CAN-Viewer
446 301 585 0 (de)
446 301 599 0 (en)
of version 1.10

Operating Instructions

2nd Edition

This publication is not subject to any update service.
You will find the new version in INFORM under
www.wabco-auto.com

WABCO

Version 002/03.10
815 010 118 3(en)

Table of Contents

Explanation of symbols

! Additional instructions, information or tips that you should always observe.

- List

– Step

1 General safety instructions

Only trained and qualified personnel may work with the CAN-Viewer.

You must follow the specifications and instructions of the vehicle manufacturer unconditionally.

Comply with the company and national accident prevention/health & safety regulations.

Make sure that the gearbox is in neutral and the hand brake is applied. When working on the brake system, the vehicle must be secured against rolling away.

Wear any necessary protective clothing.

The workplace has to be dry and sufficiently lighted and ventilated.

2 Introduction

2.1 CAN-Viewer

By means of the CAN-Viewer, CAN messages from any CAN bus can be easily read out, decoded in plain text, and saved. This is accomplished without complicated measuring technology and time.

Main field of application for the CAN-Viewer is tracing the flow of messages from EBS CAN bus to trailer interface.

! The ability to display the communication between tractor vehicle and trailer permits checking the data exchange function, which is part of the duty of care required for acceptance of the vehicle.

2.2 CAN bus

Controller Area Network (CAN) bus
The CAN bus is an asynchronous (time-shifted), serial bus system, originally developed by Bosch to connect electronic control devices in motor vehicles and was introduced jointly with Intel to reduce cable harnesses and thereby weight.
[Source: www.wikipedia.de - Die freie Enzyklopädie (03/2006)]

Instead of using an electrical circuit for each transmitted signal, the "bus" is based on a communication platform which regulates the relaying of messages between several devices.

In a practical context, the process may be imagined as follows:
While the rear light was actuated by means of guiding a current to the rear light, the bus system only relays a message: "Light switch to rear light: Switch on!"

Translating all control signals into messages requires a "greater intelligence" of the connected devices, at the same time this implies that many devices can exchange information, virtually at same time, using a very limited number of cable connections. For systems such as EBS it is also possible to exchange much more complex information e. g. on current pad wear or on individual wheel speeds in the case of ABS.

3 Program surface

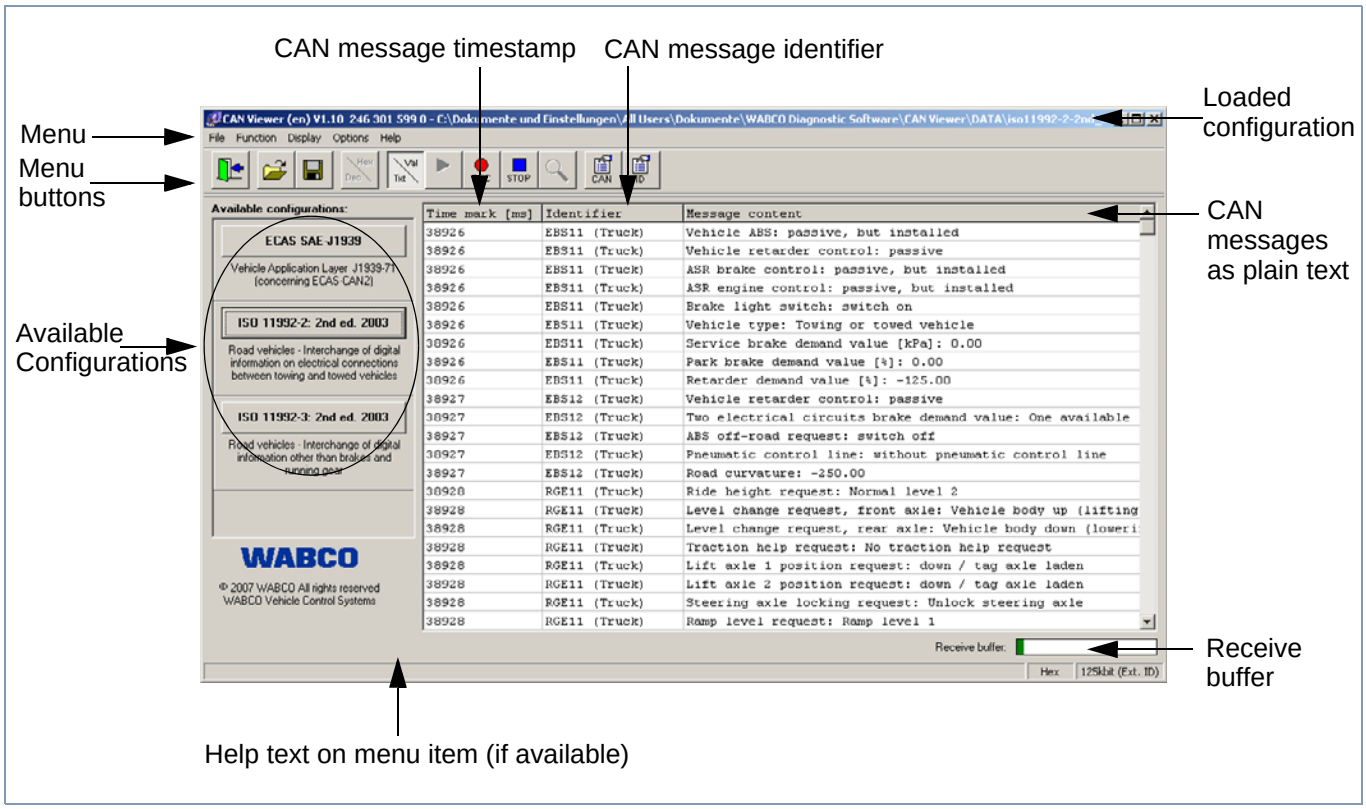

Fig. 1 Program surface with display of current CAN messages

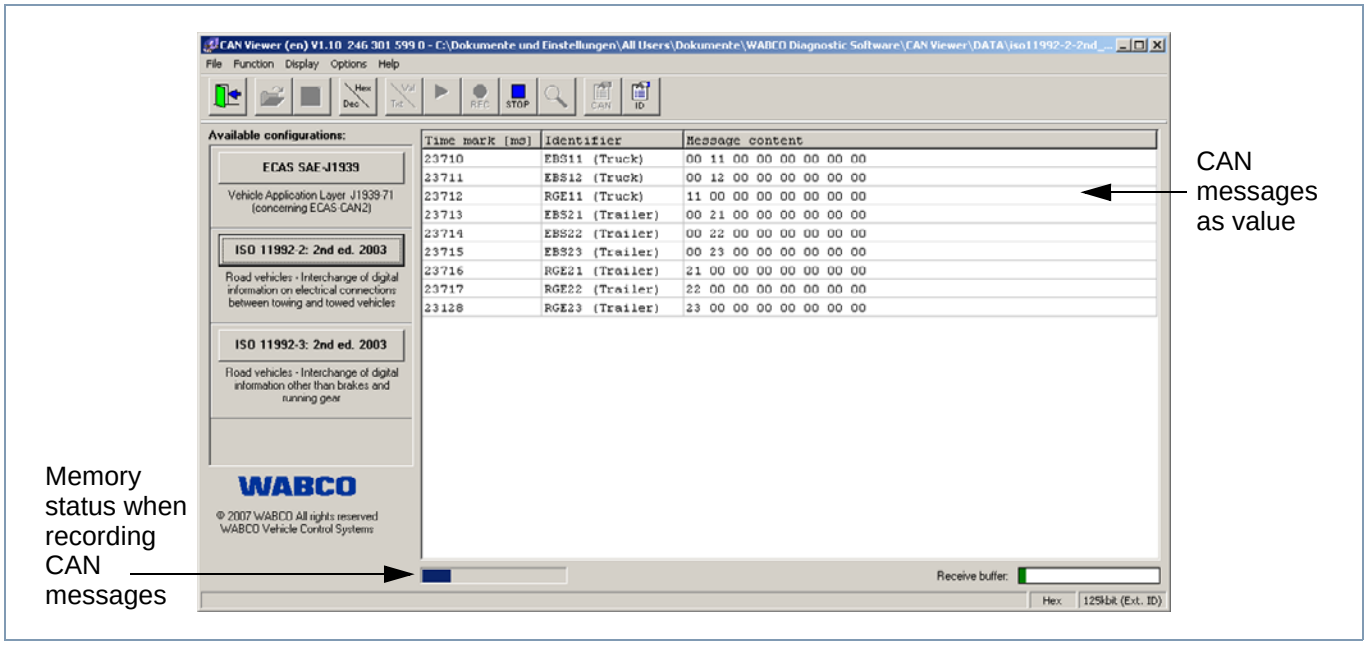

Fig. 2 CAN-Viewer in recording mode

Button	Menu/Menu item	Description
	File management	
	Open	Opening and loading a configuration file
	Save	Saving the current settings to a configuration file
	Exit	Exiting the program and the CAN connection
	Function	
	Starting CAN reception	Starting the reception and display of CAN messages
	Recording of the CAN data	Recording the CAN messages in a protocol
	Stop	Stopping the recording process and saving to a file
	Searching CAN messages	Search and display of all CAN messages from any CAN bus
	indicator	
	Decimal or hexadecimal display ! The hexadecimal system has the value 16 as its base. (our decimal system has the base 10.) Numerical values are written using the numerals 0 to 9 and the capital letters A to F.	Switching display of protocol between decimal and hexadecimal values **Not activated:** Display using hexadecimal notation **Activated:** Display using decimal notation ! This function is ignored if the CAN data is displayed as plain text (see menu item "CAN data display").
	CAN data display	Switching display of protocol between CAN data as value or plain text **Not activated:** Display of CAN data as value **Activated:** Display of CAN data as plain text ! The plain text of values can only be displayed if a description of these values is available in the configuration file.

Button	Menu/Menu item	Description
	Options	
	CAN configuration...	Setting the CAN bus parameters (see chapter 4.5 Display of one or a number of specific CAN message(s))
	Identifier configuration...	Setting the CAN messages filter (see chapter 4.6 Filtering the displayed CAN messages (identifier configuration))
	Settings ...	Adjusting general program settings e.g. interface, data file location, ... (see chapter 5.1 Program settings)
	Help	
	About ...	Opening the program information (e.g. version, serial number, ...)

4 Application

4.1 Set-up of cable connection to vehicle via ISO 7638 (7-pin)

Fig. 3 Cable connection via ISO 7638

Required components:

- 446 301 001 0 Diagnostic interface with order number 446 301 022 0 or 446 301 030 0 from version 1.20 (or alternatively the diagnostic interface 446 301 000 0 with order number 446 301 021 0 together with CAN Converter 446 300 470 0)
- 446 300 360 0 CAN diagnostic adapter
- 446 300 458 0 CAN-Viewer connection cable for ISO 7638

! A fault-free cable connection to the vehicle is a basic requirement for the following instructions.

4.2 Set-up of cable connection to vehicle via ISO 12098 (15-pin)

Fig. 4 Cable connection via ISO 12098

Required components:

- 446 301 001 0 Diagnostic interface with order number 446 301 022 0 or 446 301 030 0 from version 1.20 (or alternatively the diagnostic interface 446 301 000 0 with order number 446 301 021 0 together with CAN Converter 446 300 470 0)
- 446 300 459 0 CAN-Viewer connection cable for ISO 12098

! A fault-free cable connection to the vehicle is a basic requirement for the following instructions.

4.3 Software installation and activation

– Plug the USB flash drive into the USB port of the PC.

– To install the software on the PC, start the file SETUP.EXE in the main directory of the USB flash drive.

– Follow the installation instructions on the screen.

After you have installed the software you must activate it; the preferred mode is online.

Activation means linking the software with the PC hardware. Use of the software is limited to **one** PC per obtained licence.

WABCO permits you to use the software in accordance with the licensing conditions for ten days without activation.

! If you have any questions, the WABCO website **www.wabco-auto.com**, or your WABCO partner, will also provide further assistance.

4.4 Display of messages from the tractor vehicle trailer interface (ISO 11992-2 or 11992-3)

In the event that the CAN-Viewer is currently receiving CAN messages, proceed as follows:

– Click this button to end receive mode.

– From the list of *Available Configurations*, select the corresponding ISO by clicking the respective button.

The CAN-Viewer automatically starts the reception of CAN messages.

– Click the button if you want to switch the current display of CAN data to plain text.

If you only want to view specific CAN messages, you can also filter them (see chapter 4.6 Filtering the displayed CAN messages (identifier configuration)).

! Scope and content of the CAN messages corresponds to the selected ISO. There you will also find specific information.

4.5 Display of messages from any CAN bus

In the event that the CAN-Viewer is currently receiving CAN messages, proceed as follows:

– Click the button to end this mode.

– Click the button to open the CAN configuration dialog.

– In the dialog, select the corresponding *Baudrate* and the *Identifier* for the desired bus.

– Click *OK* to close the dialog.

The CAN-Viewer automatically starts the reception of CAN messages.

! It is not possible to display the values as plain text because the required decoding of the CAN data is not been stored.

4.6 Display of one or a number of specific CAN message(s)

In the event that the CAN-Viewer is currently receiving CAN messages, proceed as follows:

– Click this button to end receive mode.

– Click the button to open the CAN configuration dialog (see fig. 5) .

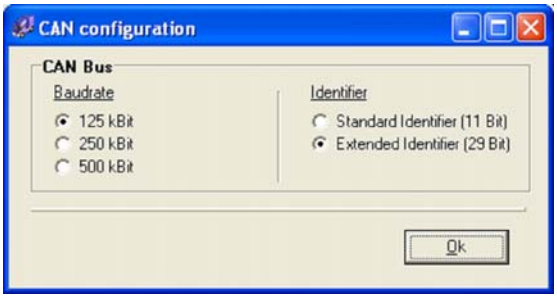

Fig. 5 Dialog for the CAN configuration

– In the dialog, select the corresponding *Baudrate* and the *Identifier* for the desired bus.

– Click *OK* to close the dialog.

– Click the button to open the dialog for filtering the CAN messages.

Since no identifiers are yet known or have been defined, proceed as follows:

– Click the button *New* to define a new identifier.

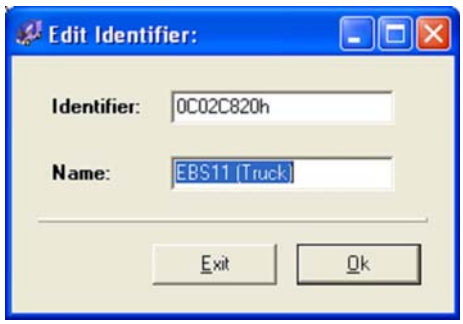

Fig. 6 Dialog for editing an identifier

– In this dialog, enter the ID and any name you choose (see fig. 6).

– Click *OK* to close this dialog.

You may repeat this procedure for all desired CAN messages.

– Click *OK* to close the dialog for defining identifiers as well.

– Click this button to start reception.

! It is not possible to display the values as plain text because the required decoding of the CAN data is not been stored.

4.7 Filtering the displayed CAN messages (identifier configuration)

In the event that the CAN-Viewer is currently receiving CAN messages, proceed as follows:

– Click this button to end receive mode.

– Click the button to open the dialog for filtering the CAN messages.

– In the upper choice list (IDENTIFIER), select the desired identifier by checking the corresponding checkbox.

– In the lower choice list (PARAMETER), select the desired parameter for the selected identifier by checking the corresponding checkbox.

– Click *OK* to close this dialog.

Only the desired CAN messages will now be displayed in the program window.

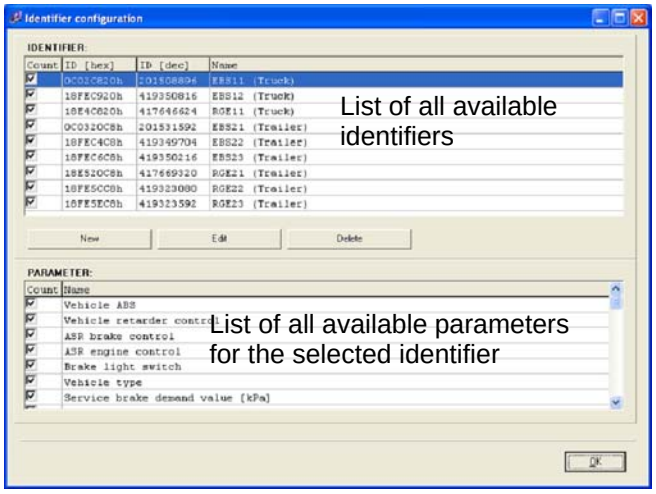

Fig. 7 Dialog for filtering the displayed CAN messages (identifier configuration)

4.8 Creating / Editing an identifier

In the event that the CAN-Viewer is currently receiving CAN messages, proceed as follows:

– Click this button to end receive mode.

– Click the button to open the dialog for filtering the CAN messages.

– Click the button *New* to define a new identifier.

– To change an identifier, you need to select it first.

– Click the *Edit* button to start the dialog.

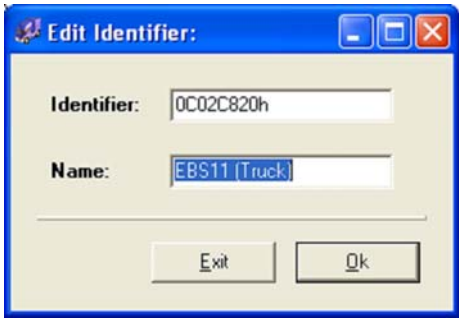

Fig. 8 Dialog for editing an identifier

– In this dialog, enter the ID of the identifier and any name you choose .

– Click *OK* to close this dialog.

You may repeat this procedure for all desired CAN messages.

– Click *OK* to close the dialog for defining identifiers as well.

4.9 Creating a configuration file with your own settings

– Adjust the display of the CAN messages as required.

– Click the button to open the dialog for saving a configuration dialog.

– Specify a location and a name for the configuration file.

– Click the button *Save* to close this dialog.

! When saving a configuration file, the information for decoding to plain text is lost.

4.10 Loading your own configuration file

– Click this button to open the dialog.

– Select the desired configuration file.

– Click the button *Open* to close this dialog.

The CAN-Viewer automatically starts the reception of CAN messages.

4.11 Recording and saving a protocol with CAN messages to a file

In the event that the CAN-Viewer is currently receiving CAN messages, proceed as follows:

– Click this button to start reception.

– Click this button to start the recording mode.

– Click this button to stop recording after the desired time.

After recording has been stopped, the dialog for saving the record file is opened automatically.

– Specify a location and a name for the record file.

– Click the button *Save* to close this dialog.

! The CAN-Viewer is only able to record a limited number of CAN messages. You can see the current memory status in the bottom left corner of the program window (see fig. 2).

As soon as all the memory has been used up, the recording process is stopped automatically and a dialog for saving the record file is opened.

The default setting for the number of CAN messages that can be received may be adjusted in the menu *Options/Settings* between 10,000 and max. 1,000,000. To achieve this, values between 10 and 1,000 need to be entered.

! The content of the CAN messages is only saved as a value not as plain text.

5 Administration

5.1 Program settings

Tab (Function)	Description
Serial port	Configure connection to the Interface
Currently selected interface or USB device	Displays the currently set interface, e. g. COM1 or USB
Change interface	Configuring the interface
Data file location	Defining storage locations for the files
Reading directory	Reading directory for reading out the configuration files
Writing directory	Default directory for saving files
Program options	General program options
Start program in full screen mode	The program window is opened in full screen mode.
Display help texts (bubble help)	A help text is displayed when the mouse cursor is held in position over the respective button.
Number of CAN messages when recording	For a more detailed description see chapter 4.11
User data	User details
Company	Company name
User name	Name of the registered user
Serial number	Serial number

5.2 Creating a configuration file for decoding CAN messages in plain text

The following remarks apply with respect to creating a configuration file:

- A configuration file is structured according to a syntax similar to Windows INI files.
- Comment lines begin with a semicolon **;** and are thereby ignored during the process of interpreting the file.
- The structure consists of three sections which are marked accordingly:
  - **[FileInfo]:** This section contains information on the configuration file.
  - **[Config]:** This section contains details regarding the CAN bus configuration. The standard configurations also contain the description texts.
  - **[Identifier]:** This section contains all the data related to the CAN identifiers and the standard configurations also contain the descriptions on interpreting the content of the CAN data.
- Each section is described by keywords and appertaining values separated by an equals sign **=**. Here the syntax must be strictly adhered to, otherwise the configuration cannot be read in correctly.
- The predefined configuration files in the CAN-Viewer reading directory may be of assistance.

5.2.1 Section [FileInfo]:

Keyword	Value	Description	Explanation
FileType	CANLOGGERCONFIG CANLOGGERUSERCONFIG	Standard configuration Own configuration	Describes whether this configuration file is a standard configuration or one of your own.
FileStructureVersion	1.0		This number indicates the file structure version. So far, version 1.0 is the only one.

Table 3 - Keywords from section [FileInfo]

Example:
[FileInfo]
FileType = CANLOGGERCONFIG
FileStructureVersion = 1.0

5.2.2 Section [Config]

Keyword	Value	Description	Explanation
Name	Free text		This key contains the text that is used in the configuration button in the main dialog. It is only taken into account for the standard configurations. The text in the button does not wrap and should therefore be kept short (single-line).
Hint	Free text		This key contains the text that is displayed underneath the configuration button in the main dialog and is only taken into account for standard configurations. The text is wrapped automatically and may therefore be a little longer than the name.
ExtIdentifier	0 1	Standard Identifier (11 bit) Extended Identifier (29 bit)	Here is specified whether it is a standard (11 bit) or extended (29 bit) identifier. ! This entry is mandatory.
baud rate	0	125 kBit	Here the CAN baud rate is set. ! This entry is mandatory.
	1	250 kBit	
	2	500 kBit	

Table 4 - Keywords from section [Config]

Example:
[Config]
Name = ISO11992 (2001)
Hint = Tractor vehicle / Trailer interface
ExtIdentifier = 1
Baud rate = 1

5.2.3 Section [Identifier]

The structure of entries is always as follows:	
IDxxx_	• xxx is a serial number (001..0.099). • A maximum of 99 identifiers are supported. • There must not be any gaps within the numbering system.
The parameters pertaining to the identifier are always structured as follows:	
IDxxx_Pyy	• yy is a serial number (001...99). • A maximum of 99 parameters are supported. • There must not be any gaps within the numbering system.

The bit encoding of parameters is structured as follows:	
IDxxx_Pyy_BCzzz	• zzz is a serial number (001...256). • There must not be any gaps within the numbering system.

.

Keyword	Value	Description	Explanation
IDxxx_Name	Free text		This entry contains the name of the identifier in plain text. If this name is available, it will be displayed in the CAN-Viewer during the receiving process. ! This entry is not mandatory.
IDxxx_ID	decimal or hexadecimal		Here the identifier value is specified. This value may be entered in decimal or hexadecimal notation. ! This entry is not mandatory.
IDxxx_Visible	0 1	Identifier is not displayed Identifier is displayed	At this point it is specified whether or not the identifier should be displayed during reception.
IDxxx_Pyy_Name	Free text		Name of the parameter as it is to be displayed. This entry is mandatory.
IDxxx_Pyy_Visible	0 1	Identifier is not displayed Parameter is displayed	At this point it is specified whether or not the identifier should be displayed during reception.
IDxxx_Pyy_BytePos	Integer		Description in which data byte of the CAN message the parameter begins. The numbering system begins with 1. ! This entry is mandatory.

Keyword	Value	Description	Explanation
IDxxx_Pyy_ByteSize	Integer		Description of how many data bytes the parameter comprises. The numbering system begins with 1. ! This entry is not mandatory. The default value is 1.
IDxxx_Pyy_BitPos	Integer		Description at which bit position the parameter begins. The numbering system begins with 1. ! This entry is mandatory.
IDxxx_Pyy_BitSize	Integer		Description of how many data bits the parameter comprises. The numbering system begins with 1. ! This entry is mandatory.
IDxxx_Pyy_ShowType	1 2 3 4 5	Display bit-encoded (a text is assigned to each bit combination) Display as integer value Display as hexadecimal value Display as floating-point number (with two decimal digits) Binary display	Description of how the parameter is displayed.
IDxxx_Pyy_Formula	0 1 2 3 4 5 6 7 8 9	Y = C0 * X +C1 Y = C0 * (X + C1) Y = C0 / (X + C1) + C2 Y = X / C0 + C1 Y = (X + C0) / C1 Y = (X + C0) / C1 + C2 Y = C0 * X Y = X / C0 Y = X + C0 Y = X * C0 / C1	The parameter values may be converted prior to being displayed. ! This entry is mandatory. If it is not available, no conversion will occur.
IDxxx_Pyy_Formula_C0 IDxxx_Pyy_Formula_C1 IDxxx_Pyy_Formula_C2		Constant 1 Constant 2 Constant 3	Here the three possible constants may be specified for the conversion by means of a formula.
IDxxx_Pyy_BCzzz_Value			The values for bit-encoded display are stored here. In this context, xxx is a serial number. The text for this number must also be specified.
IDxxx_Pyy_BCzzz_Text	Free text		The text for the corresponding value.
IDxxx_Pyy_MinVal IDxxx_Pyy_MaxVal		Minimum value Maximum value	Minimum and maximum values of the computed parameters. If the computed value is outside these limits, "NTBU" is displayed in the CAN-Viewer.

Table 5 - Keywords from section [Identifier]

Example (one identifier with two parameters):

```
[Identifier]
ID001_Name              = EBS11 (Truck)
ID001_ID                = 201508896
ID001_Visible           = 1
ID001_P01_Name          = Vehicle ABS
ID001_P01_Visible       = 1
ID001_P01_BytePos       = 1
ID001_P01_ByteSize      = 1
ID001_P01_BitPos        = 1
ID001_P01_BitSize       = 2
ID001_P01_ShowType      = 1
ID001_P01_BC001_Value   = 0
ID001_P01_BC001_Text    = passive, but installed
ID001_P01_BC002_Value   = 1
ID001_P01_BC002_Text    = active
;-----------------------------------
ID001_P02_Name          = Vehicle retarder control
ID001_P02_Visible       = 1
ID001_P02_BytePos       = 1
ID001_P02_ByteSize      = 1
ID001_P02_BitPos        = 3
ID001_P02_BitSize       = 2
ID001_P02_ShowType      = 1
ID001_P02_BC001_Value   = 0
ID001_P02_BC001_Text    = passive
ID001_P02_BC002_Value   = 1
ID001_P02_BC002_Text    = active
```

6 Further information on the CAN bus

6.1 Transmission method

The CAN bus is either a copper wire or fibre glass design. The CAN bus system is used for fast transmission of data between control units. The CAN bus operates according to the "multi-master principle": A number of control devices (= bus nodes) with equal priority are interconnected by a topological arrangement.

In the case of copper cables, the CAN bus operates using differential signals. Generally, 3 lines are integrated:

- CAN_HIGH
- CAN_LOW
- CAN_GND (ground)

CAN_LOW contains the complementary level of CAN_HIGH against ground. Common-mode interferences are suppressed by these means because the differential remains the same.

CAN transmits data through a binary model of either "dominant" bits and "recessive" bits, relative to status, acting on the bus lines. A dominant bit here overwrites a recessive bit.

6.2 Transfer rate and line length

A CAN bus is differentiated as either a high speed or a low speed bus. A high speed bus has a maximum transmission rate of 1 Mbit/s, a low speed bus 125 kBit/s.

The maximum (theoretical) line length is, for 1 Mbit/s 40 m, for 500 kBit/s 100 m and for 125 kBit/s 500 m. These maximum values derive from the fact that the time a signal is present on the bus (bit time, bit/seconds) is correspondingly shorter the higher the transfer rate. With increasing line length, the bit also requires more time to reach the other end of the bus. The time that a signal is present on the bus must therefore not be shorter than the time required for the signal to spread.

The maximum number of bus nodes at the physical layer depends on the bus driver components used. Ordinary modules permit 32, 64 or up to 110 (with restrictions up to 128) nodes per line.

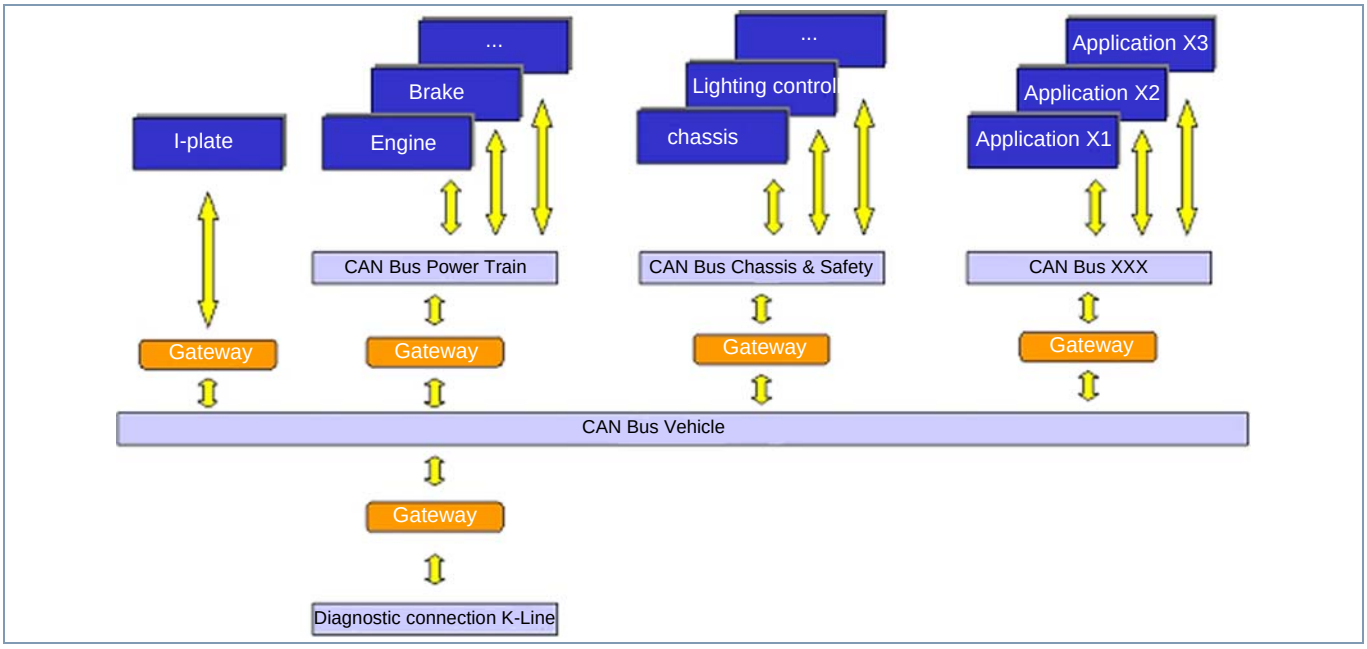

Fig. 9 Example of a CAN bus set-up

[Contents in chapters 6.1 - 6.5 from source: www.wikipedia.de - Die freie Enzyklopädie (03/2006)]

6.3 Object identifier

The object identifier designates the content of a message, not the device. In a measuring system, it is possible, for example, to assign an identifier to each of the parameters *temperature*, *voltage*, *pressure*. The recipients use the identifier to determine whether or not the message is relevant to them. The object identifier also serves the purpose of assigning a priority to messages.

The specification defines two different identifier formats:

- 11bit identifiers, also called "base frame format“.
- 29 bit identifiers, also called "extended frame format".

A node may be recipient and sender of messages with any number of identifiers. However, a maximum of one sender may be assigned to an identifier (for the arbitration to function).

6.4 Arbitration (negotiating media access), priority

Bus access is resolved free of losses by means of a bitwise arbitration (as just an allocation of resources to the different devices as possible) on the basis of the identifiers of the messages to be transmitted. For this purpose, each transmitter senses the bus while he is sending the identifier. In the event that two nodes are transmitting simultaneously, the first dominant bit of the two overwrites the corresponding recessive bit of the other node. This process is detected by the other node, which then stops the attempt to transmit so that the other node may transfer its data. If both nodes use the same identifier, an error frame is generated (see structure 6.5 Frame structure). For this reason, the standard recommends that an identifier should only be used by a maximum of one node.

This procedure also establishes a hierarchy of messages in relation to one another. The message with the lowest identifier may "always" be transmitted. For the transmission of time-critical messages it is therefore possible to assign an identifier of high priority (= low ID, e.g. 0) to grant priority during transmission. However, even for messages with a high priority it is not possible to determine the precise moment of transmission in advance (non-deterministic behaviour).

6.5 Frame structure

There are two types of frames:

Data Frame	is used to transport up to 8 objects of data
Remote Frame	is used to request a data frame from another node
Error Frame	signals to all nodes that a fault condition was detected in transmission
Overload Frame	is used to enforce an intermission between data and remote frames

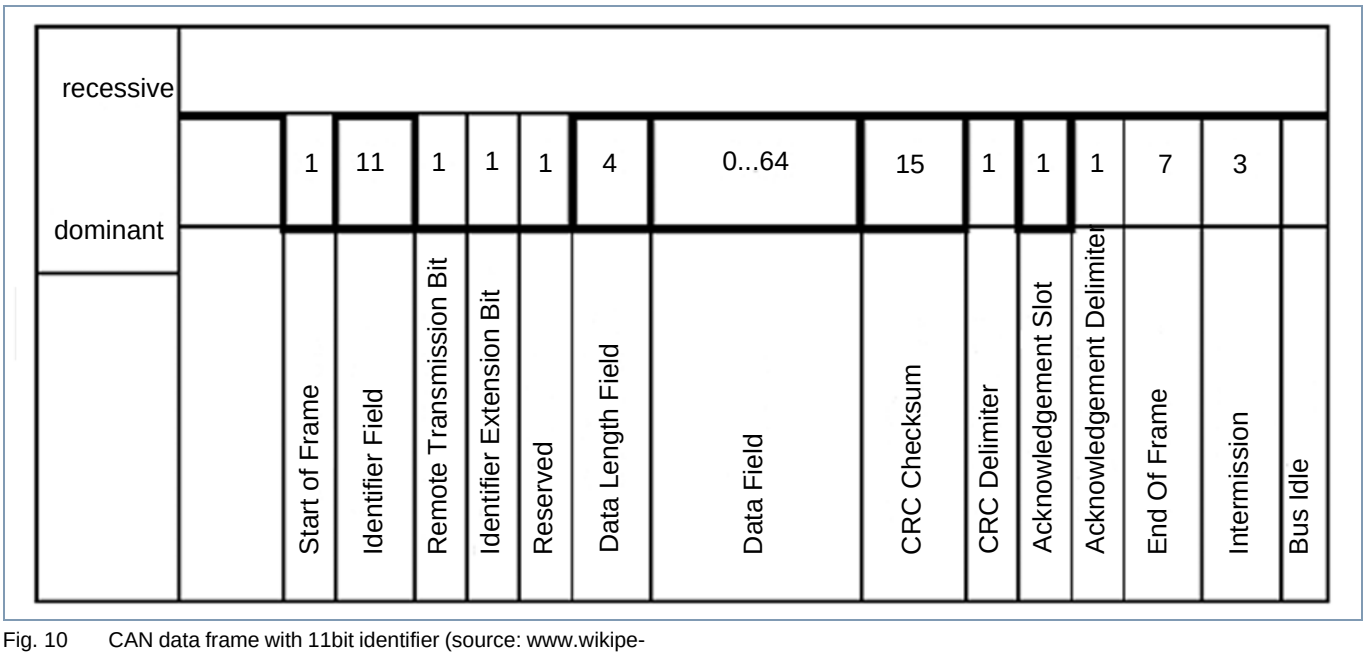

Fig. 10 CAN data frame with 11bit identifier (source: www.wikipedia.de - Die freie Enzyklopädie)