Elektrobit

MCAL Wrapper (McalExt) documentation

for CORTEXM S32K14X

product release 8.8.7

Elektrobit Automotive GmbH
Am Wolfsmantel 46
91058 Erlangen, Germany
Phone: +49 9131 7701 0
Fax: +49 9131 7701 6333
Email: info.automotive@elektrobit.com

Technical support

https://www.elektrobit.com/support

Legal disclaimer

Confidential information.

Table of Contents

1. Overview of MCAL Integration release notes

Welcome to the MCAL release notes. These release notes are target-specific and derivative-specific.

Chapter 2, “Supported toolchain“ provides information about the supported toolchain.

Chapter 3, “Scope of this release“ provides specific information about the hardware-dependent third-party modules contained in this EB tresos AutoCore release:

- AUTOSAR version and revision of your hardware-dependent modules
- SWS version and revision
- Module version
- Supplier of your hardware-dependent modules

1.1. Location of MCAL documentation

Depending on the platform release that you purchased, some of the modules may be supplied to Elektrobit Automotive GmbH by third-parties. All MCAL modules are documented outside of these release notes. This documentation contains additional information about the third-party MCAL modules and the patches that were made by Elektrobit Automotive GmbH.

You can find the MCAL module documentation in the following locations:

- `$TRESOS_BASE/doc/5.0_MCAL_modules/MCAL_Wrapper_documentation.pdf`
- `$TRESOS_BASE/plugins/<McalExt plugin>/MCAL_Delivery/plugins/<Module name>/doc`
- In the online help of EB tresos Studio.

For information about the online help in EB tresos Studio, see the EB tresos Studio user documentation.

2. Supported toolchain

This release of EB tresos AutoCore supports IAR 8.40.3

2.1. Toolchain options

The toolchain options summarize under which conditions this release needs to be built. The release is tested using these toolchain options. If you change the compiler options, consider this release *untested*.

Compiler	**Options**
	`--no_wrap_diagnostics -e --cpu Cortex-M4f --fpu FPv4-SP --debug --endian little --cpu_mode thumb --no_clustering -DOSB_TOOL=OSB_iar -Ohz --no_mem_idioms --no_explicit_zero_opt --require_prototypes -DS32K1XX -DS32K144 -DIAR -DUSE_SW_VECTOR_MODE -DI_CACHE_ENABLE -DENABLE_FPU -DMCAL_ENABLE_USER_MODE_SUPPORT --diag_suppress=Pa050`
Assembler	**Options**
	`--cpu Cortex-M4f -r -g`
Linker	**Options**
	`--no_wrap_diagnostics --entry boardResetStart --cpu Cortex-M4f --fpu FPv4-SP --enable_stack_usage --skip_dynamic_initialization`

3. Scope of this release

3.1. Platforms Module

This release of the `Platforms` module contains the mandatory and derivative-specific implementation part of the `Base` module.

This `Platforms` module shall be used only for S32K14X derivatives.

This module is tested only on hardware with the same sub-derivative as the third-party MCAL version. Other sub-derivatives are not tested.

3.2. Third-party MCAL version

This release contains the MCAL release SW32K1_RTD_4.4_1.0.1_HF01_D2209 from NXP.

This release of EB tresos AutoCore is tested only with sub-derivative s32k144_lqfp100.

3.3. Third-party MCAL modules

This release includes the hardware-dependent third-party MCAL modules listed in the table below.

Module name	AUTOSAR version and revision	SWS version and revision	Module version	Supplier
Adc	4.4.0	4.4.0	1.0.1	NXP
BaseNXP	4.4.0	4.4.0	1.0.1	NXP
Can	4.4.0	4.0.0	1.0.1	NXP
Crc	4.4.0	4.4.0	1.0.1	NXP
Crypto	4.4.0	4.4.0	1.0.1	NXP
Dio	4.4.0	4.4.0	1.0.1	NXP
Eep	4.4.0	4.4.0	1.0.1	NXP
Eth	4.4.0	4.4.0	1.0.1	NXP
Fee	4.4.0	2.0.0	1.0.1	NXP
Fls	4.4.0	4.4.0	1.0.1	NXP
Gpt	4.4.0	4.4.0	1.0.1	NXP

Module name	AUTOSAR version and revision	SWS version and revision	Module version	Supplier
I2c	4.4.0	4.4.0	1.0.1	NXP
I2s	4.4.0	4.4.0	1.0.1	NXP
Icu	4.4.0	4.4.0	1.0.1	NXP
Lin	4.4.0	4.4.0	1.0.1	NXP
Mcl	4.4.0	4.3.0	1.0.1	NXP
Mcu	4.4.0	4.4.0	1.0.1	NXP
Ocu	4.4.0	0.2.4	1.0.1	NXP
Port	4.4.0	4.4.0	1.0.1	NXP
Pwm	4.4.0	0.2.4	1.0.1	NXP
Qdec	4.4.0	4.4.0	1.0.1	NXP
Rm	4.4.0	4.4.0	1.0.1	NXP
Spi	4.4.0	3.2.0	1.0.1	NXP
Uart	4.4.0	4.4.0	1.0.1	NXP
Wdg	4.4.0	4.4.0	1.0.1	NXP
Wdg	4.4.0	4.4.0	1.0.1	NXP
Resource	4.4.0	4.4.0	1.0.1	NXP

Table 3.1. Third-party hardware-dependent modules

3.4. Third-party MCAL AUTOSAR APIs tested by Elektrobit Automotive GmbH

The following third-party MCAL AUTOSAR APIs are verified against the AUTOSAR standard. The scope of Elektrobit Automotive GmbH testing of the MCAL AUTOSAR APIs is limited to generation and compilation.

Vendor AUTOSAR MCAL Module	API Name
Adc	▶ Adc_Init ▶ Adc_SetupResultBuffer ▶ Adc_DeInit ▶ Adc_StartGroupConversion ▶ Adc_StopGroupConversion

Vendor AUTOSAR MCAL Module	API Name
	▶ Adc_ReadGroup ▶ Adc_EnableHardwareTrigger ▶ Adc_DisableHardwareTrigger ▶ Adc_EnableGroupNotification ▶ Adc_DisableGroupNotification ▶ Adc_GetGroupStatus ▶ Adc_GetStreamLastPointer ▶ Adc_GetVersionInfo ▶ Adc_SetPowerState ▶ Adc_GetCurrentPowerState ▶ Adc_GetTargetPowerState ▶ Adc_PreparePowerState ▶ Adc_Main_PowerTransitionManager
Can	▶ Can_Init ▶ Can_GetVersionInfo ▶ Can_DeInit ▶ Can_SetBaudrate ▶ Can_SetControllerMode ▶ Can_DisableControllerInterrupts ▶ Can_EnableControllerInterrupts ▶ Can_CheckWakeup ▶ Can_GetControllerErrorState ▶ Can_GetControllerMode ▶ Can_GetControllerRxErrorCounter ▶ Can_GetControllerTxErrorCounter ▶ Can_Write ▶ Can_MainFunction_Write ▶ Can_MainFunction_Read ▶ Can_MainFunction_BusOff ▶ Can_MainFunction_Wakeup

Vendor AUTOSAR MCAL Module	API Name
	Can_MainFunction_Mode
Crc	Crc_CalculateCRC8 Crc_CalculateCRC8H2F Crc_CalculateCRC16 Crc_CalculateCRC16ARC Crc_CalculateCRC32 Crc_CalculateCRC32P4 Crc_CalculateCRC64 Crc_GetVersionInfo
Crypto	Crypto_Init Crypto_GetVersionInfo Crypto_ProcessJob Crypto_CancelJob Crypto_KeyElementSet Crypto_KeySetValid Crypto_KeyElementGet Crypto_KeyElementCopy Crypto_KeyElementCopyPartial Crypto_KeyCopy Crypto_KeyElementIdsGet Crypto_RandomSeed Crypto_KeyGenerate Crypto_KeyDerive Crypto_KeyExchangeCalcPubVal Crypto_KeyExchangeCalcSecret Crypto_CertificateParse Crypto_CertificateVerify Crypto_MainFunction
Dio	Dio_ReadChannel Dio_WriteChannel

Vendor AUTOSAR MCAL Module	API Name
	▶ Dio_ReadPort ▶ Dio_WritePort ▶ Dio_ReadChannelGroup ▶ Dio_WriteChannelGroup ▶ Dio_GetVersionInfo ▶ Dio_FlipChannel
Eep	▶ Eep_Init ▶ Eep_SetMode ▶ Eep_Read ▶ Eep_Write ▶ Eep_Erase ▶ Eep_Compare ▶ Eep_Cancel ▶ Eep_GetStatus ▶ Eep_GetJobResult ▶ Eep_GetVersionInfo ▶ Eep_MainFunction
Eth	▶ Eth_Init ▶ Eth_SetControllerMode ▶ Eth_GetControllerMode ▶ Eth_GetPhysAddr ▶ Eth_SetPhysAddr ▶ Eth_UpdatePhysAddrFilter ▶ Eth_WriteMii ▶ Eth_ReadMii ▶ Eth_GetCounterValues ▶ Eth_GetRxStats ▶ Eth_GetTxStats ▶ Eth_GetTxErrorCounterValues ▶ Eth_GetCurrentTime

Vendor AUTOSAR MCAL Module	API Name
	▶ Eth_EnableEgressTimeStamp ▶ Eth_GetEgressTimeStamp ▶ Eth_GetIngressTimeStamp ▶ Eth_ProvideTxBuffer ▶ Eth_Transmit ▶ Eth_Receive ▶ Eth_TxConfirmation ▶ Eth_GetVersionInfo ▶ Eth_MainFunction
Fee	▶ Fee_Init ▶ Fee_SetMode ▶ Fee_Read ▶ Fee_Write ▶ Fee_Cancel ▶ Fee_GetStatus ▶ Fee_GetJobResult ▶ Fee_InvalidateBlock ▶ Fee_GetVersionInfo ▶ Fee_EraseImmediateBlock ▶ Fee_JobEndNotification ▶ Fee_JobErrorNotification ▶ Fee_MainFunction
Fls	▶ Fls_Init ▶ Fls_Erase ▶ Fls_Write ▶ Fls_Cancel ▶ Fls_GetStatus ▶ Fls_GetJobResult ▶ Fls_Read ▶ Fls_Compare

Vendor AUTOSAR MCAL Module	API Name
	▶ Fls_SetMode ▶ Fls_GetVersionInfo ▶ Fls_BlankCheck ▶ Fls_MainFunction
Gpt	▶ Gpt_GetVersionInfo ▶ Gpt_Init ▶ Gpt_DeInit ▶ Gpt_GetTimeElapsed ▶ Gpt_GetTimeRemaining ▶ Gpt_StartTimer ▶ Gpt_StopTimer ▶ Gpt_EnableNotification ▶ Gpt_DisableNotification ▶ Gpt_SetMode ▶ Gpt_DisableWakeup ▶ Gpt_EnableWakeup ▶ Gpt_CheckWakeup ▶ Gpt_GetPredefTimerValue
Icu	▶ Icu_Init ▶ Icu_DeInit ▶ Icu_SetMode ▶ Icu_DisableWakeup ▶ Icu_EnableWakeup ▶ Icu_CheckWakeup ▶ Icu_SetActivationCondition ▶ Icu_DisableNotification ▶ Icu_EnableNotification ▶ Icu_GetInputState ▶ Icu_StartTimestamp ▶ Icu_StopTimestamp

Vendor AUTOSAR MCAL Module	API Name
	▶ Icu_GetTimestampIndex ▶ Icu_ResetEdgeCount ▶ Icu_EnableEdgeCount ▶ Icu_EnableEdgeDetection ▶ Icu_DisableEdgeDetection ▶ Icu_DisableEdgeCount ▶ Icu_GetEdgeNumbers ▶ Icu_StartSignalMeasurement ▶ Icu_StopSignalMeasurement ▶ Icu_GetTimeElapsed ▶ Icu_GetDutyCycleValues ▶ Icu_GetVersionInfo
Lin	▶ Lin_Init ▶ Lin_CheckWakeup ▶ Lin_GetVersionInfo ▶ Lin_SendFrame ▶ Lin_GoToSleep ▶ Lin_GoToSleepInternal ▶ Lin_Wakeup ▶ Lin_WakeupInternal ▶ Lin_GetStatus
Mcu	▶ Mcu_Init ▶ Mcu_InitRamSection ▶ Mcu_InitClock ▶ Mcu_DistributePllClock ▶ Mcu_GetPllStatus ▶ Mcu_GetResetReason ▶ Mcu_GetResetRawValue ▶ Mcu_PerformReset ▶ Mcu_SetMode

Vendor AUTOSAR MCAL Module	API Name
	▶ Mcu_GetVersionInfo ▶ Mcu_GetRamState
Ocu	▶ Ocu_Init ▶ Ocu_DeInit ▶ Ocu_StartChannel ▶ Ocu_StopChannel ▶ Ocu_SetPinState ▶ Ocu_SetPinAction ▶ Ocu_GetCounter ▶ Ocu_SetAbsoluteThreshold ▶ Ocu_SetRelativeThreshold ▶ Ocu_DisableNotification ▶ Ocu_EnableNotification ▶ Ocu_GetVersionInfo
Port	▶ Port_Init ▶ Port_SetPinDirection ▶ Port_RefreshPortDirection ▶ Port_GetVersionInfo ▶ Port_SetPinMode
Pwm	▶ Pwm_Init ▶ Pwm_DeInit ▶ Pwm_SetDutyCycle ▶ Pwm_SetPeriodAndDuty ▶ Pwm_SetOutputToIdle ▶ Pwm_GetOutputState ▶ Pwm_DisableNotification ▶ Pwm_EnableNotification ▶ Pwm_SetPowerState ▶ Pwm_GetCurrentPowerState ▶ Pwm_GetTargetPowerState

Vendor AUTOSAR MCAL Module	API Name
	▶ Pwm_PreparePowerState ▶ Pwm_GetVersionInfo ▶ Pwm_Main_PowerTransitionManager
Spi	▶ Spi_Init ▶ Spi_DeInit ▶ Spi_WriteIB ▶ Spi_AsyncTransmit ▶ Spi_ReadIB ▶ Spi_SetupEB ▶ Spi_GetStatus ▶ Spi_GetJobResult ▶ Spi_GetSequenceResult ▶ Spi_GetVersionInfo ▶ Spi_SyncTransmit ▶ Spi_GetHWUnitStatus ▶ Spi_Cancel ▶ Spi_SetAsyncMode ▶ Spi_MainFunction_Handling
Wdg	▶ Wdg_Init ▶ Wdg_SetMode ▶ Wdg_SetTriggerCondition ▶ Wdg_GetVersionInfo
Wdg	▶ Wdg_Init ▶ Wdg_SetMode ▶ Wdg_SetTriggerCondition ▶ Wdg_GetVersionInfo

Table 3.2. Third-party MCAL AUTOSAR APIs tested by Elektrobit Automotive GmbH

3.5. Third-party MCAL CDD APIs tested by Elektrobit Automotive GmbH

The following third-party MCAL CDD APIs are verified against the modules implementation. The scope of Elektrobit Automotive GmbH testing of the MCAL CDD APIs is limited to generation and compilation.

Vendor CDD MCAL Module	API Name
Uart	▶ I2c_Init ▶ I2c_DeInit ▶ I2c_PrepareSlaveBuffer ▶ I2c_SyncTransmit ▶ I2c_AsyncTransmit ▶ I2c_StartListening ▶ I2c_GetStatus ▶ I2c_GetVersionInfo
Uart	▶ I2s_Init ▶ I2s_DeInit ▶ I2s_SyncTransmit ▶ I2s_AsyncTransmit ▶ I2s_GetStatus ▶ I2s_AbortTransmit ▶ I2s_GetVersionInfo
Uart	▶ Mcl_Init ▶ Mcl_DeInit ▶ Mcl_SetDmaInstanceCommand ▶ Mcl_GetDmaInstanceStatus ▶ Mcl_SetDmaChannelCommand ▶ Mcl_GetDmaChannelStatus ▶ Mcl_SetDmaChannelGlobalList ▶ Mcl_SetDmaChannelTransferList ▶ Mcl_SetDmaChannelScatterGatherList ▶ Mcl_GetDmaChannelParam

Vendor CDD MCAL Module	API Name
	▶ Mcl_SetDmaChannelScatterGatherConfig ▶ Mcl_SetTrgMuxInput ▶ Mcl_SetTrgMuxLock ▶ Mcl_SelectCommonTimebase ▶ Mcl_GetVersionInfo
Uart	▶ Qdec_Init ▶ Qdec_DeInit ▶ Qdec_GetState ▶ Qdec_SetClockMode ▶ Qdec_CounterState ▶ Qdec_ResetState ▶ Qdec_DisableOverFlowNotification ▶ Qdec_EnableOverFlowNotification ▶ Qdec_SetMode ▶ Qdec_ReportEvents
Uart	▶ Rm_Init ▶ Rm_Mpu_SetRegionConfig ▶ Rm_Mpu_EnableRegion ▶ Rm_Mpu_SetAccessMode ▶ Rm_Mpu_GetErrorDetails ▶ Rm_GetVersionInfo
Uart	▶ Uart_Init ▶ Uart_Deinit ▶ Uart_SetBaudrate ▶ Uart_GetBaudrate ▶ Uart_Abort ▶ Uart_SetBuffer ▶ Uart_SyncSend ▶ Uart_SyncReceive ▶ Uart_AsyncReceive

Vendor CDD MCAL Module	API Name
	▶ Uart_AsyncSend ▶ Uart_GetStatus ▶ Uart_GetVersionInfo

Table 3.3. Third-party MCAL CDD APIs verified by Elektrobit Automotive GmbH

3.6. Third-party MCAL patches by Elektrobit Automotive GmbH

3.6.1. Use of original or patched version in third-party MCAL modules

The EB tresos Installer is the installation tool for the third-party MCAL modules. For more information on the EB tresos Installer, see 1.1_EB_tresos_installation_guide.pdf. In the EB tresos Installer, you can choose one of the following options:

- During installation it is possible to disable the update package. Afterwards, only original MCAL modules are part of your installation. For more information, see Chapter 6, “McalExt module description“.
- If you also install the update package, all changes by Elektrobit Automotive GmbH that are described in Section 6.3, “McalExt file description” are part of your EB tresos AutoCore installation.

  Additionally, with this update package you can switch between the original version and the patched version of each MCAL module.

The update package includes the `perform_MCAL_change.bat` batch file in the `McalExt` module. Use this batch file to switch from one version to another.

In the `perform_MCAL_change.bat`, you need to specify one of the following parameters:

- `EB_update`: Update the files in the module with content of Elektrobit Automotive GmbH.
- `origin`: Reset the files in the module with vendor content.

WARNING **Changes due to execution of the batch file affect all projects**

The changes affect all projects that use the changed module. Therefore, execute the batch-file before generating code for your project.

4. Overview of McalExt documentation

This documentation is target-specific and derivative-specific.

This user guide describes the concepts and the configuration of the module:

- `McalExt`

4.1. Background information

`McalExt` is a wrapper module that connects the vendor MCAL delivery, EB tresos Studio, and the Elektrobit Automotive GmbH (EB) build environment. It allows you to use the vendor MCAL with EB tresos Studio with as few modifications as possible.

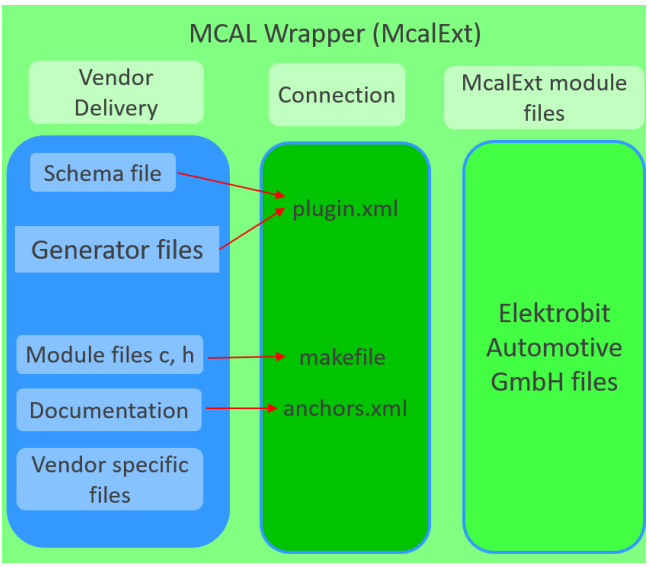

Figure 4.1. MCAL Wrapper (McalExt) overview

The MCAL Wrapper (`McalExt`) documentation also:

- provide information about the vendor delivery package, see Section 6.1, “Vendor delivery package”
- provide information about the connection with the vendor delivery package, see Section 6.2, “Connection with vendor delivery package”
- provide information about Elektrobit Automotive GmbH files, see Section 6.3, “McalExt file description”

5. Using the McalExt module

5.1. Add McalExt and MCAL modules to project

- Add the `McalExt` module to the project configuration.

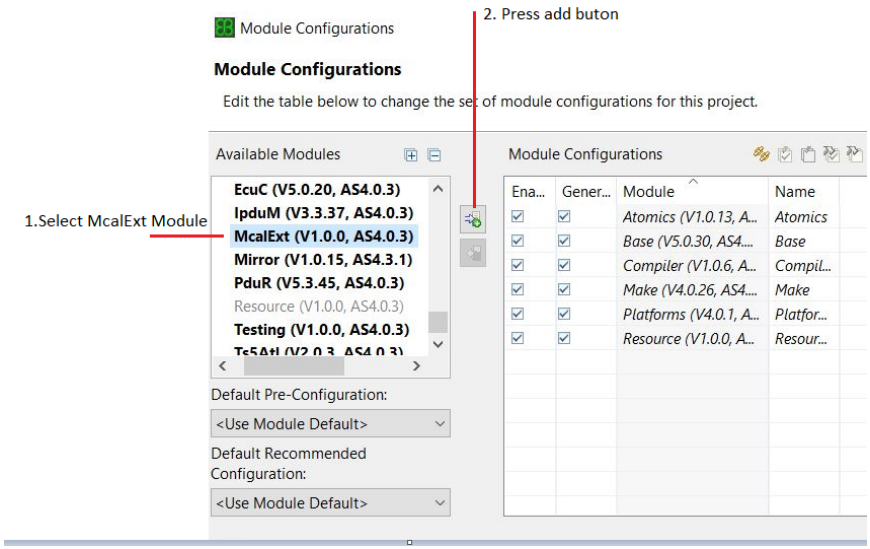

Figure 5.1. Add McalExt Module

- Add the needed MCAL modules to the project configuration

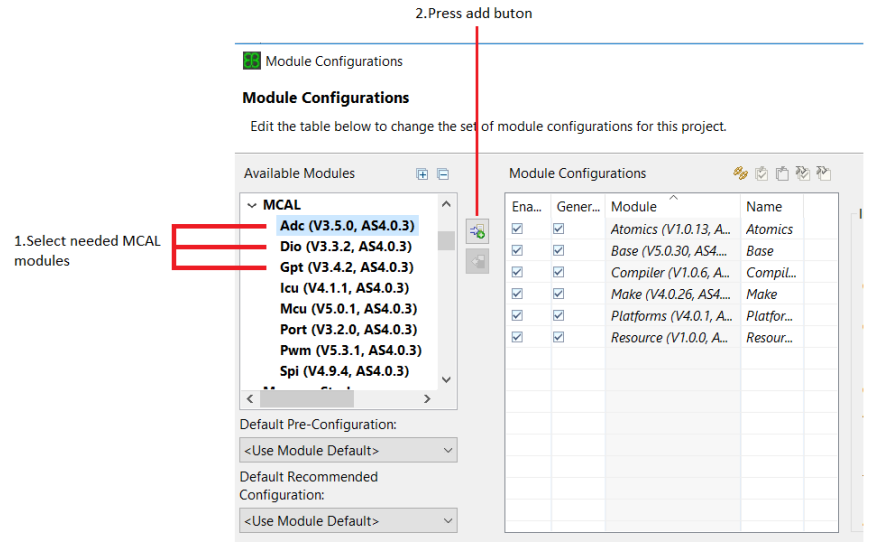

Figure 5.2. Add Mcal Module

5.2. EB build environment

In order to use the `McalExt` module in a project that is using the EB build environment, you must do the following:

- Add the `McalExt` module to the project configuration. see Figure 5.1, “Add McalExt Module”
- Add the needed MCAL modules to the project configuration. see Figure 5.2, “Add Mcal Module”

5.3. User build environment

When a different build enviroment (other than the EB build environment) is used in the project, you must do the following:

- Add the `McalExt` module to the project configuration. see Figure 5.1, “Add McalExt Module”
- Add the needed MCAL modules to the project configuration. see Figure 5.2, “Add Mcal Module”
- Add all the files that are configured in the module makefiles and all files configured in the IncludePaths and FilesToBuild from `McalExt` module configuration, see IncludePaths and FilesToBuild description in Section 6.3, “McalExt file description”

6. McalExt module description

6.1. Vendor delivery package

This represents the package that is delivered by different MCAL vendors (e.g. Infineon, Renesas, NXP, etc.) and contains different MCAL modules. For the MCAL modules and version that are integrated in this release please see Section 3.3, “Third-party MCAL modules”.

6.2. Connection with vendor delivery package

The vendor delivery has a structure that cannot be used directly in the EB build environment. The `McalExt` module is introduced to make the connection between the EB build environment and the vendor delivery. The connection is made in the following files of the `McalExt` module:

`plugin.xml` connection, see Section 6.2.1, “`plugin.xml` connection description”

`makefile` connection, see Section 6.2.2, “Makefile connection description”

`anchors.xml` connection, see Section 6.2.3, “`anchors.xml` connection description”

6.2.1. `plugin.xml` connection description

- Schema file, e.g.:

```
<schema>
<manager class="dreisoft.tresos.autosar2.resourcehandling.AutosarSchemaManager"/>
<!-- Define the file(s) from which to load the schemas -->
<resource value="MCAL_Delivery/PathToSchemaFile/ModuleName.xdm"  type="xdm"/>
</schema>
```

  PathToSchemaFile - represents the path to where the module schema file is located in the MCAL vendor delivery.

  ModuleName - represents the name of the schema file that should be used.

- Code Generator e.g.:

```
<!-- common template path parameters -->
```

```
<parameter name="templates"
mode="generate,verify" value="MCAL_Delivery/PathToGenerator"/>
```

PathToGenerator - represents the path where the generator is located in the vendor delivery.

- Ant code generator, e.g.:

```
<generator moduleId="ModuleId"
class="dreisoft.tresos.generator.ng.api.NGGenerator"
id="ModuleId_UniqueNGGeneratorId"
step="post"> <!-- run after code-generation -->
<parameter name="buildfile" value="MCAL_Delivery/PathToAntGenerator/AntGeneratorFile.xml"/>
</generator>
```

PathToAntGenerator - represents the path where the ant generator file is located in the vendor delivery.

AntGeneratorFile - represents the name of the ant generator file delivered by the vendor.

NOTE This is applicable only if vendor provided the `AntGeneratorFile.xml` file.

6.2.2. Makefile connection description

For each MCAL module that is integrated in the `McalExt` a make folder exists that contains the makefiles for the respective MCAL module:

- `Module_defs.mak` file - registers the file(s) that are present in vendor delivery and the files that are generated for this module.

```
McalExt_GEN_FILES += $(McalExt_OUTPUT_PATH)\inc\ModuleName_Cfg.h
McalExt_GEN_FILES += $(McalExt_OUTPUT_PATH)\inc\ModuleName_PBcfg.h
McalExt_GEN_FILES += $(McalExt_OUTPUT_PATH)\src\ModuleName_PBcfg.c
CC_INCLUDE_PATH   += $(McalExt_CORE_PATH)\MCAL_Delivery\PathToHeaderFiles
```

ModuleName - represents the name of the header files that are generated.

PathToHeaderFiles - represents the path from the vendor delivery where the static header files are located.

- `Module_rules.mak` file - registers the specific module file(s) that are needed for compilation.

```
Module_src_FILES += $(McalExt_CORE_PATH)\MCAL_Delivery\PathToSourceFile\Module_Name.c
Module_src_FILES += $(McalExt_OUTPUT_PATH)\src\ModuleName_PBcfg.c
```

ModuleName - represents the name of the C files that are generated or are present in the vendor delivery.

PathToSourceFile - represents the path from the vendor delivery where the C static files are located.

All the MCAL modules makefiles will be included in `McalExt_defs.mak` and `McalExt_rules.mak` files only if the respective module is used in the EB tresos Studio project:

- `McalExt_defs.mak` file

```
ifeq ($(McalExt_Can_USED),true)
ifeq ($(Can_VARIANT),ModuleNameVariant)
include $(McalExt_CORE_PATH)\make\make_Can\Can_defs.mak
endif
endif
```

- `McalExt_rules.mak` file

```
ifeq ($(McalExt_Can_USED),true)
ifeq ($(Can_VARIANT),ModuleNameVariant)
LIBRARIES_TO_BUILD     += Can_src
include $(McalExt_CORE_PATH)\make\make_Can\Can_rules.mak
endif
endif
```

ModuleNameVariant - represents the MCAL module name that is used in the EB tresos Studio project.

6.2.3. `anchors.xml` connection description

- `anchors.xml` file registers the MCAL documentation that is shown in the EB tresos Studio help window, e.g:

```
<topic label="DocName" href="PathToDoc/DocName"/>
<topic label="DocName" href="PathToDoc/DocName"/>
<topic label="DocName" href="PathToDoc/DocName"/>
```

PathToDoc - represents the path where the MCAL documents are located.

DocName - represents the name of the MCAL documents.

6.3. McalExt file description

Some patches made by Elektrobit Automotive GmbH are due to missing or incomplete files in the MCAL vendor delivery. Additionally, some EB tresos Studio features are enabled. These patches are separated from the original installation files.

- `config` - In this folder there is located the `McalExt` wrapper schema file `McalExt.xdm` that allows you to configure the following parameters that can be used in the project:
  - **PlatformModuleDefine** – you can configure defines that will be generated in `Platforms_Modules.h` file and that can be used in the project, e.g.: configure an Mcu configuration pointer that will be used in the Mcu_Init() function.

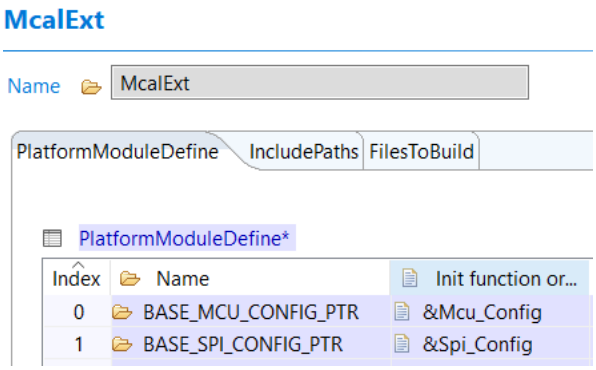

Figure 6.1. Platform Module Define parameter configuration

Generated file content example:

```
#define BASE_MCU_CONFIG_PTR              &Mcu_Config
#define BASE_SPI_CONFIG_PTR              &Spi_Config
```

Usage of defines (EcuM_DriverInitListOne()), e.g.:

```
/* *** Call service Init of module Mcu *** */
Mcu_Init(BASE_MCU_CONFIG_PTR);
```

  - **IncludePaths** – allows you to configure different paths that need to be included by the build environment. This will be generated in the McalExtWrapper.mak file:

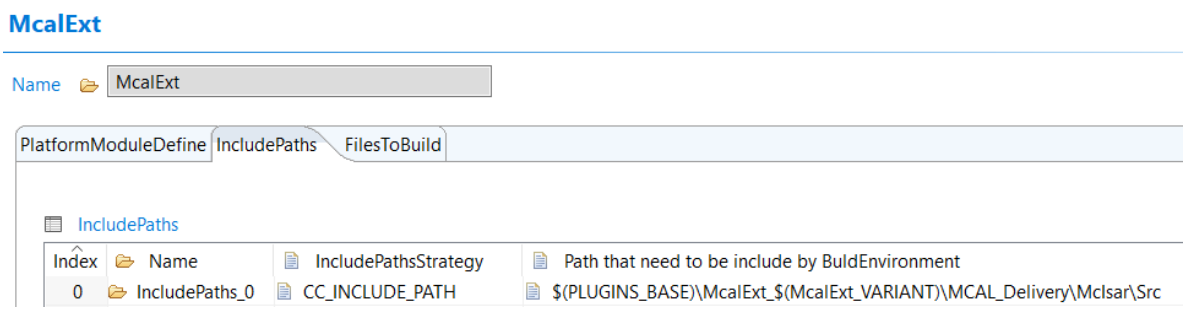

Figure 6.2. Path that needs to be included by BuildEnvironment

Generated content in `McalExtWrapper.mak` file:

```
CC_INCLUDE_PATH += $(PLUGINS_BASE)\McalExt_$(McalExt_VARIANT)\MCAL_Delivery\McIsar\Src
```

- **FilesToBuild** - allows you to configure different files that need to be compiled. This will be generated in the `McalExtWrapper.mak` file:

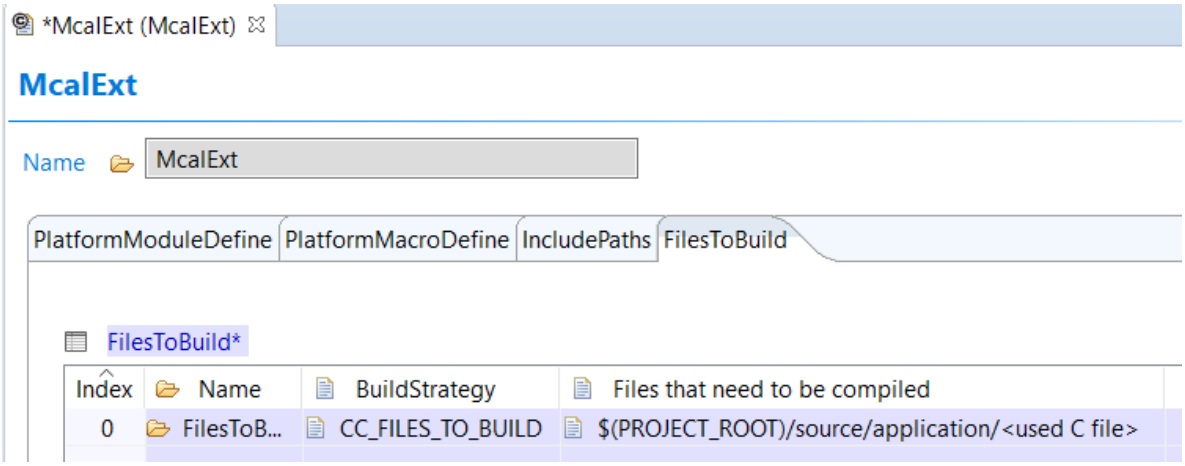

Figure 6.3. Files that need to be compiled

Generated content in `McalExtWrapper.mak` file:

```
CC_FILES_TO_BUILD  += $(PROJECT_ROOT)\source\application\Eb_Intgr_BswM_UserCallouts.c
```

- `config_ext` - In this folder there are present `Mcu` preconfiguration and recommended configurations:
  - Preconfiguration files that contain the parameter configuration value that should not be modified e.-g. McuResetReasonConf.
  - Recommended configuration – the configuration that was validated by Elektrobit Automotive GmbH while performing IP3/QP2 (if this was ordered), e.g.: Mcu recommended configuration that contains

the clock configuration and other `Mcu` related parameter configuration. You can decide if the recommended configuration is used or not.

- `doc` - In this folder there are added MCAL module documentations if aplicable.
- `generate` - Contains the files that will be generated by `McalExt` wrapper plugin.
- `include` - Contains the header file(s) that are created/patched by Elektrobit Automotive GmbH.
- `make` - Contains the makefiles for integrated MCAL module, see Section 6.2.2, “Makefile connection description”
- `MCAL_Delivery` - Contains the vendor delivery files.
- `resources` - Includes several XML-based service needs assistant or properties files that are provided by Elektrobit Automotive GmbH. These files support you to faster complete a valid configuration.
  - `Dem_Events.xml` – Dem event generation in the EB tresos AutoCore Generic `Dem` module.
  - EcuM initialization in the EB tresos AutoCore Generic `EcuM` module.
  - SchM Main function handling in the EB tresos AutoCore Generic `Rte` module.
- `swcd` - Includes the BSWMD files that are mandatory since AUTOSAR 4.0. Those files are used by BSW modules and EB tresos Studio wizards provided by Elektrobit Automotive GmbH.
  - Generation of exclusive areas in the EB tresos AutoCore Generic `Rte` module.
  - Mapping of the Main function in the EB tresos AutoCore Generic `Rte` module.
  - Generation of MemMap header file(s) in EB tresos AutoCore Generic `MemMap` module.
- `src` - Contains source file(s) that should be compiled, created by Elektrobit Automotive GmbH.

6.4. MCAL version update

When the MCAL version is updated, you must verify the following:

- Check if all the used files are at the same location. If the vendor does not modify the MCAL installed structure, then the used path should be the same.
- During integration, Elektrobit Automotive GmbH applies patches on the vendor files due to some bugs or due to some incompatibility. These patches are located in the files with the `.EB_update` extension. When you install the new MCAL version, you must to verify if those patches need to be applied on the new MCAL version files.