
CycleTime-Attributename: Msg: Sig: ☐ Use default names. ☐ Auto-Panels only for node:
Comment: Marker for Update-Sigs: None

Bus 2:
Select AIDA Config. ...
Add Database ... Min/Max Def.
CycleTime-Attributename: Msg: Sig: ☐ Use default names. ☐ Auto-Panels only for node:
Comment: Marker for Update-Sigs: None

Bus 3:
Select AIDA Config. ...
Add Database ... Min/Max Def.
CycleTime-Attributename: Msg: Sig: ☐ Use default names. ☐ Auto-Panels only for node:
Comment: Marker for Update-Sigs: None

Config. Comment: Demo configuration
Select CMCTR app ...

Picture 9: the Communicator configuration dialog window

As for the DBC files 2, the procedure is basically the same with the only difference that you may assign more than one file to each bus. Absolute paths will be replaced with relative paths when the configuration is saved to disk. See also: section 2.2.6.1 Min/Max Values 3.

Note: At the moment multi-selection of files is not supported by the file selector. Therefore each newly selected file's path is appended to the file paths already listed in the text field.

For each bus you may configure which attribute names for cycle time should be used. If you check the check box titled `Use default names` 5, the default cycle time attribute names *GenMsgCycleTime* and *GenSigCycleTime* are used. Additionally, you can type in a further specific attribute name for messages and signals respectively 4. Note: if `Use default names` 5 is checked and in addition specific name(s) are defined, both are used to determine cycle times from the database. This is necessary because there are DBC files where e.g. *GenMsgCycleTime* and *Cycletime* are used at the same time.

By checking the `Auto-Panels only for node` check box 6 and typing a valid node name in the accompanying text field, you may restrict the set of messages to display within the auto panels to those, which are relevant to the selected node, i.e. only its tx- and rx-messages.

In the text fields titled `Comment` 7, you may define a comment for each bus. This comment is displayed within the main window behind the bus number. The `Configuration Comment` 9 is not used anywhere else within Communicator, in the current version.

For an explanation of the function of the controls titled `Marker for Update-Sigs` 8 see section 2.2.6.2 Update-Bit Signals.

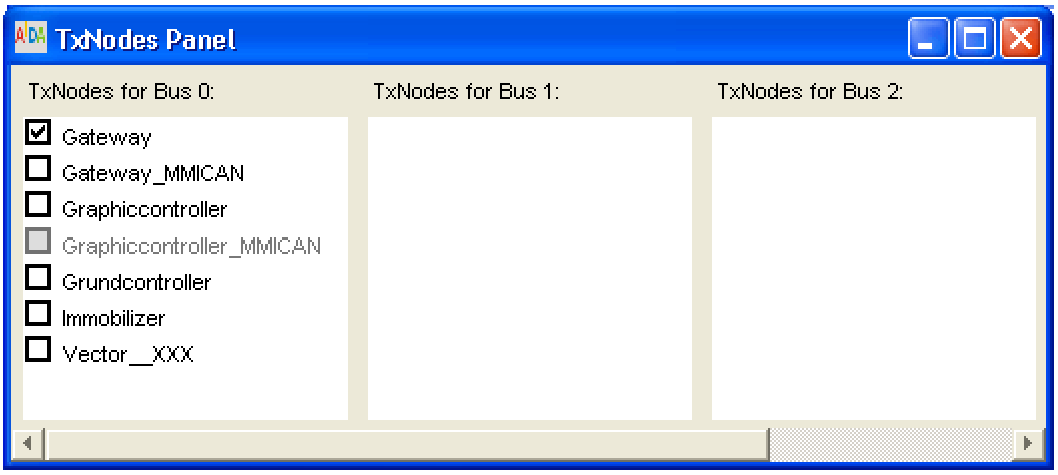
In addition, you may define a specific CMCTR-based application to use with your configuration file, i.e. an application which uses the CMCTR API to extend the functionality of the Communicator. Either type in the applications's full path directly 10 or use the file selector dialog box which pops up when you click the `Select CMCTR app` button. To learn more about the usage of the CMCTR API see the CMCTR and NMVAGO sections in the POOL Standard Modules documentation.

Altogether, there are 4 logical buses which you can assign to different (but also the same) AIDA driver stacks.

Note: All absolute paths (for **.aida-cfg* 1, **.dbc* 2, **.pi* 10) will be replaced with relative paths as soon as the configuration is saved to disk (relative to the save location of the configuration file).

After you've closed the configuration dialog window by clicking on the `OK` button 11, in the main window the list of the automatically generated panels will be displayed for the currently selected bus (use the `Panels` menu to show the lists for a certain bus number).

2.2.3 TxNodes Panel



TxNodes Panel

TxNodes for Bus 0:
- ☑ Gateway
- ☐ Gateway_MMICAN
- ☐ Graphiccontroller
- ☐ Graphiccontroller_MMICAN
- ☐ Grundcontroller
- ☐ Immobilizer
- ☐ Vector__XXX

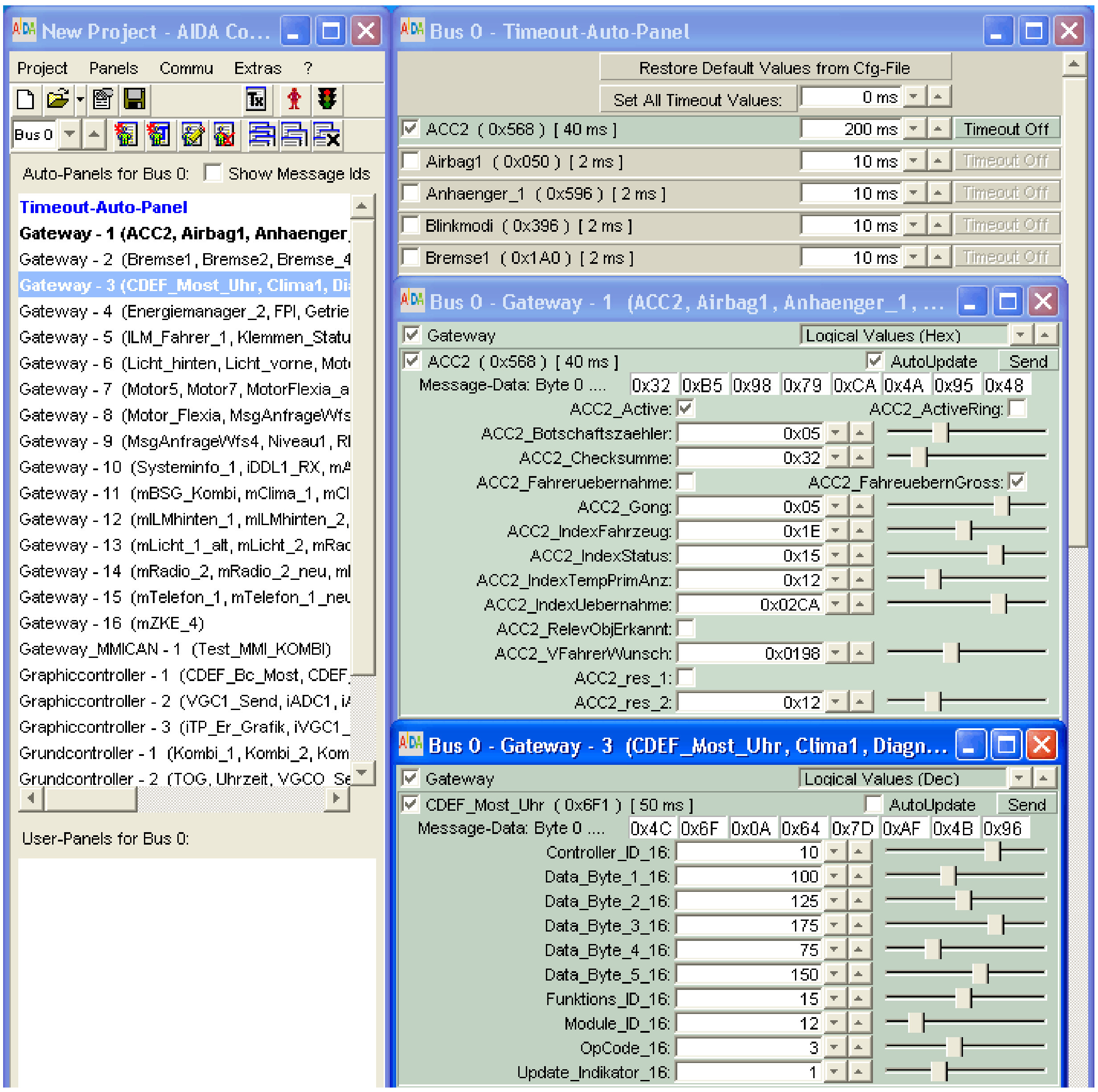
TxNodes for Bus 1:

TxNodes for Bus 2:

Picture 10: the TxNodes Panel.

The TxNodes panel lists all nodes defined in the DBC database for all 4 buses. The nodes with assigned tx messages are displayed with a check box, the nodes which only have rx messages assigned are displayed grayed. With the check box, all controls for messages and signals of the corresponding tx node are activated/deactivated. That way, the cyclic transmission of the corresponding CAN messages is activated/deactivated (if the check box in front of the message name within an auto panel, a standard user panel or a timeout panel is selected). The node is put on the bus resp. taken from it (and this way switched to receiving mode).

2.2.4 Auto Panels



New Project - AIDA Co...

Project Panels Commu Extras ?

Bus 0

Auto-Panels for Bus 0: ☐ Show Message Ids

Timeout-Auto-Panel
Gateway - 1 (ACC2, Airbag1, Anhaenger
Gateway - 2 (Bremse1, Bremse2, Bremse_4
Gateway - 3 (CDEF_Most_Uhr, Clima1, Di
Gateway - 4 (Energiemanager_2, FPI, Getrie
Gateway - 5 (ILM_Fahrer_1, Klemmen_Statu
Gateway - 6 (Licht_hinten, Licht_vorne, Mot
Gateway - 7 (Motor5, Motor7, MotorFlexia_a
Gateway - 8 (Motor_Flexia, MsgAnfrageWfs
Gateway - 9 (MsgAnfrageWfs4, Niveau1, Rl
Gateway - 10 (Systeminfo_1, iDDL1_RX, mA
Gateway - 11 (mBSG_Kombi, mClima_1, mCl
Gateway - 12 (mILMhinten_1, mILMhinten_2,
Gateway - 13 (mLicht_1_alt, mLicht_2, mRad
Gateway - 14 (mRadio_2, mRadio_2_neu, ml
Gateway - 15 (mTelefon_1, mTelefon_1_neu
Gateway - 16 (mZKE_4)
Gateway_MMICAN - 1 (Test_MMI_KOMBI)
Graphiccontroller - 1 (CDEF_Bc_Most, CDEF
Graphiccontroller - 2 (VGC1_Send, iADC1, iA
Graphiccontroller - 3 (iTP_Er_Grafik, iVGC1_
Grundcontroller - 1 (Kombi_1, Kombi_2, Kom
Grundcontroller - 2 (TOG, Uhrzeit, VGCO_Se

User-Panels for Bus 0:

Bus 0 - Timeout-Auto-Panel

Restore Default Values from Cfg-File

Set All Timeout Values: 0 ms

☑	ACC2 (0x568) [40 ms]	200 ms	Timeout Off
☐	Airbag1 (0x050) [2 ms]	10 ms	Timeout Off
☐	Anhaenger_1 (0x596) [2 ms]	10 ms	Timeout Off
☐	Blinkmodi (0x396) [2 ms]	10 ms	Timeout Off
☐	Bremse1 (0x1A0) [2 ms]	10 ms	Timeout Off

Bus 0 - Gateway - 1 (ACC2, Airbag1, Anhaenger_1, ...

☑ Gateway Logical Values (Hex)

☑ ACC2 (0x568) [40 ms] ☑ AutoUpdate Send

Message-Data: Byte 0 .... 0x32 0xB5 0x98 0x79 0xCA 0x4A 0x95 0x48

ACC2_Active:	☑
ACC2_ActiveRing:	☐
ACC2_Botschaftszaehler:	0x05
ACC2_Checksumme:	0x32
ACC2_Fahreruebernahme:	☐
ACC2_FahreuebernGross:	☑
ACC2_Gong:	0x05
ACC2_IndexFahrzeug:	0x1E
ACC2_IndexStatus:	0x15
ACC2_IndexTempPrimAnz:	0x12
ACC2_IndexUebernahme:	0x02CA
ACC2_RelevObjErkannt:	☐
ACC2_VFahrerWunsch:	0x0198
ACC2_res_1:	☐
ACC2_res_2:	0x12

Bus 0 - Gateway - 3 (CDEF_Most_Uhr, Clima1, Diagn...

☑ Gateway Logical Values (Dec)

☑ CDEF_Most_Uhr (0x6F1) [50 ms] ☐ AutoUpdate Send

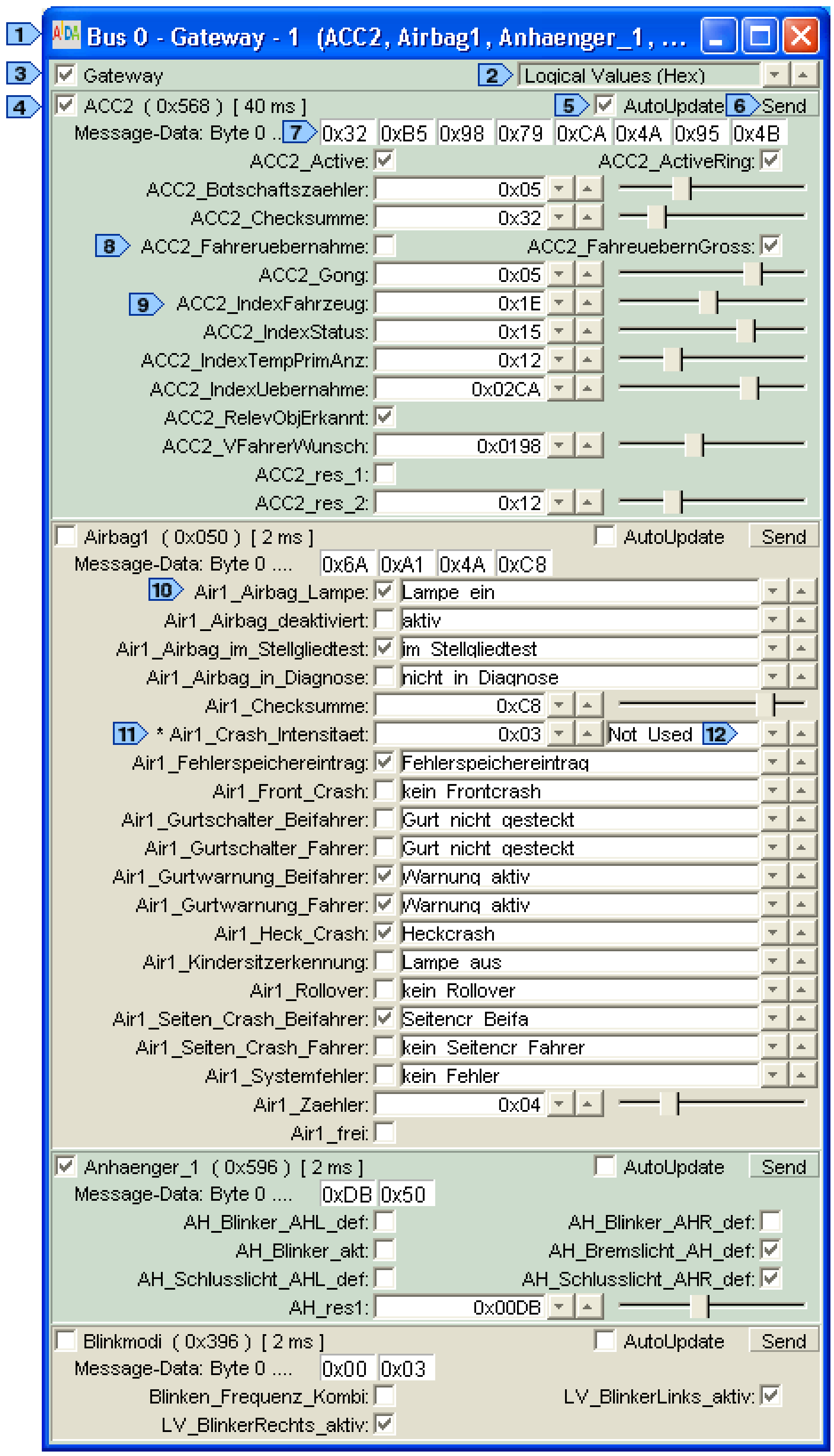
Message-Data: Byte 0 .... 0x4C 0x6F 0x0A 0x64 0x7D 0xAF 0x4B 0x96

Controller_ID_16:	10
Data_Byte_1_16:	100
Data_Byte_2_16:	125
Data_Byte_3_16:	175
Data_Byte_4_16:	75
Data_Byte_5_16:	150
Funktions_ID_16:	15
Module_ID_16:	12
OpCode_16:	3
Update_Indikator_16:	1

Picture 11: the main window showing the list of the automatically generated panels.

In the upper half of the main window you see the list of the automatically generated panels for the currently selected bus control. A click on a list entry opens the corresponding auto panel.

Note: If you select a panel for the first time, it may take about 1 to 3 seconds until the panel finally appears, depending on the number of controls (resp. messages and signals) in the panel. (On older systems, e.g. Pentium II with 400 MHz and 128 MB RAM or less, it may even take up to 20 or 30 seconds.)



1 Bus 0 - Gateway - 1 (ACC2, Airbag1, Anhaenger_1, ...

3 [✓] Gateway 2 Logical Values (Hex)

4 [✓] ACC2 (0x568) [40 ms] 5 [✓] AutoUpdate 6 Send

Message-Data: Byte 0 .. 7 0x32 0xB5 0x98 0x79 0xCA 0x4A 0x95 0x4B

ACC2_Active:	[✓]	ACC2_ActiveRing:	[✓]
ACC2_Botschaftszaehler:	0x05		
ACC2_Checksumme:	0x32		
8 ACC2_Fahreruebernahme:	[ ]	ACC2_FahreuebernGross:	[✓]
ACC2_Gong:	0x05		
9 ACC2_IndexFahrzeug:	0x1E		
ACC2_IndexStatus:	0x15		
ACC2_IndexTempPrimAnz:	0x12		
ACC2_IndexUebernahme:	0x02CA		
ACC2_RelevObjErkannt:	[✓]		
ACC2_VFahrerWunsch:	0x0198		
ACC2_res_1:	[ ]		
ACC2_res_2:	0x12		

[ ] Airbag1 (0x050) [2 ms] [ ] AutoUpdate Send

Message-Data: Byte 0 .... 0x6A 0xA1 0x4A 0xC8

10 Air1_Airbag_Lampe:	[✓]	Lampe ein
Air1_Airbag_deaktiviert:	[ ]	aktiv
Air1_Airbag_im_Stellgliedtest:	[✓]	im Stellgliedtest
Air1_Airbag_in_Diagnose:	[ ]	nicht in Diagnose
Air1_Checksumme:	0xC8	
11 * Air1_Crash_Intensitaet:	0x03	Not Used 12
Air1_Fehlerspeichereintrag:	[✓]	Fehlerspeichereintrag
Air1_Front_Crash:	[ ]	kein Frontcrash
Air1_Gurtschalter_Beifahrer:	[ ]	Gurt nicht gesteckt
Air1_Gurtschalter_Fahrer:	[ ]	Gurt nicht gesteckt
Air1_Gurtwarnung_Beifahrer:	[✓]	Warnung aktiv
Air1_Gurtwarnung_Fahrer:	[✓]	Warnung aktiv
Air1_Heck_Crash:	[✓]	Heckcrash
Air1_Kindersitzerkennung:	[ ]	Lampe aus
Air1_Rollover:	[ ]	kein Rollover
Air1_Seiten_Crash_Beifahrer:	[✓]	Seitencr Beifa
Air1_Seiten_Crash_Fahrer:	[ ]	kein Seitencr Fahrer
Air1_Systemfehler:	[ ]	kein Fehler
Air1_Zaehler:	0x04	
Air1_frei:	[ ]	

[✓] Anhaenger_1 (0x596) [2 ms] [ ] AutoUpdate Send

Message-Data: Byte 0 .... 0xDB 0x50

AH_Blinker_AHL_def:	[ ]	AH_Blinker_AHR_def:	[ ]
AH_Blinker_akt:	[ ]	AH_Bremslicht_AH_def:	[✓]
AH_Schlusslicht_AHL_def:	[ ]	AH_Schlusslicht_AHR_def:	[✓]
AH_res1:	0x00DB		

[ ] Blinkmodi (0x396) [2 ms] [ ] AutoUpdate Send

Message-Data: Byte 0 .... 0x00 0x03

Blinken_Frequenz_Kombi:	[ ]	LV_BlinkerLinks_aktiv:	[✓]
LV_BlinkerRechts_aktiv:	[✓]		

Picture 12: an auto panel

The title bar of each auto panel 1▷ lists corresponding bus number, tx node and panel number, as well as all messages it contains (either their names or their ids, depending on the bus specific setting of check box `Show Message Ids` within the main window).

With the list spinner 2▷ in the top right corner you can select the display mode for signal values within a panel: logical values in decimal or hexadecimal representation or physical values with units.

The top control within each auto panel is a check box with the name of the corresponding tx node 3▷. With this, all controls for messages and signals of the corresponding tx node are activated/deactivated. That way, the cyclic transmission of the corresponding CAN messages is activated/deactivated (if the check box in front of the message name 4▷ is selected). The node is put on the bus resp. taken from it (and this way switched to receiving mode).

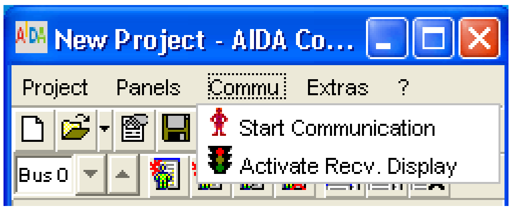
To activate a message for cyclic transmission, select the check box in front of the message name 4▷ while the corresponding tx node's check box 3▷ is also selected.

To select a message for reception, select the check box in front of the message name 4▷ while the corresponding tx node's check box 3▷ is deselected.

Altogether, there are 5 different signal types: Bool1 8▷, Bool1 with value description list 10▷, Integer 9▷, Integer with value description list 11▷ and Real.

Note: Integer signals with value description list 11▷ are marked with an asterisk before the signal name. With CTRL + double click on the list spinner 12▷ resp. value slider you can toggle between these two controls.

Note: Changes of the settings for message and signal controls, a window's position and visibility etc. are not saved automatically. Only if you change the basic settings (i.e. the assigned files or the arrangement of the user panels), a dialog box will remind you to save the configuration before changing or closing a project. Use menu item `Project - Save Configuration` to save your configuration.



Picture 13: the „Runner"menu entry

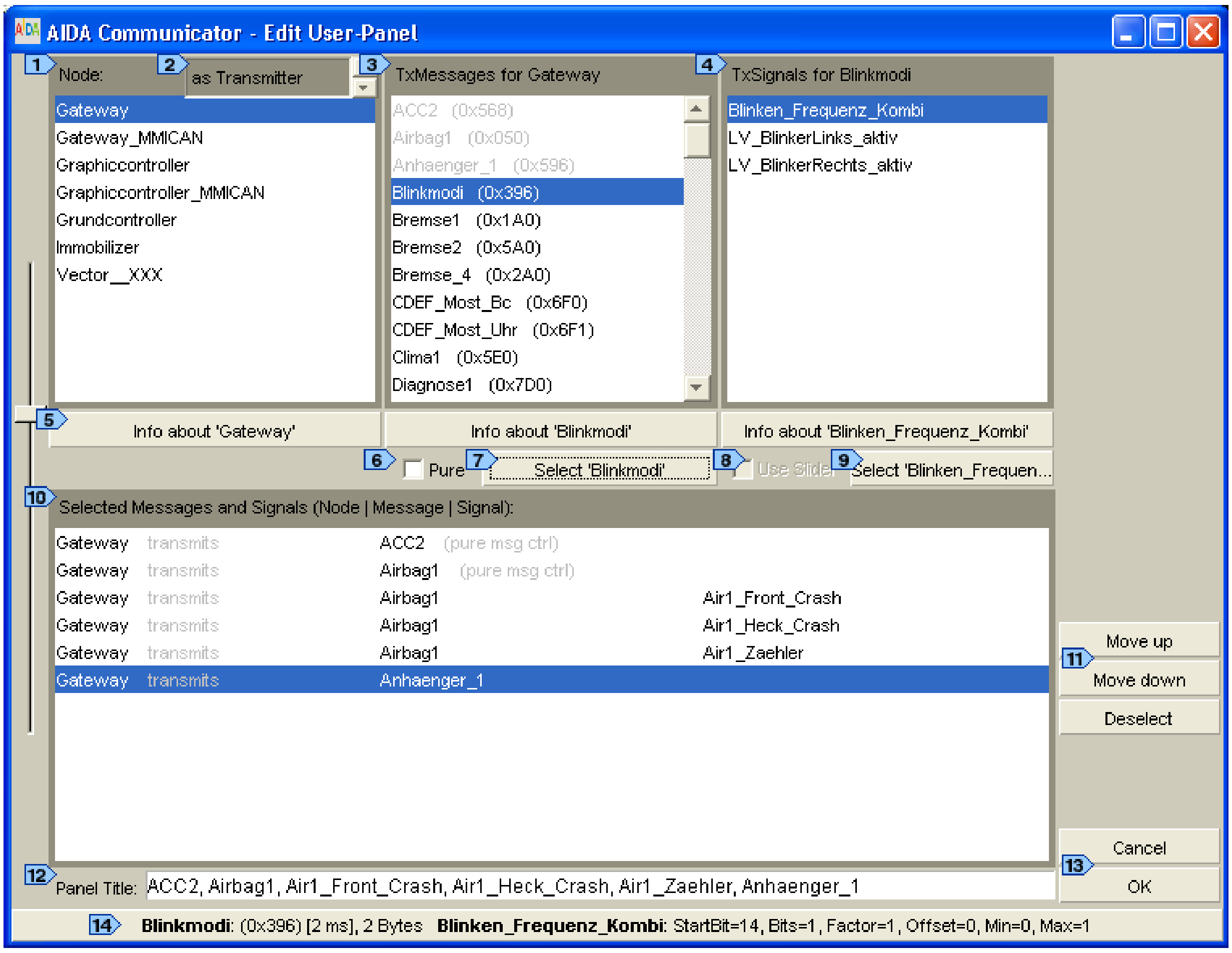
After the different assigned configuration files have been opened successfully, you can alternately start and stop the communication by selecting menu item `Commu - Start/Stop Communication` or clicking the "Runner" tool button ( ).

2.2.5 Standard User Panels

In the lower half of the main window you see the list of the user-defined panels for the currently selected bus control. A click on a list entry opens the corresponding user panel.

To define a new user panel for the currently selected bus, select menu item `Panels – New Standard User-Panel ...` or click the corresponding tool button ( ). The user panel configuration dialog pops up.

By clicking on the `Select "*"` buttons 7 9 you can add messages or signals to your user panel.


AIDA Communicator - Edit User-Panel

1 Node: 2 as Transmitter | 3 TxMessages for Gateway | 4 TxSignals for Blinkmodi

Gateway
Gateway_MMICAN
Graphiccontroller
Graphiccontroller_MMICAN
Grundcontroller
Immobilizer
Vector__XXX

ACC2 (0x568)
Airbag1 (0x050)
Anhaenger_1 (0x596)
Blinkmodi (0x396)
Bremse1 (0x1A0)
Bremse2 (0x5A0)
Bremse_4 (0x2A0)
CDEF_Most_Bc (0x6F0)
CDEF_Most_Uhr (0x6F1)
Clima1 (0x5E0)
Diagnose1 (0x7D0)

Blinken_Frequenz_Kombi
LV_BlinkerLinks_aktiv
LV_BlinkerRechts_aktiv

5 Info about 'Gateway' | Info about 'Blinkmodi' | Info about 'Blinken_Frequenz_Kombi'

6 Pure 7 Select 'Blinkmodi' 8 Use Slider 9 Select 'Blinken_Frequen...

10 Selected Messages and Signals (Node | Message | Signal):

```
Gateway  transmits  ACC2  (pure msg ctrl)
Gateway  transmits  Airbag1  (pure msg ctrl)
Gateway  transmits  Airbag1       Air1_Front_Crash
Gateway  transmits  Airbag1       Air1_Heck_Crash
Gateway  transmits  Airbag1       Air1_Zaehler
Gateway  transmits  Anhaenger_1
```

Move up
11 Move down
Deselect
Cancel
13 OK

12 Panel Title: ACC2, Airbag1, Air1_Front_Crash, Air1_Heck_Crash, Air1_Zaehler, Anhaenger_1

14 **Blinkmodi**: (0x396) [2 ms], 2 Bytes **Blinken_Frequenz_Kombi**: StartBit=14, Bits=1, Factor=1, Offset=0, Min=0, Max=1

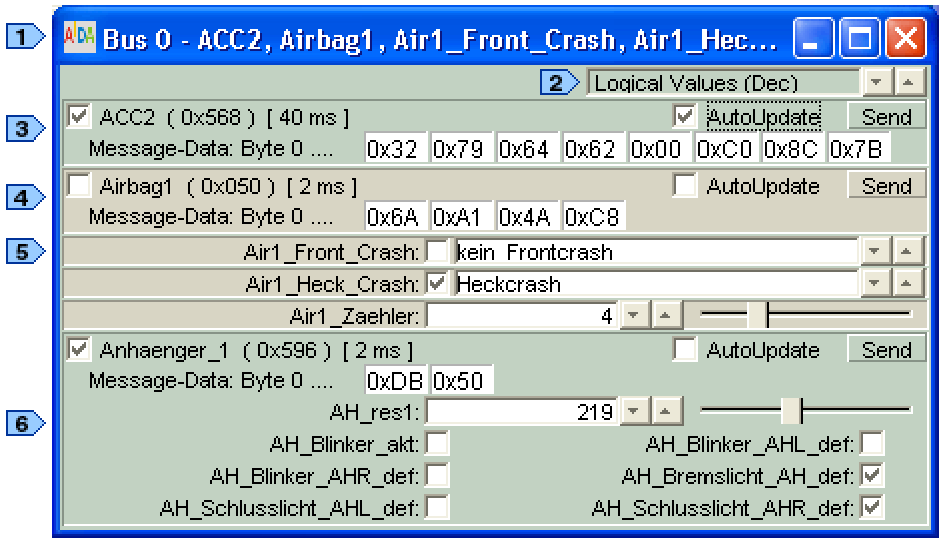
Picture 14: the user panel configuration dialog window

Check the `Pure` check box 6 to select pure message controls, i.e. message controls without their corresponding embedded signal controls. This way you can build very compact user panels which hold only the signal controls you are interested in and at the same time make available all controls necessary to configure the corresponding message, like the `AutoUpdate` check box, the check box for cyclic transmission resp. reception and the `Send` button.

The `Use slider` check box 8 is only enabled for signals with an assigned value description list and a bit width above 1 bit. For such signals as default a list spinner listing the value descriptions is used as second input control within a signal control. If you want to use a slider instead, check the `Use slider` check box before adding the signal to your user panel. Note: Within a panel you can toggle between slider and spinner with CTRL + double click on these controls.

In the text field at the bottom 12 you may set a name for your user panel. With this name the user panel is listed in the list field within the main window.

After clicking the `OK` button 13 for the selections in Picture 14, the following user panel is generated:



Bus 0 - ACC2, Airbag1, Air1_Front_Crash, Air1_Hec...

Logical Values (Dec)

[✓] ACC2 (0x568) [40 ms]   [✓] AutoUpdate   Send
Message-Data: Byte 0 .... 0x32 0x79 0x64 0x62 0x00 0xC0 0x8C 0x7B

[ ] Airbag1 (0x050) [2 ms]   [ ] AutoUpdate   Send
Message-Data: Byte 0 .... 0x6A 0xA1 0x4A 0xC8

Air1_Front_Crash: [ ] kein Frontcrash
Air1_Heck_Crash: [✓] Heckcrash
Air1_Zaehler: 4

[✓] Anhaenger_1 (0x596) [2 ms]   [ ] AutoUpdate   Send
Message-Data: Byte 0 .... 0xDB 0x50
AH_res1: 219
AH_Blinker_akt: [ ]   AH_Blinker_AHL_def: [ ]
AH_Blinker_AHR_def: [ ]   AH_Bremslicht_AH_def: [✓]
AH_Schlusslicht_AHL_def: [ ]   AH_Schlusslicht_AHR_def: [✓]

Picture 15: the user panel generated from the selections made in picture 14

As you can see by comparison of Picture 15 with Picture 12, there are three fundamental differences between user panels and auto panels:

1. A user panel never has a node check box (in the upper left corner), because the selected messages and signals may belong to different nodes.

2. A user panel may have pure message controls 3 4, i.e. message controls without their corresponding embedded signal controls. (For comparison, there's also a message control with its embedded signal controls 6.)

3. A user panel may have single signal controls 5, i.e. signal controls which are not embedded into their corresponding message control.

The title bar displays the bus number followed by the user panel's title 1▷. In the same way as for auto panels you may also select between the three different representation modes `Logical Values (Decimal)`, `Logical Values (Hexadecimal)` or `Physical Values` 2▷.

Note: All settings for a message and its signals are synchronized for all corresponding controls among auto panels and user panels.

2.2.6 Timeout Panels

For testing purposes, the cyclic transmission of messages may be suspended for a defined period of time (timeout). **Note:** The timeout value must be greater than the cycle time.

The following time response (Zeitverhalten) is implemented:

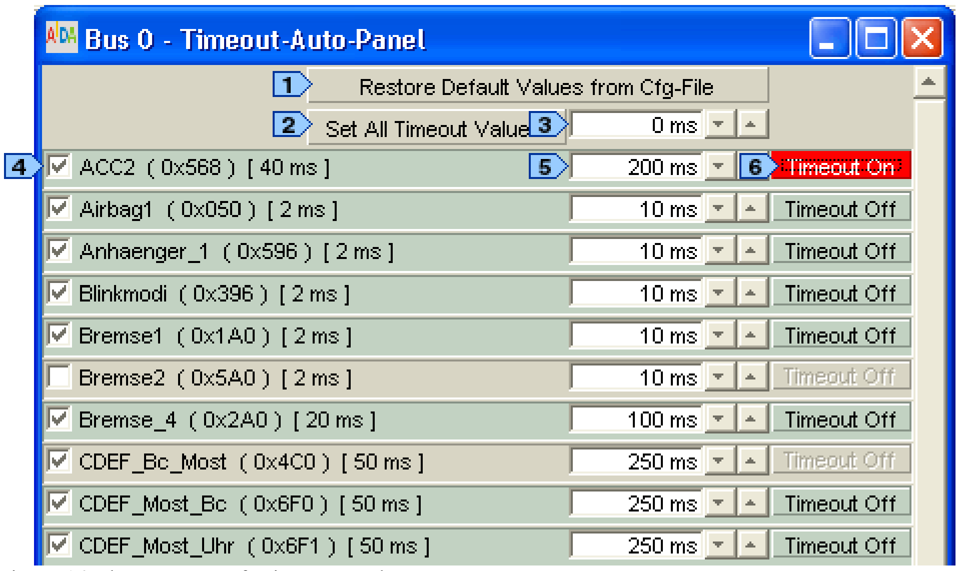
```
Timebase [ms]
0         50        100       150       200       250       300
|---------|---------|---------|---------|---------|---------|------

1▷Example 1: message with f=20Hz, timeout = 150ms
              150
x---------x----T----x---------o---------o---------x---------x------
                    [------------150ms------------]

2▷Example 2: message with f=20Hz, timeout = 120ms
              120
x---------x----T----x---------o---------o---x---------x---------x--
                    [---------120ms---------]

T = trigger timeout
x = send message
o = suppress
```

In the first example 1▷, following a timeout of 150ms, the message is transmitted within its original cycle; in the second example 2▷, following a timeout of 120ms, the message is transmitted with a shift towards its original cycle.

The following functionality is implemented:

1. After the timeout has been triggered, the message is send once again within the original time pattern (Zeitraster). Then the defined timeout starts.

2. After the timeout has expired, the message is resend. The first transmission follows immediately, the rest according to the defined cycle time.
3. Per bus, one timeout-panel is generated automatically. It contains all messages with cycle time values greater than 0ms. The timeout-panel is listed as the first entry in the auto-panels list field within the Communicator main window. Note: In the auto-panels list field as well as the user-panels list field, entries for timeout-panels are always displayed in a blue font color to differentiate them from standard auto-panels and user-panels.



Bus 0 - Timeout-Auto-Panel
1
Restore Default Values from Cfg-File
2
Set All Timeout Value
3
4
5
6

	Set All Timeout Value	0 ms
✓ ACC2 (0x568) [40 ms]	200 ms	Timeout On
✓ Airbag1 (0x050) [2 ms]	10 ms	Timeout Off
✓ Anhaenger_1 (0x596) [2 ms]	10 ms	Timeout Off
✓ Blinkmodi (0x396) [2 ms]	10 ms	Timeout Off
✓ Bremse1 (0x1A0) [2 ms]	10 ms	Timeout Off
Bremse2 (0x5A0) [2 ms]	10 ms	Timeout Off
✓ Bremse_4 (0x2A0) [20 ms]	100 ms	Timeout Off
✓ CDEF_Bc_Most (0x4C0) [50 ms]	250 ms	Timeout Off
✓ CDEF_Most_Bc (0x6F0) [50 ms]	250 ms	Timeout Off
✓ CDEF_Most_Uhr (0x6F1) [50 ms]	250 ms	Timeout Off

Picture 16: the upper part of a timeout panel

4. Each message is represented with its 4▷`name, ( ID ), [ cycle time ],` 5▷an input field for the timeout value [ms], 4▷a check box for cyclic transmission (resp. reception, depending on the state of its corresponding tx node) and 6▷a toggle button to trigger the timeout.
5. When a timeout-panel is generated for the very first time, all timeout values are set to the fivefold of the cycle time values.
6. Additionally, each timeout-panel possesses 3▷an input field and 2▷an accompanying push button, which allow to set the values of all timeout fields panel-globally. The user-defined value is committed with the push button.
7. After activation of 6▷the timeout button, it remains in its active/activated state until the timeout is expired. Then the timeout button is reset to its inactive/deactivated state.
8. When cyclic transmission of a message is deactivated, obviously the timeout functionality is deactivated too. 6▷The timeout button is reset to its inactive/deactivated state.
9. Likewise, if 6▷the timeout button is reset manually, the timeout functionality is deactivated too.
10. While cyclic transmission of a message is switched off, 6▷its timeout button is disabled.

11. When a configuration is saved (`Save Configuration` / `Save Configuration As ...`), the current values of 5 the timeout fields are stored.
12. When a configuration is loaded, the values of 5 the timeout fields are restored from the saved values.
13. With 1 the `Restore Default Values from Cfg-File` push button all timeout fields of a selected timeout panel can be reset to the values obtained from the configuration file.
14. In addition, timeout panels may also be user-defined (`New Timeout User-Panel`). The configuration dialog is similar to the one used for standard user panels (see section 2.2.5), with the difference that only messages can be selected.
15. A timeout correction value ensures, that the actual interval between the last message sent after the timeout was triggered and the first message after expiration of the timeout is exactly the defined timeout value. It is used globally for all timeout values. The timeout correction value is determined with a CAN-Trace-Tool and depends on the performance of the host PC. It is stored as entry *TimeoutCorrection_ms* in the *aida-cmctr.ini* file (see section 4.2) and can be changed via `Extras - Global Settings`.

2.2.7 VolCANo Extensions

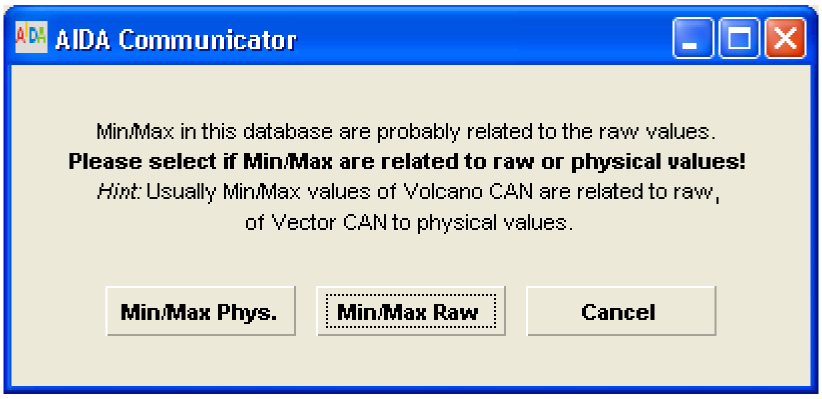
2.2.7.1 Min/Max Values

The treatment of minimum and maximum values is different for Vector CAN and Volcano CAN databases. Because both use the same file name extension, an automatic recognition can not be guaranteed.

Therefore, if a configuration is created or edited, the first DBC file selected for addition to the bus is searched for an occurrence of the term *"Volcano"* within the comment header. If the term is found, a dialog window pops up containing the phrase: *"Min/Max in this database are probably related to the raw values!"*.

In addition, independent from an occurrence of the term *"Volcano"*, for each DBC file the following phrase is shown: *"Please select if Min/Max are related to raw or physical values. Hint: Usually Min/Max values of Volcano CAN are related to raw, of Vector CAN to physical values."*

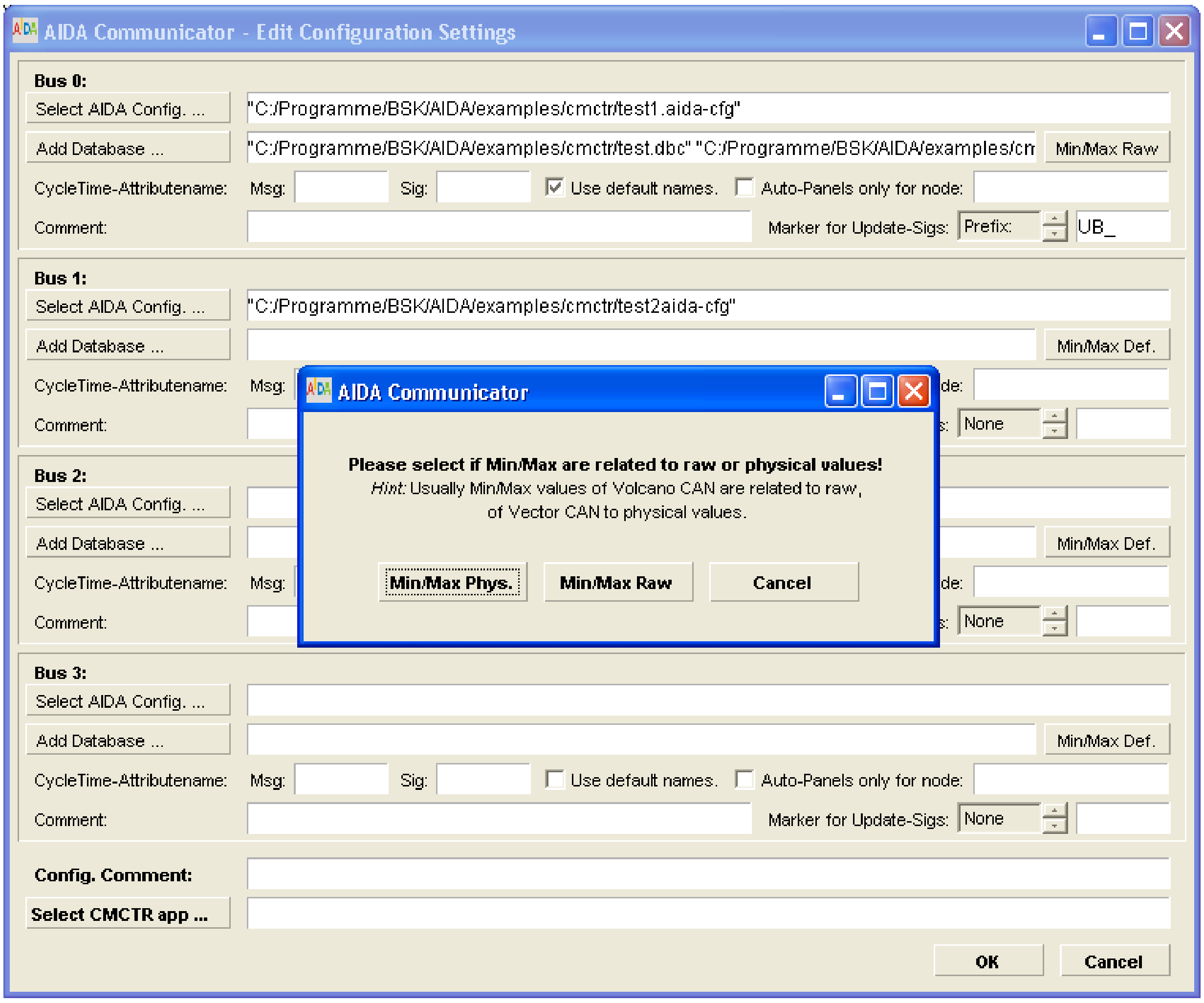
Next follow three buttons labeled [*Min/Max raw*], [*Min/Max phys.*] and [*Cancel*].



AIDA Communicator

Min/Max in this database are probably related to the raw values.
Please select if Min/Max are related to raw or physical values!
Hint: Usually Min/Max values of Volcano CAN are related to raw,
of Vector CAN to physical values.

Picture 17: the dialog box



AIDA Communicator - Edit Configuration Settings

Bus 0:
Select AIDA Config. ... "C:/Programme/BSK/AIDA/examples/cmctr/test1.aida-cfg"
Add Database ... "C:/Programme/BSK/AIDA/examples/cmctr/test.dbc" "C:/Programme/BSK/AIDA/examples/cm Min/Max Raw
CycleTime-Attributename: Msg: Sig: ☑ Use default names. ☐ Auto-Panels only for node:
Comment: Marker for Update-Sigs: Prefix: UB_

Bus 1:
Select AIDA Config. ... "C:/Programme/BSK/AIDA/examples/cmctr/test2aida-cfg"
Add Database ... Min/Max Def.
CycleTime-Attributename: Msg:
Comment: None

Bus 2:
Select AIDA Config. ...
Add Database ... Min/Max Def.
CycleTime-Attributename: Msg:
Comment: None

Bus 3:
Select AIDA Config. ...
Add Database ... Min/Max Def.
CycleTime-Attributename: Msg: Sig: ☐ Use default names. ☐ Auto-Panels only for node:
Comment: Marker for Update-Sigs: None

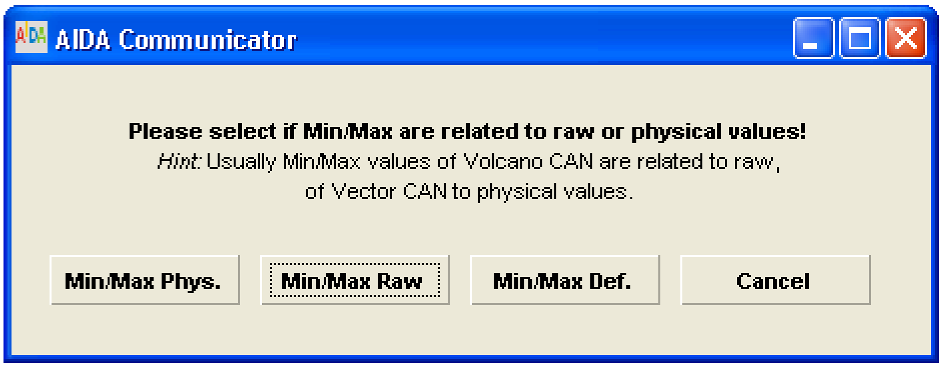
Config. Comment:
Select CMCTR app ...

AIDA Communicator

Please select if Min/Max are related to raw or physical values!
Hint: Usually Min/Max values of Volcano CAN are related to raw,
of Vector CAN to physical values.

Picture 18: the main configuration dialog with the dialog box

After a selection has been made by the user (in the case of *Cancel* the file is not opened) the selected type ([*Min/Max raw*] / [*Min/Max phys.*]) is displayed next to the file names with a push button. If this button is applied, the previously described selection dialog is shown again, but with an additional button ([*Min/Max def.*]), which for now has the same effect as ([*Min/Max phys.*]) – this may change in future versions.


AIDA Communicator

Please select if Min/Max are related to raw or physical values!
Hint: Usually Min/Max values of Volcano CAN are related to raw,
of Vector CAN to physical values.

Picture 19: the dialog box

2.2.7.2 Update-Bit Signals

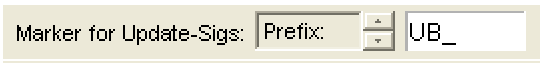
Volcano DBC files define special signals, the so called update-bit signals, which belong to one or more other base signals of the same message.

This coupling is defined in the DBC file only through the signal names:

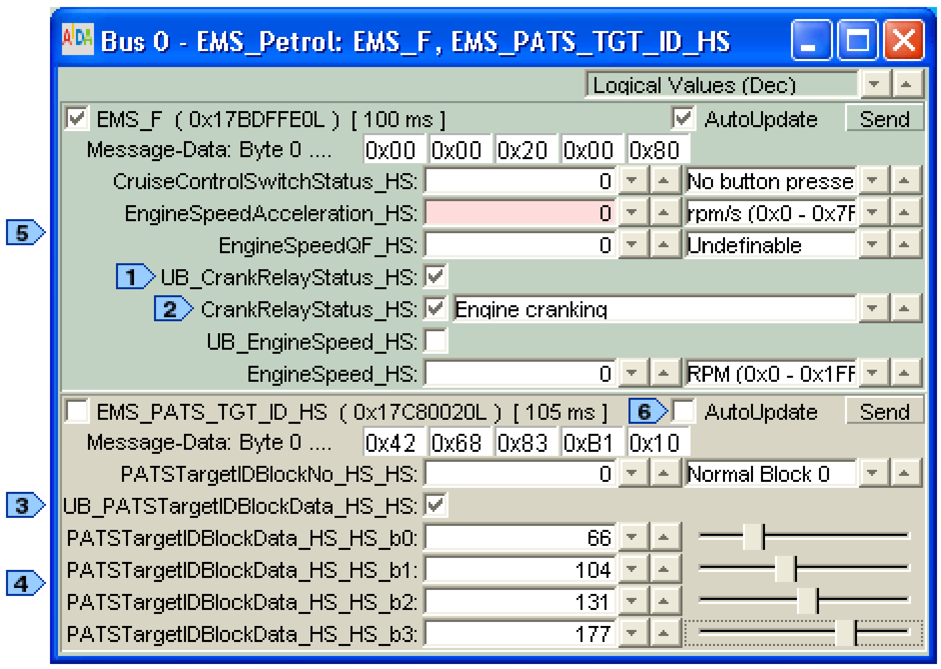
1.) Base signals and their corresponding update-bit signal have the same base name, the name of the update-bit signal is marked through a special prefix or suffix (in most cases through the prefix *"UB_"*).

2.) Additionally, the base name of the base signals may have a trailing *"_bN"*, where N stands for a number between 0 and 7. This means that one update-bit signal may belong to several base signals.

In the main configuration dialog of Communicator, you may define for each bus such a marker string, and select between prefix or suffix. The settings are stored in the configuration file.


Marker for Update-Sigs: Prefix: UB_

Picture 20: part of the main configuration dialog



Bus 0 - EMS_Petrol: EMS_F, EMS_PATS_TGT_ID_HS

Logical Values (Dec)

✓ EMS_F (0x17BDFFE0L) [100 ms]		✓ AutoUpdate	Send
Message-Data: Byte 0	0x00 0x00 0x20 0x00 0x80		
CruiseControlSwitchStatus_HS:	0	No button presse	
EngineSpeedAcceleration_HS:	0	rpm/s (0x0 - 0x7F	
EngineSpeedQF_HS:	0	Undefinable	
1 UB_CrankRelayStatus_HS:	✓		
2 CrankRelayStatus_HS:	✓	Engine cranking	
UB_EngineSpeed_HS:	☐		
EngineSpeed_HS:	0	RPM (0x0 - 0x1FF	
☐ EMS_PATS_TGT_ID_HS (0x17C80020L) [105 ms]		6 ☐ AutoUpdate	Send
Message-Data: Byte 0	0x42 0x68 0x83 0xB1 0x10		
PATSTargetIDBlockNo_HS_HS:	0	Normal Block 0	
UB_PATSTargetIDBlockData_HS_HS:	✓		
PATSTargetIDBlockData_HS_HS_b0:	66		
PATSTargetIDBlockData_HS_HS_b1:	104		
PATSTargetIDBlockData_HS_HS_b2:	131		
PATSTargetIDBlockData_HS_HS_b3:	177		

Picture 21: a user panel with two messages which contain Volcano update-bit signals

Within the panels, the effects are the following:

1.) Update-bit signal 1 3 and the base signal(s) 2 4 it belongs to are arranged one below the other within a message control 5.

2.) Modification of a base signal's value 2 4 causes the accompanying update-bit signal 1 3 automatically to be set. Note: When modifying base signals with trailing *"_bN"* 4, to ensure consistency of data for the Volcano compound signal, it is recommended to deselect the `AutoUpdate` check box 6 until all changes are done. This way you can first change all related *"_bN"* -signals before the changed message data is actually send.

3.) After the message 5 has been sent, all its update-bit signals 1 are reset automatically.

2.2.8 Communicator with Network Management

2.2.8.1 NMVAGO

Module NMVAGO uses the Communicator module CMCTR as library to add network management features as specified in the OSEK-NM-Specification, version 2.5.2. It supports the VAG Network Management only. At some time in the future, module NMOSEK (2.2.8.2) will support various types of network management, including VAG network management. NMVAGO will be obsolete then.

If VAG Network Management is desired then module NMVAGO can simply be used in replacement for module CMCTR. To start NMVAGO see 4.1.2 and 4.1.5.

If NMVAGO is started 'stand-alone' (4.1.3) then network management is activated when the communication is started. The network management state of each node will be derived from the Communicator's cyclic Tx messages for that node. If the Communicator is transmitting at least one message cyclically then the corresponding node will be 'awake', otherwise it will try to go to 'sleep' mode.

When NMVAGO is loaded as library then the network management has to be activated explicitly by the application (see 4.1.6 and `$(AIDAHOME)examples/nmtest1.pool` and `$(AIDAHOME)examples/nmtest2.pool`).

See NMVAGO.PLI for further details (the file is part of the POOL Standard Modules documentation).

2.2.8.2 NMOSEK

Module NMOSEK uses the Communicator module CMCTR as library to add network management features as specified in the OSEK-NM-Specification, version 2.5.2. Currently, only indirect network management is supported.

If network management is desired then module NMOSEK can simply be used in replacement for module CMCTR. To start NMOSEK see 4.1.3 and 4.1.6.

If NMOSEK is started 'stand-alone' (4.1.3) then network management is activated when the communication is started. Depending on the 'Automatic state control' checkbox in the 'Configuration Panel' (see below) the network management can be activated in two modes (***activation mode***):

Automatic mode:

The network management state of each node will be derived from the Communicator's cyclic Tx messages for that node. If the Communicator is transmitting at least one message cyclically then the corresponding node will be 'awake', otherwise it will try to go to 'sleep' mode.

Awake mode:

The network management state of each node will be 'awake'.

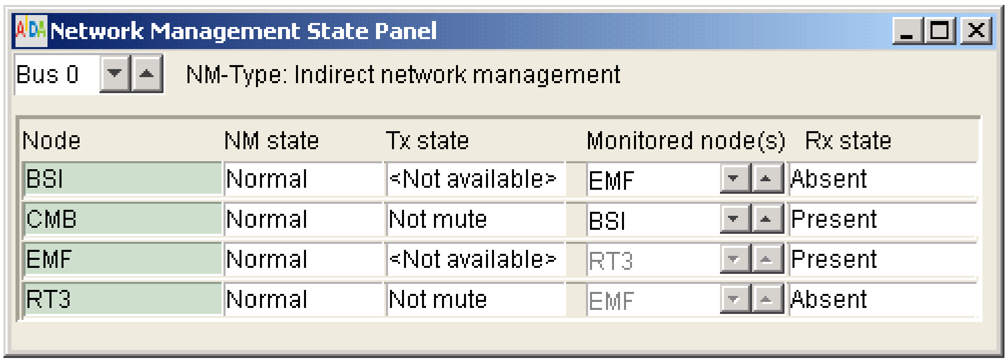
The difference will be visible in the NM state (see State Panel below), but has no further influence on the bus activity.

When NMOSEK is loaded as library then the network management has to be activated explicitly by the application (see 4.1.6 and `$(AIDAHOME)examples/nmosek_tstapp1.pool`).

See NMOSEK.PLI for further details (the file is part of the POOL Standard Modules documentation).

NMOSEK has three additional windows, the 'State Panel, the 'Configuration Panel' and the 'Configuration Log Window'. The position, size and visibility of these windows will be stored and restored from the Communicator configuration file (**.cmctr-cfg*). By default (when a new configuration is generated), the State Panel and the Configuration Panel are visible, the Configuration Log Window is invisible. All windows can be opened via the Communicator Menu `Extensions` → `NM-Panels`.

State Panel:


Network Management State Panel
Bus 0
NM-Type: Indirect network management

Node	NM state	Tx state	Monitored node(s)	Rx state
BSI	Normal	<Not available>	EMF	Absent
CMB	Normal	Not mute	BSI	Present
EMF	Normal	<Not available>	RT3	Present
RT3	Normal	Not mute	EMF	Absent

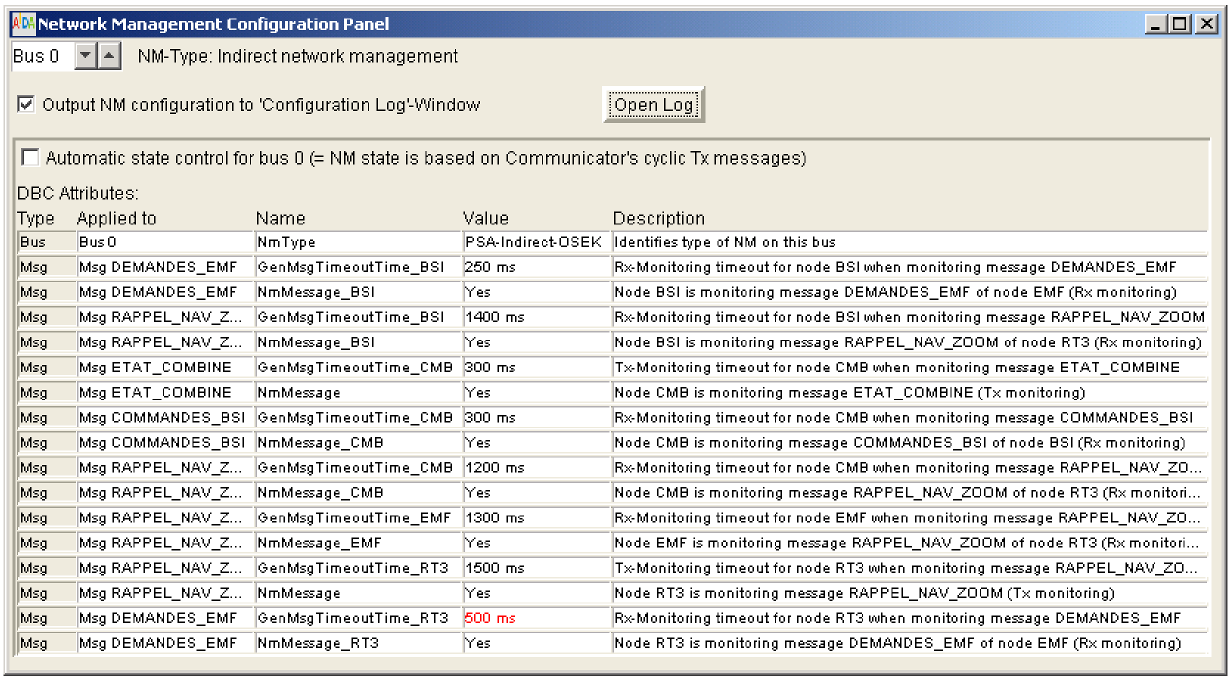
Picture 22: the network management state panel

The state panel displays the relevant network management information for each bus. The bus can be chosen by the bus control:

For every monitoring node (whether it is monitoring its own transmission or the reception from other nodes) there exists a row. The background color of a node is green, if the node is activated for transmitting. A node can be activated via the Communicator auto panels (2.2.4) or the TxNodes panel (2.2.3). The other fields in the row show the following network management information for that node (if it is activated):

- NM state: reflects the network management state according to the OSEK-NM-Specification
- Tx state: Tx-monitoring state; state of the node's transmission branch (if it monitoring its own state)
- Rx state: Rx-monitoring state for each monitored node. If the monitoring node has more than 1 monitored node then the monitored node can be chosen by the spinner control. The Rx-state always displays the state for the monitored node chosen by the spinner-control.

Configuration Panel:



Bus 0 NM-Type: Indirect network management

☑ Output NM configuration to 'Configuration Log'-Window

☐ Automatic state control for bus 0 (= NM state is based on Communicator's cyclic Tx messages)

DBC Attributes:

Type	Applied to	Name	Value	Description
Bus	Bus 0	NmType	PSA-Indirect-OSEK	Identifies type of NM on this bus
Msg	Msg DEMANDES_EMF	GenMsgTimeoutTime_BSI	250 ms	Rx-Monitoring timeout for node BSI when monitoring message DEMANDES_EMF
Msg	Msg DEMANDES_EMF	NmMessage_BSI	Yes	Node BSI is monitoring message DEMANDES_EMF of node EMF (Rx monitoring)
Msg	Msg RAPPEL_NAV_Z...	GenMsgTimeoutTime_BSI	1400 ms	Rx-Monitoring timeout for node BSI when monitoring message RAPPEL_NAV_ZOOM
Msg	Msg RAPPEL_NAV_Z...	NmMessage_BSI	Yes	Node BSI is monitoring message RAPPEL_NAV_ZOOM of node RT3 (Rx monitoring)
Msg	Msg ETAT_COMBINE	GenMsgTimeoutTime_CMB	300 ms	Tx-Monitoring timeout for node CMB when monitoring message ETAT_COMBINE
Msg	Msg ETAT_COMBINE	NmMessage	Yes	Node CMB is monitoring message ETAT_COMBINE (Tx monitoring)
Msg	Msg COMMANDES_BSI	GenMsgTimeoutTime_CMB	300 ms	Rx-Monitoring timeout for node CMB when monitoring message COMMANDES_BSI
Msg	Msg COMMANDES_BSI	NmMessage_CMB	Yes	Node CMB is monitoring message COMMANDES_BSI of node BSI (Rx monitoring)
Msg	Msg RAPPEL_NAV_Z...	GenMsgTimeoutTime_CMB	1200 ms	Rx-Monitoring timeout for node CMB when monitoring message RAPPEL_NAV_ZO...
Msg	Msg RAPPEL_NAV_Z...	NmMessage_CMB	Yes	Node CMB is monitoring message RAPPEL_NAV_ZOOM of node RT3 (Rx monitori...
Msg	Msg RAPPEL_NAV_Z...	GenMsgTimeoutTime_EMF	1300 ms	Rx-Monitoring timeout for node EMF when monitoring message RAPPEL_NAV_ZO...
Msg	Msg RAPPEL_NAV_Z...	NmMessage_EMF	Yes	Node EMF is monitoring message RAPPEL_NAV_ZOOM of node RT3 (Rx monitori...
Msg	Msg RAPPEL_NAV_Z...	GenMsgTimeoutTime_RT3	1500 ms	Tx-Monitoring timeout for node RT3 when monitoring message RAPPEL_NAV_ZO...
Msg	Msg RAPPEL_NAV_Z...	NmMessage	Yes	Node RT3 is monitoring message RAPPEL_NAV_ZOOM (Tx monitoring)
Msg	Msg DEMANDES_EMF	GenMsgTimeoutTime_RT3	500 ms	Rx-Monitoring timeout for node RT3 when monitoring message DEMANDES_EMF
Msg	Msg DEMANDES_EMF	NmMessage_RT3	Yes	Node RT3 is monitoring message DEMANDES_EMF of node EMF (Rx monitoring)

Picture 23: the network management configuration panel

The configuration panel displays the information to the DBC attributes that were evaluated for the network management configuration for each bus.

The bus can be chosen by the bus control: Bus 0

In stand-alone mode there is the 'Automatic state control' checkbox available. The activation mode of the network management can be toggled here. The state of the checkbox will be saved into and restored from the project file (*.*cmctr-cfg*), individually for each bus.

The 'Output NM configuration' checkbox allows to output the attribute information as well as warnings and errors related to the configuration to the 'Configuration Log Window'. The state of the checkbox will be saved into and restored from the project file (*.*cmctr-cfg*).

The 'Open Log' opens the 'Configuration Log Window'.

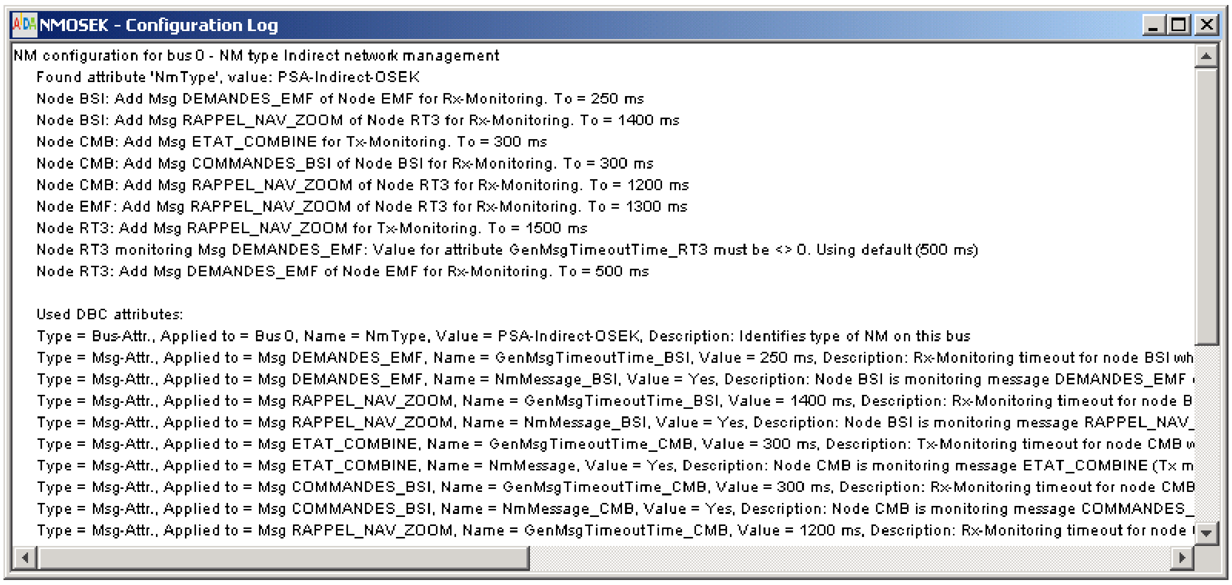
The following attributes are supported:

- Bus attribute 'NWM-Typ' or 'NmType' with value 'Indirect-OSEK' or 'PSA-Indirect-OSEK' or 'RSA-Indirect-OSEK' to identify the Indirect Network Management. Currently there is no difference between the three variants.
- Message attribute ' NmMessage' with value 'yes': If this attribute is found then the sending node is defined to be Tx-monitoring its own transmission. If there are more than one message with this attribute for the same node then only the first message will be used, the other messages are ignored.

- Message attribute ' NmMessage_<ECU>' with value 'yes', where "<ECU>" is a node in the DBC: If this attribute is found for a message of node X, then node X is regarded as 'monitored' by node <ECU>, i.e. node <ECU> is monitoring this message of node X (Rx-Monitoring). If this attribute is found for a message of node <ECU> then the message is ignored. A monitoring node can monitor an arbitrary number of nodes, so this attribute may be defined for messages of other nodes (except for X and <ECU>) also.
- Message attribute GenMsgTimeoutTime_<ECU> (value = integer): If the message is not transmitted by node <ECU>, but by node X: Timeout for Rx-monitoring (attribute NmMessage_<ECU> must be defined for the same message; → node <ECU> is Rx-monitoring this message). If the message is transmitted by node <ECU>: Timeout for Tx-monitoring (attribute NmMessage must be defined for the same message).

If GenMsgTimeoutTime_<ECU> is not found or invalid then a default value (500 ms) is used.

Configuration Log Window:



```
NM configuration for bus 0 - NM type Indirect network management
    Found attribute 'NmType', value: PSA-Indirect-OSEK
    Node BSI: Add Msg DEMANDES_EMF of Node EMF for Rx-Monitoring. To = 250 ms
    Node BSI: Add Msg RAPPEL_NAV_ZOOM of Node RT3 for Rx-Monitoring. To = 1400 ms
    Node CMB: Add Msg ETAT_COMBINE for Tx-Monitoring. To = 300 ms
    Node CMB: Add Msg COMMANDES_BSI of Node BSI for Rx-Monitoring. To = 300 ms
    Node CMB: Add Msg RAPPEL_NAV_ZOOM of Node RT3 for Rx-Monitoring. To = 1200 ms
    Node EMF: Add Msg RAPPEL_NAV_ZOOM of Node RT3 for Rx-Monitoring. To = 1300 ms
    Node RT3: Add Msg RAPPEL_NAV_ZOOM for Tx-Monitoring. To = 1500 ms
    Node RT3 monitoring Msg DEMANDES_EMF: Value for attribute GenMsgTimeoutTime_RT3 must be <> 0. Using default (500 ms)
    Node RT3: Add Msg DEMANDES_EMF of Node EMF for Rx-Monitoring. To = 500 ms

    Used DBC attributes:
    Type = Bus-Attr., Applied to = Bus 0, Name = NmType, Value = PSA-Indirect-OSEK, Description: Identifies type of NM on this bus
    Type = Msg-Attr., Applied to = Msg DEMANDES_EMF, Name = GenMsgTimeoutTime_BSI, Value = 250 ms, Description: Rx-Monitoring timeout for node BSI wh
    Type = Msg-Attr., Applied to = Msg DEMANDES_EMF, Name = NmMessage_BSI, Value = Yes, Description: Node BSI is monitoring message DEMANDES_EMF
    Type = Msg-Attr., Applied to = Msg RAPPEL_NAV_ZOOM, Name = GenMsgTimeoutTime_BSI, Value = 1400 ms, Description: Rx-Monitoring timeout for node B
    Type = Msg-Attr., Applied to = Msg RAPPEL_NAV_ZOOM, Name = NmMessage_BSI, Value = Yes, Description: Node BSI is monitoring message RAPPEL_NAV_
    Type = Msg-Attr., Applied to = Msg ETAT_COMBINE, Name = GenMsgTimeoutTime_CMB, Value = 300 ms, Description: Tx-Monitoring timeout for node CMB w
    Type = Msg-Attr., Applied to = Msg ETAT_COMBINE, Name = NmMessage, Value = Yes, Description: Node CMB is monitoring message ETAT_COMBINE (Tx m
    Type = Msg-Attr., Applied to = Msg COMMANDES_BSI, Name = GenMsgTimeoutTime_CMB, Value = 300 ms, Description: Rx-Monitoring timeout for node CMB
    Type = Msg-Attr., Applied to = Msg COMMANDES_BSI, Name = NmMessage_CMB, Value = Yes, Description: Node CMB is monitoring message COMMANDES_
    Type = Msg-Attr., Applied to = Msg RAPPEL_NAV_ZOOM, Name = GenMsgTimeoutTime_CMB, Value = 1200 ms, Description: Rx-Monitoring timeout for node
```

Picture 24: the NMOSEK configuration log window

If the 'Output NM configuration' checkbox in the 'Configuration Panel' is checked then the attribute information as well as warnings and errors related to the configuration is displayed here, individually for each bus.

3 Installation

3.1 Required Files

All necessary files are part of the AIDA Installation.

Note: To configure and use driver stacks, the Stacker AIDA_Stacker.exe, the stack components **.component* and possibly additional hardware drivers (e.g. CAN driver) are required.

4 Frequently Asked Questions

4.1 How to build your own Commander configuration launching Communicator?

Note: To learn more, you should read the following sections in the AIDA Commander documentation: Building your own Configuration, Editing Commander-Start-Batches.

The original *communicator.cmdr-cfg* entries are as follows:

Start-Batch

```
config.setMainBounds,274,1,718,648

core.listenfor,0,"$(AIDABIN)pi.exe","$(AIDAHOME)","-arg:\"$(cmctr-cfg-file)\"
cmctr.pi"

config.presetNamedLocation, Communicator, 568,1
```

Toolbutton

```
core,close,force

core.listenfor,0,"$(AIDABIN)pi.exe","$(AIDAHOME)","cmctr.pi"
```

The file extension **.cmctr-cfg* is linked to the file `$(AIDAHOME)`*communicator.cmdr-cfg* via a Windows registry entry.

This means, if you want your own Commander configuration launched from the explorer via file extension **.cmctr-cfg*, you have to save it under the same name at the same location. Before you change the standard *communicator.cmdr-cfg* you should make a backup copy.

Here are some typical start batch entries for the different cases:

4.1.1 Stand Alone (CMCTR)

```
core.listenfor,0,"$(AIDABIN)pi.exe","$(AIDAHOME)","-arg:\"$(cmctr-cfg-file)\"
cmctr.pi"
```

4.1.2 Communicator with VAG Network Management (NMVAGO) stand-alone

```
core.listenfor,0,"$(AIDABIN)pi.exe","$(AIDAHOME)","-arg:\"$(cmctr-cfg-file)\"
NMVAGO.pi"
```

4.1.3 Communicator with Indirect Network Management (NMOSEK) stand-alone

```
core.listenfor,0,"$(AIDABIN)pi.exe","$(AIDAHOME)","-arg:\"$(cmctr-cfg-file)\"
NMOSEK.pi"
```

4.1.4 Using Communicator (CMCTR) from your own application

e.g. tst_cmctr.pi, see $(AIDAHOME)examples/tst_cmctr.pool

```
core.listenfor,0,"$(AIDABIN)pi.exe","$(AIDAHOME)examples","-arg:\"$(cmctr-cfg-
file)\" tst_cmctr.pi"
```

4.1.5 Using NMVAGO from your own application

- e.g. nmtest1.pi, see $(AIDAHOME)examples/nmtest1.pool

  ```
  core.listenfor,0,"$(AIDABIN)pi.exe","$(AIDAHOME)examples","-arg:\"$(cmctr-
  cfg-file)\" nmtest1.pi"
  ```

- e.g. nmtest2.pi, see $(AIDAHOME)examples/nmtest2.pool

  ```
  core.listenfor,0,"$(AIDABIN)pi.exe","$(AIDAHOME)examples","-arg:\"$(cmctr-
  cfg-file)\" nmtest2.pi"
  ```

4.1.6 Using NMOSEK from your own application

e.g. nmosek_tstapp1.pi, see $(AIDAHOME)examples/nmosek_tstapp1.pool

```
core.listenfor,0,"$(AIDABIN)pi.exe","$(AIDAHOME)examples","-arg:\"$(cmctr-cfg-
file)\" nmosek_tstapp1.pi"
```

4.1.7 Together with other POOL applications

e.g. the POOL examples tastentelefon.pi and stopwatcha.pi, see $(AIDAHOME)examples

```
core.listenfor,0,"$(AIDABIN)pi.exe","$(AIDAHOME)examples","-arg:\"$(cmctr-cfg-
file)\" cmctr.pi tastentelefon.pi -arg:-t stopwatcha.pi"
```

4.2 How to change the global Communicator settings (i.e. editing *aida-cmctr.ini*)?

For each user, the global settings (like “Most recently used files list”, “Opening the last *.cmctr-cfg at startup” etc.) are saved in the file *aida-cmctr.ini* which is located in the directory referenced by the environment variable `$(USERPROFILE)`. On Windows systems, normally this is C:/Documents and Settings/`$(USERNAME).$(USERDOMAIN)`. (To check if this environment variable is defined on your system, open a console window and type `set USERPROFILE`. To define this environment variable on Windows XP systems, right click your Windows **Workplace** icon, select the **Advanced** tab and click the **Environment Variables** button.)

The file has the following structure:

```
<?xml version="1.0" encoding="windows-1252"?>
<BSKCfg xmlns="http://www.bsk-germany.com/XMLCfg/1.0">
  <section name="AIDA Communicator">
    <Int32 name="WndLeft">0</Int32>
    <Int32 name="WndTop">0</Int32>
    <Int32 name="WndWidth">274</Int32>
    <Int32 name="WndHeight">831</Int32>
    <CharString name="InitialDirectory">K:/Cmctr-Cfgs/</CharString>
    <section name="FileHistory">
      <CharString name="File1">K:/Cmctr-Cfgs/Test1.cmctr-cfg</CharString>
      <CharString name="File2">K:/Cmctr-Cfgs/Test2.cmctr-cfg</CharString>
      <CharString name="File3">K:/Cmctr-Cfgs/Test3.cmctr-cfg</CharString>
    </section>
    <Boolean name="OpenLastCmctrCfg">false</Boolean>
    <Boolean name="ResetAutoUpdate">true</Boolean>
    <CharString name="DbcEditorFullPath">C:/CANoe/CANdb.exe</CharString>
    <Int32 name="TimeoutCorrection_ms">0</Int32>
  </section>
</BSKCfg>
```

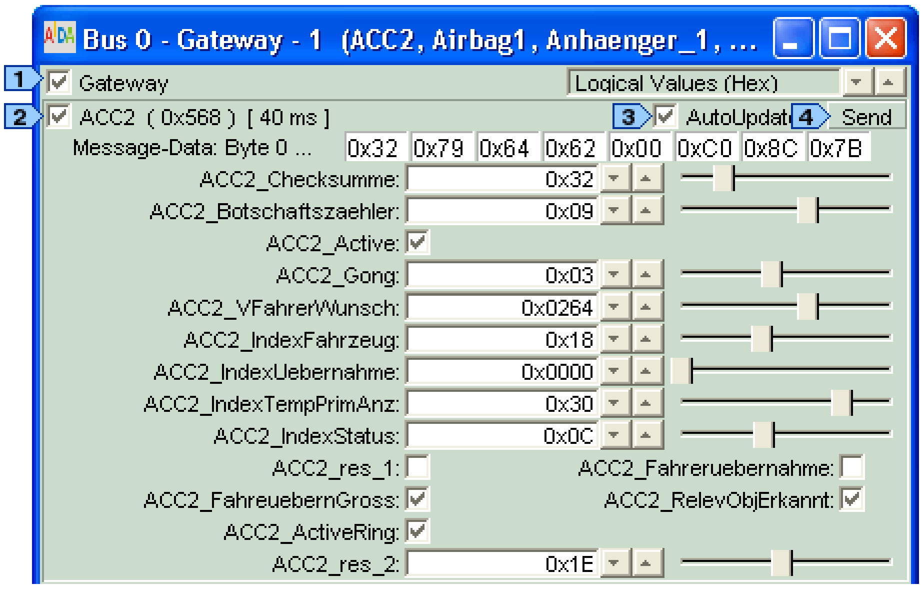
If `OpenLastCmctrCfg` is set to true, the most recently used **.cmctr-cfg* will be loaded at startup without configuration file parameter.

If `ResetAutoUpdate` is set to true, a message's **`AutoUpdate`** check box within the CAN panels will be automatically deselected at deselection of the corresponding check box for cyclic transmission, otherwise is keeps its current state.

To change these global settings, close all Communicator instances and edit the *aida-cmctr.ini* file in your preferred text editor.

4.3 How to select a message for reception, transmission, etc. ?

The different combinations of settings relevant for a message and its corresponding tx node and their effect:


Bus 0 - Gateway - 1 (ACC2, Airbag1, Anhaenger_1, ...

1 ✓ Gateway — Logical Values (Hex)

2 ✓ ACC2 (0x568) [40 ms] — 3 ✓ AutoUpdat — 4 Send

Message-Data: Byte 0 ... 0x32 0x79 0x64 0x62 0x00 0xC0 0x8C 0x7B

ACC2_Checksumme:	0x32
ACC2_Botschaftszaehler:	0x09
ACC2_Active:	✓
ACC2_Gong:	0x03
ACC2_VFahrerWunsch:	0x0264
ACC2_IndexFahrzeug:	0x18
ACC2_IndexUebernahme:	0x0000
ACC2_IndexTempPrimAnz:	0x30
ACC2_IndexStatus:	0x0C
ACC2_res_1:	☐
ACC2_Fahreruebernahme:	☐
ACC2_FahreuebernGross:	✓
ACC2_RelevObjErkannt:	✓
ACC2_ActiveRing:	✓
ACC2_res_2:	0x1E

Picture 22: the upper part of an auto panel

TxNode 1	Msg 2	`AutoUpdate` 3	**Effect**	`Send` **button** 4
☐	✓	--	The message is selected for reception.	--
☐	☐	--	The message is inactive.	--
✓	☐	☐	The message is not selected for cyclic trans-mission. Changing the message data doesn't result in spontaneous transmission.	The current data is send spontaneously.
✓	☐	✓	The message is not selected for cyclic trans-mission. After the message data is changed, the message is send spontaneously.	
✓	✓	☐	The message is selected for cyclic transmission. Value changes are ignored.	Pressing the send button results in a spontaneous trans-mission of the new message data and additionally, starting with the next cycle the new data is adopted.

✓	✓	✓	The message is selected for cyclic transmission. Value changes are adopted starting with the next cycle.	The current data is send spontaneously out of cycle.

5 Index

[…]