Elektrobit

MCAL Wrapper (McalExt) documentation

for TRICORE TC38XQ

product release 8.8.4

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>/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 TASKING_TriCore-VX_v6.3r1p2

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**
\ctc\bin\ctc.exe	`-core=tc1.6.2 --iso=99 -O2 --eabi=BCFHNSW -AGKpvX --trade-off=2 --fp-model=1 --switch=auto --integer-enumeration --default-near-size=0 --global-type-checking -DUSE_TASKING_INIT=1 -DOSB_TOOL=OSB_tasking`
Assembler	**Options**
\ctc\bin\astc.exe	`-core=tc1.6.2 --list-format=1 --optimize=gs`
Linker	**Options**
\ctc\bin\ltc.exe	`-OcLtXY --core=mpe:vtc --global-type-checking -lc`

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 TC38XQ 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 MC-ISAR_AS422_TC3xx_BASIC_2.0.0 MC-ISAR_AS422_TC3xx_-COM-E_2.0.0 MC-ISAR_AS422_TC3xx_CD_2.0.0 MC-ISAR_AS422_TC3xx_Demo_2.0.0 from Infineon.

This release of EB tresos AutoCore is tested only with sub-derivative TC387.

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	4.2.2	4.2.2	20.0.2	Infineon
Can	4.2.2	4.2.2	20.0.2	Infineon
CanTrcv	4.2.2	4.2.2	20.0.1	Infineon
CanTrcv	4.2.2	4.2.2	20.0.1	Infineon
Crc	4.2.2	4.2.2	20.0.2	Infineon
Dio	4.2.2	4.2.2	20.0.1	Infineon
Dma	4.2.2	4.2.2	20.0.2	Infineon

Module name	AUTOSAR version and revision	SWS version and revision	Module version	Supplier
Dsadc	4.2.2	4.2.2	20.0.1	Infineon
Eth	4.2.2	4.2.2	20.0.1	Infineon
Fee	4.2.2	4.2.2	20.0.2	Infineon
Fls	4.2.2	4.2.2	20.0.1	Infineon
FlsLoader	4.2.2	4.2.2	20.0.0	Infineon
Fr	4.2.2	4.2.2	20.0.1	Infineon
Gpt	4.2.2	4.2.2	20.0.1	Infineon
Hssl	4.2.2	4.2.2	20.0.1	Infineon
I2c	4.2.2	4.2.2	20.0.2	Infineon
Icu	4.2.2	4.2.2	20.0.1	Infineon
Iom	4.2.2	4.2.2	20.0.0	Infineon
Lin	4.2.2	4.2.2	20.0.2	Infineon
McalLib	4.2.2	4.2.2	20.0.1	Infineon
Mcu	4.2.2	4.2.2	20.0.2	Infineon
Ocu	4.2.2	4.2.2	20.0.1	Infineon
Port	4.2.2	4.2.2	20.0.1	Infineon
Pwm	4.2.2	4.2.2	20.0.2	Infineon
Sent	4.2.2	4.2.2	20.0.1	Infineon
Smu	4.2.2	4.2.2	20.0.2	Infineon
Spi	4.2.2	4.2.2	20.0.2	Infineon
Stm	4.2.2	4.2.2	20.0.0	Infineon
Uart	4.2.2	4.2.2	20.0.1	Infineon
Wdg	4.2.2	4.2.2	20.0.1	Infineon
ResourceM	4.2.2	4.2.2	20.0.0	Infineon

Table 3.1. Third-party hardware-dependent modules

3.4. Third-party MCAL patches by Elektrobit Automotive GmbH

3.4.1. 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.

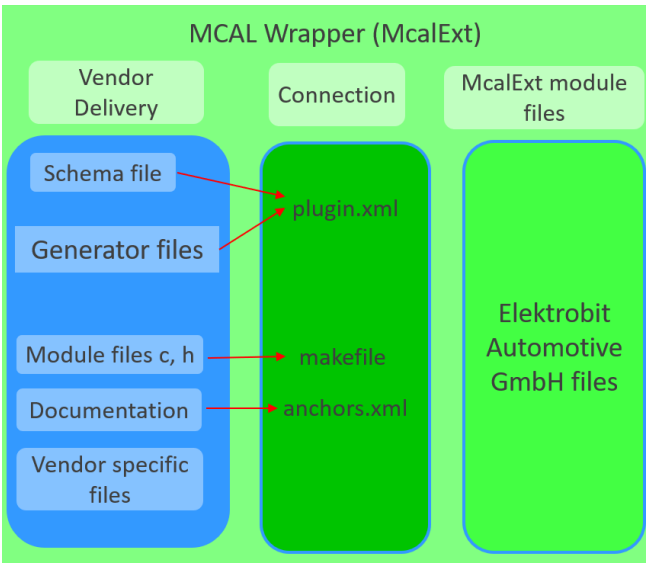

Figure 4.1. MCAL Wrapper (McalExt) overview

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.

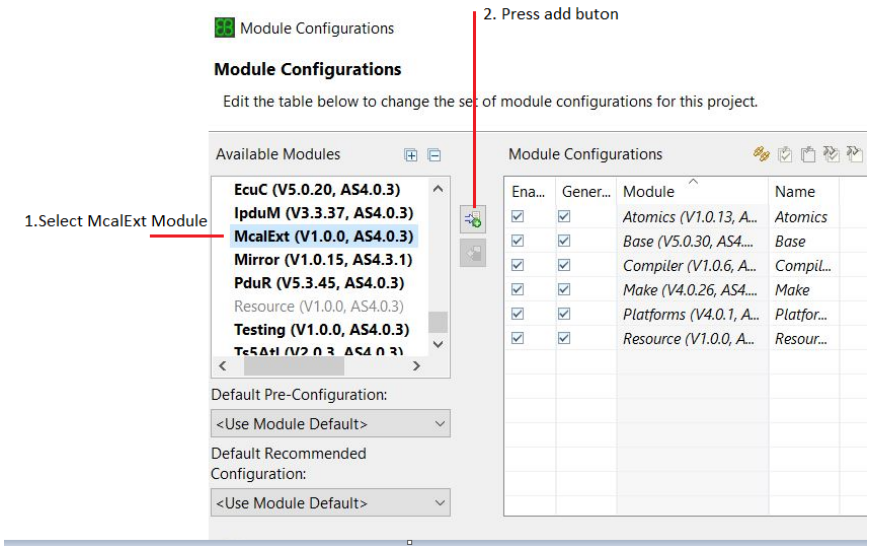

Figure 5.1. Add McalExt Module

- Add the needed MCAL modules to the project configuration

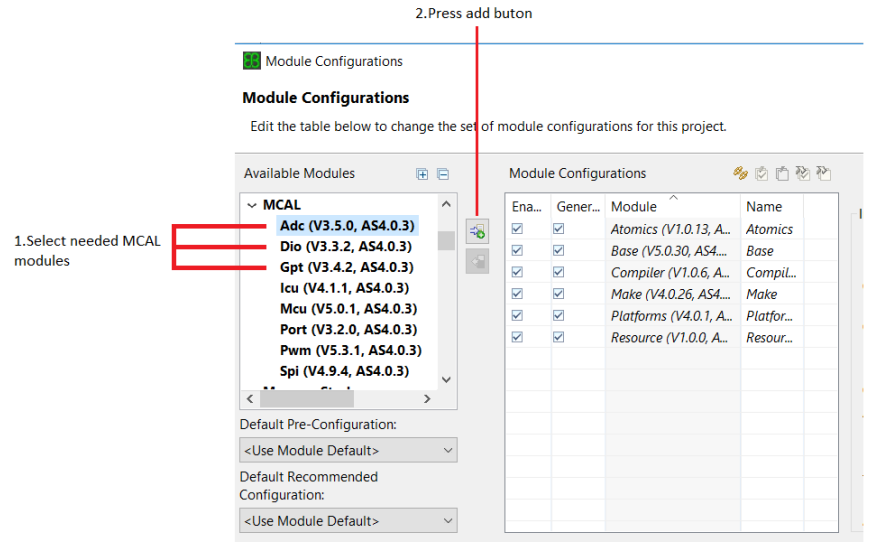

Figure 5.2. Add Mcal Module

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.

- The `McalExt` wrapper schema file `McalExt.xdm` located in `config` folder 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.

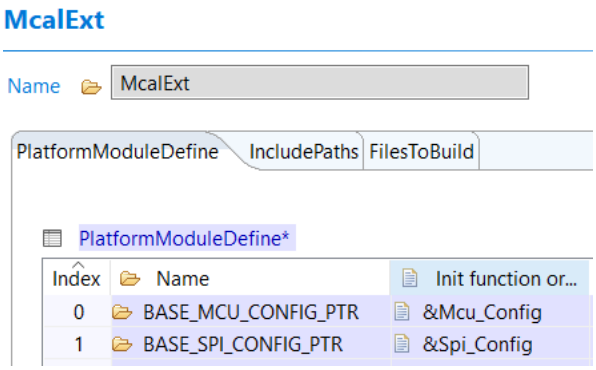

Figure 6.1. Platform Module Define parameter configuration

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:

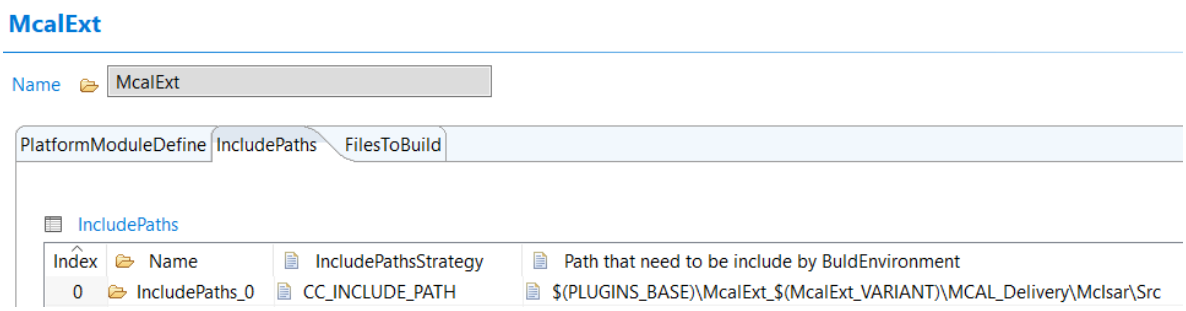

Figure 6.2. Path that needs to be included by BuildEnvironment

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:

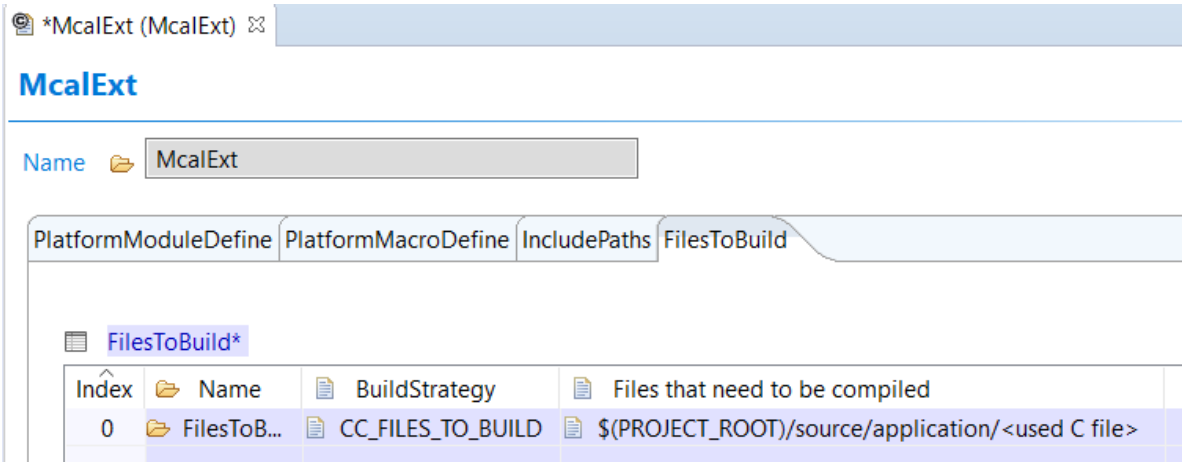

Figure 6.3. Files that need to be compiled

Generated content in `McalExtWrapper.mak` file:

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

- Preconfiguration and recommended configuration – The config_ext folder can contain files for different MCAL modules, e.g:
  - 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.

- `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.
- `include` - contains header file(s) that are created/patched by Elektrobit Automotive GmbH.
- `src` - contains source file(s) that should be compiled, created by Elektrobit Automotive GmbH.
- `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.

6.4. Use Spi template for asynchronous mode

The SPI uses the service of the DMA for data transmission in asynchronous mode. The Dma Channels used by Spi are configurable and the allocation of the same is done in MCU. Each channel can work in software triggered mode or hardware triggered mode. For SPI to use DMA channel, the channel should be configured as hardware trigger type.

The SPI uses the service of the DMA for data transmission in asynchronous mode. The Dma Channels used by Spi are configurable and the allocation of the same is done in MCU. Each channel can work in software triggered mode or hardware triggered mode. For SPI to use DMA channel, the channel should be configured as hardware trigger type.

In this implementation the Combined Software/Hardware Controlled Mode will be used. Hence the first DMA transfer will be triggered by software when setting the corresponding request channel bit in the Software Transaction Request Register, while the hardware requests are still disabled. The subsequent DMA transfers are triggered by the corresponding hardware request, this happens after the hardware requests have been enabled.

The QSPI Peripheral's RX or TX Interrupt Line is used to trigger the DMA transactions. Each QSPI Peripheral uses two DMA channels to trigger reception and transmission respectively.

The DMA Mechanism is used for the asynchronous transmission mode. It is not used during synchronous transmission mode. The SPI handler driver initializes the DMA channels used.

The DMA channel number used by SPI will be used as the priority number as per HW recommendation.

QSPI RX and TX have to be mapped to DMA - not to interrupts. Infineon IRQ module is capable to map interrupt nodes to CPU0/CPU1/CPU2 interrupts or to DMA. This functionality have to be implemented in the user application as part of integration code with EB's Autocore OS or Safety OS and with AutoCore Generic.

Please copy following files

- plugins\McalExt_TS_*\templates\Spi_UserCode.h
- plugins\McalExt_TS_*\templates\Spi_UserCode.c

into your user application and add them to your build environment.

Please change following macros in Spi_UserCode.h to your needs

- #define QSPI_SCR_RX_PRIO (4u)
- #define QSPI_SCR_TX_PRIO (5u)

This example will map QSPI2RX to DMA channel 4 and QSPI2TX to DMA channel 5.

In order to map QSPI RX and TX the template function Spi_InitDMAUsr should be executed after Spi_Init function. Therefore, please add include Spi_UserCode.h in BswM_UserCallouts.c after include of Spi.h. Additionally add function call Spi_InitDMAUsr after Spi_Init, e.g. SPI_INIT_FUNC function macro.

6.5. 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.