MCAL User Manual for McalLib

32-bit TriCore^{TM} AURIX^{TM} TC3xx microcontroller

About this document

Scope and purpose

This User Manual is intended to enable users to integrate the Microcontroller Abstraction Layer (MCAL) software for the TriCore^{TM} AURIX^{TM} family of 32-bit microcontrollers.

This document describes responsibilities of integrator in-charge of integrating MCAL software with the basic software (BSW) stack. This document also provides detailed information on safety, configuration and functions along with examples of usage of significant features.

Note: Detailed information about package installation, safety and other generic information that are common across all modules are provided in MCAL User Manual General.

Intended audience

This document is intended for anyone using the McalLib module of the TC3xx MCAL software.

Document conventions

Table 1 Conventions

Convention	Explanation
Bold	Emphasizes heading levels, column headings, table and figure captions, screen names, windows, dialog boxes, menus, sub-menus
Italics	Denotes variable(s) and reference(s)
`Courier New`	Denotes APIs, functions, interrupt handlers, events, data types, error handlers, file/folder names, directories, command line inputs, code snippets
>	Indicates that a cascading sub-menu opens when you select a menu item
[cover parentID=<alpha numeric value>]	Used for traceability completeness. Reader should ignore these.

Reference documents

This User Manual should be read in conjunction with the following documents:

- AURIX^{TM} TC3xx MCAL User Manual General

Table of contents

1 McalLib driver

1.1 User information

1.1.1 Description

The MCAL Library (MCALLIB) provides a set of utility routines for use by the MCAL drivers. The services provided are ENDINIT management, global-local memory address translation, timer based delay, retrieval of CPU identifier, abstraction of TriCore-intrinsic instruction and spinlock.

1.1.2 Hardware-software mapping

This section describes the system view of the MCALLIB driver and peripherals administered by it.

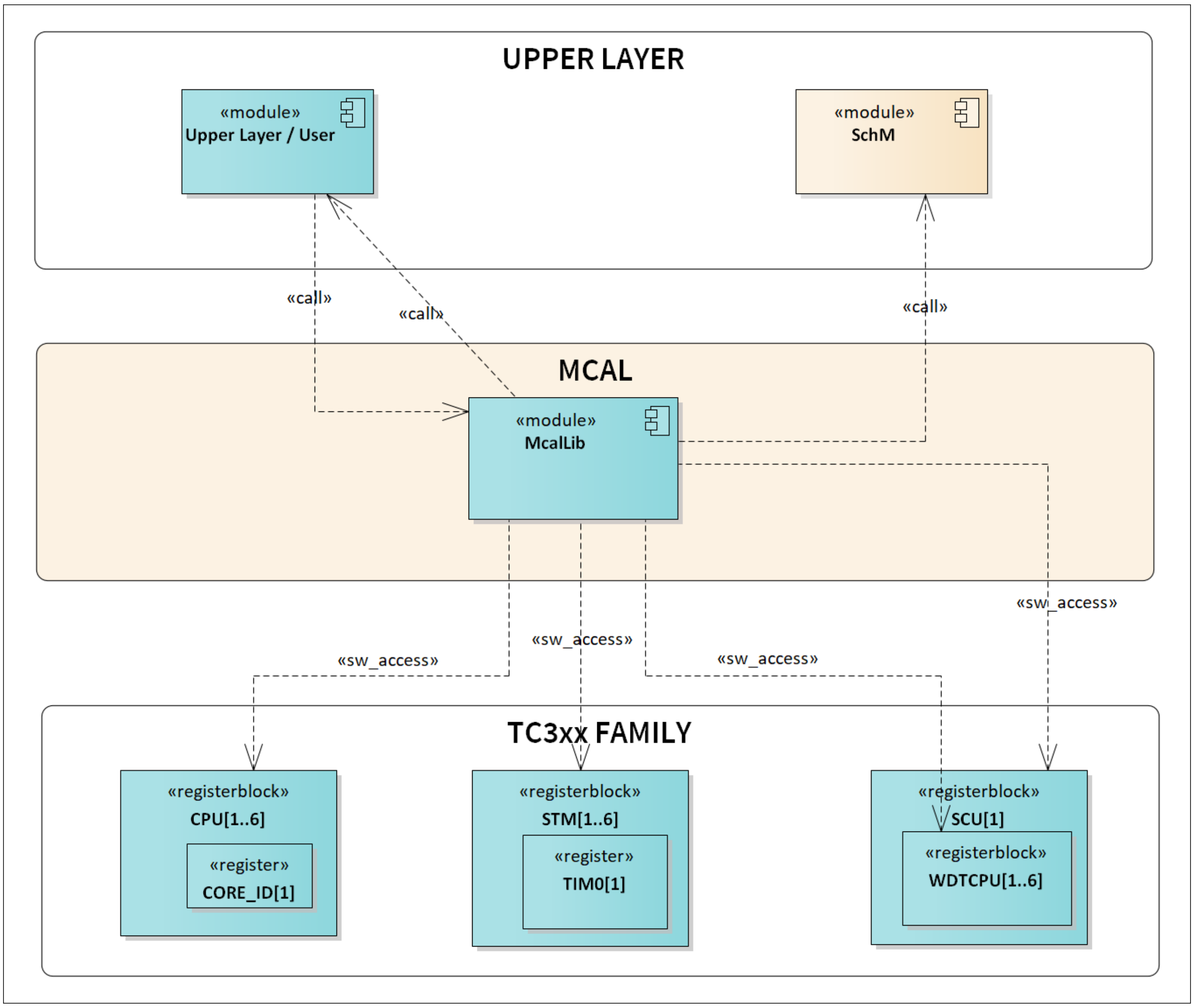

Figure 1 **Mapping of hardware-software interfaces**

1.1.2.1 CPU: primary hardware peripheral

Hardware functional features

The MCALLIB driver uses the CPU to retrieve the CPU ID on which the code is executing. The key hardware functional features used by the driver is:

- CPU registers

The unsupported features of the CPU is:

- Debug support

Users of the hardware

The MCALLIB driver exclusively utilizes the CPU module.

Hardware diagnostic features

The SMU alarms configured for the CPU are not monitored by the MCALLIB driver.

Hardware events

Hardware events from CPU are not used by the MCALLIB driver.

1.1.2.2 SCU: dependent hardware peripheral

Hardware functional features

The MCALLIB driver depends on the SCU IP for the clock functionality. The driver requires the fSPB and fSTM clock signals for functioning.

ENDINIT feature of the SCU is implemented by the MCALLIB driver.

Users of the hardware

The SCU IP supplies clock for all the peripherals. The MCU driver is responsible for configuring the clock tree. To avoid conflicts due to simultaneous writes, update to all the ENDINIT protected registers is performed using the MCALLIB APIs.

MCALLIB and WDG driver both update the WDT peripheral related registers. In order to avoid register corruption due to concurrent writes, all the writes to the WDT registers is performed under the same critical section by MCALLIB and WDG driver both.

Hardware diagnostic features

The SMU alarms configured for the SCU IP are not monitored by the MCALLIB driver.

Hardware events

Hardware events from the SCU are not used by the MCALLIB driver.

1.1.2.3 STM: primary hardware peripheral

Hardware functional features

The MCALLIB driver only reads the System Timer Bits [31:0]. The key hardware functional features used by the driver is:

- Free-running system timer

The unsupported features of the STM is:

- Compare match operation

Users of the hardware

The MCALLIB driver provides API to read current STM tick count. The MCU, FR, Eth, FlsLoader, UART and CanTrcv_17_W9255 driver uses the API for delay generation.

Hardware diagnostic features

Not applicable.

Hardware events

Hardware events from STM are not used by the MCALLIB driver.

1.1.3 File structure

1.1.3.1 C file structure

The section provides details of the C files of the MCALLIB driver.

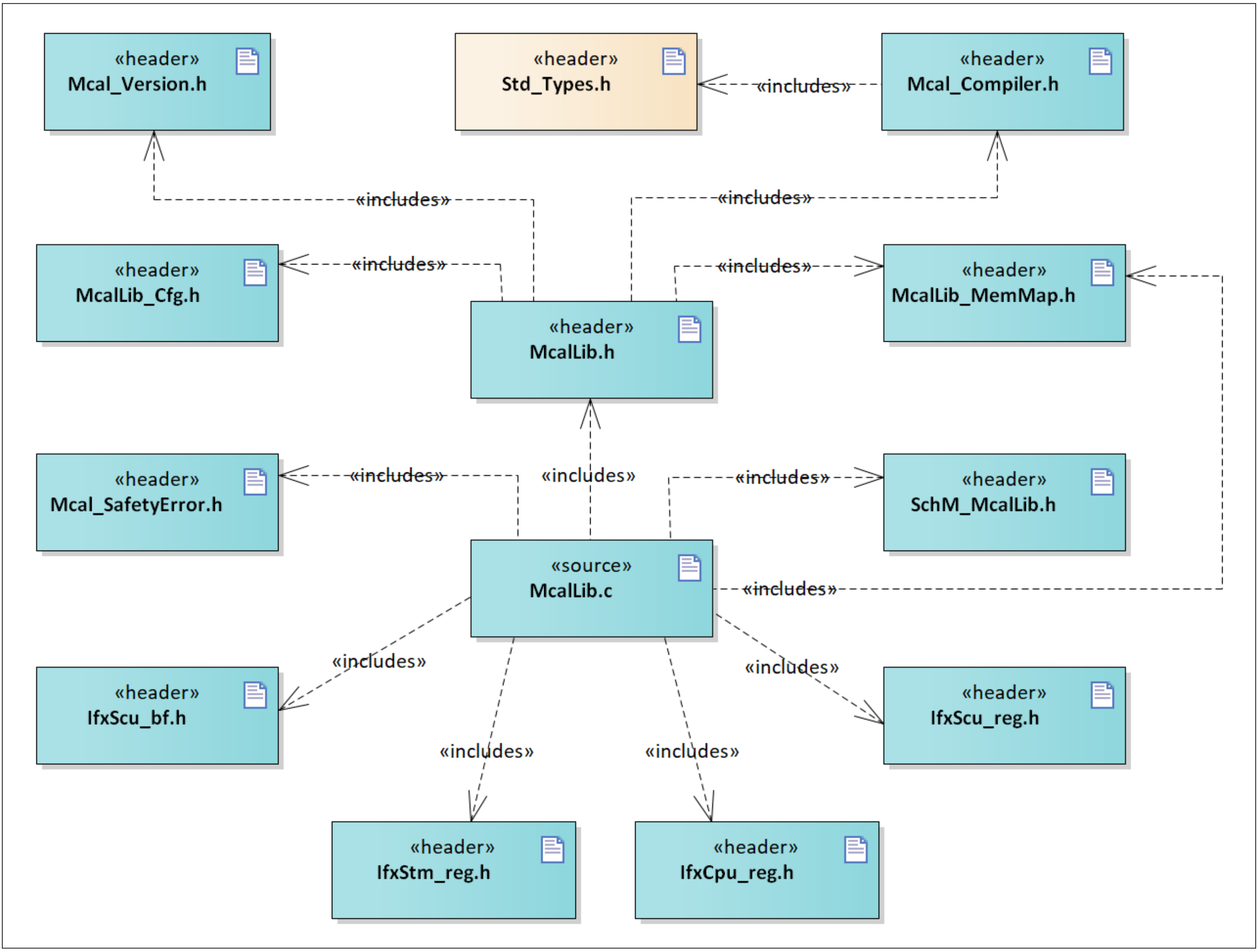

Figure 2 McalLib_C_File_Structure-1.png

Table 2 C file structure

File name	Description
`IfxCpu_reg.h`	SFR header file for CPU

Table 2 C file structure (continued)

File name	Description
`IfxScu_bf.h`	SFR header file for SCU
`IfxScu_reg.h`	SFR header file for SCU
`IfxStm_reg.h`	SFR header file for STM
`McalLib.c`	Static source code for the MCALLIB.
`McalLib.h`	Static header file defining prototypes of data structure and APIs exported by the MCALLIB.
`McalLib_Cfg.h`	Generated header file providing information on number of cores, DSPR, PSPR (start and end addresses) and system and backup clock information.
`McalLib_MemMap.h`	Header file containing the memory section definitions used by the MCALLIB.
`Mcal_Compiler.h`	Header file providing abstraction for TriCore^{TM}-intrinsic instruction.
`Mcal_SafetyError.h`	Header file containing the prototype of the API for reporting safety-related errors
`Mcal_Version.h`	Header file providing macros related to the AUTOSAR version 4.4.0 or 4.2.2. *Note: There are two instances of this header file, one for each AUTOSAR version. However based on the build system, only one file will be included by MCALLIB driver at any point of time.*
`SchM_McalLib.h`	Header file providing prototype of SchM interfaces needed by the MCALLIB driver.
`Std_Types.h`	Standard type declaration file as defined by AUTOSAR. It is independent of compiler or platform.

1.1.3.2 Code generator plugin files

The section provides details of the code generator plugin files of the MCALLIB driver.

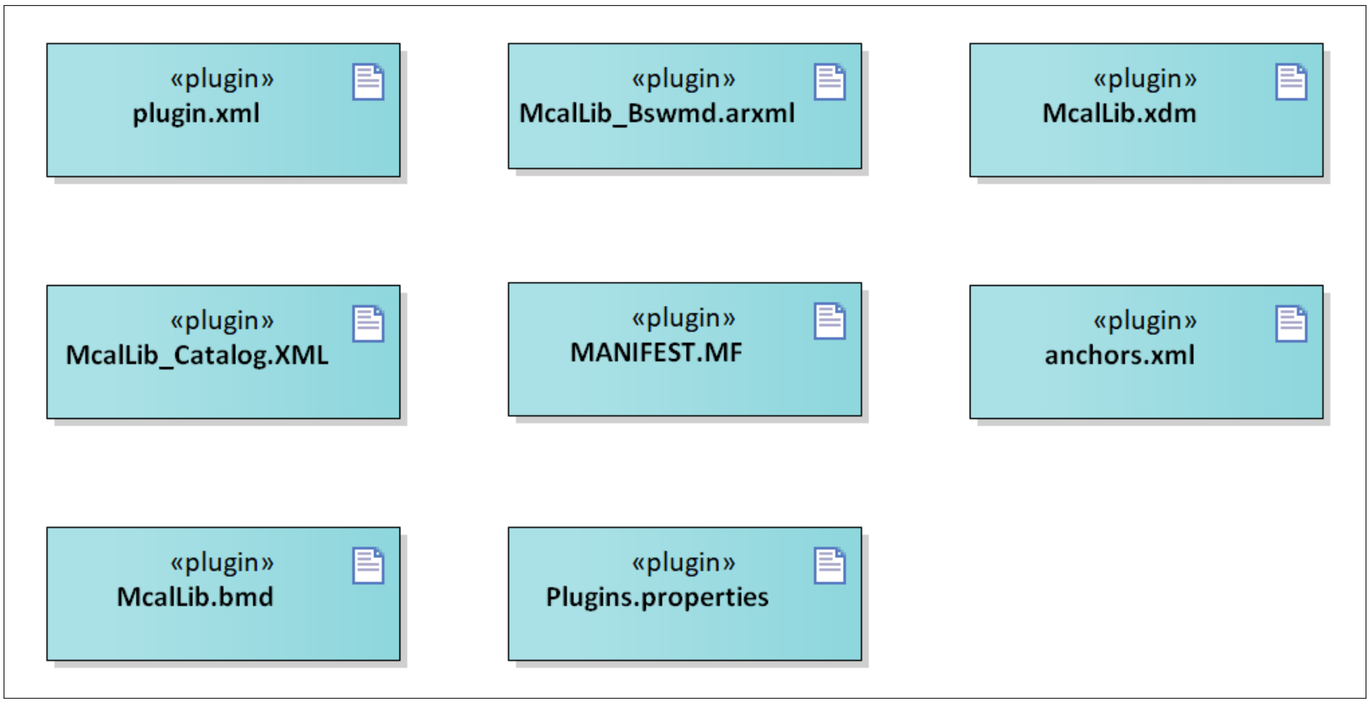

Figure 3 McalLib_Code_Generator_Plugin_Files-1.png

Table 3 Code generator plugin files

File name	Description
`MANIFEST.MF`	Tresos plugin support file containing the metadata for the MCALLIB driver.
`McalLib.bmd`	AUTOSAR format XML data model schema file (for each device).
`McalLib.xdm`	Tresos format XML data model schema file.
`McalLib_Bswmd.arxml`	AUTOSAR format module description file.
`McalLib_Catalog.XML`	AUTOSAR format catalog file as per catalog_V3_0_0.ml.xsd.
`Plugins.properties`	Tresos plugin support file for the MCALLIB driver.
`anchors.xml`	Tresos anchors support file for the MCALLIB driver.
`plugin.xml`	Tresos plugin support file for the MCALLIB driver.

1.1.4 Integration hints

This section lists the key points, that an integrator or user of the MCALLIB must consider. In general, the APIs of MCALLIB driver may be invoked from several CPU cores in parallel with some restrictions, which are also described in this section.

1.1.4.1 Intergration with AUTOSAR stack

This section lists the modules, which are not part of MCAL, but required to integrate the MCALLIB driver.

- **EcuM**

  EcuM module is not required for integrating the MCALLIB driver.

- **Memory mapping**

  Memory mapping is a concept from AUTOSAR that allows relocation of text, variables, constants and configuration data to user-specific memory regions. To achieve this, all the re-locatable elements of the driver are encapsulated in different memory-section macros. These macros are defined in the `McalLib_MemMap.h` file.

  The `McalLib_MemMap.h` file is provided in the MCAL package as a stub code. The integrator must place appropriate compiler pragmas within the memory-section macros. The pragmas ensure that the elements are re-located to the correct memory region. A sample implementation listing the memory-section macros is shown as followos:

```
/**** GLOBAL RAM DATA -- NON-CACHED LMU ****/
#if defined MCALLIB_START_SEC_VAR_CLEARED_ASIL_B_GLOBAL_32
 /*****User pragmas here for Non-cached LMU*****/
 #undef MCALLIB_START_SEC_VAR_CLEARED_ASIL_B_GLOBAL_32
 #undef MEMMAP_ERROR
#elif defined MCALLIB_STOP_SEC_VAR_CLEARED_ASIL_B_GLOBAL_32
 /*****User pragmas here for Non-cached LMU*****/
 #undef MCALLIB_STOP_SEC_VAR_CLEARED_ASIL_B_GLOBAL_32
 #undef MEMMAP_ERROR

/**************** Static Global Constants Sections ************************/
/**** Static Global Constants -- PF[x] ****/
#elif defined MCALLIB_START_SEC_CONST_ASIL_B_GLOBAL_8
 /*****User pragmas here for PF[x]*****/
 #undef MCALLIB_START_SEC_CONST_ASIL_B_GLOBAL_8
 #undef MEMMAP_ERROR
#elif defined MCALLIB_STOP_SEC_CONST_ASIL_B_GLOBAL_8
 /*****User pragmas here for PF[x]*****/
 #undef MCALLIB_STOP_SEC_CONST_ASIL_B_GLOBAL_8
 #undef MEMMAP_ERROR
#elif defined MCALLIB_START_SEC_CONST_ASIL_B_GLOBAL_32
 /*****User pragmas here for PF[x]*****/
 #undef MCALLIB_START_SEC_CONST_ASIL_B_GLOBAL_32
 #undef MEMMAP_ERROR
#elif defined MCALLIB_STOP_SEC_CONST_ASIL_B_GLOBAL_32
 /*****User pragmas here for PF[x]*****/
 #undef MCALLIB_STOP_SEC_CONST_ASIL_B_GLOBAL_32
 #undef MEMMAP_ERROR

/**** CODE -- PF[x] ****/
#elif defined MCALLIB_START_SEC_CODE_ASIL_B_GLOBAL
 /*****User pragmas here for PF[x]*****/
 #undef MCALLIB_START_SEC_CODE_ASIL_B_GLOBAL
 #undef MEMMAP_ERROR
#elif defined MCALLIB_STOP_SEC_CODE_ASIL_B_GLOBAL
 /*****User pragmas here for PF[x]*****/
 #undef MCALLIB_STOP_SEC_CODE_ASIL_B_GLOBAL
 #undef MEMMAP_ERROR
#endif


#if defined MEMMAP_ERROR
#error "MCALLIB_MemMap.h, wrong pragma command"
#endif
```

- **DET**
  The DET module is not required for integrating the MCALLIB driver.
- **DEM**
  The DEM module is not required for the integration of MCALLIB driver.
- **SchM**

The SchM module is a part of the RTE that manages the BSW. The MCALLIB driver uses the exclusive areas defined in the `SchM_McalLib.h` file to protect the SFRs and variables from concurrent accesses from different threads. The SchMs identified for the MCALLIB driver are:

- PeripheralEndInit
- SafetyEndInit
- CpuEndInit
- StmTimerResolution

The files `SchM_McalLib.h` and `SchM_McalLib.c` are provided in the MCAL package as an example code and needs to updated by the integrator. The user must implement the SchM functions defined by the

MCALLIB driver as **suspend / resume** of interrupts for the CPU on which the API is invoked. A sample implementation of the SchM functions is shown as below:

```
/**** Sample implementation of SchM_McalLib.c ****/
#include "SchM_McalLib.h"

void SchM_Enter_McalLib_PeripheralEndInit(void)
{
 /* Start of Critical Section */
 SuspendAllInterrupts();
}

void SchM_Exit_McalLib_PeripheralEndInit(void)
{
 /* End of Critical Section */
 ResumeAllInterrupts();
}

void SchM_Enter_McalLib_SafetyEndInit(void)
{
 /* Start of Critical Section */
 SuspendAllInterrupts();
}

void SchM_Exit_McalLib_SafetyEndInit(void)
{
 /* End of Critical Section */
 ResumeAllInterrupts();
}

void SchM_Enter_McalLib_CpuEndInit(void)
{
 /* Start of Critical Section */
 SuspendAllInterrupts();
}

void SchM_Exit_McalLib_CpuEndInit(void)
{
 /* End of Critical Section */
 ResumeAllInterrupts();
}

void SchM_Enter_McalLib_StmTimerResolution(void)
{
 /* Start of Critical Section */
 SuspendAllInterrupts();
}

void SchM_Exit_McalLib_StmTimerResolution(void)
{
 /* End of Critical Section */
```

```
    ResumeAllInterrupts();
  }
```

- **Safety error**

  The MCALLIB driver will report all the detected safety errors through the `Mcal_ReportSafetyError()` API.

  The driver performs only detection and reporting of the safety errors. The handling of the reported errors shall be done by the user. The `Mcal_ReportSafetyError()` API is provided in the files `Mcal_SafetyError.c` and `Mcal_SafetyError.h` as a stub code, and must be updated by the integrator to handle the reported errors.

  Note: All DET errors are also reported as safety errors (error code used is same as DET).
- **Notifications and callbacks**

  The MCALLIB driver does not provide any callbacks or notifications
- **Operating System (OS)**

  The integrator shall implement the APIs routed from the MCALLIB via `McalLib_OsStub.h` file when the User-1 mode is used by any driver.

1.1.4.2 Multicore and Resource Manager

The MCALLIB driver supports execution of its APIs from all CPU cores. The following are the key points to be considered with respect to multicore in the driver:

- MCALLIB services accessing global hardware resources (like safety and peripheral endinit protection) would create a critical section and a spinlock around these accesses, which will serialize the shared hardware resource access across cores.

  Code section:

  The executable code of <Mod> driver is placed under single MemMap section. It can be relocated to any PFlash region.

  Data section:

  The sections marked as global should be relocated to the non-cached LMU region.

  Constants:

  The marked as global should be relocated to the non-cached LMU region.

Note: Relocating of code, data or constants to a distant memory region would impact execution timings.

1.1.4.3 MCU support

The MCALLIB driver does not use any services provided by the MCU driver.

1.1.4.4 Port support

The MCALLIB driver does not use any services provided by the PORT driver.

1.1.4.5 DMA support

The MCALLIB driver does not use any services provided by the DMA driver.

1.1.4.6 Interrupt connections

The MCALLIB driver does not use any interrupt source.

1.1.4.7 Example usage

The MCALLIB is a library. All the APIs provided are independent of each other, therefore, there is no example usage for this driver.

1.1.5 Key architectural considerations

1.1.5.1 User mode

The integrator shall implement the APIs routed from the MCALLIB via `McalLib_OsStub.h` file when the User-1 mode is used by any driver.

1.1.5.2 Spinlock

Timeout value that is passed as an input parameter to the `Mcal_GetSpinlock()` API must be in the range of 1 microsecond to 1048575 microseconds (timeout when passed as 1 indicate as 1microsecond to this API).

1.2 Assumptions of Use (AoU)

The AoU for the MCALLIB driver are as follows.

- **ASIL level of calling module**

User shall ensure that the ASIL level of the calling module is same as that of MCALLIB.
[cover parentID MCALLIB={3982DA82-28CB-453e-8D1C-4B80B83BE3CF}]

- **Common critical section**

User shall ensure that core specific interrupts are disabled in the critical sections SchM_Enter_Wdg_CpuEndInit and SchM_Enter_McalLib_CpuEndInit.
[cover parentID MCALLIB={70616172-E23B-4d86-9C20-7C5DF26143D7}]

- **ENDINIT Protected Register Access**

User shall ensure that all the ENDINIT protected registers are modified using only the write ENDINIT APIs

(Mcal_WriteCpuEndInitProtReg, Mcal_WriteSafetyEndInitProtReg,

Mcal_WriteSafetyEndInitProtRegMask, Mcal_WritePeripEndInitProtReg).
[cover parentID MCALLIB={845BAE75-B05D-49dc-822F-7480A13C4A84}]

- **Parameter range check for Mcal_SetBitAtomic and Mcal_GetBitAtomic**

The MCALLIB user shall ensure the following while using the APIs Mcal_SetBitAtomic and Mcal_GetBitAtomic:

- Sum of the input parameter BitPos and BitLen should not be greater than 32 bits

- BitLen should always be constant and non-zero value
[cover parentID MCALLIB={E28707C1-2DDB-451b-8DA6-3625A9EB2244}]

- **Password check**

User shall verify the password set by calling the GetPassword APIs (Mcal_GetCpuWdgPassword,

Mcal_GetSafetyEndInitPassword, Mcal_GetPeripheralEndInitPassword) since the APIs related to setting of password does not authenticate the password and has no means to notify such an error.
[cover parentID MCALLIB={D3EA116F-A029-4b83-A6E8-BB03A72E7C9B}]

- **STM timer resolution**

User shall call the Mcal_DelayResetTickCalibration API after any change in the clock tree to update the STM timer resolution.
[cover parentID MCALLIB={EF8478C5-1EDD-459e-B5DF-E729EE956664}]

- **Test, Test and set spinlock mechanism**

User shall ensure that the lock address passed to the Mcal_Getspinlock API must be at a non-cached memory address. This API shall not be called within an ISR.
[cover parentID MCALLIB={8EADA6CF-0B73-430a-9545-B24315AAF137}]

- **Valid address (base + offset) are passed as register address for McalLib API**

Valid address (base + offset) shall be passed as the register address to the Mcal_WriteSafetyEndInitProtReg16 API.
[cover parentID MCALLIB={81931B95-E9B4-4caa-BF12-7B2E84F1BC58}]

- **Valid CSFR address (only offset) are passed as register address for McalLib APIs**

Valid CSFR address (only offset) shall be passed as the register address for Core Specific SFRs to the Mcal_WriteSafetyEndInitProtReg API
[cover parentID MCALLIB={817DF82C-39C1-4767-B78B-9ECE9F585305}]

- **Valid Pointer to be passed to APIs**

User shall ensure the correctness of the pointer that is passed as an input parameter before invoking the MCALLIB APIs.

[cover parentID MCALLIB={35C4D569-ECE0-4fF4-A361-4E2E5A06D535}]

- **Valid value are passed as parameter for MCALLIB APIs**

User shall ensure the correctness of the data value that is passed as an input parameter before invoking the MCALLIB APIs.

[cover parentID MCALLIB={42C179FC-3D4A-4648-B422-6BB895B43B4F}]

1.3 Reference information

1.3.1 Configuration interfaces

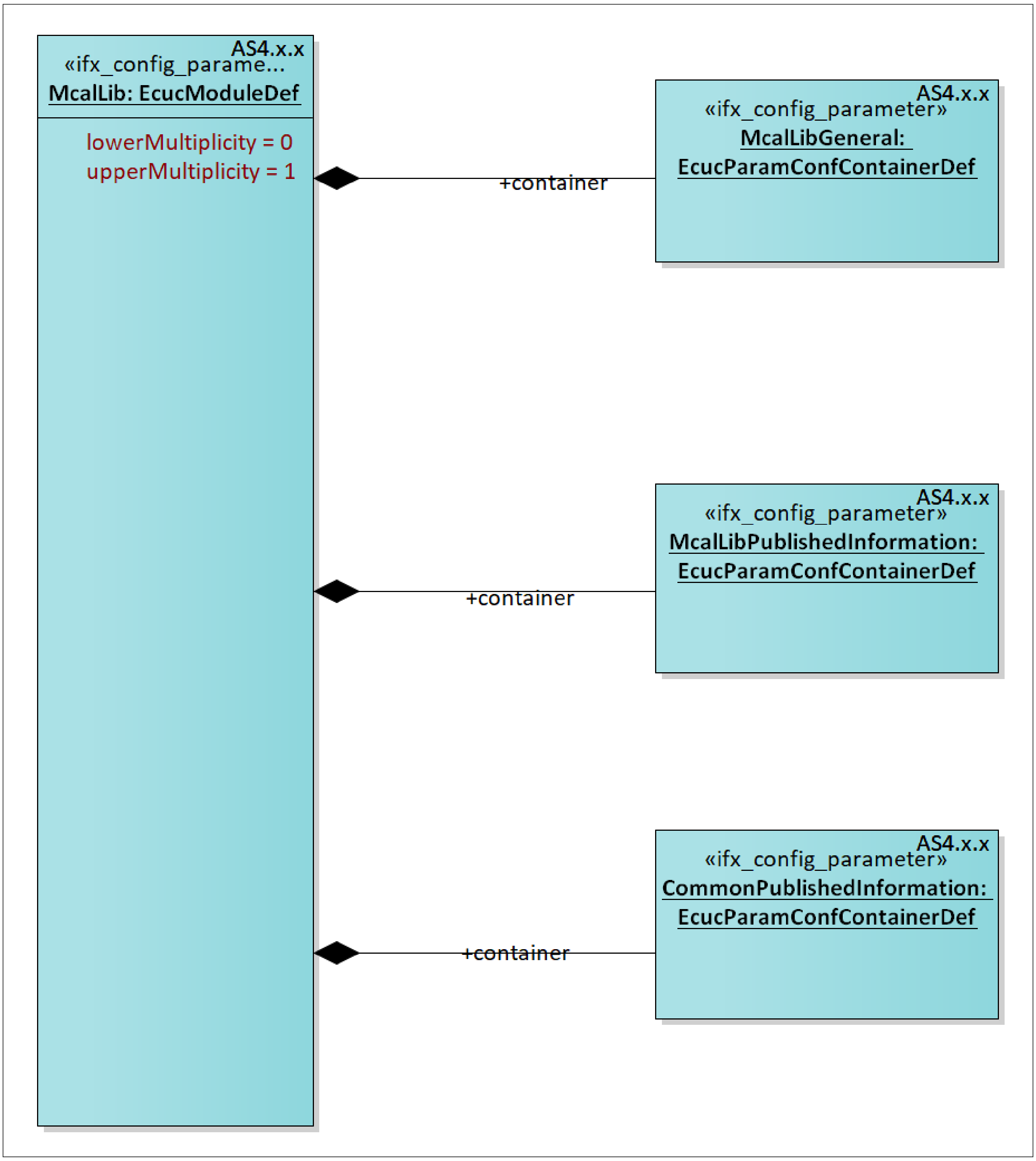

Figure 4 **Container hierarchy along with their configuration parameters**

1.3.1.1 Container: McalLibGeneral

Container for all the general configuration parameters for the MCALLIB driver.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.1.1 McalLibSafetyEnable

Table 4 Specification for McalLibSafetyEnable

Name	`McalLibSafetyEnable`		
Description	Switch to enable reporting of safety error. True : Safety error reporting is enabled. False: Safety error reporting is disabled. The detection of safety related errors is enabled by default to ensure that safety issues are addressed during the product lifecycle.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	TRUE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2 Container: CommonPublishedInformation

This container holds all the published information of the Mcal Library.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.2.1 ArMajorVersion

Table 5 Specification for ArMajorVersion

Name	`ArMajorVersion`		
Description	Major version number of the AUTOSAR specification on which the implementation is based on.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef

Table 5 Specification for ArMajorVersion (continued)

Range	0 - 255		
Default value	As per the selected Autosar version		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2.2 ArMinorVersion

Table 6 Specification for ArMinorVersion

Name	`ArMinorVersion`		
Description	Minor version number of the AUTOSAR specification on which the implementation is based on.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per the selected Autosar version		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2.3 ArPatchVersion

Table 7 Specification for ArPatchVersion

Name	`ArPatchVersion`		
Description	Patch version number of the AUTOSAR specification on which the implementation is based on.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per the selected Autosar version		

Table 7 Specification for ArPatchVersion (continued)

Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2.4 ModuleId

Table 8 Specification for ModuleId

Name	`ModuleId`		
Description	Module ID of MCALLIB.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 65535		
Default value	255		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2.5 Release

Table 9 Specification for Release

Name	`Release`		
Description	Specifies the derivative for which the configuration project is created.		
Multiplicity	1..1	**Type**	EcucStringParamDef
Range	String		
Default value	As per hardware derivative		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-

Table 9 Specification for Release (continued)

Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2.6 SwMajorVersion

Table 10 Specification for SwMajorVersion

Name	`SwMajorVersion`		
Description	Specifies the major version of the driver software.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per the driver version		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2.7 SwMinorVersion

Table 11 Specification for SwMinorVersion

Name	`SwMinorVersion`		
Description	Specifies the minor version of the driver software.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	0		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL

Table 11 Specification for SwMinorVersion (continued)

Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2.8 SwPatchVersion

Table 12 Specification for SwPatchVersion

Name	`SwPatchVersion`		
Description	Specifies the patch version of the driver software.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	0		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2.9 VendorId

Table 13 Specification for VendorId

Name	`VendorId`		
Description	Vendor ID for Infineon.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 65535		
Default value	17		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3 Container: McalLib

This is the parent container for all configuration parameters of MCALLIB.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.4 Container: McalLibPublishedInformation

Container for all the published information of MCALLIB.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.4.1 McalLibBackUpClockFrequency

Table 14 Specification for McalLibBackUpClockFrequency

Name	`McalLibBackUpClockFrequency`		
Description	Specifies the frequency of the back-up clock.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	100 - 100		
Default value	100		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.2 McalLibDsprCore0EndAddr

Table 15 Specification for McalLibDsprCore0EndAddr

Name	`McalLibDsprCore0EndAddr`		
Description	Specifies the end address of DSPR for Core 0. *Note: The range of the parameter depends on device. The specified range is for the superset device (TC39x).*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x7003BFFF - 0x7003BFFF		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-

Table 15 Specification for McalLibDsprCore0EndAddr (continued)

Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.3 McalLibDsprCore0StartAddr

Table 16 Specification for McalLibDsprCore0StartAddr

Name	`McalLibDsprCore0StartAddr`		
Description	Specifies the start address of DSPR for core 0.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x70000000 - 0x70000000		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.4 McalLibDsprCore1EndAddr

Table 17 Specification for McalLibDsprCore1EndAddr

Name	`McalLibDsprCore1EndAddr`		
Description	Specifies the end address of DSPR for Core 1. *Note: The range of the parameter depends on device. The specified range is for the superset device (TC39x).* *Note: If Core 1 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x6003BFFF - 0x6003BFFF		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-

Table 17 Specification for McalLibDsprCore1EndAddr (continued)

Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.5 McalLibDsprCore1StartAddr

Table 18 Specification for McalLibDsprCore1StartAddr

Name	`McalLibDsprCore1StartAddr`		
Description	Specifies the start address of DSPR for Core 1. *Note: If Core 1 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x60000000 - 0x60000000		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.6 McalLibDsprCore2EndAddr

Table 19 Specification for McalLibDsprCore2EndAddr

Name	`McalLibDsprCore2EndAddr`		
Description	Specifies the end address of DSPR for Core 2. *Note: The range of the parameter depends on device. The specified range is for the superset device (TC39x).* *Note: If Core 2 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x50017FFF - 0x50017FFF		
Default value	Depends on device		

Table 19 Specification for McalLibDsprCore2EndAddr (continued)

Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.7 McalLibDsprCore2StartAddr

Table 20 Specification for McalLibDsprCore2StartAddr

Name	`McalLibDsprCore2StartAddr`		
Description	Specifies the start address of DSPR for Core 2. *Note: If Core 2 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x50000000 - 0x50000000		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.8 McalLibDsprCore3EndAddr

Table 21 Specification for McalLibDsprCore3EndAddr

Name	`McalLibDsprCore3EndAddr`		
Description	Specifies the end address of DSPR for Core 3. *Note: The range of the parameter depends on device. The specified range is for the superset device (TC39x).* *Note: If Core 3 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x40017FFF - 0x40017FFF		
Default value	Depends on device		

Table 21 **Specification for McalLibDsprCore3EndAddr (continued)**

Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.9 McalLibDsprCore3StartAddr

Table 22 **Specification for McalLibDsprCore3StartAddr**

Name	`McalLibDsprCore3StartAddr`		
Description	Specifies the start address of DSPR for Core 3. *Note: If Core 3 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x40000000 - 0x40000000		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.10 McalLibDsprCore4EndAddr

Table 23 **Specification for McalLibDsprCore4EndAddr**

Name	`McalLibDsprCore4EndAddr`		
Description	Specifies the end address of DSPR for Core 4. *Note: The range of the parameter depends on device. The specified range is for the superset device (TC39x).* *Note: If Core 4 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x30017FFF - 0x30017FFF		
Default value	Depends on device		

Table 23 Specification for McalLibDsprCore4EndAddr (continued)

Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.11 McalLibDsprCore4StartAddr

Table 24 Specification for McalLibDsprCore4StartAddr

Name	`McalLibDsprCore4StartAddr`		
Description	Specifies the start address of DSPR for Core 4. *Note: If Core 4 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x30000000 - 0x30000000		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.12 McalLibDsprCore5EndAddr

Table 25 Specification for McalLibDsprCore5EndAddr

Name	`McalLibDsprCore5EndAddr`		
Description	Specifies the end address of DSPR for Core 5. *Note: The range of the parameter depends on device. The specified range is for the superset device (TC39x).* *Note: If Core 5 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x10017FFF - 0x10017FFF		
Default value	Depends on device		

Table 25 Specification for McalLibDsprCore5EndAddr (continued)

Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.13 McalLibDsprCore5StartAddr

Table 26 Specification for McalLibDsprCore5StartAddr

Name	`McalLibDsprCore5StartAddr`		
Description	Specifies the start address of DSPR for Core 5. *Note: If Core 5 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x10000000 - 0x10000000		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.14 McalLibMcalAvailableCores

Table 27 Specification for McalLibMcalAvailableCores

Name	`McalLibMcalAvailableCores`		
Description	Specifies the number of cores available for the selected device.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	1 - 6		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-

Table 27 Specification for McalLibMcalAvailableCores (continued)

Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.15 McalLibPsprCore0EndAddr

Table 28 Specification for McalLibPsprCore0EndAddr

Name	`McalLibPsprCore0EndAddr`		
Description	Specifies the end address of PSPR for Core 0. *Note: The range of the parameter depends on device. The specified range is for the superset device (TC39x).*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x7010FFFF - 0x7010FFFF		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.16 McalLibPsprCore0StartAddr

Table 29 Specification for McalLibPsprCore0StartAddr

Name	`McalLibPsprCore0StartAddr`		
Description	Specifies the start address of PSPR for Core 0.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x70100000 - 0x70100000		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-

Table 29 **Specification for McalLibPsprCore0StartAddr (continued)**

Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.17 McalLibPsprCore1EndAddr

Table 30 **Specification for McalLibPsprCore1EndAddr**

Name	`McalLibPsprCore1EndAddr`		
Description	Specifies the end address of PSPR for Core 1. *Note: The range of the parameter depends on device. The specified range is for the superset device (TC39x).* *Note: If Core 1 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x6010FFFF - 0x6010FFFF		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.18 McalLibPsprCore1StartAddr

Table 31 **Specification for McalLibPsprCore1StartAddr**

Name	`McalLibPsprCore1StartAddr`		
Description	Specifies the start address of PSPR for Core 1. *Note: If Core 1 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x60100000 - 0x60100000		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-

Table 31 Specification for McalLibPsprCore1StartAddr (continued)

Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.19 McalLibPsprCore2EndAddr

Table 32 Specification for McalLibPsprCore2EndAddr

Name	`McalLibPsprCore2EndAddr`		
Description	Specifies the end address of PSPR for Core 2. *Note: The range of the parameter depends on device. The specified range is for the superset device (TC39x).* *Note: If Core 2 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x5010FFFF - 0x5010FFFF		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.20 McalLibPsprCore2StartAddr

Table 33 Specification for McalLibPsprCore2StartAddr

Name	`McalLibPsprCore2StartAddr`		
Description	Specifies the start address of PSPR for Core 2. *Note: If Core 2 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x50100000 - 0x50100000		
Default value	Depends on device		

Table 33 **Specification for McalLibPsprCore2StartAddr (continued)**

Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.21 McalLibPsprCore3EndAddr

Table 34 **Specification for McalLibPsprCore3EndAddr**

Name	`McalLibPsprCore3EndAddr`		
Description	Specifies the end address of PSPR for Core 3. *Note: The range of the parameter depends on device. The specified range is for the superset device (TC39x).* *Note: If Core 3 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x4010FFFF - 0x4010FFFF		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.22 McalLibPsprCore3StartAddr

Table 35 **Specification for McalLibPsprCore3StartAddr**

Name	`McalLibPsprCore3StartAddr`		
Description	Specifies the start address of PSPR for Core 3. *Note: If Core 3 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x40100000 - 0x40100000		
Default value	Depends on device		

Table 35 Specification for McalLibPsprCore3StartAddr (continued)

Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.23 McalLibPsprCore4EndAddr

Table 36 Specification for McalLibPsprCore4EndAddr

Name	`McalLibPsprCore4EndAddr`		
Description	Specifies the end address of PSPR for Core 4. *Note: The range of the parameter depends on device. The specified range is for the superset device (TC39x).* *Note: If Core 4 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x3010FFFF - 0x3010FFFF		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.24 McalLibPsprCore4StartAddr

Table 37 Specification for McalLibPsprCore4StartAddr

Name	`McalLibPsprCore4StartAddr`		
Description	Specifies the start address of PSPR for Core 4. *Note: If Core 4 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x30100000 - 0x30100000		
Default value	Depends on device		

Table 37 Specification for McalLibPsprCore4StartAddr (continued)

Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.25 McalLibPsprCore5EndAddr

Table 38 Specification for McalLibPsprCore5EndAddr

Name	`McalLibPsprCore5EndAddr`		
Description	Specifies the end address of PSPR for Core 5. *Note: The range of the parameter depends on device. The specified range is for the superset device (TC39x).* *Note: If Core 5 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x1010FFFF - 0x1010FFFF		
Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.26 McalLibPsprCore5StartAddr

Table 39 Specification for McalLibPsprCore5StartAddr

Name	`McalLibPsprCore5StartAddr`		
Description	Specifies the start address of PSPR for Core 5. *Note: If Core 5 does not exist for the selected device, then the parameter holds a value 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0x10100000 - 0x10100000		

Table 39 Specification for McalLibPsprCore5StartAddr (continued)

Default value	Depends on device		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.2 Functions - Type definitions

This section lists all the data type of the MCALLIB driver.

1.3.2.1 unsigned_int

Table 40 Specification for unsigned_int

Syntax	`unsigned_int`	
Type	unsigned int	
File	`Mcal_Compiler.h`	
Range	32 bit	
Description	This data type is used for defining structure members that are bit fields. Rationale: As per AUTOSAR, all primitive data types needs to have compiler abstraction	
Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3 Functions - APIs

This section lists all the APIs of the MCALLIB driver.

1.3.3.1 Mcal_WriteSafetyEndInitProtReg16

Table 41 Specification for `Mcal_WriteSafetyEndInitProtReg16` API

Syntax	`void  Mcal_WriteSafetyEndInitProtReg16`
	`(`
	`    void * const RegAddress,`
	`    const uint16 DataValue`
	`)`

Service ID	0x81
Sync/Async	Synchronous

Table 41 Specification for `Mcal_WriteSafetyEndInitProtReg16` API (continued)

ASIL Level	B	
Re-entrancy	Non Reentrant	
Parameters (in)	DataValue	Value to be written to the register located at RegAddress.
Parameters (out)	RegAddress	Safety Endinit protected register address having 16 bit access *Note: The pointer will be pointer to volatile since the address passed is of a register.*
Parameters (in - out)	-	-
Return	void	-
Description	The API unlocks the safety ENDINIT protection, updates the protected register with 16-bit accesses and then locks back the safety ENDINIT protection. The API writes the value specified in 'DataValue' into the safety ENDINIT protected register, whose address is specified in 'RegAddress'.	
Source	IFX	
Error handling	MCALLIB_E_PARAM_POINTER	
Configuration dependencies	-	
User hints	-	
SFR accessed	CPU_COMPAT(w), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SEICON0(rw), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.2 Mcal_WriteSafetyEndInitProtRegMask

Table 42 Specification for `Mcal_WriteSafetyEndInitProtRegMask` API

Syntax	`void  Mcal_WriteSafetyEndInitProtRegMask`
	`(`
	`    void * const RegAddress,`
	`    const uint32 DataValue,`
	`    const uint32 Mask`
	`)`

Service ID	0x8F
Sync/Async	Synchronous
ASIL Level	B

Table 42 **Specification for `Mcal_WriteSafetyEndInitProtRegMask` API (continued)**

Re-entrancy	Non Reentrant	
Parameters (in)	DataValue Mask	Value to be written to the register located at RegAddress. Mask value for updating the registers. Bits set as 1 in the mask, will be updated in 'RegAddress', all the other bits are unchanged.
Parameters (out)	RegAddress	Address for the safety ENDINIT protected register. *Note: The pointer will be pointer to volatile since the address passed is of a register.*
Parameters (in - out)	-	-
Return	void	-
Description	The API updates the safety ENDINIT protected register, for which the address is specified by 'RegAddress'. The API also supports write access to safety endinit protected CSFRs, for which the 16-bit offset is specified by 'RegAddress'. The register is updated with the corresponding data value for the bit position where the mask value is 1. The remaining bits retain their original value. If register address is null pointer, then a safety error is reported. The API disables the safety ENDINIT protection, updates the protected register and then enables the safety ENDINIT protection.	
Source	IFX	
Error handling	MCALLIB_E_PARAM_POINTER	
Configuration dependencies	-	
User hints	None	
SFR accessed	CPU_COMPAT(w), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SEICON0(rw), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.3 McalLib_GetVersionInfo

Table 43 **Specification for `McalLib_GetVersionInfo` API**

Syntax	`void  McalLib_GetVersionInfo`
	`(`
	`    Std_VersionInfoType * const versioninfo`
	`)`

Service ID	0x79
Sync/Async	Synchronous

Table 43 **Specification for `McalLib_GetVersionInfo` API (continued)**

ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	versioninfo	Pointer to store the version information of the MCALLIB driver.
Parameters (in - out)	-	-
Return	void	-
Description	The API returns the version information of the MCALLIB driver.	
Source	IFX	
Error handling	MCALLIB_E_PARAM_POINTER	
Configuration dependencies	-	
User hints	-	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.4 Mcal_GetCpuIndex

Table 44 **Specification for `Mcal_GetCpuIndex` API**

Syntax	`uint32 Mcal_GetCpuIndex ( void )`	
Service ID	0x89	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	Index of the core on which the API is called
Description	The API retrieves the index of the core on which the API is invoked.	

Table 44 **Specification for `Mcal_GetCpuIndex` API (continued)**

	Note: For CPU5, although the actual core ID is 6, the API reports the index as 5. This maintains continuity of index from CPU0 to CPU5.
Source	IFX
Error handling	-
Configuration dependencies	-
User hints	-
SFR accessed	CPU_CORE_ID(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.5 Mcal_GetCpuPhysicalId

Table 45 **Specification for `Mcal_GetCpuPhysicalId` API**

Syntax	`uint32  Mcal_GetCpuPhysicalId`
	`(`
	`  void`
	`)`	

Service ID	0x8B	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	Identification number of the core.
Description	The API retrieves the identification number of the core on which the API is invoked. *Note: For CPU0 to CPU4, the identification number of the core is 0 to 4 respectively. For CPU5, the identification number of the core is 6.*	
Source	IFX	
Error handling	-	
Configuration dependencies	-	

Table 45 Specification for `Mcal_GetCpuPhysicalId` API (continued)

User hints	-	
SFR accessed	CPU_CORE_ID(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.6 Mcal_DelayGetTick

Table 46 Specification for `Mcal_DelayGetTick` API

Syntax	`uint32 Mcal_DelayGetTick ( void )`	
Service ID	0x8A	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	Lowest 32 bits of STM0.TIM0.
Description	The API retrieves the current value of the lowest 32-bits of the register STM0.TIM0.	
Source	IFX	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.7 Mcal_DelayResetTickCalibration

Table 47 Specification for `Mcal_DelayResetTickCalibration` API

Syntax	`uint32  Mcal_DelayResetTickCalibration`
	`(`
	`  void`
	`)`	

Service ID	0x86	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Non Reentrant	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	System timer (STM) resolution: Value of 1 STM tick in nano seconds.
Description	The API is invoked by the MCU driver to indicate to the MCALLIB driver, that the clock tree is updated. As a result of invocation of the API, the MCALLIB driver takes the following actions: - Calculates the STM resolution based on the new clock tree. - Old STM resolution is updated with the newly calculated value within the library. *Note: The API is expected to be invoked only by the MCU driver, which is responsible for configuring the clock tree.*	
Source	IFX	
Error handling	MCALLIB_E_CLKDISABLE	
Configuration dependencies	-	
User hints	The MCU clock tree should be initialized prior calling the API. The API is allowed to be called only by the MCAL MCU driver. *Note: In the flowchart, the value of Ndiv,Pdiv,K2 div are NDIV+1,PDIV+1(from SYSPLLCON0 register) and K2DIV+1(from SYSPLLCON1 register respectively)* *Note: In the flowchart, the value of TIMER_RESOL_1_NANOSEC is 10^9, which is used to return STM timer resolution in 1ns resolution.*	
SFR accessed	SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.8 Mcal_DelayTickResolution

Table 48 Specification for `Mcal_DelayTickResolution` API

Syntax	`uint32 Mcal_DelayTickResolution ( void )`	
Service ID	0x8C	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	System timer(STM) resolution: Value of 1 STM tick in nano second.
Description	The API retrieves the resolution of a STM in nanosecond. *Note: A return value of 0 indicates that STM is switched off or the Mcal_DelayResetTickCalibration API was never invoked.*	
Source	IFX	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.9 Mcal_GetBitAtomic

Table 49 Specification for `Mcal_GetBitAtomic` API

Syntax	`uint32 Mcal_GetBitAtomic ( const uint32 DataValue, const uint8 BitPos, const uint8 BitLen )`
Service ID	NA

Table 49 Specification for `Mcal_GetBitAtomic` API (continued)

Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	NA	
Parameters (in)	DataValue BitPos BitLen	Value of the variable or register from which bits need to extracted. Starting bit position of the data to be extracted. Bit length of the data to be extracted.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	Bits extracted from 'DataValue'
Description	The API extracts bits of data from the 32-bit value. The start position and length of the data to be extracted is specified by BitPos and BitLen respectively. *Note: The API is implemented as a macro.*	
Source	IFX	
Error handling	-	
Configuration dependencies	-	
User hints	-	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.10 Mcal_SetBitAtomic

Table 50 Specification for `Mcal_SetBitAtomic` API

Syntax	`void Mcal_SetBitAtomic (uint32 * const DataPtr, const uint8 BitPos, const uint8 BitLen, const uint32 Data)`
Service ID	NA
Sync/Async	Synchronous
ASIL Level	B
Re-entrancy	NA

```
void  Mcal_SetBitAtomic
(
    uint32 * const DataPtr,
    const uint8 BitPos,
    const uint8 BitLen,
    const uint32 Data
)
```

Table 50 Specification for `Mcal_SetBitAtomic` API (continued)

Parameters (in)	DataPtr BitPos BitLen Data	Variable or register address to be updated. Starting bit position of the data to be modified. Bit length of the data to be modified Value to be updated to address pointed by DataPtr
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The API atomically stores 'Data' at the address location pointed by 'DataPtr'. The start position and length of the data to be updated is specified by 'BitPos' and BitLen' respectively. Only the bits specified by BitPos and BitLen is updated, all the other bits are unchanged. *Note: The API is implemented as a macro.*	
Source	IFX	
Error handling	-	
Configuration dependencies	-	
User hints	-	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.11 Mcal_GetGlobalDsprAddress

Table 51 Specification for `Mcal_GetGlobalDsprAddress` API

Syntax	`uint32  Mcal_GetGlobalDsprAddress`
	`(`
	`    const uint32 CpuId,`
	`    const uint32 LocalDsprAddress`
	`)`	

Service ID	0x7B	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	CpuId LocalDsprAddress	Physical CPU Core ID *Note: For CPU5 the physical core ID is 6.* Local DSPR address for which the global DSPR address is to be returned

Table 51 **Specification for `Mcal_GetGlobalDsprAddress` API (continued)**

Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	- If the passed parameter is a valid core ID and local DSPR address, then the API returns the global DSPR address. - If the passed parameter is valid global DSPR address corresponding to the passed CpuId then the API returns the passed address as is. - If the passed parameter (CpuId or LocalDsprAddress or both) is invalid then the API returns value 0.
Description	The API returns the global address of a local DSPR address of the specified CPU.	
Source	IFX	
Error handling	-	
Configuration dependencies	-	
User hints	None.	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.12 Mcal_GetGlobalPsprAddress

Table 52 **Specification for `Mcal_GetGlobalPsprAddress` API**

Syntax	`uint32  Mcal_GetGlobalPsprAddress (     const uint32 CpuId,     const uint32 LocalPsprAddress )`	
Service ID	0x7D	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	CpuId LocalPsprAddress	Physical Core ID Local PSPR address for which global PSPR address is to be returned
Parameters (out)	-	-
Parameters (in - out)	-	-

Table 52 **Specification for `Mcal_GetGlobalPsprAddress` API (continued)**

Return	uint32	- If the passed parameter is a valid core ID and local PSPR address, then the API returns the global PSPR address. - If the passed parameter is valid global PSPR address corresponding to the passed CpuId then the API returns the passed address as is. - If the passed parameter (CpuId or LocalPsprAddress or both) is invalid then the API returns a value of 0.
Description	The API returns the global address of a local PSPR address of the specified CPU.	
Source	IFX	
Error handling	-	
Configuration dependencies	-	
User hints	None.	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.13 Mcal_GetLocalDsprAddress

Table 53 **Specification for `Mcal_GetLocalDsprAddress` API**

Syntax	`uint32 Mcal_GetLocalDsprAddress`
	`(`
	`    const uint32 GlobalDsprAddress`
	`)`	

Service ID	0x83	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	GlobalDsprAddress	Global DSPR address
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	- If passed parameter is a valid global DSPR address, then routine return local DSPR address. - If passed parameter is valid local DSPR address corresponding to currently executing CPU then routine returns the passed address as is. - If passed parameter is an invalid address then routine return a value of 0.

Table 53 **Specification for `Mcal_GetLocalDsprAddress` API (continued)**

Description	The API returns the local DSPR address for a global DSPR address.
Source	IFX
Error handling	-
Configuration dependencies	-
User hints	None.
SFR accessed	CPU_CORE_ID(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.14 Mcal_GetLocalPsprAddress

Table 54 **Specification for `Mcal_GetLocalPsprAddress` API**

Syntax	`uint32  Mcal_GetLocalPsprAddress`
	`(`
	`    const uint32 GlobalPsprAddress`
	`)`	

Service ID	0x84	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	GlobalPsprAddress	Global PSPR address
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	- If the passed parameter is a valid global PSPR address, then the API returns local PSPR address - If the passed parameter is valid local PSPR address corresponding to currently executing CPU then the API returns the passed address as is. - If the passed parameter is an invalid address then the API returns a value of 0.
Description	The API returns the local PSPR address for a global PSPR address.	
Source	IFX	
Error handling	-	

Table 54 **Specification for `Mcal_GetLocalPsprAddress` API (continued)**

Configuration dependencies	-
User hints	None.
SFR accessed	CPU_CORE_ID(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.15 Mcal_GetPeripheralEndInitPassword

Table 55 **Specification for `Mcal_GetPeripheralEndInitPassword` API**

Syntax	`uint32  Mcal_GetPeripheralEndInitPassword`
	`(`
	`  void`
	`)`	

Service ID	0x82	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	Current peripheral ENDINIT password.
Description	The API retrieves the peripheral ENDINIT password installed in the EPW bitfield of EICON0 register. *Note: The API reads the current password stored in EICON.EPW, and inverts the bits 0 to 5 of the password before reporting.*	
Source	IFX	
Error handling	-	
Configuration dependencies	-	
User hints	-	
SFR accessed	SCU_EICON0(r)	

Table 55 **Specification for `Mcal_GetPeripheralEndInitPassword` API (continued)**

	Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.16 Mcal_GetCpuWdgPassword

Table 56 **Specification for `Mcal_GetCpuWdgPassword` API**

Syntax	`uint32  Mcal_GetCpuWdgPassword (   void )`	
Service ID	0x88	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	Currently installed password for the CPU watchdog.
Description	The API retrieves the ENDINIT password for the watchdog of the CPU on which the API is invoked. *Note: The API reads the current password stored in CON0.PW, and inverts the bits 0 to 5 of the password before reporting.*	
Source	IFX	
Error handling	-	
Configuration dependencies	-	
User hints	-	
SFR accessed	CPU_CORE_ID(r), SCU_WDTCPU_CON0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.17 Mcal_GetSafetyEndInitPassword

Table 57 **Specification for `Mcal_GetSafetyEndInitPassword` API**

Syntax	`uint32 Mcal_GetSafetyEndInitPassword ( void )`	
Service ID	0x87	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	Currently installed safety ENDINIT password.
Description	The API retrieves the safety ENDINIT password installed in the EPW bit field of SEICON0 register. *Note: The API reads the current password stored in SEICON0.EPW and inverts the bits 0 to 5 of the password before reporting.*	
Source	IFX	
Error handling	-	
Configuration dependencies	-	
User hints	-	
SFR accessed	SCU_SEICON0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.18 Mcal_WriteSafetyEndInitProtReg

Table 58 **Specification for `Mcal_WriteSafetyEndInitProtReg` API**

Syntax	`void Mcal_WriteSafetyEndInitProtReg ( void * const RegAddress, const uint32 DataValue )`

Table 58 **Specification for `Mcal_WriteSafetyEndInitProtReg` API (continued)**

Service ID	0x7F	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Non Reentrant	
Parameters (in)	DataValue	Value to be written to the register located at RegAddress.
Parameters (out)	RegAddress	Address for the safety ENDINIT protected register. *Note: The pointer will be pointer to volatile since the address passed is of a register.*
Parameters (in - out)	-	-
Return	void	-
Description	The API unlocks the safety ENDINIT protection, updates the protected register and then locks back the safety ENDINIT protection. The API also supports write access to safety ENDINIT protected CSFRs, for which the 16-bit offset is specified by 'RegAddress'. The API writes the value specified in 'DataValue' into the safety ENDINIT protected register, whose address is specified in 'RegAddress'.	
Source	IFX	
Error handling	MCALLIB_E_PARAM_POINTER	
Configuration dependencies	-	
User hints	-	
SFR accessed	CPU_COMPAT(w), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SEICON0(rw), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.19 Mcal_SetCpuWdgPassword

Table 59 **Specification for `Mcal_SetCpuWdgPassword` API**

Syntax	`uint32  Mcal_SetCpuWdgPassword`
	`(`
	`    const uint32 Password`
	`)`

Service ID	0x85
Sync/Async	Synchronous

Table 59 **Specification for `Mcal_SetCpuWdgPassword` API (continued)**

ASIL Level	B	
Re-entrancy	Non Reentrant on same CPU, Reentrant for other CPUs	
Parameters (in)	Password	New password to be installed for CPU ENDINIT protection
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	Previously installed password
Description	The API installs a new ENDINIT password for the watchdog of the CPU on which the API is invoked. The interface internally prepares the password (both for static and automatic password sequencing), installs the password and returns the previously installed password. *Note: Bits 0 to 5 of the previously installed password is inverted before reporting.*	
Source	IFX	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	CPU_CORE_ID(r), SCU_WDTCPU_CON0(rw), SCU_WDTCPU_SR(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.20 Mcal_SetPeripheralEndInitPassword

Table 60 **Specification for `Mcal_SetPeripheralEndInitPassword` API**

Syntax	`uint32 Mcal_SetPeripheralEndInitPassword ( const uint32 Password )`	
Service ID	0x7C	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Non Reentrant	
Parameters (in)	Password	New password to be installed for peripheral ENDINIT.

Table 60 Specification for `Mcal_SetPeripheralEndInitPassword` API (continued)

Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	Previously installed password
Description	The API installs a new peripheral ENDINIT password. The interface internally prepares the password, installs the password and returns the previously installed password. *Note: Bits 0 to 5 of the previously installed password is inverted before reporting.*	
Source	IFX	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	SCU_CCUCON0(r), SCU_EICON0(rw), SCU_OSCCON(r), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.21 Mcal_SetSafetyEndInitPassword

Table 61 Specification for `Mcal_SetSafetyEndInitPassword` API

Syntax	`uint32  Mcal_SetSafetyEndInitPassword ( const uint32 Password )`	
Service ID	0x80	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Non Reentrant	
Parameters (in)	Password	New password to be installed for safety ENDINIT protection
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	Previously installed password

Table 61 Specification for `Mcal_SetSafetyEndInitPassword` API (continued)

Description	The API installs a new safety ENDINIT password. The interface internally prepares the password, installs the password and returns the previously installed password. *Note: Bits 0 to 5 of the previously installed password is inverted before reporting.*
Source	IFX
Error handling	-
Configuration dependencies	-
User hints	None
SFR accessed	SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SEICON0(rw), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.22 Mcal_GetSpinlock

Table 62 Specification for `Mcal_GetSpinlock` API

Syntax	`void Mcal_GetSpinlock (volatile uint32 * const LockAddress, const uint32 Timeout)`	
Service ID	0x8D	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Non Reentrant	
Parameters (in)	LockAddress	Address of the spinlock to be acquired.
	Timeout	Maximum wait time(micro second) to acquire the spinlock.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The API acquires the passed spinlock atomically. It is implemented in MCALLIB using Test, Test and Set Spinlock (TTAS) mechanism. A Timeout shall be passed as input parameter to spinlock API so that TTAS does not enter into an indefinite loop. If spinlock is not acquired within the specified timeout, then the control returns to the application after reporting a safety error.	

Table 62 **Specification for `Mcal_GetSpinlock` API (continued)**

Source	IFX
Error handling	MCALLIB_E_TIMEOUT_FAILED, MCALLIB_E_PARAM_POINTER
Configuration dependencies	-
User hints	User shall ensure that when this interface is used the McalLibSafetyEnable parameter shall be enabled to detect timeout.
SFR accessed	SCU_CCUCON0(r), SCU_OSCCON(r), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.23 Mcal_ReleaseSpinlock

Table 63 **Specification for `Mcal_ReleaseSpinlock` API**

Syntax	`void  Mcal_ReleaseSpinlock ( volatile uint32 * const LockAddress )`	
Service ID	0x8E	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Non Reentrant	
Parameters (in)	LockAddress	Address of the spinlock to be released.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The API releases the spinlock pointed to by the lock address.	
Source	IFX	
Error handling	MCALLIB_E_PARAM_POINTER	
Configuration dependencies	-	
User hints	-	
SFR accessed	-	

Table 63 Specification for `Mcal_ReleaseSpinlock` API (continued)

Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.24 Mcal_WriteCpuEndInitProtReg

Table 64 Specification for `Mcal_WriteCpuEndInitProtReg` API

Syntax	`void  Mcal_WriteCpuEndInitProtReg (     void * const RegAddress,     const uint32 DataValue )`	
Service ID	0x7E	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Non Reentrant	
Parameters (in)	DataValue	Value to be written to the register located at RegAddress.
Parameters (out)	RegAddress	Address of the CPU ENDINIT protected register. *Note: The pointer will be pointer to volatile since the address passed is of a register.*
Parameters (in - out)	-	-
Return	void	-
Description	The API unlocks the CPU ENDINIT protection, updates the protected register and then locks back the CPU ENDINIT protection. The API also supports write access to CPU ENDINT protected CSFRs, for which the 16-bit offset is specified by 'RegAddress'. The API writes the value specified in 'DataValue' into the CPU ENDINIT protected register, whose address is specified through ‘RegAddress’.	
Source	IFX	
Error handling	MCALLIB_E_PARAM_POINTER	
Configuration dependencies	-	
User hints	None	
SFR accessed	CPU_BIV(w), CPU_BTV(w), CPU_CORE_ID(r), CPU_DCON0(w), CPU_ISP(w), CPU_PCON0(w), CPU_PMA0(w), CPU_PMA1(w), CPU_SEGEN(w), SCU_WDTCPU_CON0(rw), SCU_WDTCPU_SR(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	

Table 64 **Specification for `Mcal_WriteCpuEndInitProtReg` API (continued)**

Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.25 Mcal_WritePeripEndInitProtReg

Table 65 **Specification for `Mcal_WritePeripEndInitProtReg` API**

Syntax	`void  Mcal_WritePeripEndInitProtReg`
	`(`
	`    void * const RegAddress,`
	`    const uint32 DataValue`
	`)`	

Service ID	0x7A	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Non Reentrant	
Parameters (in)	DataValue	Value to be written to the register located at RegAddress..
Parameters (out)	RegAddress	Address of the peripheral ENDINIT protected register. *Note: The pointer will be pointer to volatile since the address passed is of a register.*
Parameters (in - out)	-	-
Return	void	-
Description	The API unlocks the peripheral ENDINIT protection, updates the protected register and then locks back the peripheral ENDINIT protection. The API writes the value specified in 'DataValue' into the peripheral ENDINIT protected register, whose address is specified through ‘RegAddress’.	
Source	IFX	
Error handling	MCALLIB_E_PARAM_POINTER	
Configuration dependencies	-	
User hints	None	
SFR accessed	SCU_CCUCON0(r), SCU_EICON0(rw), SCU_OSCCON(r), SCU_SYSPLLCON0(r), SCU_SYSPLLCON1(r), STM_TIM0(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.4 Notifications and Callbacks

The MCALLIB driver does not provide any notifications or callbacks.

1.3.5 Scheduled functions

The MCALLIB driver does not provide any scheduled functions.

1.3.6 Interrupt service routines

The MCALLIB driver does not provide any interrupt handlers.

1.3.7 Callout

The driver does not support any callout functions.

1.3.8 Errors Handling

This section describes the various errors reported by the MCALLIB driver.

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
MCALLIB_E_CLKDISABLE: The error code is reported if the STM clock divider is zero and the returned STM resolution is zero.	IFX	0xD0	SAFETY	0xD0	SAFETY
MCALLIB_E_PARAM_POINTER: The error code is reported if the API is invoked with a null pointer as a parameter.	IFX	0xC9U	SAFETY	0xC9U	SAFETY
MCALLIB_E_TIMEOUT_FAILED: The error code is reported if the spinlock could not be acquired in the specified timeout.	IFX	0xCCU	SAFETY	0xCCU	SAFETY

1.3.9 Deviations and limitations

This section describes the deviations and limitations of the MCALLIB driver.

1.3.9.1 Deviations

This section describes the deviations of the MCALLIB driver.

1.3.9.1.1 Software specification deviations

The MCALLIB driver does not have any deviations.

1.3.9.1.2 AMDC Violations

The MCALLIB driver does not have any AMDC violations..

1.3.9.1.3 VSMD Violations

The MCALLIB driver does not have any VSMD violations.

1.3.9.2 Limitations

This section describes the limitation of the MCALLIB driver.

Table 66 Known Limitation

Reference	Limitation
STM timer resolution	When the STM clock divider is zero, the resolution calculated in the Mcal_DelayResetTickCalibration() API is zero. User must ensure that the value of CCUCON0.STMDIV is not zero before using this MCALLIB API.

Revision history

Table 67 **Revision history**

Date	Version	Description
2020-11-10	2.0	Document is released.
2020-11-09	1.2	SFR access information for APIs updated.
2020-10-26	1.1	- Description of `Mcal_WriteSafetyEndInitProtReg`, `Mcal_WriteSafetyEndInitProtRegMask` and `Mcal_WriteCpuEndInitProtReg` APIs updated to include support for write access to CSFRs - Ranges of DSPR and PSPR updated and notes added for device dependency
2020-08-13	1.0	Document is released.
2020-08-07	0.1	- Initial draft - The MCALLIB driver chapter moved from MC-ISAR_TC3xx_UM_BASIC to this document

Edition 2020-11-10

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about any aspect of this document?

Email: erratum@infineon.com

Document reference
IFX-ocr1484806431059