

HYUNDAI AUTOEVER

---

# AUTOSAR EcuM User Manual

DOC. NO

SCOPE OF APPLICATION All Project/Engineering  
Responsibility : Classic AUTOSAR Team

File Name EcuM\_UM.pdf

Creation JC Kim 2022/08/30

Check HM Kim 2022/08/30

Approval JH Jeong 2022/08/30

Edition Date: 2022/08/30

Document Management System

Any user/Gahyun Kim Classic AUTOSAR Team. This document contains proprietary information of HyundaiAutoEver and is not to be reproduced or duplicated without permission. Any such act could result in restrictions imposed by company rules and related laws.

## Document Change History

Date (YYYY-MM-DD)	Ver.	Editor	Chap	Description (before → after revision)
2016-04-05	2.5.3	Sanghoon Bae		<ul style="list-style-type: none"> <li>EcuM UM separated</li> </ul>
2016-04-29	2.5.4	Sanghoon Bae	5.10	<ul style="list-style-type: none"> <li>Updated EcuMPartitionRef configuration by applying AUTOSAR 4.1</li> </ul>
2016-07-13	2.6.0	Sanghoon Bae	4.3.3 5.8, 5.9, 5.10, 5.11 6.3.7.1 7.2.1 9.1	<ul style="list-style-type: none"> <li>Items added to the Deviations section</li> <li>Removed FlexModuleConfigurationReference</li> <li>Added DriverInitItem configuration</li> <li>Added EcuMCalloutUserIncludeFiles configuration</li> <li>Updated generator error messages</li> <li>Caveat on calling EcuM_SetWakeupEvent added</li> <li>Caveat on using ICU GPT Wakeup added</li> </ul>
2016-08-29	2.6.2	Sanghoon Bae	5.3 6.3.7 9.1	<ul style="list-style-type: none"> <li>Updated description on OS Spinlock Reference configuration</li> <li>Moved caveats on EcuM_SetWakeupEvent to the design caveats section in Appendix</li> </ul>
2016-10-07	2.6.3	Sanghoon Bae	5.8 5.9 9.1.1 9.3	<ul style="list-style-type: none"> <li>Updated info on Driver Init Item configuration</li> <li>Added info on how to add a module to be initialized</li> <li>Caveat on implementing Wakeup Interrupt added</li> </ul>
2016-11-15	2.7.0	Sanghoon Bae	5.2 5.4 5.7 5.6 5.14 7.2 9.1.1	<ul style="list-style-type: none"> <li>Updated Configuration Guide</li> <li>Caveat on configuring EcuM Sleep Mode added</li> <li>Added generator error message description</li> <li>Updated changes on configuring initialization items</li> </ul>
2016-11-24	2.7.1	Sanghoon Bae	4.3	<ul style="list-style-type: none"> <li>Change Logs updated</li> </ul>
2017-03-24	2.7.2	Sanghoon Bae	5.6 5.7 6.3.2.6 9.1.1 9.1.2 9.3.2 9.3.3	<ul style="list-style-type: none"> <li>Updated Configuration Guide</li> <li>Added description on abnormal shutdowns</li> <li>Updated description on adding modules to be initialized</li> <li>Added info on how to add a wakeup source</li> <li>Updated caveats on design</li> </ul>
2017-05-18	2.7.3	Sanghoon Bae	4.3	<ul style="list-style-type: none"> <li>Change Logs updated</li> </ul>
2017-07-20	2.7.4	Sanghoon Bae	9.2	<ul style="list-style-type: none"> <li>Added a guide on ICU Wakeup configuration</li> <li>Added a guide on GPT Wakeup configuration</li> <li>Updated use case-specific configuration guides</li> </ul>
2017-10-18	2.8.0	Sanghoon Bae	5.2 5.7 9.1.2 9.2.4 9.2.5	<ul style="list-style-type: none"> <li>Updated description on ComM Channel Reference configuration when configuring a wakeup source</li> <li>Added description on Chorus MCU Low Power</li> </ul>
2017-12-14	2.8.1	Sanghoon Bae	5.2 9.2.5	<ul style="list-style-type: none"> <li>Revised description on the EcuM LP callout option</li> <li>Inserted info on RH850 Deepstop into the LP Callout guide</li> </ul>
2018-04-17	2.9.0	Sanghoon Bae	4.3.3 6.3 7.2.1 9.2.4 9.2.5	<ul style="list-style-type: none"> <li>Added API call implementation to the Deviations section</li> <li>Removed Multicore-related generator error message</li> <li>Revised the Chorus guide</li> <li>Added description on new API to manage EcuM states</li> </ul>

				<ul style="list-style-type: none"> <li>Added a guide on Cypress LP callout</li> </ul>
2018-06-19	2.10.0	Sanghoon Bae	5.3 7.2.1 9.2.4 9.2.6 8	<ul style="list-style-type: none"> <li>Reflected changes on Os Event Ref configuration</li> <li>Included Os Event Ref-related generator message</li> <li>Caveat on using ADC in Chorus Standby Mode added</li> <li>Added a guide and caveats on the issue of not being able to control PAD when entering Chorus Standby</li> <li>SWP Error Code items added</li> </ul>
2018-11-28	2.10.1	Manje Woo	4.3.2 5.1.1 7.2.1 9.2.8	<ul style="list-style-type: none"> <li>Added limitations of Watchdog in Sleep state</li> <li>Removed MCU Type configuration</li> <li>Removed MCU Type error message and added GptChannelTickMax error message</li> <li>Added watchdog handling in Sleep state</li> </ul>
2019-07-30	3.0.0	Manje Woo	4.3.2 4.3.3 5.2 5.3 5.4 5.5 7.1 7.2.1	<ul style="list-style-type: none"> <li>Eliminated description on Multi-Core ECUs not supporting Sleep</li> <li>Added Multi-Core Sleep to the Deviations section</li> <li>EcuMFlexGeneral: Added Loop Count Max</li> <li>EcuMConfiguration: Added prefixes</li> <li>EcuMCommonConfiguration: Added description on the changed shutdown synchronization configuration</li> <li>EcuMDefaultShutdownTarget: Changed Default State and Default Sleep Mode Ref values</li> <li>Generator Option: Eliminated EcuMConfPrefix</li> <li>Error Messages: Eliminated ERR010024</li> </ul>
2019-10-28	3.0.0.0	Manje Woo	4.3.3 5	<ul style="list-style-type: none"> <li>Added EcuMNormalMcuModeRef to the non-supported list</li> <li>Configuration category changed</li> </ul>
2020-02-13	3.0.1.0	Manje Woo	5.4 8.2 9.2.4 9.2.7	<ul style="list-style-type: none"> <li>More detailed description on Os Task Ref</li> <li>Det error added</li> <li>Updated description on EcuM_L2HTransition_Callout_App</li> <li>Added a guide on periodic Wakeup/Sleep without returning to RUN Mode</li> </ul>
2020-03-19	3.0.2.0	Manje Woo	4.3.1	<ul style="list-style-type: none"> <li>Generator modified for RTE R44</li> </ul>
2020-11-30	3.0.3.0	Manje Woo	9.2.5 9.2.6	<ul style="list-style-type: none"> <li>MCU series name used instead of MCU vendor name</li> <li>Added a “guide to using MCU LowPower Transition Callout (General)”</li> <li>Moved the previous 9.2.6 and 9.2.7 to under 9.2.6</li> </ul>
2021-01-08	3.0.4.0	Manje Woo	4.3.1	<ul style="list-style-type: none"> <li>Edited code for RTE R44 and applied static verification results</li> </ul>
2021-08-17	3.0.5.0	Junho Cho	9.2.6 All	<ul style="list-style-type: none"> <li>Added description on Lowpower Guide</li> <li>Revised templates</li> </ul>
2021-09-01	3.0.6.0	Junho Cho	4.3	<ul style="list-style-type: none"> <li>Change Logs updated</li> </ul>
2021-11-23	3.1.0.0	Junho Cho	4.3	<ul style="list-style-type: none"> <li>Change Logs updated</li> </ul>
2021-12-17	3.1.1.0	Junho Cho	4.3	<ul style="list-style-type: none"> <li>Change Logs updated</li> </ul>

2022-06-24	3.1.2.0	Joochan Kim	4.3 5.9 9.1.1 9.3	<ul style="list-style-type: none"> <li>• Change Logs updated</li> <li>• Added EcuModuleUsed option</li> <li>• Updated configuration path when add Module reference</li> <li>• Moved some contents from SWP Technical Review</li> </ul>
2022-08-03	3.1.3.0	Joochan Kim	4.3 9.3.6	<ul style="list-style-type: none"> <li>• Change Logs updated</li> <li>• Added Design Notes</li> </ul>
2022-08-30	3.1.4.0	Joochan Kim	4.3	<ul style="list-style-type: none"> <li>• Change Logs updated</li> </ul>

# Table of Contents

1. OVERVIEW .....	2 -
2. REFERENCE.....	2 -
3. AUTOSAR SYSTEM .....	3 -
3.1 MODE MANGEMENT STACK .....	3 -
3.2 ECUM MODULE .....	3 -
4. PRODUCT RELEASE NOTES.....	3 -
4.1 OVERVIEW .....	3 -
4.2 SCOPE OF THE RELEASE .....	3 -
4.3 MODULE RELEASE NOTES .....	3 -
4.3.1 Change Logs.....	3 -
4.3.2 Limitations .....	19 -
4.3.3 Deviations .....	20 -
5. CONFIGURATION GUIDE.....	22 -
5.1 ECUMGENERAL .....	22 -
5.2 ECUMFLEXGENERAL .....	25 -
5.3 ECUMCONFIGURATION .....	26 -
5.4 ECUMCOMMONCONFIGURATION.....	27 -
5.5 ECUMDEFAULTSHUTDOWNTARGET .....	28 -
5.6 ECUMDEMEVENTPARAMETERREFS .....	28 -
5.7 ECUMSLEEPMODE.....	28 -
5.8 ECUMWAKEUPSOURCE .....	29 -
5.9 ECUMDRIVERINITLISTZERO .....	29 -
5.10 ECUMDRIVERINITLISTONE .....	30 -
5.11 ECUMCALLOUTUSERINCLUDEFILES .....	30 -
5.12 ECUMFLEXCONFIGURATION.....	31 -
5.13 ECUMFLEXUSERCONFIG .....	31 -
5.14 ECUMALARMCLOCK.....	31 -
5.15 ECUMSETCLOCKALLOWEDUSERS .....	32 -
5.16 ECUMGoDownALLOWEDUSERS .....	32 -
5.17 ECUMRESETMODE.....	32 -
5.18 ECUMSHUTDOWNCAUSE.....	32 -
6. APPLICATION PROGRAMMING INTERFACE (API).....	32 -
6.1 TYPE DEFINITIONS .....	32 -
6.2 MACRO CONSTANTS.....	35 -
6.3 FUNCTIONS.....	35 -
6.3.1 General .....	35 -
6.3.2 Initialization and Shutdown Sequences .....	36 -
6.3.3 Shutdown Management.....	40 -
6.3.4 Wakeup Handling .....	43 -
6.3.5 Miscellaneous .....	44 -

6.3.6	Scheduled Functions .....	- 46 -
6.3.7	Callbacks from Wakeup Sources .....	- 46 -
6.3.8	Generic Callouts.....	- 47 -
6.3.9	Callouts from the Startup Phase .....	- 48 -
6.3.10	Callouts from the Shutdown Phase .....	- 50 -
6.3.11	Callouts from the Sleep Phase .....	- 52 -
6.3.12	Callouts from the Up Phase .....	- 55 -
6.3.13	Alarm Clock .....	- 57 -
<b>7.</b>	<b>GENERATOR .....</b>	<b>- 60 -</b>
7.1	GENERATOR OPTION .....	- 60 -
7.2	GENERATOR MESSAGE .....	- 60 -
7.2.1	Error Messages .....	- 60 -
7.2.2	Warning Messages .....	- 69 -
7.2.3	Information Messages .....	- 72 -
<b>8.</b>	<b>SWP ERROR CODE .....</b>	<b>- 74 -</b>
8.1	DEM ERROR.....	- 74 -
8.1.1	ECUM_E_CONFIGURATION_DATA_INCONSISTENT .....	- 74 -
8.1.2	ECUM_E_IMPROPER_CALLER .....	- 74 -
8.2	DET ERROR.....	- 74 -
<b>9.</b>	<b>APPENDIX.....</b>	<b>- 76 -</b>
9.1	FUNCTION-SPECIFIC CONFIGURATION GUIDE.....	- 76 -
9.1.1	How to add a module to be initialized .....	- 76 -
9.1.2	How to add a wakeup source .....	- 78 -
9.2	USE CASE-SPECIFIC CONFIGURATION GUIDE.....	- 79 -
9.2.1	A guide to using GPT Timer Wakeup .....	- 79 -
9.2.2	A guide to using ICU Wakeup .....	- 82 -
9.2.3	Things to note when implementing Wakeup ISR .....	- 85 -
9.2.4	A guide to using Chorus MCU LowPower Transition Callout.....	- 86 -
9.2.5	A guide to writing S6J3xxx MCU LowPower Callout.....	- 88 -
9.2.6	A guide to using MCU LowPower Transition Callout (General).....	- 91 -
9.2.7	Watchdog handling in Sleep state .....	- 97 -
9.3	DESIGN NOTES .....	- 98 -
9.3.1	Cancellation prohibited while executing Off / Reset command of controller .....	- 98 -
9.3.2	Application Task before completion of SWP initialization .....	- 98 -
9.3.3	Integration Code Check .....	- 98 -
9.3.4	Prohibit duplicate calls to App Mode Active during Wakeup .....	- 99 -
9.3.5	Prohibit implementation of Interrupt logic that uses both High Power / Low Power inside AppMode_WakeupEventValidated .....	- 99 -
9.3.6	Write ICU ISR for registered Wakeup Source (Chorus Only) .....	- 99 -

## 1. Overview

This document provides references and guidance for users on parameter configuration and system design in the Hyundai AutoEver AUTOSAR EcuM module.

The document has been created based on the AUTOSAR standard SRS/SWS. For more detailed functional description, please refer to the reference documents below.

The following terms on configuration categories mean:

- Changeable (C) : Items that can be configured by user
- Fixed (F) : Items that cannot be changed by user
- NotSupported(N) : Items that are not used

## 2. Reference

Sl. No.	Title	Version
1	AUTOSAR BSW Service API Guide.doc	1.0.0
2	AUTOSAR_SWS_ECUStateManager.pdf	3.0.0
3	AUTOSAR_EXP_ModemanagementGuide.pdf	1.0.0

## 3. AUTOSAR System

### 3.1 Mode Management Stack

Hyundai AutoEver AUTOSAR's Mode Management Stack consists of the EcuM module, which manages ECU states, and the BswM module, which serves as a BSW mode manager.

### 3.2 EcuM Module

The ECU Manager module manages ECU states (Wakeup, Sleep, Shutdown, etc.) and performs the following functions:

- Initializes and de-initializes the OS, the SchM, and the BswM as well as some basic software modules based on ECU states.
- Configures the MCU for Sleep and Shutdown when requested
- Manages all wakeup events on the ECU.
- Decides to change ECU states when requested by SWCs or other BswM modules that use ECU states
- Notify the BswM after changing ECU states

## 4. Product Release Notes

### 4.1 Overview

This chapter is intended to provide the release information on the Hyundai AutoEverEcuM module, describing the features and restrictions of different versions of the EcuM software product.

### 4.2 Scope of the Release

All content in this document applies only to the following Hyundai AutoEverEcuM module.

Module name	AUTOSAR version	SWS version	Module version
EcuM	4.0.3	3.0.0	3.1.4

※ The module version refers to the SW version of the BswModule description (Bswmd) file of each module.

### 4.3 Module Release Notes

#### 4.3.1 Change Logs

- Version 3.1.4.0
  - Improvement



## ■ Code improvements to comply with the UNECE Cyber Security regulations

Rationale	Violation of UNECE Cyber Security regulations occurred
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

### ➤ Version 3.1.3.0

#### - Improvement

## ■ Improved to sort input file comments in generated files

Rationale	Even though there is no change, a change point is created in the file as the corresponding comment part is changed.
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ■ Code improvements to comply with the UNECE Cyber Security regulations

Rationale	Violation of UNECE Cyber Security regulations occurred
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

### ➤ Version 3.1.2.0

#### - New feature

## ■ Added ModuleUsed option to set whether to initialize the module to EcuMDriverInitListZero, EcuMDriverInitListOne, and EcuMDriverRestartList

Rationale	As the use of common use of the Hyundai AUTOSAR Classic platform (mobilgene Classic) increases, for user convenience, a setting that allows to determine whether to initialize the module is provided.
Impact on Behavior	True : The module is added to the InitList and initialization proceeds (existing behavior)  False : The module is excluded from InitList and initialization is not

	performed.
Impact on Settings	Same as above
Required ASW actions	Default Value : true ModuleUsed option can be changed to false if it is necessary to exclude a module at the discretion of the user

## - Improvement

### ■ Added EcuM\_ErrorHook call condition to support Dem unused SWP

Rationale	In the SWP without the dem module, it does not have to consider that the dem writes to the damaged NVRAM block.
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 3.1.1.0

## - Improvement

### ■ Code improvements to comply with the UNECE Cyber Security regulations

Rationale	Customer requirement
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 3.1.0.0

## - New feature

### ■ Post-build development

Rationale	Customer requirement
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 3.0.6.0

### - Improvement

#### ■ Changed to a 64-bit generator

Rationale	Need to solve the Out of Memory issue
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 3.0.5.0

### - Improvements

#### ■ Robustness of EcuM code improved

Rationale	Code robustness improved through EcuM race condition analysis
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

#### ■ Added Lowerpower Guide

Rationale	User convenience
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 3.0.4.0

### - Improvements

#### ■ Edited code for RTE R44

Rationale	Need to support compatibility with RTE R44 (AUTOSAR 4.4.0-based RTE) - Need to add an input parameter for SchM_Init()
-----------	--

Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ■ Addressed/justified static analysis violations

Rationale	Need to reflect x86 test results
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 3.0.3.0

### - Improvements

## ■ Addressed/justified static analysis violations

Rationale	Changed criteria for static analysis
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ■ User Manual updated

Rationale	Need to add info on Low Power Transition Callout (see 9.2.6)
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 3.0.2.0

### - Improvement

## ■ Generator modified for RTE R44

Rationale	Need to support compatibility with RTE R44 (AUTOSAR 4.4.0-based RTE)
-----------	--

	<ul style="list-style-type: none"> <li>- IncludedDatatTypeSets needed for Swcd_Bsw_EcuM.arxml</li> <li>- bswDistinguishedPartition needed for Bswmd_EcuM.arxml</li> </ul>
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 3.0.1.0

### - Improvements

#### ■ Large ARXM file support in the generator

Rationale	Need to use a ARXML file of over 100MB as generator input
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

#### ■ User Manual updated

Rationale	<p>Need to add Det errors that could occur in EcuM</p> <p>Need to reflect changes to EcuM_L2HTransition_Callout_App</p> <p>Description on “Os Task Ref” in EcuMCommonConfiguration is not clear.</p> <p>Need to add a guide on transition to Low Power after a periodic wakeup without reverting back to RUN Mode</p>
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 3.0.0.0

### - Improvement

#### ■ Changed configuration item properties to make code available

Rationale	Need to change configuration item properties along with code disclosure
Impact on Behavior	None

Impact on Settings	None
Required ASW actions	None

## ➤ Version 3.0.0

### - New feature

#### ■ Support for transition to Sleep in Multi-Core MCU

Rationale	Multi-Core MCU needs to be put to sleep
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

### - Improvements

#### ■ Changed the maximum number of cores supported in Multi-Core (3 -> 255)

Rationale	Unable to respond to MCUs with more than 4 cores as 3 is the maximum
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

#### ■ Fixed error caused by unnecessary SetEvent() during Multi-Core Shutdown

Rationale	E_OS_STATE error occurred during Multi-Core Shutdown
Impact on Behavior	None
Impact on Settings	Need to remove EcuMOsTaskRef and EcuMOsEventRef from EcuM>EcuMConfiguration>EcuMCommonConfiguration
Required ASW actions	None

#### ■ Halt Sequence refactoring

Rationale	Not easy to read as it consists of many conditional statements and some checks are redundant
-----------	--

Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ■ T1 Integration code split off into integration\_EcuM

Rationale	Need to eliminate the EcuM module's dependence on the T1 tool
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ■ EcuM\_AL\_SetProgrammableInterrupts() callout enabled in a Multi-Core environment

Rationale	Need to align with Single-Core
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ■ Added configuration items, including EcuMWakeupSource, for which symbol prefixes can be set

Rationale	Need to maintain consistency with the changed MCAL of F1KM
Impact on Behavior	None
Impact on Settings	<p>The added items are as follows:</p> <p>EcuM&gt;EcuMConfiguration&gt;EcuMPrefixSleepMode</p> <p>EcuM&gt;EcuMConfiguration&gt;EcuMPrefixWakeupSource</p> <p>EcuM&gt;EcuMConfiguration&gt;EcuMPrefixAlarmClock</p> <p>EcuM&gt;EcuMConfiguration&gt;EcuMPrefixFlexUserConfig</p> <p>EcuM&gt;EcuMConfiguration&gt;EcuMPrefixResetMode</p> <p>EcuM&gt;EcuMConfiguration&gt;EcuMPrefixShutdownCause</p>
Required ASW actions	None

## ■ Fixed the default state of EcuMDefaultShutdownTarget as EcuMStateSleep

Rationale	Need to support CAN Remote Wakeup for Sleep non-supported MCUs
Impact on Behavior	None
Impact on Settings	EcuM>EcuMConfiguration>EcuMCommonConfiguration>EcuMDefaultShutdownTarget should be set to EcuMStateSleep.
Required ASW actions	None

## ➤ Version 2.10.1

### - Improvements

#### ■ Added info on Watchdog handling in Sleep state

Rationale	Need to provide info on Watchdog handling in the Sleep state
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

#### ■ Removed EcuMMcuType configuration

Rationale	Need to remove configuration items that are not in use
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 2.10.0

### - Improvements

#### ■ A reset to occur after a timeout when PLL is not locked after MCU initialization

Rationale	There is an issue of getting trapped in a deadlock when PLL is not locked.
Impact on Behavior	A reset will occur when PLL is not locked for a certain amount of time after MCU initialization.
Impact on Settings	None
Required ASW actions	None



## ■ Added SWP error information

Rationale	Added SWP error information
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ■ Caveats on using ADC in Chorus Standby Mode added

Rationale	Caveats on using ADC in Chorus Standby Mode added
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 2.9.0

### - New features

## ■ Added a method that does not involve Task Activation when AppMode request is made.

Rationale	When a Sleep request is made after entering No COM, OS Limit occurs due to Multiple Task Activation in MCU without PM.
Impact on Behavior	Fg3 Task Activation doesn't work when the new method is used.
Impact on Settings	BswM Harmonization needs to be performed again when the new method is used.
Required ASW actions	Need to change the existing App Mode after reviewing BswM UM and Odin Manual

## ■ T1 Integration support

Rationale	Support needed for adding T1 Integration
Impact on Behavior	None
Impact on Settings	None

Required ASW actions	None
----------------------	------

## ■ Created a Cypress User Callout guide

Rationale	Cypress Pm.c to be managed by splitting into SWP Code and User Callout Code
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	Need to move what was previously implemented by the user in Pm.c to LP Callout after reviewing the guide

## - Improvement

## ■ Correction made on the Chorus ICU ISR guide

Rationale	Some errors found in the Chorus ICU ISR guide
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	Write ICU ISR after reviewing the revised guide

## ➤ Version 2.8.1

## - New feature

## ■ N/A

## - Improvement

## ■ Created a guide to using RH850 Deepstop

Rationale	Created a guide to using RH850 Deepstop
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	Implement LP after reviewing the UM implementation guide.

## ➤ Version 2.8.0

## - New feature

## ■ Developed Sleep Sequence for an environment for new MCUs (Chorus, RH850 Deepstop)

Rationale	Developed Sleep Sequence for an environment where the existing Pm structure cannot be used
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	Implement LP after reviewing the UM implementation guide.

## - Improvement

### ■ Enabled to change the ComM Channel Reference of a wakeup source

Rationale	User can now change the communication-related parameter of a wakeup source.
Impact on Behavior	None
Impact on Settings	User should configure ComM Reference when adding communication-related wakeup sources
Required ASW actions	None

## ➤ Version 2.7.4

### - New feature

#### ■ N/A

### - Improvement

#### ■ Added use case-specific configuration guides

Rationale	Created guides on how to implement and configure ICU/GPT Wakeup
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 2.7.3

### - New feature

#### ■ N/A

### - Improvement

#### ■ More robust fail-save 'Enable Interrupt' when entering Lowpower

Rationale	More robust fail-save 'Enable Interrupt' when entering Lowpower
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 2.7.2

### - New feature

- N/A

### - Improvements

- Enabled to change EcuMWakeupSource

Rationale	There are cases where wakeup sources are registered during application design.
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	User should register wakeup sources if additionally required.

- Addressed a library configuration issue

Rationale	A definition library that can be changed by configuration is used.
Impact on Behavior	None
Impact on Settings	No need to redistribute the library, even with configuration changes
Required ASW actions	None

- Added API prototypes for some Integration code

Rationale	Compile warning is issued as some of the integration APIs are not declared as prototype.
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

- Fixed compile warning

Rationale	Fixed the issue of compile warning occurring during build
-----------	---

Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

- A reset, instead of OFF, to occur if something fails after shutdown

Rationale	May wait in the OFF state if something fails after shutdown Hook
Impact on Behavior	SW reset will occur in case of a malfunction.
Impact on Settings	None
Required ASW actions	None

## ➤ Version 2.7.1

### - New feature

- N/A

### - Improvements

- Restored Ecu Mode

Rationale	Restored Ecu Mode in case where other BSW modules might use it, as well as for backward compatibility
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

## ➤ Version 2.7.0

### - New feature

- N/A

### - Improvements

- Fixed the issue where generation continues even with duplicate IDs in some configurations

Rationale	Duplicate ID values are not checked in some configurations during generation.
-----------	---

Impact on Behavior	None
Impact on Settings	Need to change IDs if duplicate values are found
Required ASW actions	None

■ Error to be thrown when a wakeup source is not configured in Sleep Mode

Rationale	Generation works properly once a wakeup source has been registered despite not being configured in Sleep Mode.
Impact on Behavior	None
Impact on Settings	Ensure wakeup sources are configured in Sleep Mode
Required ASW actions	None

■ Made changes to the header generation rule and mandatory containers for DriverInitItem generation

Rationale	The existing header generation rule cannot accommodate some ECUs.
Impact on Behavior	None
Impact on Settings	- Modified the header generation rule - Removed Module Ref from mandatory containers (modules without Ecuc can be registered)
Required ASW actions	None

■ Removed modes that are not used in ECU Mode Declaration Group

Rationale	Removed ECU modes that are no longer in use after ODIN 2016b
Impact on Behavior	None
Impact on Settings	Need to change configuration if there is an additional ECU mode to be used
Required ASW actions	None

■ Error to be thrown when EcuM\_SetWakeupEvent is not called by Wakeup ISR

Rationale	Need to address a situation where EcuM_SetWakeupEvent is not called by Wakeup ISR
Impact on Behavior	Error to be thrown when EcuM_SetWakeupEvent is not called during wakeup
Impact on Settings	None
Required ASW actions	Ensure EcuM_SetWakeupEvent is not called by Wake-Up ISR

- New feature

- N/A

- Improvements

- Fixed the issue where a wakeup ISR that occurred in a particular section of entering Sleep is not processed

Rationale	An ISR that occurred at a certain point while entering Sleep Sequence was not processed and entered Sleep.
Impact on Behavior	Stricter conditions for entering Sleep
Impact on Settings	None
Required ASW actions	None

- Fixed ODIN/SystemDesk validation error

Rationale	<ul style="list-style-type: none"> <li>- There are elements that are not supported in PDF.</li> <li>- Swcd is not generated according to System Desk Schema.</li> </ul>
Impact on Behavior	None
Impact on Settings	None
Required ASW actions	None

- EcuMDriverInitItemIndex configuration enabled

Rationale	While it is possible for the user to configure EcuMDriverInitItemIndex, it's a fixed container.
Impact on Behavior	None
Impact on Settings	EcuMDriverInitItemIndex configuration enabled
Required ASW actions	None

➤ Version 2.6.2

- Optimized EcuM\_ClearWakeupEvent execution time
- Fixed ECUM\_SLAVE\_CORE\_MASK generation error when Multi-Core is in use

➤ Version 2.6.1

Removed ECU Modes that are not in use

➤ Version 2.6.0

- Changed configurations and code to be generated for Driver Init List items by proactively

applying AUTOSAR 4.2.1

- Accommodated additional vendor-specific actions required for MCU initialization by

generating the Mcu Post init function

- Wakeup-related callout generation enabled
- Tidied up sections within code
- Version 2.5.5
  - Corrected logic to fix Get OS resource - Release OS resource mismatches
- Version 2.5.4
  - Updated EcuMPartitionRef by applying AUTOSAR 4.1
  - EcuM\_WakeupRestartSeq call location changed
  - Port interface dedicated to Swcd segregated
- Version 2.5.3
  - Precompile option that affects library code modified
- Version 2.5.2
  - Changed the parameter type of EcuM\_GetResetReason API
- Version 2.5.1
  - MPC574x and Cypress supported
- Version 2.5.0
  - Wakeup-related bug fixed
  - EcuM\_GetResetReason API added
- Version 2.4.2
  - Internal update without any feature
- Version 2.4.1
  - Static analysis results reflected
- Version 2.4.0
  - Internal update without any feature
- Version 2.3.2
  - Added parameter 'PmEnabled'
  - Change made to allow Initialization of other modules only in the main core

## 4.3.2 Limitations

- Only EcuMFlex is supported while EcuMFixed is not.
  - List of configurations not supported

Container	Parameter
EcuMFixedConfiguration	EcuMNvramReadallTimeout EcuMNvramWriteallTimeout EcuMRunMinimumDuration EcuMFixedModuleConfigurationRef EcuMComMCommunicationAllowedList EcuMNormalMcuModeRef
EcuMDriverInitListThree	



EcuMDriverInitListTwo	
EcuMFixedUserConfig	EcuMFixedUser
EcuMTTII	EcuMDivisor EcuMSleepModeRef
EcuMShutdownTarget	EcuMShutdownTargetId
EcuMFixedGeneral	EcuMIncludeComM EcuMIncludeNvM EcuMIncludeNvramMgr EcuMTTIIEnabled EcuMTTIIWakeupSourceRef

## - List of APIs not supported

EcuM Fixed APIs
Std_ReturnTypeEcuM_RequestRUN (EcuM_UserType user)
Std_ReturnTypeEcuM_ReleaseRUN (EcuM_UserType user)
Std_ReturnTypeEcuM_RequestPOST_RUN (EcuM_UserType user)
Std_ReturnTypeEcuM_ReleasePOST_RUN (EcuM_UserType user)
void EcuM_KillAllIRUNRequests (void)
Std_ReturnTypeEcuM_GetState (EcuM_StateType* state)
EcuM_WakeupStatusTypeEcuM_GetStatusOfWakeupSource (EcuM_WakeupSourceType sources)
void EcuM_CB_NfyNvMJobEnd (uint8 Serviceld, NvM_RequestResultTypeJobResult)
void EcuM_AL_DriverInitTwo (const EcuM_ConfigType* ConfigPtr)
void EcuM_AL_DriverInitThree (const EcuM_ConfigType* ConfigPtr)
void EcuM_OnRTESStartup (void)
void EcuM_OnEnterRun (void)
void EcuM_OnExitRun (void)
void EcuM_OnExitPostRun (void)
void EcuM_OnPrepShutdown (void)
void EcuM_OnGoSleep (void)
EcuM_WakeupReactionTypeEcuM_OnWakeupReaction (EcuM_WakeupReactionTypewact)

- RAM Hash
  - RAM hash generation before entering Halt Sleep and RAM hash checking after releasing Halt Sleep are not supported.
- Polling Sleep
  - Polling Sleep via EcuM\_GoPoll() API is not supported.
- Watchdog handling in Sleep state
  - To use Watchdog in the Sleep state, it needs to be triggered by the user. See Appendix 9.2.7.

### 4.3.3 Deviations

- In configuring EcuMDriverInitListZero/One, only the modules that were configured in EcuM could be added to the list, but with AUTOSAR R4.2.1 applied proactively, users can now add any module of their choice to the list of modules to be initialized. In the process, FlexModuleConfigurationReference has been removed.
- In order to use API calls, instead of task activation, when requesting AppMode, ECU Mode

Handling and some of the relevant APIs from AUTOSAR R4.3.0 have been adopted in advance.

- To use Sleep in Multi-Core MCUs, the MultiCore SHUTDOWN Phase and SLEEP Phase contained in AUTOSAR R4.4.0 have been adopted in advance.
- EcuMFlexConfiguration and EcuMNormalMcuModeRef are not to be configured, as there is no need for a MCU mode switch after wakeup from sleep.

## 5. Configuration Guide

The EcuM configuration for the AUTOSAR platform distributed by Hyundai AutoEver reflects the mode management policy of Hyundai AutoEver and therefore any changes require consultation with Hyundai AutoEver.

### 5.1 EcuMGeneral

EcuMGeneral defines general configuration parameters regarding EcuM code generation.

Parameter Name	Value	Category
Dev Error Detect <sup>1)</sup>	true	C
Include Det	true	C
Version Info Api	false	C
Main Function Period	0.01(sec)	C

#### 1) Dev Error Detect

- Whether or not to use the error reporting feature via Det\_ReportError
- The list of errors that can be reported when the feature is enabled is as follows:

Apild	ErrorId	Description
ECUM_GETVERSIONINFO_SID (0x00)	ECUM_E_NULL_POINTER (0x12)	The output pointer parameter value of the API EcuM_GetVersionInfo is NULL_PTR.
ECUM_INIT_SID (0x01)	ECUM_E_NULL_POINTER (0x12)	The global variable that stores a configuration pointer of each module is NULL_PTR.
ECUM_GODOWN_SID (0x1F)	ECUM_E_UNINIT (0x10)	The API EcuM_GoDown was called prior to EcuM module initialization.
	ECUM_E_SHUTDOWN_FAILED (0x19)	In the case of Multicore, the value of EcuM_GucShutdownMask is 0.
ECUM_SHUTDOWN_SID (0x02)	ECUM_E_UNINIT (0x10)	The API EcuM_Shutdown was called prior to EcuM module initialization.
ECUM_SELECTSHUTDOWNTARGET_SID (0x06)	ECUM_E_UNINIT (0x10)	The API EcuM_SelectShutdownTarget was called before EcuM module initialization.
	ECUM_E_STATE_PAR_OUT_OF_RANGE (0x16)	A target, passed as an input parameter to the API EcuM_SelectShutdownTarget, was none of ECUM_STATE_SLEEP / RESET / OFF.
	ECUM_E_INVALID_PAR (0x13)	A target, passed as an input parameter to the API EcuM_SelectShutdownTarget, was either one of ECUM_STATE_SLEEP or RESET, but an invalid mode was passed.
ECUM_GETSHUTDOWNTARGET_SID (0x09)	ECUM_E_UNINIT (0x10)	The API EcuM_GetShutdownTarget was called prior to EcuM module initialization.
	ECUM_E_NULL_POINTER (0x12)	The output pointer parameter value of the API EcuM_GetShutdownTarget is NULL_PTR.
ECUM_GETLASTSHUTDOWNTARGET_SID (0x08)	ECUM_E_UNINIT (0x10)	The API EcuM_GetLastShutdownTarget was called prior to EcuM module initialization.

	ECUM_E_NULL_POINTER (0x12)	The output pointer parameter value of the API EcuM_GetLastShutdownTarget is NULL_PTR.
ECUM_SELECTSHUTDOWNCAUSE_SID (0x1B)	ECUM_E_UNINIT (0x10)	The API EcuM_SelectShutdownCause was called prior to EcuM module initialization.
	ECUM_E_INVALID_PAR (0x13)	An invalid cause was passed as an input parameter to the API EcuM_SelectShutdownCause.
ECUM_GETSHUTDOWNCAUSE_SID (0x1C)	ECUM_E_UNINIT (0x10)	The API EcuM_GetShutdownCause was called prior to EcuM module initialization.
	ECUM_E_NULL_POINTER (0x12)	The output pointer parameter value of the API EcuM_GetShutdownCause is NULL_PTR.
ECUM_GETMOSTRECENTSHUTDOWN_SID (0x1D)	ECUM_E_UNINIT (0x10)	The API EcuM_GetMostRecentShutdown was called before EcuM module initialization.
	ECUM_E_NULL_POINTER (0x12)	The output pointer parameter value of the API EcuM_GetMostRecentShutdown is NULL_PTR.
ECUM_GETNEXTRECENTSHUTDOWN_SID (0x1E)	ECUM_E_UNINIT (0x10)	The API EcuM_GetNextRecentShutdown was called prior to EcuM module initialization.
	ECUM_E_NULL_POINTER (0x12)	The output pointer parameter value of the API EcuM_GetNextRecentShutdown is NULL_PTR.
ECUM_GETPENDINGWAKEUP_EVENTS_SID (0x0d)	ECUM_E_UNINIT (0x10)	The API EcuM_GetPendingWakeupEvents was called before EcuM module initialization.
ECUM_CLEARWAKEUPEVENT_SID (0x16)	ECUM_E_UNINIT (0x10)	The API EcuM_ClearWakeupEvent was called before EcuM module initialization.
	ECUM_E_UNKNOWN_WAKEUP_SOURCE (0x17)	An unknown wakeup source was passed as an input parameter to the API EcuM_ClearWakeupEvent.
ECUM_GETVALIDATEDWAKEUPEVENTS_SID (0x15)	ECUM_E_UNINIT (0x10)	The API EcuM_GetValidatedWakeupEvents was called before EcuM module initialization.
ECUM_GETEXPIREDWAKEUP_EVENTS_SID (0x19)	ECUM_E_UNINIT (0x10)	The API EcuM_GetExpiredWakeupEvents was called before EcuM module initialization.
ECUM_SELECTBOOTTARGET_SID (0x12)	ECUM_E_UNINIT (0x10)	The API EcuM_SelectBootTarget was called before EcuM module initialization.
	ECUM_E_STATE_PAR_OUT_OF_RANGE (0x16)	A target, passed as an input parameter to the API EcuM_SelectBootTarget, is none of eCUM_BOOT_TARGET_APP / OEM_BOOTLOADER / SYS_BOOTLOADER.

ECUM_GETBOOTTARGET_SID (0x13)	ECUM_E_UNINIT (0x10)	The API EcuM_GetBootTarget was called before EcuM module initialization.
	ECUM_E_NULL_POINTER (0x12)	The output pointer parameter value of the API EcuM_GetBootTarget is NULL_PTR.
ECUM_GOHALT_SID (0x20)	ECUM_E_UNINIT (0x10)	The API EcuM_GoHalt was called prior to EcuM module initialization.
	ECUM_E_SHUTDOWN_FAILED (0x19)	In the case of Multicore, the value of EcuM_GucHaltMask is not 0.
	ECUM_E_UNKNOWN_WAKEUP_SOURCE (0x17)	A wakeup was not caused by the wakeup source set in Halt Sleep.
ECUM_GOPOLL_SID (0x21)	ECUM_E_UNINIT (0x10)	The API EcuM_GoPoll was called before EcuM module initialization.
	ECUM_E_SHUTDOWN_FAILED (0x19)	In the case of Multicore, the value of EcuM_GucPollMask is not 0.
ECUM_MAINFUNCTION_SID (0x18)	ECUM_E_UNINIT (0x10)	The API EcuM_MainFunction was called before EcuM module initialization.
ECUM_SETRELWAKEUPALARM_SID (0x22)	ECUM_E_UNINIT (0x10)	The API EcuM_SetRelWakeupAlarm was called before EcuM module initialization.
	ECUM_E_INVALID_PAR (0x13)	An invalid alarm clock user was passed as an input parameter to the API EcuM_SetRelWakeupAlarm.
	ECUM_E_SERVICE_DISABLED (0x11)	The API EcuM_SetRelWakeupAlarm was called when AlarmClockPresent = false is set.
ECUM_SETABSWAKEUPALARM_SID (0x23)	ECUM_E_UNINIT (0x10)	The API EcuM_SetAbsWakeupAlarm was called before EcuM module initialization.
	ECUM_E_INVALID_PAR (0x13)	An invalid alarm clock user was passed as an input parameter to the API EcuM_SetAbsWakeupAlarm.
	ECUM_E_SERVICE_DISABLED (0x11)	The API EcuM_SetAbsWakeupAlarm was called when AlarmClockPresent = false is set.
ECUM_ABORTWAKEUPALARM_SID (0x24)	ECUM_E_UNINIT (0x10)	The API EcuM_AbortWakeupAlarm was called before EcuM module initialization.
	ECUM_E_INVALID_PAR (0x13)	An invalid alarm clock user was passed as an input parameter to the API EcuM_AbortWakeupAlarm.
	ECUM_E_SERVICE_DISABLED (0x11)	The API EcuM_AbortWakeupAlarm was called when AlarmClockPresent = false is set.
ECUM_GETCURRENTTIME_SID (0x25)	ECUM_E_UNINIT (0x10)	The API EcuM_GetCurrentTime was called before EcuM module initialization.
	ECUM_E_NULL_POINTER (0x12)	The output pointer parameter value of the API EcuM_GetCurrentTime is NULL_PTR.
	ECUM_E_SERVICE_DISABLED (0x11)	The API EcuM_GetCurrentTime was called when AlarmClockPresent = false is set.
ECUM_GETWAKEUPTIME_SID (0x26)	ECUM_E_UNINIT (0x10)	The API EcuM_GetWakeupTime was called before EcuM module initialization.

	ECUM_E_NULL_POINTER (0x12)	The output pointer parameter value of the API EcuM_GetWakeupTime is NULL_PTR.
	ECUM_E_SERVICE_DISABLED (0x11)	The API EcuM_GetWakeupTime was called when AlarmClockPresent = false is set.
ECUM_SETCLOCK_SID (0x27)	ECUM_E_UNINIT (0x10)	The API EcuM_SetClock was called before EcuM module initialization.
	ECUM_E_INVALID_PAR (0x13)	An invalid SetClockAllowedUser was passed as an input parameter to the API EcuM_AbortWakeupAlarm.
	ECUM_E_SERVICE_DISABLED (0x11)	The API EcuM_SetClock was called when AlarmClockPresent = false is set.
ECUM_SETWAKEUPEVENT_SID (0x0c)	ECUM_E_UNINIT (0x10)	The API EcuM_SetWakeupEvent was called before EcuM module initialization.
	ECUM_E_UNKNOWN_WAKEUP_SOURCE (0x17)	An unknown wakeup source was passed as an input parameter to the API EcuM_SetWakeupEvent.
ECUM_VALIDATEWAKEUPEVENT_SID (0x14)	ECUM_E_UNINIT (0x10)	The API EcuM_ValidateWakeupEvent was called before EcuM module initialization.
	ECUM_E_UNKNOWN_WAKEUP_SOURCE (0x17)	An unknown wakeup source was passed as an input parameter to the API EcuM_ValidateWakeupEvent.
ECUM_GETRESETREASON_SID (0x28)	ECUM_E_UNINIT (0x10)	The API EcuM_GetResetReason was called before EcuM module initialization.

- 2) Include Det
- 3) Version Info Api
- 4) Main Function Period

## 5.2 EcuMFlexGeneral

Parameter Name	Value	Categor
Enable Def Behaviour <sup>1)</sup>	false	C
Reset Loop Detection <sup>2)</sup>		C
Alarm Clock Present <sup>3)</sup>	true/false	C
Alarm Wakeup Source <sup>4)</sup>	ECUM_WKSOURCE_GPT <sup>5)</sup>	C
	_6)	
Pm Enabled <sup>7)</sup>	true/false	C
Lp Transition Callout Enabled <sup>8)</sup>	true/false	C
Loop Count Max <sup>9)</sup>	2000000	C

- 1) Enable Def Behaviour
  - Whether or not to use the error event reporting feature via Dem\_ReportErrorStatus
  - See EcuMDemEventParameterRefs for the list of error events that can be reported when the feature is enabled.
- 2) Reset Loop Detection
  - To use the Reset Loop Detection feature during initialization, set this to true.
- 3) Alarm Clock Present
  - Set this to true only if Low Power Mode is used.
- 4) Alarm Wakeup Source

- Only applicable if Low Power Mode is used.
- 5) For FREESCALE MPC560x and RH850
- 6) For INFINEON TC27x
- 7) Set this to true when Sleep Mode is controlled by the PM module.
- 8) Set this to true when LowPower Transition Callout is used.
- 9) Loop Count Max
  - The maximum loop count as a failsafe to prevent EcuM from being stuck in an infinite loop
  - Used for waiting until PLL is locked and Multicore shutdown synchronization
  - If the set value is too small, the loop will end before enough time has passed and a reset or Det error will occur.
  - Note: The wait time when the loop count max is set to 2000000 (the actual wait time can vary depending on MCU type, MCAL version, compile options, etc.)

MCU	Clock Type	Frequency	PLL Lock Wait Time	Note
MPC560xB	FIRC	16MHz	17.333s	
SPC58xx	IRCOSC	16MHz	20.941s	
RH850 F1L	HS IntOSC	8MHz	15.125s	
RH850 F1K	HS IntOSC	8MHz	15.082s	
RH850 F1KM	HS IntOSC	8MHz	30.231s	
TC2xx	SRI Clock	100MHz	0.947s	
TC3xx	SRI Clock	100MHz	2.712s	
S32K14x	FIRC	48MHz	N/A	Reset when incomplete within 8ms of initial Wdg timeout

MCU	Clock Type	Frequency	Multi-Core Shutdown Wait Time	Note
SPC58xx	PLL	180MHz	156.644ms	
TC2xx	PLL	200MHz	137.648ms	
TC3xx	PLL	300MHz	194.968ms	

## 5.3 EcuMConfiguration

Parameter Name	Value	Categor
Prefix Sleep Mode <sup>1)</sup>	-	C
Prefix Wakeup Source <sup>2)</sup>	-	C
Prefix Alarm Clock <sup>3)</sup>	-	C
Prefix Flex User Config <sup>4)</sup>	-	C
Prefix Reset Mode <sup>5)</sup>	-	C
Prefix Shutdown Cause <sup>6)</sup>	-	C

- 1) Prefix Sleep Mode
  - Set a prefix string for generating a symbol of what was configured in EcuMSleepMode.
- 2) Prefix Wakeup Source
  - Set a prefix string for generating a symbol of what was configured in EcuMWakeupSource.
- 3) Prefix Alarm Clock
  - Set a prefix string for generating a symbol of what was configured in EcuMAlarmClock.
- 4) Prefix Flex User Config
  - Set a prefix string for generating a symbol of what was configured in EcuMFlexUserConfig.
- 5) Prefix Reset Mode

- Set a prefix string for generating a symbol of what was configured in EcuMResetMode.
- 6) Prefix Shutdown Cause
  - Set a prefix string for generating a symbol of what was configured in EcuMShutdownCause.

## 5.4 EcuMCommonConfiguration

Parameter Name	Value	Categor
Config Consistency Hash <sup>1)</sup>	123456789	C
Os Task Ref <sup>2)</sup> (Vendor specific)		C
Default App Mode <sup>3)</sup>		C
OS Resource <sup>4)</sup>		C
Os Spinlock Ref <sup>5)</sup> (Vendor specific)		C
Os Event Ref <sup>6)</sup> (Vendor specific)		C

- 1) Config Consistency Hash
  - This is used for checking the hash value to ensure the consistency of the configuration.
- 2) Os Task Ref
  - Applicable only for Multicore shutdown synchronization
  - If configured: Shutdown synchronization is performed using OsSetEvent.
    - i. OsTask to request Shutdown (EcuM\_GoDown, EcuM\_GoHalt, EcuM\_GoPoll calls) should be configured.
    - ii. Os Events configured in Os Event Ref (6) should be referenced by EventRef of this OsTask configuration.
    - iii. The OsTask configuration should reference all Os Applications for slave cores in AccessingApplication.
  - If not configured: EcuM internally performs shutdown synchronization using a spinlock and flag.
- 3) Default App Mode
  - Input parameter of StartOS() to be called during initialization
- 4) OS Resource
  - Input parameter of GetResource() / ReleaseResource() when going into Sleep
  - The goal is to lock or unlock the Scheduler in Sleep states.
  - In the case of Multicore, each core should have one standard resource as OsResource, and Ref configuration should be done in the order in which cores are numbered.
  - No need for configuration if a task that performs a transition to Sleep is tied to tasks that are to be locked as internal resource
- 5) Os Spinlock Ref
  - In the case of Multicore, this parameter must be set for shutdown synchronization.
  - When configuring the OsSpinlock, all Os Applications allocated to cores should be referenced in AccessingApplication.
- 6) Os Event Ref
  - Applicable only for Multicore shutdown synchronization
  - If configured, the number of Os Events should be equal to the number of all slave cores, and Os Events should be referenced by OS Task configured in Os Task Ref (2).
  - If configured, shutdown synchronization is performed using OsSetEvent. If not configured, shutdown synchronization is performed using a spinlock and flag.



## 5.5 EcuMDefaultShutdownTarget

Parameter Name	Value	Categor
Default State <sup>1)</sup>	EcuMStateSleep	C
Default Reset Mode Ref <sup>2)</sup>	ECUM_RESET_MCU	C
Default Sleep Mode Ref <sup>3)</sup>	ECUM_SLEEP_STOP <sup>4)</sup>	C
	ECUM_SLEEP_STBY <sup>5)</sup>	
	ECUM_SLEEP_VLPS <sup>6)</sup>	

- 1) Default State
  - Default value for the selected shutdown target after system boot
  - The value is maintained until it is changed via EcuM\_SelectShutdownTarget()
- 2) Default Reset Mode Ref
  - Default value for Reset Mode when Default State = Reset
  - This parameter must be set as it will be used for Off Sequence even if the default state is not Reset.
- 3) Default Sleep Mode Ref
  - Default value for Sleep Mode when Default State = Sleep.
  - May be left empty if not necessary
- 4) For FREESCALE MPC560x, RENESAS RH850 F1L/F1K/F1KM, and INFINEON TC2xx/TC3xx
- 5) For ST Micro SPC58xx
- 6) For NXP S32K14x

## 5.6 EcuMDemEventParameterRefs

Parameter Name	Value	Categor
ECUM_E_IMPROPER_CALLER <sup>1)</sup>		C
ECUM_E_CONFIGURATION_DATA_INCONSISTENT <sup>2)</sup>		C
ECUM_E_RAM_CHECK_FAILED <sup>3)</sup>		C

- 1) ECUM\_E\_IMPROPER\_CALLER
  - This occurs if a user that was not configured in GoDownAllowedUser was entered for EcuM\_GoDown API calls.
- 2) ECUM\_E\_CONFIGURATION\_DATA\_INCONSISTENT
  - This occurs if the validity check of post-build configuration data failed.
- 3) ECUM\_E\_RAM\_CHECK\_FAILED
  - This occurs if the RAM hash check failed during wakeup from Halt Sleep.

## 5.7 EcuMSleepMode

Parameter Name	Value	Categor
Id <sup>1)</sup>		C
Mcu Mode Ref <sup>2)</sup>		C
Suspend <sup>3)</sup>		C
Wakeup Source Mask <sup>4)</sup>		C

- 1) Id
  - Set the relevant MCU mode ID by referring to the User Manual of the MCU module.
- 2) Mcu Mode Ref

- Reference to the relevant MCU mode
- 3) Suspend
  - Set this to true for Halt Sleep, and false for Polling Sleep.
  - If set as Halt Sleep, you can enter Sleep only via EcuM\_GoHalt().
  - If set as Polling Sleep, you can enter Sleep only via EcuM\_GoPoll().
- 4) Wakeup Source Mask
  - Set this for wakeup sources.

\* Wakeup sources (except for Id 0-4) that are configured in EcuMWakeupSource(5.8) should be configured for at least one Sleep mode.

If there is a wakeup source that is not configured for any Sleep modes, a generation error will be thrown.

## 5.8 EcuMWakeupSource

Parameter Name	Value	Categor
Id <sup>1)</sup>		C
Polling <sup>2)</sup>		C
Reset Reason Ref <sup>3)</sup>		C
Validation Timeout <sup>4)</sup>	0	C
ComM Channel Ref <sup>5)</sup>		C

- 1) Id
  - Only preassigned wakeup sources can be configured in the range of 0-4. (From AUTOSAR)
  - As for FREESCALE MPC560x, ECUM\_WKSOURCE\_GPT (5) should be set.
  - You can add more wakeup sources beyond that range.
- 2) Polling
  - You can enable a wakeup only for the wakeup sources that are set to true during Polling Sleep.
- 3) Reset Reason Ref
  - When reset by the configured reset reason, wakeup sources are treated as pending.
- 4) Validation Timeout
  - A wakeup, if occurred, is recognized as normal only when the wakeup source calls EcuM\_ValidateWakeupEvent() before the set timeout expires.
  - If not configured, calling EcuM\_ValidateWakeupEvent() is not necessary.
- 5) ComM Channel Ref
  - If the wakeup source is a communication channel, set this to reference the relevant ComMChannel.

## 5.9 EcuMDriverInitListZero

EcuMDriverInitListZero allows you to configure Initialization code for AUTOSAR standard modules. This takes place in the ECUM\_STATE\_STARTUP\_ZERO phase, and you can only configure initialization for pre-compile configuration.

Since the list of modules to be initialized is already confirmed at the time of platform distribution, you should maintain the existing list and only add to it only when there is a new module to be initialized. See 9.1.1 for how to add a new module.

Parameter Name	Value	Categor
Module ID <sup>1)</sup>		C
Module Parameter <sup>2)</sup>		C
Module Service <sup>3)</sup>		C
Index <sup>4)</sup> (Vendor specific)		C
Module Post Build Ptr <sup>5)</sup> (Vendor specific)		C
Module Ref <sup>6)</sup>		C
Module Used <sup>7)</sup>		C

- 1) Module ID
  - Name the module.
  - E.g.Mcu
- 2) Module Parameter
  - Set a parameter type for the initialization function.
  - NULL\_PTR: when using a null pointer
  - POSTBUILD\_PTR: when using a post-build pointer
  - VOID: when the function parameter is void
- 3) Module Service
  - Set an initialization function.
  - E.g. Init
- 4) Index
  - Set the order of initialization code. Execute initialization in ascending order.
- 5) Module Post Build Ptr
  - If the module parameter is set as POSTBUILD\_PTR, enter a pointer value to be used.
- 6) Module Ref
  - Set a reference for the module.
- 7) Module Used
  - Set whether to initialize the module.

## 5.10 EcuMDriverInitListOne

EcuMDriverInitListOne allows you to configure Initialization code for AUTOSAR standard modules. This takes place in the ECUM\_STATE\_STARTUP\_ONE phase, and you can configure initialization for both pre-compile and post-build configurations.

Parameter settings are the same as EcuMDriverInitListZero.

As with EcuMDriverInitListZero, the list of modules to be initialized is already confirmed at the time of platform distribution, so you should maintain the existing list and add more to it only when there is a new module to be initialized. See 9.1.1 for how to add a new module.

## 5.11 EcuMCalloutUserIncludeFiles

Parameter Name	Value	Categor
Callout User Include File <sup>1)</sup> (Vendor specific)		C

- 1) Callout User Include File
  - This is where you can configure an additional header file when configuring EcuMDriverInitListZero/One items.

## 5.12 EcuMFlexConfiguration

Any modification to the application is unnecessary.

Parameter Name	Value	Categor
Normal Mcu Mode Ref <sup>1)</sup>	-	N
Normal Mcu Clock Ref <sup>2)</sup> (Vendor specific)		C
Flex Gpt Configuration Ref <sup>3)</sup> (Vendor specific)		C
Partition Ref <sup>4)</sup> (Vendor specific)		C

- 1) Normal Mcu Mode Ref
  - Set an input parameter of Mcu\_SetMode() when waking up from Sleep.
  - This is not to be configured in the Hyundai AutoEver platform.
- 2) Normal Mcu Clock Ref
  - An input parameter of Mcu\_InitClock() that is generated along with MCU initialization (Mcu\_Init)
- 3) Flex Gpt Configuration Ref
  - Reference to the Gpt channel to be used as an alarm clock
- 4) Partition Ref
  - Applicable only for Multicore
  - Reference to all partitions to use EcuM
  - The main partition to use other BSW modules should be set as Bsw Module Execution = true in EcucPartition.

## 5.13 EcuMFlexUserConfig

EcuMFlexUserConfig allows you to set users that request a shutdown or use an alarm clock.

Parameter Name	Value	Categor
Flex User <sup>1)</sup>		C
Flex Ecuc Partition Ref <sup>2)</sup>		N

- 1) Flex User
  - This parameter is used to set a unique ID of a user. Usually for BSW modules, Module ID is used.
- 2) Flex Ecuc Partition Ref
  - This is not to be configured as it does not affect code generation.

## 5.14 EcuMAlarmClock

Parameter Name	Value	Categor
Id <sup>1)</sup>	0	C
Time Out <sup>2)</sup>	0	C
User <sup>3)</sup>	ECUM_USER_PM	C

- 1) Id
  - Set an Id of an alarm clock. If the PM module is present, only ECUM\_ALARM\_PM should be configured.
- 2) Time Out
  - The initial value of an alarm clock should be set to 0.
- 3) User
  - Only ECUM\_USER\_PM should be set as a user that can change the alarm clock.

## 5.15 EcuMSetClockAllowedUsers

Parameter Name	Value	Categor
Set Clock Allowed User Ref <sup>1)</sup>		C

- 1) Set Clock Allowed User Ref
  - Users that are allowed to call the EcuM\_SetClock() API

## 5.16 EcuMGoDownAllowedUsers

Parameter Name	Value	Categor
Go Down Allowed User Ref <sup>1)</sup>	ECUM_USER_BSWM, ECUM_USER_XCP	C

- 1) Go Down Allowed User Ref
  - This parameter denotes users that are allowed to call the EcuM\_GoDown() API and should be set as ECUM\_USER\_BSWM and ECUM\_USER\_XCP to be able to request Reset and Off.

## 5.17 EcuMResetMode

Parameter Name	Value	Categor
Id <sup>1)</sup>		C

- 1) Id
  - Configure reset modes.

## 5.18 EcuMShutdownCause

Parameter Name	Value	Categor
Id <sup>1)</sup>		C

- 1) Id
  - Configure shutdown causes.

# 6. Application Programming Interface (API)

## 6.1 Type Definitions

### 6.1.1.1 EcuM\_ConfigType

<b>Name:</b>	EcuM_ConfigType	
<b>Type:</b>	Structure	
<b>Range:</b>	-	The content of this structure depends on the post-build configuration of EcuM.
<b>Description:</b>	A pointer to such a structure shall be provided to the ECU State Manager initialization routine for configuration.	

## 6.1.1.2 EcuM\_StateType

<b>Name:</b>	EcuM_StateType		
<b>Type:</b>	Uint8		
<b>Range</b>	ECUM_STATE_STARTUP	0x10	--
	ECUM_STATE_STARTUP_ONE	0x11	--
	ECUM_STATE_STARTUP_TWO	0x12	--
	ECUM_STATE_STARTUP_THREE	0x13	--
	ECUM_STATE_WAKEUP	0x20	--
	ECUM_STATE_WAKEUP_ONE	0x21	--
	ECUM_STATE_WAKEUP_VALIDATION	0x22	--
	ECUM_STATE_WAKEUP_REACTION	0x23	--
	ECUM_STATE_WAKEUP_TWO	0x24	--
	ECUM_STATE_WAKEUP_WAKESLEEP	0x25	--
	ECUM_STATE_WAKEUP_TTII	0x26	--
	ECUM_STATE_RUN	0x30	--
	ECUM_STATE_APP_RUN	0x32	--
	ECUM_STATE_APP_POST_RUN	0x33	--
	ECUM_STATE_SHUTDOWN	0x40	--
	ECUM_STATE_PREP_SHUTDOWN	0x44	--
	ECUM_STATE_GO_SLEEP	0x49	--
	ECUM_STATE_GO_OFF_ONE	0x4d	--
	ECUM_STATE_GO_OFF_TWO	0x4e	--
	ECUM_STATE_SLEEP	0x50	--
	ECUM_STATE_OFF	0x80	--
	ECUM_STATE_RESET	0x90	--
<b>Description:</b>	ECU State Manager states.		

## 6.1.1.3 EcuM\_UserType

<b>Name:</b>	EcuM_UserType		
<b>Type:</b>	Uint8		
<b>Description:</b>	Unique value for each user.		

## 6.1.1.4 EcuM\_WakeupSourceType

<b>Name:</b>	EcuM_WakeupSourceType		
<b>Type:</b>	Uint32		
<b>Range:</b>	ECUM_WKSOURCE_POWER	0x00000001	Power cycle (bit 0)
	ECUM_WKSOURCE_RESET(default)	0x00000002	Hardware reset (bit 1). If hardware cannot distinguish between a power cycle and a reset reason, then this shall be the default wakeup source.
	ECUM_WKSOURCE_INTERNAL_RESET	0x00000004	Internal reset of $\mu$ C (bit 2) The internal reset typically only resets the $\mu$ C core but not

			peripherals or memory controllers. The exact behavior is hardware specific. This source may also indicate an unhandled exception.
	ECUM_WKSOURCE_INTERNAL_WDG	0x00000008	Reset by internal watchdog (bit 3)
	ECUM_WKSOURCE_EXTERNAL_WDG	0x00000010	Reset by external watchdog (bit 4), if detection supported by hardware
	ECUM_WKSOURCE_GPT	0x00000020	Gpt
	ECUM_WKSOURCE_CAN1RX	0x00000040	Can1Rx
	ECUM_WKSOURCE_CAN1RX_POLL	0x00000080	Can1Rx Polling
	ECUM_WKSOURCE_LIN0RX	0x00000100	Lin0Rx
<b>Description:</b>	EcuM_WakeupSourceType defines a bitfield with 5 pre-defined positions (see Range). The bitfield provides one bit for each wakeup source. In WAKEUP, all bits cleared indicates that no wakeup source is known. In STARTUP, all bits cleared indicates that no reason for restart or reset is known. In this case, ECUM_WKSOURCE_RESET shall be assumed.		

## 6.1.1.5 EcuM\_WakeupStateType

<b>Name:</b>	EcuM_WakeupStateType		
<b>Type:</b>	Uint8		
<b>Range:</b>	ECUM_WKSTATUS_NONE	0	No pending wakeup event was detected
	ECUM_WKSTATUS_PENDING	1	The wakeup event was detected but not yet validated
	ECUM_WKSTATUS_VALIDATED	2	The wakeup event is valid
	ECUM_WKSTATUS_EXPIRED	3	The wakeup event has not been validated and has expired therefore
	ECUM_WKSTATUS_DISABLED	4	The wakeup source is disabled and does not detect wakeup events.
	ECUM_WKSTATUS_ENABLED	5	The wakeup source is enabled
<b>Description:</b>	The type describes the possible states of a wakeup source.		

## 6.1.1.6 EcuM\_BootTargetType

<b>Name:</b>	EcuM_BootTargetType		
<b>Type:</b>	Uint8		
<b>Range:</b>	ECUM_BOOT_TARGET_APP	0	The ECU will boot into the application
	ECUM_BOOT_TARGET_OEM_BOOTLOADER	1	The ECU will boot into the OEM bootloader
	ECUM_BOOT_TARGET_SYS_BOOTLOADER	2	The ECU will boot into the system supplier bootloader
<b>Description:</b>	This type represents the boot targets the ECU Manager module can be configured with. The default boot target is ECUM_BOOT_TARGET_OEM_BOOTLOADER.		

## 6.1.1.7 EcuM\_ResetType

<b>Name:</b>	EcuM_ResetType		
<b>Type:</b>	Uint8		
<b>Range:</b>	ECUM_RESET_MCU	0	Microcontroller reset via Mcu_PerformReset
	ECUM_RESET_WDG	1	Watchdog reset via WdgM_PerformReset
	ECUM_RESET_IO	2	Reset by toggling an I/O line.
<b>Description:</b>	This type describes the reset mechanisms supported by the ECU State Manager. It can be extended by configuration.		

## 6.1.1.8 EcuM\_ShutdownCauseType

<b>Name:</b>	EcuM_ShutdownCauseType		
<b>Type:</b>	Uint8		
<b>Range:</b>	ECUM_CAUSE_UNKNOWN	0	No cause was set.
	ECUM_CAUSE_ECU_STATE	1	ECU state machine entered a state for shutdown
	ECUM_CAUSE_WDGM	2	Watchdog Manager detected a failure
	ECUM_CAUSE_DCM	3	Diagnostic Communication Manager requests a shutdown due to a service request
<b>Description:</b>	This type describes the reset mechanisms supported by the ECU State Manager. It can be extended by configuration.		

## 6.2 Macro Constants

NONE

## 6.3 Functions

### 6.3.1 General

#### 6.3.1.1 EcuM\_GetVersionInfo

<b>Function Name</b>	EcuM_GetVersionInfo
<b>Syntax:</b>	FUNC(void, ECUM_CODE) EcuM_GetVersionInfo (P2VAR(Std_VersionInfoType, AUTOMATIC, ECUM_APPL_DATA)versioninfo)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	versioninfo
<b>Return Value</b>	None
<b>Description</b>	EcuM_GetVersionInfo returns the version information of this module.
<b>Preconditions</b>	ECUM_VERSION_INFO_API should be configured as 'TRUE'
<b>Configuration Dependency</b>	This API is available only if configuration parameter EcuMVersionInfoApi is set to true.
<b>In Communication with application SW-C</b>	Rte_Call_<P>_GetVersionInfo(Std_VersionInfoType* versioninfo) <P> : R-Port Name



## 6.3.2 Initialization and Shutdown Sequences

### 6.3.2.1 EcuM\_Init

<b>Function Name</b>	EcuM_Init
<b>Syntax:</b>	FUNC(void, ECUM_CODE) EcuM_Init(void)
<b>Service ID</b>	0x01
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	EcuM_Init Initializes the ECU state manager and carries out the startup procedure. The function will never return (it calls StartOS).
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

### 6.3.2.2 EcuM\_StartupTwo

<b>Function Name</b>	EcuM_StartupTwo
<b>Syntax:</b>	FUNC(void, ECUM_CODE) EcuM_StartupTwo(void)
<b>Service ID</b>	0x1a
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	In this state, the initialization of BSW modules which needs OS support is carried out.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

### 6.3.2.3 EcuM\_GoHalt

<b>Function Name</b>	EcuM_GoHalt
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_GoHalt(void)
<b>Service ID</b>	0x20
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	None

<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	Instructs the ECU State Manager module to go into a sleep mode where the microcontroller is halted, depending on the selected shutdown target.
<b>Preconditions</b>	The ECU state manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	Rte_Call_<P>_GoHalt(void) <P> : R-Port Name

## 6.3.2.4 EcuM\_GoPoll

<b>Function Name</b>	EcuM_GoPoll
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_GoPoll(void)
<b>Service ID</b>	0x21
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	Instructs the ECU State Manager module to go into a polling mode depending on the selected shutdown target.
<b>Preconditions</b>	The ECU state manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	Rte_Call_<P>_GoPoll(void) <P> : R-Port Name

## 6.3.2.5 EcuM\_GoDown

<b>Function Name</b>	EcuM_GoDown
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_GoDown(uint16 caller)
<b>Service ID</b>	0x1f
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	caller
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	This Service instructs the ECU State Manager module to perform a power off or a reset depending on the selected shutdown target.
<b>Preconditions</b>	The ECU state manager must be initialized.
<b>Configuration Dependency</b>	This API is available only to users configured to EcuMGoDownAllowedUsers.
<b>In Communication with application SW-C</b>	Rte_Call_<P>_GoDown(uint16 caller) <P> : R-Port Name

## 6.3.2.6 EcuM\_Shutdown

<b>Function Name</b>	EcuM_Shutdown
<b>Syntax:</b>	FUNC(void, ECUM_CODE) EcuM_Shutdown(void)
<b>Service ID</b>	0x02
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	EcuM_Shutdown typically called from the shutdown hook, this function takes over execution control and will carry out GO OFF II activities.
<b>Preconditions</b>	The ECU state manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

※ Upon entering Shutdown due to irregular events, DET will be generated and SW reset.

## 6.3.2.7 EcuM\_RequestRUN

<b>Function Name</b>	EcuM_RequestRUN
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_RequestRUN (EcuM_UserType user)
<b>Service ID</b>	0x03
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	user
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	Places a request for the RUN state. Requests can be placed by every user made known to the state manager at configuration time.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.2.8 EcuM\_ReleaseRUN

<b>Function Name</b>	EcuM_ReleaseRUN
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_ReleaseRUN (EcuM_UserType user)
<b>Service ID</b>	0x04
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant

<b>Parameters (In)</b>	user
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	Releases a RUN request previously done with a call to EcuM_RequestRUN. The service is intended for implementing AUTOSAR ports.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.3 Shutdown Management

### 6.3.3.1 EcuM\_SelectShutdownTarget

<b>Function Name</b>	EcuM_SelectShutdownTarget
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_SelectShutdownTarget (EcuM_StateType target, uint8 mode)
<b>Service ID</b>	0x06
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	target, mode
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	EcuM_SelectShutdownTarget selects the shutdown target. EcuM_SelectShutdownTarget is part of the ECU Manager Module port interface.
<b>Preconditions</b>	The ECU state manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	Rte_Call_<P>_SelectShutdownTarget(EcuM_StateType target, uint8 mode) <P> : R-Port Name

### 6.3.3.2 EcuM\_GetShutdownTarget

<b>Function Name</b>	EcuM_GetShutdownTarget
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_GetShutdownTarget (P2VAR(EcuM_StateType, AUTOMATIC, ECUM_APPL_DATA) target, P2VAR(uint8, AUTOMATIC, ECUM_APPL_DATA) mode)
<b>Service ID</b>	0x09
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	target, mode
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	This function returns always the selected shutdown target as set by EcuM_SelectShutdownTarget.
<b>Preconditions</b>	The ECU state manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	Rte_Call_<P>_GetShutdownTarget(EcuM_StateType* target, uint8* mode) <P> : R-Port Name

### 6.3.3.3 EcuM\_GetLastShutdownTarget

<b>Function Name</b>	EcuM_GetLastShutdownTarget
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_GetLastShutdownTarget (P2VAR(EcuM_StateType, AUTOMATIC, ECUM_APPL_DATA) target, P2VAR(uint8, AUTOMATIC, ECUM_APPL_DATA) mode)
<b>Service ID</b>	0x08

<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	target, mode
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	This routine is intended for primary use in STARTUP or RUN state. The return value describes the ECU state from which the last wakeup or power up occurred. This function shall return always the same value until the next shutdown
<b>Preconditions</b>	The ECU state manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	Rte_Call_<P>_GetLastShutdownTarget (EcuM_StateType* target, uint8* mode) <P> : R-Port Name

## 6.3.3.4 EcuM\_SelectShutdownCause

<b>Function Name</b>	EcuM_SelectShutdownCause
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_SelectShutdownCause(EcuM_ShutdownCauseType cause)
<b>Service ID</b>	0x1b
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	cause
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	This routine is intended for primary use in STARTUP or RUN state. The return value describes the ECU state from which the last wakeup or power up occurred. This function shall return always the same value until the next shutdown.
<b>Preconditions</b>	The ECU state manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	Rte_Call_<P>_SelectShutdownCause (EcuM_ShutdownCauseType cause) <P> : R-Port Name

## 6.3.3.5 EcuM\_GetShutdownCause

<b>Function Name</b>	EcuM_GetShutdownCause
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_GetShutdownCause (P2VAR(EcuM_ShutdownCauseType, AUTOMATIC, ECUM_APPL_DATA) cause)
<b>Service ID</b>	0x1c
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	cause
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	This routine is intended for primary use in STARTUP or RUN state. The return value describes the ECU state from which the last wakeup or power up

	occurred. This function shall return always the same value until the next shutdown
<b>Preconditions</b>	The ECU state manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	Rte_Call_<P>_GetShutdownCause (EcuM_ShutdownCauseType* cause) <P> : R-Port Name

## 6.3.3.6 EcuM\_GetMostRecentShutdown

<b>Function Name</b>	EcuM_GetMostRecentShutdown
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_GetMostRecentShutdown (P2VAR(EcuM_StateType, AUTOMATIC, ECUM_APPL_DATA)target, P2VAR(uint8, AUTOMATIC, ECUM_APPL_DATA)mode, P2VAR(EcuM_ShutdownCauseType, AUTOMATIC, ECUM_APPL_DATA)cause, P2VAR(uint32, AUTOMATIC, ECUM_APPL_DATA)time)
<b>Service ID</b>	0x1d
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	Target, Mode, cause, time
<b>Return Value</b>	Std_ReturnType E_OK
<b>Description</b>	EcuM_GetMostRecentShutdown returns information about the most recent shutdown operation. EcuM_GetMostRecentShutdown is part of the ECU Manager Module port interface.
<b>Preconditions</b>	The ECU state manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	Rte_Call_<P>_GetMostRecentShutdown (EcuM_StateType* target, uint8* mode, EcuM_ShutdownCauseType* cause, uint32* time) <P> : R-Port Name

## 6.3.3.7 EcuM\_GetNextRecentShutdown

<b>Function Name</b>	EcuM_GetNextRecentShutdown
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_GetNextRecentShutdown (P2VAR(EcuM_StateType, AUTOMATIC, ECUM_APPL_DATA)target, P2VAR(uint8, AUTOMATIC, ECUM_APPL_DATA)mode, P2VAR(EcuM_ShutdownCauseType, AUTOMATIC, ECUM_APPL_DATA)cause, P2VAR(uint32, AUTOMATIC, ECUM_APPL_DATA)time)
<b>Service ID</b>	0x1e
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	Target, Mode, cause, time
<b>Return Value</b>	Std_ReturnType E_OK
<b>Description</b>	EcuM_GetNextRecentShutdown returns information about the next recent shutdown operation. All stored shutdown information can be read by first calling

	EcuM_GetMostRecentShutdown and then looping over EcuM_GetNextRecentShutdown until an error is returned.
<b>Preconditions</b>	The ECU state manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	Rte_Call_<P>_GetNextRecentShutdown (EcuM_StateType* target, uint8* mode, EcuM_ShutdownCauseType* cause, uint32* time) <P> : R-Port Name

## 6.3.4 Wakeup Handling

### 6.3.4.1 EcuM\_GetPendingWakeupEvents

<b>Function Name</b>	EcuM_GetPendingWakeupEvents
<b>Syntax:</b>	FUNC(EcuM_WakeupSourceType, ECUM_CODE) EcuM_GetPendingWakeupEvents(void)
<b>Service ID</b>	0x0d
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant, Non-Interruptible
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	EcuM_WakeupSourceType
<b>Description</b>	This routine is responsible for returning the wakeup events, which have been set but not yet validated.
<b>Preconditions</b>	The ECU state Manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

### 6.3.4.2 EcuM\_ClearWakeupEvent

<b>Function Name</b>	EcuM_ClearWakeupEvent
<b>Syntax:</b>	FUNC(void, ECUM_CODE) EcuM_ClearWakeupEvent (EcuM_WakeupSourceType sources)
<b>Service ID</b>	0x16
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant, Non-Interruptible
<b>Parameters (In)</b>	Sources
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This routine is responsible for clearing all wake up events like pending, validated and expired events.
<b>Preconditions</b>	The ECU state Manager must be initialized
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None



## 6.3.4.3 EcuM\_GetValidatedWakeupEvents

<b>Function Name</b>	EcuM_GetValidatedWakeupEvents
<b>Syntax:</b>	FUNC(EcuM_WakeupSourceType, ECUM_CODE) EcuM_GetValidatedWakeupEvents(void)
<b>Service ID</b>	0x15
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant, Non-Interruptible
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	EcuM_WakeupSourceType
<b>Description</b>	This routine is responsible for returning all the wakeup events that have passed the validation.
<b>Preconditions</b>	The ECU state Manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.4.4 EcuM\_GetExpiredWakeupEvents

<b>Function Name</b>	EcuM_GetExpiredWakeupEvents
<b>Syntax:</b>	FUNC(EcuM_WakeupSourceType, ECUM_CODE) EcuM_GetExpiredWakeupEvents(void)
<b>Service ID</b>	0x19
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant, Non-Interruptible
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	EcuM_WakeupSourceType
<b>Description</b>	This routine is responsible for returning all the wakeup events that have been set but failed the validation. Events that do not need validation must never be reported by this function.
<b>Preconditions</b>	The ECU state Manager must be initialized
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.5 Miscellaneous

### 6.3.5.1 EcuM\_SelectBootTarget

<b>Function Name</b>	EcuM_SelectBootTarget
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_SelectBootTarget (EcuM_BootTargetType target)
<b>Service ID</b>	0x12
<b>Sync/Async</b>	Synchronous

<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	target
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	EcuM_SelectBootTarget selects a boot target. EcuM_SelectBootTarget is part of the ECU Manager Module port interface.
<b>Preconditions</b>	The ECU state manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	Rte_Call_<P>_SelectBootTarget (EcuM_BootTargetType target) <P> : R-Port Name

## 6.3.5.2 EcuM\_GetBootTarget

<b>Function Name</b>	EcuM_GetBootTarget
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_GetBootTarget (P2VAR(EcuM_BootTargetType, AUTOMATIC, ECUM_APPL_DATA) target)
<b>Service ID</b>	0x13
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	target
<b>Return Value</b>	Std_ReturnType (E_OK)
<b>Description</b>	EcuM_GetBootTarget returns the current boot target. EcuM_GetBootTarget is part of the ECU Manager Module port interface.
<b>Preconditions</b>	The ECU state manager must be initialized.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	Rte_Call_<P>_GetBootTarget (EcuM_BootTargetType* target) <P> : R-Port Name

## 6.3.5.3 EcuM\_GetResetReason

<b>Function Name</b>	EcuM_GetResetReason
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_GetResetReason(P2VAR(Mcu_ResetType, AUTOMATIC, ECUM_APPL_DATA) resetreason)
<b>Service ID</b>	0x28
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	Resetreason
<b>Return Value</b>	Std_ReturnType (E_OK)
<b>Description</b>	EcuM_GetResetReason returns the current value of the EcuMResetReason.
<b>Preconditions</b>	The ECU state manager must be initialized.

<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.6 Scheduled Functions

### 6.3.6.1 EcuM\_MainFunction

<b>Function Name</b>	EcuM_MainFunction
<b>Syntax:</b>	FUNC(void, ECUM_CODE) EcuM_MainFunction(void)
<b>Service ID</b>	0x18
<b>Sync/Async</b>	None
<b>Reentrancy</b>	None
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This Service is to implement all activities of ECU State manager while the OS is up and running.
<b>Preconditions</b>	Startup I must be completed.
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.7 Callbacks from Wakeup Sources

### 6.3.7.1 EcuM\_SetWakeupEvent

<b>Function Name</b>	EcuM_SetWakeupEvent
<b>Syntax:</b>	FUNC(void, ECUM_CODE) EcuM_SetWakeupEvent(EcuM_WakeupSourceType sources)
<b>Service ID</b>	0x0C
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant, Non-Interruptible
<b>Parameters (In)</b>	sources
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This routine is responsible for setting up off wakeup event. It takes the value and stores in the internal variable. This function must start the wakeup validation timeout timer.
<b>Preconditions</b>	The ECU state Manager must be initialized
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.7.2 EcuM\_ValidateWakeupEvent

<b>Function Name</b>	EcuM_ValidateWakeupEvent
<b>Syntax:</b>	FUNC(void, ECUM_CODE) EcuM_ValidateWakeupEvent (EcuM_WakeupSourceType sources)
<b>Service ID</b>	0x14
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	sources
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	EcuM_ValidateWakeupEvent() routine is responsible for validating all the WAKEUP events The validation shall be valid when ANDing the parameter events with the internal variable of pending wakeup events results in a value other than null.
<b>Preconditions</b>	The ECU state Manager must be initialized
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.8 Generic Callouts

### 6.3.8.1 EcuM\_ErrorHook

<b>Function Name</b>	EcuM_ErrorHook
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_ErrorHook(Std_ReturnType reason)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	reason
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	In unrecoverable error situations, the ECU State Manager will call this callout. It is up the system integrator to react accordingly (reset, halt, restart, safe state etc.)
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

### 6.3.8.2 EcuM\_LoopDetection

<b>Function Name</b>	EcuM_LoopDetection
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_LoopDetection(void)
<b>Service ID</b>	None
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	Void

<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	In unrecoverable error situations, the ECU State Manager will call this callout. It is up to the system integrator to react accordingly (reset, halt, restart, safe state etc.)
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.9 Callouts from the Startup Phase

### 6.3.9.1 EcuM\_AL\_SetProgrammableInterrupts

<b>Function Name</b>	EcuM_AL_SetProgrammableInterrupts
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_AL_SetProgrammableInterrupts(void)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This call allows the system designer to notify that the GO OFF I state is about to be entered.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

### 6.3.9.2 EcuM\_AL\_DriverInitZero

<b>Function Name</b>	EcuM_AL_DriverInitZero
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_AL_DriverInitZero(void)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This callout shall provide driver initialization and other hardware-related startup activities for loading the post-build configuration data. Note: Here only pre-compile and link-time configurable modules may be used.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None

<i>In Communication with application SW-C</i>	None
---	------

## 6.3.9.3 EcuM\_DeterminePbConfiguration

<b>Function Name</b>	EcuM_DeterminePbConfiguration
<b>Syntax:</b>	FUNC(P2CONST(EcuM_ConfigType, ECUM_CONST, ECUM_CONST), ECUM_CALLOUT_CODE) EcuM_DeterminePbConfiguration(void)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	EcuM_ConfigType*
<b>Description</b>	This callout should evaluate some condition, like port pin or NVRAM value, to determine which post-build configuration shall be used in the remainder of the startup process. It shall load this configuration data into a piece of memory that is accessible by all BSW modules and shall return a pointer to the EcuM post-build configuration as a base for all BSW module post-build configurations.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.9.4 EcuM\_AL\_DriverInitOne

<b>Function Name</b>	EcuM_AL_DriverInitOne
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_AL_DriverInitOne (P2CONST(EcuM_ConfigType, AUTOMATIC, ECUM_APPL_CONST) ConfigPtr)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	ConfigPtr
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This callout shall provide driver initialization and other hardware-related startup activities in case of a power on reset
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.10 Callouts from the Shutdown Phase

### 6.3.10.1 EcuM\_OnGoOffOne

<b>Function Name</b>	EcuM_OnGoOffOne
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_OnGoOffOne(void)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This call allows the system designer to notify that the GO OFF I state is about to be entered.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

### 6.3.10.2 EcuM\_OnGoOffTwo

<b>Function Name</b>	EcuM_OnGoOffTwo
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_OnGoOffTwo(void)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This call allows the system designer to notify that the GO OFF II state is about to be entered.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

### 6.3.10.3 EcuM\_AL\_SwitchOff

<b>Function Name</b>	EcuM_AL_SwitchOff
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_AL_SwitchOff(void)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None

<b>Return Value</b>	None
<b>Description</b>	This callout shall take the code for shutting off the power supply of the ECU. If the ECU cannot un power itself, a reset may be an adequate reaction.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.10.4 EcuM\_AL\_Reset

<b>Function Name</b>	EcuM_AL_Reset
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_AL_Reset(EcuM_ResetType reset)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	reset
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This callout shall take the code for resetting the ECU.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.10.5 EcuM\_RequestReset

<b>Function Name</b>	EcuM_RequestReset
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CALLOUT_CODE) EcuM_RequestReset (EcuM_UserType user)
<b>Service ID</b>	None
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	user
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	Integration Code For Request Reset using Release Run API
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.10.6 EcuM\_RequestOff

<b>Function Name</b>	EcuM_RequestOff
----------------------	-----------------



<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CALLOUT_CODE) EcuM_RequestOff (EcuM_UserType user)
<b>Service ID</b>	None
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	user
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	Integration Code For Request Off using Release Run API
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.11 Callouts from the Sleep Phase

### 6.3.11.1 EcuM\_EnableWakeupSources

<b>Function Name</b>	EcuM_EnableWakeupSources
<b>Syntax:</b>	FUNC(void, ECUM_CODE) EcuM_EnableWakeupSources (EcuM_WakeupSourceType wakeupSource)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	wakeupSource
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	The ECU Manager Module calls EcuM_EnableWakeupSource to allow the system designer to notify wakeup sources defined in the wakeupSource bitfield that SLEEP will be entered and to adjust their source accordingly.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

### 6.3.11.2 EcuM\_GenerateRamHash

<b>Function Name</b>	EcuM_GenerateRamHash
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_GenerateRamHash(void)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None

<b>Return Value</b>	None
<b>Description</b>	Used just before Putting the ECU physically to sleep
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.11.3 EcuM\_SleepActivity

<b>Function Name</b>	EcuM_SleepActivity
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_SleepActivity(void)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This callout is invoked periodically in all reduced clock sleep modes. It is explicitly allowed to poll wakeup sources from this callout and to call wakeup notification functions to indicate the end of the sleep state to the ECU State Manager.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.11.4 EcuM\_CheckWakeup

<b>Function Name</b>	EcuM_CheckWakeup
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_CheckWakeup (EcuM_WakeupSourceType wakeupSource)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	wakeupSource
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This callout is called by the EcuM to poll a wakeupsource. It shall also be called by the ISR of a wakeup source to set up the PLL and check other wakeup sources that may be connected to the same interrupt.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.11.5 EcuM\_CheckRamHash

<b>Function Name</b>	EcuM_CheckRamHash
<b>Syntax:</b>	FUNC(uint8, ECUM_CALLOUT_CODE) EcuM_CheckRamHash(void)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This callout is intended to provide a RAM integrity test.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.11.6 EcuM\_DisableWakeupSources

<b>Function Name</b>	EcuM_DisableWakeupSources
<b>Syntax:</b>	FUNC(void, ECUM_CODE) EcuM_DisableWakeupSources (EcuM_WakeupSourceType wakeupSource)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	wakeupSource
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	The callout shall set up the given wakeup source(s) so that they are not able to wake up the ECU.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.11.7 EcuM\_AL\_DriverRestart

<b>Function Name</b>	EcuM_AL_DriverRestart
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_AL_DriverRestart (P2CONST (EcuM_ConfigType, AUTOMATIC, ECUM_APPL_CONST) ConfigPtr)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	ConfigPtr
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None

<b>Description</b>	This callout shall provide driver initialization and other hardware-related startup activities in case of a power on reset.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.11.8 EcuM\_SetMode

<b>Function Name</b>	EcuM_SetMode
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_SetMode (Mcu_ModeTypeMcuMode))
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	McuMode
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This callout shall provide set MCU mode for sleep sequence
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.11.9 EcuM\_RequestSleep

<b>Function Name</b>	EcuM_RequestSleep
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CALLOUT_CODE) EcuM_RequestSleep (EcuM_UserType user)
<b>Service ID</b>	None
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Reentrant
<b>Parameters (In)</b>	user
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	Std_ReturnType (E_OK or E_NOT_OK)
<b>Description</b>	Integration Code For Request Sleep using Release Run API
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.12 Callouts from the Up Phase

## 6.3.12.1 EcuM\_StartWakeupSources

<b>Function Name</b>	EcuM_StartWakeupSources
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_StartWakeupSources (EcuM_WakeupSourceTypewakeupSource)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	wakeupSource
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	The callout shall start the given wakeup source(s) so that they are ready to perform wakeup validation.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.12.2 EcuM\_CheckValidation

<b>Function Name</b>	EcuM_CheckValidation
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_CheckValidation (EcuM_WakeupSourceTypewakeupSource)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	wakeupSource
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This callout is called by the EcuM to validate a wakeup source. If a valid wakeup has been detected, it shall be reported to EcuMviaEcuM_ValidateWakeupEvent().
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.12.3 EcuM\_StopWakeupSources

<b>Function Name</b>	EcuM_StopWakeupSources
<b>Syntax:</b>	FUNC(void, ECUM_CALLOUT_CODE) EcuM_StopWakeupSources (EcuM_WakeupSourceTypewakeupSource)
<b>Service ID</b>	0x00
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non-Reentrant
<b>Parameters (In)</b>	EcuM_WakeupSourceTypewakeupSource
<b>Parameters (Inout)</b>	None

<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	The callout shall stop the given wakeup source(s) after unsuccessful wakeup validation.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>In Communication with application SW-C</b>	None

## 6.3.13 Alarm Clock

### 6.3.13.1 EcuM\_SetRelWakeupAlarm

<b>Function Name</b>	EcuM_SetRelWakeupAlarm	
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_SetRelWakeupAlarm (EcuM_UserType user, uint32 time)	
<b>Service ID</b>	0x22	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant	
<b>Parameters (In)</b>	User, time	
<b>Parameters (Inout)</b>	None	
<b>Parameters (Out)</b>	None	
<b>Return Value</b>	Std_ReturnType	E_OK: The service has succeeded E_NOT_OK: The service failed E_NOT_SUPPORTED: The service is not supported by this hardware E_EARLIER_ACTIVE: An earlier alarm is already set
<b>Description</b>	This Service sets a user's wakeup alarm relative to the current point in time.	
<b>Preconditions</b>	Startup I must be completed.	
<b>Configuration Dependency</b>	This API is available only if configuration parameter EcuMAlarmClockPresent is set to true. Also this API is available only to users configured to EcuMAlarmClockUser.	
<b>In Communication with application SW-C</b>	Rte_Call_<P>_SetRelWakeupAlarm (EcuM_UserType user, uint32 time) <P> : R-Port Name	

### 6.3.13.2 EcuM\_SetAbsWakeupAlarm

<b>Function Name</b>	EcuM_SetAbsWakeupAlarm	
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_SetAbsWakeupAlarm (EcuM_UserType user, uint32 time)	
<b>Service ID</b>	0x23	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant	
<b>Parameters (In)</b>	User, time	
<b>Parameters (Inout)</b>	None	
<b>Parameters (Out)</b>	None	
<b>Return Value</b>	Std_ReturnType	E_OK: The service has succeeded E_NOT_OK: The service failed E_NOT_SUPPORTED: The service is not supported by this hardware

	E_EARLIER_ACTIVE: An earlier alarm is already set E_PAST: The given point in time has already passed
<b>Description</b>	EcuM_SetAbsWakeupAlarm sets the user's wakeup alarm to an absolute point in time.
<b>Preconditions</b>	Startup I must be completed.
<b>Configuration Dependency</b>	This API is available only if configuration parameter EcuMAlarmClockPresent is set to true. Also this API is available only to users configured to EcuMAlarmClockUser.
<b>In Communication with application SW-C</b>	Rte_Call_<P>_SetAbsWakeupAlarm (EcuM_UserType user, uint32 time) <P> : R-Port Name

## 6.3.13.3 EcuM\_AbortWakeupAlarm

<b>Function Name</b>	EcuM_AbortWakeupAlarm	
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_AbortWakeupAlarm (EcuM_UserType user)	
<b>Service ID</b>	0x24	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant	
<b>Parameters (In)</b>	User	
<b>Parameters (Inout)</b>	None	
<b>Parameters (Out)</b>	None	
<b>Return Value</b>	Std_ReturnType	E_OK: The service has succeeded E_NOT_OK: The service failed E_NOT_SUPPORTED: The service is not supported by this hardware E_NOT_ACTIVE: No owned alarm found
<b>Description</b>	EcuM_AbortWakeupAlarm aborts the wakeup alarm previously set by this user.	
<b>Preconditions</b>	Startup I must be completed.	
<b>Configuration Dependency</b>	This API is available only if configuration parameter EcuMAlarmClockPresent is set to true. Also this API is available only to users configured to EcuMAlarmClockUser.	
<b>In Communication with application SW-C</b>	Rte_Call_<P>_AbortWakeupAlarm (EcuM_UserType user) <P> : R-Port Name	

## 6.3.13.4 EcuM\_GetCurrentTime

<b>Function Name</b>	EcuM_GetCurrentTime	
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_GetWakeupTime (P2VAR(uint32, AUTOMATIC, ECUM_APPL_DATA) time)	
<b>Service ID</b>	0x25	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant	
<b>Parameters (In)</b>	None	
<b>Parameters (Inout)</b>	None	
<b>Parameters (Out)</b>	Time	
<b>Return Value</b>	Std_ReturnType	E_OK: The service has succeeded E_NOT_OK: The service failed E_NOT_SUPPORTED: The service is not supported by this hardware
<b>Description</b>	EcuM_GetCurrentTime returns the current value of the EcuM clock (i.e. the time since battery connect).	

<b>Preconditions</b>	Startup I must be completed.
<b>Configuration Dependency</b>	This API is available only if configuration parameter EcuMAlarmClockPresent is set to true.
<b>In Communication with application SW-C</b>	Rte_Call_<P>_GetCurrentTime (uint32* time) <P> : R-Port Name

## 6.3.13.5 EcuM\_GetWakeupTime

<b>Function Name</b>	EcuM_GetWakeupTime	
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_GetWakeupTime (P2VAR(uint32, AUTOMATIC, ECUM_APPL_DATA) time)	
<b>Service ID</b>	0x26	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant	
<b>Parameters (In)</b>	None	
<b>Parameters (Inout)</b>	None	
<b>Parameters (Out)</b>	Time	
<b>Return Value</b>	Std_ReturnType	E_OK: The service has succeeded E_NOT_SUPPORTED: The service is not supported by this hardware
<b>Description</b>	EcuM_GetWakeupTime returns the current value of the master alarm clock (the minimum absolute time of all user alarm clocks).	
<b>Preconditions</b>	Startup I must be completed.	
<b>Configuration Dependency</b>	This API is available only if configuration parameter EcuMAlarmClockPresent is set to true.	
<b>In Communication with application SW-C</b>	Rte_Call_<P>_GetWakeupTime (uint32* time) <P> : R-Port Name	

## 6.3.13.6 EcuM\_SetClock

<b>Function Name</b>	EcuM_SetClock	
<b>Syntax:</b>	FUNC(Std_ReturnType, ECUM_CODE) EcuM_GetWakeupTime (P2VAR(uint32, AUTOMATIC, ECUM_APPL_DATA) time)	
<b>Service ID</b>	0x27	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant	
<b>Parameters (In)</b>	User, Time	
<b>Parameters (Inout)</b>	None	
<b>Parameters (Out)</b>	None	
<b>Return Value</b>	Std_ReturnType	E_OK: The service has succeeded E_NOT_OK: The service failed E_NOT_SUPPORTED: The service is not supported by this hardware E_NOT_ALLOWED: This service is privileged (BSW only)
<b>Description</b>	EcuM_SetClock sets the EcuM clock time to the provided value. This API is useful for testing the alarm services; Alarms that take days to expire can be tested.	
<b>Preconditions</b>	Startup I must be completed.	
<b>Configuration Dependency</b>	This API is available only if configuration parameter EcuMAlarmClockPresent is set to true. Also this API is available only to users configured to EcuMSetClockAllowedUsers.	



<i>In Communication with application SW-C</i>	Rte_Call_<P>_SetClock (uint32* time) <P> : R-Port Name
---	---

## 7. Generator

### 7.1 Generator Option

Option	Description
S	Generates mode switch interfaces for ECU modes and service-specific client-server interfaces and P-Ports in Swcd_Bsw_EcuM.arxml
DataType	Generates Data Type in Swcd_Bsw_EcuM.arxml, If AUTOSAR_DataTypes is not present
OsConfPrefix	Adds "OsConf" prefix when generating OS-related macros

### 7.2 Generator message

#### 7.2.1 Error Messages

**ERR010001:** EcuMWakeupSourceId in the range of [0 – ‘Wakeup Source ID’] should be configured with the short name as below.

Other wakeup source's ID should start from ‘num’.

This error occurs, if EcuMWakeupSource ID and shortname is not matched as below table.

Wakeup Source ID	short name
0	ECUM_WKSOURCE_POWER
1	ECUM_WKSOURCE_RESET
2	ECUM_WKSOURCE_INTERNAL_RESET
3	ECUM_WKSOURCE_INTERNAL_WDG
4	ECUM_WKSOURCE_EXTERNAL_WDG

**ERR010002:** Value of the parameter 'EcuMValidationTimeout' in the container ‘EcuMWakeupSource’ is reset to <0>, when configured value of the parameter 'EcuMWakeupSourceId' in the container ‘EcuMWakeupSource’ is in the range of <0-4>.

This error message occurs, if the value of the parameter 'EcuMValidationTimeout' in the container 'EcuMWakeupSource' is not configured as <0>, when the value of the parameter 'EcuMWakeupSourceId' in the container 'EcuMWakeupSource' is configured in the range <0-4>.

**ERR010003:** 'ECUM\_WKSOURCE\_RESET' of the container EcuMWakeupSource should be configured with the ID 1.

This error occurs, if 'ECUM\_WKSOURCE\_RESET'’s id in not 1.

**ERR010004:** ResetModeId in the range of [0 – ‘Reset Mode ID’] should be configured with the short name as below.

Other reset mode's ID should start from ‘num’.

This error occurs, if Reset Mode ID and shortname is not matched as below table.

Reset Mode ID	short name
0	ECUM_RESET_MCU
1	ECUM_RESET_WDG
2	ECUM_RESET_IO

**ERR010005:** ShutdownCauseId in the range of [0 – ‘Shutdown Cause ID’] should be configured with the short name as below.

Other shutdown cause's ID should start from ‘num’.

This error occurs, if Reset Mode ID and shortname is not matched as below table.

Shutdown Cause ID	short name
0	ECUM_CAUSE_UNKNOWN
1	ECUM_CAUSE_ECU_STATE
2	ECUM_CAUSE_WDGM
3	ECUM_CAUSE_DCM

**ERR010006:** If there are multiple containers for the same ‘Container Name’, the parameter ‘Parameter Name’ should be equal in all containers.

This error occurs, if same containers with same name has different parameters.

**ERR010007:** The container ‘Container Name’ should be configured at least lower multiplicity ‘lower’.

This error occurs, if container is not configured at least lower multiplicity.

**ERR010008:** The short name of ‘Container Name’ should be unique.

This error occurs, if container is not unique.

**ERR010009:** The DriverListItem ‘Drive Init’ of ‘Root’ should be unique.

This error occurs, if DriverListItem is not unique.

**ERR010010:** The parameter ‘Parameter Name’ of ‘Container Name’ should be unique.

This error occurs, if parameter is not unique.

**ERR010011:** The container ‘Container Name’ should be configured within lower multiplicity ‘lower’ and upper multiplicity ‘upper’.

This error occurs, if container is not configured within lower multiplicity and upper multiplicity.

**ERR010012:** Value of the parameter 'EcuModuleID' in the container 'EcuMDriverInitItem' should not be configured as <Det>, since value of the parameter 'EcuMIncludeDet' in the container 'EcuMGeneral' is configured as <0>.

This error occurs, if the value of the parameter 'EcuModuleID' in the container 'EcuMDriverInitItem' is configured as <Det> when the value of the parameter 'EcuMIncludeDet' in the container

'EcuMGeneral' is configured as <false/0>.

**ERR010013: Value of the parameter 'EcuMModuleID' in the container 'Container Name' should not be configured as <EcuM>.**

This error occurs, if the value configured for the Parameter 'EcuMModuleID' of container 'EcuMDriverInitItem' is <EcuM>.

**ERR010014: The parameter OsAppEcucPartitionRef should be configured in case of Multi-Core.**

This error occurs, if OsAppEcucPartitionRef is not configured in case of Multi-Core.

**ERR010015: The parameter OsApplicationCoreAssignment should be configured in case of Multi-Core.**

This error occurs, if OsApplicationCoreAssignment is not configured in case of Multi-Core.

**ERR010016: The parameter OsNumberOfCores should be equal to the number of Container OsApplication with different OsCoreAssignment.**

This error occurs, if OsNumberOfCores is not equal to the number of Container OsApplication with different OsCoreAssignment.

**ERR010017: The parameter EcuMPartitionRef should be configured for EcucPartition which includes Master EcuM in case of Multi-Core.**

This error occurs, if EcuMPartitionRef is not configured for EcucPartition which includes Master EcuM in case of Multi-Core.

**ERR010018: The parameter EcuMPartitionRef should be configured for All EcucPartition to execute EcuM in case of Multi-Core.**

**And BswModuleExecution parameter of Main EcucPartition should be configured to true.**

This error occurs, if EcuMPartitionRef is not configured for All EcucPartition to execute EcuM in case of Multi-Core and BswModuleExecution parameter of Main EcucPartition is not configured to true.

**ERR010019: The number of OsEventRef should be equal to the number of slave cores because this OsTask is configured for shutdown or sleep synchronization in case of Multi-Core.**

This error occurs, if number of OsEventRef items is not equal to the number of slave cores.

**ERR010020: All events configured for EcuMOsEventRef should be referenced in Os Tasks configured for EcuMOsTaskRef.**

This error occurs, if an Os Task of OsTaskRef does not contain Os Events in OsEventRef.

**ERR010021: The reference path <Path> provided for the parameter 'Parameter Name' in the container 'Container Name' which is referred by the parameter 'Parameter1 Name' in the container**

**‘Container1 Name’ is incorrect.**

This error occurs, if the reference path is incorrect for the parameters 'OsTaskEventRef' or 'OsTaskAccessingApplication' in the container 'OsTask' which is referred by the parameter 'EcuMOsTaskRef' in the container 'EcuMFlexConfiguration'.

Parameter Name	Container Name	Parameter1 Name	Container1 Name
OsTaskEventRef	OsTask	EcuMOsTaskRef	EcuMFlexConfiguration
OsTaskAccessingApplication			

**ERR010022: The reference path is empty for the parameter ‘Parameter Name’ in the container ‘Container Name’ which is referred by the parameter ‘EcuMOsTaskRef’ in the container ‘EcuMFlexConfiguration’.**

This error occurs, if the reference path is empty for the parameters 'OsTaskEventRef' or 'OsTaskAccessingApplication' in the container 'OsTask' which is referred by the parameter 'EcuMOsTaskRef' in the container 'EcuMFlexConfiguration'.

Container Name	Parameter Name
OsTask	OsTaskEventRef
	OsTaskAccessingApplication

**ERR010025: Main Function Period’ period’ should be matched with the period of Bsw Timing Event’te period’.**

This error occurs, if main function period is not matched with the period of Bsw Timing Event.

**ERR010030: The module parameter for 'Module' should not be POSTBUILD\_PTR to initialize in DriverInitListZero.**

This error occurs, if module parameter for 'Module' is configured POSTBUILD\_PTR in DriverInitListZero.

**ERR010040: When module 'ModuleID' is configured for POSTBUILD\_PTR, 'ModulePostBuildPtr' Should be configured.**

This error occurs, if ‘ModulePostBuildPtr’ value is empty when 'ModuleID'’s module parameter value is configured as POSTBUILD\_PTR in DriverInitItem.

**ERR010041: To configure Mcu\_Init() for EcuMDriverInitList, the parameter of NormalMcuClockRef should be configured.**

This error occurs, if NormalMcuClockRef is not configured.

**ERR042054: Number of fields is not same for the entity ‘entity’.**

This error occurs, if number of fields is not same for the entity.

**ERR042055: The input arxmls are not validated against the schema. Please correct the arxml as per**

schema or If you need any support contact HYUNDAI AUTOEVER Corp.

This is an Unexpected Error. On the occurrence of this error contact HYUNDAI AUTOEVER Corp.

**ERR010056:** The input arxmls are not validated against the schema. This error may be due to the incorrect configuration of the element(s) 'Element Name'. Please correct the arxml as per schema or If you need any support contact HYUNDAI AUTOEVER Corp.

This error occurs, if the structure fields that are to be generated in the C Source file are empty. Contact HYUNDAI AUTOEVER Corp.

**ERR042057:** Field 'field' is empty in the entity 'entity'.

This error occurs, if field is empty.

**ERR042058:** The value 'value' of the structure element 'structure element name' in structure 'structure name' is float value. The value 'value' will be truncated since, it's data type is 'type' and it should be an integer number, within the range of 'range'.

This error occurs, if 'value' of 'field' is float value.

**ERR042059:** The value <Value> of the structure element 'Structure Element Name' in structure 'Structure Name' is not within the range. The value <Value> should be within the range of <Min Value> - <Max Value>, as its data type is <Type>.

This error occurs, if the Value of the structure element < Min value or Value of the structure element > Max value.

**ERR042060:** The value <Value> of the structure element 'Structure Element Name' in structure 'Structure Name' is not within the range. The value 'value' should be either 'min' or 'max', as it's data type is 'boolean'

This error occurs, if the Value of the structure element either 'min' or 'max'

**ERR042061:** The value configured for the element 'Structure Element Name' in structure 'Structure Name' should follow C syntax <[a-zA-Z][a-zA-Z0-9W\_]>.

This error occurs, if the Value of the structure element does not adhere to C syntax for variables

**ERR010062:** 'Component Name' Component is not present in the input file(s).

This error occurs, if any one of EcuM, Mcu or Os components is not present in any of the input ECU Configuration Description File(s).

**ERR042063:** The reference path <Path> provided for the parameter 'Parameter Name' in the container 'Container Name', having short name 'Container Short Name' is incorrect.

This error occurs, if incorrect reference is provided for the reference parameter.

Container Name	Parameter Name
EcuMCommonConfiguration	EcuMDefaultAppMode
	EcuMOSResource
EcuMDemEventParameterRefs	ECUM_E_CONFIGURATION_DATA_INCONSISTENT
	ECUM_E_RAM_CHECK_FAILED
	ECUM_E_IMPROPER_CALLER
EcuMDefaultShutdownTarget	EcuMDefaultResetModeRef
	EcuMDefaultSleepModeRef
EcuMSleepMode	EcuMSleepModeMcuModeRef
	EcuMWakeupSourceMask
EcuMWakeupSource	EcuMComMChannelRef
	EcuMResetReasonRef
EcuMFlexConfiguration	EcuMOSTaskRef
	EcuMFlexGptConfigurationRef
	EcuMFlexModuleConfigurationRef
	EcuMNormalMcuModeRef
EcuMAlarmClock	EcuMAlarmClockUser
EcuMGoDownAllowedUsers	EcuMGoDownAllowedUserRef
EcuMSetClockAllowedUsers	EcuMSetClockAllowedUserRef
EcuMFlexGeneral	EcuMAlarmWakeupSource

**ERR042064:** The reference path is empty for the parameter 'Parameter Name' in the container 'Container Name', having short name 'Container Short Name'.

This error occurs, if References provided for the parameter 'Parameter Name' in the container 'Container Name' are empty.

Container Name	Parameter Name
EcuMCommonConfiguration	EcuMDefaultAppMode
	EcuMOSResource
EcuMSleepMode	EcuMSleepModeMcuModeRef
	EcuMWakeupSourceMask
EcuMWakeupSource	EcuMResetReasonRef
EcuMFlexConfiguration	EcuMNormalMcuModeRef
EcuMAlarmClock	EcuMAlarmClockUser
EcuMGoDownAllowedUsers	EcuMGoDownAllowedUserRef
EcuMSetClockAllowedUsers	EcuMSetClockAllowedUserRef

**ERR042065:** The parameter 'Parameter Name' in the container 'Container Name' should be configured.

This error occurs, if the parameter 'Parameter Name' in the container 'Container Name' is not configured.

Container Name	Parameter Name
BSW-IMPLEMENTATION	AR-RELEASE-VERSION
	VENDOR-ID

Container Name	Parameter Name
	SW-VERSION
BSW-MODULE-DESCRIPTION	MODULE-ID
EcuMCommonConfiguration	EcuMConfigConsistencyHash
EcuMDefaultShutdownTarget	EcuMDefaultState
EcuMDriverInitItem	EcuMModuleID
	EcuMModuleService
EcuMSleepMode	EcuMSleepModelId
	EcuMSleepModeSuspend
EcuMWakeupSource	EcuMWakeupSourceId
	EcuMWakeupSourcePolling
EcuMAlarmClock	EcuMAlarmClockId
	EcuMAlarmClockTimeOut
EcuMFlexUserConfig	EcuMFlexUser
EcuMResetMode	EcuMResetModelId
EcuMShutdownCause	EcuMShutdownCauseId
EcuMGeneral	EcuMDevErrorDetect
	EcuMIncludeDet
	EcuMVersionInfoApi
	EcuMMainFunctionPeriod
EcuMFlexGeneral	EcuMAlarmClockPresent
	EcuMResetLoopDetection

**ERR042066:** The value configured for the parameter 'Parameter Name' in the container 'Container Name' should follow the pattern: <Pattern>.

This error occurs, when the parameter 'Parameter Name' is not configured as per the pattern.

Parameter Name	Container Name	Pattern	Example
AR-RELEASE-VERSION	BSW-IMPLEMENTATION	4.[0-9]+.[0-9]+	4.0.3
SW-VERSION		1.[0-9]+.[0-9]+	1.0.0
EcuMModuleID	EcuMDriverInitItem	[a-zA-Z][a-zA-Z0-9W_]*	Adc
EcuMModuleService			Init
EcuMWakeUpReactionSeqName	EcuMFlexConfiguration	[a-zA-Z][a-zA-Z0-9W_]*	functionname

**ERR042067:** Value of the parameter 'EcuMDevErrorDetect' in the container 'EcuMGeneral' should not be configured as <1>, since value of the parameter 'EcuMIncludeDet' in the container 'EcuMGeneral' is configured as <0>.

This error occurs, if value of the parameter 'EcuMDevErrorDetect' in the container 'EcuMGeneral' is configured as <1>, when value of the parameter 'EcuMIncludeDet' in the container 'EcuMGeneral' is configured as <0>.

**ERR042068:** Parameter 'Dependent Parameter Name' in the container 'Dependent Container Name' should not be configured, since value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is configured as 'original\_value'.

This error occurs, if dependent parameter is configured.

**ERR042069:** Parameter 'Parameter Name' in the container 'Container Name' should be configured, since value of the parameter 'Parameter1 Name' in the container 'Container1 Name' is configured as <Value>.

This error occurs in one of the following conditions:

1. EcuMDefaultSleepModeRef is not configured and EcuMDefaultShutdownTarget is configured as EcuMStateSleep or
2. EcuMDefaultResetModeRef is not configured and EcuMDefaultShutdownTarget is configured as EcuMStateReset or
3. EcuMFlexGptConfigurationRef is not configured and EcuMAlarmClockPresent is configured as 1.

Parameter Name	Container Name	Parameter1 Name	Container1 Name	Value
EcuMDefaultSleepModeRef	EcuMDefaultShutdownTarget	EcuMDefaultState	EcuMDefaultShutdownTarget	EcuMStateSleep
EcuMDefaultResetModeRef				EcuMStateReset
EcuMFlexGptConfigurationRef	EcuMFlexConfiguration	EcuMAlarmClockPresent	EcuMFlexGeneral	1

**ERR042070:** Value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name' should not be configured, since value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is configured.

This error occurs, if dependent parameter Name' is configured.

**ERR042071:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should be less than the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.

This error occurs, if actual parameter is greater than or equal to dependent parameter.

**ERR042072:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should be less than or equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.

This error occurs, if actual parameter is greater than dependent parameter.

**ERR042073:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should be greater than the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.



This error occurs, if actual parameter is less than or equal to dependent parameter.

**ERR042074:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should be greater than or equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.

This error occurs, if actual parameter is less than dependent parameter.

**ERR042075:** Value of the parameter 'Actual Parameter Name' in the container container 'Actual Container Name' should be equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.

This error occurs, if actual parameter is not equal to dependent parameter.

**ERR042076:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should not be equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.

This error occurs, if actual parameter is equal to dependent parameter.

**ERR042077:** Multiplicity of the container 'Actual Container Name' should be equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.

This error occurs, if multiplicity of actual container is not equal to dependent parameter.

**ERR042078:** Value of the parameter 'EcuMMainFunctionPeriod' in the container 'EcuMGeneral' should not be configured as <0>.

This error occurs, if the configured value for the parameter 'EcuMMainFunctionPeriod' in the container 'EcuMGeneral' is <0>.

**ERR042079:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should be configured as 'original\_value'.

This error occurs, if actual parameter is not configured as original value.

**ERR042080:** Calculated number of ticks = INT ('Actual Parameter' / 'Actual Main Period') should not be zero.

This error occurs, if calculated ticks is zero.

**ERR042081:** 'EcuMModulePostBuildPtr' should be configured only when Module 'ModuleID' is configured for POSTBUILD\_PTR.

This error occurs, if ModulePostBuildPtr is configured when 'ModuleID's Module Parameter value is configured as POSTBUILD\_PTR in DriverInitItem.

**ERR042082:** EcuM Wakeup Source 'EcuMSourceName' Should be referenced in at least one EcuM

**Sleep Mode(Except Wakeup Source ID 0~4).**

This error occurs, if EcuM Wakeup Source(Except Wakeup Source Id 0~4) is not reference in any EcuM Sleep mode.

**ERR042083: Either Module ID or Module Ref should be configured for Driver Init Item index 'DriverInitItemIndex'.**

This error occurs, if both Module Id and Module Reference are not configured for Driver Init Item.

**ERR010084: The 'GptChannelTickValueMax' of the GptChannelConfiguration referenced in 'EcuMFlexGptConfigurationRef' should be set.**

This error occurs, if 'GptChannelTickValueMax' of the GptChannelConfiguration for EcuM is empty.

## 7.2.2 Warning Messages

**WRN042001: Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should be configured as 'original\_value'.**

This warning occurs, if actual parameter is not configured as original value.

**WRN042002: Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should not be configured as 'original\_value', since value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name' is configured as 'dependent value'.**

This warning occurs, if actual parameter is configured as 'original\_value'.

**WRN042003: Parameter 'Dependent Parameter Name' in the container 'Dependent Container Name' should not be configured, since value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is configured as 'original\_value'.**

This warning occurs, if dependent parameter is configured.

**WRN042004: Parameter 'Dependent Parameter Name' in the container 'Dependent Container Name' should be configured, since value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is configured as 'original\_value'.**

This warning occurs, if dependent parameter is not configured.

**WRN042005: Value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name' should not be configured, since value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is configured.**

This warning occurs, if dependent parameter Name' is configured.

**WRN042006: Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should be less than the value of the parameter 'Dependent Parameter Name' in the container**

**'Dependent Container Name'.**

This warning occurs, if actual parameter is greater than or equal to dependent parameter.

**WRN042007: Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should be less than or equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.**

This warning occurs, if actual parameter is greater than dependent parameter.

**WRN042008: Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should be greater than the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.**

This warning occurs, if actual parameter is less than or equal to dependent parameter.

**WRN042009: Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should be greater than or equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.**

This warning occurs, if actual parameter is less than dependent parameter.

**WRN042010: Value of the parameter 'Actual Parameter Name' in the container container 'Actual Container Name' should be equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.**

This warning occurs, if actual parameter is not equal to dependent parameter.

**WRN042011: Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should not be equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.**

This warning occurs, if actual parameter is equal to dependent parameter.

**WRN042012: Multiplicity of the container 'Actual Container Name' should be equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.**

This warning occurs, if multiplicity of actual container is not equal to dependent parameter.

**WRN042013: Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' should not be configured as 'original\_value'.**

This warning occurs, if actual parameter is configured as original value.

**WRN042014: Parameter 'Parameter Name' is not divisible by the parameter 'EcuMMainFunctionPeriod'.**

This warning occurs, if any one of the following parameters is not divisible by the parameter 'EcuMMainFunctionPeriod'.

Parameter Name
EcuMValidationTimeout
EcuMAlarmClockTimeOut

**WRN042015: Rename the prefix of short name to 'ECUM\_WKSOURCE\_' for consistency.**

This warning occurs, if prefix for shortname is not correct.

**WRN042016: Rename the prefix of short name to 'ECUM\_ALARM\_' for consistency.**

This warning occurs, if prefix for shortname is not correct.

**WRN042017: Rename the prefix of short name to 'ECUM\_SLEEP\_' for consistency.**

This warning occurs, if prefix for shortname is not correct.

**WRN042018: Rename the prefix of short name to 'ECUM\_RESET\_' for consistency.**

This warning occurs, if prefix for shortname is not correct.

**WRN042019: Rename the prefix of short name to 'ECUM\_CAUSE\_' for consistency.**

This warning occurs, if prefix for shortname is not correct.

**WRN042020: Rename the prefix of short name to 'ECUM\_USER\_' for consistency.**

This warning occurs, if prefix for shortname is not correct.

## 7.2.3 Information Messages

**INF042001:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is ignored, since the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name' is configured as 'dependent value'.

This information occurs, if value of the actual parameter is ignored.

**INF042002:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is configured as 'original\_value', when value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name' is configured as 'dependent value'.

This information occurs, if actual parameter is configured as 'original\_value'.

**INF042003:** Parameter 'Dependent Parameter Name' in the container 'Dependent Container Name' is configured, when value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is configured as 'original\_value'.

This information occurs, if dependent parameter is configured.

**INF042004:** Parameter 'Dependent Parameter Name' in the container 'Dependent Container Name' is not configured, when value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is configured as 'original\_value'.

This information occurs, if dependent parameter is not configured.

**INF042005:** Value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name' should not be configured, since value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is configured.

This information occurs, if dependent parameter Name' is configured.

**INF042006:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is not less than the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.

This information occurs, if actual parameter is greater than or equal to dependent parameter.

**INF042007:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is not less than or equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.

This information occurs, if actual parameter is greater than dependent parameter.

**INF042008:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is not greater than the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.

This information occurs, if actual parameter is less than or equal to dependent parameter.

**INF042009:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is not greater than or equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.

This information occurs, if actual parameter is less than dependent parameter.

**INF042010:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is not equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.

This information occurs, if actual parameter is not equal to dependent parameter.

**INF042011:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.

This information occurs, if actual parameter is equal to dependent parameter.

**INF042012:** Multiplicity of the container 'Actual Container Name' is not equal to the value of the parameter 'Dependent Parameter Name' in the container 'Dependent Container Name'.

This information occurs, if multiplicity of actual container is not equal to dependent parameter.

**INF042013:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is configured as 'original\_value'.

This information occurs, if actual parameter is configured as original value.

**INF042014:** Value of the parameter 'Actual Parameter Name' in the container 'Actual Container Name' is not configured as 'original value'.

This information occurs, if actual parameter is not configured as original value.

## 8. SWP Error Code

### 8.1 Dem Error

#### 8.1.1 ECUM\_E\_CONFIGURATION\_DATA\_INCONSISTENT

ErrorId Symbol	ECUM_E_CONFIGURATION_DATA_INCONSISTENT
Description	The consistency of post-build configuration data is checked during initialization and if an error is found, it will be reported.
Cause	ASW
Platform default Action	NO RESET
Functional impact	If an error occurs, the module will be initialized not with the value intended, but with other data.
Related module(s)	None
MCU	Common
Error type	configuration, code
Possible fixes in application	Need to change the configuration set to be used for initialization (/EcuM/EcuMConfiguration/EcuMFlexConfiguration/EcuMFlexModuleConfigurationRef) How to fix the error: Check EcuM_EcuM ->EcuMConfiguration - >EcuMCommonConfiguration -> Config Consistency Hash value. (123456789 Default)

#### 8.1.2 ECUM\_E\_IMPROPER\_CALLER

ErrorId Symbol	ECUM_E_IMPROPER_CALLER
Description	The appropriateness of a user requesting Shutdown (OFF/RESET) is verified (to see if the user is configured as allowed users in GoDownAllowedUsers) and if an error is found, it will be reported.
Cause	ASW
Platform default Action	NO RESET
Functional impact	If this error occurs, the Shutdown request will not be processed.
Related module(s)	None
MCU	Common
Error type	configuration, code
Possible fixes in application	Need to add an application user to request Shutdown (/EcuM/EcuMConfiguration/EcuMFlexConfiguration/EcuMGoDownAllowedUsers) How to fix the error: Check EcuM_EcuM ->EcuMConfiguration - >EcuMFlexConfiguration ->EcuMGoDownAllowedUsers ->GoDownAllowedUserRef configuration.

### 8.2 Det Error

Type or error	Relevance	Related error code	Value
A service was called prior to initialization	Development	ECUM_E_UNINIT	0x10
A function was called which was disabled by configuration	Development	ECUM_E_SERVICE_DISABLED	0x11
A null pointer was passed as an	Development	ECUM_E_NULL_POINTER	0x12

argument			
A parameter was invalid (unspecific)	Development	ECUM_E_INVALID_PAR	0x13
A state, passed as an argument to a service, was out of range (specific parameter test)	Development	ECUM_E_STATE_PAR_OUT_OF_RANGE	0x16
An unknown wakeup source was passed as a parameter to an API	Development	ECUM_E_UNKNOWN_WAKEUP_SOURCE	0x17
The following error code is issued when return value of StartCore was not equal to E_OK.	Development	ECUM_E_START_CORE_FAILED	0x18
The following error code is issued when shutdown failed	Development	ECUM_E_SHUTDOWN_FAILED	0x19
The following error code is issued when sleep is requested with invalid Sleep Mode	Development	ECUM_E_UNKNOWN_SLEEP_MODE	0x20
The following error code is issued when misuse EcuM_GoHalt and EcuM_GoPoll	Development	ECUM_E_INVALID_API_CALL	0x21
The following error code is issued when shutdown is requested with invalid Shutdown Target	Development	ECUM_E_INVALID_SHUTDOWNTARGET	0x22
The following error code is issued when run request is duplicated failed	Development	ECUM_E_MULTIPLE_RUN_REQUESTS	0x23



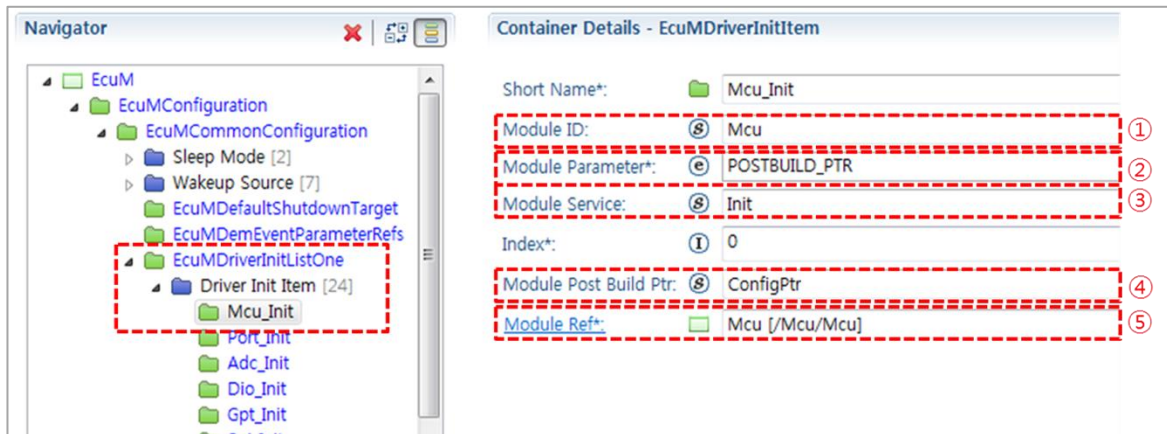
## 9. Appendix

### 9.1 Function-specific Configuration Guide

#### 9.1.1 How to add a module to be initialized

To add or change modules to be initialized, follow the steps shown below.

##### ■ EcuMDriverInitItem configuration



Generation rule for an initialization function: **<Module ID>** + \_ + **<Module Service>** + ( **<Module Post Build Ptr>** )

If the rule is applied to the above configuration, it will be Mcu + \_ + Init + ( ConfigPtr ).

Therefore, the function name generated will be

Mcu\_Init(ConfigPtr).

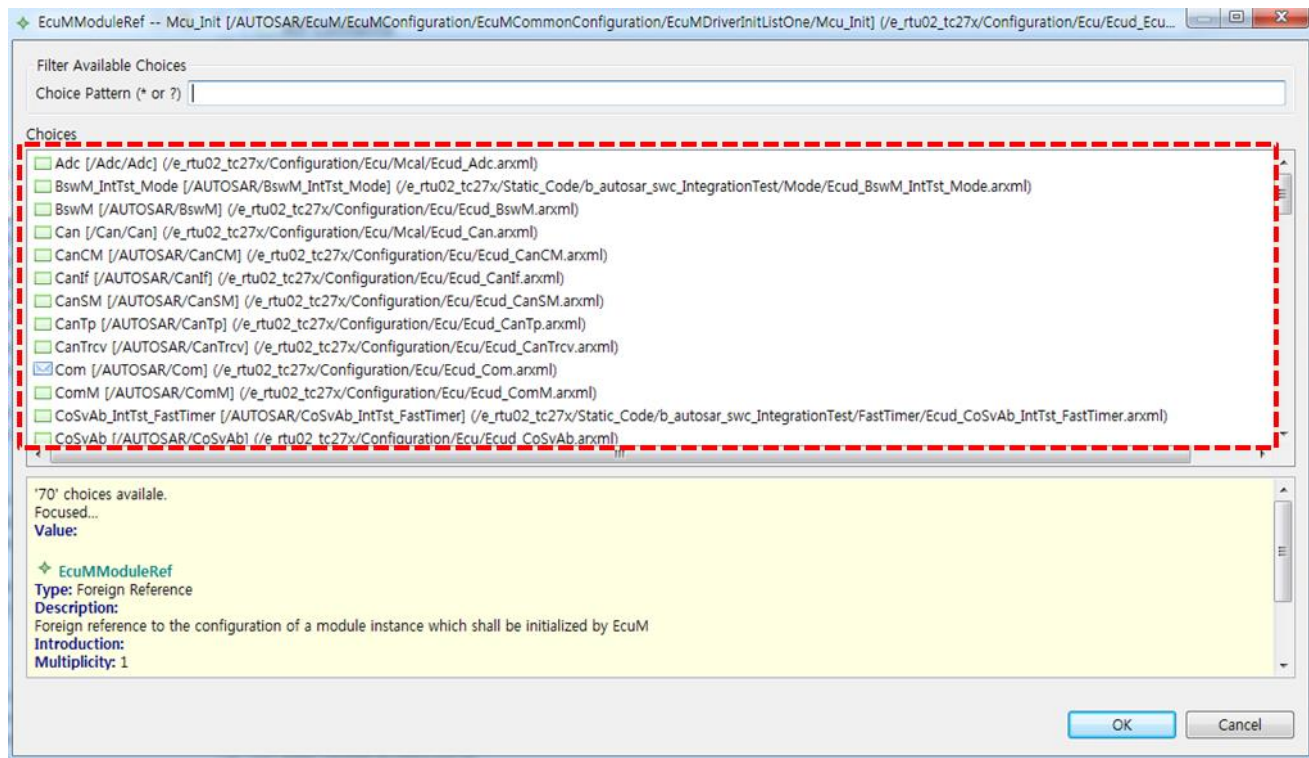
①: Enter **<Module ID>**. If it is left empty, **<Module ID>** will be generated by referencing Module Reference.

⑤: Register Module Reference. No need to register if there is no Ecuc or no header to be created.

When **<Module ID>** is generated through Module Reference, it is generated by referencing VendorID and VendorApilnfix of the module.

※ To register a module reference, the Ecud/Bswmd file of the module should be added to Scons → Genertaion → EcuM → Input Files List (the EcuM Generate Input file of Generate.py in the case of the old version Toolset), which will then generate a proper header file.

※ If both Module ID and Module Reference are left empty, **<Module ID>** can not be generated, which will cause a generation error.



②: Set the relevant initialization function parameter.

Void	VOID
NULL_PTR	NULL_PTR
Others (particular Ptr value)	POSTBUILD_PTR

③: Enter a Service name for the module initialization function.

E.g. Dem\_PreInit(NULL\_PTR);

④: Enter a parameter value required for initialization only if the Module Parameter is POSTBUILD\_PTR.

E.g. &IcuConfigSet,

## ■ Header generation rule

1. If Module Ref is configured: The header file will be generated by referencing Module Name, VendorID, and VendorApiInfix (if configured).

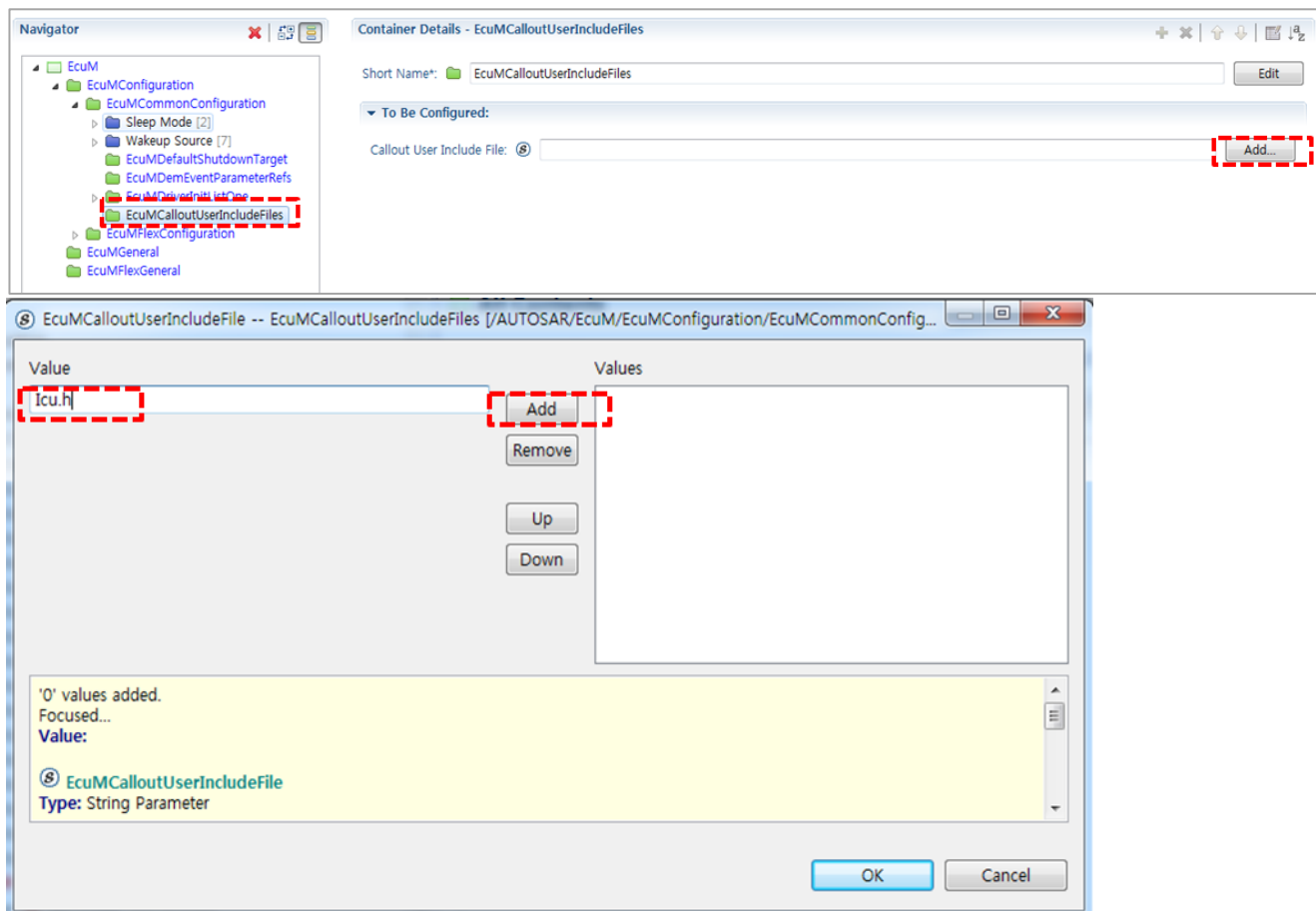
(Need to register to Scons → Genertaion → EcuM → Input Files List or

in the case of the old version Toolset, the EcuM Generate Input file of Generate.py)

2. If Module Ref is not configured: No header file will be linked to the initialization. If a header file needs to be linked, it should be added to the User Include Files as shown below.

## ■ User Include Files configuration

If you followed the above steps but couldn't find the header file, resulting in a build error, you should directly add a header file as illustrated below.



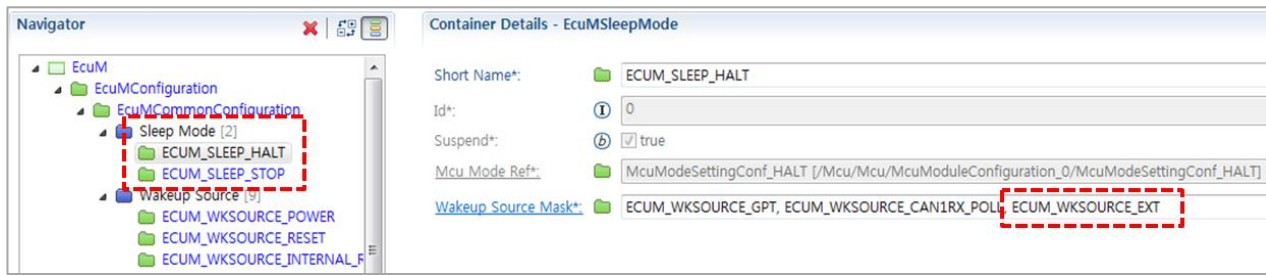
## 9.1.2 How to add a wakeup source

If you want to add a wakeup source directly from the application, follow the instructions below.

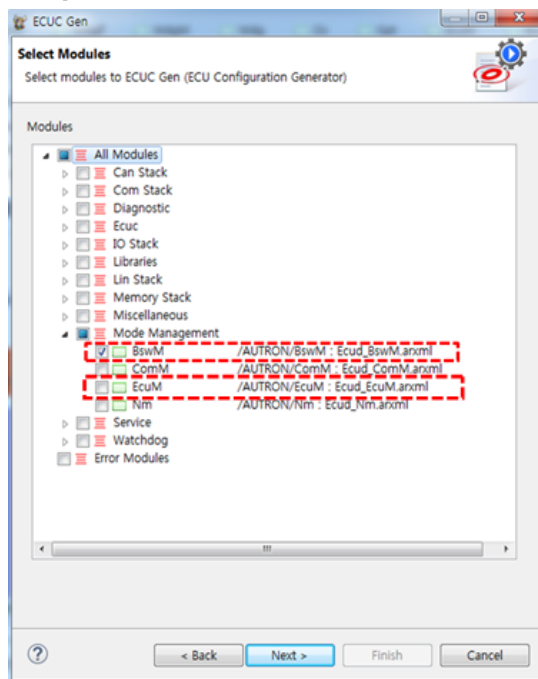
Container Details - EcuMWakeupSource							
Index	Short Name	Validation Tim...	Id	Polling	ComM Chann...	Reset Reason Ref	
0	ECUM_WKSOURCE_POWER	π	0	false		MCU_RESET_UNDEFINED [/Mcu/...	
1	ECUM_WKSOURCE_RESET	π	1	false		MCU_RESET_UNDEFINED [/Mcu/...	
2	ECUM_WKSOURCE_INTERNAL_RESET	π	2	false		MCU_RESET_UNDEFINED [/Mcu/...	
3	ECUM_WKSOURCE_INTERNAL_WDG	π	3	false		MCU_RESET_UNDEFINED [/Mcu/...	
4	ECUM_WKSOURCE_EXTERNAL_WDG	π	4	false		MCU_RESET_UNDEFINED [/Mcu/...	
5	ECUM_WKSOURCE_GPT	π	5	false		MCU_RESET_UNDEFINED [/Mcu/...	
6	ECUM_WKSOURCE_CAN1RX	π	6	false	ComMCha...	MCU_RESET_UNDEFINED [/Mcu/...	
7	ECUM_WKSOURCE_CAN1RX POLL	π	7	false		MCU_RESET_UNDEFINED [/Mcu/...	
8	ECUM_WKSOURCE_EXT	π	8	false		MCU_RESET_UNDEFINED [/Mcu/...	

- Short Name: Follow the rule of ECUM\_WKSOURCE\_<Wakeup source name>.
- Id: Assign a number that follows the previous number. (0-5 not allowed to be used)
- Polling: Set it to false.
- ComM Channel: For a communication-related wakeup source, register the relevant ComM Channel Reference.
- ResetReasonRef: This is fixed as MCU\_RESET\_UNDEFINED.

After adding a wakeup source, you should add it to the Wakeup Source Mask for all Sleep modes. Otherwise, a generation error occurs. (ERR042082)



Check the boxes for both BswM and EcuM, perform harmonization, and generate the relevant BswM rule.



## 9.2 Use Case-specific Configuration Guide

### 9.2.1 A guide to using GPT Timer Wakeup

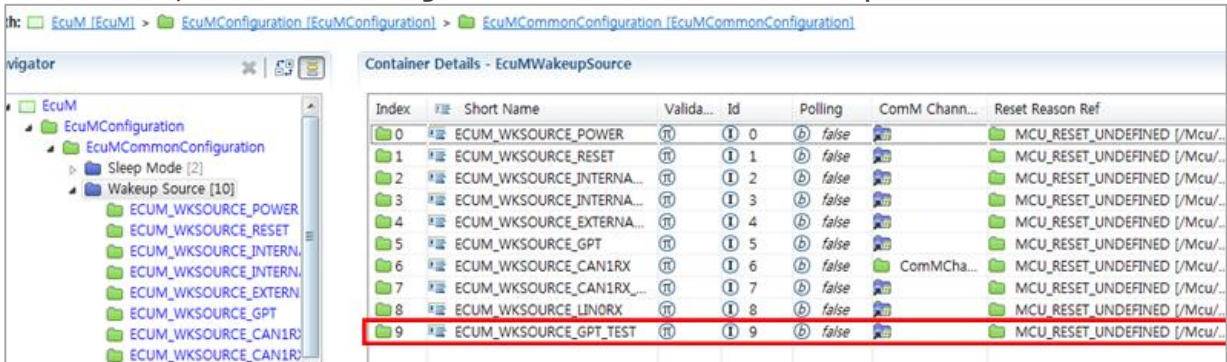
To use GPT Timer Wakeup in Low Power, follow the instructions below.

※ For Renesas RH850 MCUs, if the EcuM functionality is enabled in the PM module, you cannot use User ISR Notification. To use User ISR Notification, you should not enable the EcuM functionality in the PM module. (See the PM Manual.)

※ This configuration is provided as an example, so please refer to the MCAL Manual to learn more about the configuration.

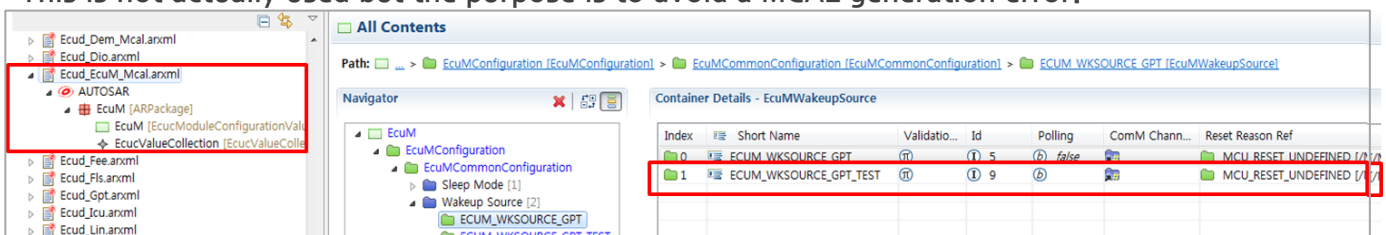
## 9.2.1.1 EcuM configuration

To add a EcuM wakeup source that the Timer will use, follow the instructions below.  
In this case, the Id should be greater than five and have a unique value.

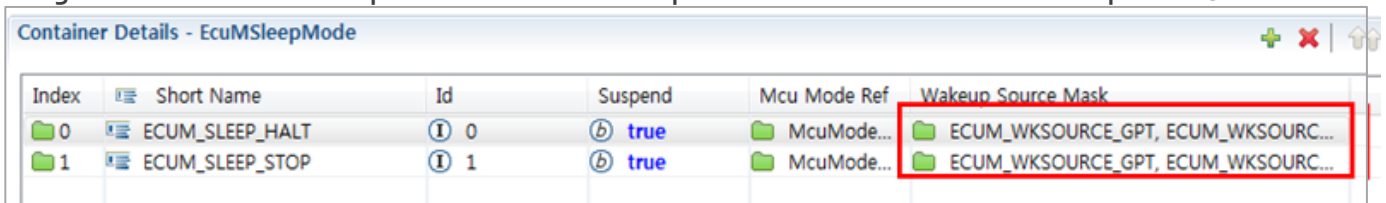


In the case of Bolero MCUs, add the wakeup source with the same Short Name and Id to the EcuM\_Mcal configuration as well. (EcuM\_Mcal.arxml)

This is not actually used but the purpose is to avoid a MCAL generation error.

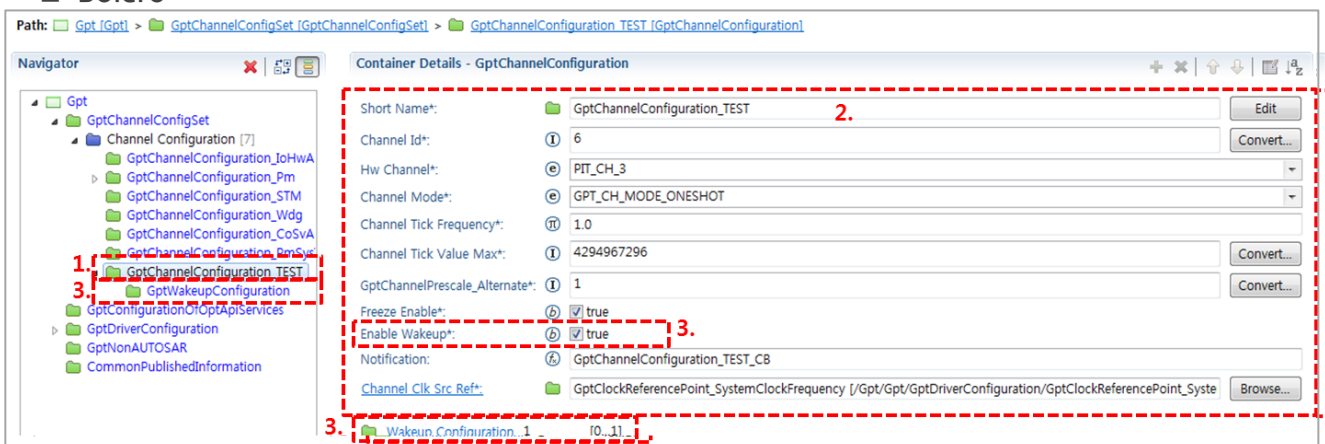


Register the added wakeup source in the Wakeup Source Mask of the EcuM Sleep Mode.



## 9.2.1.2 GPT Channel configuration

### ■ Bolero

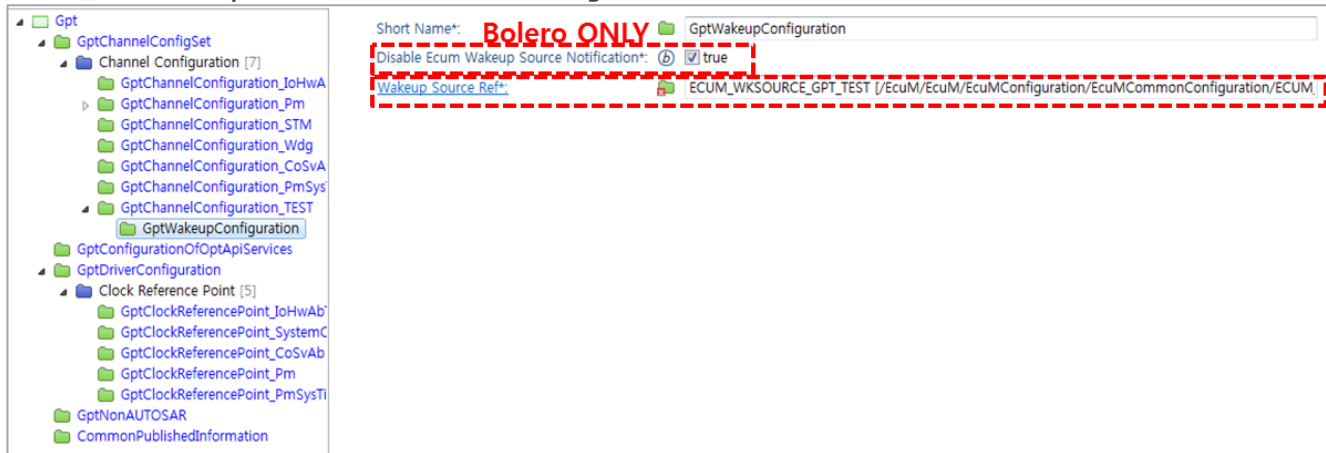


1. Create a new GPT Channel.



2. See the example given above to set up the Channel and Notification API.

3. Check true for Enable Wakeup, and set up the Gpt Wakeup Configuration with the wakeup source added in 9.2.1.1. Set Disable EcuM Wakeup Source Notification to true. This is to prevent redundant EcuM\_SetWakeupEvent calls from occurring inside a MCAL ISR.



If the EcuM functionality is enabled in the PM module, the GPT operation mode is switched to “sleep mode” upon entering Sleep, and after wakeup, it is switched to “normal mode.”

Upon entering the GPT sleep mode, TimerDisable and DisableNotification will be executed for all GPT channels, resulting in canceling the timer. Only if Enable Wakeup = True is set and Gpt\_EnableWakeup is called for a GPT channel, TimerInterruptEnable will be maintained for that GPT channel.

If the PM module does not use the EcuM functionality, the GPT operation mode will always be normal mode.

## ■ RH850

In the case of RH850 MCUs, if the EcuM functionality is enabled in the PM module, User ISR Notification cannot be used. Therefore, the following describes a case where the EcuM functionality is not enabled.

Container Details - GptChannelConfiguration	
Short Name*:	Gpt_USER_TAUJ0I1
Channel Id*:	0
Timer Input Selection*:	TAUJ0I1
Channel Mode*:	GPT_CH_MODE_CONTINUOUS
Channel Tick Frequency*:	0
Channel Tick Value Max*:	65535
Enable Wakeup*:	false
Notification:	Gpt_Notification
Channel Clk*:	CK1

1. Create a new GPT Channel.
2. See the example given above to set up the Channel and Notification API.
3. Set Enable Wakeup to false.  
No Wakeup Configuration is required.

### 9.2.1.3 OS ISR configuration

Register an interrupt of the timer channel to be used in OS configuration.

### 9.2.1.4 Implementation of ISR Notification

- See section 9.2.3 for Wakeup ISR implementation. The implementation should be done in CDD.
- If a wakeup ISR requests a non-periodic LP task, the Gpt IRS Notification API should call EcuM\_SetWakeupEvent. Here, power mode is not to be considered.
- When implementing an ISR in the application, there is no need to call EcuM\_ClearWakeupEvent. In HP mode, 'Wakeup Source Flag Clear' doesn't need to be executed while in LP mode 'Flag Clear' is executed for all wakeup sources by the PM module upon entering H2L/L2H and Sleep.

### 9.2.1.5 Gpt\_EnableWakeup call (Bolero only)

If the EcuM functionality is enabled in the PM module, Gpt\_EnableWakeup should be called for the GPT channel to enable a wakeup before entering a Sleep mode.

## 9.2.2 A guide to using ICU Wakeup

To use ICU channel wakeups, follow the instructions below.

※ This configuration is provided as an example, so please refer to the MCAL Manual to learn more about the configuration.

## 9.2.2.1 Icu Channel

<ul style="list-style-type: none"> <li>Icu             <ul style="list-style-type: none"> <li>IcuConfigSet_0                 <ul style="list-style-type: none"> <li>Channel [8]                     <ul style="list-style-type: none"> <li>IcuChannel_EdgeDetect                         <ul style="list-style-type: none"> <li>IcuSignalEdgeDetection</li> </ul> </li> <li>IcuChannel_EdgeCounter</li> <li>IcuChannel_SignalMeasurement</li> <li>IcuChannel_Timestamp                         <ul style="list-style-type: none"> <li>IcuTimestampMeasurement</li> </ul> </li> <li>IcuChannel_API</li> <li>IcuChannel_CAN1RX                         <ul style="list-style-type: none"> <li>IcuSignalEdgeDetection</li> </ul> </li> <li>IcuChannel_LIN0RX                         <ul style="list-style-type: none"> <li>IcuSignalEdgeDetection</li> </ul> </li> <li>IcuChannel_TEST_LP</li> </ul> </li> <li>IcuGeneral</li> <li>IcuNonAUTOSAR</li> <li>IcuOptionalApis</li> <li>CommonPublishedInformation</li> </ul> </li> </ul> </li></ul>	Short Name*:	IcuChannel_CAN1RX
	Id*:	5
	Hw Channel*:	WKUP_5
	Icu_EXT_ISR_IFERDigitalFilter*:	<input type="checkbox"/> false
	Icu_EXT_ISR_IFMCDigitalFilter*:	0
	IcuWKPU_ISR_WIPUER*:	<input type="checkbox"/> false
	Emios Freeze*:	<input type="checkbox"/> false
	Emios Prescaler*:	EMIOS_PRESCALER_DIVIDE_1
	IcuEmiosPrescaler_Alternate*:	EMIOS_PRESCALER_DIVIDE_1
	Emios Digital Filter*:	EMIOS_DIGITAL_FILTER_BYPASSED
	Emios Bus Select*:	EMIOS_BUS_INTERNAL_COUNTER
	Default Start Edge*:	ICU_FALLING_EDGE
	Measurement Mode*:	ICU_MODE_SIGNAL_EDGE_DETECT
	DMAEnable*:	<input type="checkbox"/> false
	User Mode For Dutycycle*:	SAIC
	Overflow Notification*:	NULL_PTR
	Wakeup Capability*:	<input type="checkbox"/> false
	Signal Measure Without Interrupt*:	<input type="checkbox"/> false
	Disable EcuM Wakeup Source Notification*:	<input checked="" type="checkbox"/> true

ShortName: Configured by user

ID: Configured by user

HwChannel : Set this by referring to the H/W Manual.

Default Start Edge: Configured by user

Measurement Mode: ICU\_MODE\_SIGNAL\_EDGE\_DETECT

Wakeup Capability: False

Disable EcuM Wakeup Source Notification: True

Add Signal Edge Detection as indicated below.

<ul style="list-style-type: none"> <li>Icu             <ul style="list-style-type: none"> <li>IcuConfigSet_0                 <ul style="list-style-type: none"> <li>Channel [8]                     <ul style="list-style-type: none"> <li>IcuChannel_EdgeDetect                         <ul style="list-style-type: none"> <li>IcuSignalEdgeDetection</li> </ul> </li> <li>IcuChannel_EdgeCounter</li> <li>IcuChannel_SignalMeasurement</li> <li>IcuChannel_Timestamp                         <ul style="list-style-type: none"> <li>IcuTimestampMeasurement</li> </ul> </li> <li>IcuChannel_API</li> <li>IcuChannel_CAN1RX                         <ul style="list-style-type: none"> <li>IcuSignalEdgeDetection</li> </ul> </li> <li>IcuChannel_LIN0RX                         <ul style="list-style-type: none"> <li>IcuSignalEdgeDetection</li> </ul> </li> </ul> </li> </ul> </li> </ul> </li></ul>	Short Name*:	IcuSignalEdgeDetection
	Signal Notification*:	CheckWakeup_CAN1RX

Register the Notification API to be used.



## 9.2.2.2 OS ISR configuration

Register an interrupt of the wakeup channel to be used in OS configuration.

## 9.2.2.3 Implementation of ISR Notification

- See section 9.2.3 for Wakeup ISR implementation. The implementation should be done in CDD.
- If a wakeup ISR requests a non-periodic LP task, the Icu ISR Notification API should call EcuM\_SetWakeupEvent. Here, power mode is not to be considered.
- When implementing an ISR in the application, there is no need to call EcuM\_ClearWakeupEvent. In HP mode, 'Wakeup Source Flag Clear' doesn't need to be executed while in LP mode 'Flag Clear' is executed for all wakeup sources by the PM module upon entering H2L/L2H and Sleep.

AppMode\_WakeupEventValidated is a function used to enable a wakeup in Low Power mode. Therefore, the user code to be executed inside the ISR Notification should not be called from within AppMode\_WakeupEventValidated.

If the user code is placed in the AppMode\_WakeupEventValidated function, the function may not be called despite an interrupt having occurred.

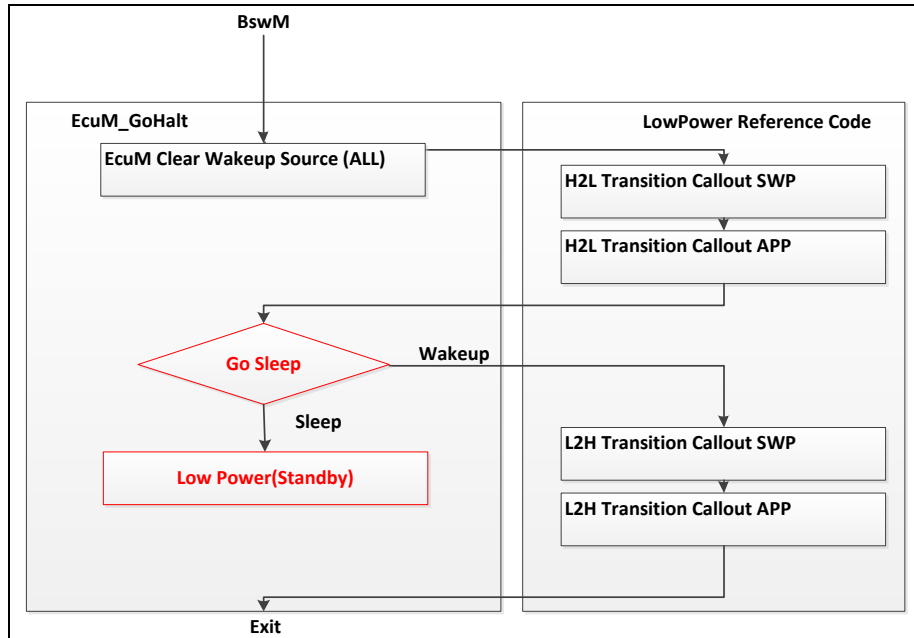
```
graph TD
    ISR_Enter[ISR Enter] --> ISR_Notification[ISR Notification]
    subgraph User_Code [User Code]
        EcuM_SetWakeupEvent[EcuM_SetWakeupEvent]
        Wakeup_Pending{Wakeup Pending}
        User_Code_Block[User Code]
    end
    subgraph Platform_Reference_Code [Platform Reference Code]
        AppMode_WakeupEventValidated[AppMode_WakeupEventValidated  
Do Not Implement User Code Here]
    end
    ISR_Notification --> EcuM_SetWakeupEvent
    EcuM_SetWakeupEvent --> Wakeup_Pending
    Wakeup_Pending -- No --> AppMode_WakeupEventValidated
    Wakeup_Pending -- Yes --> User_Code_Block
    AppMode_WakeupEventValidated --> User_Code_Block
    User_Code_Block --> ISR_Exit[ISR Exit]
```

## User Code Execution

## 9.2.4 A guide to using Chorus MCU LowPower Transition Callout

LowPower Transition Callouts are invoked by EcuM\_GoHalt to support Low Power mode in an environment where the PM module is not present. Chorus MCUs support entering Sleep and wakeups via these callouts.

### 1) LowPower Callout Structure



EcuM\_GoHalt invokes 4 types of callouts in each stage and these callouts exist in Reference Code → Lowpower\_Callout.c.

SWP callouts are not changeable as their code is fixed when the platform is distributed, while App callouts can be implemented as per the project requirements.

See the following implementation guide for further details.

Once the two H2L callouts are executed, Sleep Sequence will proceed.

If there is no issue entering Sleep, the MCU goes into Sleep. As the Chorus MCU enters the Standby mode, it will enter D-Run once it is woken up. (Non-backup area data could be lost.)

The MCU may come out of EcuM\_GoHalt by failing to enter Sleep in the following cases:

- Occurrence of an early wakeup
- Failure to enter Sleep

In such cases, it fails to enter Standby and the L2H callout is invoked. Thus, the logic to handle each possible situation should be implemented within the callout in the application.

## 2) APP Callout implementation guide

※ As the following code is provided as an example, use it as a reference for the implementation.

### - EcuM\_H2LTransition\_Callout\_App

```
/* Enable Wakeup Signal */
Icu_EnableEdgeDetection(IcuChannel_1_Configuration);
Icu_EnableNotification(IcuChannel_1_Configuration);

Icu_EnableEdgeDetection(IcuChannel_2_Configuration);
Icu_EnableNotification(IcuChannel_2_Configuration);

.

/* User Callout for Sleep Sequence */
< User Code for H2L Transition >
```

First, in H2L Transition, you should enable the Edge Detection of the ICU Channel to be used for an external wakeup.

Enable the Edge Detection for all channels by referencing Channel Name and Icu\_Cfg.h in Icu configuration.

E.g.

```
#define IcuConf_IcuChannel_IcuChannel_TEST_LP ((Icu_ChannelType)5U)
```

Later in the application, implement the task to be executed in H2L Transition.

\* Using a particular PAD on the Chorus MCU requires extra work to be done.

If you don't set the SIUL2\_SCR0 bit as shown below, the PAD value may not be changed after Sleep - Wakeup.

The default value is provided so that all PAD values are changeable. Refer to section 16.2.2.11 of the MCU Manual to change them as per the requirements of each controller.

\* According to MCU Errata, bit 23 and bit 31 must be set at all times.

```
/* Chorus PAD MUX SEL SET & PAD26 ERRATA */
REG_BIT_SET32(SIUL_SCR0, 0x1FFUL);
```

### - EcuM\_L2HTransition\_Callout\_App

```
/* User Callout for Wakeup Sequence */

/* Disable all Wakeup Signals */
Icu_DisableEdgeDetection(IcuChannel_1_Configuration);
Icu_DisableNotification(IcuChannel_1_Configuration);

Icu_DisableEdgeDetection(IcuChannel_2_Configuration);
Icu_DisableNotification(IcuChannel_2_Configuration);
```

There is one case where this callout is invoked by the Chorus MCU.

#### 1) An early wakeup while entering Sleep

Implement the logic to address this situation in the application.  
e.g. Restoring what was done in the H2L callout

## - ICU ISR implementation guide

```
CheckWakeup_TEST
{
    <Enter Critical Section>

    Icu_DisableEdgeDetection(IcuChannel_1_Configuration);
    Icu_DisableNotification(IcuChannel_1_Configuration);

    <Exit Critical Section>

    /* Wakeup Sequence */
    EcuM_SetWakeupEvent(WakeupSource);
}
```

Implement the ICU wakeup ISR for all wakeup ICU channels that are used in the application.

The implemented ISR name should be configured in the Signal Notification of the corresponding ICU channel.

(See section 9.2.2.1.)

If a wakeup occurred in the channel, the EdgeDetection and Notification of that ICU channel should be disabled first.

After that, EcuM\_SetWakeupEvent is called to pass the wakeup to the EcuM.

As for the wakeup source, use the wakeup source configured by referring to section 9.1.2.

A guide to writing S6J3xxx MCU LowPower Callout

## 9.2.5 A guide to writing S6J3xxx MCU LowPower Callout

### 9.2.5.1 App\_LowPower\_WakeupEnable

- Insert ICU Channel Enable code for a wakeup upon entering Low Power.
- Add code by referencing non-communication ICU channels used for wakeups, as shown in the below example.

```
/* Insert Wakeup Source Enable Code in Application Use */

Icu_EnableEdgeDetection(IcuConf_IcuChannel_IGN_Wake_Up);
Icu_EnableEdgeDetection(IcuConf_IcuChannel_TAIL_LAMP_IN);
Icu_EnableEdgeDetection(IcuConf_IcuChannel_EINT0);
```

### 9.2.5.2 App\_LowPower\_WakeupDisable

- Insert ICU Channel Disable code for an early wakeup while entering Low Power.
- Add code by referencing non-communication ICU channels used for wakeups, as shown in the below example.

```
/* Insert Wakeup Source Disable Code in Application Use */
```

```
Icu_DisableEdgeDetection(IcuConf_IcuChannel_IGN_Wake_Up);
Icu_DisableEdgeDetection(IcuConf_IcuChannel_TAIL_LAMP_IN);
Icu_DisableEdgeDetection(IcuConf_IcuChannel_EINT0);
```

### 9.2.5.3 App\_LowPower\_CheckRemoteWakeup

- Code to check if an early wakeup occurred while entering Low Power
- Checks whether a wakeup occurred or not by reading an IO pin state
- For communication channels like CAN and LIN, write code by checking the Rx pin on the controller.  
(red-colored)
- In case where other actions are required for an early wakeup, reference communication channel checking logic and make sure wakeup\_mask is updated if the wakeup needs to be processed.(blue-colored)
- This is written based on an example of an early wakeup when 'IGN Pin is High' upon entering Low Power, in the case of application-specific wakeup source handling.

```
/* Insert Early Wakeup Check Code For Wakeup Sources */
```

```
IoHwAb_LevelTypeLddLevel1;
IoHwAb_LevelTypeLddLevel2;
IoHwAb_LevelTypeLddLevel3;

*wakeup_mask = 0x00000000ul;

/* Detect IO Level for Communication RX PIN */
(void)IoHwAb_DigDirReadDirect((uint16)P3_22_C_CAN_RX, &LddLevel1);

if (LddLevel1 == (IoHwAb_LevelType)IOHWAB_LOW)
{
    *wakeup_mask |= ECUM_WKSOURCE_C_CAN;
}

(void)IoHwAb_DigDirReadDirect((uint16)P3_19_M_CAN_RX, &LddLevel2);

if (LddLevel2 == (IoHwAb_LevelType)IOHWAB_LOW)
{
    *wakeup_mask |= ECUM_WKSOURCE_MM_CAN;
}

/* Insert Application Wakeup Source Check Code */
(void)IoHwAb_DigDirReadDirect((uint16)P2_19_EINT3_WAKE_UP, &LddLevel3);

if (LddLevel3 == (IoHwAb_LevelType)IOHWAB_HIGH)
{
    *wakeup_mask |= ECUM_WKSOURCE_IGN;
}
```

### 9.2.5.4 App\_LowPower\_CheckWakeup

- Code to return to High Power after notifying the EcuM when an early wakeup has been detected by reading an IO pin.
- Write code by referring to the EcuM Set Wakeup logic for the communication channels of the Pm.c.

(This example is written based on IGN Wakeup)

\* Need to change App Mode code for the AppMode Wakeup Validated function to be called by the added wakeup source. (See sections 9.2.1 and 9.2.2 of the BswM UM.)

```
/* Insert Set Wakeup Event Code For Application Wakeup Source */  
if ((wakeup_mask & ECUM_WKSOURCE_IGN) == ECUM_WKSOURCE_IGN)  
{  
    EcuM_SetWakeupEvent(ECUM_WKSOURCE_IGN);  
    EcuM_ClearWakeupEvent(ECUM_WKSOURCE_IGN);  
}
```

## 9.2.5.5 App\_LowPower\_DisablePeri

- All the peripherals that could potentially interrupt the entry into Low Power should be disabled before going into Low Power.
- Application-specific peripherals should be disabled inside this API.

```
/* Insert Disable Peripheral Code in Application Use */  
  
⟨APP Peripheral Disable Code⟩
```

## 9.2.6 A guide to using MCU LowPower Transition Callout (General)

In case the PM module is not used, SWP provides LowPower Transition Callout to support actions required before/after low power mode transitions.

Actions required for SWP to perform are to be written in the callouts below.

- EcuM\_H2LTransition\_Callout\_Swp() : This is called for High Power → Low Power transition.
- EcuM\_L2HTransition\_Callout\_Swp() : This is called for Low Power → High Power transition.

**Actions required for APP to perform should be written by the user in the callouts below.**

For required actions, direct guidance from the MCU manufacturer is needed. (Errata, required peripheral control, etc.)

- EcuM\_H2LTransition\_Callout\_App() : This is called for High Power → Low Power transition.
- EcuM\_L2HTransition\_Callout\_App() : This is called for Low Power → High Power transition.

General support details by MCU is as follows:

LP task support depends on whether the PM module is supported or not, and supported LP modes and wakeup behavior depend on the characteristics of the MCU. Please refer to the table below for the MCU that you would like to work with.

MCU	LP Task Support	Supported LP Mode	Upon Wakeup
MPC56x	O	Stop	Maintains the context
F1L	O	Stop	Maintains the context
F1K/F1KM	O	Stop	Maintains the context
SPC58x	X	Standby	Performs a reset
TC2x (AURIX)	X	-	-
TC3x (AURIX)	X	Standby	Performs a reset
Traveo II (CYTx)	X	DeepSleep	Maintains the context
S32K	X	VLPS	Maintains the context
S32G	X	-	-

### 9.2.6.1 A guide to using callouts when a wakeup causes a reset

If Chorus Standby mode or RH850 DeepStop mode is used, being woken up in LowPower mode triggers a reset.

■ Please take note of the following information prior to implementing the callout.

1. Since this happens before the OS is started, interrupts cannot be used before Interrupt Enable is set inside the callout.
2. Since data is not stored upon entering LP, all actions should be completed before entering Lowpower.  
(e.g. Reenter Lowpower once all SPI communication sequences are over.)
3. Need to consider backing up & restoring GPIO configuration and states. As port levels are not maintained, external devices might malfunction.
4. Since Wakeup Unit is not initialized, Wakeup Factor Register can be used for implementation.



When reentering Lowpower, the wakeup flag needs to be initialized.

5. Since platform and MCAL APIs cannot be used, F/W code should be used to implement the whole content.

6. When a wakeup occurs, the wakeup factor should be read in as Chorus does not specifically show the reset reason.

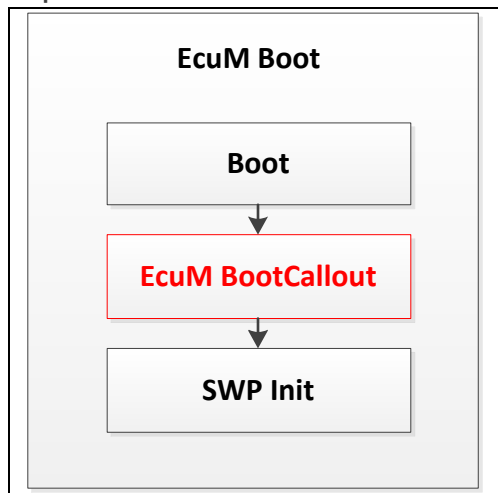
In the case of RH850, the reset reason is displayed as Deepstop.

7. With the Chorus MCU, if ADC is used in Standby mode, Stb-10bit mode should be used.

8. If SIUL2\_SCR0 bit is not set before entering Standby in the Chorus MCU, a particular PAD value may not be changed after a wakeup. See the MCU Manual and section 9.2.4 of this Manual for the EcuM\_H2LTransition\_Callout\_App implementation.

## 9.2.6.1.1 EcuM\_BootCallout

If there is a requirement to reenter the LP mode after performing a particular action prior to entering High Power (platform initialization) (e.g. SPI communication), or to perform a particular action when waking up in Low Power (e.g. Backup Data Restore, etc.), you can use a callout provided by EcuM to implement it.



By checking the wakeup factor for the respective MCU to determine that the reset was caused by a wakeup in Low Power, you can implement proper actions for being woken up in Low Power.

```

/* User Callout for Before SWP Start */

/* Check Wakeup Factor */

If <Wakeup Factor == External Wakeup>
{
<Necessary Functionality Initialize>

<User Action>

<Enter LP Again or Proceed to SWP Init>

```

```
}

```

## 9.2.6.1.2 EcuM\_H2LTransition\_Callout\_App

In this H2L callout, enable ICU channels for wakeups and directly implement the actions that are to be performed by the application.

```
/* Enable Wakeup Signals */
Icu_EnableEdgeDetection(IcuConf_IcuChannel_IcuChannel_CAN0RX);
Icu_EnableNotification(IcuConf_IcuChannel_IcuChannel_CAN0RX);

...
```

## 9.2.6.1.3 EcuM\_L2HTransition\_Callout\_App

In this L2H callout, disable ICU channels for wakeups and directly implement the actions that are to be performed by the application.

```
/* Disable all Wakeup Signals */
Icu_DisableEdgeDetection(IcuConf_IcuChannel_IcuChannel_CAN0RX);
Icu_DisableNotification(IcuConf_IcuChannel_IcuChannel_CAN0RX);
```

### A guide to using callouts when a wakeup does not cause a reset

If S32K VLPS mode or CYTxxxDeepSleep mode is used, the PM module is not provided and a wakeup is not followed by a reset. Also, LP Task is not supported in principle and if necessary users should implement it themselves.

#### ■ Please take note of the following information prior to implementing the callout.

1. Since this callout is executed before the L2H transition function is performed, what was disabled in the H2L Transition function is not to be executed.
2. The callout construction should include determining a wakeup was caused by a valid wakeup factor.
3. Consider the fact that an interrupt could occur for this callout.

## 9.2.6.1.4 EcuM\_H2LTransition\_Callout\_App

In this H2L callout, enable ICU channels for wakeups and directly implement the actions that are to be performed by the application.

```
/* Enable Wakeup Signals */
Icu_EnableEdgeDetection(IcuConf_IcuChannel_IcuChannel_CAN0RX);
Icu_EnableNotification(IcuConf_IcuChannel_IcuChannel_CAN0RX);

.
```

## 9.2.6.1.5 EcuM\_L2HTransition\_Callout\_App

In this L2H callout, disable ICU channels for wakeups and directly implement the actions that are to be performed by the application.

```
/* Disable all Wakeup Signals */  
Icu_DisableEdgeDetection(IcuConf_IcuChannel_IcuChannel_CAN0RX);  
Icu_DisableNotification(IcuConf_IcuChannel_IcuChannel_CAN0RX);  
.
```

## 9.2.6.2 A guide to using callouts for LP Task when a wakeup does not cause a reset

As mentioned above, in principle LP Task is not supported, but you can use a callout to enable an LP Task-like capability. For example, if a certain functionality needs to be performed after a wakeup before reentering Sleep, this can be realized using a callout provided by EcuM.

(※ The functionality mentioned above is given just as guidance and requires further verification.)

### ■ Please take note of the following information prior to implementing the callout.

1. Since this callout is executed before the L2H transition function is performed, what was disabled in the H2L Transition function is not to be executed.
2. The callout construction should include determining a wakeup was caused by a valid wakeup factor.
3. Consider the fact that an interrupt could occur for this callout.

















### ■ A guide to reentering Sleep upon executing a certain task after a periodic wakeup

This guide is written based on S32K MCUs and its concept is as follows:

- A preconfigured Timer triggers periodic wakeups.
- Upon the wakeup, a desired action is performed in EcuM\_Wakeup\_Callout within the EcuM\_SetMode function.
- Afterwards, whether the wakeup is going to be followed by entry into High Power mode or not is determined.
- Since this functionality should return to VLPS mode, Mcu\_SetMode is executed once again.

The S32K MCU should use the LPTMR that can operate in Sleep mode (VLPS mode) and also the Watchdog should be considered.

- 1) Watchdog configuration
  - As the Watchdog should operate in VLPS mode, the SIRC clock is used.
- 2) GPT configuration
  - The GPT configuration should include setting up the LPTMR and the SIRCDIV2 clock that operate in VLPS mode.
  - Periodic wakeups can be triggered by using the GPT channel configured.

Short Name*:	 GptChannelConfiguration_LpTimer
Channel Id*:	 5 <span>Convert...</span>
Hw Channel*:	 LPTMR_0_CH_0
Channel Mode*:	 GPT_CH_MODE_CONTINUOUS
Channel Tick Frequency*:	 1.0E6
Lptmr Prescaler:	 4 <span>Convert...</span>
Lptmr Channel Clk Src:	 SIRCDIV2_CLK
LPitIsExternalTrigger*:	 <input type="checkbox"/> false
LPitEnReloadOnTrigger*:	 <input type="checkbox"/> false
LPitEnStopOnInterrupt*:	 <input type="checkbox"/> false
LPitEnStartOnTrigger*:	 <input type="checkbox"/> false
Channel Tick Value Max*:	 65535 <span>Convert...</span>
Freeze Enable*:	 <input checked="" type="checkbox"/> true
Enable Wakeup*:	 <input type="checkbox"/> false
Notification:	 GptNotification_LpTimer
Channel Clk Src Ref*:	 GptClockReferencePoint_LpTimer [/Gpt/Gpt/GptDriverConfiguration/GptClockReferencePoint_LpTimer] (/e_r) <span>Browse...</span>

## 9.2.6.2.1 EcuM\_H2LTransition\_Callout\_App

- In this H2L callout, you can control the GPT to enable a brief wakeup in Sleep mode.
- The user should consider the start time of GPT interrupts.

```
SchM_Enter_EcuM_WAKEUP_STATUS_PROTECTION();
.
Gpt_EnableNotification(GptChannelConfiguration_LpTimer);
Gpt_StartTimer(GptChannelConfiguration_LpTimer, 50000);
.
SchM_Exit_EcuM_WAKEUP_STATUS_PROTECTION();
```

## 9.2.6.2.2 EcuM\_L2HTransition\_Callout\_App

- In this L2H callout, you can control the GPT to disable the functionality.
- The user should consider when to stop GPT interrupts.

```
SchM_Enter_EcuM_WAKEUP_STATUS_PROTECTION();
.
Gpt_DisableNotification(GptChannelConfiguration_LpTimer);
Gpt_StopTimer(GptChannelConfiguration_LpTimer);
.
SchM_Exit_EcuM_WAKEUP_STATUS_PROTECTION();
```

## 9.2.6.2.3 GptNotification\_LpTimer (User configured)

- Execute Wdgdog Trigger inside GptNotification\_LpTimer so that a Watchdog reset won't occur.

```
void GptNotification_LpTimer(void)
{
```

```
WdgStack_TriggerWatchdog();  
}
```

## 9.2.6.2.4 EcuM\_Wakeup\_Callout

- Implement App code in EcuM\_Wakeup\_Callout.

```
void EcuM_Wakeup_Callout(void)  
{  
    //App Code  
}
```

## 9.2.6.2.5 EcuM\_SetMode

- This is a callout for mode transitions used to request a full wakeup or going back to Sleep mode during periodic wakeups.
- Use a do-while loop up until the point of clock initialization.

```
EcuM_WakeupSourceTypevalidatedWakeupEvents;  
  
do  
{  
    .  
    Mcu_SetMode(McuMode);  
    .  
    EnableAllInterrupts();  
    EcuM_Wakeup_Callout();  
    DisableAllInterrupts();  
    validatedWakeupEvents = EcuM_GetValidatedWakeupEvents();  
} while (validatedWakeupEvents != ECUM_WKSOURCE_NONE);
```

## 9.2.7 Watchdog handling in Sleep state

While the ECU is in SLEEP, the WdgM module is suspended. If the hardware watchdog cannot or should not be activated in SLEEP, the Watchdog should be triggered at regular intervals in order to prevent a watchdog reset from occurring.

Watchdog trigger location

- 1) If the MCU is not reset after a wakeup:
  - When the PM module is used: Pm\_MainFunction
  - When the PM module is not used: Notification of the GPT that is configured to operate in SLEEP
- 2) If the MCU is reset after a wakeup: EcuM\_BootCallout (Implemented by user)

Watchdog trigger cycle (Need to set up by considering the WdgM timeout cycle, See the WdgM User Manual.)

- 1) When the PM module is used: As the Watchdog is triggered after the Low Power task is executed, the cycle of the Low Power task should be set by the user to ensure the Watchdog timeout does not expire.
- 2) When the PM module is not used: As the Watchdog is triggered following a timeout of the GPT that is set to operate in SLEEP, the user should set the GPT cycle so that the Watchdog timeout does not expire.

Additionally, to prevent the Watchdog from being reset while entering/returning SLEEP, SWP triggers the Watchdog at the time of entering/returning SLEEP.

## 9.3 Design Notes

### 9.3.1 Cancellation prohibited while executing Off / Reset command of controller

After requesting the controller's Off / Reset command in the application, the cancellation request is prohibited before the operation is completed.

Therefore, after the final judgment in the application, the Off / Reset command must be requested.

When explaining based on the App\_Mode sample,

After RequestOff / Reset request, RequestRUN command is not accepted.

However, after RequestSleep, which is a Sleep (Low Power mode) request, RequestRUN request is accepted to switch to High Power mode.

### 9.3.2 Application Task before completion of SWP initialization

When the initialization of SWP is completed, the time of completion of SWP initialization is notified to App\_Mode by ECUM\_STATE\_RUN. However, since Rte\_Start is performed during the STARTUP process, the application Task operates before the completion of SWP initialization.

That is, before AppMode\_InitCompleted is called, it should be considered that the application Task is already running.

It is necessary to check whether this situation affects the controller, and if you want to perform the application Task after the completion of SWP initialization, set the Disabled Mode for ECUM\_STATE\_STARTUP\_XXX in the event performing the application Task so that the event is not triggered in the STARTUP state. do. (Refer to the Mode Management User Manual for related information)

In addition, there are cases where Application API is called with Direct Function Call from Rte\_Start, so be careful.

### 9.3.3 Integration Code Check

Incorrect modification of Integration Code may cause problems in SWP operation.

If there are modifications to the code below after distribution, you should review whether there are any problems with the modifications.

(The explanation below is an example for reference.)

- 1) Static\_Code/Integration\_Code/integration\_EcuM/fixedcode/EcuM\_Boot.c  
→ Since it is a fixed code, the user should not modify it arbitrarily.
- 2) Static\_Code/Integration\_Code/integration\_EcuM/usercode/EcuM\_Callout.c  
→ In case of EcuM\_SetMode, user can modify for LP Task implementation which is case 9.2.6.3.
- 3) Static\_Code/Reference\_Code/App\_Mode.c  
→ The user can call the initialization code of ASW in AppMode\_InitCompleted.  
→ Check whether Remote Wakeup processing is correct in AppMode\_WakeupEventValidated.  
→ Check Full Com / No Com conversion in AppMode\_ComMMModeSwitched\_XXXX (communication channel).
- 4) Static\_Code/Reference\_Code/LowPower\_Callout.c

- ➔ EcuM\_H2LTransition\_Callout\_App
- ➔ EcuM\_L2HTransition\_Callout\_App
- ➔ In CheckWakeup\_XXXX, refer to 9.2.4 to implement ICU Wakeup ISR for Wakeup ICU Channel in Application.

5) Static\_Code/Reference\_Code/LowPower\_Callout.h

### 9.3.4 Prohibit duplicate calls to App Mode Active during Wakeup

When it wakes up from Low Power, it is finally converted to High Power through the App Mode Active request inside the ISR. At this time, it should be reviewed whether there is a case where the App Mode Active request overlaps due to the simultaneous occurrence of ISR.

### 9.3.5 Prohibit implementation of Interrupt logic that uses both High Power / Low Power inside AppMode\_WakeupEventValidated

In case of Interrupt that uses both High Power and Low Power (Ex RF ISR), logic should not be implemented inside AppMode\_WakeupEventValidated.

Since AppMode\_WakeupEventValidated is a function for Wake-Up in LowPowerMode, the corresponding API may not be called when Interrupt occurs in High Power.

### 9.3.6 Write ICU ISR for registered Wakeup Source (Chorus Only)

In order to handle wakeup that occurs during sleep entry, the contents of ICU ISR should be implemented in Lp Callout Code (Refer to 9.2 Use Case-specific Configuration Guide).

If there is no such content, the wakeup that occurs during sleep entry can be ignored.