MCAL User Manual for Icu_17_TimerIp

32-bit TriCore™ AURIX™ 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™ AURIX™ 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 Icu_17_TimerIp 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™ TC3xx MCAL User Manual General
- Specification of ICU Driver, AUTOSAR_SWS_ICU_Driver, AUTOSAR Release 4.2.2
- Specification of ICU Driver, AUTOSAR_SWS_ICU_Driver, AUTOSAR Release 4.4.0

Table of contents

1 ICU driver

1.1 User information

1.1.1 Description

The ICU driver is responsible for providing standard signal measurement services specified by AUTOSAR. The underlying capture engine of an ICU channel can be a TIM channel of the GTM unit, a CC6 comparator of the CCU6 module, an ERU channel or a GPT12 timer.

1.1.2 Hardware-software mapping

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

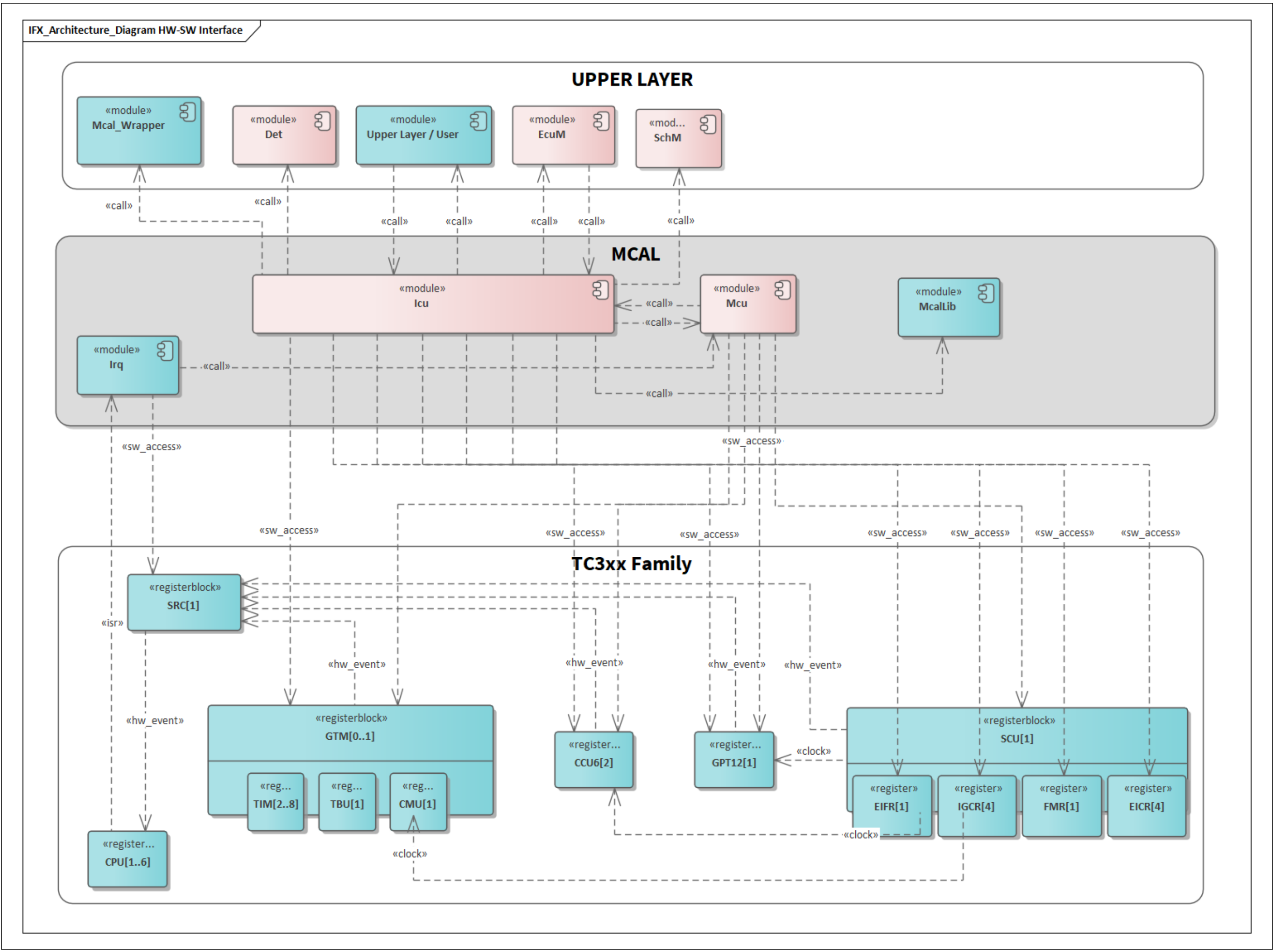

Figure 1 Mapping of hardware-software interfaces

1.1.2.1 GTM-TIM: primary hardware peripheral

Hardware functional features

The ICU driver uses the GTM-TIM for edge detection, edge counting, signal measurement and time stamping of input signal.

The key GTM-TIM features used by the ICU driver are:

- Filter configuration
- Channel clock source configuration
- TIM channel in TPWM, TIEM and TIPM modes
- Timeout detection(TDU)

The unsupported features of the GTM-TIM are:

- The ICU driver does not support the usage of ARU in TIM
- Generating filter input using lookup table functionality is not used
- The GTM interrupt mode, GTM_INTERRUPT_PULSE_MODE, cannot be used to achieve the ICU functions. In this mode, the interrupt bit in IRQ_NOTIFY register is always cleared if IRQ_EN is enabled. Hence, the MCU driver whose responsibility is to route the ISR to the ICU driver cannot check/determine/validate the interrupt source (from IRQ_NOTIFY flags). Therefore, GTM_INTERRUPT_PULSE_MODE selection is not available

Users of the hardware

A TIM channel of the GTM is exclusively used by the ICU driver. The TIM channel is not shared with any other driver. The MCU driver provides APIs to program the GTM SFRs. The ICU driver uses these APIs to write the GTM SFRs. The MCU driver also provides configuration and initializes the PORT pin connection to the TIM channel. Additionally, updates to channel-specific SFRs are performed by the ICU driver. Since these channels are exclusively reserved for the ICU driver, access to the channel-specific SFRs from other drivers or user software is not allowed.

Hardware diagnostic features

Not applicable.

Hardware events

The ICU driver uses the following hardware events from the GTM-TIM IP:

- New measurement value(NEWVAL): For edge detection, edge counting, signal measurement and time stamping
- CNT counter overflow(CNTOFL): To identify counter overflows
- Time out detection(TODET): To identify the timeout reached for the input signal

1.1.2.2 CCU6: primary hardware peripheral

Hardware functional features

The ICU driver uses the CC6 slices of the T12 timer instance for realizing the signal measurement, time stamping and signal edge detection functions.

The key CCU6 features used by the ICU driver are:

- Clock divider and pre-scalar configuration
- Selection of input signal of CC6 slice from PORT pin
- CC6 comparator in the Capture mode

The unsupported features of the CCU6 are:

- CCU6 kernel's T13 is not used by the ICU driver
- Compare modes and multi-input capture modes are not used by the ICU driver.

- Dead Time control and output modulations
- Hall sensor mode
- Multi-Channel mode
- Synchronous start feature

Users of the hardware

A CC6x comparator belonging to T12 of the CCU6 kernel is used by an ICU channel. A CCU6 kernel can be reserved to PWM or ICU. However all the comparators of a kernel reserved for the ICU are exclusive for the ICU. The ADC can also use the CCU6 trigger events that do not impact any functionality of the ICU. The MCU driver provides APIs to program the CCU6 SFRs. The ICU driver uses these APIs to write the CCU6 SFRs. Additionally, updates to channel-specific SFRs are performed by the ICU driver. Since these channels are exclusively reserved for the ICU driver, access to the channel-specific SFRs from other drivers or user software is not allowed.

Hardware diagnostic features

Not applicable.

Hardware events

The ICU driver uses the edge detected interrupt generated by the CC6 slices.

1.1.2.3 ERU: primary hardware peripheral

The ERU module can be used for signal edge detection and notification purpose.

Hardware functional features

The key ERU features used by the ICU driver are:

- Channel edge detection feature of the ERU functional block is configured and accessed by the ICU driver.
- ICU also configures the PORT pin selection for the ERU channel.

The unsupported features of the ERU are:

- Hardware event on a pattern detection at the input is not used by the ICU driver as the feature is not relevant for the ICU functionality.
- ERU filter external input filter register, filter clock pre-divider and glitch filter depth configuration parameters will not be part of the ICU driver as these parameters are applicable to the complete ERU unit and not per channel.

Users of the hardware

- An ERU input channel for input selection and ERU output channel for interrupt trigger are used by the ICU driver. As the two ERU output channels share the same interrupt line, both channels shall be allocated to the same module. An ERU input channel can be used by ICU, ADC or DSADC driver.

Hardware diagnostic features

Not applicable.

Hardware events

The ICU driver handles the edge detect interrupts generated by the ETL block.

1.1.2.4 GPT12: primary hardware peripheral

Hardware functional features

The ICU driver uses the GPT timer of the GPT12 peripheral to realize edge detection, edge counting and incremental interface modes.

The key GPT12 features used by the ICU driver are:

- Clock pre-scalar configuration
- Input selection from PORT pin
- Timer in Counter and incremental interface modes

The unsupported features of the GPT12 are:

- Timer and Gated timer modes
- GPT2 CAPREL mode
- Time concatenation of T2/T4 and T5/T6
- Reload and Capture mode of T2/T4

Users of the hardware

A GPT timer instance of the GPT12 is used by the ICU driver. The GPT12 timer block can be shared with GPT driver. The MCU driver provides APIs to program the GPT12 SFRs. The ICU driver uses these APIs to write the GPT12 SFRs. Additionally, updates to channel-specific SFRs are performed by the ICU driver. Since these channels are exclusively reserved for the ICU driver, access to the channel-specific SFRs from other drivers or user software is not allowed.

Hardware diagnostic features

Not applicable.

Hardware events

The ICU driver uses the following hardware event from the GPT12 IP:

- Timer counter overflow event
- Count edge detection interrupt in incremental interface mode

1.1.2.5 SCU: dependent hardware peripheral

Hardware functional features

The ICU driver depends on the SCU IP for the clock, ENDINIT and reset functions. The driver requires the fSPB and fGTM clock signals for functioning.

Users of the hardware

The SCU IP supplies clock for all the peripherals and the MCU driver, and 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.

Hardware diagnostic features

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

Hardware events

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

1.1.3 File structure

1.1.3.1 C file structure

This section provides details of the C files of the ICU driver.

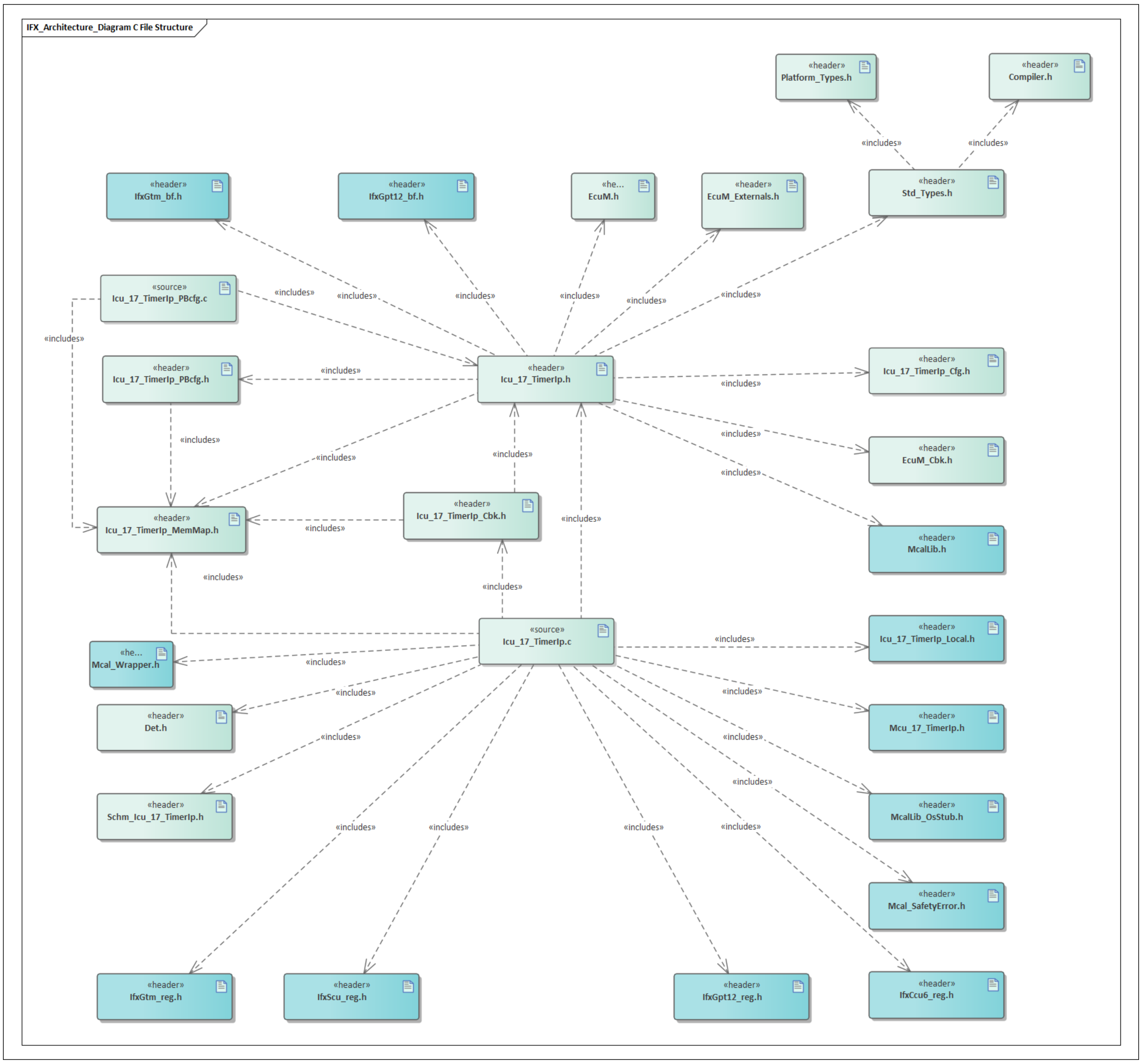
IFX_Architecture_Diagram C File Structure
«header» Platform_Types.h
«header» Compiler.h
«header» Std_Types.h
«header» IfxGtm_bf.h
«header» IfxGpt12_bf.h
«he... EcuM.h
«header» EcuM_Externals.h
«source» Icu_17_TimerIp_PBcfg.c
«header» Icu_17_TimerIp_PBcfg.h
«header» Icu_17_TimerIp.h
«header» Icu_17_TimerIp_Cfg.h
«header» EcuM_Cbk.h
«header» McalLib.h
«header» Icu_17_TimerIp_MemMap.h
«header» Icu_17_TimerIp_Cbk.h
«source» Icu_17_TimerIp.c
«he... Mcal_Wrapper.h
«header» Icu_17_TimerIp_Local.h
«header» Det.h
«header» Mcu_17_TimerIp.h
«header» Schm_Icu_17_TimerIp.h
«header» McalLib_OsStub.h
«header» Mcal_SafetyError.h
«header» IfxGtm_reg.h
«header» IfxScu_reg.h
«header» IfxGpt12_reg.h
«header» IfxCcu6_reg.h
«includes»

Figure 2 **Icu_C_File_Structure-1.png**

Table 2 **C file structure**

File name	Description
`Compiler.h`	Provides abstraction from compiler-specific keywords
`Det.h`	Provides the exported interfaces of Development Error Tracer
`EcuM.h`	Header file exporting the declarations of the EcuM

(table continues...)

Table 2 (continued) C file structure

File name	Description
`EcuM_Cbk.h`	Header file containing declarations of the EcuM callbacks. *Note: This file is available only for AUTOSAR version 4.2.2*
`EcuM_Externals.h`	Header file containing declarations of the EcuM callbacks. *Note: This file is available only for AUTOSAR version 4.4.0*
`Icu_17_TimerIp.c`	File (static) containing implementation of APIs
`Icu_17_TimerIp.h`	Header file (static) defining prototypes of configuration data structures and APIs
`Icu_17_TimerIp_Cbk.h`	Header file to declare the callback APIs
`Icu_17_TimerIp_Cfg.h`	Header file (generated) containing constants and pre-processor macros
`Icu_17_TimerIp_Local.h`	Header file defining type definition of global data and inline APIs, which can used across source files
`Icu_17_TimerIp_MemMap.h`	File (static) containing the memory section definitions used by the ICU driver
`Icu_17_TimerIp_PBcfg.c`	File (generated) containing objects to data structures
`Icu_17_TimerIp_PBcfg.h`	File (generated) containing declaration of the post-build configuration data structures
`IfxCcu6_reg.h`	SFR header file for CCU6
`IfxGpt12_bf.h`	SFR header file for GPT12
`IfxGpt12_reg.h`	SFR header file for GPT12
`IfxGtm_bf.h`	SFR header file for GTM
`IfxGtm_reg.h`	SFR header file for GTM
`IfxScu_reg.h`	SFR header file for SCU
`McalLib.h`	Static header file defining prototypes of data structure and APIs exported by the MCALLIB.
`McalLib_OsStub.h`	McalLib_OsStub.h provides macros to support user mode of Tricore. This shall be included by other drivers to call OS APIs.
`Mcal_SafetyError.h`	Header file containing the prototype of the API for reporting safety-related errors
`Mcal_Wrapper.h`	Provides the exported interfaces for Production Error and Runtime Development Errors. Implemented by default to include functions of Dem.h and Det.h files. This file can be modified by the user but function prototype is not user modifiable.
`Mcu_17_TimerIp.h`	Header file defining prototypes of data structures and APIs of Timer IPs (GTM, CCU6 and GPT12), containing functions such as initialization, enable, interrupt handlers and other services and is included by Mcu_17_TimerIp.c source file
`Platform_Types.h`	Platform-specific type declaration file as defined by AUTOSAR
`Schm_Icu_17_TimerIp.h`	File containing the critical sections declarations
`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

This section provides details of the code generator plugin files of the ICU driver.

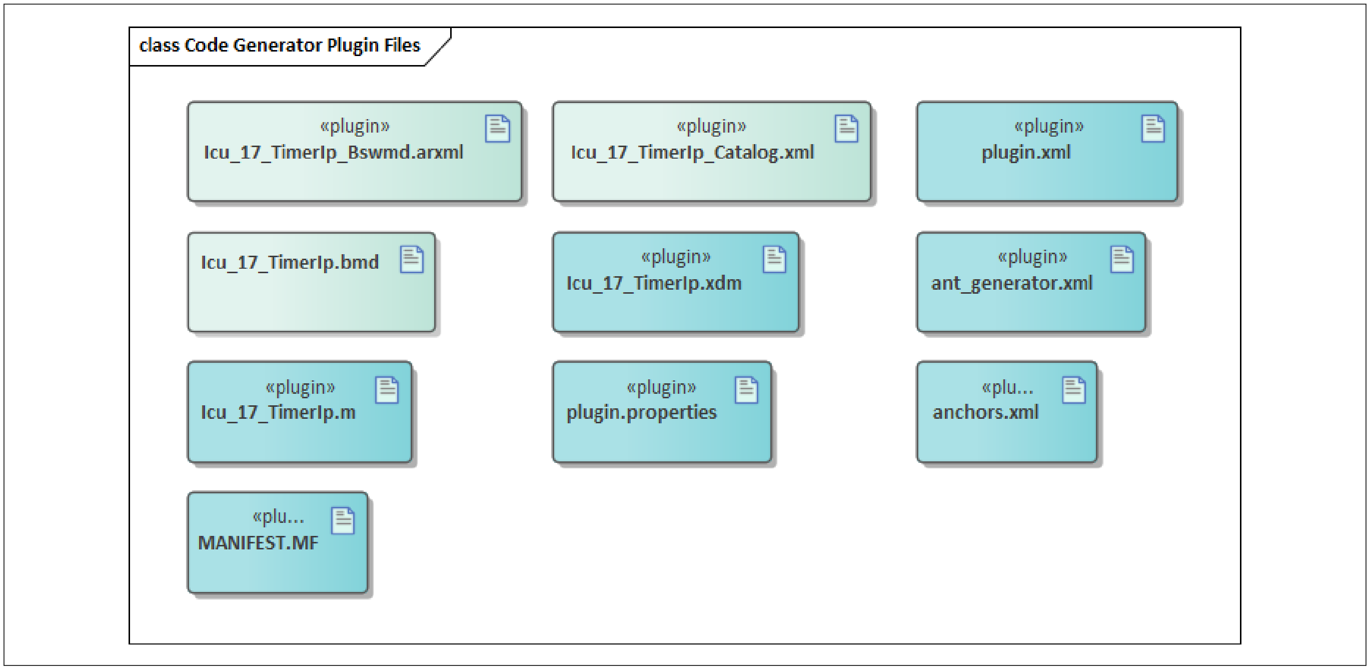
class Code Generator Plugin Files
«plugin» Icu_17_TimerIp_Bswmd.arxml
«plugin» Icu_17_TimerIp_Catalog.xml
«plugin» plugin.xml
Icu_17_TimerIp.bmd
«plugin» Icu_17_TimerIp.xdm
«plugin» ant_generator.xml
«plugin» Icu_17_TimerIp.m
«plugin» plugin.properties
«plu... anchors.xml
«plu... MANIFEST.MF

Figure 3 **Icu_Code_Generator_Plugin_Files-1.png**

Table 3 **Code generator plugin files**

File name	Description
`Icu_17_TimerIp.m`	Code template macro file for ICU driver
`Icu_17_TimerIp.xdm`	Tresos format XML data model schema file
`Icu_17_TimerIp_Bswmd.arxml`	AUTOSAR format module description file
`Icu_17_TimerIp_Catalog.xml`	AUTOSAR format catalog file
`MANIFEST.MF`	Tresos plugin support file containing the metadata for the ICU driver
`anchors.xml`	Tresos anchors support file for the ICU driver
`ant_generator.xml`	Tresos support file to generate and rename multiple post-build configuration when using variation point
`plugin.properties`	Tresos plugin support file for the ICU driver
`plugin.xml`	Tresos plugin support file for the ICU driver

1.1.4 Integration hints

This section lists the key points that an integrator or user of the ICU driver must consider. The ICU measurement modes and hardware configurations as follows:

Table 4 **ICU measurement modes and hardware configurations**

Measurement mode	Supported hardware	Supported features
Edge detection	GTM(TIM), CCU6, GPT12 and ERU	Notifications, wake-up capable, timeout (only on TIM)

(table continues...)

Table 4 (continued) ICU measurement modes and hardware configurations

Measurement mode	Supported hardware	Supported features
Multi Edge detection	GTM(TIM) and GPT12	Notifications
Edge count	GTM(TIM) and GPT12	Edge counting up to 32-bit, timeout (only on TIM)
Signal measurement	GTM(TIM) and CCU6	High time, low time, period and duty cycle
Time stamp	GTM(TIM) and CCU6	Linear and circular buffer, notifications
Incremental interface mode	GPT12(Only T2, T3 and T4)	Detect direction and position from incremental encoder. Notification on encoder counter overflow/ underflow and on every count edge.

1.1.4.1 Integration with AUTOSAR stack

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

- **EcuM**

  The ECU Manager module is a part of the AUTOSAR stack that manages common aspects of ECU. Specifically, in the context of the MCAL, the EcuM is used for initialization and de-initialization of the software drivers. User shall configure the wake-up information in EcuM configuration, which will be assigned to every wake-up capable ICU channel. The EcuM module provided in the MCAL package is a stub code and needs to be replaced with a complete EcuM module during the integration phase.
- **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 `Icu_17_TimerIp_MemMap.h` file.

  The `Icu_17_TimerIp_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 shown as follows.

```
/**** GLOBAL RAM DATA -- NON-CACHED LMU ****/
#if defined ICU_17_TIMERIP_START_SEC_VAR_CLEARED_ASIL_B_GLOBAL_32
 /*****User pragmas here for Non-cached LMU*****/
 #undef ICU_17_TIMERIP_START_SEC_VAR_CLEARED_ASIL_B_GLOBAL_32
 #undef MEMMAP_ERROR
#elif defined ICU_17_TIMERIP_STOP_SEC_VAR_CLEARED_ASIL_B_GLOBAL_32
 /*****User pragmas here for Non-cached LMU*****/
 #undef ICU_17_TIMERIP_STOP_SEC_VAR_CLEARED_ASIL_B_GLOBAL_32
 #undef MEMMAP_ERROR


/**** CORE[x] RAM DATA -- DSPR ****/ /*[x]=0..5*/
#elif defined ICU_17_TIMERIP_START_SEC_VAR_CLEARED_ASIL_B_CORE[x]_32
 /*****User pragmas here for CORE[x] DSPR*****/
 #undef ICU_17_TIMERIP_START_SEC_VAR_CLEARED_ASIL_B_CORE[x]_32
 #undef MEMMAP_ERROR
#elif defined ICU_17_TIMERIP_STOP_SEC_VAR_CLEARED_ASIL_B_CORE[x]_32
 /*****User pragmas here for CORE[x] DSPR*****/
 #undef ICU_17_TIMERIP_STOP_SEC_VAR_CLEARED_ASIL_B_CORE[x]_32
 #undef MEMMAP_ERROR

/**** CORE[x] RAM DATA INIT -- DSPR ****/ /*[x]=0..5*/
#elif defined ICU_17_TIMERIP_START_SEC_VAR_INIT_ASIL_B_CORE[x]_32
 /*****User pragmas here for CORE[x] DSPR*****/
 #undef ICU_17_TIMERIP_START_SEC_VAR_INIT_ASIL_B_CORE[x]_32
 #undef MEMMAP_ERROR
#elif defined ICU_17_TIMERIP_STOP_SEC_VAR_INIT_ASIL_B_CORE[x]_32
 /*****User pragmas here for CORE[x] DSPR*****/
 #undef ICU_17_TIMERIP_STOP_SEC_VAR_INIT_ASIL_B_CORE[x]_32
 #undef MEMMAP_ERROR

/**** GLOBAL CONST DATA -- PF[x] ****/
#elif defined ICU_17_TIMERIP_START_SEC_CONST_ASIL_B_GLOBAL_32
 /*****User pragmas here for PF[x]*****/
 #undef ICU_17_TIMERIP_START_SEC_CONST_ASIL_B_GLOBAL_32
 #undef MEMMAP_ERROR
#elif defined ICU_17_TIMERIP_STOP_SEC_CONST_ASIL_B_GLOBAL_32
 /*****User pragmas here for PF[x]*****/
 #undef ICU_17_TIMERIP_STOP_SEC_CONST_ASIL_B_GLOBAL_32
 #undef MEMMAP_ERROR

/**** GLOBAL CONFIG DATA -- PF[x] ****/
#elif defined ICU_17_TIMERIP_START_SEC_CONFIG_DATA_ASIL_B_GLOBAL_UNSPECIFIED
 /*****User pragmas here for PF[x]*****/
 #undef ICU_17_TIMERIP_START_SEC_CONFIG_DATA_ASIL_B_GLOBAL_UNSPECIFIED
 #undef MEMMAP_ERROR
#elif defined ICU_17_TIMERIP_STOP_SEC_CONFIG_DATA_ASIL_B_GLOBAL_UNSPECIFIED
 /*****User pragmas here for PF[x]*****/
 #undef ICU_17_TIMERIP_STOP_SEC_CONFIG_DATA_ASIL_B_GLOBAL_UNSPECIFIED
 #undef MEMMAP_ERROR
```

```
/**** CORE[x] CONFIG DATA -- PF[x] ****/ /*[x]=0..5*/
#elif defined ICU_17_TIMERIP_START_SEC_CONFIG_DATA_ASIL_B_CORE[x]_UNSPECIFIED
 /*****User pragmas here for PF[x]*****/
 #undef ICU_17_TIMERIP_START_SEC_CONFIG_DATA_ASIL_B_CORE[x]_UNSPECIFIED
 #undef MEMMAP_ERROR
#elif defined ICU_17_TIMERIP_STOP_SEC_CONFIG_DATA_ASIL_B_CORE[x]_UNSPECIFIED
 /*****User pragmas here for PF[x]*****/
 #undef ICU_17_TIMERIP_STOP_SEC_CONFIG_DATA_ASIL_B_CORE[x]_UNSPECIFIED
 #undef MEMMAP_ERROR


/**** CODE -- PF[x] ****/
#elif defined ICU_17_TIMERIP_START_SEC_CODE_ASIL_B_GLOBAL
 /*****User pragmas here for PF[x]*****/
 #undef ICU_17_TIMERIP_START_SEC_CODE_ASIL_B_GLOBAL
 #undef MEMMAP_ERROR
#elif defined ICU_17_TIMERIP_STOP_SEC_CODE_ASIL_B_GLOBAL
 /*****User pragmas here for PF[x]*****/
 #undef ICU_17_TIMERIP_STOP_SEC_CODE_ASIL_B_GLOBAL
 #undef MEMMAP_ERROR

#endif

#if defined MEMMAP_ERROR
#error "Icu_17_TimerIp_MemMap.h, wrong pragma command"
#endif
```

- **DET**

  The DET module is a part of the AUTOSAR stack that handles all the development errors reported by the BSW modules. The Icu driver reports all the development errors to the DET module through the `Det_ReportError()` API. The user of the ICU driver must process all the errors reported to the DET module through the `Det_ReportError()` API.

  The `Det.h` and `Det.c` files are provided in the MCAL package as a stub code and needs to be replaced with a complete DET module during the integration phase.

- **Mcal_Wrapper**

  This Driver performs reporting of the Production and Runtime errors. The Handling of the reported errors shall be done by the user. The `Mcal_Wrapper_Det_ReportRuntimeError()` API, `Mcal_Wrapper_Dem_SetEventStatus()` API and `Mcal_Wrapper_Dem_ReportErrorStatus()` API are provided in the Mcal_Wrapper.c and Mcal_Wrapper.h files as a stub code, and can be updated by the integrator to handle the reported errors. The files `Mcal_Wrapper.c` and `Mcal_Wrapper.h` are user modifiable but function prototype is not user modifiable and by default the Mcal_Wrapper functions shall call AUTOSAR DEM and DET Modules.

  Production Errors are not applicable for ICU driver. The user of the ICU driver shall process all the runtime errors reported to the Mcal_Wrapper module. The interface used for reporting Runtime Error via `Mcal_Wrapper_Det_ReportRuntimeError()` API. The `Mcal_Wrapper.c` and `Mcal_Wrapper.h` files are provided in the MCAL package as a stub code and can be replaced with a user specific runtime error handling module/s during the integration phase.

- **SchM**

The SchM module is a part of the RTE that manages the BSW Scheduler. The Icu driver uses the exclusive areas defined in the `SchM_Icu_17_TimerIp.h` file to protect the SFRs and variables from concurrent accesses from different threads. The SchMs identified for the ICU driver are:

- ResetEdgeCount
- SetActivationCondition
- GtmEnableEdgeCount
- GtmGetDutyCycle
- CcuGetDutyCycle
- CcuGetTimeElapsed

The `SchM_Icu_17_TimerIp.h` and `SchM_Icu_17_TimerIp.c` files 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 Icu 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 follows:

```
/**** Sample implementation of SchM_Icu_17_TimerIp.c ****/
#include "Os.h"

/* Disable the interrupts for entering critical section */
void SchM_Enter_Icu_17_TimerIp_ResetEdgeCount(void)
{
 SuspendAllInterrupts();
}
/* Re-enable the interrupt for exiting the critical section */
void SchM_Exit_Icu_17_TimerIp_ResetEdgeCount(void)
{
 ResumeAllInterrupts();
}

/* Disable the interrupts for entering critical section */
void SchM_Enter_Icu_17_TimerIp_SetActivationCondition(void)
{
 SuspendAllInterrupts();
}
/* Re-enable the interrupt for exiting the critical section */
void SchM_Exit_Icu_17_TimerIp_SetActivationCondition(void)
{
 ResumeAllInterrupts();
}

/* Disable the interrupts for entering critical section */
void SchM_Enter_Icu_17_TimerIp_GtmEnableEdgeCount(void)
{
 SuspendAllInterrupts();
}
/* Re-enable the interrupt for exiting the critical section */
void SchM_Exit_Icu_17_TimerIp_GtmEnableEdgeCount(void)
{
 ResumeAllInterrupts();
}

/* Disable the interrupts for entering critical section */
void SchM_Enter_Icu_17_TimerIp_GtmGetDutyCycle(void)
{
 SuspendAllInterrupts();
}
/* Re-enable the interrupt for exiting the critical section */
void SchM_Exit_Icu_17_TimerIp_GtmGetDutyCycle(void)
{
 ResumeAllInterrupts();
}
/* Disable the interrupts for entering critical section */
void SchM_Enter_Icu_17_TimerIp_CcuGetDutyCycle(void)
{
 SuspendAllInterrupts();
}
/* Re-enable the interrupt for exiting the critical section */
void SchM_Exit_Icu_17_TimerIp_CcuGetDutyCycle(void)
{
 ResumeAllInterrupts();
}
/* Disable the interrupts for entering critical section */
void SchM_Enter_Icu_17_TimerIp_CcuGetTimeElapsed(void)
{
 SuspendAllInterrupts();
}
/* Re-enable the interrupt for exiting the critical section */
void SchM_Exit_Icu_17_TimerIp_CcuGetTimeElapsed(void)
{
 ResumeAllInterrupts();
}
```

- **Safety error**

  The ICU 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 `Mcal_SafetyError.c` and `Mcal_SafetyError.h` files 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 ICU driver does not implement any notifications. However, ICU driver reports the detection of edges and desired timestamps are captured through the notification functions. These notification functions can be configured by the user in Tresos for each channel (in edge detect and time stamping mode).

  ICU does not expect any callbacks from application. But the ICU needs the callback ISR from the MCU.

- **Operating system (OS)**

  OS or the application must ensure correct type of service and interrupt priority is configured in the SR register. The enabling and disabling of interrupts must also be managed by the OS or application.

  The OS files provided by MCAL package are only an example code and must be updated by the integrator with the actual OS files for the desired function.

1.1.4.2 Multicore and Resource Manager

The Icu driver supports execution of its APIs simultaneously from all CPU cores. The user has to allocate each channel of the ICU to CPU cores at pre-compile time using the Resource Manager module. The following are the key points to be considered with respect to multicore in the ICU driver:

- Each ICU channel can be allocated to any core using the Resource Manager.
- For ICU channel dependent on ERU, channels using `OGU[x]` and `OGU[x+4]` where `(x=0-3)`, must be allocated to same core as these two channels share same interrupt line. For example, ICU channels using OGU0 and OGU4 should be allocated to same core.
- The locating of constants, variables and configuration data to correct memory space should be done by the user. Memory sections are marked GLOBAL(common to all cores) and CORE[x](specific to a CPU core). The following should be considered by the user to ensure better performance of the driver:

  Code section:

The executable code of the ICU driver is placed under single MemMap section. It can be relocated to any PFlash/DFlash region.

Data section:

The RAM variable memory sections marked as specific to a core should be re-located to the DSPR/DLMU of the same core. The sections marked as global should be relocated to the non-cached LMU region.

Configuration data and constants:

The configuration data sections marked as specific to a core should be re-located to the PFlash/DFlash of the same core. The sections marked as global should be relocated to the PFlash/DFlash of the master core.

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

Note: If the driver operates from single (master) core, all the sections may be relocated to the PFlash/DSPR/DLMU of the same CPU core.

1.1.4.3 MCU support

The Icu driver is dependent on MCU driver for clock configuration and timer-IP related services. The initialization of the ICU driver must be initialized only after successful completion of the MCU initialization. The following must be considered while configuring the MCU driver in the Eb tresos:

- GTM-TIM Icu channel: The GTM-TIM channels used by Icu driver must be reserved in the MCU configuration for exclusive use by Icu. The reserved TIM channel can be used for one ICU channel. The port pin connection to the TIM channel must be configured in the MCU.
- CCU6 Icu channel: The CCU6 kernel used by Icu driver must be reserved in the MCU configuration for exclusive use by Icu. Hence, each of the three comparators(CC60, CC61 and CC62) of the reserved CCU6 kernel can be used for three different ICU channels.
- GPT12 Icu channel: The GPT12 timer used by Icu driver must be reserved in the MCU configuration for exclusive use by Icu. The reserved timer can be used for one ICU channel.
- ERU Icu channel: The ERS channel and corresponding OGU channel used by Icu driver must be reserved in the MCU configuration for exclusive use by Icu. An ERS channel can be paired with any OGU channel.

1.1.4.4 Port support

The PORT driver configures the port pins of the entire microcontroller. The user must configure port pins used by the ICU driver through the Port configuration and initialize the port pins prior to invoking the ICU driver initialization.

1.1.4.5 DMA support

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

1.1.4.6 Interrupt connections

The interrupt configuration registers of different hardware used by ICU channels are as follows:

Table 5 SRC registers

Hardware used	SRC register
GTM-TIM	SRC_GTMTIMwx (w= TIM module; x= TIM channel)
CCU6	SRC_CCU6xSRy (x= CCU6 kernel; y=0-3)
ERU	SRC_SCUERUx (x=0-3)

(table continues...)

Table 5 (continued) SRC registers

Hardware used	SRC register
GPT12	SRC_GPT120Tx (x=2-6, GPT12 timer)

All the ISR to GTM-TIM must be routed to the `Mcu_17_Gtm_TimChannelIsr` API, which further invokes `Icu_17_TimerIp_Timer_Isr`. The example ISR handling is shown as follows:

```
ISR(GTMTIM0SR0_ISR)
{
 /* Enable Global Interrupts */
 ENABLE();
 /* Parameter is TIM module number and TIM channel number */
 Mcu_17_Gtm_TimChannelIsr(0, 0); /* For TIM 0 CH0*/
}
```

All the ISR to CCU6 comparator must be routed to the `Mcu_17_Ccu6_ChannelIsr` API, which further invokes `Icu_17_TimerIp_Timer_Isr`. The example ISR handling is shown as follows:

```
ISR(CCU60SR0_ISR)
{
 /* Enable Global Interrupts */
 ENABLE();
 /* Parameter are CCU6 kernel and comparator */
 Mcu_17_Ccu6_ChannelIsr(CCU6_KERNEL_0,CCU6_CHANNEL_0);
}
```

If the CCU6 ISR is pre-empted and an input edge is observed during the pre-emption, the high time, low time and duty cycle values measured by the ICU module are not correct. Hence if the user foresees such corner case, it is recommended to execute the CCU6 ISR `CCU6xSRy_ISR` in critical section.

All the ISR to GPT12 timer must be routed to the `Mcu_17_Gpt12_ChannelIsr` API, which further invokes `Icu_17_TimerIp_Timer_Isr`. The example ISR handling is shown as follows:

```
ISR(GPT12_T2_ISR)
{
 /* Enable Global Interrupts */
 ENABLE();
 /* Parameter is GPT12 timer number 0 for T2, 1 for T3 and so on.*/
 Mcu_17_Gpt12_ChannelIsr(0); /* For T2 timer */
}
```

All the ISR to ERU channel must be routed to the `Mcu_17_Eru_GatingIsr` API, which further invokes `Icu_17_TimerIp_Timer_Isr`. The example ISR handling is shown as follows:

```
ISR(SCUERUSR0_ISR)
{
 /* Enable Global Interrupts */
 ENABLE();
 /* Call Mcu Interrupt function, parameter is SRC index. */
 Mcu_17_Eru_GatingIsr(0); /* For ERS 0*/
}
```

1.1.4.7 Example usage

Initialization

User must include `Icu_17_TimerIp.h` file to access the ICU configuration structure needed for initialization.

```
/* Include Icu.h to access configuration structures */
#include "Icu_17_TimerIp.h"
/* Module Initialization */
void Icu_Sample_Init(void)
{
 /* MCU initializations */
 Mcu_Init(&Mcu_Config);
 (void)Mcu_InitClock( 0 );
 while(Mcu_GetPllStatus() != MCU_PLL_LOCKED)
 {
 };
 (void)Mcu_DistributePllClock();

 /* Initialize ICU */
 Icu_17_TimerIp_Init(&Icu_17_TimerIp_Config);

 /* Initialize Check for ICU */
 Error = Icu_17_TimerIp_InitCheck(&Icu_17_TimerIp_Config);

 if (Error == E_OK)
 {
 /*ICU InitCheck should pass, then Call other APIs related to ICU */
 }
}
```

Edge count mode

Edge counting activity on an edge count configured ICU channel starts by the call to `Icu_17_TimerIp_EnableEdgeCount`. The `Icu_17_TimerIp_GetEdgeNumbers` API returns the number of edges counted after the call to `Icu_17_TimerIp_Init` or `Icu_17_TimerIp_ResetEdgeCount`. Edge counting activity is stopped by the call to `Icu_17_TimerIp_DisableEdgeCount`. The `Icu_17_TimerIp_ResetEdgeCount` API resets the number of counted edges even if the edge counting activity is stopped.

```
/* ICU module Initialization is necessary to start the edge counting feature */
Icu_17_TimerIp_Init(&Icu_17_TimerIp_Config);
Icu_17_TimerIp_EnableEdgeCount(<logical channel symbolic name>);

/* Provide edges on the port pin of the logical channel */

/* Read the number of edges detected. Returns the number of provided edges */
CountedEdges = Icu_17_TimerIp_GetEdgeNumbers(<logical channel symbolic name>);

/* Reset the counted edges */
Icu_17_TimerIp_ResetEdgeCount(<logical channel symbolic name>);

/* Read edge count after reset edge count. "0" will be returned */
CountedEdges = Icu_17_TimerIp_GetEdgeNumbers(<logical channel symbolic name>);

/* Provide edges on the port pin of the logical channel */

/* Read the number of edges detected. Returns the number of provided edges */
CountedEdges = Icu_17_TimerIp_GetEdgeNumbers(<logical channel symbolic name>);

/* Disable edge counting */
Icu_17_TimerIp_DisableEdgeCount(<logical channel symbolic name>);

/* Reset the counted edges */
Icu_17_TimerIp_ResetEdgeCount(<logical channel symbolic name>);

/* Read the number of edges detected. Returns "0" as edge counting is reset.*/
CountedEdges = Icu_17_TimerIp_GetEdgeNumbers(<logical channel symbolic name>);

/* Provide edges on the port pin of the logical channel */

/* Read the number of edges detected. Returns "0" as edge counting is disabled.*/
CountedEdges = Icu_17_TimerIp_GetEdgeNumbers(<logical channel symbolic name>);
```

Time stamp mode

Capturing of time stamps on the configured active edge is started by the call to `Icu_17_TimerIp_StartTimestamp`. `Icu_17_TimerIp_EnableNotification` should be invoked to receive notifications after receiving a specific number(configured at the start of activity) of timestamps (applicable only if the notification function is configured in the module configuration). `Icu_17_TimerIp_DisableNotification` stops issuing the notifications.

`Icu_17_TimerIp_StopTimestamp` stops capturing time stamps. `Icu_17_TimerIp_GetTimestampIndex` returns the buffer position which is to be filled next.

```
 /* ICU module Initialization is necessary to start the time stamping feature */
 Icu_17_TimerIp_Init(&Icu_17_TimerIp_Config);

 /* Buffer to fill timestamps. BUFFER_SIZE indicates the size of the buffer */
 Icu_17_TimerIp_ValueType Buffer[BUFFER_SIZE];

 /* Start the time stamping activity. NOTIFY_INTERVAL is the number of timestamps to be
received to issue notification*/
 Icu_17_TimerIp_StartTimestamp(<logical channel symbolic name>, Buffer, BUFFER_SIZE,
NOTIFY_INTERVAL);

 /* Enable notifications to receive notifications */
 Icu_17_TimerIp_EnableNotification(<logical channel symbolic name>);

 /* Provide edges on the port pin of the logical channel */

 /* Read the buffer index next to be filled */
 NextIndex = Icu_17_TimerIp_GetTimestampIndex(<logical channel symbolic name>);

 /* Notification function would have been invoked if sufficient number of edges are provided */

 /* Disable Notifications */
 Icu_17_TimerIp_DisableNotification(<logical channel symbolic name>);

 /* Provide edges on the port pin of the logical channel */

 /* Read the buffer index next to be filled. The index read here will be different from the
previous read as the time stamping activity is still active */
 NextIndex = Icu_17_TimerIp_GetTimestampIndex(<logical channel symbolic name>);

 /* Notification function would not have been invoked even if sufficient number of edges are
provided */

 /* Disable time stamping */
 Icu_17_TimerIp_StopTimestamp(<logical channel symbolic name>);

 /* Provide edges on the port pin of the logical channel */

 /* Notification function would not have been invoked even if sufficient number of edges are
provided */

 /* Read the buffer index next to be filled. The index read here will be same from the previous
read as the time stamping activity is disabled */

 NextIndex = Icu_17_TimerIp_GetTimestampIndex(<logical channel symbolic name>);
```

Signal measurement mode

Measurement of high time, low time, period or duty cycle (as per the configuration) starts after the call to `Icu_17_TimerIp_StartSignalMeasurement`. The availability of a new measured value is identified by the call to

`Icu_17_TimerIp_GetInputState`. `Icu_17_TimerIp_GetTimeElapsed` reads the measured high time, low time or period. `Icu_17_TimerIp_GetDutyCycleValues` reads the measured coherent period and active time. The signal measurement activity is stopped by the call to `Icu_17_TimerIp_StopSignalMeasurement` and restarted by the call to `Icu_17_TimerIp_StartSignalMeasurement`.

```
/* ICU module Initialization is necessary to start the signal measurement feature */
Icu_17_TimerIp_Init(&Icu_17_TimerIp_Config);

/* Start signal measurement activity */
Icu_17_TimerIp_StartSignalMeasurement(<non duty cycle logical channel symbolic name>);
Icu_17_TimerIp_StartSignalMeasurement(<duty cycle logical channel symbolic name>);

/* Provide edges on the port pin of the logical channel */

/* Check channel status before reading the corresponding values.*/
if(ICU_ACTIVE == Icu_17_TimerIp_GetInputState(<non duty cycle logical channel symbolic name>))
{
 /* Channel status is active read the measured time */
 SignalMeasureValue = Icu_17_TimerIp_GetTimeElapsed(<non duty cycle logical channel symbolic
name>);

 /* Read the measured time again, this will return 0 */
 SignalMeasureValue = Icu_17_TimerIp_GetTimeElapsed(<non duty cycle logical channel symbolic
name>);
}
if(ICU_ACTIVE == Icu_17_TimerIp_GetInputState(<duty cycle logical channel symbolic name>))
{
 Icu_17_TimerIp_DutyCycleType DutyCycle;

 /* Channel status is active, read the duty cycle values */
 Icu_17_TimerIp_GetDutyCycleValues(<duty cycle logical channel symbolic name>, &DutyCycle);

 /* read the duty cycle values again. this will return 0 */
 Icu_17_TimerIp_GetDutyCycleValues(<duty cycle logical channel symbolic name>, &DutyCycle);
}

/* Stop signal measurement activity */
Icu_17_TimerIp_StopSignalMeasurement(<non duty cycle logical channel symbolic name>);
Icu_17_TimerIp_StopSignalMeasurement(<duty cycle logical channel symbolic name>);

/* Provide edges on the port pin of the logical channel */

/* Check channel status, will return IDLE as signal measurement activity is stopped */
ChannelState = Icu_17_TimerIp_GetInputState(<non duty cycle logical channel symbolic name>));
ChannelState = Icu_17_TimerIp_GetInputState(<duty cycle logical channel symbolic name>));

/* Start signal measurement activity */
Icu_17_TimerIp_StartSignalMeasurement(<non duty cycle logical channel symbolic name>);
Icu_17_TimerIp_StartSignalMeasurement(<duty cycle logical channel symbolic name>);

/* Provide edges on the port pin of the logical channel */

/* Check channel status, will return ACTIVE as signal measurement activity is started */
ChannelState = Icu_17_TimerIp_GetInputState(<non duty cycle logical channel symbolic name>));
ChannelState = Icu_17_TimerIp_GetInputState(<duty cycle logical channel symbolic name>));
```

Edge detection mode

Detection of edges (rising, falling or any as per configuration) on an edge detect configured ICU channel starts after the call to `Icu_17_TimerIp_Init`. The ICU channel status is identified by the call to `Icu_17_TimerIp_GetInputState`. `Icu_17_TimerIp_EnableNotification` should be invoked to receive notifications on the configured edges (applicable only if the notification function is configured in the module configuration). `Icu_17_TimerIp_DisableNotification` stops issuing the notifications. The edge detection activity is stopped by the call to `Icu_17_TimerIp_DisableEdgeDetection` and re-enabled by the call to

`Icu_17_TimerIp_EnableEdgeDetection`. `Icu_17_TimerIp_EnableMultiEdgeDetection` detects multiple edges and issue notifications after multiple edges are detected.

```
/* ICU module Initialization is necessary to start the edge detection activity */
Icu_17_TimerIp_Init(&Icu_17_TimerIp_Config);

/* Provide edges on the port pin of the logical channel */

/* Check channel status, will return ACTIVE and no notifications are detected */
ChannelState = Icu_17_TimerIp_GetInputState(<logical channel symbolic name>);

/* Read channel state again, will return IDLE */
ChannelState = Icu_17_TimerIp_GetInputState(<logical channel symbolic name>);

/* Enable Notifications */
Icu_17_TimerIp_EnableNotification(<logical channel symbolic name>);

/* Provide edges on the port pin of the logical channel */

/* Channel respective notification function would have been issued.*/

/* Disable Notifications */
Icu_17_TimerIp_DisableNotification(<logical channel symbolic name>);

/* Provide edges on the port pin of the logical channel */

/* Channel respective notification function would not have been issued */

/* Check channel status, will return ACTIVE */
ChannelState = Icu_17_TimerIp_GetInputState(<logical channel symbolic name>);

/* Disable Edge detection */
Icu_17_TimerIp_DisableEdgeDetection(<logical channel symbolic name>);

/* Provide edges on the port pin of the logical channel */

/* Channel respective notification function would not have been issued */

/* Check channel status, will return IDLE and no notifications are detected */
ChannelState = Icu_17_TimerIp_GetInputState(<logical channel symbolic name>);

/* Enable multiple edge detection. EDGE_COUNT number of edge will be detected */
Icu_17_TimerIp_EnableMultiEdgeDetection(<logical channel symbolic name>, EDGE_COUNT);

/* Enable notifications */
Icu_17_TimerIp_EnableNotification(<logical channel symbolic name>);

/* Provide multiple edges on the port pin of the logical channel */

/* Notification would have been issued after detecting EDGE_COUNT edges.*/
```

General API

`Icu_17_TimerIp_SetActivationCondition` should be invoked after the call to `Icu_17_TimerIp_Init` and only for channels which are in edge detection, time stamping and edge counting modes. `Icu_17_TimerIp_EnableWakeup` and `Icu_17_TimerIp_DisableWakeup` should be invoked after the call to `Icu_17_TimerIp_Init` and only on a wakeup capable channel. `Icu_17_TimerIp_SetMode` should be invoked after the call to `Icu_17_TimerIp_Init` to change the state of the ICU driver to **SLEEP** or **NORMAL**.`Icu_17_TimerIp_DeInit`, should be invoked after the call to `Icu_17_TimerIp_Init` to reset the initialization state of ICU module. After the call to `Icu_17_TimerIp_DeInit`, `Icu_17_TimerIp_Init` should be invoked again to start any functionality of the ICU driver.

```
/* ICU module Initialization is necessary to start the ICU channel activities */
Icu_17_TimerIp_Init(&Icu_17_TimerIp_Config);

/* Default edge for detection, edge counting and time stamping is taken from configuration */

/* Change the default active edge to RISING_EDGE */
Icu_17_TimerIp_SetActivationCondition(<logical channel symbolic name>,ICU_RISING_EDGE);

/* Edge detection, edge counting and time stamping will be done on rising edge */

/* Enable wakeup for a wakeup capable channel */
Icu_17_TimerIp_EnableWakeup(<wakeup capable logical channel symbolic name>);

/* Change mode to sleep */
Icu_17_TimerIp_SetMode(ICU_MODE_SLEEP);

/* provide signal on the wakeup capable channel */

/* EcuMCheckWakeup will be invoked from ISR to indicate a wakeup signal */

/* Change mode to normal */
Icu_17_TimerIp_SetMode(ICU_MODE_NORMAL);

/* Disable wakeup for a wakeup capable channel */
Icu_17_TimerIp_DisableWakeup(<wakeup capable logical channel symbolic name>);

/* Change mode to sleep */
Icu_17_TimerIp_SetMode(ICU_MODE_SLEEP);

/* provide signal on the wakeup capable channel */

/* EcuMCheckWakeup will not be invoked */
```

Incremental interface mode

The encoder count is set to 0 after initialization. Detection of encoder edges on an incremental interface configured ICU channel starts after the call to `Icu_17_TimerIp_StartIncInterface`. The encoder count is identified by the call to `Icu_17_TimerIp_ReadEncCount` and direction by the call to

`Icu_17_TimerIp_ReadEncCountDir`. `Icu_17_TimerIp_CalibratePos` should be invoked to calibrate the initial encoder position. The incremental interface activity is stopped by the call to `Icu_17_TimerIp_StopIncInterface`.

```
/* ICU module Initialization is necessary to start the incremental interface activity */
Icu_17_TimerIp_Init(&Icu_17_TimerIp_Config);

/* Check encoder count and direction, will return 0 and UP direction(HW default) */
EncCount = Icu_17_TimerIp_ReadEncCount(<logical channel symbolic name>);
EncDir = Icu_17_TimerIp_ReadEncCountDir(<logical channel symbolic name>);

/* Enable incremental interface channel */
Icu_17_TimerIp_StartIncInterface(<logical channel symbolic name>);

/* Provide edges on the port pin(both counter and direction signal) of the logical channel */

/* No notifications are issued */

/* Check encoder count and direction, will return non zero and current counter direction(HW
default) as per the given input signals */
EncCount = Icu_17_TimerIp_ReadEncCount(<logical channel symbolic name>);
EncDir = Icu_17_TimerIp_ReadEncCountDir(<logical channel symbolic name>);

/* Update the encoder position */
Icu_17_TimerIp_CalibratePos(<logical channel symbolic name>, <new counter position to be set>);

/* Enable Notifications */
Icu_17_TimerIp_EnableNotification(<logical channel symbolic name>);

/* Provide edges on the port pin(both counter and direction signal) of the logical channel */

/* Notifications for each count edge is issued */
/* Notifications for counter overflow/underflow is issued if the encoder counter overflows or
underflows */

/* Check encoder count and direction, will return non zero and current counter direction(HW
default) as per the given input signals starting from the encoder position set */
EncCount = Icu_17_TimerIp_ReadEncCount(<logical channel symbolic name>);
EncDir = Icu_17_TimerIp_ReadEncCountDir(<logical channel symbolic name>);

/* Disable incremental interface channel */
Icu_17_TimerIp_StopIncInterface(<logical channel symbolic name>);

/* Provide edges on the port pin(both counter and direction signal) of the logical channel */

/* Check encoder count and direction, will return same as the previous call as the incremental
mode is disabled. */
EncCount = Icu_17_TimerIp_ReadEncCount(<logical channel symbolic name>);
EncDir = Icu_17_TimerIp_ReadEncCountDir(<logical channel symbolic name>);
```

Timeout functionality

Timeout functionality can be enabled in configuration on ICU channel using TIM hardware and in edge detect or edge count mode. After ICU initialization timeout functionality is disabled. The feature can be enabled by invoking `Icu_17_TimerIp_SetTimeoutValue` API with appropriate timeout value. The configured timeout notification is issued if the configured edge is not identified until the timeout expires. The timeout detection can be disabled in runtime by configuring the timeout value to 0.

```
/* ICU module Initialization is necessary to start the edge detection/edge count activity */
Icu_17_TimerIp_Init(&Icu_17_TimerIp_Config);

/* Provide some configured timeout edges and stop the signal */

/* Timeout notification is not issued since the timeout value is not configured */

Icu_17_TimerIp_SetTimeoutValue(<logical channel symbolic name>,<timeout value>);

/* Provide some configured timeout edges and stop the signal and wait for the timeout to expire
*/

/* Timeout notification is not issued since the notifications are not enabled */

/* Enable Notifications */
Icu_17_TimerIp_EnableNotification(<logical channel symbolic name>);

/* Provide some configured timeout edges and stop the signal */

/* Timeout notification is issued after the timeout value expired */
```

1.1.5 Key architectural considerations

1.1.5.1 Overflow handling for signal measurement

For ICU channel in signal measurement mode using the GTM-TIM hardware, the measured value overflow will be detected and measured value will be returned as 0 until next valid measurement.

1.1.5.2 Accessing shared SFR

ICU channel using ERU will access `MODULE_SCU.EICR`, `MODULE_SCU.IGCR` and `MODULE_SCU.FMR` to configure the corresponding ERS and OGU channels. The ICU driver ensures the access is done atomically and with proper masks to not update the unintended part of register. Hence any application module accessing these register must also perform atomic access and with proper mask to ensure no interference with the ICU functionalities.

1.1.5.3 GPT12 Timer allocations in incremental interface mode

If a GPT12 Timer(T2/T3) is configured to use the clear timer feature, the user shall configure the GPT12 T4 timer for the ICU driver with the proper clear timer trigger. Also the GPT12 Timer(T2/T3) and T4 shall be assigned to the same core.
[cover parentID ICU={966EBAA3-B5EE-4aa3-BC5A-15227D054C11}]

1.2 Assumptions of Use (AoU)

The AoU for the ICU driver are as follows.

- **Edge counter overflow**

Edge counter value returned by the API restarts the count from 0 once it reaches 0xFFFFFFFF.

User should consider this behavior while using the Icu_17_TimerIp_GetEdgeNumbers API.

[cover parentID ICU={3A84EBDA-FA2B-49dd-9CBB-67ADDAD9CFE6}]

- **Execution sequence for Initialization Check**

If configured, Icu_17_TimerIp_InitCheck() API shall be called after the ICU driver initialization and before starting any functionality of the ICU driver.

[cover parentID ICU={578BB26C-7DA4-4f00-8D8D-B8F6D5418CD2}]

- **Generated configuration structure - AoU**

User shall ensure the generated configuration structures are correct against the intended GUI configurations.

[cover parentID ICU={64E84977-40E5-4ebb-9F06-FE12BE63B8E3}]

- **ICU common ERU ISR**

ICU channels using OGU[x] and OGU[x+4] (x=0-3), in case of ERU hardware, shall be allocated to the same core as these two channels share the same interrupt line. For example, ICU channels using OGU0 and OGU4 should be allocated to the same core.

[cover parentID ICU={B6252F81-63DA-442d-B8EE-1CA887EEA003}]

- **ICU signal measurement for CCU6 hardware**

For an ICU channel configured in the signal measurement mode using

CCU6 hardware, the user shall ensure the measured value is in the 16-bit range. The overflow cannot be identified due to hardware limitation.

[cover parentID ICU={5FE037D2-D596-4418-9ADA-0E39896716F8}]

1.3 Reference information

1.3.1 Configuration interfaces

Supported configuration variant: Post-Build

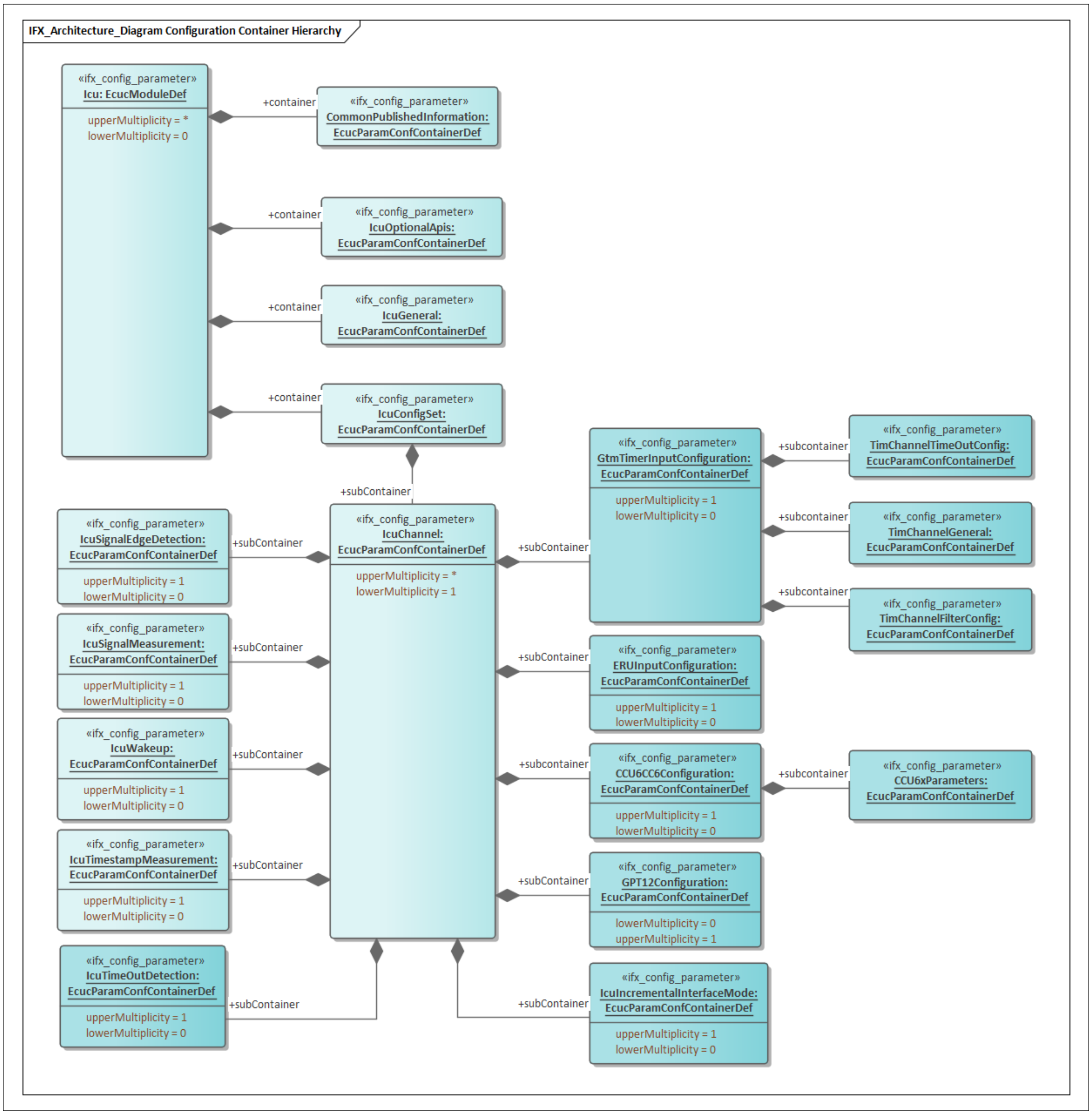
IFX_Architecture_Diagram Configuration Container Hierarchy
«ifx_config_parameter» Icu: EcucModuleDef
upperMultiplicity = *
lowerMultiplicity = 0
+container
«ifx_config_parameter» CommonPublishedInformation: EcucParamConfContainerDef
«ifx_config_parameter» IcuOptionalApis: EcucParamConfContainerDef
«ifx_config_parameter» IcuGeneral: EcucParamConfContainerDef
«ifx_config_parameter» IcuConfigSet: EcucParamConfContainerDef
+subContainer
«ifx_config_parameter» IcuChannel: EcucParamConfContainerDef
upperMultiplicity = *
lowerMultiplicity = 1
«ifx_config_parameter» IcuSignalEdgeDetection: EcucParamConfContainerDef
«ifx_config_parameter» IcuSignalMeasurement: EcucParamConfContainerDef
«ifx_config_parameter» IcuWakeup: EcucParamConfContainerDef
«ifx_config_parameter» IcuTimestampMeasurement: EcucParamConfContainerDef
«ifx_config_parameter» IcuTimeOutDetection: EcucParamConfContainerDef
«ifx_config_parameter» GtmTimerInputConfiguration: EcucParamConfContainerDef
«ifx_config_parameter» ERUInputConfiguration: EcucParamConfContainerDef
«ifx_config_parameter» CCU6CC6Configuration: EcucParamConfContainerDef
«ifx_config_parameter» GPT12Configuration: EcucParamConfContainerDef
«ifx_config_parameter» IcuIncrementalInterfaceMode: EcucParamConfContainerDef
+subcontainer
«ifx_config_parameter» TimChannelTimeOutConfig: EcucParamConfContainerDef
«ifx_config_parameter» TimChannelGeneral: EcucParamConfContainerDef
«ifx_config_parameter» TimChannelFilterConfig: EcucParamConfContainerDef
«ifx_config_parameter» CCU6xParameters: EcucParamConfContainerDef
upperMultiplicity = 1
lowerMultiplicity = 0

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

1.3.1.1 Container: TimChannelTimeOutConfig

This holds the timeout detection configurations for the TIM channels.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.1.1 TimChTimeOutCounterFreqSelect

Table 6 Specification for TimChTimeOutCounterFreqSelect

Name	`TimChTimeOutCounterFreqSelect`		
Description	The parameter decides the timeout counter frequency for the TIM Channel. The parameter is non-editable if IcuTimeoutFeature is TIMEOUT_DISABLED.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	GTM_CONFIGURABLE_CLOCK_0: Timeout counters use configurable clock 0 GTM_CONFIGURABLE_CLOCK_1: Timeout counters use configurable clock 1 GTM_CONFIGURABLE_CLOCK_2: Timeout counters use configurable clock 2 GTM_CONFIGURABLE_CLOCK_3: Timeout counters use configurable clock 3 GTM_CONFIGURABLE_CLOCK_4: Timeout counters use configurable clock 4 GTM_CONFIGURABLE_CLOCK_5: Timeout counters use configurable clock 5 GTM_CONFIGURABLE_CLOCK_6: Timeout counters use configurable clock 6 GTM_CONFIGURABLE_CLOCK_7: Timeout counters use configurable clock 7		
Default value	GTM_CONFIGURABLE_CLOCK_0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	IcuTimeoutFeature		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.2 TimChTimeOutEdge

Table 7 Specification for TimChTimeOutEdge

Name	`TimChTimeOutEdge`		
Description	The parameter decides the edge for time out detection for the TIM Channel. The parameter is non-editable if IcuTimeoutFeature is TIMEOUT_DISABLED.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	TDU_BOTH_EDGES: Both edges starts/resets the timeout counter TDU_FALLING_EDGE: Falling edge starts/resets the timeout counter TDU_RISING_EDGE: Rising edge starts/resets the timeout counter		
Default value	TDU_RISING_EDGE		

(table continues...)

Table 7 (continued) Specification for TimChTimeOutEdge

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

1.3.1.1.3 TimChannelTimeoutInputSelect

Table 8 Specification for TimChannelTimeoutInputSelect

Name	`TimChannelTimeoutInputSelect`		
Description	The parameter decides the input signal for the time out detection for the TIM Channel. The parameter is non-editable if IcuTimeoutFeature is TIMEOUT_DISABLED.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	INPUT_OF_CURRENT_TIM_CHANNEL: Input to the timeout detection is the input assigned to current TIM channel INPUT_OF_PREVIOUS_TIM_CHANNEL: Input to the timeout detection is the input of the previous TIM channel		
Default value	INPUT_OF_CURRENT_TIM_CHANNEL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	IcuTimeoutFeature		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2 Container: CCU6CC6Configuration

The container contains the configuration for CC6x comparator if the hardware unit selected is CCU6.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Pre-Compile

1.3.1.2.1 CCChannelInputSelection

Table 9 **Specification for CCChannelInputSelection**

Name	`CCChannelInputSelection`		
Description	Input selection for Cc6xchannel. *Note: The parameter reads from property file and depends upon channel selection. Default value is set to first encountered value from Resource property file.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	CCINx_[inputline]: x can be A,B,C and so on depending on HW. 'inputline' can be PORT PIN or internal signal from other peripheral.		
Default value	CCINx_[inputline]		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	CCU6KernelUsed, Cc6xChannel		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2.2 CCU6KernelUsed

Table 10 **Specification for CCU6KernelUsed**

Name	`CCU6KernelUsed`		
Description	The parameter is a list of CCU6 kernels available to the ICU Driver. The chosen CCU6 kernel shall be reserved for usage by ICU. The value of the parameter cannot be changed across variants since the referred value, McuCcu6ModuleAllocationConf, is a pre-compiler parameter. *Note: Default value is set to blank as user has to select the appropriate reference value from MCU.*		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: McuCcu6ModuleAllocationConf		
Default value	NULL		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL

(table continues...)

Table 10 **(continued) Specification for CCU6KernelUsed**

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

1.3.1.2.3 Cc6xChannel

Table 11 **Specification for Cc6xChannel**

Name	`Cc6xChannel`		
Description	Selection of a CC6x channel. Since the CCU6InterruptNode parameter cannot be varied across variants, the Cc6xChannel parameter cannot be varied across variants. *Note: Default value is chosen as Cc60 as it is the CCU6 lowest comparator.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	Cc60: Selection of CC60 Capture Cc61: Selection of CC61 Capture Cc62: Selection of CC62 Capture		
Default value	Cc60		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**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: CCU6xParameters

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.3.1 CCU6InterruptNode

Table 12 **Specification for CCU6InterruptNode**

Name	`CCU6InterruptNode`
Description	Interrupt node to be used for the kernel. *Note: Default value is chosen as 0 as Hardware Default value is 0.* *Note: The parameter directly influences the configuration of IRQ module, which is a pre-compile module. Hence the parameter value cannot be varied across variants.*

(table continues...)

Table 12 (continued) Specification for CCU6InterruptNode

Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	NODE_SR0: Service request output SR0 is selected NODE_SR1: Service request output SR1 is selected NODE_SR2: Service request output SR2 is selected NODE_SR3: Service request output SR3 is selected		
Default value	NODE_SR0		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3.2 T12ClkSelection

Table 13 Specification for T12ClkSelection

Name	`T12ClkSelection`		
Description	T12 clock divider configuration for the kernel used. *Note: Effective clock divider is 2^(value configured).* *Note: Default value is chosen as 0 as Hardware Default value is 0.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 7		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	CCU6KernelUsed		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3.3 T12PrescalerEnabled

Table 14 Specification for T12PrescalerEnabled

Name	`T12PrescalerEnabled`		
Description	The parameter to determine if the additional (1/256) pre-scalar should be added to the clock path. If T12PrescalerEnabled is false no additional pre-scalar is added else pre-scalar is added. *Note: Default value is chosen as FALSE as Hardware Default value is FALSE.*		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	CCU6KernelUsed		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4 Container: CommonPublishedInformation

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.4.1 ArMajorVersion

Table 15 Specification for ArMajorVersion

Name	`ArMajorVersion`		
Description	AUTOSAR major version.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	4		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL

(table continues...)

Table 15 (continued) Specification for ArMajorVersion

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

1.3.1.4.2 ArMinorVersion

Table 16 Specification for ArMinorVersion

Name	`ArMinorVersion`		
Description	AUTOSAR minor version.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per 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.		

1.3.1.4.3 ArPatchVersion

Table 17 Specification for ArPatchVersion

Name	`ArPatchVersion`		
Description	AUTOSAR patch version.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per 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.		

1.3.1.4.4 ModuleId

Table 18 Specification for ModuleId

Name	`ModuleId`		
Description	Parameter to provide the module identifier.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 65535		
Default value	122		
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.		

1.3.1.4.5 Release

Table 19 Specification for Release

Name	`Release`		
Description	Aurix derivative used for the implementation.		
Multiplicity	1..1	**Type**	EcucStringParamDef
Range	String		
Default value	As per the configuration.		
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 SwMajorVersion

Table 20 Specification for SwMajorVersion

Name	`SwMajorVersion`
Description	Module major version.

(table continues...)

Table 20 (continued) Specification for SwMajorVersion

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.		

1.3.1.4.7 SwMinorVersion

Table 21 Specification for SwMinorVersion

Name	`SwMinorVersion`		
Description	Module minor version.		
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.		

1.3.1.4.8 SwPatchVersion

Table 22 Specification for SwPatchVersion

Name	`SwPatchVersion`		
Description	Module patch version.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per the driver version		

(table continues...)

Table 22 (continued) Specification for SwPatchVersion

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 VendorApiInfix

Table 23 Specification for VendorApiInfix

Name	`VendorApiInfix`		
Description	The parameter is used to specify the vendor specific name. *Note: "TimerIp" is chosen as VendorApiInfix as all functionalities of ICU are achieved using the timer modules GTM-TIM, CCU6 and GPT12.*		
Multiplicity	1..1	**Type**	EcucStringParamDef
Range	String		
Default value	TimerIp		
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 VendorId

Table 24 Specification for VendorId

Name	`VendorId`		
Description	Infineon vendor ID in HIS software specification.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 65535		
Default value	17		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-

(table continues...)

Table 24 **(continued) Specification for VendorId**

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.		

1.3.1.5 Container: ERUInputConfiguration

The container contains the configuration for ERS and OGU channel if the hardware unit selected is ERU.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Pre-Compile

1.3.1.5.1 EruErsReference

Table 25 **Specification for EruErsReference**

Name	`EruErsReference`		
Description	The parameter is a reference to the ERS container in the MCU. The chosen ERS channel shall be reserved for usage by ICU. The ERS channel selected in the configuration shall be unique in the ICU configuration. The value of the parameter cannot be changed across variants since the referred value, McuEruChannelInputLineConf, is a pre-compiler parameter. *Note: Default value is set to blank as user has to select the appropriate reference value from MCU.*		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: McuEruChannelInputLineConf		
Default value	NULL		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5.2 EruInputPin

Table 26 **Specification for EruInputPin**

Name	`EruInputPin`

(table continues...)

Table 26 (continued) Specification for EruInputPin

Description	The input pin selection for the ERU unit. *Note: The parameter reads from property file and depends upon channel selection. Default value is set to first encountered value from Resource property file.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	ERU_INPUTnx_[inputline]: n is the ERU input channel selected. x can be A, B, C, D, E or F depending on hardware. 'inputline' can be PORT PIN or internal signal from other peripheral.		
Default value	ERU_INPUTnx_[inputline]		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	EruErsReference		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5.3 EruOguReference

Table 27 Specification for EruOguReference

Name	`EruOguReference`		
Description	The parameter is a reference to the ERU container in the MCU. The parameter lists down all of ERU-OGU slices (ERU output processors) available on the device. The chosen OGU channel shall be reserved for usage by ICU. The OGU channel selected in the configuration shall be unique in the ICU configuration. The value of the parameter cannot be changed across variants since the referred value, McuEruChannelOutputUnitConf, is a pre-compiler parameter. *Note: Default value is set to blank as the user has to select the appropriate reference value from MCU.*		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: McuEruChannelOutputUnitConf		
Default value	NULL		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL

(table continues...)

Table 27 (continued) Specification for EruOguReference

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

1.3.1.6 Container: GPT12Configuration

The container contains the configuration for GPTx timer if the hardware unit selected is GPT12.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Pre-Compile

1.3.1.6.1 GPT12BlockReference

Table 28 Specification for GPT12BlockReference

Name	`GPT12BlockReference`		
Description	GPT12 timer selection. Allocation is done in the MCU driver. The chosen timer cell shall be reserved by the user during the MCU driver configuration for the ICU driver. If the channel is incremental interface mode, only T2, T3 and T4 are possible. The GPT12 timer selected in the configuration shall be unique in the ICU configuration. The value of the parameter cannot be changed across variants since the referred value, McuGpt12ModuleAllocationConf, is a pre-compiler parameter. *Note: Default value is set to blank as the user has to select the appropriate reference value from MCU.*		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: McuGpt12ModuleAllocationConf		
Default value	NULL		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.2 GPT12CounterType

Table 29 Specification for GPT12CounterType

Name	`GPT12CounterType`
Description	Counting mechanism for Incremental interface mode. *Note: Default value is set to the minimum of the count inputs.*

(table continues...)

Table 29 (continued) Specification for GPT12CounterType

Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	ICU_1_COUNT_INPUT: Only Input is used for counting ICU_2_COUNT_INPUT: Both Input and Direction is used for counting		
Default value	ICU_1_COUNT_INPUT		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.3 GPT12DirPortSelection

Table 30 Specification for GPT12DirPortSelection

Name	`GPT12DirPortSelection`		
Description	Direction Input selection for the GPT12 unit. If the ICU channel is configured for Incremental interface mode, the parameter should be configured with a valid port pin. Else the parameter should be configured as NONE.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	GPT12_TnEUDx_PORTy_PINz: n is the GPT12 Timer selected x can be A or B depending on hardware, y is port number corresponding to GPT12 timer, z is port pin number corresponding to GPT12 timer NONE: No pin selected		
Default value	NONE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	GPT12BlockReference, IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.4 GPT12InputPortSelection

Table 31 Specification for GPT12InputPortSelection

Name	`GPT12InputPortSelection`		
Description	Port pin selection for input. *Note: The parameter reads from property file and depends upon channel selection. Default value is set to first encountered value from Resource property file.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	GPT12_TnINx_[inputline]: n is the GPT12 Timer selected x can be A, B, C and so on depending on hardware, y is port number corresponding to GPT12 timer, 'inputline' can be PORT PIN or internal signal from other peripheral		
Default value	GPT12_TnINx_[inputline]		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	GPT12BlockReference		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.5 GPT12TimerClearTrigger

Table 32 Specification for GPT12TimerClearTrigger

Name	`GPT12TimerClearTrigger`		
Description	GPT12 timer clear trigger in incremental interface mode. The parameter should be set to NONE for non-incremental interface mode channels.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	GPT12_T4EUD: GPT12 timer T2 is reset on a falling edge of selected T4EUD signal GPT12_T4IN: GPT12 time T3 is reset on a falling edge of selected T4IN signal NONE: GPT12 timer cannot be reset with external signal		
Default value	NONE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	IcuMeasurementMode		

(table continues...)

Table 32 (continued) Specification for GPT12TimerClearTrigger

Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.1.7 Container: GtmTimerInputConfiguration

The container contains the configuration for TIM channel if the hardware unit selected is GTM.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Pre-Compile

1.3.1.7.1 GtmTimerUsed

Table 33 Specification for GtmTimerUsed

Name	`GtmTimerUsed`		
Description	The parameter is essentially a list of GTM TIM timer cells available for usage by an ICU logical channel. Referred timer channel in MCU should have McuGtmTimChannelAllocationConf as GTM_TIM_CHANNEL_USED_BY_ICU. The GTM TIM timer selected in the configuration shall be unique in the ICU configuration. Since the dependent CCU6KernelUsed, GtmTimerUsed, GPT12BlockReference, EruErsReference , GPT12BlockReference parameter cannot be changed across variants, IcuAssignedHwUnit cannot be changed across variants. *Note: Default value is set to blank as user has to select the appropriate reference value from MCU.*		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: McuGtmTimChannelAllocationConf		
Default value	NULL		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.8 Container: IcuIncrementalInterfaceMode

The container contains the configuration parameters for Incremental interface mode.

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: Post-Build

1.3.1.8.1 IcuCounterOverflowNotification

Table 34 Specification for IcuCounterOverflowNotification

Name	`IcuCounterOverflowNotification`		
Description	The parameter is used by the ICU driver to invoke the user-defined function if incremental interface counter has overflowed. The parameter can be configured as a name or an address(numeric value) of the notification function. *Note: By default, the notification parameter will be NULL , to remove dependency from user defined functions.* *Note: The ICU driver does not validate the configured function name or address for correctness and hence the responsibility falls on the user.*		
Multiplicity	0..1	**Type**	EcucFunctionNameDef
Range	String		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE
Value configuration class	Post-Build	**Multiplicity configuration class**	Post-Build
Origin	IFX	**Scope**	LOCAL
Dependency	IcuAssignedHwUnit, IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.8.2 IcuIncrementalModeEdgeNotification

Table 35 Specification for IcuIncrementalModeEdgeNotification

Name	`IcuIncrementalModeEdgeNotification`		
Description	The parameter is used by the ICU driver to invoke the user-defined function if incremental interface mode edge is detected. The parameter can be configured as a name or an address(numeric value) of the notification function. *Note: By default, the notification parameter will be NULL , to remove dependency from user defined functions.* *Note: The ICU driver does not validate the configured function name or address for correctness and hence the responsibility falls on the user.*		
Multiplicity	0..1	**Type**	EcucFunctionNameDef
Range	String		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE

(table continues...)

Table 35 (continued) Specification for IcuIncrementalModeEdgeNotification

Value configuration class	Post-Build	**Multiplicity configuration class**	Post-Build
Origin	IFX	**Scope**	LOCAL
Dependency	IcuAssignedHwUnit, IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.9 Container: IcuTimeOutDetection

The container contains the configuration parameters for timeout detection.

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: Post-Build

1.3.1.9.1 IcuTimeOutNotification

Table 36 Specification for IcuTimeOutNotification

Name	`IcuTimeOutNotification`		
Description	The parameter is used by the ICU driver to invoke the user-defined function if the timeout time has elapsed. The parameter can be configured as a name or an address(numeric value) of the notification function. *Note: By default, the notification parameter will be NULL , to remove dependency from user defined functions.* *Note: The ICU driver does not validate the configured function name or address for correctness and hence the responsibility falls on the user.*		
Multiplicity	0..1	**Type**	EcucFunctionNameDef
Range	String		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE
Value configuration class	Post-Build	**Multiplicity configuration class**	Post-Build
Origin	IFX	**Scope**	LOCAL
Dependency	IcuAssignedHwUnit, IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.10 Container: IcuTimestampMeasurement

The container contains the configuration parameters in case the measurement mode is time stamp.

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: Post-Build

1.3.1.10.1 IcuTimestampMeasurementProperty

Table 37 Specification for IcuTimestampMeasurementProperty

Name	`IcuTimestampMeasurementProperty`		
Description	Configures the handling of the buffer in case the mode is timestamp. Implementation type of this parameter is Icu_17_TimerIp_TimestampBufferType. *Note: Default value is chosen as Linear buffer which is represented by a numerical value of 0.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	ICU_CIRCULAR_BUFFER: After reaching the end of the buffer, the driver restarts at the beginning of the buffer ICU_LINEAR_BUFFER: The buffer will be filled once		
Default value	ICU_LINEAR_BUFFER		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.10.2 IcuTimestampNotification

Table 38 Specification for IcuTimestampNotification

Name	`IcuTimestampNotification`		
Description	The parameter is used by the ICU driver to invoke the user-defined function if the requested number of time stamps are acquired. The parameter can be configured as a name or an address(numeric value) of the notification function. *Note: By default, the notification parameter will be NULL , to remove dependency from user defined functions.* *Note: The ICU driver does not validate the configured function name or address for correctness and hence the responsibility falls on the user.*		
Multiplicity	0..1	**Type**	EcucFunctionNameDef
Range	String		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE

(table continues...)

Table 38 **(continued) Specification for IcuTimestampNotification**

Value configuration class	Post-Build	**Multiplicity configuration class**	Post-Build
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	IcuTimestampApi, IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.11 Container: TimChannelFilterConfig

The container contains the filter configuration for TIM channel.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.11.1 TimChFilterCounterFreqSelect

Table 39 **Specification for TimChFilterCounterFreqSelect**

Name	`TimChFilterCounterFreqSelect`		
Description	The parameter decides the filter counter frequency for the TIM channel. The parameter modifies the FLT_CNT_FRQ of the TIM channel. *Note: Default value is set to Hardware default value.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	GTM_CONFIGURABLE_CLOCK_0: Configurable clock 0 clocks the filter counter GTM_CONFIGURABLE_CLOCK_1: Configurable clock 1 clocks the filter counter GTM_CONFIGURABLE_CLOCK_6: Configurable clock 6 clocks the filter counter GTM_CONFIGURABLE_CLOCK_7: Configurable clock 7 clocks the filter counter		
Default value	GTM_CONFIGURABLE_CLOCK_0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.11.2 TimChFilterModeForFallingEdge

Table 40 **Specification for TimChFilterModeForFallingEdge**

Name	`TimChFilterModeForFallingEdge`

(table continues...)

Table 40 (continued) Specification for TimChFilterModeForFallingEdge

Description	The parameter decides the filter mode for falling edge of the TIM channel input. *Note: Default value is set to Hardware default value.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DEGLITCH_WITH_HOLD_COUNTER: Each edge of an input signal will be filtered with an individual de-glitch threshold filter value with filter counter value unchanged after inactive edge until active edge DEGLITCH_WITH_UPDOWN_COUNTER: Each edge of an input signal will be filtered with an individual de-glitch threshold filter value with filter counter decrementing after inactive edge until active edge. IMMEDIATE_EDGE_PROPAGATION_MODE: After detection of an edge the new signal level is propagated and the new signal level remains unchanged until the configured acceptance time expires		
Default value	IMMEDIATE_EDGE_PROPAGATION_MODE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.11.3 TimChFilterModeForRisingEdge

Table 41 Specification for TimChFilterModeForRisingEdge

Name	`TimChFilterModeForRisingEdge`		
Description	The parameter decides the filter mode for rising edge of the TIM channel input. *Note: Default value is set to Hardware default value.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	DEGLITCH_WITH_HOLD_COUNTER: Each edge of an input signal will be filtered with an individual de-glitch threshold filter value with filter counter value unchanged after inactive edge until active edge DEGLITCH_WITH_UPDOWN_COUNTER: Each edge of an input signal will be filtered with an individual de-glitch threshold filter value with filter counter decrementing after inactive edge until active edge. IMMEDIATE_EDGE_PROPAGATION_MODE: After detection of an edge the new signal level is propagated and the new signal level remains unchanged until the configured acceptance time expires		
Default value	IMMEDIATE_EDGE_PROPAGATION_MODE		

(table continues...)

Table 41 (continued) Specification for TimChFilterModeForRisingEdge

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

1.3.1.11.4 TimChFilterTimeForFallingEdge

Table 42 Specification for TimChFilterTimeForFallingEdge

Name	`TimChFilterTimeForFallingEdge`		
Description	The parameter specifies the filter time for falling edge of the TIM channel input. *Note: Default value is set to Hardware default value.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 16777215		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.11.5 TimChFilterTimeForRisingEdge

Table 43 Specification for TimChFilterTimeForRisingEdge

Name	`TimChFilterTimeForRisingEdge`		
Description	The parameter specifies the filter time for rising edge of the TIM channel input. *Note: Default value is set to Hardware default value.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 16777215		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-

(table continues...)

Table 43 (continued) Specification for TimChFilterTimeForRisingEdge

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

1.3.1.11.6 TimChannelFilterEnable

Table 44 Specification for TimChannelFilterEnable

Name	`TimChannelFilterEnable`		
Description	The parameter enables filter for the channel. Sets FLT_EN for the TIM channel. *Note: Default value is set to Hardware default value.*		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.12 Container: TimChannelGeneral

The container contains the TIM channel specific configuration.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.12.1 OverflowISRThreshold

Table 45 Specification for OverflowISRThreshold

Name	`OverflowISRThreshold`

(table continues...)

Table 45 **(continued) Specification for OverflowISRThreshold**

Description	The threshold denotes the maximum latency between the actual TIM counter overflow interrupt and execution of ICU ISR. User shall configure the threshold to contain the maximum latency and the threshold shall be lesser than the actual measured time. The threshold will be in the ticks of the CMU_CLK selected for that TIM channel. The threshold should consider interrupt latency, other high priority interrupts and the latency to reach the ICU ISR. The threshold is applicable only for Signal measurement, HIGH TIME and LOW TIME. *Note: Default value is set to minimum value(0).*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 16777215		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.12.2 TimChannelClockSelect

Table 46 **Specification for TimChannelClockSelect**

Name	`TimChannelClockSelect`		
Description	The parameter decides the clock source for TIM channel. *Note: Default value is set to Hardware default value.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	GTM_CONFIGURABLE_CLOCK_0: Configurable clock 0 will be supplied to the TIM channel GTM_CONFIGURABLE_CLOCK_1: Configurable clock 1 will be supplied to the TIM channel GTM_CONFIGURABLE_CLOCK_2: Configurable clock 2 will be supplied to the TIM channel GTM_CONFIGURABLE_CLOCK_3: Configurable clock 3 will be supplied to the TIM channel GTM_CONFIGURABLE_CLOCK_4: Configurable clock 4 will be supplied to the TIM channel GTM_CONFIGURABLE_CLOCK_5: Configurable clock 5 will be supplied to the TIM channel GTM_CONFIGURABLE_CLOCK_6: Configurable clock 6 will be supplied to the TIM channel GTM_CONFIGURABLE_CLOCK_7: Configurable clock 7 will be supplied to the TIM channel		
Default value	GTM_CONFIGURABLE_CLOCK_0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-

(table continues...)

Table 46 (continued) Specification for TimChannelClockSelect

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

1.3.1.12.3 TimChannelGpr0InputSelect

Table 47 Specification for TimChannelGpr0InputSelect

Name	`TimChannelGpr0InputSelect`		
Description	The parameter decides the reference timer for GPR0 register of TIM channel. The timer selected as reference should be enabled in MCU configurations. The GPR0 input can be selected only if the channel mode is time stamp. An error will be issued if the selected TBU channel is not enabled in MCU configuration. *Note: Default value is set to Hardware default value.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	TIMEBASE_TBU_TS0: TBU_TS0 will be captured in GPR0 TIMEBASE_TBU_TS1: TBU_TS1 will be captured in GPR0 TIMEBASE_TBU_TS2: TBU_TS2 will be captured in GPR0		
Default value	TIMEBASE_TBU_TS0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	GtmTbuChannelEnable		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.12.4 TimChannelInputSelect

Table 48 Specification for TimChannelInputSelect

Name	`TimChannelInputSelect`		
Description	The parameter decides the Input for the TIM channel. *Note: Default value is set to Hardware default value.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef

(table continues...)

Table 48 (continued) Specification for TimChannelInputSelect

Range	INPUT_OF_CURRENT_TIM_CHANNEL: Input to the current TIM channel will be the input assigned to current channel. INPUT_OF_PREVIOUS_TIM_CHANNEL: Input to the current TIM channel will be the input assigned to the previous channel.		
Default value	INPUT_OF_CURRENT_TIM_CHANNEL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.12.5 TimInterruptMode

Table 49 Specification for TimInterruptMode

Name	`TimInterruptMode`		
Description	The parameter decides the interrupt mode to be used. If Level mode for interrupt is used with IcuTimeoutFeature enabled, there is a possibility that all further interrupts are not triggered. Recommendation is to use Pulse Notify mode for interrupts. *Note: Default value is set to Hardware default value.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	GTM_INTERRUPT_LEVEL_MODE: Selects level mode for interrupt GTM_INTERRUPT_PULSE_NOTIFY_MODE: Selects pulse notify mode for interrupt GTM_INTERRUPT_SINGLE_PULSE_MODE: Selects single pulse mode for interrupt		
Default value	GTM_INTERRUPT_LEVEL_MODE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	IcuTimeoutFeature		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.13 Container: Icu

Configuration of ICU (Input Capture Unit) module.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.14 Container: IcuChannel

Configuration of an individual ICU channel.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Post-Build

1.3.1.14.1 IcuAssignedHwUnit

Table 50 Specification for IcuAssignedHwUnit

Name	`IcuAssignedHwUnit`		
Description	The parameter chooses the capture engine required by an ICU channel. *Note: Default value is chosen as GTM which is represented by a numerical value of 0.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	CCU6: Selects CCU6 as hardware unit to realize ICU channel ERU: Selects ERU as hardware unit to realize ICU channel GPT12: Selects GPT12 as hardware unit to realize ICU channel GTM: Selects GTM-TIM as hardware unit to realize ICU channel		
Default value	GTM		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.14.2 IcuChannelEcucPartitionRef

Table 51 Specification for IcuChannelEcucPartitionRef

Name	`IcuChannelEcucPartitionRef`		
Description	Maps an ICU channel to zero or multiple ECUC partitions to limit the access to this channel. The ECUC partitions referenced are a subset of the ECUC partitions where the ICU driver is mapped to. *Note: Parameter support is added only for AUTOSAR schema compliance, this parameter is not used in code generation logic, hence this parameter is made editable false.*		
Multiplicity	0..*	**Type**	EcucReferenceDef

(table continues...)

Table 51 (continued) Specification for IcuChannelEcucPartitionRef

Range	Reference to Node: EcucPartition		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE
Value configuration class	Pre-Compile	**Multiplicity configuration class**	Pre-Compile
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar version 4.4.0.		

1.3.1.14.3 IcuChannelId

Table 52 Specification for IcuChannelId

Name	`IcuChannelId`		
Description	Logical channel identifier of the ICU channel. The parameters value will be assigned to the symbolic name derived from the IcuChannel container short name. The value of IcuChannelId should be unique in a configuration set. *Note: Default value is set to minimum value.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - (Total number of channels - 1)		
Default value	0		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	ECU
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.14.4 IcuDefaultStartEdge

Table 53 Specification for IcuDefaultStartEdge

Name	`IcuDefaultStartEdge`

(table continues...)

Table 53 **(continued) Specification for IcuDefaultStartEdge**

Description	Configures the default activation edge which will be used for the ICU channel. For Signal measurement, following conventions are to be adhered: PERIOD - Denotes the start of period. DUTY - Denotes the start of Period and Active time. HIGH TIME and LOW TIME, the parameter is irrelevant and will be un-editable. If BOTH EDGES is configured for DUTY CYCLE measurement, first edge seen after first call of Icu_17_TimerIp_StartSignalMeasurement is considered as default start edge for the entire Init - DeInit cycle. The parameter is unused and hence will be non-editable for incremental interface mode channel. *Note: Default value is chosen as Rising Edge which is represented by a numerical value of 0(minimum).*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	ICU_BOTH_EDGES: Both edges are used ICU_FALLING_EDGE: Falling edge is the used ICU_RISING_EDGE: Rising edge is the used		
Default value	ICU_RISING_EDGE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	IcuMeasurementMode, IcuSignalMeasurementProperty		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.14.5 IcuMeasurementMode

Table 54 **Specification for IcuMeasurementMode**

Name	`IcuMeasurementMode`

(table continues...)

Table 54 (continued) Specification for IcuMeasurementMode

Description	Configures the measurement mode of the ICU channel. ICU_MODE_SIGNAL_EDGE_DETECT: The channel is used for detecting the edges which are configured by the call of the service Icu_17_TimerIp_SetActivationCondition(). The following API services support this mode: - Icu_17_TimerIp_EnableEdgeDetection() - Icu_17_TimerIp_DisableEdgeDetection() - Icu_17_TimerIp_EnableMultiEdgeDetection() - Icu_17_TimerIp_EnableNotification() - Icu_17_TimerIp_DisableNotification() - Icu_17_TimerIp_GetInputState() - Icu_17_TimerIp_SetTimeoutValue() Edge detection mode can be configured if IcuEdgeDetectApi is switched on. *Note: Default value is chosen as Edge detection which is represented by a numerical value of 0.* ICU_MODE_SIGNAL_MEASUREMENT: The channel is used to measure different times between various configurable edges. The configuration of the period-start edges is done by configuration and cannot be changed during runtime. The following API services support this mode: - Icu_17_TimerIp_StartSignalMeasurement() - Icu_17_TimerIp_StopSignalMeasurement() - Icu_17_TimerIp_GetTimeElapsed() - Icu_17_TimerIp_GetDutyCycleValues() - Icu_17_TimerIp_GetInputState() Signal measurement mode can be configured if at least one of the following switches are set to TRUE: - IcuGetDutyCycleValuesApi - IcuGetTimeElapsedApi ICU_MODE_TIMESTAMP: The channel is used to capture timer values on the edges which are configured by the call of the service Icu_17_TimerIp_SetActivationCondition(). The following API services support this mode: - Icu_17_TimerIp_StartTimestamp() - Icu_17_TimerIp_StopTimestamp() - Icu_17_TimerIp_GetTimestampIndex() - Icu_17_TimerIp_EnableNotification() - Icu_17_TimerIp_DisableNotification() Time stamping mode can be configured if IcuTimeStampApi is switched on. ICU_MODE_EDGE_COUNTER: The channel is used to count the edges which are configured by the call of the service Icu_17_TimerIp_SetActivationCondition(). The following API services support this mode: - Icu_17_TimerIp_EnableEdgeCount() - Icu_17_TimerIp_DisableEdgeCount() - Icu_17_TimerIp_GetEdgeNumbers()

(table continues...)

Table 54 **(continued) Specification for IcuMeasurementMode**

	- Icu_17_TimerIp_ResetEdgeCount() -Icu_17_TimerIp_SetTimeoutValue() Edge counting mode can be configured if IcuEdgeVountApi is switched on. ICU_MODE_INCREMENTAL_INTERFACE: The channel is configured to count the encoder edges(using the incremental interface mode of GPT12 peripheral). The following API services support this mode: - Icu_17_TimerIp_StartIncInterface() - Icu_17_TimerIp_StopIncInterface() - Icu_17_TimerIp_CalibratePos() - Icu_17_TimerIp_ReadEncCount() - Icu_17_TimerIp_EncCountDirType() - Icu_17_TimerIp_EnableNotification() - Icu_17_TimerIp_DisableNotification() Incremental interface mode can be configured if IcuIncrementalInterfaceApi is switched on.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	ICU_MODE_EDGE_COUNTER: The channel is configured to count the edges which are set by the call of service Icu_17_TimerIp_SetActivationCondition() ICU_MODE_INCREMENTAL_INTERFACE: The channel is configured to count the encoder edges using the incremental interface mode of GPT12 peripheral ICU_MODE_SIGNAL_EDGE_DETECT: The channel is configured for detecting the edges which are set by the call of service Icu_17_TimerIp_SetActivationCondition() ICU_MODE_SIGNAL_MEASUREMENT: The channel is configured to measure signal properties. The configuration of the period start edges is done during configuration and cannot be changed during runtime ICU_MODE_TIMESTAMP: The channel is configured to capture timer values on the edges which are set by the call of service Icu_17_TimerIp_SetActivationCondition()		
Default value	ICU_MODE_SIGNAL_EDGE_DETECT		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.14.6 IcuTimeoutFeature

Table 55 **Specification for IcuTimeoutFeature**

Name	`IcuTimeoutFeature`

(table continues...)

Table 55 (continued) Specification for IcuTimeoutFeature

Description	Configuration parameter to configure the timeout functionality of ICU channel. Timeout feature shall be set to TIMEOUT_DISABLED if the IcuMeasurementMode is neither "ICU_MODE_EDGE_DETECT" nor "ICU_MODE_EDGE_COUNT". Timeout feature shall be set to TIMEOUT_DISABLED if IcuAssignedHwUnit is not "GTM". Timeout feature shall be set to TIMEOUT_DISABLED if IcuTimeoutDetectionApi is not set. Default value is set as hardware default value.		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	TIMEOUT_DISABLED: Timeout feature is disabled TIMEOUT_MIXED: Both ICU channel measurement mode and timeout feature are enabled TIMEOUT_ONLY: Only timeout feature is enabled. ICU mode selected is not applicable. DET/Safety Error is reported if any mode specific API's(except enable/disable channel) are invoked		
Default value	TIMEOUT_DISABLED		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	IcuTimeoutDetectionApi, IcuAssignedHwUnit, IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.14.7 IcuWakeupCapability

Table 56 Specification for IcuWakeupCapability

Name	`IcuWakeupCapability`		
Description	Information about the wakeup-capability of the ICU channel. TRUE: Channel is wakeup capable. FALSE: Channel is not wakeup capable. Wakeup capability value can be TRUE only if the channel is an edge detect channel. *Note: By default value is set to False, to remove the dependency from ECUM.*		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-

(table continues...)

Table 56 (continued) Specification for IcuWakeupCapability

Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.15 Container: IcuConfigSet

The container contains the configuration parameters and sub containers of the ICU driver.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.15.1 IcuMaxChannel

Table 57 Specification for IcuMaxChannel

Name	`IcuMaxChannel`		
Description	The parameter contains the number of channels configured. The parameters value will be gathered by tools during the configuration stage. calculationFormula = Number of configured ICU channels Implementation Type: Icu_17_TimerIp_ChannelType *Note: The parameter is non-editable as it is not used in any code generation. The value shall not be used for any references.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 65535		
Default value	0		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	IcuChannel		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.16 Container: IcuGeneral

Configuration of general ICU parameters.

Note: By default all the error reporting (Development, Safety and Multi-core) are enable, to ensure proper driver functionality.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.16.1 IcuDevErrorDetect

Table 58 Specification for IcuDevErrorDetect

Name	`IcuDevErrorDetect`		
Description	Enables or disables the Default Error Tracer (DET) detection and reporting. true: enabled (ON). false: disabled (OFF).		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.16.2 IcuEcucPartitionRef

Table 59 Specification for IcuEcucPartitionRef

Name	`IcuEcucPartitionRef`		
Description	Maps the ICU driver to zero or multiple ECUC partitions to make the driver API available in the according partition. *Note: Parameter support is added only for AUTOSAR schema compliance, this parameter is not used in code generation logic, hence this parameter is made editable false.*		
Multiplicity	0..*	**Type**	EcucReferenceDef
Range	Reference to Node: EcucPartition		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE
Value configuration class	Pre-Compile	**Multiplicity configuration class**	Pre-Compile
Origin	AUTOSAR_ECUC	**Scope**	LOCAL

(table continues...)

Table 59 (continued) Specification for IcuEcucPartitionRef

Dependency	-
Autosar Version	Applicable for Autosar version 4.4.0.

1.3.1.16.3 IcuIndex

Table 60 Specification for IcuIndex

Name	`IcuIndex`		
Description	Specifies the instance Id of the ICU driver. If only one instance is present, the value of the parameter should be 0. *Note: Default value is set to the minimum value.*		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	0		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar version 4.2.2.		

1.3.1.16.4 IcuInitDeInitApiMode

Table 61 Specification for IcuInitDeInitApiMode

Name	`IcuInitDeInitApiMode`		
Description	Pre-processor switch to enable or disable protected register access in Icu_17_TimerIp_Init and Icu_17_TimerIp_DeInit APIs. If IcuRuntimeApiMode is set to ICU_MCAL_SUPERVISOR, IcuInitDeInitApiMode has to be set to ICU_MCAL_SUPERVISOR. *Note: By default access level of all the runtime APIs set to Supervisor so that there is no dependency on the OS functions to write into the access protected SFR.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	ICU_MCAL_SUPERVISOR: ICU init APIs will run in supervisor mode ICU_MCAL_USER1: ICU init APIs will run in user1 mode		
Default value	ICU_MCAL_SUPERVISOR		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-

(table continues...)

Table 61 (continued) Specification for IcuInitDeInitApiMode

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

1.3.1.16.5 IcuKernelEcucPartitionRef

Table 62 Specification for IcuKernelEcucPartitionRef

Name	`IcuKernelEcucPartitionRef`		
Description	Maps the ICU kernel to zero or one ECUC partitions to assign the driver kernel to a certain core. The ECUC partition referenced is a subset of the ECUC partitions where the ICU driver is mapped to. *Note: Parameter support is added only for AUTOSAR schema compliance, this parameter is not used in code generation logic, hence this parameter is made editable false.*		
Multiplicity	0..1	**Type**	EcucReferenceDef
Range	Reference to Node: EcucPartition		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE
Value configuration class	Pre-Compile	**Multiplicity configuration class**	Pre-Compile
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar version 4.4.0.		

1.3.1.16.6 IcuMultiCoreErrorDetect

Table 63 Specification for IcuMultiCoreErrorDetect

Name	`IcuMultiCoreErrorDetect`		
Description	The parameter enables or disables the multi core related default error tracer (DET) detection and reporting. It is applicable only when DETs are enabled. IcuMultiCoreErrorDetect shall be set to false if CPU selected is single core.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		

(table continues...)

Table 63 **(continued) Specification for IcuMultiCoreErrorDetect**

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

1.3.1.16.7 IcuReportWakeupSource

Table 64 **Specification for IcuReportWakeupSource**

Name	`IcuReportWakeupSource`		
Description	Switch for enabling wakeup source reporting. true: Report wakeup source. false: Do not report wakeup source. If IcuReportWakeupSource is set then the configuration should have at least one wakeup capable channel. The parameter should be checked if any of the channels is configured as wakeup capable.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.16.8 IcuRunTimeErrorDetect

Table 65 **Specification for IcuRunTimeErrorDetect**

Name	`IcuRunTimeErrorDetect`

(table continues...)

Table 65 (continued) Specification for IcuRunTimeErrorDetect

Description	Enables or disables the Runtime Error detection and reporting. If safety is enabled(by setting the parameter IcuSafetyEnable to true), IcuRunTimeErrorDetect shall be set to true. IcuRunTimeErrorDetect shall be disabled even if safety is enabled, if IcuTimestampApi is disabled since runtime error is reported only in Icu_17_TimerIp_StopTimestamp API true: enabled (ON). false: disabled (OFF).		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	IcuTimestampApi, IcuSafetyEnable		
Autosar Version	Applicable for Autosar version 4.4.0.		

1.3.1.16.9 IcuRuntimeApiMode

Table 66 Specification for IcuRuntimeApiMode

Name	`IcuRuntimeApiMode`		
Description	Pre-processor switch to enable or disable protected register access in runtime APIs. *Note: By default access level of all the runtime APIs set to Supervisor so that there is no dependency on the OS functions to write into the access protected SFR.*		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	ICU_MCAL_SUPERVISOR: ICU runtime APIs will run in supervisor mode ICU_MCAL_USER1: ICU runtime APIs will run in user1 mode		
Default value	ICU_MCAL_SUPERVISOR		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL

(table continues...)

Table 66 (continued) Specification for IcuRuntimeApiMode

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

1.3.1.16.10 IcuSafetyEnable

Table 67 Specification for IcuSafetyEnable

Name	`IcuSafetyEnable`		
Description	Pre-processor switch for enabling the safety features of ICU driver.		
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.		

1.3.1.17 Container: IcuOptionalApis

The container contains all configuration switches for configuring optional API services of the ICU driver.

Note: All optional APIs set to False except initCheck , to minimize executable code size.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.17.1 IcuDeInitApi

Table 68 Specification for IcuDeInitApi

Name	`IcuDeInitApi`		
Description	Adds / removes the service Icu_17_TimerIp_DeInit() from the code. TRUE: Icu_17_TimerIp_DeInit() can be used. FALSE: Icu_17_TimerIp_DeInit() cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef

(table continues...)

Table 68 (continued) Specification for IcuDeInitApi

Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17.2 IcuDisableWakeupApi

Table 69 Specification for IcuDisableWakeupApi

Name	`IcuDisableWakeupApi`		
Description	Adds / removes the service Icu_17_TimerIp_DisableWakeup() from the code. IcuDisableWakeupApi may be set to true only if IcuEnableWakeupApi is true. TRUE: Icu_17_TimerIp_DisableWakeup() can be used. FALSE: Icu_17_TimerIp_DisableWakeup() cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17.3 IcuEdgeCountApi

Table 70 Specification for IcuEdgeCountApi

Name	`IcuEdgeCountApi`

(table continues...)

Table 70 **(continued) Specification for IcuEdgeCountApi**

Description	Adds / removes all services related to the edge counting functionality, as listed below, from the code: Icu_17_TimerIp_ResetEdgeCount(), Icu_17_TimerIp_EnableEdgeCount(), Icu_17_TimerIp_DisableEdgeCount(), Icu_17_TimerIp_GetEdgeNumbers(). IcuEdgeCountApi shall be set to true if there is at least one channel in edge count mode. IcuEdgeCountApi shall be set to false if there are no GPT12 and GTM channels. TRUE: The services listed above can be used. FALSE: The services listed above cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	IcuAssignedHwUnit, IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17.4 IcuEdgeDetectApi

Table 71 **Specification for IcuEdgeDetectApi**

Name	`IcuEdgeDetectApi`		
Description	Adds / removes the services related to the edge detection functionality from the code: Icu_17_TimerIp_EnableEdgeDetection(), Icu_17_TimerIp_DisableEdgeDetection() and Icu_17_TimerIp_EnableMultiEdgeDetection IcuEdgeDetectApi shall be set to true if there is at least one channel in edge detect mode. TRUE: These services can be used. FALSE: These services cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-

(table continues...)

Table 71 **(continued) Specification for IcuEdgeDetectApi**

Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17.5 IcuEnableWakeupApi

Table 72 **Specification for IcuEnableWakeupApi**

Name	`IcuEnableWakeupApi`		
Description	Adds / removes the service Icu_17_TimerIp_EnableWakeup() from the code. IcuEnableWakeupApi shall be set to true if there is at least one channel's wake-up capability is true. TRUE: Icu_17_TimerIp_EnableWakeup() can be used. FALSE: Icu_17_TimerIp_EnableWakeup() cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17.6 IcuGetDutyCycleValuesApi

Table 73 **Specification for IcuGetDutyCycleValuesApi**

Name	`IcuGetDutyCycleValuesApi`

(table continues...)

Table 73 (continued) Specification for IcuGetDutyCycleValuesApi

Description	Adds / removes the service Icu_17_TimerIp_GetDutyCycleValues() from the code. IcuGetDutyCycleValuesApi shall be set to true if there is at least one channel measuring duty cycle. IcuGetDutyCycleValuesApi may be set to true only if IcuSignalMeasurementApi is true. TRUE: Icu_17_TimerIp_GetDutyCycleValues() can be used. FALSE: Icu_17_TimerIp_GetDutyCycleValues() cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	IcuSignalMeasurementApi		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17.7 IcuGetInputStateApi

Table 74 Specification for IcuGetInputStateApi

Name	`IcuGetInputStateApi`		
Description	Adds / removes the service Icu_17_TimerIp_GetInputState() from the code. TRUE: Icu_17_TimerIp_GetInputState() can be used. FALSE: Icu_17_TimerIp_GetInputState() cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		

(table continues...)

Table 74 (continued) Specification for IcuGetInputStateApi

Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.1.17.8 IcuGetTimeElapsedApi

Table 75 Specification for IcuGetTimeElapsedApi

Name	`IcuGetTimeElapsedApi`		
Description	Adds / removes the service Icu_17_TimerIp_GetTimeElapsed() from the code. IcuGetTimeElapsedApi shall be set to true if there is at least one channel in signal measurement mode measuring a non-duty cycle value. IcuGetTimeElapsedApi may be set to true only if IcuSignalMeasurementApi is true. TRUE: Icu_17_TimerIp_GetTimeElapsed() can be used. FALSE: Icu_17_TimerIp_GetTimeElapsed() cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	IcuSignalMeasurementApi		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17.9 IcuGetVersionInfoApi

Table 76 Specification for IcuGetVersionInfoApi

Name	`IcuGetVersionInfoApi`		
Description	Adds / removes the service Icu_17_TimerIp_GetVersionInfo() from the code. TRUE: Icu_17_TimerIp_GetVersionInfo() can be used. FALSE: Icu_17_TimerIp_GetVersionInfo() cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		

(table continues...)

Table 76 (continued) Specification for IcuGetVersionInfoApi

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

1.3.1.17.10 IcuIncrementalInterfaceApi

Table 77 Specification for IcuIncrementalInterfaceApi

Name	`IcuIncrementalInterfaceApi`		
Description	Adds / removes all services related to the incremental interface functionality, as listed below, from the code: Icu_17_TimerIp_StartIncInterface(), Icu_17_TimerIp_StopIncInterface(), Icu_17_TimerIp_CalibratePos(), Icu_17_TimerIp_ReadEncCount(), Icu_17_TimerIp_ReadEncCountDir(). IcuIncrementalInterfaceApi may be set to true if there is at least one channel in incremental interface mode. IcuIncrementalInterfaceApi shall be set to false if there are no GPT12 channels. TRUE: The services listed above can be used. FALSE: The services listed above cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	IcuMeasurementMode, IcuAssignedHwUnit		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17.11 IcuInitCheckApi

Table 78 Specification for IcuInitCheckApi

Name	`IcuInitCheckApi`		
Description	Pre-processor switch for enabling/disabling the safety feature Icu_17_TimerIp_InitCheck() which verifies the initialization done by ICU driver. If the parameter is set to TRUE, the Icu_17_TimerIp_InitCheck() API can be used to verify the initialization done by ICU driver . If this parameter is set to FALSE, Icu_17_TimerIp_InitCheck() API cannot be used.		
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.		

1.3.1.17.12 IcuSetModeApi

Table 79 Specification for IcuSetModeApi

Name	`IcuSetModeApi`		
Description	Adds / removes the service Icu_17_TimerIp_SetMode() from the code. TRUE: Icu_17_TimerIp_SetMode() can be used. FALSE: Icu_17_TimerIp_SetMode() cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL

(table continues...)

Table 79 (continued) Specification for IcuSetModeApi

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

1.3.1.17.13 IcuSignalMeasurementApi

Table 80 Specification for IcuSignalMeasurementApi

Name	`IcuSignalMeasurementApi`		
Description	Adds / removes the services Icu_17_TimerIp_StartSignalMeasurement() and Icu_17_TimerIp_StopSignalMeasurement() from the code. IcuSignalMeasurementApi shall be set to true if there is at least one channel in signal measurement mode. IcuSignalMeasurementApi shall be set to false if there are no GTM and CCU6 channels. TRUE: Icu_17_TimerIp_StartSignalMeasurement() and Icu_17_TimerIp_StopSignalMeasurement() can be used. FALSE: Icu_17_TimerIp_StartSignalMeasurement() and Icu_17_TimerIp_StopSignalMeasurement() cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	IcuAssignedHwUnit, IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17.14 IcuTimeoutDetectionApi

Table 81 Specification for IcuTimeoutDetectionApi

Name	`IcuTimeoutDetectionApi`

(table continues...)

Table 81 (continued) Specification for IcuTimeoutDetectionApi

Description	Adds / removes all services related to the timeout detection functionality, as listed below, from the code: Icu_17_TimerIp_SetTimeoutValue IcuTimeoutDetectionApi shall be set to false if there are no GTM channels. TRUE: The services listed above can be used. FALSE: The services listed above cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
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.		

1.3.1.17.15 IcuTimestampApi

Table 82 Specification for IcuTimestampApi

Name	`IcuTimestampApi`		
Description	Adds / removes all services related to the time stamping functionality, as listed below, from the code: Icu_17_TimerIp_StartTimestamp(), Icu_17_TimerIp_StopTimestamp(), Icu_17_TimerIp_GetTimestampIndex(). IcuTimestampApi shall be set to true if there is at least one channel in time stamping mode. IcuTimestampApi shall be set to false if there are no GTM channels and no CCU6 channels. TRUE: The services listed above can be used. FALSE: The services listed above cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-

(table continues...)

Table 82 (continued) Specification for IcuTimestampApi

Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	IcuAssignedHwUnit, IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.17.16 IcuWakeupFunctionalityApi

Table 83 Specification for IcuWakeupFunctionalityApi

Name	`IcuWakeupFunctionalityApi`		
Description	Adds / removes the service Icu_17_TimerIp_CheckWakeup() from the code. IcuWakeupFunctionalityApi shall be set to true if at least one channel's wake-up capability is set to true. IcuWakeupFunctionalityApi shall be set to false if there are no channels with wake-up capability set to true. TRUE: Icu_17_TimerIp_CheckWakeup() can be used. FALSE: Icu_17_TimerIp_CheckWakeup() cannot be used.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.18 Container: IcuSignalEdgeDetection

The container contains the configuration (parameters) in case the measurement mode is edge detection.

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: Post-Build

1.3.1.18.1 IcuSignalNotification

Table 84 Specification for IcuSignalNotification

Name	`IcuSignalNotification`		
Description	The parameter is used by the ICU driver to invoke the user-defined function if the configured edge is detected. The parameter can be configured as a name or an address(numeric value) of the notification function. *Note: By default, the notification parameter will be NULL , to remove dependency from user defined functions.* *Note: The ICU driver does not validate the configured function name or address for correctness and hence the responsibility falls on the user.*		
Multiplicity	0..1	**Type**	EcucFunctionNameDef
Range	String		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE
Value configuration class	Post-Build	**Multiplicity configuration class**	Post-Build
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.19 Container: IcuSignalMeasurement

The container contains the configuration (parameters) in case the measurement mode is signal measurement.

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: Post-Build

1.3.1.19.1 IcuSignalMeasurementProperty

Table 85 Specification for IcuSignalMeasurementProperty

Name	`IcuSignalMeasurementProperty`
Description	Configures the property that could be measured in case the mode is signal measurement. The signal measurement property cannot be changed during runtime. Duty cycle can only be selected if IcuGetDutyCyclesApi is available. High time/low time/period can be selected only if IcuGetTimeElapsedApi is available. For period measurement, IcuDefaultStartEdge should not be Both edges. Implementation type: Icu_17_TimerIp_SignalMeasurementPropertyType. *Note: Default value is chosen as Low time which is represented by a numerical value of 0.*

(table continues...)

Table 85 (continued) Specification for IcuSignalMeasurementProperty

Multiplicity	1..1	**Type**	EcucEnumerationParamDef
Range	ICU_DUTY_CYCLE: The channel is configured to read values which are needed for calculating the duty cycle (coherent active and period time) ICU_HIGH_TIME: The channel is configured for reading the elapsed signal high time ICU_LOW_TIME: The channel is configured for reading the elapsed signal low time ICU_PERIOD_TIME: The channel is configured for reading the elapsed signal period time		
Default value	ICU_LOW_TIME		
Post-build variant value	TRUE	**Post-build variant multiplicity**	-
Value configuration class	Post-Build	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	IcuGetTimeElapsedApi, IcuGetDutyCycleValuesApi, IcuMeasurementMode		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.20 Container: IcuWakeup

The container contains the configuration (parameters) needed to configure a wake-up capable channel.

Post-Build Variant Multiplicity: TRUE

Multiplicity Configuration Class: Post-Build

1.3.1.20.1 IcuChannelWakeupInfo

Table 86 Specification for IcuChannelWakeupInfo

Name	`IcuChannelWakeupInfo`		
Description	If the wakeup-capability is true the wakeup source referenced is transmitted to the ECU State Manager (EcuM). IcuChannelWakeupInfo is editable only if channel wakeup capability is true and IcuReportWakeupSource is true. Implementation type: reference to EcuM_WakeupSourceType. *Note: By default wake-up info is set to NULL , to remove dependency from EcuM wake-up configurations.*		
Multiplicity	0..1	**Type**	EcucSymbolicNameReferenceDef
Range	Reference to Node: EcuMWakeupSource		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE

(table continues...)

Table 86 (continued) Specification for IcuChannelWakeupInfo

Value configuration class	Post-Build	**Multiplicity configuration class**	Post-Build
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.2 Functions - Type definitions

1.3.2.1 Icu_17_TimerIp_NotifiPtrType

Table 87 Specification for Icu_17_TimerIp_NotifiPtrType

Syntax	`Icu_17_TimerIp_NotifiPtrType`
Type	Pointer to a function of type void Function_Name (void)
File	`Icu_17_TimerIp.h`
Description	Channel notification function pointer (notification function applicable in case of channel configured for edge detect or time stamp mode).
Source	IFX
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.2.2 Icu_17_TimerIp_ModeType

Table 88 Specification for Icu_17_TimerIp_ModeType

Syntax	`Icu_17_TimerIp_ModeType`	
Type	Enumeration	
File	`Icu_17_TimerIp.h`	
Range	0 - ICU_17_TIMERIP_MODE_NORMAL	Normal operation, all used interrupts are enabled according to the notification requests.
	1 - ICU_17_TIMERIP_MODE_SLEEP	Reduced power operation. In sleep mode only those notifications are available which are configured as wakeup capable.
Description	Allow enabling / disabling of all interrupts which are not required for the ECU wakeup.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.3 Icu_17_TimerIp_ChannelType

Table 89 Specification for Icu_17_TimerIp_ChannelType

Syntax	`Icu_17_TimerIp_ChannelType`	
Type	uint8	
File	`Icu_17_TimerIp.h`	
Range	0-82	The range includes the total number of TIM channels, CCU6 comparators, ERU channels and GPT12 timers. The maximum number of channels may vary depending on the device variant. 82 is considering the superset device variant. *Note: This is the maximum possible valid range. The actual valid range is 0-(Total number of ICU channels configured - 1)*
Description	Numeric identifier of an ICU channel.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.4 Icu_17_TimerIp_EncCountDirType

Table 90 Specification for Icu_17_TimerIp_EncCountDirType

Syntax	`Icu_17_TimerIp_EncCountDirType`	
Type	Enumeration	
File	`Icu_17_TimerIp.h`	
Range	1 - ICU_17_TIMERIP_ENC_COUNT_DOWN	Encoder counting down
	0 - ICU_17_TIMERIP_ENC_COUNT_UP	Encoder counting up
Description	Encoder counting direction for an incremental interface mode ICU channel.	
Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.5 Icu_17_TimerIp_InputStateType

Table 91 Specification for Icu_17_TimerIp_InputStateType

Syntax	`Icu_17_TimerIp_InputStateType`	
Type	Enumeration	
File	`Icu_17_TimerIp.h`	
Range	1 - ICU_17_TIMERIP_ACTIVE	An activation edge has been detected

(table continues...)

Table 91 (continued) Specification for Icu_17_TimerIp_InputStateType

	0 - ICU_17_TIMERIP_IDLE	No activation edge has been detected since the last call of Icu_17_TimerIp_GetInputState() or Icu_17_TimerIp_Init()
Description	Input state of an ICU channel.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.6 Icu_17_TimerIp_ConfigType

Table 92 Specification for Icu_17_TimerIp_ConfigType

Syntax	`Icu_17_TimerIp_ConfigType`	
Type	Structure	
File	`Icu_17_TimerIp.h`	
Range	--	The elements of the data structure are specific to the micro-controller
Description	The data type contains initialization data.	
Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.7 Icu_17_TimerIp_ActivationType

Table 93 Specification for Icu_17_TimerIp_ActivationType

Syntax	`Icu_17_TimerIp_ActivationType`	
Type	Enumeration	
File	`Icu_17_TimerIp.h`	
Range	0 - ICU_17_TIMERIP_RISING_EDGE	An appropriate action will be executed when a rising edge occurs on the ICU input signal
	1 - ICU_17_TIMERIP_FALLING_EDGE	An appropriate action will be executed when a falling edge occurs on the ICU input signal
	2 - ICU_17_TIMERIP_BOTH_EDGES	An appropriate action will be executed when either a rising or falling edge occur on the ICU input signal
	3 - ICU_17_TIMERIP_NO_EDGE	No edge is selected
Description	Definition of the type of activation of an ICU channel.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.8 Icu_17_TimerIp_ValueType

Table 94 Specification for Icu_17_TimerIp_ValueType

Syntax	`Icu_17_TimerIp_ValueType`	
Type	uint32	
File	`Icu_17_TimerIp.h`	
Range	0-16777215	
Description	Width of the buffer for timestamp ticks and measured elapsed time ticks. 24-bit range for GTM(TIM) Channel. 16-bit range for GPT12 and CCU6 Channel.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.9 Icu_17_TimerIp_DutyCycleType

Table 95 Specification for Icu_17_TimerIp_DutyCycleType

Syntax	`Icu_17_TimerIp_DutyCycleType`	
Type	Structure	
File	`Icu_17_TimerIp.h`	
Range	Icu_17_TimerIp_ValueType ActiveTime	Coherent active time measured on a channel
	Icu_17_TimerIp_ValueType PeriodTime	Coherent period time measured on a channel
Description	Contains the values needed for calculating duty cycles.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.10 Icu_17_TimerIp_IndexType

Table 96 Specification for Icu_17_TimerIp_IndexType

Syntax	`Icu_17_TimerIp_IndexType`	
Type	uint16	
File	`Icu_17_TimerIp.h`	
Range	0-65535	
Description	Type, to abstract the return value of the service Icu_17_TimerIp_GetTimestampIndex(). Since circular buffer handling is supported and Icu_17_TimerIp_GetTimestampIndex can return zero as a legally true value.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.11 Icu_17_TimerIp_EdgeNumberType

Table 97 Specification for Icu_17_TimerIp_EdgeNumberType

Syntax	`Icu_17_TimerIp_EdgeNumberType`	
Type	uint32	
File	`Icu_17_TimerIp.h`	
Range	0-4294967295	
Description	Type to abstract the return value of the service Icu_17_TimerIp_GetEdgeNumbers().	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.12 Icu_17_TimerIp_MeasurementModeType

Table 98 Specification for Icu_17_TimerIp_MeasurementModeType

Syntax	`Icu_17_TimerIp_MeasurementModeType`	
Type	Enumeration	
File	`Icu_17_TimerIp.h`	
Range	0 - ICU_17_TIMERIP_MODE_SIGNAL_EDGE_DETECT	Edge Detection Mode
	1 - ICU_17_TIMERIP_MODE_SIGNAL_MEASUREMENT	Mode for measuring different times between various configurable edges
	2 - ICU_17_TIMERIP_MODE_TIMESTAMP	Mode for capturing timer values on configurable edges
	3 - ICU_17_TIMERIP_MODE_EDGE_COUNTER	Mode for counting edges on configurable edges
	4 - ICU_17_TIMERIP_MODE_INCREMENTEL_INTERFACE	Incremental Interface mode
Description	Definition of ICU measurement mode. Member of a config structure.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.13 Icu_17_TimerIp_SignalMeasurementPropertyType

Table 99 Specification for Icu_17_TimerIp_SignalMeasurementPropertyType

Syntax	`Icu_17_TimerIp_SignalMeasurementPropertyType`
Type	Enumeration
File	`Icu_17_TimerIp.h`

(table continues...)

Table 99 (continued) Specification for Icu_17_TimerIp_SignalMeasurementPropertyType

Range	0 - ICU_17_TIMERIP_LOW_TIME	The channel is configured for reading the elapsed signal low time
	1 - ICU_17_TIMERIP_HIGH_TIME	The channel is configured for reading the elapsed signal high time
	2 - ICU_17_TIMERIP_PERIOD_TIME	The channel is configured for reading the elapsed signal period time
	3 - ICU_17_TIMERIP_DUTY_CYCLE	The channel is configured to read values which are needed for calculating the duty cycle (coherent active and period time)
Description	Definition of the measurement property type.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.14 Icu_17_TimerIp_TimestampBufferType

Table 100 Specification for Icu_17_TimerIp_TimestampBufferType

Syntax	`Icu_17_TimerIp_TimestampBufferType`	
Type	Enumeration	
File	`Icu_17_TimerIp.h`	
Range	0 - ICU_17_TIMERIP_LINEAR_BUFFER	Buffer will be filled once
	1 - ICU_17_TIMERIP_CIRCULAR_BUFFER	After reaching the end of the buffer, the driver restarts at the beginning of the buffer
Description	Definition of the timestamp measurement property type.	
Source	AUTOSAR	
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 ICU driver.

1.3.3.1 Icu_17_TimerIp_CalibratePos

Table 101 Specification for `Icu_17_TimerIp_CalibratePos` API

Syntax	`void Icu_17_TimerIp_CalibratePos ( const Icu_17_TimerIp_ChannelType Channel, const uint16 Position )`

(table continues...)

Table 101 (continued) Specification for `Icu_17_TimerIp_CalibratePos` API

Service ID	0x23	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel Position	Numeric identifier of the ICU channel Start point to be set
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function calibrates the start point for incremental interface mode functionality. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	IFX	
Error handling	ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuIncrementalInterfaceApi	
User hints	-	
SFR accessed	CPU_CORE_ID(r), GPT12_T2(w), GPT12_T3(w), GPT12_T4(w), GPT12_T5(w), GPT12_T6(w) *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 Icu_17_TimerIp_CheckWakeup

Table 102 Specification for `Icu_17_TimerIp_CheckWakeup` API

Syntax	`void  Icu_17_TimerIp_CheckWakeup`
	`(`
	`    const EcuM_WakeupSourceType WakeupSource`
	`)`

Service ID	0x15
Sync/Async	Synchronous
Safety Level	Refer to the release notes for the safety related info
Re-entrancy	Reentrant for different channel

(table continues...)

Table 102 (continued) Specification for `Icu_17_TimerIp_CheckWakeup` API

Parameters (in)	WakeupSource	Information on wakeup source to be checked. The associated ICU channel can be determined from configuration data.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	Checks if a wakeup capable ICU channel is the source for a wakeup event and calls the ECU state manager service EcuM_SetWakeupEvent in case of a valid ICU channel wakeup event. For multicore, the ICU channel should be allocated to the core in which this function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT	
Configuration dependencies	IcuReportWakeupSource,IcuWakeupFunctionalityApi	
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.	

1.3.3.3 Icu_17_TimerIp_DeInit

Table 103 Specification for `Icu_17_TimerIp_DeInit` API

Syntax	`void  Icu_17_TimerIp_DeInit`
	`(`
	`  void`
	`)`	

Service ID	0x01	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	-	-
Parameters (out)	-	-

(table continues...)

Table 103 **(continued) Specification for `Icu_17_TimerIp_DeInit` API**

Parameters (in - out)	-	-
Return	void	-
Description	The function de-initializes the ICU driver. For multicore, the function will de-initialize those channels allocated to the core in which the function is invoked. Additionally if called from master core, de-initialize the resources shared among the cores. *Note: User shall not call Icu_17_TimerIp_DeInit during a running operation (e. g. timestamp measurement or edge counting)*	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_SLAVE_CORE_INIT	
Configuration dependencies	IcuDeInitApi	
User hints	-	
SFR accessed	CCU6_CC63SR(w), CCU6_CC6SR(w), CCU6_CMPMODIF(rw), CCU6_CMPSTAT(rw), CCU6_IEN(rw), CCU6_INP(rw), CCU6_MODCTR(rw), CCU6_PISEL0(rw), CCU6_PISEL2(rw), CCU6_PSLR(rw), CCU6_T12(w), CCU6_T12MSEL(rw), CCU6_T12PR(w), CCU6_T13(w), CCU6_T13PR(w), CCU6_TCTR0(rw), CCU6_TCTR2(rw), CCU6_TCTR4(rw), CPU_COMPAT(w), CPU_CORE_ID(r), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), GPT12_PISEL(rw), GPT12_T2(w), GPT12_T2CON(w), GPT12_T3(w), GPT12_T3CON(w), GPT12_T4(w), GPT12_T4CON(w), GPT12_T5(w), GPT12_T5CON(w), GPT12_T6(w), GPT12_T6CON(w), GTM_TIM_CH_IRQ_NOTIFY(w), GTM_TIM_RST(rw), SCU_CCUCON0(r), SCU_EICR(rw), SCU_FMR(w), SCU_IGCR(rw), 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.	

1.3.3.4 Icu_17_TimerIp_DisableEdgeCount

Table 104 **Specification for `Icu_17_TimerIp_DisableEdgeCount` API**

Syntax	`void  Icu_17_TimerIp_DisableEdgeCount`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`

Service ID	0x0e
Sync/Async	Synchronous
Safety Level	Refer to the release notes for the safety related info
Re-entrancy	Reentrant for different channel

(table continues...)

Table 104 **(continued) Specification for `Icu_17_TimerIp_DisableEdgeCount` API**

Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function disables the counting of edges of the given channel. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuEdgeCountApi	
User hints	-	
SFR accessed	CPU_CORE_ID(r), GPT12_T2CON(rw), GPT12_T3CON(rw), GPT12_T4CON(rw), GPT12_T5CON(rw), GPT12_T6CON(rw), GTM_TIM_CH_CTRL(rw) *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 Icu_17_TimerIp_DisableNotification

Table 105 **Specification for `Icu_17_TimerIp_DisableNotification` API**

Syntax	`void Icu_17_TimerIp_DisableNotification ( const Icu_17_TimerIp_ChannelType Channel )`	
Service ID	0x06	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-

(table continues...)

Table 105 **(continued) Specification for `Icu_17_TimerIp_DisableNotification` API**

Parameters (in - out)	-	-
Return	void	-
Description	The function disables the notification of a channel. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	-	
User hints	-	
SFR accessed	CCU6_IEN(rw), CPU_COMPAT(w), CPU_CORE_ID(r), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), GPT12_T2(w), GPT12_T3(w), GPT12_T4(w), GPT12_T5(w), GPT12_T6(w), GTM_TIM_CH_IRQ_EN(w), SCU_CCUCON0(r), SCU_IGCR(rw), 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.	

1.3.3.6 Icu_17_TimerIp_DisableWakeup

Table 106 **Specification for `Icu_17_TimerIp_DisableWakeup` API**

Syntax	`void  Icu_17_TimerIp_DisableWakeup`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`	

Service ID	0x03	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-

(table continues...)

Table 106 (continued) Specification for `Icu_17_TimerIp_DisableWakeup` API

Return	void	-
Description	The function disables the wakeup capability of a single ICU channel. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuDisableWakeupApi	
User hints	-	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.7 Icu_17_TimerIp_EnableEdgeCount

Table 107 Specification for `Icu_17_TimerIp_EnableEdgeCount` API

Syntax	
	```
	void  Icu_17_TimerIp_EnableEdgeCount
	(
	    const Icu_17_TimerIp_ChannelType Channel
	)
	```

Service ID	0x0d	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function enables the counting of edges of the given channel. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	

(table continues...)

Table 107 (continued) Specification for `Icu_17_TimerIp_EnableEdgeCount` API

Configuration dependencies	IcuEdgeCountApi
User hints	-
SFR accessed	CPU_CORE_ID(r), GPT12_T2CON(rw), GPT12_T3CON(rw), GPT12_T4CON(rw), GPT12_T5CON(rw), GPT12_T6CON(rw), GTM_TIM_CH_CNT(r), GTM_TIM_CH_CTRL(rw), GTM_TIM_CH_IRQ_EN(w), GTM_TIM_CH_IRQ_NOTIFY(w) *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 Icu_17_TimerIp_EnableMultiEdgeDetection

Table 108 Specification for `Icu_17_TimerIp_EnableMultiEdgeDetection` API

Syntax	`void Icu_17_TimerIp_EnableMultiEdgeDetection ( const Icu_17_TimerIp_ChannelType Channel, const uint32 EdgeCount )`	
Service ID	0x19	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel EdgeCount	Numeric identifier of the ICU channel Number of edges before interrupt occurs
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The multi edge detection feature is provided to generate interrupts after specified number of edges. It is possible to enable this feature at runtime for desired number of edges. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	IFX	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_PARAM_EDGE_NUMBER, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	

(table continues...)

Table 108 (continued) Specification for `Icu_17_TimerIp_EnableMultiEdgeDetection` API

Configuration dependencies	IcuEdgeDetectApi
User hints	-
SFR accessed	CPU_CORE_ID(r), GPT12_T2(w), GPT12_T2CON(rw), GPT12_T3(w), GPT12_T3CON(rw), GPT12_T4(w), GPT12_T4CON(rw), GPT12_T5(w), GPT12_T5CON(rw), GPT12_T6(w), GPT12_T6CON(rw), GTM_TIM_CH_CNTS(w), GTM_TIM_CH_CTRL(rw), GTM_TIM_CH_IRQ_NOTIFY(w) *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.

1.3.3.9 Icu_17_TimerIp_EnableNotification

Table 109 Specification for `Icu_17_TimerIp_EnableNotification` API

Syntax	`void  Icu_17_TimerIp_EnableNotification ( const Icu_17_TimerIp_ChannelType Channel )`	
Service ID	0x07	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function enables the notification on the given channel. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE, ICU_17_TIMERIP_E_INVALID_NOTIF	
Configuration dependencies	-	

(table continues...)

Table 109 (continued) Specification for `Icu_17_TimerIp_EnableNotification` API

User hints	-
SFR accessed	CCU6_IEN(rw), CCU6_IS(r), CCU6_ISR(w), CPU_COMPAT(w), CPU_CORE_ID(r), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), GPT12_T2(rw), GPT12_T3(rw), GPT12_T4(rw), GPT12_T5(rw), GPT12_T6(rw), GTM_TIM_CH_IRQ_EN(w), GTM_TIM_CH_IRQ_NOTIFY(rw), SCU_CCUCON0(r), SCU_EIFR(r), SCU_FMR(w), SCU_IGCR(rw), 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.

1.3.3.10 Icu_17_TimerIp_EnableWakeup

Table 110 Specification for `Icu_17_TimerIp_EnableWakeup` API

Syntax	`void  Icu_17_TimerIp_EnableWakeup`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`	

Service ID	0x04	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function (re-)enables the wakeup capability of the given ICU channel. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuEnableWakeupApi	
User hints	-	

(table continues...)

Table 110 **(continued) Specification for `Icu_17_TimerIp_EnableWakeup` API**

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.

1.3.3.11 Icu_17_TimerIp_GetDutyCycleValues

Table 111 **Specification for `Icu_17_TimerIp_GetDutyCycleValues` API**

Syntax	`void  Icu_17_TimerIp_GetDutyCycleValues`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel,`
	`    Icu_17_TimerIp_DutyCycleType * const DutyCycleValues`
	`)`	

Service ID	0x11	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	DutyCycleValues	Pointer to a buffer where the results (active time and period time) will be placed
Parameters (in - out)	-	-
Return	void	-
Description	The function reads the coherent active time and period time for the given ICU Channel. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_PARAM_POINTER, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuGetDutyCycleValuesApi	
User hints	For a GTM channel the overflow is identified and ZERO will be returned. For a CCU6 channel there is no unique way to identify overflow and hence and the input signal must be within the 16-bit range.	

(table continues...)

Table 111 (continued) Specification for `Icu_17_TimerIp_GetDutyCycleValues` API

SFR accessed	CPU_CORE_ID(r), GTM_TIM_CH_GPR0(r), GTM_TIM_CH_GPR1(r), GTM_TIM_CH_IRQ_NOTIFY(rw) *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.

1.3.3.12 Icu_17_TimerIp_GetEdgeNumbers

Table 112 Specification for `Icu_17_TimerIp_GetEdgeNumbers` API

Syntax	`Icu_17_TimerIp_EdgeNumberType Icu_17_TimerIp_GetEdgeNumbers`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`	

Service ID	0x0f	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Icu_17_TimerIp_EdgeNumberType	Edge Count for an ICU channel.
Description	The function reads the number of counted edges. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuEdgeCountApi	
User hints	This API can be invoked even if edge counting activity is not active.	

(table continues...)

Table 112 (continued) Specification for `Icu_17_TimerIp_GetEdgeNumbers` API

SFR accessed	CPU_CORE_ID(r), GPT12_T2(r), GPT12_T3(r), GPT12_T4(r), GPT12_T5(r), GPT12_T6(r), GTM_TIM_CH_CNT(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.

1.3.3.13 Icu_17_TimerIp_GetInputState

Table 113 Specification for `Icu_17_TimerIp_GetInputState` API

Syntax	`Icu_17_TimerIp_InputStateType  Icu_17_TimerIp_GetInputState`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`	

Service ID	0x08	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Icu_17_TimerIp_InputStateType	ICU_17_TIMERIP_ACTIVE/ICU_17_TIMERIP_IDLE
Description	The function returns the status of the ICU input. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuGetInputStateApi	
User hints	If Icu_GetInputState API is invoked for a channel in multi-edge detection mode, the channel status is set to ACTIVE only after the required number of edges are detected and not on the next detected single active edge.	

(table continues...)

Table 113 **(continued) Specification for `Icu_17_TimerIp_GetInputState` API**

SFR accessed	CCU6_CMPMODIF(w), CCU6_CMPSTAT(r), CPU_CORE_ID(r), GPT12_T2(rw), GPT12_T3(rw), GPT12_T4(rw), GPT12_T5(rw), GPT12_T6(rw), GTM_TIM_CH_IRQ_NOTIFY(rw), SCU_EIFR(r), SCU_FMR(w) *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 Icu_17_TimerIp_GetTimeElapsed

Table 114 **Specification for `Icu_17_TimerIp_GetTimeElapsed` API**

Syntax	`Icu_17_TimerIp_ValueType  Icu_17_TimerIp_GetTimeElapsed`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`	

Service ID	0x10	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Icu_17_TimerIp_ValueType	Signal Low time, High timer or period value for the channel.
Description	This function reads the elapsed Signal Measurement Time for the given channel For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuGetTimeElapsedApi	
User hints	For a GTM channel the overflow is identified and ZERO will be returned. For a CCU6 channel there is no unique way to identify overflow and hence and the input signal must be within the 16-bit range.	

(table continues...)

Table 114 (continued) Specification for `Icu_17_TimerIp_GetTimeElapsed` API

SFR accessed	CCU6_CC6R(r), CCU6_CC6SR(r), CCU6_CMPMODIF(w), CCU6_CMPSTAT(r), CPU_CORE_ID(r), GTM_TIM_CH_GPR1(r), GTM_TIM_CH_IRQ_NOTIFY(rw) *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 Icu_17_TimerIp_GetTimestampIndex

Table 115 Specification for `Icu_17_TimerIp_GetTimestampIndex` API

Syntax	`Icu_17_TimerIp_IndexType  Icu_17_TimerIp_GetTimestampIndex`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`	

Service ID	0x0b	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Icu_17_TimerIp_IndexType	Timestamp index next to be written.
Description	The function reads the timestamp index of the given channel. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH	
Configuration dependencies	IcuTimestampApi	
User hints	This API will return the size of the buffer if the buffer is full and buffer configuration is linear. This API can be invoked even if there is no active time stamping activity.	

(table continues...)

Table 115 **(continued) Specification for `Icu_17_TimerIp_GetTimestampIndex` API**

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.

1.3.3.16 Icu_17_TimerIp_GetVersionInfo

Table 116 **Specification for `Icu_17_TimerIp_GetVersionInfo` API**

Syntax	`void  Icu_17_TimerIp_GetVersionInfo`
	`(`
	`    Std_VersionInfoType * const versioninfo`
	`)`	

Service ID	0x12	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	versioninfo	Pointer to where to store the version information.
Parameters (in - out)	-	-
Return	void	-
Description	The function returns the version information of the ICU driver.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_PARAM_VINFO	
Configuration dependencies	IcuGetVersionInfoApi	
User hints	The API can be called before ICU initialization.	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.17 Icu_17_TimerIp_InitCheck

Table 117 Specification for `Icu_17_TimerIp_InitCheck` API

Syntax	`Std_ReturnType  Icu_17_TimerIp_InitCheck`
	`(`
	`    const Icu_17_TimerIp_ConfigType * const ConfigPtr`
	`)`	

Service ID	0x30	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	ConfigPtr	Pointer to a selected configuration structure
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Std_ReturnType	E_OK - if initialization comparison is success. E_NOT_OK - In Case of - Driver is not initialized - Input config Pointer is Null - Input config pointer is other than the one used for Init - Global Variables or SFR is not set as expected - Icu driver is not in Normal Mode
Description	Will check against all SFRs or variables initialized by Init API including initialization status flag. It does not modify any SFR/variable, only a read operation is done. If any failure in comparison, it reports an error. For multicore, the function will check the initialization of those channels allocated to the core in which this function is invoked. Additionally for master core, the function will check the initialization of the resources which are shared among cores.	
Source	IFX	
Error handling	-	
Configuration dependencies	IcuInitCheckApi	
User hints	-	

(table continues...)

Table 117 (continued) Specification for `Icu_17_TimerIp_InitCheck` API

SFR accessed	CCU6_CC63SR(r), CCU6_CC6SR(r), CCU6_CLC(r), CCU6_CMPSTAT(r), CCU6_IEN(r), CCU6_INP(r), CCU6_MODCTR(r), CCU6_PISEL0(r), CCU6_PISEL2(r), CCU6_PSLR(r), CCU6_T12MSEL(r), CCU6_T12PR(r), CCU6_T13PR(r), CCU6_TCTR0(r), CCU6_TCTR2(r), CPU_CORE_ID(r), GPT12_CLC(r), GPT12_PISEL(r), GPT12_T2CON(r), GPT12_T3CON(r), GPT12_T4CON(r), GPT12_T5CON(r), GPT12_T6CON(r), GTM_TIM_CH_CTRL(r), GTM_TIM_CH_ECTRL(r), GTM_TIM_CH_FLT_FE(r), GTM_TIM_CH_FLT_RE(r), GTM_TIM_CH_IRQ_EN(r), GTM_TIM_CH_IRQ_MODE(r), GTM_TIM_CH_IRQ_NOTIFY(r), GTM_TIM_CH_TDUV(r), SCU_EICR(r), SCU_EIFR(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 Icu_17_TimerIp_ReadEncCount

Table 118 Specification for `Icu_17_TimerIp_ReadEncCount` API

Syntax	`uint16 Icu_17_TimerIp_ReadEncCount`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`	

Service ID	0x24	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint16	Encoder counter value
Description	The function reads the current encoder count value. The encoder count and direction are not impacted by the call of the API. If a DET/Safety error is identified, 0 is returned. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	IFX	
Error handling	ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	

(table continues...)

Table 118 **(continued) Specification for `Icu_17_TimerIp_ReadEncCount` API**

Configuration dependencies	IcuIncrementalInterfaceApi
User hints	-
SFR accessed	CPU_CORE_ID(r), GPT12_T2(r), GPT12_T3(r), GPT12_T4(r), GPT12_T5(r), GPT12_T6(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 Icu_17_TimerIp_ReadEncCountDir

Table 119 **Specification for `Icu_17_TimerIp_ReadEncCountDir` API**

Syntax	`Icu_17_TimerIp_EncCountDirType  Icu_17_TimerIp_ReadEncCountDir (     const Icu_17_TimerIp_ChannelType Channel )`	
Service ID	0x25	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Icu_17_TimerIp_EncCountDirType	Counting direction
Description	The function to read the direction of rotation. The encoder count and direction are not impacted by the call of the API. If a DET/Safety error is identified, ICU_17_TIMERIP_ENC_COUNT_UP is returned. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	IFX	
Error handling	ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuIncrementalInterfaceApi	
User hints	-	

(table continues...)

Table 119 (continued) Specification for `Icu_17_TimerIp_ReadEncCountDir` API

SFR accessed	CPU_CORE_ID(r), GPT12_T2CON(r), GPT12_T3CON(r), GPT12_T4CON(r), GPT12_T5CON(r), GPT12_T6CON(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 Icu_17_TimerIp_ResetEdgeCount

Table 120 Specification for `Icu_17_TimerIp_ResetEdgeCount` API

Syntax	`void Icu_17_TimerIp_ResetEdgeCount`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`	

Service ID	0x0c	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function resets the value of the counted edges to zero. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuEdgeCountApi	
User hints	No active edge should be seen on the input pin during the execution of the API.	
SFR accessed	CPU_CORE_ID(r), GPT12_T2(w), GPT12_T3(w), GPT12_T4(w), GPT12_T5(w), GPT12_T6(w), GTM_TIM_CH_CNT(r), GTM_TIM_CH_CTRL(rw) *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 continues...)

Table 120 **(continued) Specification for `Icu_17_TimerIp_ResetEdgeCount` API**

Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.21 Icu_17_TimerIp_SetActivationCondition

Table 121 **Specification for `Icu_17_TimerIp_SetActivationCondition` API**

Syntax	`void  Icu_17_TimerIp_SetActivationCondition`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel,`
	`    const Icu_17_TimerIp_ActivationType Activation`
	`)`	

Service ID	0x05	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel Activation	Numeric identifier of the ICU channel Type of activation edge to be configured - ICU_17_TIMERIP_RISING_EDGE - ICU_17_TIMERIP_FALLING_EDGE - ICU_17_TIMERIP_BOTH_EDGES
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function sets the activation-edge for the given channel. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_PARAM_ACTIVATION, ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE, ICU_17_TIMERIP_E_BUSY_CHANNEL	
Configuration dependencies	-	
User hints	The channel on which Icu_SetActivationCondition is invoked, must not have any on-going operations to ensure proper functionality. The API must be invoked only on channels configured in edge detection, edge counting and time stamping mode. The API will issue a ICU_E_PARAM_CHANNEL DET/Safety error if the channel parameter corresponds to a signal measurement channel.	

(table continues...)

Table 121 **(continued) Specification for `Icu_17_TimerIp_SetActivationCondition` API**

SFR accessed	CCU6_CMPMODIF(w), CCU6_IEN(rw), CCU6_ISR(w), CPU_COMPAT(w), CPU_CORE_ID(r), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), GPT12_T2(w), GPT12_T2CON(rw), GPT12_T3(w), GPT12_T3CON(rw), GPT12_T4(w), GPT12_T4CON(rw), GPT12_T5(w), GPT12_T5CON(rw), GPT12_T6(w), GPT12_T6CON(rw), GTM_TIM_CH_CTRL(rw), GTM_TIM_CH_IRQ_NOTIFY(w), SCU_CCUCON0(r), SCU_EICR(rw), SCU_FMR(w), 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 Icu_17_TimerIp_SetTimeoutValue

Table 122 **Specification for `Icu_17_TimerIp_SetTimeoutValue` API**

Syntax	`void  Icu_17_TimerIp_SetTimeoutValue`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel,`
	`    const uint32 TimeOut`
	`)`	

Service ID	0x26	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel TimeOut	Numeric identifier of the ICU channel TimeOut period in ticks
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function sets the timeout value of the given channel. After ICU initialization timeout is disabled. User has to invoke the Icu_17_TimerIp_SetTimeoutValue API to set the timeout value and hence enabling the timeout feature. All further calls of the API updates the timeout value. If the timeout value is set as 0, no more timeout notifications will be issued. If the timeout and edge interrupt occur at same instance, edge interrupt is given and no notification will be issued. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	IFX	

(table continues...)

Table 122 **(continued) Specification for `Icu_17_TimerIp_SetTimeoutValue` API**

Error handling	ICU_17_TIMERIP_E_INVALID_MODE, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_PARAM_TIMEOUT, ICU_17_TIMERIP_E_UNINIT
Configuration dependencies	IcuTimeoutDetectionApi
User hints	-
SFR accessed	GTM_TIM_CH_CTRL(rw), GTM_TIM_CH_TDUV(rw) *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 Icu_17_TimerIp_StartIncInterface

Table 123 **Specification for `Icu_17_TimerIp_StartIncInterface` API**

Syntax	`void Icu_17_TimerIp_StartIncInterface`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`	

Service ID	0x21	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function starts the incremental interface mode activity of the ICU channel. The encoder count and direction are not impacted by the call of the API. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	IFX	
Error handling	ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuIncrementalInterfaceApi	

(table continues...)

Table 123 **(continued) Specification for `Icu_17_TimerIp_StartIncInterface` API**

User hints	-	
SFR accessed	CPU_CORE_ID(r), GPT12_T2CON(rw), GPT12_T3CON(rw), GPT12_T4CON(rw), GPT12_T5CON(rw), GPT12_T6CON(rw) *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.	

1.3.3.24 Icu_17_TimerIp_StartSignalMeasurement

Table 124 **Specification for `Icu_17_TimerIp_StartSignalMeasurement` API**

Syntax	`void Icu_17_TimerIp_StartSignalMeasurement`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`	

Service ID	0x13	
Sync/Async	Asynchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function starts the measurement of signals. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuSignalMeasurementApi	
User hints	-	

(table continues...)

Table 124 (continued) Specification for `Icu_17_TimerIp_StartSignalMeasurement` API

SFR accessed	CCU6_CC6SR(w), CCU6_IEN(rw), CCU6_ISR(w), CCU6_T12MSEL(rw), CCU6_TCTR4(rw), CPU_CORE_ID(r), GTM_TIM_CH_CTRL(rw), GTM_TIM_CH_ECNT(r), GTM_TIM_CH_ECTRL(rw), GTM_TIM_CH_IRQ_EN(w), GTM_TIM_CH_IRQ_NOTIFY(w) *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.

1.3.3.25 Icu_17_TimerIp_StartTimestamp

Table 125 Specification for `Icu_17_TimerIp_StartTimestamp` API

Syntax	`void  Icu_17_TimerIp_StartTimestamp`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel,`
	`    Icu_17_TimerIp_ValueType * const BufferPtr,`
	`    const uint16 BufferSize,`
	`    const uint16 NotifyInterval`
	`)`	

Service ID	0x09	
Sync/Async	Asynchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel BufferPtr BufferSize NotifyInterval	Numeric identifier of the ICU channel Pointer to the buffer-array where the timestamp values will be placed. Size of the external buffer (number of entries) Notification interval (number of events).
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function starts the capturing of timer values on the edges. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	

(table continues...)

Table 125 (continued) Specification for `Icu_17_TimerIp_StartTimestamp` API

Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_POINTER, ICU_17_TIMERIP_E_PARAM_NOTIFY_INTERVAL, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_PARAM_BUFFER_SIZE, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE, ICU_17_TIMERIP_E_PARAM_IMPLAUSIBLE_NOTIFY_INTERVAL
Configuration dependencies	IcuTimestampApi
User hints	-
SFR accessed	CCU6_IEN(rw), CCU6_ISR(w), CCU6_T12MSEL(rw), CPU_CORE_ID(r), GTM_TIM_CH_CTRL(rw), GTM_TIM_CH_IRQ_EN(w), GTM_TIM_CH_IRQ_NOTIFY(w) *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.

1.3.3.26 Icu_17_TimerIp_StopIncInterface

Table 126 Specification for `Icu_17_TimerIp_StopIncInterface` API

Syntax	`void  Icu_17_TimerIp_StopIncInterface`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`	

Service ID	0x22	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function stops the incremental interface mode activity of the ICU channel. The encoder count and direction are not impacted by the call of the API. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	IFX	
Error handling	ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_INVALID_MODE	

(table continues...)

Table 126 (continued) Specification for `Icu_17_TimerIp_StopIncInterface` API

Configuration dependencies	IcuIncrementalInterfaceApi
User hints	-
SFR accessed	CPU_CORE_ID(r), GPT12_T2CON(rw), GPT12_T3CON(rw), GPT12_T4CON(rw), GPT12_T5CON(rw), GPT12_T6CON(rw) *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.

1.3.3.27 Icu_17_TimerIp_StopSignalMeasurement

Table 127 Specification for `Icu_17_TimerIp_StopSignalMeasurement` API

Syntax	`void Icu_17_TimerIp_StopSignalMeasurement`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`	

Service ID	0x14	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function stops the measurement of signals of the given channel. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuSignalMeasurementApi	
User hints	-	

(table continues...)

Table 127 **(continued) Specification for `Icu_17_TimerIp_StopSignalMeasurement` API**

SFR accessed	CCU6_IEN(rw), CCU6_T12MSEL(rw), CPU_CORE_ID(r), GTM_TIM_CH_CTRL(rw) *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.

1.3.3.28 Icu_17_TimerIp_StopTimestamp

Table 128 **Specification for `Icu_17_TimerIp_StopTimestamp` API**

Syntax	`void  Icu_17_TimerIp_StopTimestamp`
	`(`
	`    const Icu_17_TimerIp_ChannelType Channel`
	`)`	

Service ID	0x0a	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function stops the timestamp measurement of the given channel. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_NOT_STARTED, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuTimestampApi	
User hints	None.	
SFR accessed	CCU6_T12MSEL(rw), CPU_CORE_ID(r), GTM_TIM_CH_CTRL(rw), GTM_TIM_CH_IRQ_EN(w) *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.	

1.3.3.29 Icu_17_TimerIp_EnableEdgeDetection

Table 129 Specification for `Icu_17_TimerIp_EnableEdgeDetection` API

Syntax	`void  Icu_17_TimerIp_EnableEdgeDetection ( const Icu_17_TimerIp_ChannelType Channel )`	
Service ID	0x16	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function enables / re-enables the detection of edges of the given channel. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuEdgeDetectApi	
User hints	-	
SFR accessed	CCU6_ISR(w), CCU6_T12MSEL(rw), CPU_COMPAT(w), CPU_CORE_ID(r), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), GPT12_T2(w), GPT12_T2CON(rw), GPT12_T3(w), GPT12_T3CON(rw), GPT12_T4(w), GPT12_T4CON(rw), GPT12_T5(w), GPT12_T5CON(rw), GPT12_T6(w), GPT12_T6CON(rw), GTM_TIM_CH_CTRL(rw), GTM_TIM_CH_IRQ_NOTIFY(w), SCU_CCUCON0(r), SCU_EICR(rw), SCU_FMR(w), 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.	

1.3.3.30 Icu_17_TimerIp_DisableEdgeDetection

Table 130 Specification for `Icu_17_TimerIp_DisableEdgeDetection` API

Syntax	`void Icu_17_TimerIp_DisableEdgeDetection ( const Icu_17_TimerIp_ChannelType Channel )`	
Service ID	0x17	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channel	
Parameters (in)	Channel	Numeric identifier of the ICU channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function disables the detection of edges of the given channel. For multicore, the ICU channel should be allocated to the core in which the function is invoked.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_CHANNEL, ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH, ICU_17_TIMERIP_E_INVALID_MODE	
Configuration dependencies	IcuEdgeDetectApi	
User hints	-	
SFR accessed	CCU6_T12MSEL(rw), CPU_COMPAT(w), CPU_CORE_ID(r), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), GPT12_T2CON(rw), GPT12_T3CON(rw), GPT12_T4CON(rw), GPT12_T5CON(rw), GPT12_T6CON(rw), GTM_TIM_CH_CTRL(rw), SCU_CCUCON0(r), SCU_EICR(rw), 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.	

1.3.3.31 Icu_17_TimerIp_Init

Table 131 Specification for `Icu_17_TimerIp_Init` API

Syntax	`void  Icu_17_TimerIp_Init`
	`(`
	`    const Icu_17_TimerIp_ConfigType * const ConfigPtr`
	`)`	

Service ID	0x00	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	ConfigPtr	Pointer to a selected configuration structure
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	The function initializes the driver. For multicore, the function will initialize those channels allocated to the core in which this function is invoked. Additionally for master core, the function will initialize the resources which are shared among cores.	
Source	AUTOSAR	
Error handling	ICU_17_TIMERIP_E_INIT_FAILED, ICU_17_TIMERIP_E_ALREADY_INITIALIZED, ICU_17_TIMERIP_E_MASTER_CORE_UNINIT, ICU_17_TIMERIP_E_CORE_NOT_CONFIGURED	
Configuration dependencies	-	
User hints	Signal measurement will not be started after Init. A call to Icu_17_TimerIp_StartSignalMeasurement is required to start the signal measurement activity.	

(table continues...)

Table 131 (continued) Specification for `Icu_17_TimerIp_Init` API

SFR accessed	CCU6_CC63SR(w), CCU6_CC6SR(w), CCU6_CMPMODIF(rw), CCU6_CMPSTAT(rw), CCU6_IEN(rw), CCU6_INP(rw), CCU6_ISR(rw), CCU6_MODCTR(rw), CCU6_PISEL0(rw), CCU6_PISEL2(rw), CCU6_PSLR(rw), CCU6_T12(w), CCU6_T12MSEL(rw), CCU6_T12PR(w), CCU6_T13(w), CCU6_T13PR(w), CCU6_TCTR0(rw), CCU6_TCTR2(rw), CCU6_TCTR4(rw), CPU_COMPAT(w), CPU_CORE_ID(r), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), GPT12_PISEL(rw), GPT12_T2(w), GPT12_T2CON(w), GPT12_T3(w), GPT12_T3CON(w), GPT12_T4(w), GPT12_T4CON(w), GPT12_T5(w), GPT12_T5CON(w), GPT12_T6(w), GPT12_T6CON(w), GTM_TIM_CH_CTRL(rw), GTM_TIM_CH_ECTRL(w), GTM_TIM_CH_FLT_FE(w), GTM_TIM_CH_FLT_RE(w), GTM_TIM_CH_IRQ_EN(w), GTM_TIM_CH_IRQ_MODE(w), GTM_TIM_CH_IRQ_NOTIFY(w), GTM_TIM_CH_TDUV(w), SCU_CCUCON0(r), SCU_EICR(rw), SCU_FMR(w), 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.

1.3.3.32 Icu_17_TimerIp_SetMode

Table 132 Specification for `Icu_17_TimerIp_SetMode` API

Syntax	`void Icu_17_TimerIp_SetMode`
	`(`
	`    const Icu_17_TimerIp_ModeType Mode`
	`)`	

Service ID	0x02	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Non Reentrant	
Parameters (in)	Mode	ICU_17_TIMERIP_MODE_NORMAL: Normal operation, all used interrupts are enabled according to the notification requests. ICU_17_TIMERIP_MODE_SLEEP: Reduced power mode. In sleep mode only those notifications are available which are configured as wakeup capable.
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-

(table continues...)

Table 132 (continued) Specification for `Icu_17_TimerIp_SetMode` API

Description	The function sets the ICU mode. The DET, ICU_17_TIMERIP_E_BUSY_OPERATION, is issued if SLEEP mode is requested during a running operation of edge count channel, incremental interface channel, time stamp channel or signal measurement channel. For multicore, the function sets the mode of the core with which the function is invoked.
Source	AUTOSAR
Error handling	ICU_17_TIMERIP_E_BUSY_OPERATION, ICU_17_TIMERIP_E_UNINIT, ICU_17_TIMERIP_E_PARAM_MODE
Configuration dependencies	IcuSetModeApi
User hints	None.
SFR accessed	CCU6_IEN(rw), CCU6_IS(r), CCU6_ISR(w), CPU_COMPAT(w), CPU_CORE_ID(r), CPU_SYSCON(w), CPU_TPS_EXTIM_CLASS_EN(w), CPU_TPS_EXTIM_ENTRY_LVAL(w), CPU_TPS_EXTIM_EXIT_LVAL(w), GPT12_T2(rw), GPT12_T2CON(rw), GPT12_T3(rw), GPT12_T3CON(rw), GPT12_T4(rw), GPT12_T4CON(rw), GPT12_T5(rw), GPT12_T5CON(rw), GPT12_T6(rw), GPT12_T6CON(rw), GTM_TIM_CH_IRQ_EN(w), GTM_TIM_CH_IRQ_NOTIFY(rw), SCU_CCUCON0(r), SCU_EIFR(r), SCU_FMR(w), SCU_IGCR(rw), 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.

1.3.4 Notifications and Callbacks

This section lists all the notifications and callbacks of the ICU driver.

1.3.4.1 Icu_17_TimerIp_Timer_Isr

Table 133 Specification for `Icu_17_TimerIp_Timer_Isr` API

Syntax	`void  Icu_17_TimerIp_Timer_Isr`
	`(`
	`    const uint32 Channel,`
	`    const uint32 Flags`
	`)`	

Service ID	0x20	
Sync/Async	Synchronous	
Safety Level	Refer to the release notes for the safety related info	
Re-entrancy	Reentrant for different channels	
Parameters (in)	Channel	Logical channel identifier.
	Flags	Interrupt flags responsible for ISR

(table continues...)

Table 133 **(continued) Specification for `Icu_17_TimerIp_Timer_Isr` API**

Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	Callback function from MCU to service timer (ERU, GTM-TIM, CCU6 and GPT12) interrupts for all modes of ICU. The ISR is reentrant because access to any non-channel based timer resource is protected by protection mechanisms.	
Source	IFX	
Error handling	ICU_17_TIMERIP_E_INVALID_ISR	
Configuration dependencies	-	
User hints	-	
SFR accessed	CCU6_CC6R(r), CCU6_CC6SR(r), CCU6_IEN(rw), CCU6_T12MSEL(rw), CPU_CORE_ID(r), GPT12_T2(rw), GPT12_T2CON(rw), GPT12_T3(rw), GPT12_T3CON(rw), GPT12_T4(rw), GPT12_T4CON(rw), GPT12_T5(rw), GPT12_T6(rw), GTM_TIM_CH_CNTS(r), GTM_TIM_CH_CTRL(rw), GTM_TIM_CH_ECNT(r), GTM_TIM_CH_GPR0(r), GTM_TIM_CH_GPR1(r), GTM_TIM_CH_IRQ_EN(rw), GTM_TIM_CH_IRQ_NOTIFY(rw) *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.	

1.3.5 Scheduled functions

The ICU driver does not provide any scheduled functions.

1.3.6 Interrupt service routines

The ICU driver does not provide any interrupt handlers.

1.3.7 Callout

The ICU driver does not provide any callout.

1.3.8 Errors Handling

This section describes the various error types reported by the ICU driver.

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
ICU_17_TIMERIP_E_ALREADY_INITIALIZED: Icu_17_TimerIp_Init API service called when the ICU driver and the hardware are already initialized.	AUTOSAR	0x17	DET_SAFETY	0x17	DET_SAFETY
ICU_17_TIMERIP_E_BUSY_CHANNEL: Activation edge of a time stamp channel modified during an active time stamping operation.	IFX	0xCC	SAFETY	0xCC	SAFETY
ICU_17_TIMERIP_E_BUSY_OPERATION: Icu_17_TimerIp_SetMode is called when a channel is in running condition.	AUTOSAR	0x16	DET_SAFETY	0x16	DET_SAFETY
ICU_17_TIMERIP_E_CORE_CHANNEL_MISMATCH: An API is called with the channel not allocated to executing core.	IFX	0x65	DET_SAFETY	0x65	DET_SAFETY
ICU_17_TIMERIP_E_CORE_NOT_CONFIGURED: Error reported when the ICU driver is not configured for the core in which an API is invoked.	IFX	0x64	DET_SAFETY	0x64	DET_SAFETY
ICU_17_TIMERIP_E_INIT_FAILED: Configuration pointer is NULL_PTR.	AUTOSAR	0x0D	DET_SAFETY	0x0D	DET_SAFETY
ICU_17_TIMERIP_E_INVALID_ISR: ISR invoked on a spurious interrupt.	IFX	0xC9	SAFETY	0xC9	SAFETY
ICU_17_TIMERIP_E_INVALID_MODE: API is invoked in SLEEP mode, with an intention to modify channel hardware registers or global variables.	IFX	0xCA	SAFETY	0xCA	SAFETY
ICU_17_TIMERIP_E_INVALID_NOTIF: Notification invoked on a non- notification function configured channel.	IFX	0xCB	SAFETY	0xCB	SAFETY
ICU_17_TIMERIP_E_MASTER_CORE_UNINIT: Error reported when slave core init is called without initializing master core.	IFX	0x66	DET_SAFETY	0x66	DET_SAFETY

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
ICU_17_TIMERIP_E_NOT_STARTED: An ICU API which stops a particular feature is called before the start of that feature.	AUTOSAR	0x15	DET_SAFETY	0x15	RUNTIME
ICU_17_TIMERIP_E_PARAM_ACTIVATION: Invalid activation parameter in API.	AUTOSAR	0x0C	DET_SAFETY	0x0C	DET_SAFETY
ICU_17_TIMERIP_E_PARAM_BUFFER_SIZE: Invalid buffer size used in API.	AUTOSAR	0x0E	DET_SAFETY	0x0E	DET_SAFETY
ICU_17_TIMERIP_E_PARAM_CHANNEL: Invalid channel number or the channel is not configured for the required measurement mode.	AUTOSAR	0x0B	DET_SAFETY	0x0B	DET_SAFETY
ICU_17_TIMERIP_E_PARAM_EDGE_NUMBER: Edge count parameter is set as zero or not in the range supported by channel. Valid only for multi edge detection API.	IFX	0x21	DET_SAFETY	0x21	DET_SAFETY
ICU_17_TIMERIP_E_PARAM_IMPLAUSIBLE_NOTIFY_INTERVAL: Notify interval is greater than buffer size in case of a Linear buffer.	IFX	0xCD	SAFETY	0xCD	SAFETY
ICU_17_TIMERIP_E_PARAM_MODE: Invalid mode is passed for the API.	AUTOSAR	0x0F	DET_SAFETY	0x0F	DET_SAFETY
ICU_17_TIMERIP_E_PARAM_NOTIFY_INTERVAL: Icu_17_TimerIp_StartTimeStamp API called with invalid NotifyInterval parameter. Zero is considered as invalid value.	AUTOSAR	0x18	DET_SAFETY	0x18	DET_SAFETY
ICU_17_TIMERIP_E_PARAM_POINTER: API called with invalid pointer.	AUTOSAR	0x0A	DET_SAFETY	0x0A	DET_SAFETY
ICU_17_TIMERIP_E_PARAM_TIMEOUT: Timeout value is not in valid range.	IFX	0xCE	SAFETY	0xCE	SAFETY
ICU_17_TIMERIP_E_PARAM_VINFO: Icu_17_TimerIp_GetVersionInfo API called with a NULL_PTR.	AUTOSAR	0x19	DET_SAFETY	0x19	DET_SAFETY

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
ICU_17_TIMERIP_E_SLAVE_CORE_INIT: Error reported when master de-initialization is called without de-initializing slave core.	IFX	0x67	DET_SAFETY	0x67	DET_SAFETY
ICU_17_TIMERIP_E_UNINIT: API service used without the driver initialization.	AUTOSAR	0x14	DET_SAFETY	0x14	DET_SAFETY

1.3.9 Deviations and limitations

The section describes the deviations and limitations of the ICU driver.

1.3.9.1 Deviations

This section describes the deviation for the ICU driver.

1.3.9.1.1 Software specification deviations

This section describes the deviations from software specification.

Table 134 Known deviations

Reference	Deviation
For all requirements related to Runtime errors	Reporting of Runtime error: Det_ReportRuntimeError is done through Mcal_Wrapper_Det_ReportRuntimeError interface. This is applicable for only AUTOSAR 4.4.0. All runtime error related datatypes and modified interfaces inclusion shall be done via Mcal_Wrapper.h

1.3.9.1.2 AMDC Violations

The ICU driver does not have any AMDC violations.

1.3.9.1.3 VSMD Violations

This section describes the violations reported by the EB VSMD checker tool with respect to AUTOSAR.

Table 135 Violations reported by VSMD checker tool for EB03

Rule ID:	EB03

(table continues...)

Table 135 (continued) Violations reported by VSMD checker tool for EB03

VSMD Node(s):	/AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuSignalEdgeDetection /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuSignalEdgeDetection/IcuSignalNotification /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuSignalMeasurement /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuTimestampMeasurement /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuTimestampMeasurement/IcuTimestampNotification /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuWakeup /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuWakeup/IcuChannelWakeupInfo /AURIX2G/EcucDefs/Icu/IcuGeneral/IcuKernelEcucPartitionRef
Description:	The StMD node has LOWER-MULTIPLICITY=0 and UPPER-MULTIPLICITY=1. The VSMD-node shall get the OPTIONAL-attribute instead of creating a list!
Additional Information :	

Table 136 Violations reported by VSMD checker tool for EB09

Rule ID:	EB09
VSMD Node(s):	/AURIX2G/EcucDefs/Icu
Description:	EB specific rule to check consistency of parameter postBuildVariantUsed.
Additional Information :	

Table 137 Violations reported by VSMD checker tool for EcucSws_1014

Rule ID:	EcucSws_1014
VSMD Node(s):	/AURIX2G/EcucDefs/Icu /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuMeasurementMode /AURIX2G/EcucDefs/Icu/IcuGeneral /AURIX2G/EcucDefs/Icu/IcuOptionalApis
Description:	Additional vendor specific parameter definitions (using ParameterTypes), container definitions and references shall be added to the VSMD according to the alphabetical order.
Additional Information :	

Table 138 Violations reported by VSMD checker tool for EcucSws_1035

Rule ID:	EcucSws_1035
VSMD Node(s):	/AURIX2G/EcucDefs/Icu /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuChannelEcucPartitionRef /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuChannelId /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuDefaultStartEdge /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuMeasurementMode /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuSignalEdgeDetection /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuSignalMeasurement /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuSignalMeasurement/IcuSignalMeasurementProperty /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuTimestampMeasurement /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuTimestampMeasurement/IcuTimestampMeasurementProperty /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuWakeup /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuChannel/IcuWakeup/IcuChannelWakeupInfo /AURIX2G/EcucDefs/Icu/IcuConfigSet/IcuMaxChannel /AURIX2G/EcucDefs/Icu/IcuGeneral/IcuEcucPartitionRef /AURIX2G/EcucDefs/Icu/IcuGeneral/IcuKernelEcucPartitionRef /AURIX2G/EcucDefs/Icu/IcuOptionalApis/IcuGetVersionInfoApi
Description:	For Containers, Parameters and References elements UUID must be unique (also between StMD and VSMD).
Additional Information :	

Table 139 Violations reported by VSMD checker tool for EcucSws_2101

Rule ID:	EcucSws_2101
VSMD Node(s):	/AURIX2G/EcucDefs/Icu/POST_BUILD_VARIANT_USED

(table continues...)

Table 139 (continued) Violations reported by VSMD checker tool for EcucSws_2101

Description:	For each ConfigurationVariant supported by the ModuleDef, there must be one ImplementationConfigClass element. In VSMD, the ImplementationConfigClass is mandatory.
Additional Information :	

Table 140 Violations reported by VSMD checker tool for EcucSws_6003

Rule ID:	EcucSws_6003
VSMD Node(s):	/AURIX2G/EcucDefs/Icu
Description:	The SHORT-NAME of the AR-PACKAGEs of StMD and VSMD must be different to ensure a unique SHORT-NAME-path.
Additional Information :	

Table 141 Violations reported by VSMD checker tool for TpsEcuc_06051_ASR41

Rule ID:	TpsEcuc_06051_ASR41
VSMD Node(s):	/AURIX2G/EcucDefs/Icu/POST_BUILD_VARIANT_USED
Description:	
Additional Information :	

1.3.9.2 Limitations

This section describes the limitations of the ICU driver.

Table 142 Known limitations

Reference	Limitation
`Icu_17_TimerIp_GetInputState`, `Icu_17_TimerIp_GetTimeElapsed`, `Icu_17_TimerIp_GetDutyCycleValues` API	For an ICU channel configured in the signal measurement mode using the CCU6 hardware, the measured value will be in the 16-bit range. The overflow cannot be identified due to hardware limitation.
`Icu_17_TimerIp_GetInputState`, `Icu_17_TimerIp_GetTimeElapsed`, `Icu_17_TimerIp_GetDutyCycleValues` API	For an ICU channel configured in the signal measurement mode using the GTM-TIM hardware, the measured value will be in the 24-bit range. If the input signal is such that the measured value is more than 24-bit, zero shall be returned.

Revision history

Table 143 Revision History

Date	Version	Description
2024-08-02	10.0	Released
2024-07-05	9.1	• Updated the Section 1.1.3.C file Structure to hide the link between Mcal_Wrapper.h and Det.h. • Updated section 1.1.4.6 Interrupt connections sample code, corrected the TIM channel id corresponding to the interrupt.
2023-12-07	9.0	Released
2023-12-05	8.1	• Added CcuGetDutyCycle, CcuGetTimeElasped Exclusive Areas in Section 1.1.4.1.Integration with AUTOSAR stack
2023-07-04	8.0	Released
2023-06-12	7.1	• Updated the section 1.1.2.Hardware-software mapping to include Mcal_Wrapper module and removed Dem module. • Updated the Section 1.1.3.C file Structure to include Mcal_Wrapper.h and removed Dem.h. • In Section 1.1.4.1.Integration with AUTOSAR stack, the following points are modified - Instead of DEM Module, Mcal_Wrapper Section is added. - Moved Runtime Error description from DET to Mcal_Wrapper Module. • Updated the description of return type E_NOT_OK information in Section 1.3.3.17.Icu_17_TimerIp_InitCheck() API. • Updated the section 1.3.9.1.1: Software Specification Deviations for Autosar requirements. - Added the Reference "For all requirements related to Runtime errors". - Updated Description to add Mcal_Wrapper Module Information. • ASIL Level has been updated to Safety level in Section 1.3.3.Functions - APIs, 1.3.4. Notifications and Callbacks.
2022-09-26	7.0	Released
2022-09-26	6.1	Updated Icu_17_TimerIp_Init and Icu_17_TimerIp_InitCheck check sfr accesses in HSI section
2021-11-08	6.0	Released
2021-11-03	5.1	‘Mapping of hardware-software interfaces’ figure is corrected
2021-10-26	5.0	Released
2021-10-26	4.1	1.Updated description and dependency section for TimInterruptMode parameter and removed information from limitation section. 2.Config variant attribute table information is removed and added this information in 'Configuration interfaces' section.
2021-03-23	4.0	Released
2021-03-23	3.1	Added a limitation because of errata GTM_TC.H021
2020-11-27	3.0	Released

(table continues...)

Table 143 (continued) Revision History

2020-11-26	2.1	- Added accessed SFR information for all API and callback.
2020-11-18	2.0	Released
2020-10-19	1.1	- TimChannelPortPinSelect configuration parameter removed.
2020-08-14	1.0	Released
2020-08-03	0.1	- Initial Version - ICU driver chapter moved from MC-ISAR_TC3xx_UM_Basic to this document - Timeout feature introduced for GTM-TIM channels. - Incremental interface mode, external counter reset feature added. - Notification in incremental interface mode added. - Added AMDC and VSMD violation tables

Edition 2024-08-02

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