

XCP - Universal Measurement and Calibration Protocol

User Manual
(Your First Steps)

Version 1.0.2



Authors:	Klaus Emmert
Version:	1.0.2
Status:	Released (in preparation/completed/inspected/released)

Motivation For This Work

The motivation for using the XCP Software Component is very simple. XCP is the component that helps you to look in the ECU, to display values or set them. And that is all working just via the underlying bus system as CAN, Ethernet, ...you do not need any development environment.

Together with CANape you have a great choice of how the data could be displayed.

WARNING

All application code in any of the Vector User Manuals are for training purposes only. They are slightly tested and designed to understand the basic idea of using a certain component or a set of components.

Contents

1	Welcome to the XCP User Manual	6
1.1	Beginners with XCP start here ?	6
1.2	For Advanced Users	6
1.3	Special topics	6
1.4	Documents this one refers to.....	6
2	About This Document	7
2.1	How This Documentation Is Set-Up	7
2.2	Legend and Explanation of Symbols.....	7
3	XCP Software Component – An Overall View.....	8
3.1	What Is XCP.....	8
3.2	What Is the XCP Component	8
3.3	Tools And Files – The Generation Tool	8
3.4	What The Component Does.....	9
4	XCP – A More Detailed View.....	11
4.1	Basic Mechanism of the XCP.....	11
4.2	Measurement Modes	11
4.2.1	Using XCP In Polling Mode.....	11
4.2.2	Using XCP For Data Acquisition / Event	12
5	XCP IN 8 STEPS.....	13
5.1	STEP 1 Installation Of The Tool.....	14
5.2	STEP 2 Extract CANbedded Software Components	14
5.3	STEP 3 Configuration With The Generation Tool (GENy)	15
5.4	STEP 4 Generate Files	16
5.5	STEP 5 Add CANbedded To Your Project.....	17
5.6	STEP 6 Adapt Your Application Files.....	18
5.6.1	Including, Initialization And Cyclic Calls	18
5.6.2	Connect your application to the XCP	18
5.7	STEP 7 Compile And Link Your Project.....	19
5.8	STEP 8 Test It Via CANape	20
6	Further Information	22
6.1	Settings For Using The Data Acquisition Mode / Events	22
7	List Of Experiences.....	25
7.1	Topic 1	25

8 Index 1

Illustrations

Figure 3-1	CANbedded Software Components Together With The XCP Component.....	8
Figure 3-2	Generation Process For Vector CANbedded Software Components	9
Figure 4-1	Visualization Of Data Using CANape.....	11
Figure 4-2	XCP Polling Mode Using Two Messages.....	12
Figure 4-3	XCP Event Mode.....	12
Figure 5-1	8 Steps to CANdesc.....	13
Figure 5-2	Configuration In The Generation Tool For Basic XCP Usage.....	15
Figure 5-3	Set Driver Configuration In CANape	20
Figure 5-4	XCP Driver Settings	20
Figure 5-5	Settings For XCP On CAN	21
Figure 5-6	ECU Is Online	21
Figure 6-1	Components / XCP	22
Figure 6-2	Components / XCP	22
Figure 6-3	Components / XCP	23
Figure 6-4	Edit Measurement List In CANape.....	24

1 Welcome to the XCP User Manual

1.1 Beginners with XCP start here ?

You need some **information** about this document?

What is **XCP**?

1.2 For Advanced Users

Start reading **here**.

8 Steps for Flash Bootloader integration.

1.3 Special topics

Working with Data Acquisition Lists (**DAQ**)

1.4 Documents this one refers to...

CANape Manual

Chapter 2

Chapter 3.1

Chapter 4

Chapter 5

Chapter 6.1

2 About This Document




This document gives you an understanding of the Universal Measurement And Calibration Protocol. You will receive general information, a step-by-step tutorial to get the XCP running and further information to use the amount of functionality that XCP offers.

2.1 How This Documentation Is Set-Up

Chapter	Content
Chapter 1	The welcome page is to navigate in the document. The main parts of the document can be accessed from here via hyperlinks
Chapter 2	It contains some formal information about this document, an explanation of legends and symbols.
Chapter 3	In this chapter you get a brief introduction into the XCP and the Software Component.
Chapter 4	Here you find some more insight in the XCP Software Component.
Chapter 5	Here are the 8 steps for you to integrate the XCP Software Component and how to connect these settings with your application.
Chapter 6	This chapter provides you with some further information and a deeper insight in the XCP Software Component.
Chapter 7	In this last chapter there is a list of experiences with XCP.

2.2 Legend and Explanation of Symbols

You find these symbols at the right side of the document. Use this helpful feature to find fast the topics, you need information about.

Symbol	Meaning
	Follow the eye and you will be led to an example.
	You will find key words and information in short sentences in the margin. This will greatly simplify your search for topics.
	The footprints will lead you through the steps until you can use the XCP Software Component.

These areas to the right of the text contain brief items of information that will facilitate your search for specific topics.

3 XCP Software Component – An Overall View

3.1 What Is XCP

The XCP Calibration Protocol was standardized by the European ASAM working committee for standardization of interfaces used in calibration and measurement data acquisition. XCP is a higher level protocol used for communication between a measurement and calibration system (e.g. CANape) and an ECU.

CANape is a PC tool of Vector Informatik GmbH designed for measurement and calibration. To use CANape e.g. for CAN you need a PC card like CANcardXL to connect you ECU to your PC (CANape) via CAN.

XCP is a further development of the established CCP (CAN Calibration Protocol) using different media like CAN, Ethernet (TCP/IP, UDP/IP), USB, SCI. The future trends will be FlexRay, TTCAN, Firewire.

CANape is a PC tool of Vector Informatik GmbH designed for measurement and calibration.

To use CANape for CAN you need a PC card like CANcardXL for the CAN connection

3.2 What Is the XCP Component

XCP is a CANbedded Software Component delivered as source code (.c, .h) and can be parameterized via the Generation Tool.

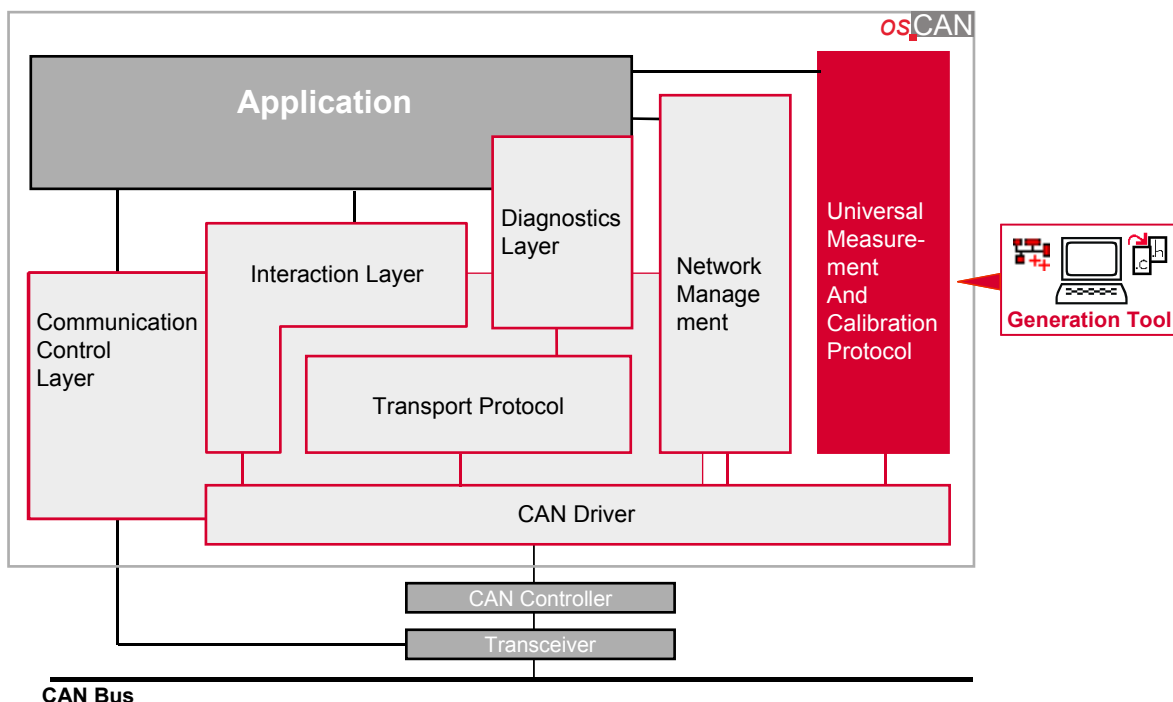


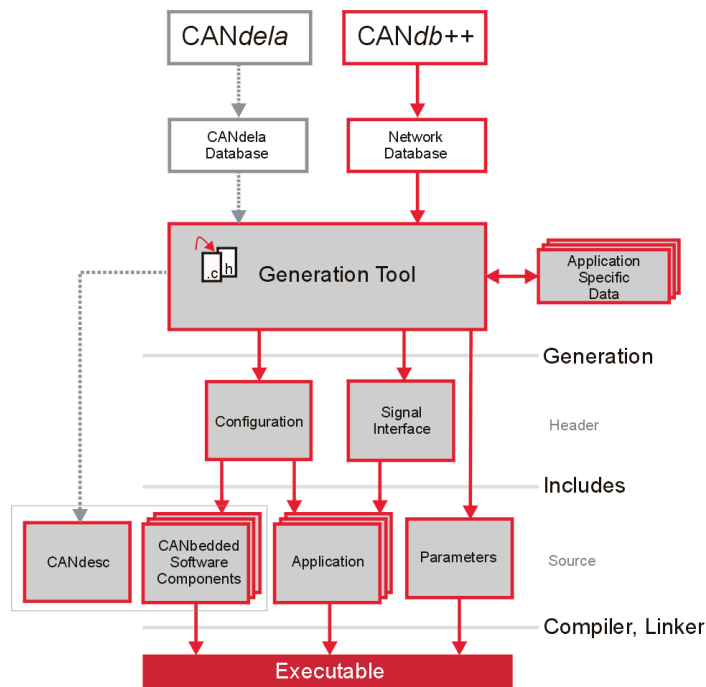
Figure 3-1 CANbedded Software Components Together With The XCP Component

3.3 Tools And Files – The Generation Tool

The Generation Tool is a PC Tool. It generates configuration files and signal interface files for the CANbedded Software Components. The Generation Tool needs the DBC file to generate the files.

The DBC file normally is designed by the vehicle manufacturer and distributed to all suppliers that develop an ECU.

In connection with the source code of each component and your application, CANbedded can be compiled and linked (see Figure 3-2).



The standard generation process for Vectors Software Components.

CANdesc is a completely generated Software Component.

Figure 3-2 Generation Process For Vector CANbedded Software Components

3.4 What The Component Does

Basic Feature Of XCP

- Automatic Detection of ASAM MC2 Description
- Automatic Detection of Device Parameters (Plug&Play)
- Read/Write Access to Device Memory
- Checksum Calculation of Memory Areas
- Access Protection (Seed&Key)
- Emulation Memory Page Switching
- Synchronous, Time-Triggered and Event-Triggered Measurement Data Acquisition
- Flash and EEPROM Programming

Advanced Features Of XCP

- Block Transfer Mode For Upload, Download And Flashing Similar To ISO-TP
- Acquiring Measurement Data with Time Stamp and Time Synchronization

- Downloading Flash Programming Routines
- Dynamic DAQ Lists
- Cold Start Measuring (Start Measurement at Power-On)

4 XCP – A More Detailed View

You use XCP to “look” into an ECU, to read out any memory location you want to using only the underlying bus system. There is no development environment or emulator necessary. Together with e.g. CANape you can visualize this data in many ways. See some examples in the figure below.

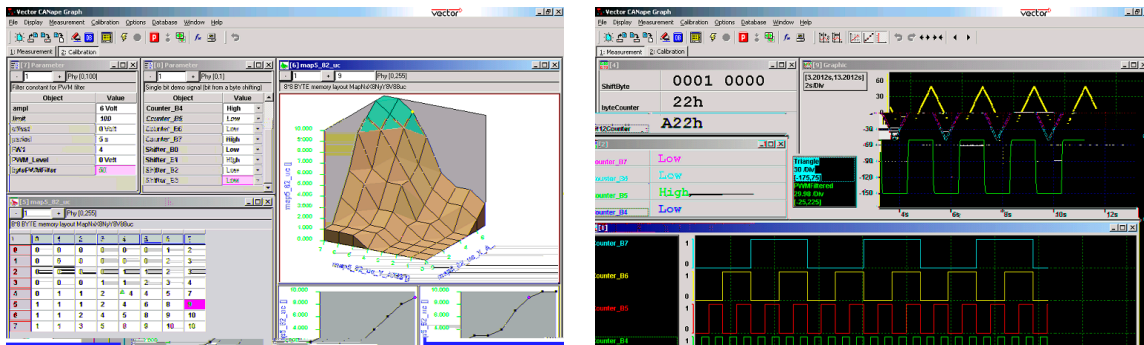


Figure 4-1 Visualization Of Data Using CANape

To get this benefit you need to include the XCP Software Component in your project. As you see in the Figure 3-1 you only need a Driver (in this specific example it is the CAN Driver) to get XCP running.

4.1 Basic Mechanism of the XCP

Basically XCP uses only two messages, a Master and a Slave message. The master message is sent from CANape to the ECU, the slave message from the ECU to CANape. Via these two messages all communication for XCP between CANape and the ECU is handled.

4.2 Measurement Modes

There are mainly two different measurement modes, the polling mode and an event triggered mode (also called Data Acquisition Mode). The main difference between both is the bus load.

4.2.1 Using XCP In Polling Mode

The Polling Mode is the easiest way to use the XCP Driver and needs no further adaptation in your application except for the initialization of the XCP Software Component.

The mechanism is very simple. CANape requests information via a request message (Master Id). The Precopy Function for this message (more about precopy mechanism see UserManual_CANDriver) calls the function XcpCommand to interpret the content of the XCP message. Then XCP initiates the transmission of the response message (Slave ID) containing the requested values.

The XCP Software Component is independent of all other CANbedded Software Components except for the driver.

XCP Polling Mode

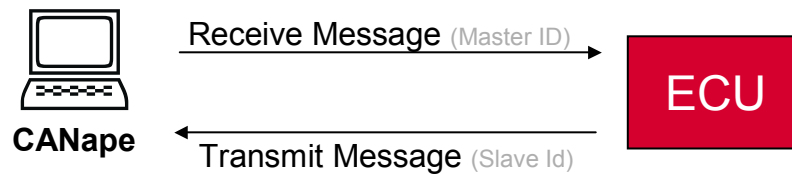


Figure 4-2 XCP Polling Mode Using Two Messages

As you see there are two messages necessary to get the values updated.

4.2.2 Using XCP For Data Acquisition / Event

Another way to get information from the ECU is to let the ECU send this information on its own. This is the Data Acquisition mechanism (Event).

In the Generation Tool you can set the different timings you want to get the data updated. The application has to call the functions **XcpEvent(channel)** with the same call cycle as you did the setting in the Generation Tool. [More about setting the events...](#)

In CANape you can assign this call cycle to a set of signals. This signal list (DAQ list) is sent to the ECU and triggers the cyclic sending of all signals of the list.

XCP Event Mode

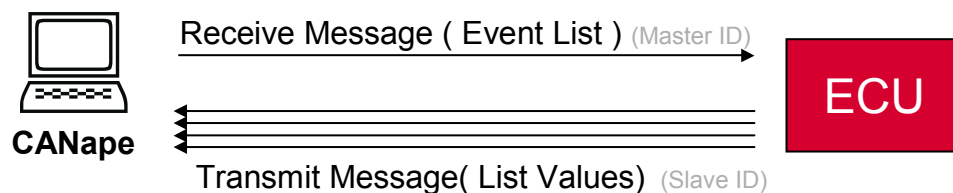


Figure 4-3 XCP Event Mode

In this case the Receive Message is only necessary to initiate the DAQ mechanism. Once initiated there is only one message necessary to get the values updated.

The DAQ list in the ECU needs RAM memory.

You can define the amount of memory that should be reserved for this list in the Generation Tool.

5 XCP IN 8 STEPS

STEP 1 : **INSTALLATION OF TOOL**

Install the PC Generation Tool by following the online instructions.

STEP 2: **CANBEDDED SOFTWARE COMPONENTS**

Extract the C and H files from your CANbedded Software Component delivery into your application software structure.

STEP 3: **CONFIGURATION WITH GENERATION TOOL**

Do the settings in the Generation Tool, especially the settings for XCP are described in detail.

STEP 4: **GENERATE FILES**

Generate the files in the appropriate folders.

STEP 5: **ADD CANBEDDED TO YOUR PROJECT**

Add the CANbedded C and H files to your project or makefile.

STEP 6: **ADAPT YOUR APPLICATION FILES**

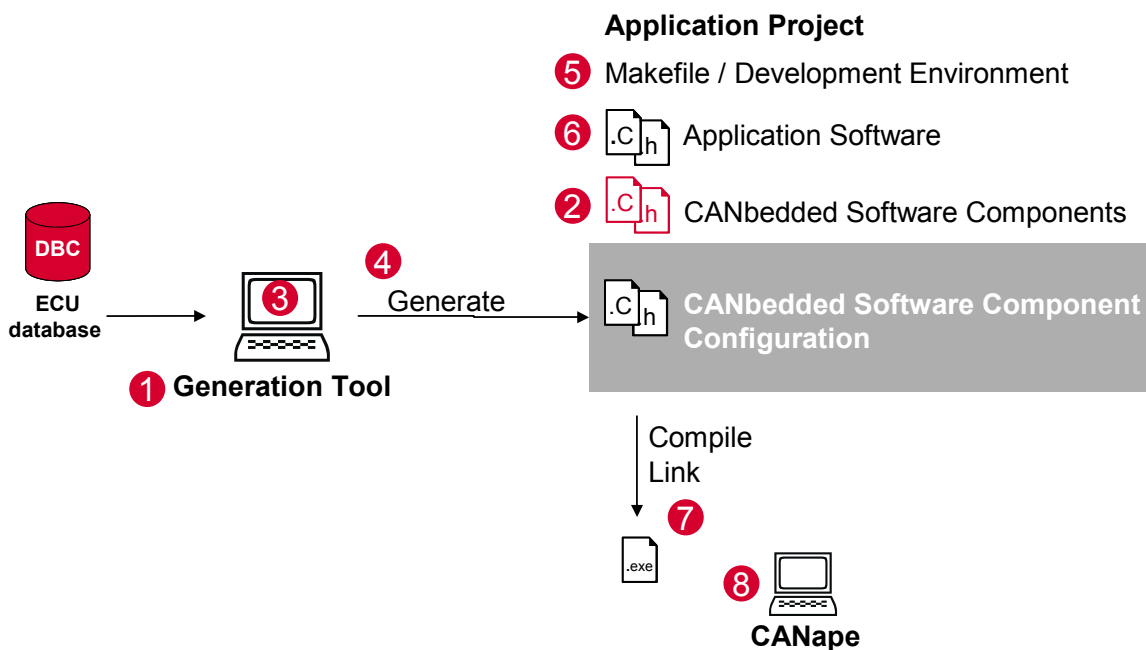
Now your application files must be modified to use the CANbedded Software Components (includes, cyclic calls, initialization) and do the call back functions.

STEP 7: **COMPILE AND LINK YOUR PROJECT**

Compile and link the complete project and download it to your test hardware or development environment.

STEP 8: **TEST IT VIA CANAPE**

Via a appropriate Tester send diagnostic request messages and verify the response messages.



See the steps again in a more visual way.

Figure 5-1 8 Steps to CANdesc



Install the PC
Tool.

5.1 STEP 1 Installation Of The Tool

The Generation Tool is delivered together with the CANbedded Software Components.

Extract the files to an appropriate folder and follow the installation instructions.



Generation Tool

**Included with CANbedded
Software Component Delivery**

[Back](#) to 8 Steps overview

5.2 STEP 2 Extract CANbedded Software Components

The amount of CANbedded Components in your delivery depends on your project.

Copy all C and H files that are necessary for the components into your application project folder.

To use XCP it is only necessary to have a Driver. In this case we use CAN as bus protocol, so we need a CAN Driver.

Refer to the corresponding UserManuals (e.g. CANDriver UserManual) to get further information about the files of the different Software Components.

[Back](#) to 8 Steps overview





5.3 STEP 3 Configuration With The Generation Tool (GENy)

To use XCP we only need the CAN Driver and the XCP Software Component. Switch these components on via Component Selection window.

It is very easy to get XCP running and you only have to do a few settings in the Generation Tool.

The following settings only allow the **polling mode of the XCP**.

Select XCP in the components' list.

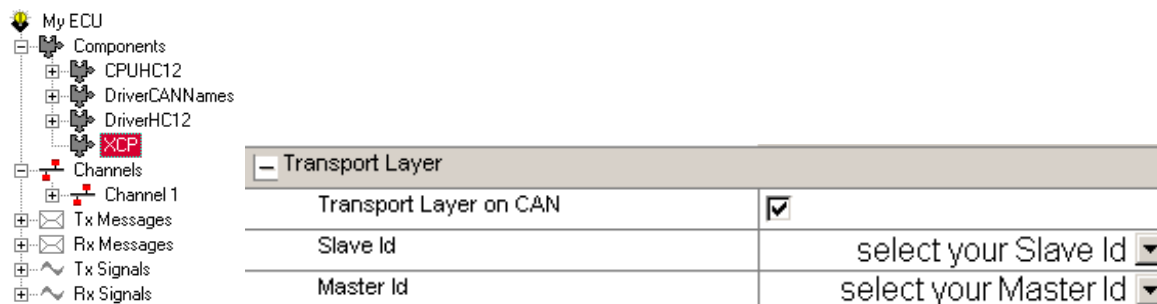


Figure 5-2 Configuration In The Generation Tool For Basic XCP Usage

Make sure that at least these three settings are done. The Transport Layer on CAN setting must be checked and the two Ids selected. Use the pull down menu to find the appropriate messages.


Additionally make sure that the following settings are done correctly:

- Baud rate
- Acceptance filters
- Register Settings for the CAN Controller
- Path for the generated files

[Back](#) to 8 Steps overview



5.4 STEP 4 Generate Files

If you have done the setting in the previous step, hit the **[Generate System]**  button.

These are the fix files that form the xcp:

XcpProf.c

XcpProf.h

The following XCP files are generated:

Xcp_cfg.h

Xcp_par.h

Xcp_par.c

These files are the specific adaptation to the CAN bus and must therefore only be used if the protocol is CAN.

Xcp_can.c

Xcp_can.h

In the window Generated Files you can check the file that the Generation Tool generates and the path where these files are generated in.

[Back](#) to 8 Steps overview



5.5 STEP 5 Add CANbedded To Your Project

What to do in this step depends on your development environment. Perhaps you work with a makefile?

Regardless this you have to add the files of CANbedded to your project. These are the files of chapter 5.2 and the generated ones to the previous step.

Always make sure that the path you generate the files in and the path your compiler is working on are the same!

At this point in time you are not able to compile and link the project. The files should be complete but there are several adaptations for you to do in your application.

Go on with the next step.

[Back](#) to 8 Steps overview.



Include, initialize and call the components cyclically.

Then connect CANdesc with your application.

5.6 STEP 6 Adapt Your Application Files

Now all files for XCP are included in your project and we can go on to do the necessary adaptations in your application files.

These adaptations can be split in two categories:

- Include, initialize and do the cyclic calls for the CANbedded Software Components.
- Connect XCP to your application.

5.6.1 Including, Initialization And Cyclic Calls

As for all other CANbedded Components the XCP must be included and initialized. To use XCP and its API you have to include the file XcpProf.h.

```
Include XcpProf.h;
```

To initialize XCP use this function call:

```
XcpInit( ); /*Interrupts must be disabled*/
```

Make sure that XcpInit is called after the call of CanInitPowerOn.

Normally the components are initialized from bottom up according to the layer model (see Figure 3-1). Always do these initializations with disabled interrupts.

This is the correct order of initialization if you use CAN Driver and XCP:

```
CanInitPowerOn();  
XcpInit();
```

5.6.2 Connect your application to the XCP

There is nothing more to do for the usage of XCP Software Component if you only want to use the polling mode (see 4.2.1).

For the Data Acquisition Mode (see 4.2.2) there are a few additional settings to be done. [more...](#)

5.7 STEP 7 Compile And Link Your Project

Now we have all includes, all initializations, the components do have the cyclic calls of their task functions and all call-back functions are provided and programmed.

Start the compiler or makefile and get the project compiled and linked.

Is it ok? No errors?

Gratulations, that's it.

Go on to the next step and do the testing.

[Back](#) to 8 Steps overview.



5.8 STEP 8 Test It Via CANape

You arrived at this step, so you are able to compile and link. Do you already have downloaded the code to your target platform?

Working with XCP most of the time you will spend doing the settings in CANape. Start CANape.

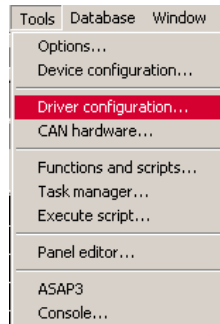


Figure 5-3 Set Driver Configuration In CANape

Via Tools/Driver configuration in CANape you get to the window below.

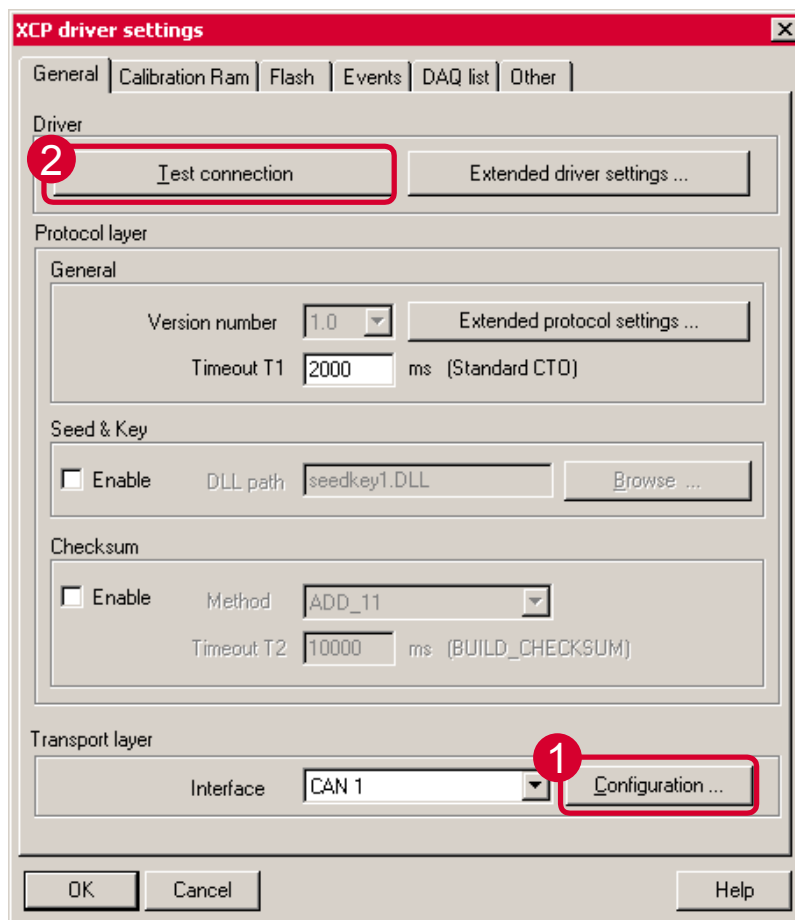


Figure 5-4 XCP Driver Settings

Before testing the connection you have to do the **[Configuration ...]** for the CAN (1). There you have to set the two messages to communicate with the ECU.

CAN_ID_MASTER and CAN_ID_SLAVE as shown in the Figure 4-2.

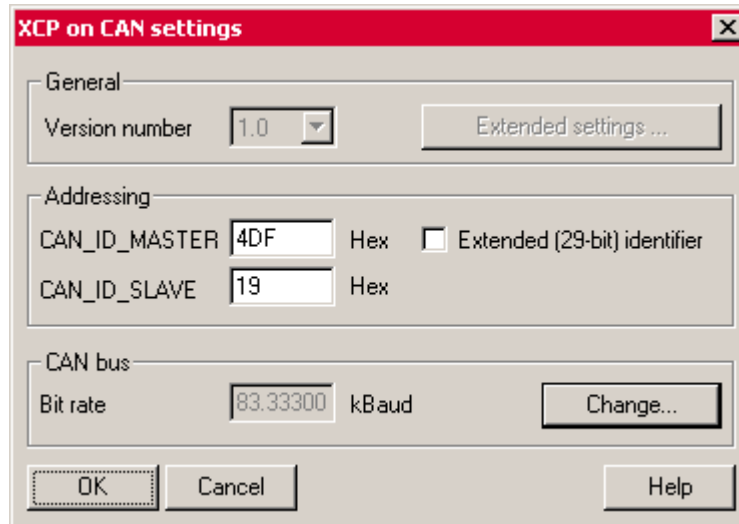


Figure 5-5 Settings For XCP On CAN

A very important setting is the baud rate. Use **[Change]** to adapt this value to your needs.

Now it is the time to test the connection (2).

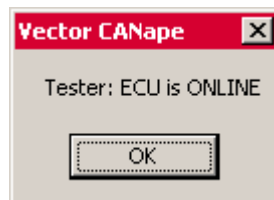


Figure 5-6 ECU Is Online

The communication between CANape and your ECU is now working correctly. The next step would be to select the variables or memory locations you want to get shown in CANape. How to do this is explained in details in the Manual of the CANape.

6 Further Information

6.1 Settings For Using The Data Acquisition Mode / Events

As mentioned in 4.2 and 4.2.2 there is another mode besides the simple to handle polling mode – the Data Acquisition Mode with the benefit of less bus load as already described.

To realize this mode you have to do additional settings in the Generation Tool and in your application. This can be divided into 3 steps:

- 1 Switch on Synchronous Data Acquisition and DAQ generation info and the Event Info

[-] Synchronous Data Acquisition (DAQ)	
Synchronous Data Acquisition (DAQ)	<input checked="" type="checkbox"/>
Memory Size	512
Prescaler	<input type="checkbox"/>
Overrun indication	<input type="checkbox"/>
DAQ/ODT message header	<input type="checkbox"/>
DAQ resume mode	<input type="checkbox"/>
DAQ general info	<input checked="" type="checkbox"/>
DAQ resolution info	<input type="checkbox"/>
[-] Events	
Event Info	<input checked="" type="checkbox"/>

Figure 6-1 Components / XCP

- 2 Define Events in the Generation Tool (Name, Cycle)

[-] Events		Allocate Items:
[-] Channel		<input checked="" type="checkbox"/> Item 1
No.		<input checked="" type="checkbox"/> Item 2
Name		<input type="checkbox"/> Item 3
Cycle		<input type="checkbox"/> Item 4
Direction		<input type="checkbox"/> Item 5
		<input type="checkbox"/> Item 6
		<input type="checkbox"/> Item 7

Figure 6-2 Components / XCP

The first event with the No. 0 is predefined. To add further events use the Allocate Items pull down menu and check the next Item.

Make sure that you check the next unused item. E.g. Item 1, Item 3 and Item 4 is **not allowed**, because Item 2 is omitted.

Events		Allocate Items:
Channel		
No.	0	
Name	My10msEvent	
Cycle	10	
Direction	DAQ	
Channel		
No.	1	
Name	My20msEvent	
Cycle	20	
Direction	DAQ	

Figure 6-3 Components / XCP

For each Event (numbering starting with 0) you should define the Name and the Cycle [ms].

Try to make this name expressive. This name will occur in the CANape list. It is much easier for you if the name contains e.g. the cycle.

Remember to generate the files before compiling again !

③ For each event call XcpEvent(x), x= 0, 1, ..., n.

Now open your application file and add the calls of the function XcpEvent(x). For each event you need one call of XcpEvent(x). The x is the No. of the event as it is defined in the Generation Tool.

Make sure that you call the XcpEvent(x) function in the correct cycle time. In the example we need 2 calls. XcpEvent(0) after any **10ms** and XcpEvent(1) after any **20ms**.

Now compile and link you project and transfer it to your target platform and start your application.

④ Assemble your DAQ lists in CANape

Open CANape and start it and stop it. We need this to transfer the DAQ information from the ECU to CANape.

Now open Edit Measurement List via this button on the right side. You see all selected signals in this window (Figure 6-4). In the pull down menu of the Measuring mode you now find the two events My10msEvent and My20msEvent.

Assign these Events to any signal you want. The signals will be added to the DAQ list and transferred any 10ms or 20ms to CANape without any further polling message.



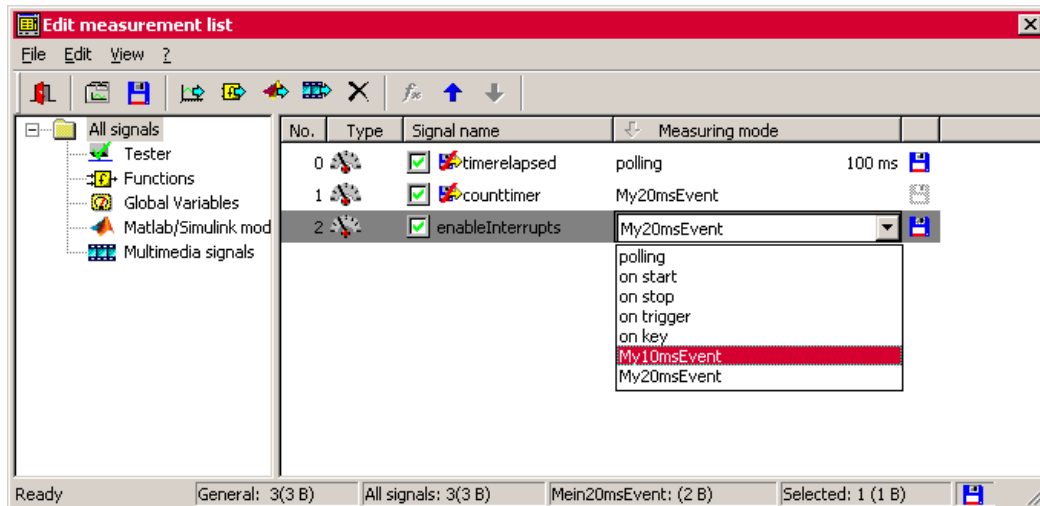


Figure 6-4 Edit Measurement List In CANape

7 List Of Experiences

A list of experiences and problems encountered will be placed here soon, which will help you to conduct focused troubleshooting.

7.1 Topic 1

Q:

A: ■

8 Index

8 STEPS	13	Initialization	18
Advanced Features	9	makefile	17
application	18	Measurement Modes	11
Basic Feature	9	Motivation	2
CAN	8	Polling Mode	11
CANape	8, 11, 23	SCI	8
CanInitPowerOn	18	STEP 1 Installation Of The Tool	14
CCP	8	STEP 2 Extract CANbedded Software Components	14
compiler	17	STEP 3 Configuration With Generation Tool (GENy)	15
Data Acquisition	12	STEP 4 Generate Files	16
Data Acquisition Mode	22	STEP 5 Add CANbedded To Your Project	17
dbc-file	9	STEP 6 Adapt Your Application Files	18
DescInitPowerOn	18	STEP 7 Compile And Link Your Project	19
development environment	17	STEP 8 Test It Via CANape	20
Ethernet	8	TTCAN	8
Event	12, 22	Universal Measurement And Calibration Protocol XCP	7
Firewire	8	USB	8
FlexRay	8	XCP	8
Further Information	22		
Generation Tool	8		
Including, Initialization And Cyclic Calls .	18		