ETAS Access Devices

Operation as ASAM XCP on Ethernet Device V1.0

Data Sheet

www.etas.com

ETAS V1.0 | Datasheet EN | 02.2024

Contents

Figures

1 Introduction

1.1 Definitions and Abbreviations

AML

ASAM-MCD 2MC (aka ASAP2) Meta Language

A2L

ASAM-MCD 2MC (aka ASAP2) Description Language

DISTAB

Display Table is the method used for Data Acquisition

ECU

Electronic Control Unit

ERAM

Emulation Random Access Memory

ETK Tool

Configuration Toolset used to configure the ETK, FETK and XETK devices

RP

Reference Page

UML

Unified Modelling Language by OMG (Object-Modelling-Group). Graphical Modelling Language for SW Engineering/Development

WP

Working Page

XCP

Extended Calibration Protocol **ASAM-MCD 1 (XCP)**

XCPoE

Extended Calibration Protocol over Ethernet

XCT

XETK Configuration Tool used to configure the (some ETKs), FETK and XETK devices

1.2 Conventions

The following typographical conventions are used in this document:

`OCI_CANTxMessage msg0 =`	Code snippets are presented on a gray background and in the Courier font. Meaning and usage of each command are explained by means of comments. The comments are enclosed by the usual syntax for comments. Changes are red marked.
Choose **File** → **Open**.	Menu commands are shown in boldface.
Click **OK**.	Buttons are shown in boldface.
Press <ENTER>.	Keyboard commands are shown in angled brackets.
The "Open File" dialog box is displayed.	Names of program windows, dialog boxes, fields, etc. are shown in quotation marks.
Select the file `setup.exe`	Text in drop-down lists on the screen, program code, as well as path- and file names are shown in the Courier font.
A *distribution* is always a one-dimensional table of sample points.	General emphasis and new terms are set in italics.

2 ETAS ETK Access Devices

The ETAS ETK Access Devices is an emulator probe family developed for the most common microcontroller families in the automotive sector (Infineon, Freescale/NXP, ST Microelectronics, Renesas; to name only a few).

The ETK family comprises ETKs, XETKs and FETKs modules. Each of these modules offers a different set of functionality and performance.

To access the ECU the ETK has to be connected directly to the host or indirectly to additional interface hardware: the ESxxx devices. The system can be used for high speed Measurement, Calibration and ECU flash programming. Support of Rapid Prototyping applications e.g. functional proto-typing - bypass depends on the functionality of the connected modules.

The XETK and the FETK modules use the open automotive "Universal Measurement and Calibration" standard "XCP on Ethernet" (TCP/IP, UDP/IP) protocol for the PC communication. The open XCP on Ethernet (XCPoE) interface allows connectiions to the XETKs and the FETKs with third party application software.

The XETK and the FETK modules can operate as **XETK** or **FETK** devices as a **NON-ASAM conform XCP-Device** (only with the ETAS INCA MCD Application) or as a **ASAM-MCD 1 conform XCP-Device** (with INCA or with a third party application software).
The necessary module specific configuration for the **XETKs and FETKs** for measurement and calibration are dynamically and individually handled by INCA depending on the contents of the A2L (**ASAM-MCD 2MC**) file and the experiment configuration. As the **ASAM conform XCP-Device** it is **NOT** possible to let INCA handle that individual configuration and dynamic initialization to download the desired device configuration to the XETK and FETK modules before it can be used with **ETAS INCA** or with a third party tool of your choice.

How this mandatory configuration is being done without the usage of ETAS INCA will be deeply explained in the following chapters. The application/tool which MUST be used for this task is called the ETAS XCT (xETK Configuration Tool).

Note:

Currently there are several ASAM XCP Version available:

From Version 1.0 (Year 2003) to 1.5 (Year 2017).

In this document not all details between the different versions are shown but mentioned where it is inevitable.

2.1 Overview of the abstraction layers ASAM-MCD 1 – ASAM-MCD 3

There were already some ASAM-MCD standards mentioned. To give a short insight of how these standards interact with each other, the Figure 1: Logical Component Model of the ASAM-MCD Abstraction Layers shows an overview of a system where ETAS Access Devices (ETK family) and/or ETAS Interface Devices (ESxxx) could be used for.

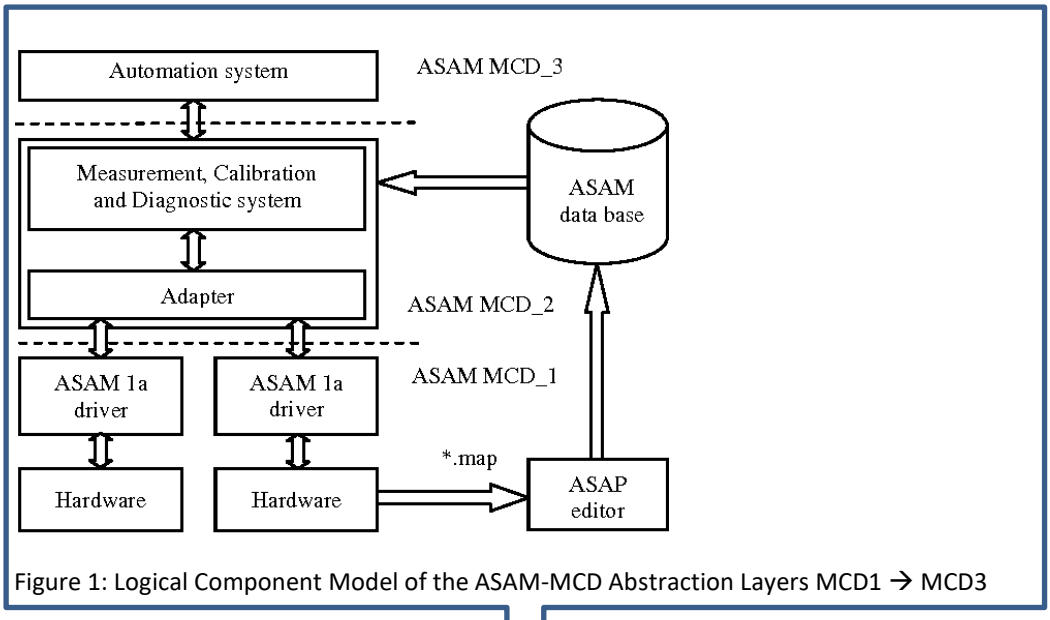


Figure 1: Logical Component Model of the ASAM-MCD Abstraction Layers MCD1 → MCD3

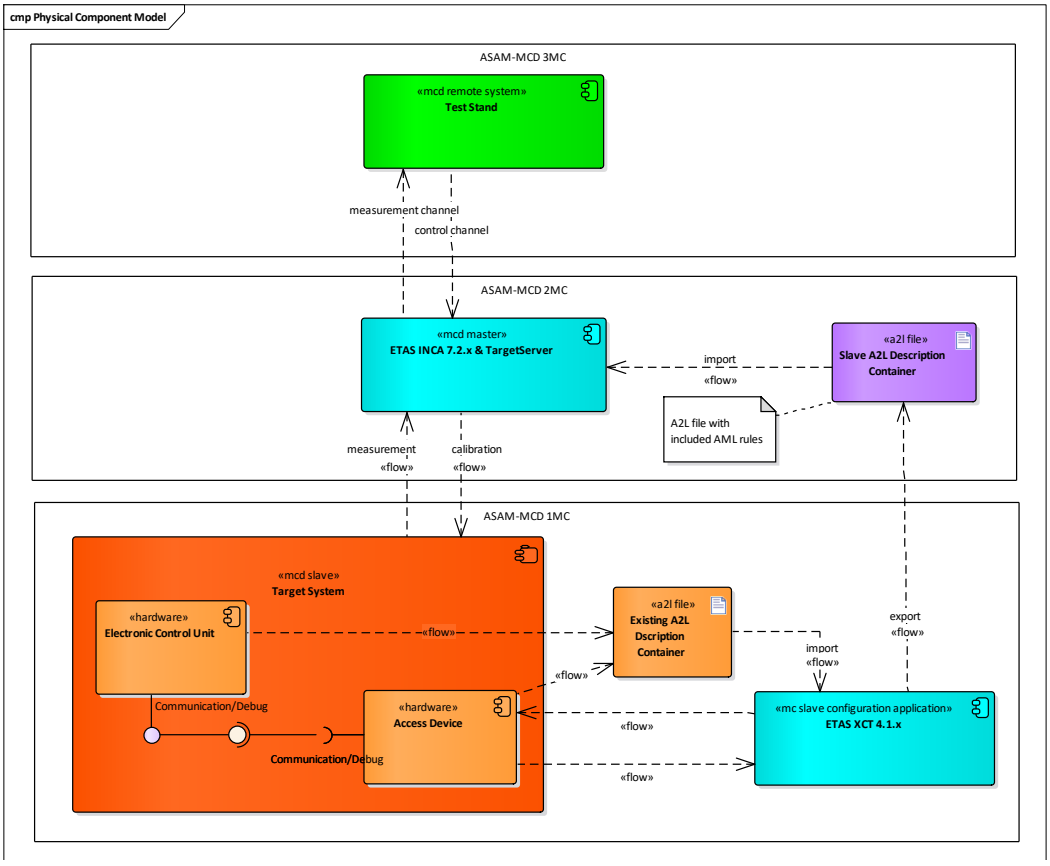


Figure 2: Physical Component Model of the ASAM-MCD Abstraction Layers MCD1 → MCD3

3 Configuration as a XCP on Ethernet Device

A Configuration is the virtual representation of all settings of the Access Device. It is always assigned to one specific device type.
An Access Device configuration is valid, if it works together with the ECU properly. Means a tool based initialization is being correctly done regarding the:

- **Syntactical** (the used configuration is suitable with the device functionality and doesn't use any config settings which are not supported by that attached device) and
- **Semantical** configuration (the values of the settings are within the expected range and the slave can measure and calibrate in its expected way)

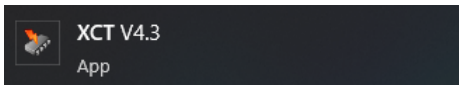
For generating, storing and applying Access Device configurations a tool other than **INCA** has to be used. That tool is called **XCT.** XCT is only being used as a configuration tool. (There are no functions to execute measurement and calibration tasks)

The **XCT** tool has the ability to configure ETAS Access Devices (ETK, XETK and FETK modules) like **INCA** is configuring the Access Device. Further it can also be used to configure the Access Device as a **ASAM XCP** slave.

How to execute the step for a valid **ASAM XCP** configuration in detail will explained from now on.

3.1 Step 1: Where it begins: XCT Project

Search for your XCT installation location on your Windows machine and locate the XCT icon:



Start the XCT tool. The initial screen of the XCT tool should look like this:

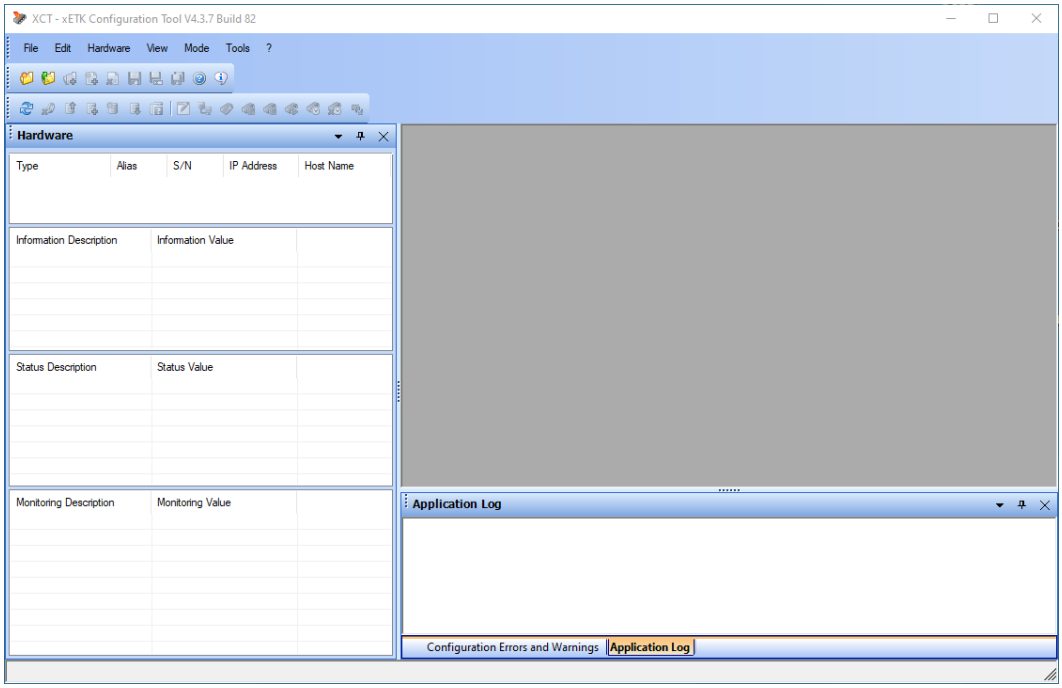


Figure 3: Initial Screen of the XCT tool

Make sure you are using the current released version of the tool. You can see it on the title bar, as depicted in Figure 3 (in this example V4.3.7 Build 82). Compare that version with the current released number of the XCT tool.

Note:

It is advised to use always the newest XCT tool regarding new features and stability updates!

3.2 Step 2: Creating a new Project by Importing an existing A2L Description Container

In order to have all your changes saved and therefore persistent, it is necessary to setup a project. In that project all the settings which are created, modified or even deleted are stored persistently.

If you recognized that your configuration is semantically incorrect resp. invalid, the project can be re-opened and necessary changes can be made until your Access Device configuration is valid.

Usually when a XCP project will be setup, there is already a valid configuration available resp. existing. That configuration was previously used with INCA or some other MCD tool. And from now on both tools (INCA and the other MCD tool) should be used at the same time with the same or different hardware or also with some other 3rd party hardware.

To satisfy that requirement the XCT tool offers the possibility to import an A2L file to use already configured settings. If they are compatible with INCA then XCT can import those specific sections. If there are non-ETAS related sections in that file, XCT will ignore those sections/elements.

The next possible use case is to create a project from scratch. Then it is necessary that all necessary information to configure your project is known.

Such as:

- Hardware Settings (JTAG, DAP frequency), Page Switch Method, etc.
- Microcontroller Type (plausibility check regarding the configured ECU memory segments in the Memory Layout Editor)
- Layout (Start Address and Size) of the ECU Software segments, the type of that memory segment (code, data, variables, etc.)
- Calibration Handles which should be used for overly, what data page should be overlayed, which size the overlay area should have, etc.
- Which Rasters/Events your ECU provides to have the performance of you Measurement/Calibration optimally setup up
- If they are compatible with INCA then XCT can import those specific sections. If there are non-ETAS

When an existing A2L container is being imported, a project is implicitly created. Therefore, there is no need to create a project explicitly.

Note:

In this document only the use case of an existing A2L description container will be considered.

To import an A2L container follow these steps:

- **File → Import A2L File…**

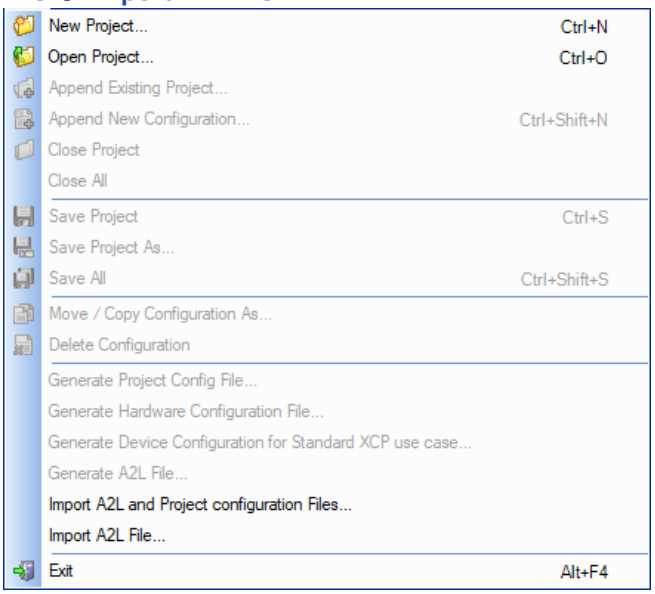


Figure 4: Import an existing A2L File

- If in the A2L container more than one Access Device is defined in order to work with the full functionality of the XETK and FETK devices, the following dialog appears:

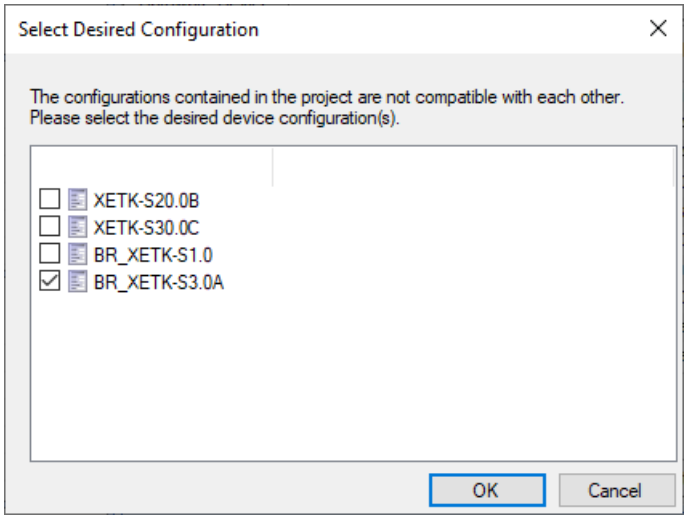
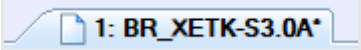


Figure 5: Access Device selection dialog

- Select the device(s) which are relevant for you project and confirm with the OK button (the selection dialog does not appear of only one Access Device is defined in the A2L file).

Note:

Whenever you change any configuration setting, the Configuration Tool displays an asterisk mark (*) besides the name of the configuration. This symbol signals that the present configuration is different to the last saved.



In the Application Log (if not visible select **View → Application** Log in the main menu, or maximize the pane when it is minimized somewhere within in XCT UI) the history and state of done actions can be seen. So if any error occurs during your configuration it is shown in the Application Log.

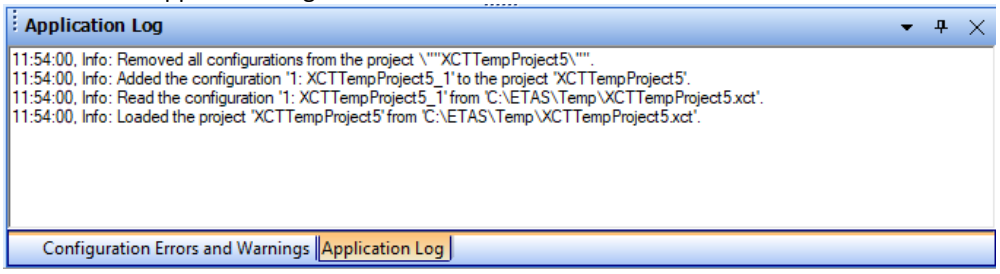


Figure 6: Application Log pane

3.3 Step 3: Setup the Hardware Settings

The settings which can be seen in the Hardware Settings Tab should not be changed, because they are configured to work with the application software running on the ECU.

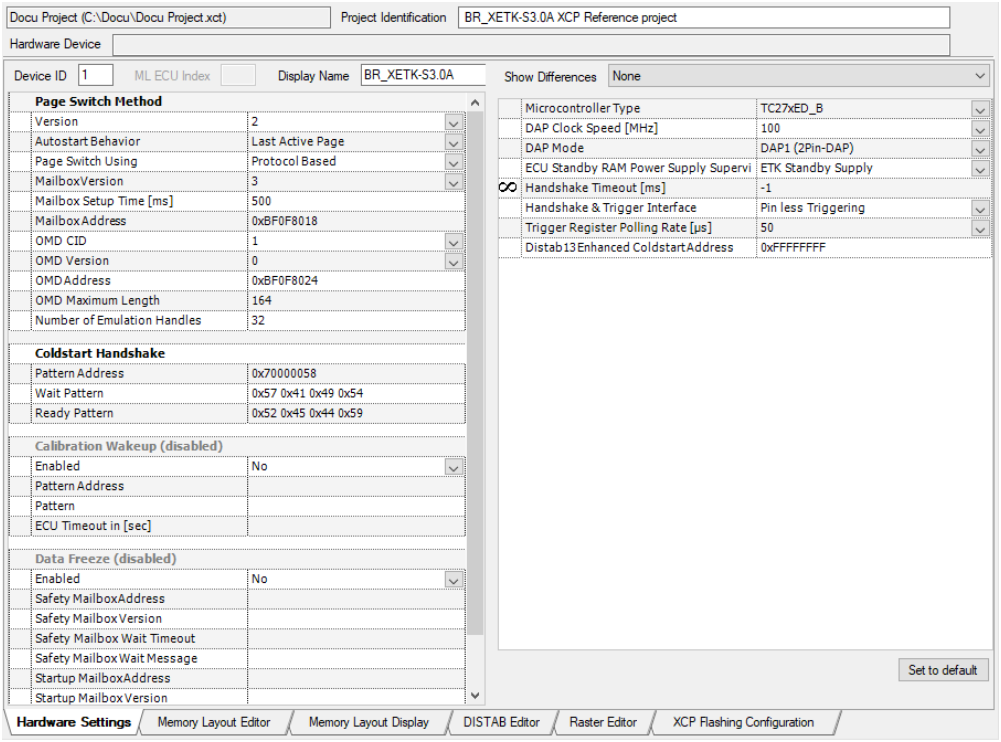


Figure 7: Settings and functions in the Hardware Settings Tab

Note:

Depending on the Access device configuration in the A2 file the Mode can be Standard Mode or Extended Mode, see Figure 8. Depending on the mode more or less Hardware Setting are depicted.

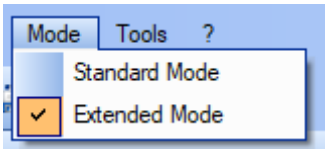


Figure 8: Mode

3.4 Step 4: Automatic conversion of project to fixed configuration

In the configuration tool you can generate the fixed configuration automatically over the menu **Edit → Convert project to fixed configuration**.

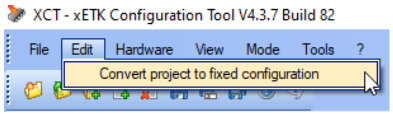


Figure 9: Convert project to fixed configuration

You will be asked if you want to convert the project. This must be confirmed with **Yes**.

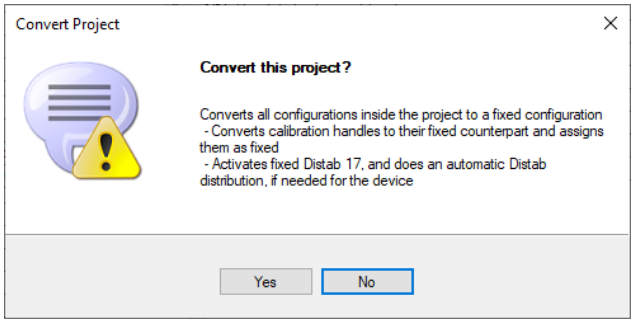


Figure 10: Convert project

When you do that, you can skip step 5 to 7 and continue with step 8 to download the configuration into ETK.

Note:

Depending on the device configuration the conversion to fixed configuration is not always error-free. It is important to check the configuration in the Distab and Raster Editor tabs for inconsistencies. For this read the points Step 5 to 7 !!!

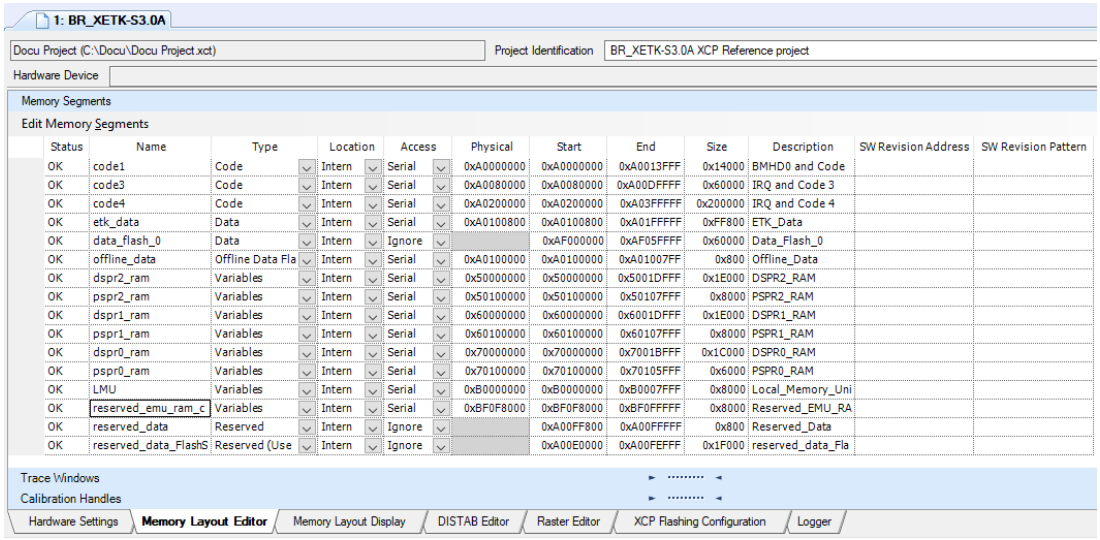
3.5 Step 5: Customize the Memory Segments

Here a valid memory configuration can be seen. The used **MCU** is an Infineon **TC27 MCU**. A single memory segment can be only of one type: data or code or variables or etc. It is not possible to mix different data types within one segment.

Special attention is necessary for the memory segments with the memory type **Data**.

In that memory type the **Data** (**Non-Volatile-Memory**) is located which is emulated respectively overlayed with emulation memory **(ERAM)** in the **MCU**. It is necessary to know how much **ERAM** is available in the MCU for the measurement and calibration task.

When there is less **ERAM** available than **Data** sections in the **ECU** application software, then not all calibration values can be emulated in the **ERAM**.



Status	Name	Type	Location	Access	Physical	Start	End	Size	Description	SW Revision Address	SW Revision Pattern
OK	code1	Code	Intern	Serial	0xA0000000	0xA0000000	0xA0013FFF	0x14000	BMHD0 and Code		
OK	code3	Code	Intern	Serial	0xA0080000	0xA0080000	0xA00DFFFF	0x60000	IRQ and Code 3		
OK	code4	Code	Intern	Serial	0xA0200000	0xA0200000	0xA03FFFFF	0x200000	IRQ and Code 4		
OK	etk_data	Data	Intern	Serial	0xA0100800	0xA0100800	0xA01FFFFF	0xFF800	ETK_Data		
OK	data_flash_0	Data	Intern	Ignore		0xAF000000	0xAF05FFFF	0x60000	Data_Flash_0		
OK	offline_data	Offline Data Fla	Intern	Serial	0xA0100000	0xA0100000	0xA01007FF	0x800	Offline_Data		
OK	dspr2_ram	Variables	Intern	Serial	0x50000000	0x50000000	0x5001DFFF	0x1E000	DSPR2_RAM		
OK	pspr2_ram	Variables	Intern	Serial	0x50100000	0x50100000	0x50107FFF	0x8000	PSPR2_RAM		
OK	dspr1_ram	Variables	Intern	Serial	0x60000000	0x60000000	0x6001DFFF	0x1E000	DSPR1_RAM		
OK	pspr1_ram	Variables	Intern	Serial	0x60100000	0x60100000	0x60107FFF	0x8000	PSPR1_RAM		
OK	dspr0_ram	Variables	Intern	Serial	0x70000000	0x70000000	0x7001BFFF	0x1C000	DSPR0_RAM		
OK	pspr0_ram	Variables	Intern	Serial	0x70100000	0x70100000	0x70105FFF	0x6000	PSPR0_RAM		
OK	LMU	Variables	Intern	Serial	0xB0000000	0xB0000000	0xB0007FFF	0x8000	Local_Memory_Uni		
OK	reserved_emu_ram_c	Variables	Intern	Serial	0xBF0F8000	0xBF0F8000	0xBF0FFFFF	0x8000	Reserved_EMU_RA		
OK	reserved_data	Reserved	Intern	Ignore		0xA00FF800	0xA00FFFFF	0x800	Reserved_Data		
OK	reserved_data_FlashS	Reserved (Use	Intern	Ignore		0xA00E0000	0xA00FEFFF	0x1F000	reserved_data_Fla		

Figure 11: Memory Layout Editor pane; The image shows only an example layout, the data areas here a bigger in size than the available **ERAM** size!

Note:

It is very helpful to have the MCU specification manual of the used MCU derivate close at hand when customizing the memory segments in the Memory Layout Editor.

Semantically wrong settings can lead to situations where the desired memory neither can be accessed nor can be calibrated.

Wrong layout settings for a certain MCU will be depicted as an error and disables the download of that wrong configuration to the access device.

3.6 Step 6: Configuration of the Calibration Handles according to the overlay of the ERAM

3.6.1 Step 6.1: Projects with Calibration Method = Fixed Size

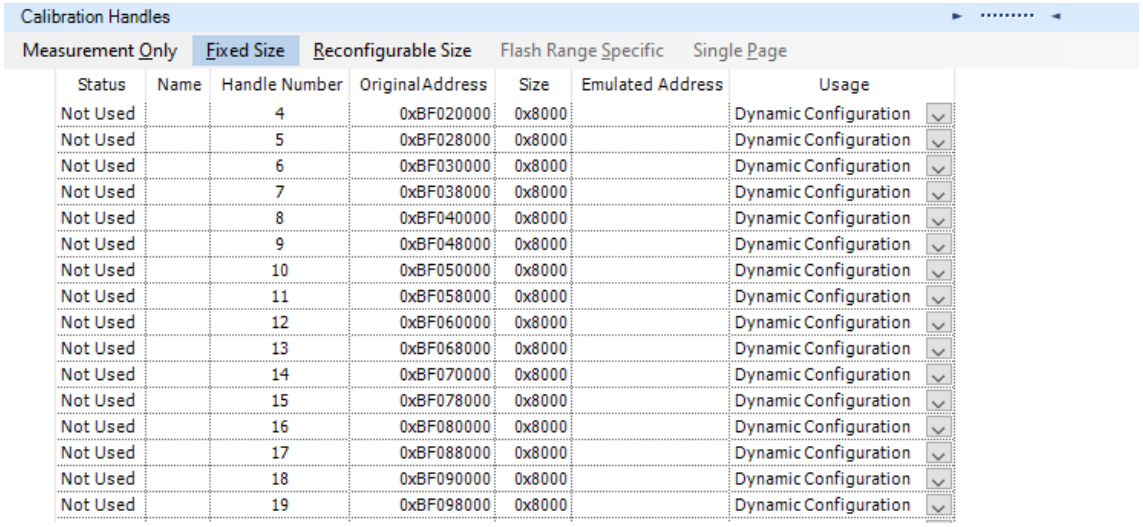
In the **Calibration Handles Editor** can be defined which RAM blocks shall form the working page in the ECU by activating the overlay mechanism. The addresses for the calibration handles can be assigned as **Dynamic Configuration, Fixed Configuration** or **Not Available**.

- **Dynamic Configuration:** The calibration handle shall be managed by the measurement and calibration tool, the emulation address will be assigned by the tool. INCA can manage the calibration handles.
- **Fixed Configuration:** A Fixed emulation address will be used for the calibration handles.
- **Not Available:** The calibration handle is reserved for other purposes and shall not be used for emulation.

In projects with the **Calibration Method = Fixed Size** all handles and the appropriate calibration RAM distribution is defined.

The appropriate `CALIBRATION_METHOD` in the A2L file is "`FixedSizeMoveableEmuRAM`".

After importing the A2L file the calibration handles are displayed without emulation addresses and the usage is set as **Dynamic Configuration**.



Status	Name	Handle Number	Original Address	Size	Emulated Address	Usage
Not Used		4	0xBF020000	0x8000		Dynamic Configuration
Not Used		5	0xBF028000	0x8000		Dynamic Configuration
Not Used		6	0xBF030000	0x8000		Dynamic Configuration
Not Used		7	0xBF038000	0x8000		Dynamic Configuration
Not Used		8	0xBF040000	0x8000		Dynamic Configuration
Not Used		9	0xBF048000	0x8000		Dynamic Configuration
Not Used		10	0xBF050000	0x8000		Dynamic Configuration
Not Used		11	0xBF058000	0x8000		Dynamic Configuration
Not Used		12	0xBF060000	0x8000		Dynamic Configuration
Not Used		13	0xBF068000	0x8000		Dynamic Configuration
Not Used		14	0xBF070000	0x8000		Dynamic Configuration
Not Used		15	0xBF078000	0x8000		Dynamic Configuration
Not Used		16	0xBF080000	0x8000		Dynamic Configuration
Not Used		17	0xBF088000	0x8000		Dynamic Configuration
Not Used		18	0xBF090000	0x8000		Dynamic Configuration
Not Used		19	0xBF098000	0x8000		Dynamic Configuration

Figure 12: Calibration Handles pane in the Memory Layout Editor

Note:

Because the third party measurement and calibration tools cannot manage the calibration handles dynamically, it is needed to assign the emulation addresses and to change the usage of the handles as **Fixed Configuration**.

Note:

There is no need to assign all available Calibration Handles. Only those Calibration Handles which are really needed should be assigned.

Because the Calibration Handles are not visible to the MCD tool and therefore no **MEMORY_SEGMENTs** are neither needed nor **CALIBRATION_HANDLEs** in the A2L container.

In the configuration tool you can enter the emulation address and the usage for each handle manually or you can select “**Assign all Handles (Fixed Configuration)”** from the context menu:

- Move the cursor to the Calibration Handles Editor.
- Click on the button **Fixed Size**.
- Select “**Assign all Handles (Fixed Configuration)”**

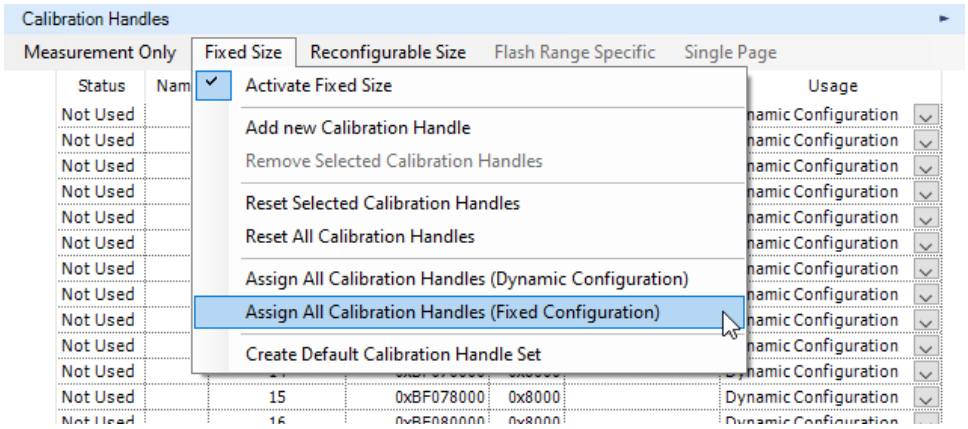
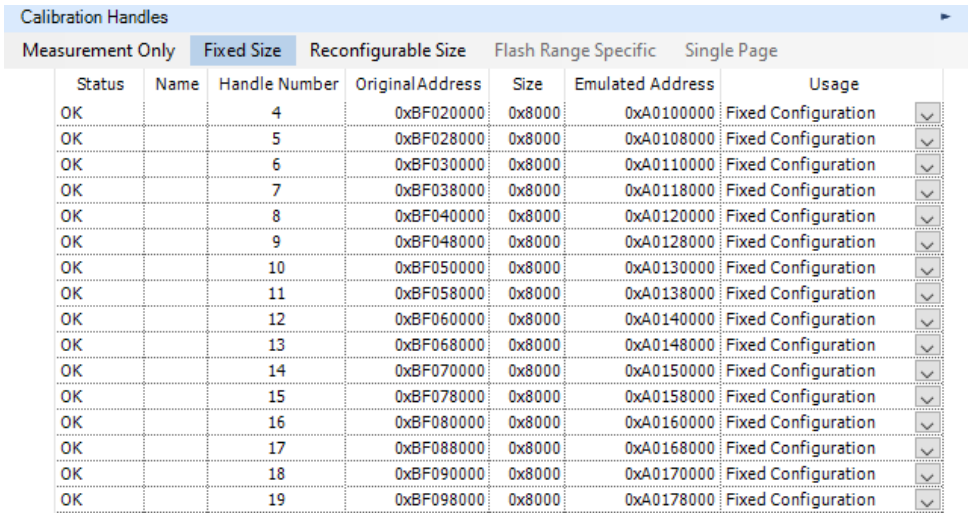


Figure 13: Assignment dialog in the Calibration Handles pane



Status	Name	Handle Number	Original Address	Size	Emulated Address	Usage
OK		4	0xBF020000	0x8000	0xA0100000	Fixed Configuration
OK		5	0xBF028000	0x8000	0xA0108000	Fixed Configuration
OK		6	0xBF030000	0x8000	0xA0110000	Fixed Configuration
OK		7	0xBF038000	0x8000	0xA0118000	Fixed Configuration
OK		8	0xBF040000	0x8000	0xA0120000	Fixed Configuration
OK		9	0xBF048000	0x8000	0xA0128000	Fixed Configuration
OK		10	0xBF050000	0x8000	0xA0130000	Fixed Configuration
OK		11	0xBF058000	0x8000	0xA0138000	Fixed Configuration
OK		12	0xBF060000	0x8000	0xA0140000	Fixed Configuration
OK		13	0xBF068000	0x8000	0xA0148000	Fixed Configuration
OK		14	0xBF070000	0x8000	0xA0150000	Fixed Configuration
OK		15	0xBF078000	0x8000	0xA0158000	Fixed Configuration
OK		16	0xBF080000	0x8000	0xA0160000	Fixed Configuration
OK		17	0xBF088000	0x8000	0xA0168000	Fixed Configuration
OK		18	0xBF090000	0x8000	0xA0170000	Fixed Configuration
OK		19	0xBF098000	0x8000	0xA0178000	Fixed Configuration

Figure 14: Fixed Configuration appearance in the Calibration Handles pane

Note:

When you select the **Fixed Configuration** assignment a default emulation address is entered; if needed replace this default address by an appropriate emulation address. The emulation address must be located in a memory segment of the type **Data**.

Note:

The usage of "**Create Default Handle Set**" from the context menu of the Calibration Handles Editor, should only be used with care!

Some Access Devices are using certain Calibration Handles for the **TRACE** usage. By creating a default handle set, these Handles will be overwritten and so the **TRACE** measurements won't work anymore!

The **TRACE** feature is dependent on the used Access Device and on the configured Project respectively **A2L** configuration.

The XCT considers this and raise a warning. In this case, delete the appropriate calibration handles or set them to Not Available.

3.6.2 Step 6.2: Projects with Calibration Method = Reconfigurable Size

In projects with the **Calibration Method = Reconfigurable Size** only the available number of handles and the complete calibration RAM is defined. INCA calculate the handle numbers and sizes depending on the need.

Figure 15: Reconfigurable Size appearance in the Calibration Handles pane

The appropriate `CALIBRATION_METHOD` in the A2L file is "`ReconfigurableSizeMoveableEmuRAM`".

In order to calibrate with third party measurement and calibration tools, the available calibration RAM and handles need to be distributed and set to **Fixed Configuration**, similar to Step 6.1: Projects with Calibration Method = Fixed Size.

Step A:

In order to distribute the available handles and calibration RAM to dedicated handles use the button **Reconfigurable Size** and select "**Convert EMU Ranges to Fixed Size**":

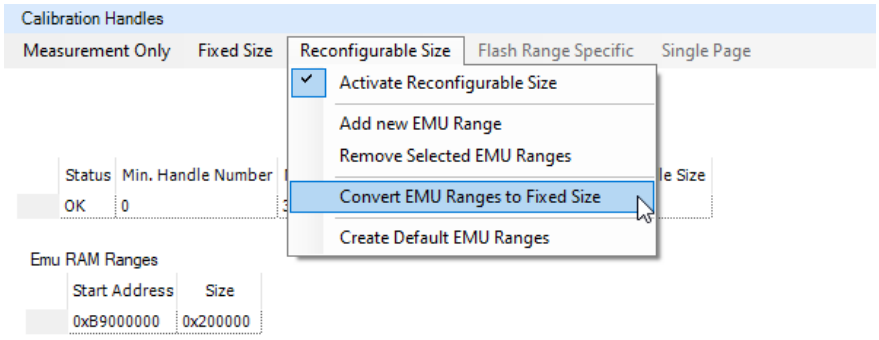
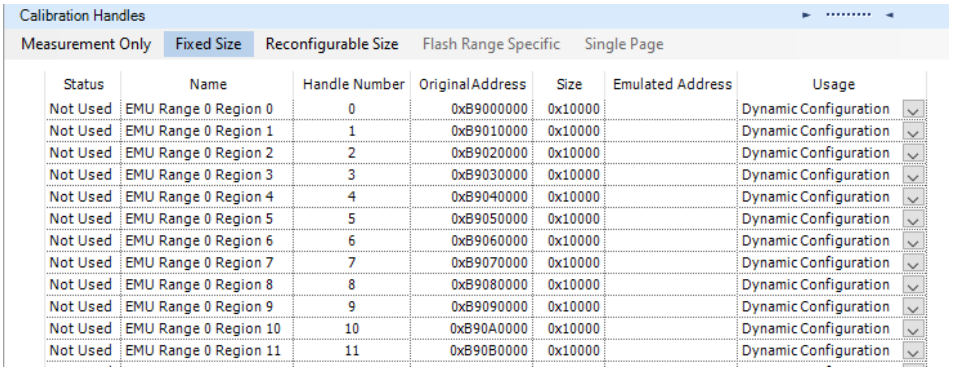


Figure 16: Context menu of configurable size Emu RAM ranges

After acknowledge of the warning the handles and calibration RAM are distributed, but still used for Dynamic Configuration.


Calibration Handles

Measurement Only | Fixed Size | Reconfigurable Size | Flash Range Specific | Single Page

Status	Name	Handle Number	Original Address	Size	Emulated Address	Usage
Not Used	EMU Range 0 Region 0	0	0xB9000000	0x10000		Dynamic Configuration
Not Used	EMU Range 0 Region 1	1	0xB9010000	0x10000		Dynamic Configuration
Not Used	EMU Range 0 Region 2	2	0xB9020000	0x10000		Dynamic Configuration
Not Used	EMU Range 0 Region 3	3	0xB9030000	0x10000		Dynamic Configuration
Not Used	EMU Range 0 Region 4	4	0xB9040000	0x10000		Dynamic Configuration
Not Used	EMU Range 0 Region 5	5	0xB9050000	0x10000		Dynamic Configuration
Not Used	EMU Range 0 Region 6	6	0xB9060000	0x10000		Dynamic Configuration
Not Used	EMU Range 0 Region 7	7	0xB9070000	0x10000		Dynamic Configuration
Not Used	EMU Range 0 Region 8	8	0xB9080000	0x10000		Dynamic Configuration
Not Used	EMU Range 0 Region 9	9	0xB9090000	0x10000		Dynamic Configuration
Not Used	EMU Range 0 Region 10	10	0xB90A0000	0x10000		Dynamic Configuration
Not Used	EMU Range 0 Region 11	11	0xB90B0000	0x10000		Dynamic Configuration

Figure 17: Automatic distributed Emu RAM ranges

Step B:

In order to set then to **Fixed Configuration**, follow the step as described in Step 6.1: Projects with Calibration Method = Fixed Size.

3.7 Step 7: Raster Settings

The used Rasters are to be defined in the **Raster Editor**, the editor is divided into different separate parts:

- **Trace Trigger Type**: Defines how the triggers are released if data is collected via trace windows
- **Raster Overview**: The Raster Overview is displayed in the left half of the editor window. After importing the A2L file all raster that are currently defined for the current project are displayed in the Raster Overview. Each row in the view represents one raster.

The added raster contains the following details:

- **Name**: The name is unique within the project.
- **t[µs]**: Nominal time period of the raster. For angle synchronous raster, where the period depends on the rotational speed, the value in this field indicates the shortest period.
- **Priority**: The entry in this field influences the priority granted to the raster within the ETAS measurement system. The value should be between the 1 and 64, the higher the value, the higher the priority. In order to ensure that the system is used to optimum capacity, the fastest measurement raster should be given the highest priority.
- **Acquisition Control**: Method used for data acquisition.
- **Trigger Source**: Which trigger sources are available depends on the Access Device device type.
- **Trigger Property**: This field displays information on the selected trigger method and cannot be edited.

Raster Type: Different Raster Types can be used, **Measurement** raster uses only a measurement channel.

Name	t [µs]	Priority	Acquisition Con...	Trigger Source	Trigger Property	Raster Type
1ms time synchronous	1000	32	DISTAB	Dynamic Hardware Trigger		Measurement
5ms time synchronous	5000	31	DISTAB	Dynamic Hardware Trigger		Measurement
10ms time synchronous	10000	30	DISTAB	Dynamic Hardware Trigger		Measurement

Figure 18: Raster Editor pane regarding the Raster Type

- **Raster Details:** The Raster Details are shown in the right part of the editor window.

The details provide further settings for the raster selected in the raster overview and they depend on the selected Acquisition Control, DISTAB version, Access Device type, trigger source and raster type. When the Access Device supports DISTAB 17 the details for the ECU Property are available. They are used to configure the performance limitation of the raster and to optimize the copy routine of the ECU

Raster Details for
1ms time synchronous

ECU Property (A2L File)	
Core	1
Maximum variables per	375000
Maximum bytes per sec	1500000
8 Bit Signals allowed	Yes
16 Bit Signals allowed	Yes
32 Bit Signals allowed	Yes
64 Bit Signals allowed	Yes

Figure 19: Raster Details in the Raster Editor pane

- **Core**: Specify the core the raster is assigned to.
- **Maximum variables per second**: Specify the performance limitation of the raster in variables per second.
- **Maximum bytes per second**: Specify the performance limitation of the raster in bytes per second.
- **8/16/32/64 bit signals allowed**: Indicates which signal sizes shall be handled at all by the ECU. This setting is used to optimize the copy routine of the ECU.

Note:

The performance limitations of the raster are mandatory. They should **NOT** be set to "0" because that would mean that the raster would not be used at all.

For using an ETAS Access Device with a third party tool, some additional raster settings needs to be done. This will be explained in chapter 3.7.1 DISTAB Fixed Mode.

3.7.1 DISTAB Fixed Mode

- In the Raster Editor pane there is the Edit Rasters button where can be Use DISTAB in Fixed Mode (XCP Device). This option configures the dynamic DISTAB in fixed (static) mode as a XCP Device.

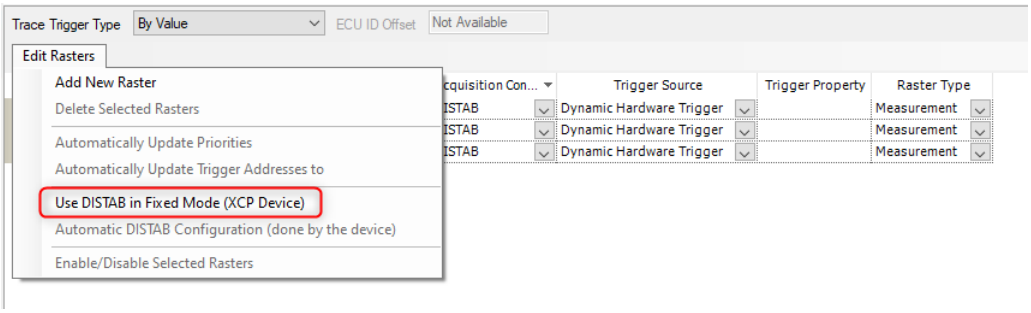


Figure 20: DISTAB Fixed Mode

- With newer firmware and XCT versions for some BR_XETK's and all FETK's the Automatic DISTAB configuration will be done by the device. Then there are no changes in the **Display Table** settings but you will get the info that device is used now as XCP device.

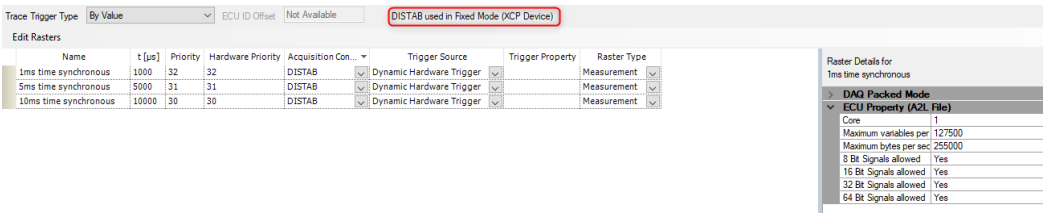


Figure 21: Raster Details properties in the Raster Editor pane for automatic DISTAB

- With older firmware and XCT versions for BR_XETK's, FETK's and XETK's the Automatic DISTAB configuration will be done by selecting **Automatic DISTAB configuration** in the menu **Edit Rasters** . Then the changes in the **Display Table** are visible and you will get

the info that device is used now as XCP device.

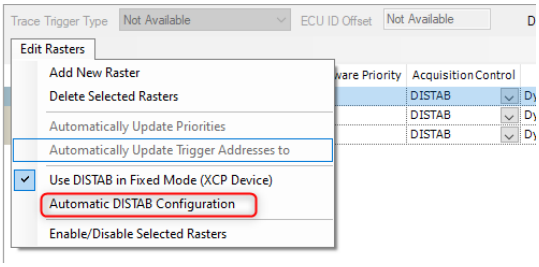


Figure 22: Automatic DISTAB configuration

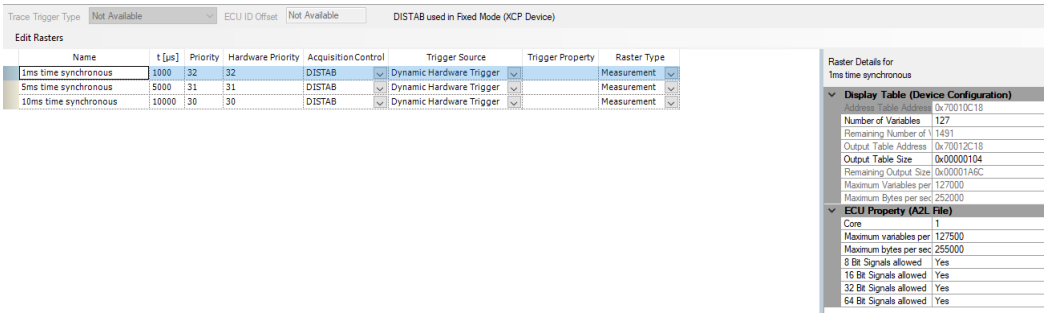


Figure 23: Raster Details properties in the Raster Editor pane

- Most likely you will then see an error symbol next to each raster and get the log information “The configured raster do not fit into the table of the Event Output Area. Please increase the table size or reduce raster”

Name	t [µs]	Priority	Hardware Priority
1ms time synchronous	1000	34	64
2ms time synchronous	2000	32	63
5ms time synchronous	5000	30	62
10ms time synchronous	10000	28	61
20ms time synchronous	20000	26	60
50ms time synchronous	50000	24	59

Figure 24: Raster Configuration Error

- This error means that in the fixed mode the maximal number of variables/bytes per second in all raster exceed the size of memory space assigned for the **Event Config Area** and/or **Event Output Area**.
- The size of the event areas can be checked in the **DISTAB Editor** window in the section **DISTAB Detailed Settings → General Settings**. The values concerning start address and size are ECU internal properties described in the A2L file of the project.
- To solve this problem you need to decrease in the **Display Table** the “Number of Variables” and the “Output Table Size”.

For example:

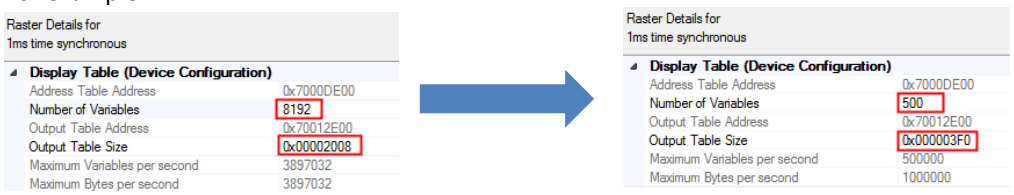


Figure 25: Manual Correction of the "Output Table Size"

Note:

As long as the error symbol is shown, the configuration **cannot be downloaded** to the FETK device. However, you can generate the corresponding A2L entries if there are no other errors. Saving the configuration is also possible.

3.8 Step 8: Configure global settings in the DISTAB Editor

When your A2L container supports only DISTAB13 there are no settings which can be modified or adjusted, due to the reason that DISTAB13 is a static version, which has been already pre-configured in the A2L container. When there would be no DISTAB available, the "**DISTAB Version [COMBO-BOX]"** would show have the value "Disabled" set.

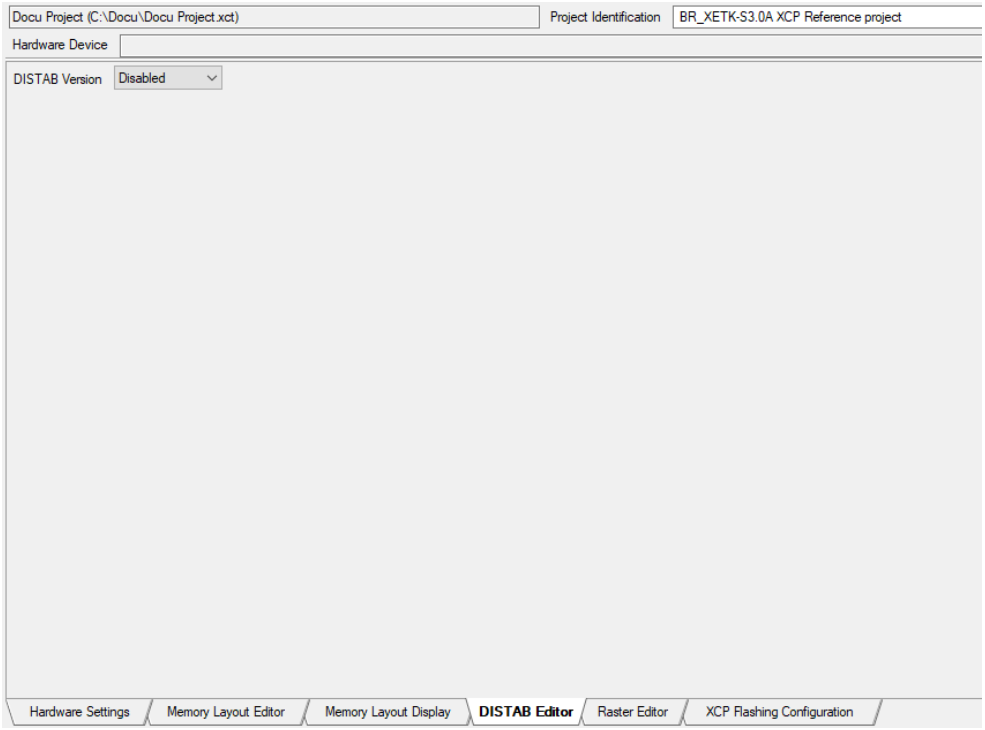


Figure 26: DISTAB Editor with disabled DISTAB Version

When the imported A2L container supports **DISTAB17** there are settings which can be modified or adjusted according to your measurement setup. The imported A2L container, for example, has already a **DISTAB17** configuration, then the **DISTAB17** Editor would be similar to the below . The adjustments, which can be made here, have direct influence to the rasters shown in the Raster Editor. When changes have to be made here, it is necessary to consult the developer of the ECU who implemented the rasters regarding their cycle time and performance throughput.

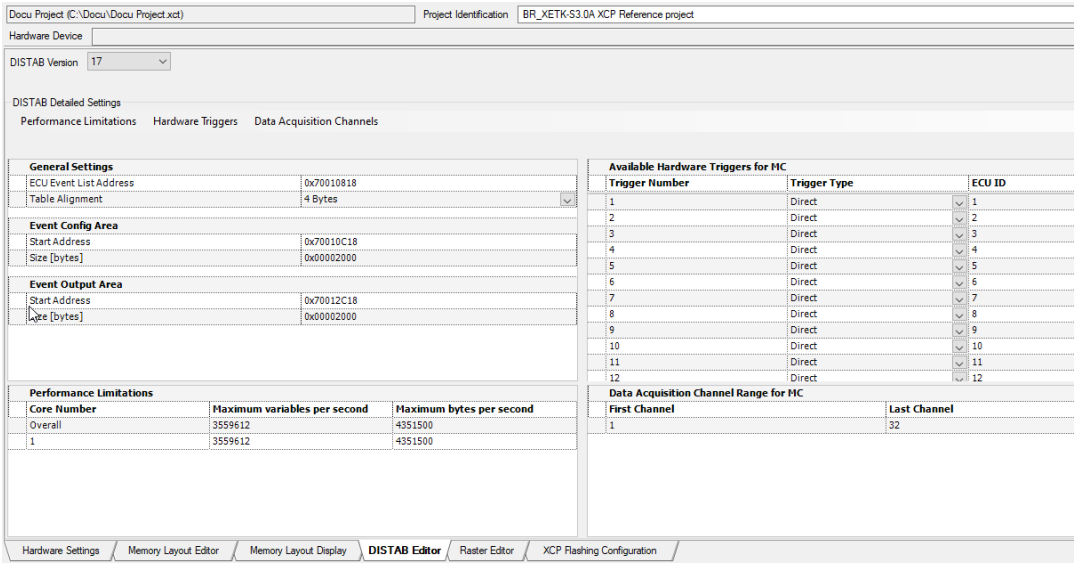


Figure 27: DISTAB Editor with selected DISTAB Version17

Note:

Always have an eye on the Application Log as errors or warnings during the adjustment of project specific values are shown there!!!

3.9 Step 9: Download Configuration to the Access Device

Downloading means to load the active configuration from the Configuration Tool to the Access Device listed in the Hardware Device List. Select in the main menu the following entry:

- **Hardware → Download To:** Writes the configuration to the selected **Access Device** from Hardware Device List.

Note:

The Access Device has to be found and it is displayed in the Hardware Device List. The currently selected configuration has to be linked with the Access Device.

3.9.1 Download to

- Select in the menu bar Hardware → Search for Hardware, or simply press on

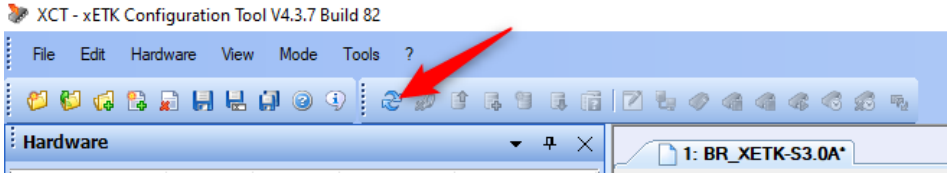


Figure 28: Search For Hardware Menu Tool Item

- The connected Access Device is listed in the **Hardware Device List**

Figure 29: Hardware device list

- By selecting the listed Access Device, both “Hardware Information List” and “Hardware Status List” are showed in the Access Device Configuration pane.
- Select in the menu bar **Hardware → Download To:**
  The dialog “Download configuration” opens.
- Click **OK** to confirm the dialog:
  The busy window “Download configuration” is being displayed during the process.
- **Hardware → Download to:**
  The configuration is being written to the Access Device, which is currently selected in

the Hardware Device List.

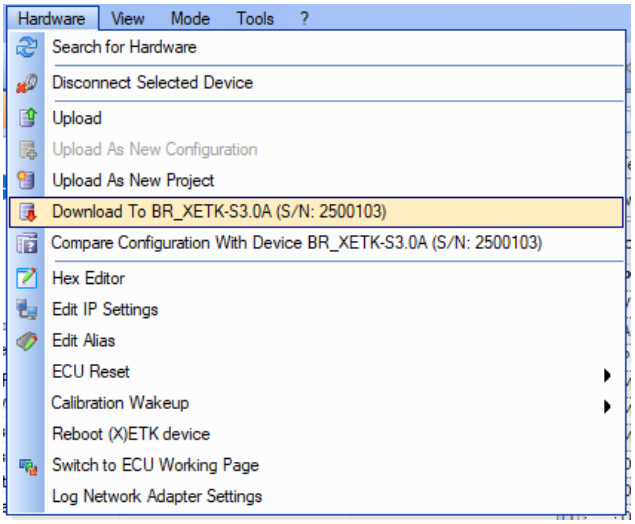


Figure 30: Menu Dialog for downloading configuration to ETK

Note:

If there is no Access Device being found or selected in the Hardware Device List, the **Download to** menu item is not available.

3.10 Step 10: Save the Access Device Configuration

- When all necessary configuration tasks are finished, then it is time to save the project to disk
- Select **File→Save Project "or" Save Project As…:**
  If you selected Save Project As, then specify the name of the project and the location on disk where it should be saved to.

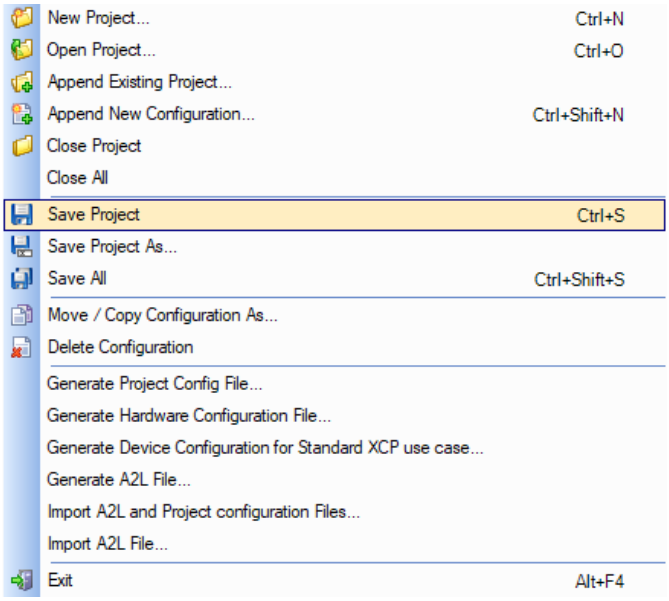


Figure 31: Menu Dialog for saving the Project File

Now the project of the Access Device should be correct, saved and downloaded to the Access Device. The modification and extension of the original or a copy of the imported A2L container can begin with **chapter 4 Adapting the A2L container for the XCPoE on Ethernet Use Case.**

3.11 Configuring and downloading of project via XCT API

The original delivered A2L can be also converted and downloaded to the ETK over the XCT API with a python script.

For this use case is an example in the folder from XCT under: *C:\Program Files\ETAS\ETKTools4.3\Manuals\XCT Console Examples\ConfigureDeviceByAlias.pyxct* which can be used with some little changes.

The script were created for the usage in 3rd party tools.

At first the lines 9 to 11 must be commented out. Here in the code snippet the first 3 lines. Then change the parameter from xctFile to a2lFile (needed because instead of project will be an .a2l file loaded) and give an alias name.

```
#if len( sys.argv ) < 3:
#print "Please pass the xct file name and the device alias as command line …
#sys.exit( 1 )


# the first command line argument is the file path of the XCT project
# that shall be downloaded
a2lFile = r"C:\BR_XETK-S3\Festkonfig\107_BRS3_raster_for_fixed_conf.a2l"
# the second command line argumant is the alias of the device to which
# shall be configured
alias = "BR_XETK-S3"
```

Note:

If there is no alias name forgiven, then the first ETK from available list will be get the download from API script.

Now in line 36 to 38 also changed the xctFile to a2lFile, LoadProject to LoadA2LFile and add the line project.ConvertToFixedProject() before print “Loaded.. in line 38.

```
print "Loading XCT project from file " + a2lFile + "\n"
project = XCT.LoadA2LFile( a2lFile )
project.ConvertToFixedProject()
print "Loaded project " + a2lFile + "\n"
```

When now the script will be executed in the command line ETK will be configured automatically.

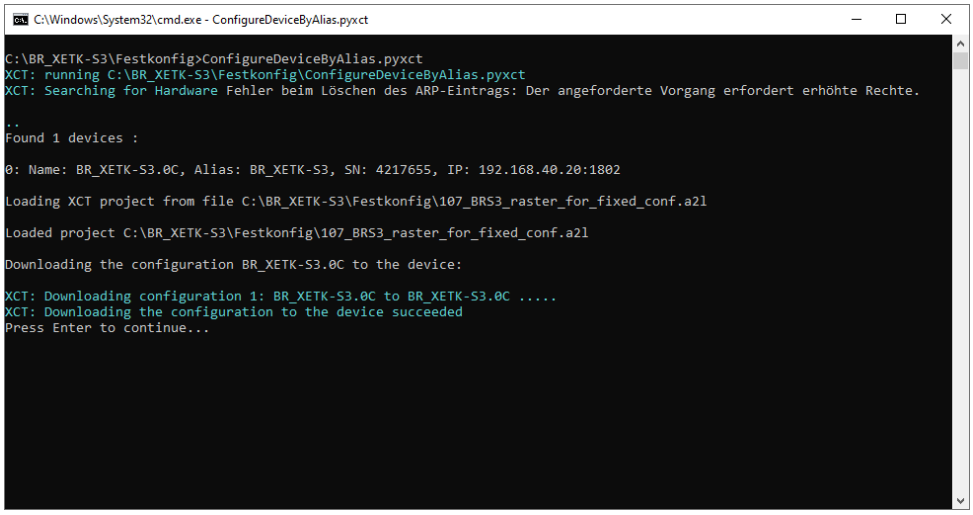


Figure 32: Configure ETK with XCT API

4 Adapting the A2L container for the XCPoE on Ethernet Use Case

The original delivered A2L file need in most of the cases to be extended with the entries necessary for using the Access Device as XCP Access Device.

A simple way to do this is generating the required entries direct from the current Configuration and merge these to the original A2L file.

To get a valid A2L container it is necessary to accomplish two steps:

1. Extending the **A2ML section** of the A2L container
2. Extending the **MOD_PAR** and **MODULE sections** of the A2L container

In the next chapters it will be shown how and where to extend the sections mentioned above.

There are chapters, which contains only a **UML** like diagram. With the help of these diagrams, it is possible to locate the position where to insert the exported **A2L** sections.

Note:

It is necessary to keep the order of these entries in order not to fault the **ASAM-MCD 2MC** standard or the rule of the **AML IF_DATA** sections.

4.1 Step 1: Generate A2L Entries

The XCT Tool offers the possibility to generate an A2L file which contains all A2L relevant entries related to the used Access Device.

Following options are relevant and available for this project:

- **IF_ DATA ETK_XETK** entries for INCA usage.
- **IF_DATA XCP** for all tools using an ETAS Access Device as standard XCP slave.
- Open the project from which you would like to export the data.
- Make sure that all values of the parameters in the editors of the Configuration Window are correct and the plausibility check shows no mistakes.
- Select in the menu bar **File→Generate A2L Entries:**
  The dialog “Save As” opens.
- Specify a file name and a place on your file system for the A2L entries file.
- Click on the **Save** button to confirm the dialog. The dialog “Please choose which **A2L** definition shall be exported” opens.
- In IF_DATA select both definitions ETK_XETK and XCP.
- Choose the version of the ETK_XETK and XCP you want to use. It is recommended to use the newest version.
- Specify the **IP address** and **XCP port** for the **XCP** export. The IP address and XCP port will be written as additional information to the A2L file and typically used by the XCP master to communicate with the Access Device.
- Select the **ETK_XETK AML** version used for writing the **A2L** data to file.
- Optional you can select which transport layer shall be used (TCP, UDP or both) and you can name for each the Transport Layer Instance

- Click **OK** to confirm the dialog. The **A2L** file is created on your file system.

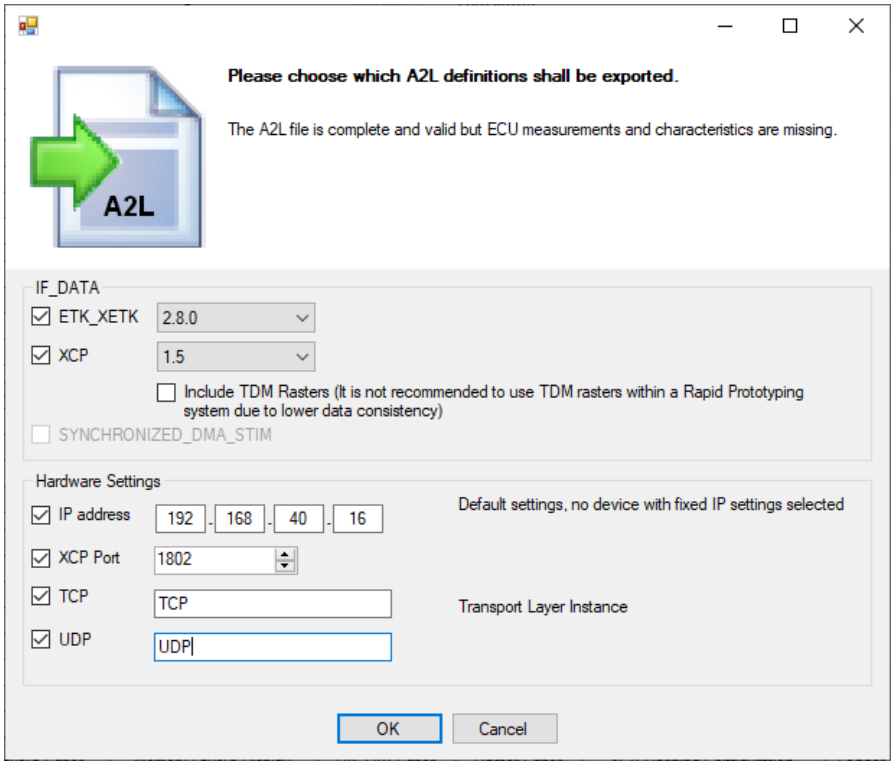


Figure 33: A2L generation dialog

4.2 Step 2: Merging XCP entries into the existing A2L

The A2L file generated in chapter 4.1 can be used for merging the missing XCP entries or for correcting the settings in the original A2L file.

4.3 Export-To-A2L **A2ML** metalanguage section for the **IF_DATA XCP** section(s)

```
struct Protocol_Layer                    /* at MODULE */
{
    …
};

struct Daq                               /* DAQ supported, at MODULE*/
{
    …
};

taggedunion Daq_Event                    /* at MEASUREMENT */
{
    ….
};

struct Pag                               /* PAG supported, at MODULE */
```

```
{
    …
};

struct Pgm                              /* PGM supported, at MODULE */
{
    …
};

struct Segment                          /* at MEMORY_SEGMENT */
{
    …
};

…
…
taggedstruct Common_Parameters
{
    block "PROTOCOL_LAYER" struct Protocol_Layer;
    block "SEGMENT" struct Segment;
    block "DAQ" struct Daq;
    block "PAG" struct Pag;
    block "PGM" struct Pgm;
    block "DAQ_EVENT" taggedunion Daq_Event;
};

struct TCP_IP_Parameters                        /* At MODULE */
{
    …
};

struct UDP_IP_Parameters                  /* at MODULE */
{
    …
};
/* Definition of the IF_DATA XCP */
"XCP" struct
{
    /* default parameters */
    taggedstruct Common_Parameters ;
    /* transport layer specific parameters  */
    /* overruling of the default parameters */
    taggedstruct
    {
        block "XCP_ON_TCP_IP" struct
        {
```

```
            /* specific for TCP_IP   */
            struct TCP_IP_Parameters;
            /* overruling of default */
            taggedstruct Common_Parameters;
        };

        block "XCP_ON_UDP_IP" struct
        {
            /* specific for UDP      */
            struct UDP_IP_Parameters;
            /* overruling of default */
            taggedstruct Common_Parameters;
        };
    };
};
```

XCP AML LOCATION IN THE A2L CONTAINER

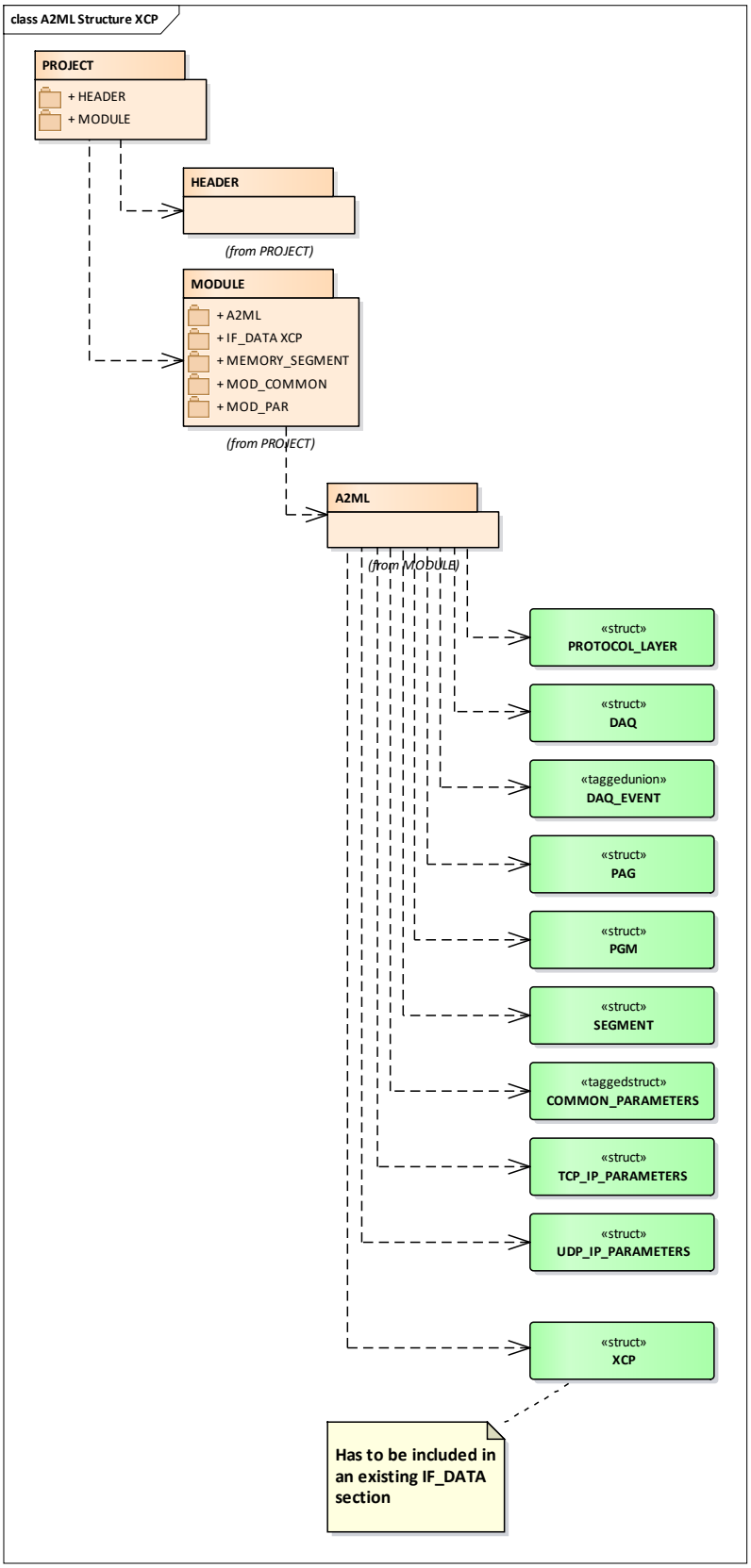


Figure 34: Location where to insert the XCP AML sections within A2ML section in the A2L container

Note:

The shaded box in the above Figure shows where to insert the exported **structs PROTOCOL_LAYER, DAQ, DAQ_EVENT, PAG; PGM, SEGMENT, COMMON_PARAMETERS, TCP_IP_PARAMETERS, UDP_IP_PARAMETERS, (IF_DATA) XCP** section.

::PROJECT::MODULE::A2ML::*

Hint:

The exported **IF_DATA XCP** section in the **A2ML** section, must not be completely copied to the original **A2ML** section, but only the inside sections/elements. Because in the existing A2L container there might be already an **IF_DATA** section with some other sections regarding Access Devices or similar. Therefore, in an **A2ML** there MUST be only one **IF_DATA** section defined; multiple sections/elements are not allowed.

4.3.1 Export-To-A2L MEMORY_SEGMENT: ASAM XCP Version 1.0 <= x.y <= 1.1

The output (example) of an exported MEMORY_SEGMENT has the following structure:

```
/* This structure reflects one line in the memory layout editor */
/begin MEMORY_SEGMENT
    etk_data_0 "ETK_Data_0" DATA FLASH INTERN 0xA0100800 0x1FF800 -1 -1 -1 -1 -1
    /begin IF_DATA XCP
       /* complete structure see below at UDP/IP-PROTOCOL */
        /begin XCP_ON_UDP_IP
        …
        /end XCP_ON_UDP_IP

        /* complete structure see below at TCP/IP-PROTOCOL */
        /begin XCP_ON_TCP_IP
        …
        /end XCP_ON_TCP_IP
    /end IF_DATA

    /* This block is non asam xcp standard, but necessary for re-import into XCT   */
    /* Because without these blocks no download of that memory segment will happen */
    /begin IF_DATA ETK_XETK
        /begin ETK_XETK_ACCESS
            SERIAL_INTERFACE
        /end ETK_XETK_ACCESS
    /end IF_DATA
/end MEMORY_SEGMENT
```

The output (example) of an exported MEMORY_SEGMENT with UDP/IP as Transport Layer:

```
/begin XCP_ON_UDP_IP
    /* VERSION */ 0x100
    /* PORT    */ 1802
    ADDRESS       "192.168.40.16"
```

```
    /begin SEGMENT
        /* SEGMENT_NUMBER     */ 0x2
        /* number of pages    */ 0x2
        /* ADDRESS_EXTENSION  */ 0x0
        /* COMPRESSION_METHOD */ 0x0
        /* ENCRYPTION_METHOD  */ 0x0

        /begin CHECKSUM
            XCP_ADD_44
        /end CHECKSUM

        /begin PAGE
            /* PAGE_NUMBER           */ 0
            /* ECU_ACCESS_TYPE       */ ECU_ACCESS_WITH_XCP_ONLY
            /* XCP_READ_ACCESS_TYPE  */ XCP_READ_ACCESS_WITH_ECU_ONLY
            /* XCP_WRITE_ACCESS_TYPE */ XCP_WRITE_ACCESS_NOT_ALLOWED
        /end PAGE

        /begin PAGE
            /* PAGE_NUMBER           */ 1
            /* ECU_ACCESS_TYPE       */ ECU_ACCESS_DONT_CARE
            /* XCP_READ_ACCESS_TYPE  */ XCP_READ_ACCESS_DONT_CARE
            /* XCP_WRITE_ACCESS_TYPE */ XCP_WRITE_ACCESS_DONT_CARE
        /end PAGE
    /end SEGMENT
/end XCP_ON_UDP_IP
```

The output (example) of an exported MEMORY_SEGMENT with TCP/IP as Transport Layer:

```
/begin XCP_ON_TCP_IP
      /* VERSION */ 0x100
      /* PORT    */ 1802
      ADDRESS "192.168.40.16"

      /begin SEGMENT
            /* SEGMENT_NUMBER     */ 0x2
            /* number of pages    */ 0x2
            /* ADDRESS_EXTENSION  */ 0x0
            /* COMPRESSION_METHOD */ 0x0
            /* ENCRYPTION_METHOD  */ 0x0

            /begin CHECKSUM
            XCP_ADD_44
            /end CHECKSUM
```

```
            /begin PAGE
                  /* PAGE_NUMBER           */ 0
                  /* ECU_ACCESS_TYPE       */ ECU_ACCESS_WITH_XCP_ONLY
                  /* XCP_READ_ACCESS_TYPE  */ XCP_READ_ACCESS_WITH_ECU_ONLY
                  /* XCP_WRITE_ACCESS_TYPE */ XCP_WRITE_ACCESS_NOT_ALLOWED
            /end PAGE


            /begin PAGE
                  /* PAGE_NUMBER           */ 1
                  /* ECU_ACCESS_TYPE       */ ECU_ACCESS_DONT_CARE
                  /* XCP_READ_ACCESS_TYPE  */ XCP_READ_ACCESS_DONT_CARE
                  /* XCP_WRITE_ACCESS_TYPE */ XCP_WRITE_ACCESS_DONT_CARE
            /end PAGE
      /end SEGMENT
/end XCP_ON_TCP_IP
```

MEMORY_SEGMENT Location within A2L container

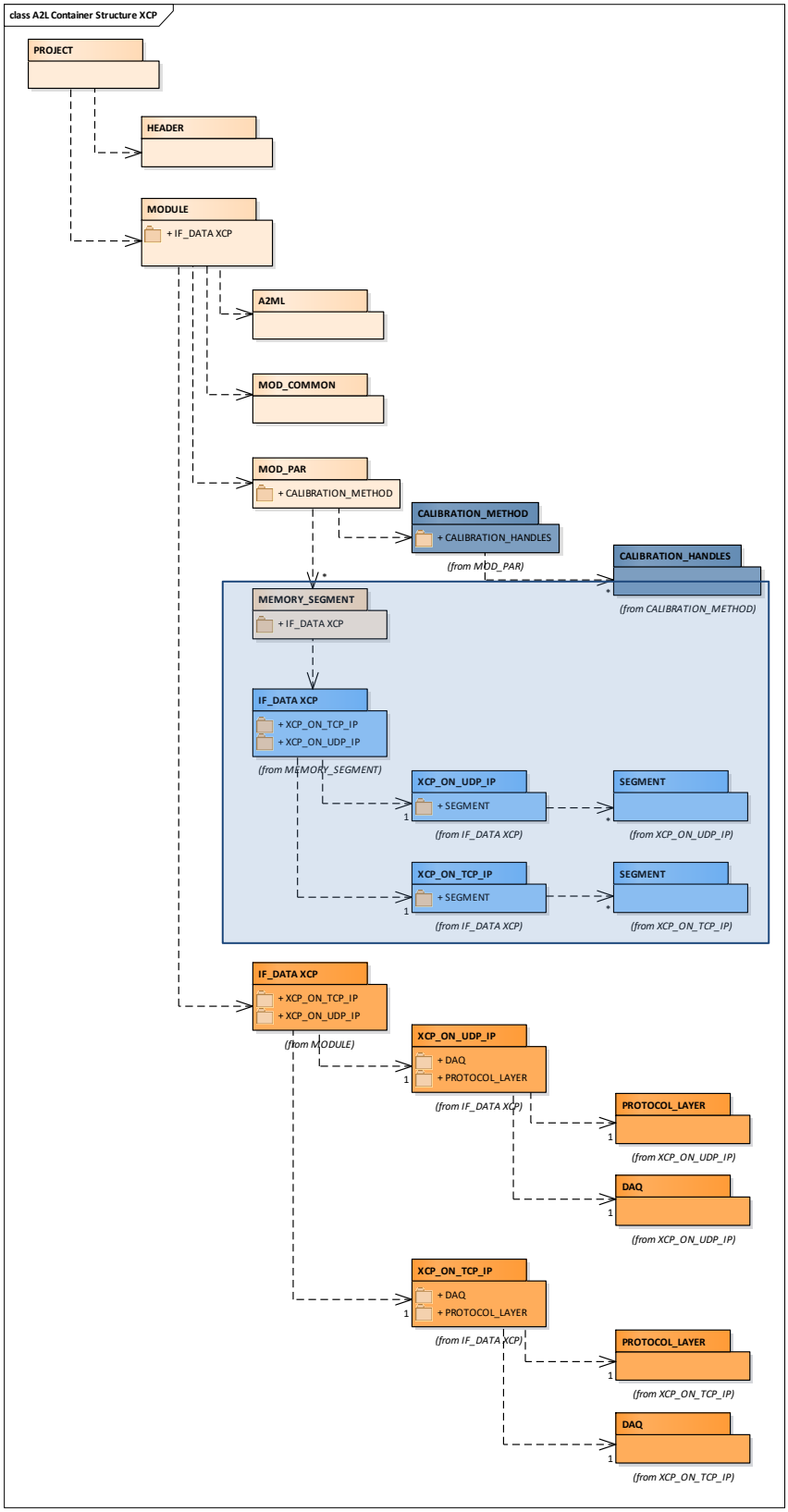


Figure 35: Location where to insert the **MEMORY_SEGEMENT** section within A2L container

Note:

The shaded box in the above Figure shows where to insert the exported IF_DATA XCP section.

→ ::PROJECT::MODULE::MOD_PAR::**MEMORY_SEGMENTs**

4.3.2 Export-To-A2L **MEMORY_SEGMENT**: ASAM XCP Version **x.y >= 1.2**

The output (example) of an exported MEMORY_SEGMENT has the following structure:

```
/begin MEMORY_SEGMENT
    etk_data_0 "ETK_Data_0" DATA FLASH INTERN 0xA0100800 0x1FF800 -1 -1 -1 -1 -1
    /begin IF_DATA XCPplus 0x0103
        /begin SEGMENT
            /* SEGMENT_NUMBER     */ 0x2
            /* number of pages    */ 0x2
            /* ADDRESS_EXTENSION  */ 0x0
            /* COMPRESSION_METHOD */ 0x0
            /* ENCRYPTION_METHOD  */ 0x0
            /begin CHECKSUM
                XCP_ADD_44
            /end CHECKSUM
            /begin PAGE
                /* PAGE_NUMBER           */ 0
                /* ECU_ACCESS_TYPE       */ ECU_ACCESS_WITH_XCP_ONLY
                /* XCP_READ_ACCESS_TYPE  */ XCP_READ_ACCESS_WITH_ECU_ONLY
                /* XCP_WRITE_ACCESS_TYPE */ XCP_WRITE_ACCESS_NOT_ALLOWED
            /end PAGE
            /begin PAGE
                /* PAGE_NUMBER           */ 1
                /* ECU_ACCESS_TYPE       */ ECU_ACCESS_DONT_CARE
                /* XCP_READ_ACCESS_TYPE  */ XCP_READ_ACCESS_DONT_CARE
                /* XCP_WRITE_ACCESS_TYPE */ XCP_WRITE_ACCESS_DONT_CARE
            /end PAGE
        /end SEGMENT
    /end IF_DATA

    /* This block is non asam xcp standard, but necessary for re-import into XCT   */
    /* Because without these blocks no download of that memory segment will happen */
    /* and therefore that memory segment won't be accessible by the mcd tool       */
    /begin IF_DATA ETK_XETK
        /begin ETK_XETK_ACCESS
            SERIAL_INTERFACE
        /end ETK_XETK_ACCESS
    /end IF_DATA
/end MEMORY_SEGMENT
```

Note:

Since the ASAM XCP Version 1.2 it is not allowed anymore to specify the Transport Layer (TCP/IP, UDP/IP, etc.) in the **MEMORY_SEGMENT** of the A2L container.

4.3.3 Export-To-A2L **CALIBRATION_METHOD**

When this block was already in the previously imported A2L included, then no further actions have to be done.

If there were no CALIBRATION_HANDLES respectively there was no CALIBRATION_METHOD specified then this exported block has to be merged into the new A2L container.

```
/begin CALIBRATION_METHOD
    "FixedSizeMoveableEmuRAM"
    1
    /begin CALIBRATION_HANDLE
        0
        0xB9000000
        0x20000
    /end CALIBRATION_HANDLE

    /begin CALIBRATION_HANDLE
        1
        0xB9020000
        0x20000
    /end CALIBRATION_HANDLE

    ….

    /begin CALIBRATION_HANDLE
        [Max supported handles - 1]
        0xB9080000
        0x20000
    /end CALIBRATION_HANDLE
/end CALIBRATION_METHOD
```

Note:

The CALIBRATION_METHOD block is an ASAM-MCD 2MC Standard element in the A2L container. Therefore, there is no distinction needed for different versions necessary.

CALIBRATION_METHOD Location within A2L container

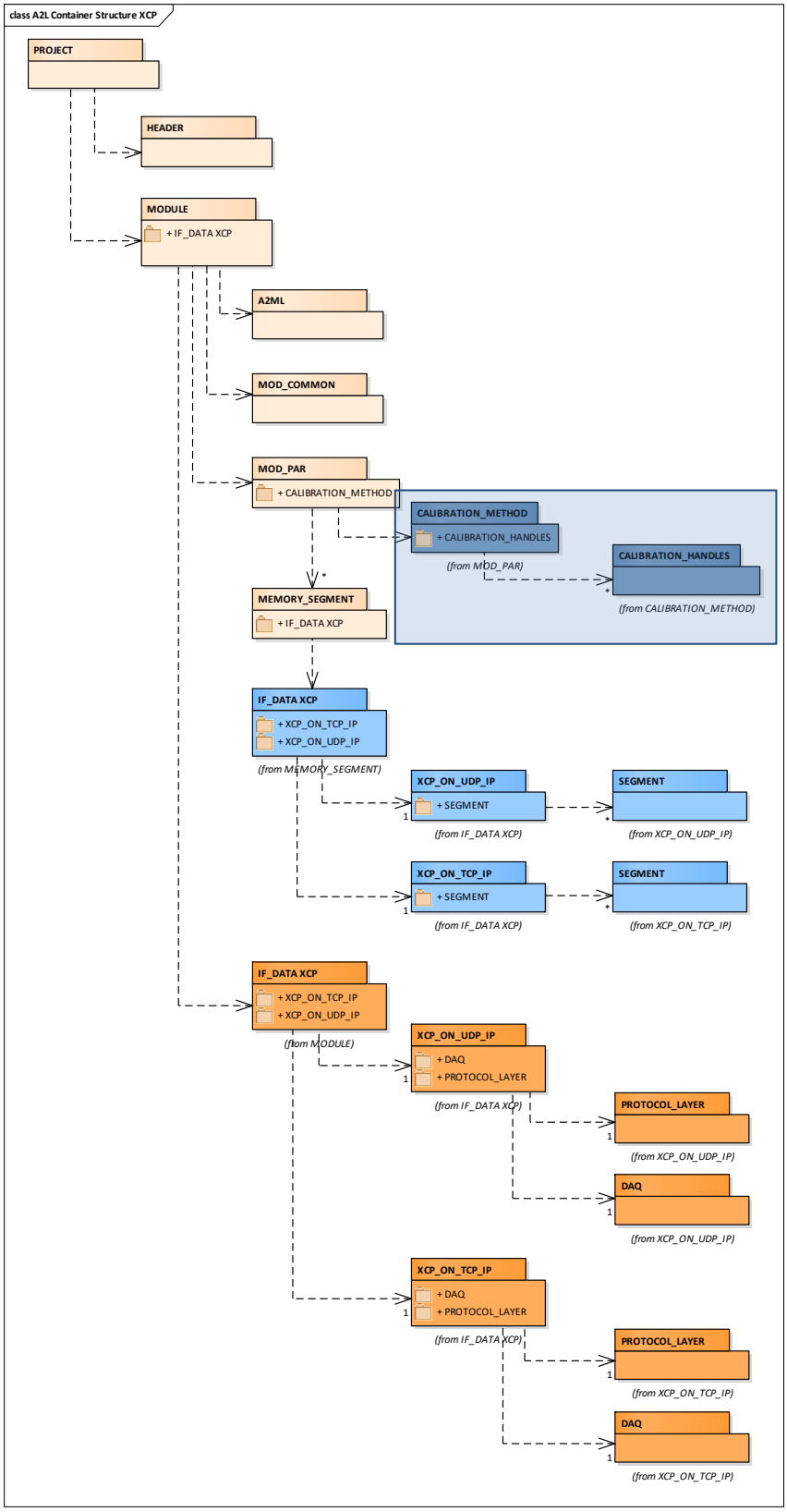


Figure 36: Location where to insert the **CALIBRATION_METHOD** section within A2L container

Note:

The shaded box in the above Figure shows where to insert the exported CALIBRATION_METHOD section.

 ::PROJECT::MODULE::MOD_PAR::**CALIBRATION_METHOD**

4.3.4 Export-To-A2L : **TRANSPORT, PROTOCOL_LAYER** and **DAQ** ASAM XCP Version **1.0 <= x.y <= 1.1**

In all IF_DATA XCP Sections, the parameters specified for XCP on TCP/IP and for XCP on UDP/IP have to be added with the information about used Port Number and IP Address. The user can then select in the third party tool if TCP/IP or UDP/IP should be used.

- **DAQ**: Some DAQ parameters:
  - MAX_DAQ
  - MAX_EVENT_CHANNEL
  - OPTIMISATION_TYPE
  - IDENTIFICATION_FIELD

  are not correct and have to be adapted with the values generated from the XCT Tool.
- **EVENT**: For all events defined in the project the EVENT parameters concerning:
  - EVENT_CHANNEL_NAME
  - EVENT_CHANNEL_SHORT_NAME
  - EVENT_CHANNEL_NUMBER
  - EVENT_CHANNEL_TIME_CYCLE
  - EVENT_CHANNEL_TIME_UNIT
  - EVENT_CHANNEL_PRIORITY

  have to be adapted with the values generated from the XCT Tool.

The output of an exported PROTOCOL_LAYER with UDP/IP and TCP/IP as Transport Layer:

```
/begin IF_DATA XCP
    /begin XCP_ON_UDP_IP
        /* VERSION */ 0x100
        /* PORT    */ 1802
        ADDRESS "192.168.40.16"
        /begin PROTOCOL_LAYER
            …
        /end PROTOCOL_LAYER
        /begin DAQ
            …
        /end DAQ
    /end XCP_ON_UDP_IP



    /begin XCP_ON_TCP_IP
        /* VERSION */ 0x100
```

```
        /* PORT    */ 1802
        ADDRESS "192.168.40.16"
        /begin PROTOCOL_LAYER
            …
        /end PROTOCOL_LAYER
        /begin DAQ
            …
        /end DAQ
    /end XCP_ON_TCP_IP
/end IF_DATA
```

Note:

Due to no additional information, the complete EVENT block elements have been shortened, in a way that only two EVENT are visible. In a real export snippet, the EVENT list is much longer with more entries.

PROTOCOL_LAYER and DAQ Location within A2L container

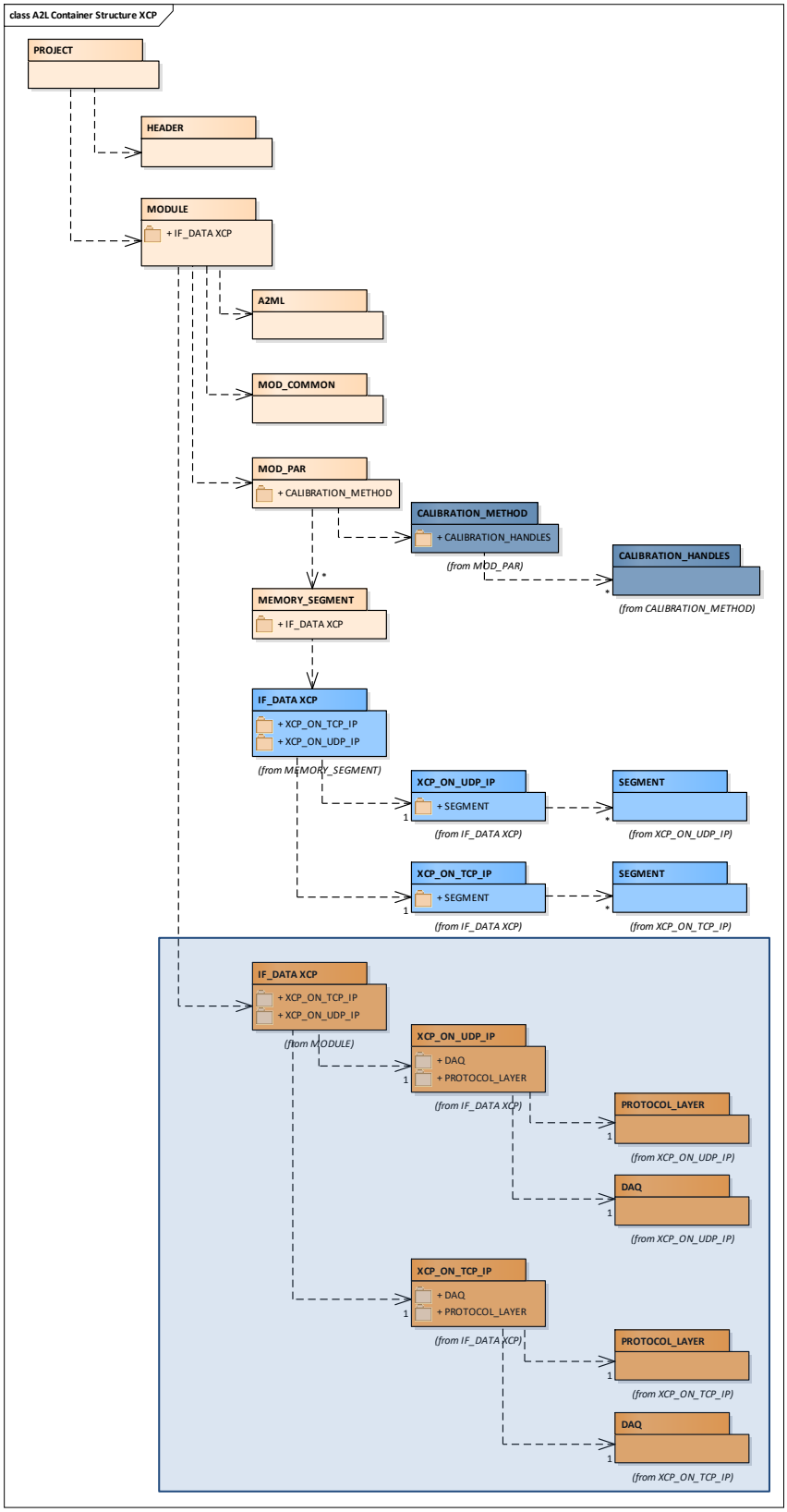


Figure 37: Location where to insert the PROTOCOL_LAYER section within A2L container

Note:

The shaded box in the above Figure shows where to insert the exported **IF_DATA XCP** section.

 ::PROJECT::MODULE::**IF_DATA XCP**

4.3.5 Export-To-A2L : **TRANSPORT**, **PROTOCOL_LAYER** and **DAQ** ASAM XCP Version **x.y >= 1.2**

The timing parameters as well as the values for MAX_CTO and MAX_DTO should be checked and changed if necessary with the values generated from Configuration Tool.

```
/begin IF_DATA XCPplus 0x0103    /* IF_DATA XCP version */
    /begin PROTOCOL_LAYER
        0x0103                       /* XCP protocol layer version */
        0x0019                       /* T1 [ms] */
        0x0019                       /* T2 [ms] */
        0x0019                       /* T3 [ms] */
        0x0019                       /* T4 [ms] */
        0x0019                       /* T5 [ms] */
        0x0005                       /* T6 [ms] */
        0x00C8                       /* T7 [ms] */
        0x20                         /* MAX_CTO */
        0x00FF                       /* MAX_DTO */

        BYTE_ORDER_MSB_FIRST
        ADDRESS_GRANULARITY_WORD
        SEED_AND_KEY_EXTERNAL_FUNCTION     "MyS&K.DLL"

        OPTIONAL_CMD  GET_ID
        OPTIONAL_CMD  SET_REQUEST
        OPTIONAL_CMD  GET_SEED
        OPTIONAL_CMD  UNLOCK
        OPTIONAL_CMD  SET_MTA
        OPTIONAL_CMD  UPLOAD
        OPTIONAL_CMD  BUILD_CHECKSUM
        OPTIONAL_CMD  DOWNLOAD
        OPTIONAL_CMD  SET_CAL_PAGE
        OPTIONAL_CMD  GET_CAL_PAGE
        OPTIONAL_CMD  COPY_CAL_PAGE
        OPTIONAL_CMD  CLEAR_DAQ_LIST
        OPTIONAL_CMD  SET_DAQ_PTR
        OPTIONAL_CMD  WRITE_DAQ
        OPTIONAL_CMD  SET_DAQ_LIST_MODE
        OPTIONAL_CMD  START_STOP_DAQ_LIST
        OPTIONAL_CMD  START_STOP_SYNCH
        OPTIONAL_CMD  GET_DAQ_CLOCK
```

```
        OPTIONAL_CMD  WRITE_DAQ_MULTIPLE
        OPTIONAL_CMD  DTO_CTR_PROPERTIES
        OPTIONAL_CMD  TIME_CORRELATION_PROPERTIES
    /end PROTOCOL_LAYER


    /begin DAQ
        DYNAMIC                                             /* DAQ_CONFIG_TYPE
*/
        64                                                  /* MAX_DAQ
*/
        64                                                  /* MAX_EVENT_CHANNEL
*/
        0                                                   /* MIN_DAQ
*/
        OPTIMISATION_TYPE_ODT_TYPE_ALIGNMENT                /* OPTIMISATION_TYPE
*/
        ADDRESS_EXTENSION_FREE                              /* ADDRESS_EXTENSION
*/
        IDENTIFICATION_FIELD_TYPE_RELATIVE_WORD_ALIGNED     /* IDENTIFICATION_FIELD
*/
        GRANULARITY_ODT_ENTRY_SIZE_DAQ_BYTE                 /*
GRANULARITY_ODT_ENTRY_SIZE_DAQ */
        8                                                   /* MAX_ODT_ENTRY_SIZE_DAQ
*/
        OVERLOAD_INDICATION_EVENT                           /* OVERLOAD_INDICATION
*/
        /begin STIM
            GRANULARITY_ODT_ENTRY_SIZE_STIM_BYTE            /*
GRANULARITY_ODT_ENTRY_SIZE_STIM */
            8                                               /* MAX_ODT_ENTRY_SIZE_STIM
*/
        /end STIM
        /begin TIMESTAMP_SUPPORTED
            1                                               /* TIMESTAMP_TICKS
*/
            SIZE_DWORD                                      /* TIMESTAMP_SIZE
*/
            UNIT_1US                                        /* RESOLUTION OF TIMESTAMP
*/
            TIMESTAMP_FIXED
        /end TIMESTAMP_SUPPORTED
        /begin EVENT
            /* EVENT_CHANNEL_NAME       */ "measure_r00"
            /* EVENT_CHANNEL_SHORT_NAME */ "measure_r"
            /* EVENT_CHANNEL_NUMBER     */ 0x28
            /* EVENT_CHANNEL_TYPE       */ DAQ
            /* MAX_DAQ_LIST             */ 0x01
            /* TIME_CYCLE               */ 0x64
            /* TIME_UNIT                */ 0x4
            /* PRIORITY                 */ 0x28
```

```
        /end EVENT
        …
        /begin EVENT
            /* EVENT_CHANNEL_NAME       */ "measure_r25"
            /* EVENT_CHANNEL_SHORT_NAME */ "measure25"
            /* EVENT_CHANNEL_NUMBER     */ 0x26
            /* EVENT_CHANNEL_TYPE       */ DAQ
            /* MAX_DAQ_LIST             */ 0x01
            /* TIME_CYCLE               */ 0x64
            /* TIME_UNIT                */ 0x4
            /* PRIORITY                 */ 0x26
        /end EVENT
    /end DAQ
    /begin XCP_ON_TCP_IP
        0x0103                                          /* XCP on TCP_IP version */
        0x5555                                          /* PORT */
        IPV6 "FE80:0000:0000:0000:0202:B3FF:FE1E:8329"  /* IPV6 ADDRESS */
        MAX_BUS_LOAD 15                                 /* MAX_BUS_LOAD */
        MAX_BIT_RATE 100
    /end XCP_ON_TCP_IP
    /begin XCP_ON_UDP_IP
        0x0103                                          /* XCP on UDP_IP version */
        0x5555                                          /* PORT */
        ADDRESS "127.0.0.1"                             /* ADDRESS */
        MAX_BUS_LOAD 80                                 /* MAX_BUS_LOAD */
        MAX_BIT_RATE 10
    /end XCP_ON_UDP_IP
/begin IF_DATA
```

Note:

In the higher versions of the XCP Standard, the Transport Layer does not contain the Protocol Layer and further block entries. This leads into more compact A2L containers with a higher clarity. The Transport Layer information is just added at the very end of the IF_DATA XCPplus in the MODULE block.

5 Contact Information

Technical Support

For details of your local sales office as well as your local technical support team and product hotlines, take a look at the website: www.etas.com/hotlines

ETAS Headquarters

ETAS GmbH
Borsigstraße 24
70469 Stuttgart
Germany

Phone: +49 711 3423-0
Fax: +49 711 3423-2106
Internet: www.etas.com