Document Title	Technical Safety Concept Status Report
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	233
Document Classification	Auxiliary

Document Status	Final
Part of AUTOSAR Release	4.2.2

Document Change History

Release	Changed by	Change Description
4.2.2	AUTOSAR Release Management	• Editorial changes
4.2.1	AUTOSAR Release Management	• Editorial changes
4.1.1	AUTOSAR Administration	• Complex Device Drivers renamed Complex Drivers
3.1.5	AUTOSAR Administration	• Minor changes in [RS_BRF_00120], [RS_BRF_00278] and chapter 5.2
3.1.4	AUTOSAR Administration	• Initial release

Disclaimer

Advice for users

AUTOSAR specifications may contain exemplary items (exemplary reference models, "use cases", and/or references to exemplary technical solutions, devices, processes or software).

Any such exemplary items are contained in the specifications for illustration purposes only, and they themselves are not part of the AUTOSAR Standard. Neither their presence in such specifications, nor any later documentation of AUTOSAR conformance of products actually implementing such exemplary items, imply that intellectual property rights covering such exemplary items are licensed under the same rules as applicable to the AUTOSAR Standard.

Table of Contents

1 Introduction

Functional safety is a system characteristic which is taken into account from the beginning, as it may influence system design decisions. Therefore AUTOSAR specifications include requirements related to functional safety.
Aspects such as complexity of the system design can be relevant for the achievement of functional safety in the automotive field.
Software is one parameter that can influence complexity on system level. New techniques and concepts for software development can be used in order to minimize complexity and therefore can ease the achievement of functional safety.

As a software standardization initiative, AUTOSAR reflects the consideration related to functional safety that is relevant for the today's automotive software development.

The aim of this document is to describe the requirements related to functional safety introduced in the release 4.0 of AUTOSAR.

- The document is intended for the user of AUTOSAR, including people involved in safety analysis.
- The document provides information for the user of the AUTOSAR specifications:
  - Safety-related requirements and safety-related features introduced in AUTOSAR 4.0;
  - For each safety-related feature in the BSW&RTE this document shows the means by which the BSW is implementing this feature (i.e. mapping BRF -> SWS)

Additionally the document will provide a technical justification of how the SWS are implementing each BRF Feature (i.e. technical argumentation).

1.1 Technical overview

The following safety mechanisms were included in the AUTOSAR release 4.0.

1.1.1 Program Flow Monitoring Related Features

Program flow monitoring is a technique for checking the correct execution of software and focuses on control flow errors.
An incorrect program flow occurs if one or more program instructions are processed either in the incorrect sequence or are not even processed at all.
Program flow errors can for example lead to data inconsistencies, data corruption, or other software failures.

1.1.2 Timing Related Features

Timing is an important property of embedded systems. Safe behavior requires that the systems actions and reactions are performed within the right time.
The right time can be described in terms of a set of timing constraints that have to be satisfied. However, an AUTOSAR software component cannot ensure proper timing

by itself. It depends on proper support by the AUTOSAR runtime environment and the basic software. During integration the timing constraints of the software components need to be ensured.

The timing-related features address the following four aspects to enable proper software component timing within the AUTOSAR framework:

- Provision of synchronized time-bases to provide a common notion of time across a network of ECUs;
- Provision of means for synchronized execution of runnables within an AUTOSAR ECU and across a network of AUTOSAR ECUs;
- Support by the AUTOSAR RTE, BSW and Methodology for deterministic timing of software components;
- Support by the AUTOSAR RTE and BSW to detect and control timing violations and prevent their propagation.

1.1.3 E-Gas Monitoring Related Features

The E-Gas Monitoring Concept is a safety concept applicable e.g. for diesel and gasoline engine management. It is standardized by the AKEGAS working group and not part of the AUTOSAR standard. It is used as an exemplary item here because it is a standardized and commonly used automotive safety concept to prevent e.g. the hazard "unintended acceleration".
The goal of the E-Gas Monitoring related features in the context of AUTOSAR 4.0 is to enable an implementation of the E-Gas Monitoring Concept within the AUTOSAR framework.

1.1.4 Communication Stack Related Features

The features related to the communication stack are addressing safety mechanisms related to communication failures modes.
The following mechanisms have been added to the communication stack:

- PDU counter to detect “out of sequence”, “lost” and “replicated” messages.
- PDU replication to detect corrupted data and to recover from this failure mode.

1.1.5 End-to-End Communication Protection Related Features

The integrity of the exchange of data between a sender and one ore more receiver(s) within an embedded system can affect functional safety. Therefore such data are transmitted using mechanisms to protect them against the effects of faults within the communication link.
Examples for such faults are random HW faults (e.g. corrupt registers of a CAN transceiver), interference (e.g. due to EMC), and systematic faults within the software implementing the VFB communication (e.g. RTE, IOC, COM and network stacks).
The End-to-End Communication Protection related features are implemented in AUTOSAR 4.0 as a standard library providing E2E communication protection mechanisms that enable sender to protect such data and the receiver to detect and handle errors in the communication link at runtime.

The End-to-End Library provides mechanisms for E2E protection, adequate for safety-related communication having requirements up to ASIL D.

The algorithms of protection mechanisms are implemented in the End-to-End Library. The callers of the End-to-End Library are responsible for the correct usage of the library, in particular for providing correct parameters the End-to-End Library routines.

The End-to-End protection allows the following:

- It protects the safety-related data elements (resp. safety-related I-PDUs) to be sent over the RTE by attaching control data,
- It verifies the safety-related data elements (resp. safety-related I-PDUs) received from the RTE using this control data, and
- It indicates that received safety-related data elements (resp. safety-related I-PDUs) are faulty, which then has to be handled by the receiver SW-C.

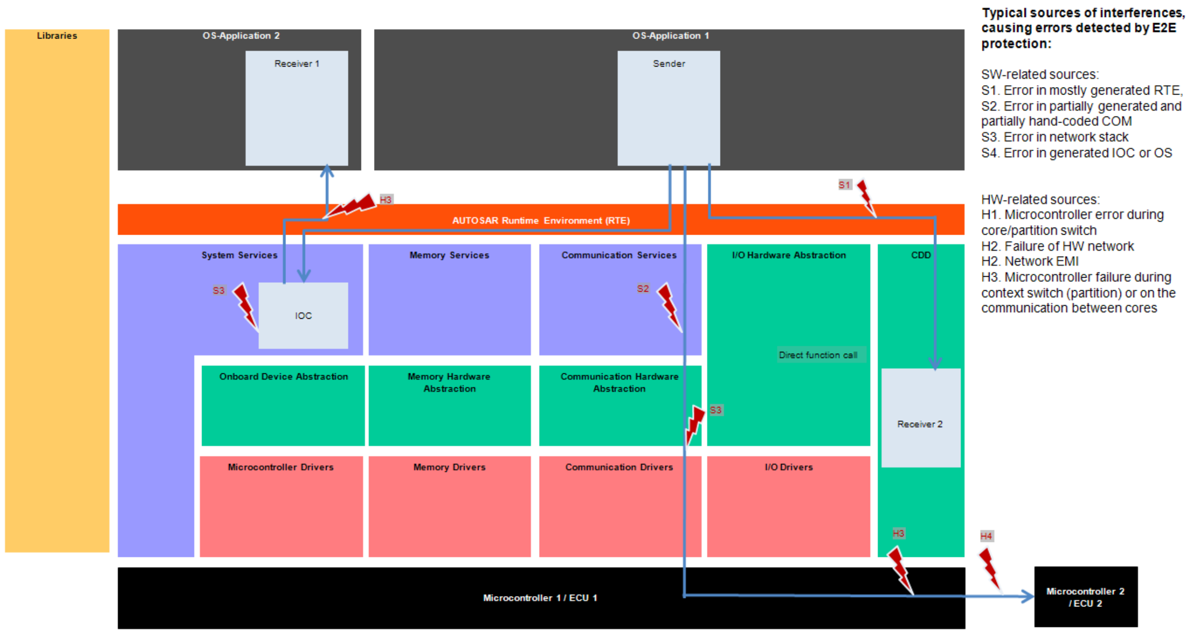


Figure 1: End-to-End Protection

1.1.6 Memory Partitioning and User/Supervisor-Modes Related Features

A modular implementation of embedded systems that consist of both safety-related software components of different ASIL´s or of safety-related and non-safety-related software components is facilitated by AUTOSAR features that ensure freedom from interference between such software components.
Memory partitioning and user/supervisor-mode features and extensions added to the OS and the RTE functionality enable groups of SW-Cs running in separate user-mode memory partitions.
The memory partitioning and user/supervisor-modes related features address the following goal:

- Ensuring freedom from interference between software components by means of memory partitioning (e.g. memory-related faults in SW-Cs do not propagate to other software modules and SW-Cs executed in user-mode have restricted access to CPU instructions like e.g. reconfiguration).

This feature allows a broad variety of implementations in order to allow different technical safety concepts on the system- and software level.
Figure 2 shows a possible implementation whereas all BSWMs are executed in one trusted/supervisor-mode memory partition (highlighted in red in Figure 2). Some SW-Cs are logically grouped and put in separate non- trusted/user-mode memory partitions (highlighted in green). Selected SW-Cs belong to the same trusted/supervisor-mode memory partition as the BSWMs (see fourth SW-C in Figure 2 highlighted in red). There may be several non-trusted/user-mode partitions, each containing one or more SW-Cs.

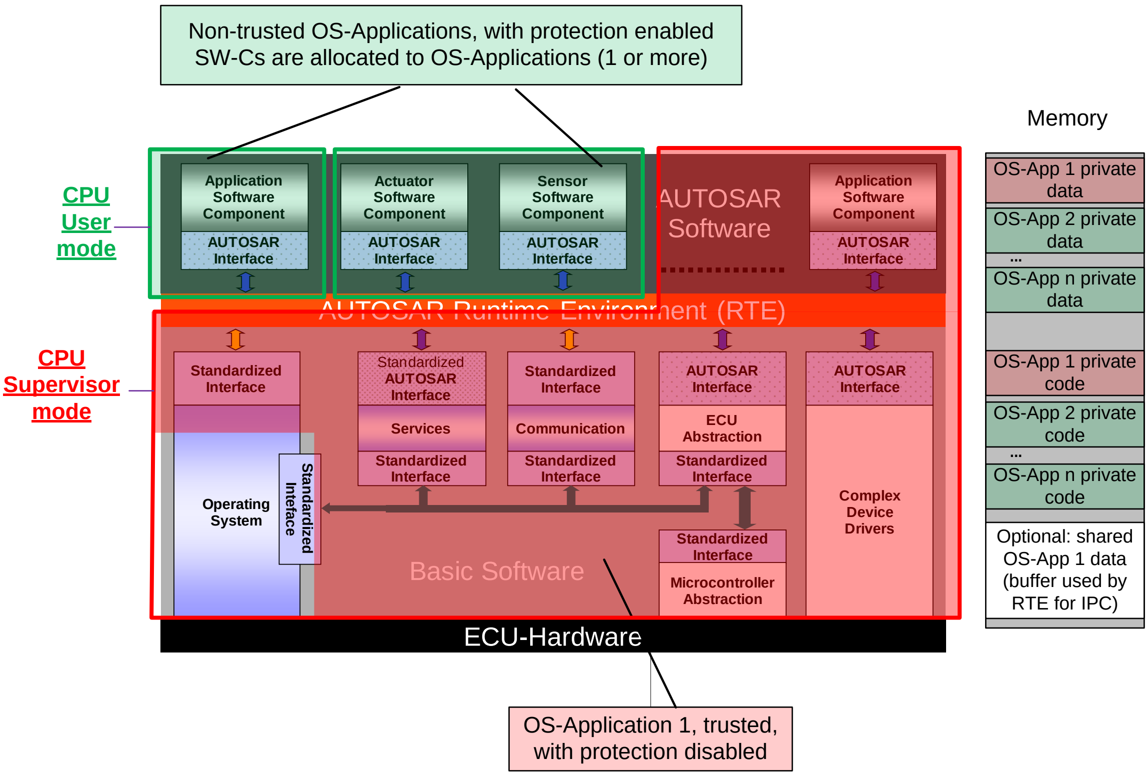


Figure 2: Memory partitioning and modes

The execution of trusted/supervisor-mode memory partition is not controlled by means of MMU/MPU hardware.
The memory access of SW-Cs executed in non-trusted/user-mode memory partitions is controlled by means of MMU/MPU hardware.

In case of a memory access violation or a CPU instruction violation in a non-trusted/user-mode partition, the OS and the RTE handle such violation by this erroneous software partition. Such error handling is either shut down or restart of all SW-Cs of this partition.

Memory partitioning and user-modes bring a possibility to have SW-C of different ASIL (or non-ASIL) on the same ECU, helping to achieve the interference freeness.

1.2 Relation between functional safety and AUTOSAR

The implementation of safety-related embedded systems using AUTOSAR needs to comply with the relevant functional safety standard for road vehicles.

To support the demonstration of compliance with such a standard, traceability from the safety requirements of the safety-related system or its elements and their respective implementations by AUTOSAR can be established using this document.

The bases of functional safety are the avoidance of faults (e.g. systematic software faults) or else the detection and handling of faults (e.g. random hardware faults) in order to mitigate their effects and thus prevent the violation of a safety goal by the embedded system.
AUTOSAR provides appropriate features and supports a systematic design approach but their functional safety compliant application is up to the user.

This document provides features of AUTOSAR that can be used to achieve functional safety. The approach during development of AUTOSAR related to functional safety is comparable to a Safety Element out of Context (SEooC) approach as described in ISO DIS 26262-10, chapter 10.

The SEooC approach is that safety goals or safety requirements of the targeted element (e.g. a software unit) are replaced by assumptions (see figure below). These assumed assumptions (e.g. failure modes to be detected and handled in this software unit) are the basis for the implementation of such a generic software element.
When using such a generic element for the development of a specific safety-related system the consistency between the assumed requirements and the requirements of the specific system need to be ensured (e.g. by the integrator of such a software).

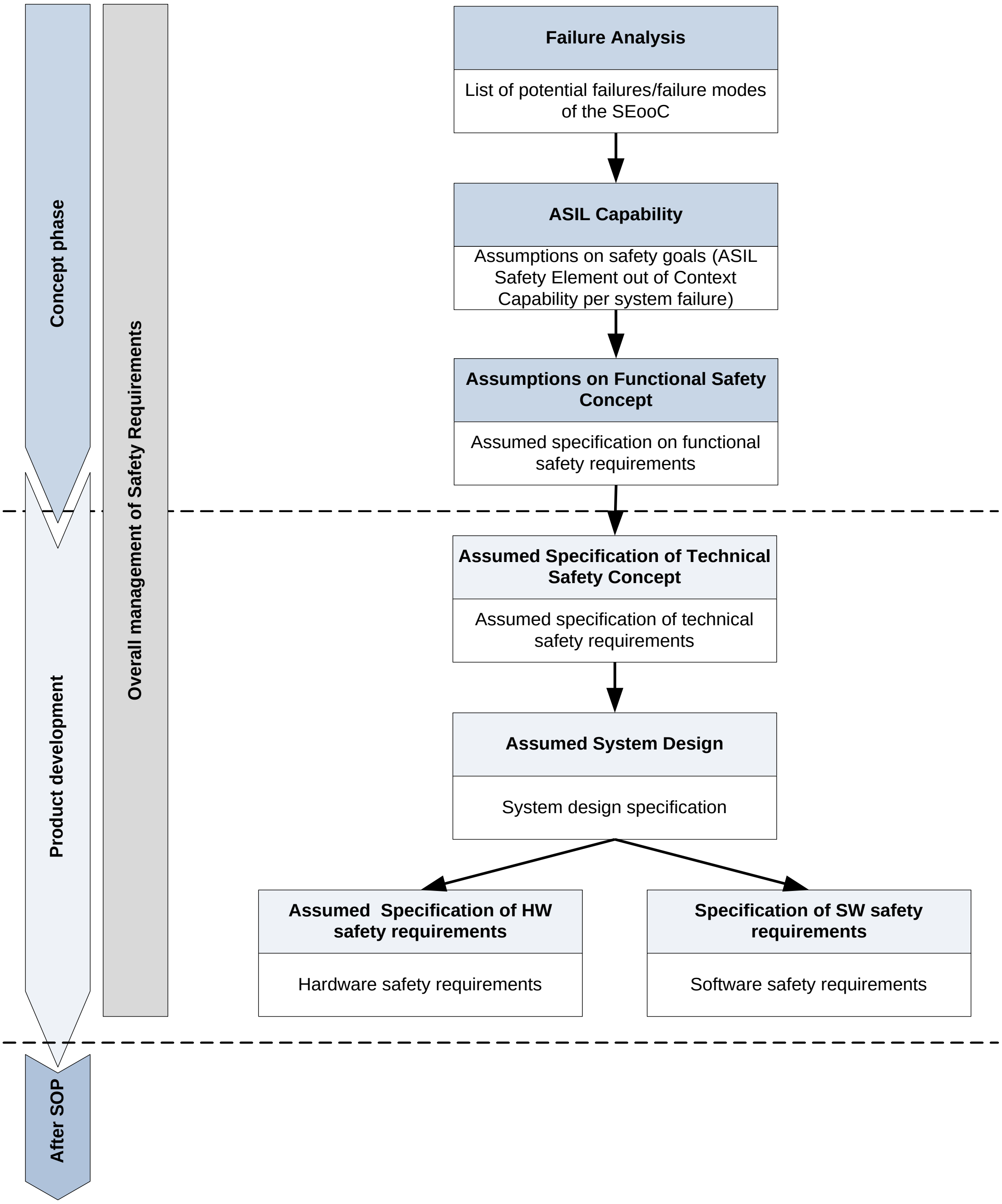


Figure 3: SEooC Development Lifecycle

The Figure 3 shows an example of such a generic break down of requirements. The lowest right box in Figure 3 represents the software requirements to be implemented either as SW-C or as a feature in a specific BSW.
Allowing time to define a SEooC will assist the correct implementation of safety requirements.

2 Scope of the document

- The document is intended for users or AUTOSAR to ease systematic system- and software engineering approach.
- Providing the following information about the AUTOSAR specifications:
  - Overview of the features introduced in the AUTOSAR release 4.0 and their according technical safety requirements; and
  - For each Feature in the BSW&RTE the document shows the means by which the elements of AUTOSAR are implementing this feature (mapping BRF -> SWS)

Note :
These features, maybe extended with supplementary features shall be the object of subsequent AUTOSAR releases. Full traceability is provided in this document for the safety related features fully usable in the AUTOSAR release 4.0; traceability is not provided for features only partially covered.

This document only covers the technical aspects of the software development concerning functional safety; the process related aspects are not considered here.

3 Constraints and assumptions

AUTOSAR defines a software architecture and a supporting methodology intended to develop E/E systems for the automotive domain but cannot guarantee functional safety for such systems.
AUTOSAR provides mechanisms to support functional safety of software-based systems (e.g. by mitigation of failure modes).
The functional safety of a particular system built by using AUTOSAR can only be fully evaluated by considering its functionality, its context of use and its implementation.

Providing evidence that a system is safe means to show that the risk of failure is acceptable low. Doing so, the risk contribution from the E/E infrastructure needs to be assessed in detail. This risk is directly connected to the occurrence of faults in the E/E infrastructure.

AUTOSAR offers standard mechanisms to support functional safety during the design-phases at the system or software level.

The full responsibility for selecting and implementing appropriate safety mechanisms as described inside the AUTOSAR framework fully resides on the implementer.

4 Safety features argument of coverage

4.1 Program Flow Monitoring Related Features

4.1.1 Overview

Program flow monitoring is a mechanism to check the correct execution of software.
The focus of this concept is the detection of program flow errors, i.e. a divergence from the valid program sequence. An incorrect program flow occurs if one or more program instructions are processed either in the incorrect sequence, not in time or are not even processed at all. Program flow errors can for example lead to data inconsistencies, data corruption, or software failures.
Logical and temporal program flow monitoring is used in the automotive industry and mentioned e.g. in ISO DIS 26262 as a measure to detect failures of the processing units (i.e. CPU, microcontroller) and as measure for the detection of failures of the HW clock.

4.1.2 [RS_BRF_00131] Logical Program Flow Monitoring

Initiator:	AUTOSAR Safety Team
Date:	27.02.2006
Short Description:	Logical program flow monitoring
Importance:	High
Description:	Add logical program flow monitoring of SW-Cs and BSW modules by means of extension of Watchdog Manager. Logical monitoring of the execution sequence of a program enables the detection of errors that cause a divergence from the valid program sequence during the error-free execution of the application. An incorrect program flow occurs if one or more program instructions are processed either in an incorrect sequence or not even processed at all. During design phase the valid program sequences are identified and modeled. During runtime the component for Logical Monitoring of Program Sequence uses this model to supervise or monitor the proper execution of program sequences. In case a divergence is detected usually the system is reset. To reduce the overhead caused by logical monitoring of program sequence, in AUTOSAR it is possible to restrict the definition of Supervised Entities (SE) to safety-related tasks/runnables. At least those have to be monitored but non safety-related tasks can be monitored as well.
Rationale:	This enables to detect to the following faults: 1. Systematic software faults 2. Random hardware faults 3. Systematic hardware faults. Faults in execution of program sequences (i.e. invalid execution of program sequences) can lead to data corruption, process crashes, or fail-silence violations. Logical program flow monitoring is required/recommended/proposed by ISO 26262, IEC 61508, MISRA.
Use Case:	Example safety-related Software Modules: - Monitoring that important steps in SW-C's computation algorithm are

	executed.
Dependencies:	Other concepts depend on this feature, e.g. “Multi-microcontroller support”, “Defensive behavior”, “Time determinism”
Conflicts:	
Supporting Material:	It is important that the checking points are placed in the program correctly. This is done by the developer or by an application-level generator (both not in the scope of AUTOSAR). Logical monitoring of program flow can be defined in various ways, both using hardware and software resources. This concept proposes a method using both software and hardware: most of the work is done by Watchdog Manager BSW-M, and part of error handling (ECU reset) is done by a HW watchdog.

Coverage Criteria of the feature

The feature is considered fulfilled if:

ID	Description
RS_BRF_00131_CC01	The feature “Logical Program Flow Monitoring” is considered fulfilled if the solution can detect errors that cause a divergence from the intended program sequence.

Coverage justification

Coverage Criteria	Coverage Justification		
	BSW module	**Requirements**	**Justification**
RS_BRF_00131_CC01	AUTOSAR_SWS_WatchdogManager	SWS_WdgM_00119, SWS_WdgM_00120, SWS_WdgM_00121, SWS_WdgM_00122, SWS_WdgM_00223, SWS_WdgM_00196, SWS_WdgM_00197, SWS_WdgM_00198, SWS_WdgM_00199, SWS_WdgM_00242, SWS_WdgM_00246, WDGM247, WDGM248, WDGM249, WDGM250, WDGM251, SWS_WdgM_00252, SWS_WdgM_00263, SWS_WdgM_00271, SWS_WdgM_00273, SWS_WdgM_00274, ECUC_WdgM_00319, ECUC_WdgM_00320, ECUC_WdgM_00321, ECUC_WdgM_00322, ECUC_WdgM_00323, ECUC_WdgM_00324, ECUC_WdgM_00343, ECUC_WdgM_00344, ECUC_WdgM_00345, ECUC_WdgM_00350, ECUC_WdgM_00351	Logical program flow monitoring using predecessor successor relations, allowed transitions and checkpoints is included into the watchdog manager which complies with the coverage argument.

4.2 Timing Related Features

The timing related features can be divided into:

1. Features related to the provision of synchronized time bases
2. Features related to synchronization of processing of asynchronous processing units
3. Features to support time deterministic implementation of applications
4. Features to support protection against timing violations

4.2.1 Features related to the provision of synchronized time bases

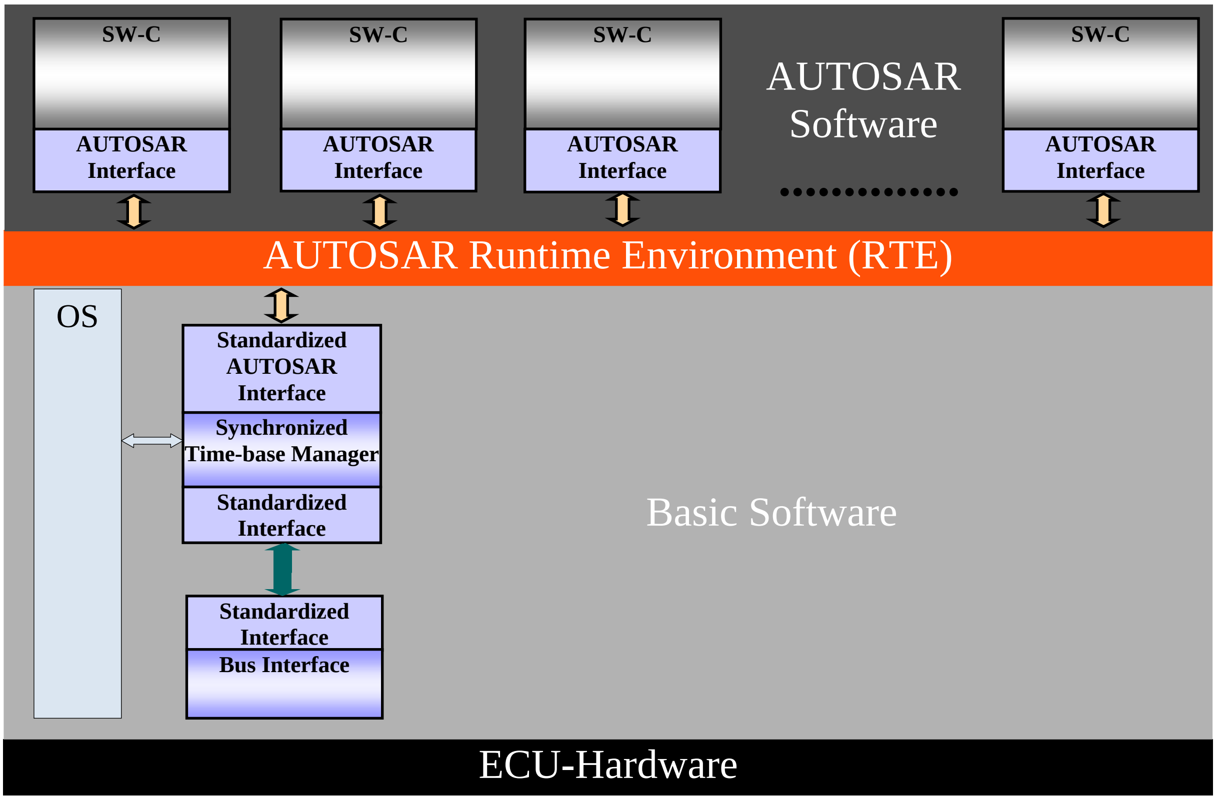
4.2.1.1 Overview

A synchronized time-base is a software time-base existing at a processing entity (e.g. a node of a distributed system) that is synchronized with software time-bases at different processing entities. A synchronized time-base can be achieved by time protocols or time agreement protocols that derive the synchronized time-base in a defined way from one or more physical time-bases. Examples are the network time protocol (NTP) and FlexRay time agreement protocol.

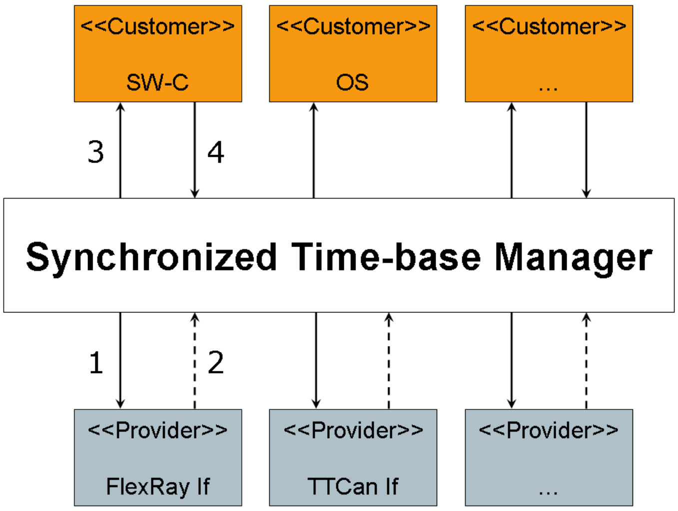
The synchronization will apply to the clock rate and optionally apply also to the clock absolute value.

A synchronized time-base allows synchronized action of the processing entities. Synchronized time-bases are often called “global time”, as e.g. the so called “FlexRay global time”. We do not use the term “global time” here because a single ECU sometimes has to cope with several synchronized time-bases which may vary in terms of rate and absolute value.

The synchronized time bases are established by the synchronized time-base manager BSW module.



Different types of customers will use the synchronized time-bases: triggered customers, active customers and notification customers. Triggering customers (runnables) is done via the OS.



4.2.1.2 [RS_BRF_00120] Provision of a synchronized time-base within a cluster

Initiator:	AUTOSAR Safety Team
Date:	27.02.2006
Short Description:	Provision of a synchronized time-base within a cluster
Importance:	High
Description:	AUTOSAR shall provide a synchronized time-base for a set of ECUs within a network cluster.
Rationale:	1/ To enable distributed SW-Cs to synchronize activities 2/ To detect and compensate for the incorrect clock of one of the ECUs

	3/ For deterministic behavior.
Use Case:	Four SW-Cs on four ECUs read wheel speed at the same time, for brake controlling algorithm.
Dependencies:	-
Conflicts:	-
Supporting Material:	Notes: 1. AUTOSAR can fulfill this requirement for systems using FlexRay or TTCAN time synchronization functionality. On other networks (e.g. using CAN) it will be more difficult to fulfill this requirement. 2. It is not constrained which networks shall be used. However, if a given network is used (e.g. CAN), then there shall be a compatible synchronization mechanism. 3. In AUTOSAR R4.0 support will be limited to FlexRay and TTCAN clusters. The extensions necessary to support this feature within CAN and LIN clusters are deferred to a later phase.

Coverage Criteria of the Feature

Constraint: Provision of synchronized time bases is restricted to FlexRay and TTCAN clusters in AUTOSAR Release 4.0.

The feature "Provision of a synchronized time-base within a cluster" is considered fulfilled if

ID	**Description**
RS_BRF_00120_CC01	There are *means to provide the synchronized time base for FlexRay and TTCAN clusters*
RS_BRF_00120_CC02	The time base is provided in a dependable way and faults are detected and handled.

Coverage justification

These 2 items are covered as follows

Coverage Criteria	**Coverage Justification**		
	BSW module	**Requirements**	**Justification**
RS_BRF_00120_CC01	AUTOSAR_SRS_SynchronizedTimeBaseManager AUTOSAR_SWS_SynchronizedTimeBaseManager	SRS_StbM_20005, SWS_StbM_00050, SWS_StbM_00080, SWS_StbM_00081, SWS_StbM_00015	*Means to provide a synchronized time base for FlexRay and TTCAN clusters*: A module "synchronized time-base manager" is introduced in the AUTOSAR basic software. This Module acquires the time base from the FlexRay or TTCAN interface.
RS_BRF_00120_CC02	AUTOSAR_SRS_SynchronizedTimeBaseManager AUTOSAR_SWS_SynchronizedTimeBaseManager	(SRS_StbM_20007, SWS_StbM_00030, SWS_StbM_00031, SWS_StbM_0003	*Provision of dependable time base and fault detection and handling*: a. The Synchronized Time-base Manager continuously provides the definition of time. If synchronization is not specified or temporarily not

	AUTOSAR_SRS_SynchronizedTimeBaseManager AUTOSAR_SWS_SynchronizedTimeBaseManager	2, SWS_StbM_00033, SWS_StbM_00034, SWS_StbM_00035, SWS_StbM_00036) SRS_StbM_20007, SWS_StbM_00030, SWS_StbM_00031, SWS_StbM_00032, SWS_StbM_00033, SWS_StbM_00034, SWS_StbM_00035, SWS_StbM_00036	available, the local time is provided instead. b. The Synchronized Time-base Manager detects loss and re-establishment of synchronized time-bases and erroneous customer calls and reports such faults to the DEM and the notification customers.

4.2.1.3 [RS_BRF_00127] Services for accessing to synchronized time-bases

Initiator:	AUTOSAR Safety Team
Date:	27.02.2006
Short Description:	Services for accessing to both local and global time
Importance:	High
Description:	AUTOSAR shall provide a service to access synchronized time bases, available to BSWMs and SWC-s.
Rationale:	To enable SWC-s to perform time-dependent actions, and in particular synchronization and monitoring.
Use Case:	A safety-related function may need to time the execution of a particular operation, or it may need to know exactly how much time has elapsed since a previous event. This timing information may also be compared or calculated with another task from another ECU and in order to achieve this both tasks must be using the same time-base.
Dependencies:	-
Conflicts:	-
Supporting Material:	Notes: 1/ Most safety related functions will be scheduled deterministically which means that they know exactly how much time has elapsed since it last started to run. However, there may be situations where more accurate timing is required within a task itself, or to help a task synchronize with another task on another ECU.

Coverage Criteria of the Feature

ID	Description

RS_BRF_00127_CC01	There are means that customers can use the synchronized time base. The following types of customers are to be considered: triggered customers, active customers and notification customers.

Coverage justification

This item is covered as follows

Coverage Criteria	Coverage Justification		
	BSW module	**Requirements**	**Justification**
RS_BRF_00127_CC01	Synchronized TimeBaseManager AUTOSAR_SRS_OS AUTOSAR_SWS_SynchronizedTimeBaseManager AUTOSAR_SWS_OS AUTOSAR_SRS_SynchronizedTimeBaseManager AUTOSAR_SWS_SynchronizedTimeBaseManager	SRS_StbM_20001, SRS_StbM_20002, SRS_StbM_20009, SRS_Os_11002, SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00025, SWS_StbM_00026, SWS_StbM_00028, SWS_StbM_00029, SWS_StbM_00037, SWS_StbM_00038, SWS_StbM_00077, SWS_StbM_00082, SWS_StbM_00083, SWS_StbM_00084, SWS_StbM_00085, SWS_Os_00206, SWS_Os_00201, SWS_Os_00013, SWS_Os_00199, SWS_Os_00227, SWS_Os_00429, SWS_Os_00430, SWS_Os_00431, SWS_Os_00462, SWS_Os_00463, SWS_Os_00435, SWS_Os_00415, SWS_Os_00416, SWS_Os_00436, SWS_Os_00437, SWS_Os_00438, SWS_Os_00417, SWS_Os_00418, SWS_Os_00419, SWS_Os_00420, SWS_Os_00421, SWS_Os_00422 SRS_StbM_20001, SRS_StbM_20003, SRS_StbM_20008, SRS_StbM_20010, SWS_StbM_00020, SWS_StbM_00025, SWS_StbM_00026, SWS_StbM_00028, SWS_StbM_00029, SWS_StbM_00037,	*Means that customers can use the synchronized time base*: a. For the triggered customer the BSW module “Synchronized Time-base Manager” provides a synchronization between the synchronized time-base and the time base used be the OS for scheduling, i.e. the OS counter b. For the active customer and the notification customer it means to provide a service interface via the RTE for SW-C or an API for BSW

		SWS_StbM_00038, SWS_StbM_00082, Chapter 11 in SWS StbM.	

4.2.1.4 [RS_BRF_00278] Sync AUTOSAR OS with Global Time from providing bus system in a well-defined way

Initiator:	BMW
Date:	31.01.2008
Short Description:	Sync AUTOSAR OS with Global Time from providing bus system in a well-defined way
Importance:	high | **medium** | low
Description:	It shall be possible to sync the AUTOSAR OS with the Global Time from providing bus system in a well defined and fast way
Rationale:	- For AUTOSAR Release 3.0, it is up to the implementer to write a “glue code” which is not a proper solution
Use Case:	- Enabling applications to run their tasks synchronous to the Global Time from providing bus system
Dependencies:	AUTOSAR OS
Conflicts:	--
Supporting Material:	--

Coverage Criteria of the Feature

The feature “Sync AUTOSAR OS with Global Time from existent bus system in a well defined way” is considered to be covered if

ID	Description
RS_BRF_00278_CC01	There are *means to provide the synchronized time base for FlexRay and TTCAN clusters*
RS_BRF_00278_CC02	Synchronization between the synchronized time-base and the time base used by the OS for scheduling is provided.

Coverage justification

These 2 items are covered as follows
.

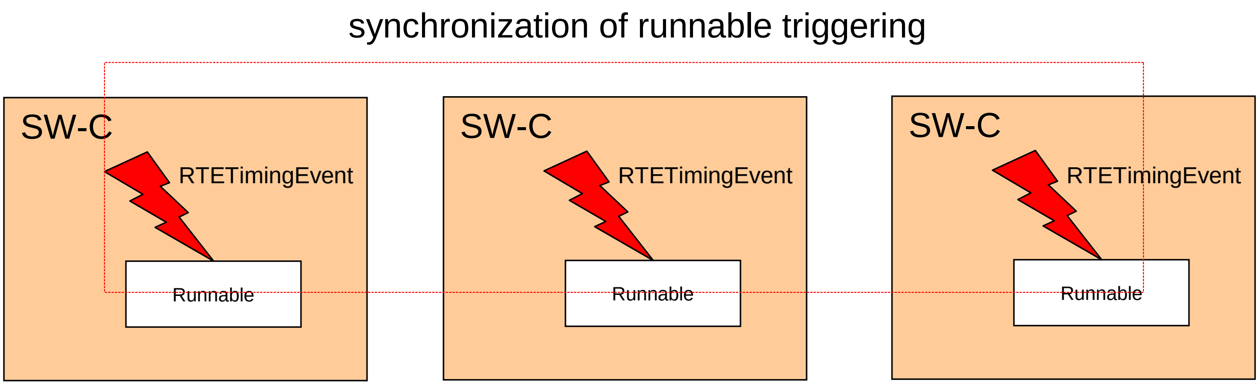
Coverage Criteria	Coverage Justification		
	BSW module	**Requirements**	**Justification**
RS_BRF_00278_CC01			is covered by RS_BRF_00120_CC01 (for further traceability see there)
RS_BRF_00278_CC02			is covered by RS_BRF_00127_CC01 (a) (for further traceability see there)

4.2.2 Features related to synchronization of processing of asynchronous processing units

4.2.2.1 Overview

To synchronize runnables within a set of SW-Cs, they have to be attached to a synchronized RTE timing. For this it must be possible to specify that a set of RTE timing events (with the same period) within a SW-C composition are synchronized.

Synchronization is possible within a single micro controller as well as across networks.



4.2.2.2 [RS_BRF_00126] Services for synchronization of SW-Cs

Initiator:	AUTOSAR Safety Team
Date:	27.02.2006
Short Description:	Services for synchronization of SW-Cs
Importance:	High
Description:	AUTOSAR shall provide mechanisms enabling SW-Cs on the same or different ECUs to synchronize their behavior
Rationale:	To enable runnables to respect their timing constraints.
Use Case:	1/ Two runnables must read data from a sensor in the same time window so that later they can vote on them; 2/ Two distributed SW-Cs (on different ECUs) perform synchronization.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

Coverage Criteria of the Feature

Constraints:

- The feature is restricted to RTE timing events only. The events are used to trigger runnables.
- The synchronization of runnables that are controlled by different AUTOSAR OS instances (e.g. if they are running on different ECUs or different µCs within

one ECU) is only possible if they are located on ECUs within the same FlexRay or TTCAN network cluster.

The feature “Services for synchronization of SW-Cs” is considered to be covered if

ID	Description
RS_BRF_00126_CC01	1. There are technical means to trigger runnables in a synchronized way, i.e. with minimum jitter and (in case of serialized processing) fixed execution order. The following cases have to be distinguished here: a. The runnables which are triggered by the synchronized timing events are mapped to the same operating system task. b. The runnables which are triggered by the synchronized timing events are mapped to different operating system tasks within one OS application. c. The runnables which are triggered by the synchronized timing events are mapped to different operating system tasks in different OS applications on the same microcontroller core. d. The runnables which are triggered by the synchronized timing events are mapped to different operating system tasks in different OS applications on different cores of the same microcontroller. e. The runnables which are triggered by the synchronized timing events are mapped to different operating system tasks in different OS applications on different microcontrollers within one ECU. The runnables which are triggered by the synchronized timing events are mapped to different operating system tasks in different OS applications on different microcontrollers within different ECUs.
RS_BRF_00126_CC02	The AUTOSAR methodology supports the specification of synchronization constraints for RTE timing events.

Coverage justification

These 2 items are covered as follows:

Coverage Criteria	Coverage Justification		
	BSW module	**Requirements**	**Justification**
RS_BRF_00126_CC01	AUTOSAR_SRS_RTE AUTOSAR_SWS_RTE	SRS_Rte_00232, rte_sws_7804, rte_sws_7805	a-c. In these cases the RTE configuration and RTE generation will take care of the synchronization of the runnables by either locating the runnables to the same task, using the same OsAlarm or OsScheduleTableExpiryPoint to implement all TimingEvents, or using different OsAlarms or OsScheduleTableExpiryPoints in different OsScheduleTables based on different OS counters but with same period and max value.
	AUTOSAR_SRS_RTE AUTOSAR_SWS_RTE	SRS_Rte_00232, rte_sws_7804 covered by RS_BRF_00120	d-f. In these cases, the RTE configuration and RTE generation will take care of the synchronization of the runnables by using OsScheduleTable ExpiryPoints in different explicitly synchronized OsScheduleTables (). Furthermore the synchronized time-base manager will take care of the explicit synchronization of the schedule tables and of the establishment of the common

		and RS_BRF_00127	synchronized time base .
RS_BRF_00126_CC02	AUTOSAR_RS_TimingExtensions	RSTM002 chapter 3.7 in	The specification of synchronization constraints is supported by the timing extensions.

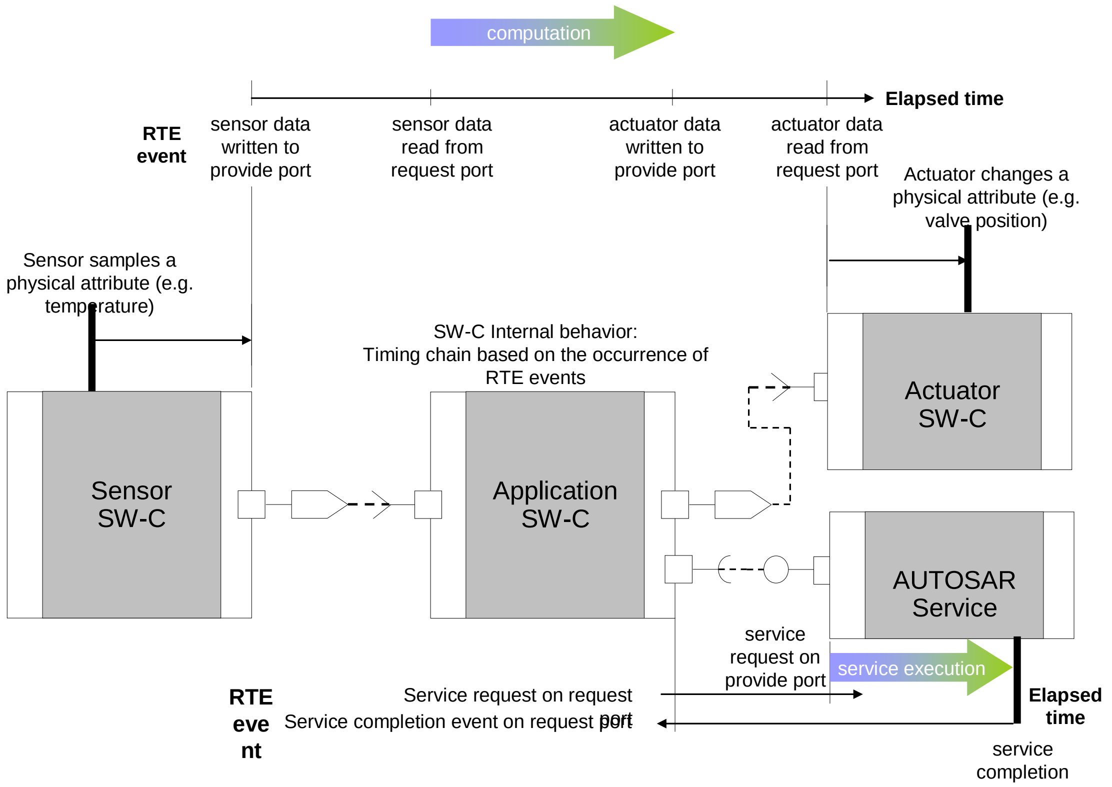
4.2.3 Features to allow time deterministic implementation of applications

4.2.3.1 Overview

Time deterministic implementation of applications requires to be able to specify timing constraints and analyse timing properties at different stages of development, i.e. during virtual integration on VFB level, development of SW-Cs, and finally the integration of SW-Cs into ECUs and of ECUs into a system of ECUs.

Furthermore, the runtime environment must provide suitable mechanisms to enforce deterministic timing.

The following Figure illustrates a specification of VFB timing.



4.2.3.2 [RS_BRF_00122] Support for timing constraints

Initiator:	AUTOSAR Safety Team
Date:	09.05.2007
Short Description:	Support for upper bounds on timing.
Importance:	High
Description:	It shall be possible to develop implementations based on AUTOSAR with verifiable timing constraints on jitter, latency and execution time. This means that task and communication scheduling strategies shall not contradict this. The requirement relates to task scheduling, communication scheduling and responsiveness to external events.
Rationale:	--
Use Case:	--
Dependencies:	RS_BRF_00121
Conflicts:	--
Supporting Material:	--

Coverage Criteria of the Feature

The feature “Support for timing constraints” is considered to be covered if

ID	Description
RS_BRF_00122_CC01	1. It is possible to specify the following timing constraints: a. a timing relation (min, max, nominal) between RTE events with a lower and upper bounds b. the time relation between a physical sensor acquisition (or a physical actuator change) and the availability of the corresponding data element on the port of a sensor SW-C (or actuator SW-C) c. constraints on the execution time (min,max) of a runnable d. constraints on the triggering rate for a runnable e. the end-to-end timing related to external communication f. the end-to-end timing related to IO accesses
RS_BRF_00122_CC02	2. The scheduling strategies allow to enforce these timing constraints by providing the following mechanisms: a. specification of non-preemptive execution of a code segment b. static time-based scheduling for all tasks or for a subset of the tasks c. the possibility to replace ISRs with time-based polling routines d. fixed-priority based scheduling e. the possibility of preemption of lower-priority tasks by higher-priority tasks

These 2 items are covered as follows:

Coverage Criteria	**Coverage Justification**		
	BSW module	**Requirements**	**Justification**
RS_BRF_00122_CC01	AUTOSAR_RS_TimingExtensions	RSTM002, RSTM003, RSTM004,	1. The specification of timing constraints and properties is possible using the AUTOSAR timing extensions as follows: a. The AUTOSAR timing extensions allow the

	AUTOSAR_Specification_of_TimingExtensions	sections 3.3, 3.6 in AUTOSAR_Specification_of_TimingExtensions	specification of event chains and of the triggering behavior of event chains.
	AUTOSAR_RS_TimingExtensions AUTOSAR_Specification_of_TimingExtensions	RSTM012 section 3.6 in AUTOSAR_Specification_of_TimingExtensions	b. The AUTOSAR timing extensions allow the specification of sensor/actuator delays.
	AUTOSAR_RS_TimingExtensions AUTOSAR_Specification_of_TimingExtensions	RSTM001, RSTM002 sections 3.2, 3.6 AUTOSAR_Specification_of_TimingExtensions	c. The AUTOSAR timing extensions allow the specification of timing events of SW-C internal behavior like start and termination of runnables and the specification of timing constraints related to these.
	AUTOSAR_RS_TimingExtensions AUTOSAR_Specification_of_TimingExtensions	RSTM001, RSTM002 sections 3.2, 3.5 in AUTOSAR_Specification_of_TimingExtensions	d. The AUTOSAR timing extensions allow to specify event triggering constraints.
	AUTOSAR_RS_TimingExtensions AUTOSAR_Specification_of_TimingExtensions	RSTM001, RSTM002 sections 3.2, 3.6 in AUTOSAR_Specification_of_TimingExtensions	e. The AUTOSAR timing extensions allow to specify timing events related to bus communication and timing constraints for these.
		RSTM001, RSTM004 sections 3.2, 3.6 in AUTOSAR_Specification_of_TimingExtensions	f. The AUTOSAR timing extensions allow to specify input/output latency constraints.

RS_BRF_00122_CC02	AUTOSAR_SRS_OS AUTOSAR_SWS_OS AUTOSAR_SRS_OS AUTOSAR_SWS_OS AUTOSAR_SRS_OS AUTOSAR_SWS_OS	SRS_Os_00097 SWS_Os_00001 SRS_Os_00098 SWS_Os_00002, SWS_Os_00007 SRS_Os_00097 SWS_Os_00001	2. The OS and the RTE provide the necessary scheduling mechanisms to enforce timing as follows: a. Non-preemptive scheduling is supported by OSEK OS. b. The Operating System provides statically configurable schedule tables based on time tables. c.-e. These features are available with OSEK OS.

4.2.3.3 [RS_BRF_00123] Responsiveness to external events

Initiator:	AUTOSAR Safety Team
Date:	09.05.2007
Short Description:	Responsiveness to external events
Importance:	High
Description:	AUTOSAR shall enable the use of external events as an initiator for scheduling.
Rationale:	As certain external events require a timely response to ensure correct behavior these events must be able to initiate tasks.
Use Case:	Schedules driven by ticks calculated from angles of an engine's crankshaft.
Dependencies:	--
Conflicts:	--
Supporting Material:	External events include IO and interrupts

Coverage Criteria of the Feature

The feature "Responsiveness to external events" is considered to be covered if

ID	**Description**
RS_BRF_00123_CC01	External events can be used as an initiator for scheduling.

This item is covered as follows:

Coverage Criteria	**Coverage Justification**		
	BSW module	**Requirements**	**Justification**
RS_BRF_00123_CC01	AUTOSAR_SRS_RTE AUTOSAR_SWS_RTE	SRS_Rte_00162, SRS_Rte_00216, rte_sws_7229, rte_sws_7212, rte_sws_7213, rte_sws_7214, rte_sws_7543, rte_sws_7215, rte_sws_7216,	1. The RTE supports the use of external events as trigger execution of runnables and BSW schedulable entities.

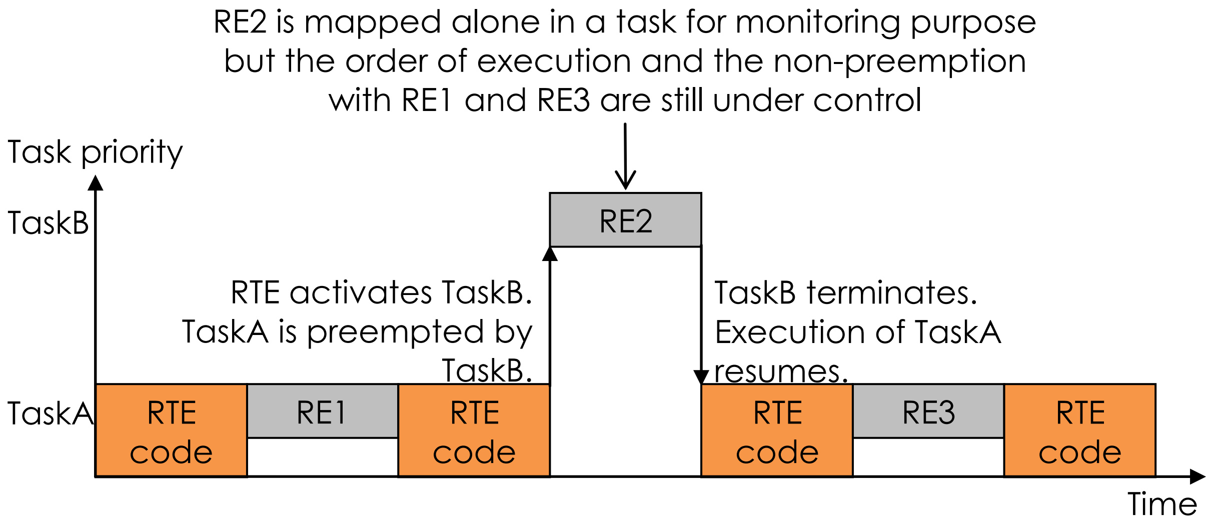
		rte_sws_7218, rte_sws_7200, rte_sws_7201, rte_sws_7207, rte_sws_7514, rte_sws_7542, rte_sws_7544, rte_sws_7545, rte_sws_7548, rte_sws_7546, rte_sws_7549, rte_sws_7282, rte_sws_7283	

4.2.4 Features related to protection against timing violation

4.2.4.1 Overview

Depending on the scalability class, the AUTOSAR OS can provide protection mechanisms against timing violation As the OS is only aware of tasks and not of runnables, the OS provides protection mechanisms on task level with the fault containment region being the OS application.

Timing protection of SW-Cs at runtime requires monitoring of runnables and preventing the propagation of timing faults from one SW-C to another. If SW-Cs require protection from each other, then their runnables have to be placed into different OS applications which imlies that they are placed into different task bodies.



4.2.4.2 [RS_BRF_00121 Runtime timing protection and monitoring

Initiator:	AUTOSAR Safety Team
Date:	27.02.2006
Short Description:	Runtime timing protection

Importance:	High
Description:	AUTOSAR shall provide statically configured runtime timing protection and monitoring. This includes monitoring that tasks are dispatched at the specified time, meet their execution time budgets, and do not monopolize OS resources.
Rationale:	To guarantee that safety-related functions will execute within their timing constraints. Tasks monopolizing the CPU shall be detected and handled (like heavy ECU load, many interrupt requests).
Use Case:	If deadline of a task is not fulfilled, then it may be restarted or an error is reported.
Dependencies:	--
Conflicts:	--
Supporting Material:	Notes: 1/ Monitoring of task execution detects scheduler misbehavior (i.e. deviations from real-time); 2/ As runnables are mapped to tasks, runnable monitoring can be done either in a cumulative manner or by assigning single runnables to tasks in ECU configuration.

Coverage Criteria of the Feature

The feature “Runtime timing protection and monitoring” is considered to be covered if:

ID	**Description**
RS_BRF_00121_CC01	The operating system provides mechanisms to detect timing faults on task level and to prevent timing faults from propagating from one OS application to another
RS_BRF_00121_CC02	The RTE provides means to make use of the task level OS timing protection mechanisms for runnables.

These 2 items are covered as follows:

Coverage Criteria	**Coverage Justification**		
	BSW module	**Requirements**	**Justification**
RS_BRF_00121_CC01	AUTOSAR_SRS_OS AUTOSAR_SWS_OS	SRS_Os_11008, SWS_Os_00028, SWS_Os_00089, SWS_Os_00033, SWS_Os_00037, SWS_Os_00048, SWS_Os_00064, SWS_Os_00465, SWS_Os_00469, SWS_Os_00470, SWS_Os_00471, SWS_Os_00472, SWS_Os_00473, SWS_Os_00474	The OS provides means to monitor execution time budgets, task activation frequencies, and resource locking times, and allows preventing fault propagation by stopping OS applications and freeing locked resources
RS_BRF_00121_CC02	AUTOSAR_SRS_RTE AUTOSAR_SWS_RTE	SRS_Rte_00160, SRS_Rte_00193, rte_sws_2697, sws_rte_7800, sws_rte_7802 in 084	The RTE provides debounced start of runnable entities and supports runnable execution chaining in order to allow a separation of runnables (which usually

			are chained within one task body) into chained tasks which then can be monitored by the task level OS mechanisms

4.2.4.3 [RS_BRF_00125] Monitoring of local time

Initiator:	AUTOSAR Safety Team
Date:	27.02.2006
Short Description:	Monitoring of local time
Importance:	High
Description:	AUTOSAR shall provide a mechanism that monitors ECU local time.
Rationale:	This is a necessary basis for deterministic execution of safety functions and for detection of failures of the system by safety integrity functions, within the guaranteed time intervals.
Use Case:	The local time is monitored to guarantee the correct timing of the safety-related runnables on the ECU.
Dependencies:	-
Conflicts:	-
Supporting Material:	Notes: 1/ This measure normally require an independent clock. This may be implemented with a HW watchdog. Alternatively, a different ECU with its local time could be used as a watchdog. Yet another solution could be to use an ADC and capacitor.

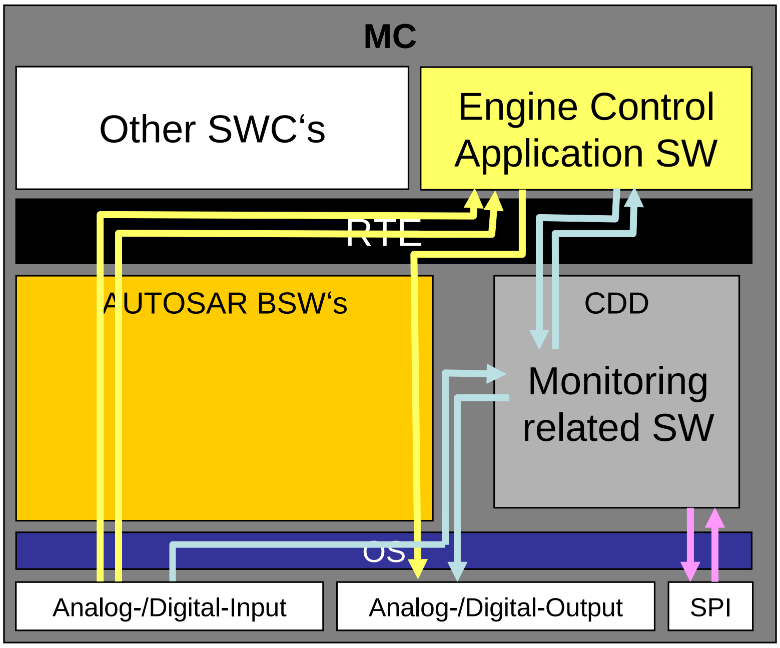
This feature is considered fulfilled as the functionality can be realized within the software component. There is no need for specific mechanisms in AUTOSAR.

4.3 E-Gas Monitoring Related Features

4.3.1 Overview

The possible realizations of the e-Gas monitoring concept in the context of AUTOSAR software architecture have been investigated. The features of this section ensure that a design approach as shown in the following figure can be used with AUTOSAR Release 4.0.

In the design approach shown below, the monitoring related software is located in a Complex Driver (CDD). A CDD allows a direct access to the related inputs and outputs.



4.3.1.1 [RS_BRF_00243] Communication protections against corruption and loss of data

Initiator:	AUTOSAR Safety Team
Date:	23 Nov 2007
Short Description:	Communication protections against corruption and loss of data
Importance:	High
Description:	If the responsibility of detection is placed in application, AUTOSAR BSW must provide a mechanism to transmit the communication protections against a corruption or a loss of data to the application (end to end protection protocol). If the responsibility of detection is placed in Complex Drivers, AUTOSAR BSW must provide a mechanism to transmit the communication protections against a corruption or a loss of data to the Complex Drivers.
Rationale:	If the Basic Software is responsible of the transmitted or the received secure data, AUTOSAR BSW must provide such mechanisms.
Use Case:	Applicable for bus system that carries Safety related data.
Conflicts:	--
Supporting Material:	--

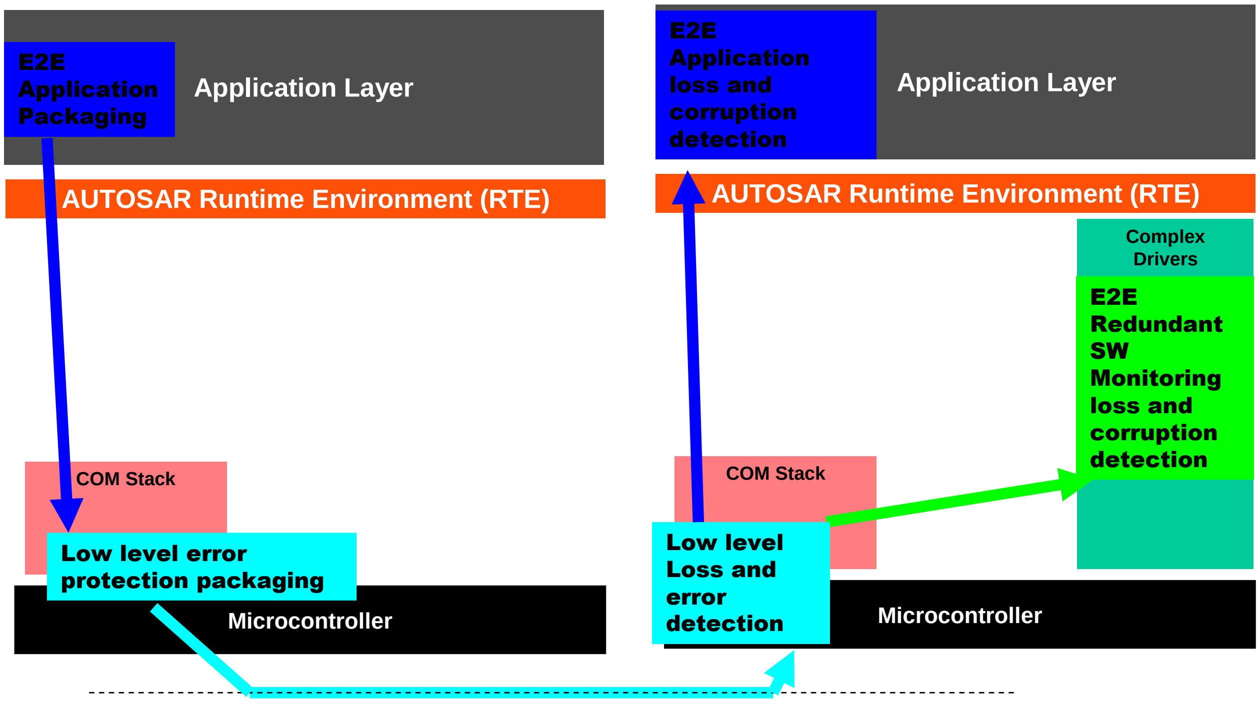
Coverage Criteria of the Feature

Constraint: It is assumed that end-to-end protection is used to protect the transmission of the necessary signals from the sender to the receiver (e.g. monitoring software).

The feature “Communication protection against corruption and loss of data” is considered fulfilled if the complete path of the data read by the Complex Drivers is protected against loss and corruption, which means:

ID	**Description**
RS_BRF_00243_CC01	the loss and corruption of data is detected if it happens on the way from the emitter node to the BSW driver of the receiver node
RS_BRF_00243_CC02	the loss and corruption of data is detected if it happens on the way from the

	Bus Specific interface to the Complex Driver



These 2 items are covered as follows

Coverage Criteria	Coverage Justification		
	BSW module	**Requirements**	**Justification**
RS_BRF_00243_CC01	AUTOSAR_SRS_Libraries AUTOSAR_SWS_E2ELibrary	SRS_LIBS_08527, SRS_LIBS_08536, SWS_E2E_00020, E2E0023, E2E0026, E2E0030, E2E0043	the detection of loss and corruption of data between the emitter node and the BSW of the receiver node is ensured by the protection mechanisms available with CAN or FlexRay communication networks (CRC, checksum, process counters)
RS_BRF_00243_CC02	AUTOSAR_SRS_Libraries AUTOSAR_SWS_E2ELibrary	SRS_LIBS_08535, E2E0026, E2E0030	the detection of loss and corruption of data between the Bus Specific Interface and the Complex Driver is ensured by the access of the Complex Driver to the frame payload dedicated to Safety and the

			application dependant end-to-end protection.

4.3.1.2 [RS_BRF_00251] Priority access to SPI bus

Initiator:	AUTOSAR Safety Team
Date:	23 Nov 2007
Short Description:	Priority access to SPI Bus
Importance:	
Description:	Exclusive / Priority access to SPI bus should be granted to software modules that carry out timing-critical monitoring protocols between the main controller and a monitoring unit connected via SPI bus. This should be possible for both these software modules being included in an AUTOSAR software component, and these modules being included in a Complex Driver.
Rationale:	We expect that there will be systems executing monitoring protocols (for example as described by the standardized E-Gas Monitoring Concept) as well as other communication via a single SPI bus. The other communication is expected to be driven by AUTOSAR components or BSW modules using the standard AUTOSAR interfaces. The monitoring protocol shall be executed as needed (with priority) otherwise an availability penalty would be imposed. Note: The E-Gas Monitoring Concept is standardized by the AKEGAS working group and not part of the AUTOSAR standard. It is used as an exemplary item here because it is a standardized automotive safety concept.
Use Case:	Carrying out a monitoring protocol in parallel with other communication on an SPI bus.
Conflicts:	--
Supporting Material:	Standardized e-Gas monitoring concept for engine management systems of gasoline and diesel engines, V 2.0, 29.04.2004

Coverage Criteria of the Feature

The feature “Priority access to SPI bus is considered fulfilled if:

ID	**Description**
RS_BRF_00251_CC01	the Monitoring SW placed in the Complex Drivers SW can have access the SPI bus with a bounded delay, this means that the priority access is scheduled so that the delay of the access to the SPI from CDD is bounded.

This item are covered as follows:

Coverage Criteria	**Coverage Justification**		
	BSW module	**Requirements**	**Justification**
RS_BRF_00251_CC01	AUTOSAR_SRS_SPIHandlerDriver AUTOSAR_SWS_SPIHandlerDriver	SRS_Spi_12037 , SWS_Spi_00002 SWS_Spi_00014, SWS_Spi_00093, SWS_Spi_00059	Priority access is defined in and provided by the SPI Handler Driver

4.3.1.3 [RS_BRF_00248] Testing and monitoring of I/O data and I/O HW

Initiator:	Safety Team
Date:	27.02.2006

Short Description:	Testing and monitoring of I/O data and I/O HW
Importance:	High
Description:	AUTOSAR shall allow the use of mechanisms for the testing and monitoring of I/O HW elements as well as the safety-related values received/transmitted using the I/O HW elements.
Rationale:	To detect errors in measured sensor data or output actuator data, and to detect failures in I/O HW.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

Coverage Criteria of the Feature

The feature "Testing and monitoring of I/O data and I/O HW" is considered fulfilled if:

ID	**Description**
RS_BRF_00248_CC01	The Monitoring SW placed in the Complex Drivers SW can perform test of the related A/D-Converter without disturbing a data acquisition related to normal operation.
RS_BRF_00248_CC02	The Monitoring SW placed in the Complex Drivers can directly perform tests of the safety-related actuators (throttle, injectors) of the shut-off path.

These 2 items are covered as follows:

Coverage Criteria	**Coverage Justification**		
	BSW module	**Requirem ents**	**Justification**
RS_BRF_00248_CC 01			Support for ADC tests is ensured because it doesn't have any impact on ADC drivers.
RS_BRF_00248_CC 02			The drivers dedicated to the injectors and the throttle actuator are Complex Drivers.and therefore can implement the necessary test procedures.

4.3.1.4 [RS_BRF_00301] Ability to make an AUTOSAR application compatible to the e-Gas monitoring Concept

Initiator:	AUTOSAR Safety Team
Date:	25 Jan 2008
Short Description:	Ability to make an AUTOSAR application compatible to the e-Gas monitoring concept
Importance:	High
Description:	It must be possible for an application to respect the safety concept known as e-GAS monitoring concept and to use the AUTOSAR standard. Note: The E-Gas Monitoring Concept is standardized by the AKEGAS working group and not part of the AUTOSAR standard. It is used as an exemplary item here because it is a standardized automotive safety concept. The feature requires that AUTOSAR standard must not make the use of the E-Gas Monitoring Concept impossible.

Rationale:	A complete analysis has been done; the result is a small set of requirements which cover the two main hypothesis considered by the e-Gas experts in the AUTOSAR safety team.
Use Case:	The e-Gas monitoring concept is a standardized automotive safety concept.
Conflicts:	
Supporting Material:	Standardized e-Gas monitoring concept for engine management systems of gasoline and diesel engines, V 2.0, 29.04.2004

Coverage Criteria of the Feature

The feature [RS_BRF_00301] Ability to make an AUTOSAR application compatible to the e-Gas monitoring Concept is covered if:

ID	Description
RS_BRF_00301_CC01	The arguments of the [RS_BRF_00243], [RS_BRF_00251], [RS_BRF_00248], [BRF00244], [BRF00245], [BRF00246], [BRF00247], [BRF00249], [BRF00250] are fulfilled.
RS_BRF_00301_CC02	The e-Gas Monitoring SW placed in the Complex Drivers can access to the raw values of the ADC inputs.
RS_BRF_00301_CC03	The e-Gas Monitoring SW placed in the Complex Drivers can access to the raw values of the DIO inputs.
RS_BRF_00301_CC04	The e-Gas Monitoring SW placed in the Complex Drivers can access to the raw values of the PWM inputs.

These 4 items are covered as follows

Coverage Criteria	Coverage Justification		
	BSW module	**Requirements**	**Justification**
RS_BRF_00301_CC01			The features [RS_BRF_00243], [RS_BRF_00251], [RS_BRF_00248], [BRF00244], [BRF00245], [BRF00246], [BRF00247], [BRF00249], [BRF00250] are fully covered.
RS_BRF_00301_CC02	AUTOSAR_SRS_ADCDriver AUTOSAR_SWS_ADCDriver	SRS_SPAL_12063, SWS_Adc_00113	ADC Drivers can provide raw data directly to the Complex Drivers
RS_BRF_00301_CC03	AUTOSAR_SRS_DIODriver AUTOSAR_SWS_DIODriver	SRS_Dio_12352, SWS_Dio_00083	DIO Drivers can provide raw data directly to the Complex Drivers
RS_BRF_00301_CC04	AUTOSAR_SRS_ICUDriver AUTOSAR_SWS_ICUDriver	SRS_Icu_12436 SWS_Ico_00211, SWS_Ico_00342, SWS_Ico_00084, SWS_Ico_00344, SWS_Ico_00106, SWS_Ico_00345, SWS_Ico_00180, SWS_Ico_00181, SWS_Ico_00022,	ICU Drivers can provide raw data directly to the Complex Drivers

		SWS_Ico_00048, ICU272, ICU265 SRS_Icu_12369 SWS_Ico_00021	

4.4 Communication Stack Related Features

4.4.1 Overview

Features related to Communication Stack aim at enhancing fault detection in order to cover communication failure modes which are not currently covered by existing mechanisms, and also providing possible recovery through redundancy.

4.4.2 Related Features

4.4.2.1 [RS_BRF_00111] Data sequence control

Initiator:	AUTOSAR Safety Team
Date:	27.02.2006
Short Description:	Data flow control
Importance:	High
Description:	AUTOSAR shall provide mechanisms for data sequence control.
Rationale:	Receivers must have the possibility to check whether a signal is received in sequence.
Use Case:	A distributed safety related powertrain control system receives a torque request signal via CAN with a sequence counter with a value higher than expected. This error is interpreted as several messages have been lost and there might be an inconsistent state within the powertrain system. This is handled with a reinitialization of the powertrain system.
Dependencies:	--
Conflicts:	--
Supporting Material:	Notes: 1/ This can be achieved by adding sequence numbers (like PDU counter) to signals or frames. 2/ If the receiver detects a wrong sequence, it may decide for example to discard the message or reinitialize communication.

Coverage Criteria of the feature

The feature is considered fulfilled if:

ID	Description
RS_BRF_00111_CC01	There are means to detect “out of sequence” messages.
RS_BRF_00111_CC02	This detection can be used only for transmission of safety-related data.
RS_BRF_00111_CC03	This detection is realized by the AUTOSAR framework (without involving

	the application).
RS_BRF_00111_CC04	Error handling is performed in case of “out of sequence” messages detected.

Coverage justification

Coverage Criteria	Coverage Justification		
	BSW module	**Requirements**	**Justification**
RS_BRF_00111_CC01	AUTOSAR_SWS_COM	COM587, COM588, COM590, COM687, COM688, COM726, COM727	Detection of “out of sequence” messages is realized by the implementation of a new safety mechanism called “I-PDU counter”
RS_BRF_00111_CC02	AUTOSAR_SWS_COM	ECUC_Com_00592, ECUC_Com_00593, ECUC_Com_00594, ECUC_Com_00595, ECUC_Com_00003	This I-PDU counter mechanism is a configurable option and thus can only be used for I-PDUs containing safety-related signals
RS_BRF_00111_CC03	AUTOSAR_SWS_COM	COM587, COM588, COM687, COM688	This I-PDU counter is handled by BSW COM module i.e. incremented by the sender COM module before transmission of a safety-related I-PDU and checked by the receiver COM module
RS_BRF_00111_CC04	AUTOSAR_SWS_COM	COM590, COM726, COM727	In case of an I-PDU counter not matching its expected value, the COM module will discard the faulty I-PDU and provide notification by callback

4.4.2.2 [RS_BRF_00241] Multiple communication links

Initiator:	AUTOSAR Safety Team
Date:	27.02.2006
Short Description:	Multiple communication links
Importance:	High
Description:	AUTOSAR shall support multiple communication links.
Rationale:	To tolerate faults on one of the channels.
Use Case:	1/ If in a given system there is redundant communication HW (like two independent CAN buses, or one CAN and one FlexRay buses), then to provide fault tolerance, one can use a safety protocol on each channel (with data protected with checksum, address id, counter and timeout for example). Then, the receiver can do 1oo2 voting (i.e. take one of two correct received messages); 2/ If one channel completely fails the second channel may be used for reduced functionality communications.
Dependencies:	RS_BRF_00206
Conflicts:	-
Supporting Material:	Notes: 1/ This assumes that at configuration time, it is possible to statically configure which communication links are used.

Argument of the feature coverage

The feature is considered fulfilled if:

ID	Description
RS_BRF_00241_CC01	There are means to send a message on different communication links and to detect “corrupted” messages and eventually to recover from this failure mode.
RS_BRF_00241_CC02	This detection can be used only for transmission of safety-related data.
RS_BRF_00241_CC03	This detection is realized by the AUTOSAR framework (without involving the application).
RS_BRF_00241_CC04	Error handling is performed in case of “corrupted” messages detected.

Coverage justification

Coverage Criteria	Coverage Justification		
	BSW module	**Requirements**	**Justification**
RS_BRF_00241_CC01	AUTOSAR_SWS_COM	COM596, COM597	Detection of “corrupted” messages and recovery is realized by the implementation of a new safety mechanism called “I-PDU replication”
RS_BRF_00241_CC02	AUTOSAR_SWS_COM	ECUC_Com_00599, ECUC_Com_00600, ECUC_Com_00601	This I-PDU replication mechanism is a configurable option and thus can only be used for I-PDUs containing safety-related signals
RS_BRF_00241_CC03	AUTOSAR_SWS_COM	COM596, COM597	Replicated I-PDUs are handled by BSW COM module i.e. replicas are compared by the receiver COM module which performs a K out of N voting
RS_BRF_00241_CC04	AUTOSAR_SWS_COM	COM596, COM597	Depending on the result of the voting algorithm, the COM module will discard faulty I-PDUs and process correct ones

4.5 E2E communication protection Related Features

4.5.1 Overview

In an embedded system the exchange of data between a sender and the receiver(s) can affect functional safety if its functional safety depends on the integrity of such data. Therefore such data are transmitted using mechanisms to protect them against the effects of faults within the communication link.

The End-to-End Communication Protection related features are implemented in AUTOSAR 4.0 as a standard library providing E2E communication protection mechanisms that enable sender to protect such data and the receiver to detect and handle errors in the communication link at runtime.

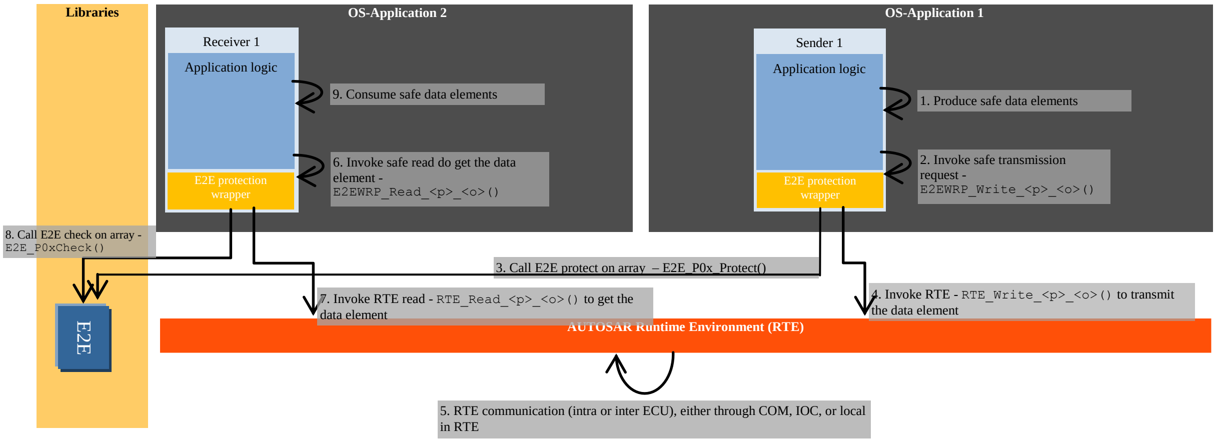


Figure 4: End-to-End Protection

The mechanism in Figure 4 is described below:

- For each RTE Write or Read function that transmits safety-related data (like Rte_Write_<p>_<o>()), there is the corresponding E2E protection wrapper function.
- The E2E protection wrapper creates a structured data element and invokes the AUTOSAR E2E Library for either the protection or the verification of safety-related data;

The E2E protection wrapper is invoked by the related Software Component;

4.5.2 Related Features

Features related to E2E communication aim to protect safety-related data exchange among SW-Cs

4.5.2.1 [RS_BRF_00114] SW-C end-to-end communication protection

Initiator:	Safety Team
Date:	27.02.2006
Short Description:	SW-C end-to-end communication protection
Importance:	High
Description:	Within this concept (feature), we define the extensions to RTE and configuration to support end-to-end safe communication between SW-Cs located on remote ECUs. End-to-end communication protection is a state-of-art in a big group of safety-related systems in different industries, including automotive. Currently, some existing network stacks provide a subset of mechanisms used by safety protocol (e.g. checksum). However, the purpose of these mechanisms are availability and fault tolerance, but not safety (FlexRay is partially an exception). Logically, the concept creates a layer between VFB and SW-Cs. This is realized by means of:

	1/ safety protocol library – a set of stateless library functions that verify the communication (e.g. if a CRC of a message is correct or is it on time), and which are invoked by RTE or SW-Cs, 2/ introduction of additional configurable attributes (fields) for SW-C ports (e.g. port address), used by safety protocol library. The port attributes keep the state information of the communication, whereas the stateless library function does the checks. Thanks to these extensions, any inter-ECU communication can be possibly used to transmit safety-related data. The safety protocol will work on any network/bus that is supported by AUTOSAR, including CAN, LIN, SPI and FlexRay. Depending on: (1) reliability and type of a used network, (2) size and criticality of the transmitted data, and (3) fault tolerance of application; the protocol needs to be appropriately configured. The configuration involves selection of used mechanisms and mechanism strength (e.g. CRC8 vs CRC16). This is left to the integrator to choose. Moreover, depending on: (1) Communication model (client-server vs. sender-receiver), (2) Communication multiplicity (1:n vs 1:1 vs n:1); some mechanisms are or aren't present (e.g. there is no destination address in 1:n sender-receiver communication). There are no dependencies to any other concepts. In particular, we do not depend on "Communication Stack" concept.
Rationale:	1/ To detect and tolerate faults in RTE, communication software and other BSWMs, as well as in communication hardware.
Use Case:	SW-Cs located on remote ECUs, exchanging safety-related data.
Dependencies:	
Conflicts:	
Supporting Material:	Concept AUTOSAR_CON_SWCEndToEndCommunicationProtection.doc

Argument of the feature coverage

The feature "SW-C end-to-end communication protection" is considered fulfilled if:

ID	**Description**
RS_BRF_00114_CC01	There is a Library with E2E protection mechanisms realized within AUTOSAR
RS_BRF_00114_CC02	The library can be invoked by SW-Cs

Coverage justification

These 2 bullets are covered as follows:

Coverage Criteria	**Coverage Justification**		
	BSW module	**Requirements**	**Justification**
RS_BRF_00114_CC01		[SRS_LIBS_08527]	The functions to protect data are realized in the BSW as a stateless library, and can be called (e.g. by a SW-C) to verify the integrity of exchanged safety-related data. The caller will get a notification about detected faults and is able to handle such faults at runtime.
RS_BRF_00114_CC02		[SRS_LIBS_08528] (CRC is provided by BSW library [SRS_LIBS_08518, SRS_LIBS_08526,	The Library contains different E2E profiles, each of them having an appropriate and consistent set of protection mechanisms to be used at data element level. The E2E profiles defined for R4.0

		SRS_LIBS_08536, SRS_LIBS_08533]) [E2E0089] [E2E0043, SWS_E2E_00070, E2E00117]	use mechanism like CRC, an alive/sequence counter and a data ID.

The E2E library detects the errors and reports them to SW-Cs callers [SWS_E2E_00012, SWS_E2E_00011, E2E0010].

4.6 Memory partitioning and user/supervisor-modes Related Features

4.6.1 Overview

The features described in this chapter are the extensions of the OS and the RTE functionality required to enable the groups of SW-Cs can run in separate memory partitions (e.g. using inter-OS-Application communication across boundaries of memory partitions) in order to provide freedom from interference between software components (e.g. memory-related faults in a SW-C does not propagate to other SW-C's and a SW-C executed in user-mode has restricted access to CPU instructions like e.g. reconfiguration).

With these extensions, it is possible to setup protection boundaries between SW-Cs.

Memory partitioning provides protection by means of restricting access to memory and memory-mapped hardware. Memory partitioning means that OS-Applications reside in different memory areas (partitions) that are protected from each other. In particular, code executing in one partition cannot modify memory of a different partition. Moreover, memory partitioning enables to protect read-only memory segments, as well as to protect memory-mapped hardware.

Supervisor/user-modes provide protection by means of restricting the access to CPU.

Note:
The mechanisms are currently applicable to the SW-Cs, and not to the BSW modules. These extensions may also be useful for debugging and testing of SW-Cs.

4.6.2 Related features

4.6.2.1 [RS_BRF_00115] SW-Cs grouped in separate user-mode memory partitions

Initiator:	AUTOSAR Safety Team
Date:	27.02.2006

Short Description:	SW-Cs grouped in separate user-mode memory partitions
Importance:	High
Description:	The feature defines the extensions of the OS and the RTE functionality that are necessary to support groups of SW-Cs running in separate user-mode memory partitions. The most important resulting AUTOSAR extension is the inter OS-Application communication (across boundaries of memory partitions). Further (smaller) extensions are in the configuration and error handling. Partitioning of BSW is not in the scope of the concept/feature – only SW-C is covered. With these extensions, it will be possible to setup protection boundaries prohibiting a propagation of some kinds of hardware and software faults. This is especially interesting when there are several SW-Cs on one ECU, and when SW-Cs have different ASIL or they come from different parties. This is also useful for debugging and testing of SW-Cs. Memory partitioning provides protection by means of restricting access to memory and memory-mapped hardware. Memory partitioning means that OS-Applications reside in different memory areas (partitions) that are protected from each other. In particular, code executing in one partition cannot modify memory of a different partition in an uncontrolled fashion, even by indirect means. Moreover, memory partitioning enables to protect read-only memory segments, as well as to protect memory-mapped hardware. Supervisor/user modes provide the protection by means of restricting the access to CPU. Currently, OS makes the notion of a partition being identified with the notion of the associated OS-Application. In other words, each OS-Application has its own memory partition, with separate stack, data and code. OS assumes (requires) an MPU for providing memory protection (by segmentation). Support for MMU (by paging) is not specified. However, there is no communication mechanism between OS-Applications offered. OS itself does not provide the communication between OS-Applications - instead, OS clearly delegates the communication between partitions (i.e. basic techniques for transferring data between protected memory regions) to RTE. RTE assumes its role, but does not provide these mechanisms yet. Therefore, inter OS-Application communication (i.e. communication between different OS-Applications within the same ECU) is the major missing functionality. Possible extensions of RTE communication modes (client-server, sender-receiver) will be sketched by this concept, so that they work not only intra-OS-Application, inter-ECU, but also inter OS-Application.
Rationale:	This prevents the following failure modes from propagating: 1. systematic software faults in SW-Cs (i.e. bugs in software, like buffer overflows, incorrect pointer arithmetic) 2. random hardware faults in SW-Cs (e.g. faults of address unit, faults in memory cells storing pointers)
Use Case:	The concept/feature enables the following combinations of SW-Cs on one ECU: SW-Cs of different ASIL SW-Cs from different vendors, SW-Cs under debugging/testing.
Dependencies:	There is a hardware dependency, which is already explicit in AUTOSAR OS. “SW-Cs grouped in separate user-mode memory partitions” is only possible on processors that provide hardware support for memory protection (MPU, MMU). Another feature ([RS_BRF_00275] Capability for Application Level SW-C Management (stop, start, restart)) is very useful for this feature, but not strictly required.
Conflicts:	--
Supporting Material:	--

Coverage Criteria of the feature

The feature is considered fulfilled if:

ID	Description
RS_BRF_00115_CC01	Autosar methodology supports the configuration of memory partitions. For each SW-C it is possible to define to which partition it belongs, and the mode of this partition.
RS_BRF_00115_CC02	OS is able to manage the OS-Applications
RS_BRF_00115_CC03	RTE provides communication between software modules belonging to different memory partitions, i.e. between SW-C and SW-C, and between SW-C and base software. RTE can use IOC, it can alternatively use OS trusted functions.
Within the scope of error handling concept	*OS is able to catch the hardware interrupts resulting from memory violations or mode violations (i.e. when an SW-C illegally accesses the memory or when SW-C calls a supervisor CPU instruction).*
Within the scope of error handling concept	*RTE and OS are able to do error handling on memory violation and mode violation, which is restarting of the SW-C partition or shutting it down.*

Coverage justification

Coverage Criteria	Coverage Justification		
	AUTOSAR specification	**Requirements**	**Justification**
RS_BRF_00115_CC01	ECU Configuration	[ECUC_EcuC_00005]	Within ECU configuration, OS-Applications belong 1-to-1 to Partitions
RS_BRF_00115_CC02	SWS OS	SWS_Os_00445 SWS_Os_00446	OS manages OS-Applications
RS_BRF_00115_CC03	SWS RTE	rte_sws_7606, rte_sws_7604, rte_sws_7610, rte_sws_5147, rte_sws_7330, rte_sws_7331, rte_sws_7334, rte_sws_7335, rte_sws_7620, rte_sws_7619, rte_sws_7617, rte_sws_7622, rte_sws_7645, rte_sws_7643, rte_sws_7644, rte_sws_7188, rte_sws_7336, rte_sws_7338, rte_sws_7339, rte_sws_7340, rte_sws_7341, rte_sws_7342, 4.3.4 Inter-Partition communication	RTE provides intra-partition communication, handles the state of partitions (e.g. restarting)

5 Requirements traceability

5.1 Referred documents

Names of the documents		
AUTOSAR_SWS_COM		
AUTOSAR_SRS_COM		
AUTOSAR_SWS_OS		
AUTOSAR_SRS_OS		
AUTOSAR_SWS_RTE		
AUTOSAR_SRS_RTE		
AUTOSAR_SRS_SynchronizedTimeBaseManager		
AUTOSAR_SWS_SynchronizedTimeBaseManager		
AUTOSAR_SWS_WatchdogManager		
AUTOSAR_SRS_ModeManagement		
AUTOSAR_TPS_TimingExtensions		
AUTOSAR_RS_TimingExtensions		
AUTOSAR_SRS_SPIHandlerDriver		
AUTOSAR_SWS_SPIHandlerDriver		
AUTOSAR_SRS_ADCDriver		
AUTOSAR_SWS_ADCDriver		
AUTOSAR_SRS_DIODriver		
AUTOSAR_SWS_DIODriver		
AUTOSAR_SRS_ICUDriver		
AUTOSAR_SWS_ICUDriver		
AUTOSAR_SRS_Libraries		
AUTOSAR_SWS_E2ELibrary		

5.2 Safety features to SRS safety related requirements

Safety feature	***Satisfied by***	***Related SRS***
RS_BRF_00131 *Logical Program Flow Monitoring*	SRS_ModeMgm_09106, SRS_ModeMgm_09143, SRS_ModeMgm_09159, SRS_ModeMgm_09162, SRS_ModeMgm_09163, SRS_ModeMgm_09169, SRS_ModeMgm_09220, SRS_ModeMgm_09221, SRS_ModeMgm_09222, SRS_ModeMgm_09223, SRS_ModeMgm_09225, SRS_ModeMgm_09226	AUTOSAR_SRS_ModeManagement
RS_BRF_00120 *Provision of a synchronized time-base within a cluster*	SRS_StbM_20002, SRS_StbM_20005, SRS_StbM_20006, SRS_StbM_20007 SRS_Os_11002	AUTOSAR_SRS_SynchronizedTimeBaseManager AUTOSAR_SRS_OS
RS_BRF_00121 *Runtime timing protection and monitoring*	SRS_Os_11008 SRS_Rte_00193, SRS_Rte_00160	AUTOSAR_SRS_OS AUTOSAR_SRS_RTE
RS_BRF_00122 *Support for timing constraints*	RSTM001, RSTM002, RSTM003, RSTM004, RSTM012 SRS_Os_00097, SRS_Os_00098 SRS_Rte_00046	AUTOSAR_RS_TimingExtensions AUTOSAR_SRS_OS AUTOSAR_SRS_RTE
RS_BRF_00123 *Responsiveness to external events*	(Same feature as RS_BRF_00031) SRS_Rte_00162, SRS_Rte_00163, SRS_Rte_00216	AUTOSAR_SRS_RTE
RS_BRF_00125 *Monitoring of local time*	According to the latest version of the time determinism concept document (MS2) the implementation of this feature is left to the application developer.	
RS_BRF_00126 *Services for synchronization of SW-Cs*	RSTM002	410 RS Timing Extensions

	SRS_Rte_00232 SRS_StbM_20002	AUTOSAR_SRS_RTE AUTOSAR_SRS_SynchronizedTimeBaseManager
RS_BRF_00127 *Services for accessing to synchronized time-bases*	SRS_StbM_20001, SRS_StbM_20002, SRS_StbM_20003, SRS_StbM_20008, SRS_StbM_20009, SRS_StbM_20010 SRS_Os_11002	AUTOSAR_SRS_SynchronizedTimeBaseManager AUTOSAR_SRS_OS
RS_BRF_00278 *Sync AUTOSAR OS with y Global Time from providing bus system in a well-defined way*	SRS_StbM_20002, SRS_StbM_20005, SRS_StbM_20006, SRS_StbM_20007 SRS_Os_11002	AUTOSAR_SRS_SynchronizedTimeBaseManager AUTOSAR_SRS_OS
RS_BRF_00111 *Data Sequence Control*	SRS_Com_02099, SRS_Com_02100, SRS_Com_02101, SRS_Com_02102	AUTOSAR_SRS_COM.doc
RS_BRF_00241 *Multiple Communication Links*	SRS_Com_02103, SRS_Com_02104, SRS_Com_02105, SRS_Com_02106	AUTOSAR_SRS_COM
RS_BRF_00115 *SW-Cs grouped in separate user-mode memory partitions*	SRS_Rte_00210 SRS_Os_11010	AUTOSAR_SRS_RTE AUTOSAR_SRS_OS
RS_BRF_00243 *Communication protections against corruption and loss of data*	SRS_LIBS_08527, SRS_LIBS_08536, SRS_LIBS_08535	AUTOSAR_SRS_Libraries
RS_BRF_00251 *Priority access to SPI bus*	SRS_Spi_12037	AUTOSAR_SRS_SPIHandlerDriver
RS_BRF_00248 *Testing and monitoring of I/O data and I/O HW*	No explicit requirement (see justification)	AUTOSAR_SRS_ADCDriver
RS_BRF_00301 *Ability to make an AUTOSAR application compatible to the e-Gas*	SRS_SPAL_12063, SRS_Dio_12352, SRS_Icu_12436, SRS_Icu_12369	AUTOSAR_SRS_ADCDriver AUTOSAR_SRS_DIODriver AUTOSAR_SRS_ICUDriver
RS_BRF_00114	SRS_LIBS_08527, SRS_LIBS_08528,	AUTOSAR_SRS_Libraries

	SRS_LIBS_08529, SRS_LIBS_08530, SRS_LIBS_08531, SRS_LIBS_08533, SRS_LIBS_08534, SRS_LIBS_08535, SRS_LIBS_08536, SRS_LIBS_08537	

5.3 SRS safety related requirements to SWS safety related requirements

5.3.1 SRS COM

Safety requirement	***Satisfied by***	***Related SWS***
SRS_Com_02099 *I-PDU Counter mechanism*	COM587, COM588, COM590, COM687, COM688, COM726, COM727	AUTOSAR_SWS_COM
SRS_Com_02100 *I-PDU Counter configuration*	ECUC_Com_00592, ECUC_Com_00593, ECUC_Com_00594, ECUC_Com_00595, ECUC_Com_00003	AUTOSAR_SWS_COM
SRS_Com_02101 *Transmission and reception using I-PDU Counter*	COM587, COM588, COM687, COM688	AUTOSAR_SWS_COM
SRS_Com_02102 *I-PDU Counter error handling*	COM590, COM726, COM727	AUTOSAR_SWS_COM
SRS_Com_02103 *I-PDU Replication mechanism*	COM596, COM597	AUTOSAR_SWS_COM
SRS_Com_02104 *I-PDU replication configuration*	ECUC_Com_00599, ECUC_Com_00600, ECUC_Com_00601	AUTOSAR_SWS_COM
SRS_Com_02105 *Transmission and reception using I-PDU Replication*	COM596, COM597	AUTOSAR_SWS_COM
SRS_Com_02106	COM596, COM597	AUTOSAR_SWS_COM

I-PDU Replication error handling		

5.3.2 SRS ModeManagement

Safety requirement	***Satisfied by***	***Related SWS***
SRS_ModeMgm_09220 *Configuration of all transition relations*	ECUC_WdgM_00343, ECUC_WdgM_00344, ECUC_WdgM_00345, ECUC_WdgM_00350, ECUC_WdgM_00351	AUTOSAR_SWS_WatchdogManager
SRS_ModeMgm_09221 *Logical program flow monitoring*	SWS_WdgM_00119, SWS_WdgM_00120, SWS_WdgM_00121, SWS_WdgM_00122, SWS_WdgM_00223, SWS_WdgM_00196, SWS_WdgM_00197, SWS_WdgM_00198, SWS_WdgM_00199, SWS_WdgM_00242, SWS_WdgM_00246, WDGM247, WDGM248, WDGM249, WDGM250, WDGM251, SWS_WdgM_00252, SWS_WdgM_00263, SWS_WdgM_00271, SWS_WdgM_00273, SWS_WdgM_00274, ECUC_WdgM_00319, ECUC_WdgM_00320, ECUC_WdgM_00321, ECUC_WdgM_00322, ECUC_WdgM_00323, ECUC_WdgM_00324, ECUC_WdgM_00343, ECUC_WdgM_00344, ECUC_WdgM_00345, ECUC_WdgM_00350, ECUC_WdgM_00351	AUTOSAR_SWS_WatchdogManager
SRS_ModeMgm_09222 *Update logical program flow monitoring*	SWS_WdgM_00263	AUTOSAR_SWS_WatchdogManager
SRS_ModeMgm_09223 *Post build time and mode dependent selectable configuration of transition relations*	ECUC_WdgM_00319, ECUC_WdgM_00320, ECUC_WdgM_00321, ECUC_WdgM_00322, ECUC_WdgM_00323, ECUC_WdgM_00324	AUTOSAR_SWS_WatchdogManager
SRS_ModeMgm_09225 *Indication of failed logical monitoring*	SWS_WdgM_00196, SWS_WdgM_00197, SWS_WdgM_00198, SWS_WdgM_00199	AUTOSAR_SWS_WatchdogManager
SRS_ModeMgm_09226	SWS_WdgM_00119, SWS_WdgM_00120,	AUTOSAR_SWS_WatchdogManager

Condition to reset the triggering condition in the Watchdog Driver in case of logical program flow failure	SWS_WdgM_00121, SWS_WdgM_00122, SWS_WdgM_00223	

5.3.3 SRS Synchronized Time-base Manager

Safety requirement	***Satisfied by***	***Related SWS***
SRS_StbM_20001 *Deal with different customer types*	SWS_StbM_00020, SWS_StbM_00025, SWS_StbM_00026, SWS_StbM_00028, SWS_StbM_00029, SWS_StbM_00037, SWS_StbM_00038, SWS_StbM_00082	
SRS_StbM_20002 *Synchronize triggered customer*	SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083	
SRS_StbM_20003 *Access to time-base value*	SWS_StbM_00082, SWS_StbM_00025, SWS_StbM_00026, SWS_StbM_00028, SWS_StbM_00029	
SRS_StbM_20005 *Perform access to time-base provider*	SWS_StbM_00050, SWS_StbM_00080, SWS_StbM_00081, SWS_StbM_00015	
SRS_StbM_20006 *Dependable provision of time*	SWS_StbM_00050	
SRS_StbM_20007 *Fault detection*	SWS_StbM_00030, SWS_StbM_00031, SWS_StbM_00032, SWS_StbM_00033, SWS_StbM_00034, SWS_StbM_00035, SWS_StbM_00036	
SRS_StbM_20008 *Notification mechanism*	SWS_StbM_00037, SWS_StbM_00038	
SRS_StbM_20009 Configuration of triggered customers	SWS_StbM_00084, SWS_StbM_00085	

SRS_StbM_20010 *System service interface*	Chapter 11 in	

5.3.4 SRS RTE

Safety requirement	***Satisfied by***	***Related SWS***
SRS_Rte_00232 [Missing Requ. on synchronization]	rte_sws_7804, rte_sws_7805	AUTOSAR_SWS_RTE
SRS_Rte_00162 *1:n External Trigger communication*	rte_sws_7229, rte_sws_7212, rte_sws_7213, rte_sws_7214, rte_sws_7543, rte_sws_7215, rte_sws_7216, rte_sws_7218, rte_sws_7200, rte_sws_7201, rte_sws_7207	AUTOSAR_SWS_RTE
SRS_Rte_00163 *Support for InterRunnableTriggering*	rte_sws_7229, rte_sws_7220, rte_sws_7555, rte_sws_7221, rte_sws_7224, rte_sws_7223, rte_sws_7203, rte_sws_7204, rte_sws_7226, rte_sws_7227, rte_sws_7228, rte_sws_7208	AUTOSAR_SWS_RTE
SRS_Rte_00216 *Triggering of BSW Schedulable Entities by occurrence of External Trigger*	rte_sws_7514, rte_sws_7542, rte_sws_7213, rte_sws_7214, rte_sws_7544, rte_sws_7545, rte_sws_7548, rte_sws_7546, rte_sws_7216, rte_sws_7218, rte_sws_7549, rte_sws_7282, rte_sws_7283	AUTOSAR_SWS_RTE
SRS_Rte_00046 *Support for 'Executable Entity runs inside' Exclusive Areas*	rte_sws_3500, rte_sws_3515, rte_sws_7522, rte_sws_7523, rte_sws_7524, rte_sws_2740, rte_sws_2741, rte_sws_2743, rte_sws_2744, rte_sws_2745, rte_sws_2746, rte_sws_1120, rte_sws_1122, rte_sws_1123, rte_sws_7250, rte_sws_7251, rte_sws_7252, rte_sws_7578, rte_sws_7579, rte_sws_7253, rte_sws_7254	AUTOSAR_SWS_RTE
SRS_Rte_00193 *Support for Runnable Entity execution chaining*	sws_rte_7800, sws_rte_7802	AUTOSAR_SWS_RTE

SRS_Rte_00160 *Debounced start of Runnable Entities*	rte_sws_2697	AUTOSAR_SWS_RTE
SRS_Rte_00210 *Support for inter OS application communication*	rte_sws_7606 rte_sws_2752 rte_sws_2753 rte_sws_2756 rte_sws_2754 rte_sws_2728 rte_sws_2755 rte_sws_2731 rte_sws_2732	AUTOSAR_SWS_RTE

5.3.5 SRS OS

Safety requirement	***Satisfied by***	***Related SWS***
SRS_Os_11002 *Synchronization with global time*	SWS_Os_00206, SWS_Os_00201, SWS_Os_00013, SWS_Os_00199, SWS_Os_00227, SWS_Os_00429, SWS_Os_00430, SWS_Os_00431, SWS_Os_00462, SWS_Os_00463, SWS_Os_00435, SWS_Os_00415, SWS_Os_00416, SWS_Os_00436, SWS_Os_00437, SWS_Os_00438, SWS_Os_00417, SWS_Os_00418, SWS_Os_00419, SWS_Os_00420, SWS_Os_00421, SWS_Os_00422	AUTOSAR_SWS_OS
SRS_Os_00097 *Existing OSEK OS*	SWS_Os_00001	AUTOSAR_SWS_OS
SRS_Os_00098 *Table based schedules*	SWS_Os_00002, SWS_Os_00007	AUTOSAR_SWS_OS
SRS_Os_11008 *Timing Protection*	SWS_Os_00028, SWS_Os_00089, SWS_Os_00033, SWS_Os_00037, SWS_Os_00048, SWS_Os_00064, SWS_Os_00465, SWS_Os_00469, SWS_Os_00470, SWS_Os_00471, SWS_Os_00472, SWS_Os_00473,	AUTOSAR_SWS_OS

	SWS_Os_00474	
SRS_Os_11010 *Protection of OS-Applications*	SWS_Os_00056	AUTOSAR_SWS_OS

5.3.6 RS Timing Extensions

Safety requirement	***Satisfied by***	***Related SWS***
RSTM001	timing events (section 3.2), timing event chains (section 3.3), event triggering constraint (section 3.5), latency constraint (section 3.6), synchronization constraint (section 3.7), execution order constraint (section 3.8)	AUTOSAR_TPS_TimingExtensions
RSTM002	event triggering constraint (section 3.5), latency constraint (section 3.6), synchronization constraint (section 3.7), execution order constraint (section 3.8)	AUTOSAR_TPS_TimingExtensions
RSTM004	timing event chains (section 3.3)	AUTOSAR_TPS_TimingExtensions
RSTM012	latency constraint (section 3.6)	AUTOSAR_TPS_TimingExtensions

5.3.7 AUTOSAR_SRS_SPIHandlerDriver

Safety requirement	***Satisfied by***	***Related SWS***
SRS_Spi_12037	SWS_Spi_00002 ,SWS_Spi_00014, SWS_Spi_00093, SWS_Spi_00059	AUTOSAR_SWS_SPIHandlerDriver

5.3.8 AUTOSAR_SRS_ADCDriver

Safety requirement	***Satisfied by***	***Related SWS***
SRS_SPAL_12063	SWS_Adc_00113	AUTOSAR_SWS_ADCDriver

5.3.9 AUTOSAR_SRS_DIODriver

Safety requirement	***Satisfied by***	***Related SWS***
SRS_Dio_12352	SWS_Dio_00083	AUTOSAR_SWS_DIODriver

5.3.10 AUTOSAR_SRS_ICUDriver

Safety requirement	***Satisfied by***	***Related SWS***
SRS_Icu_12436	SWS_Ico_00211, SWS_Ico_00342, SWS_Ico_00084, SWS_Ico_00344, SWS_Ico_00106, SWS_Ico_00345, SWS_Ico_00180, SWS_Ico_00181, SWS_Ico_00022, SWS_Ico_00048, ICU272, ICU265	AUTOSAR_SWS_ICUDriver
SRS_Icu_12369	SWS_Ico_00021	AUTOSAR_SWS_ICUDriver

5.3.11 AUTOSAR_SRS_Libraries

Safety requirement	***Satisfied by***	***Related SWS***
SRS_LIBS_08527, SRS_LIBS_08536	SWS_E2E_00020, E2E0023, E2E0026, E2E0030, E2E0043	AUTOSAR_SWS_E2ELibrary
SRS_LIBS_08535	E2E0026, E2E0030	AUTOSAR_SWS_E2ELibrary

5.4 Backward traceability

5.4.1 SWS requirements related to only one Safety Feature (BRF)

SWS requirement	*Covers the BRF*	*Related SWS requirement*
COM587, COM588, COM590, COM687, COM688, COM726, COM727, ECUC_Com_00592, ECUC_Com_00593, ECUC_Com_00594, ECUC_Com_00595, ECUC_Com_00003	**RS_BRF_00111**	To themselves
SWS_Os_00056, rte_sws_7606, rte_sws_2728, rte_sws_2753, rte_sws_2731, rte_sws_2754, rte_sws_2732, rte_sws_2752, rte_sws_2756, rte_sws_2755	**RS_BRF_00115**	To themselves
SWS_WdgM_00119, SWS_WdgM_00120, SWS_WdgM_00121, SWS_WdgM_00122, SWS_WdgM_00223, SWS_WdgM_00196, SWS_WdgM_00197, SWS_WdgM_00198, SWS_WdgM_00199, SWS_WdgM_00242, SWS_WdgM_00246, WDGM247, WDGM248, WDGM249, WDGM250, WDGM251, SWS_WdgM_00252, SWS_WdgM_00263, SWS_WdgM_00271, SWS_WdgM_00273, SWS_WdgM_00274, ECUC_WdgM_00319, ECUC_WdgM_00320, ECUC_WdgM_00321, ECUC_WdgM_00322, ECUC_WdgM_00323, ECUC_WdgM_00324, ECUC_WdgM_00343, ECUC_WdgM_00344, ECUC_WdgM_00345, ECUC_WdgM_00350, ECUC_WdgM_00351	**RS_BRF_00131**	To themselves
COM596, COM597, ECUC_Com_00599, ECUC_Com_00600, ECUC_Com_00601	**RS_BRF_00241**	To themselves
SWS_Spi_00002 ,SWS_Spi_00014, SWS_Spi_00093, SWS_Spi_00059	**RS_BRF_00251**	To themselves
SWS_Adc_00113	**RS_BRF_00301**	To themselves
SWS_Dio_00083	**RS_BRF_00301**	To themselves
SWS_Ico_00211, SWS_Ico_00342, SWS_Ico_00084, SWS_Ico_00344, SWS_Ico_00106, SWS_Ico_00345, SWS_Ico_00180, SWS_Ico_00181, SWS_Ico_00022, SWS_Ico_00048, ICU272, ICU265, SWS_Ico_00021	**RS_BRF_00301**	To themselves
SWS_E2E_00020, E2E0023, E2E0026, E2E0030, E2E0043, E2E0026, E2E0030	**RS_BRF_00243**	To themselves

5.4.2 SWS requirements related to multiple Safety Features (BRF)

- **RS_BRF_00120**

SWS requirement	*Covers the BRF*	*Related SWS requirement*
SWS_Os_00013, SWS_Os_00199, SWS_Os_00201, SWS_Os_00206, SWS_Os_00227, SWS_Os_00415, SWS_Os_00416, SWS_Os_00417,	**RS_BRF_00120**	To themselves

SWS_Os_00418, SWS_Os_00419, SWS_Os_00420, SWS_Os_00421, SWS_Os_00422, SWS_Os_00429, SWS_Os_00430, SWS_Os_00431, SWS_Os_00435, SWS_Os_00436, SWS_Os_00437, SWS_Os_00438, SWS_Os_00462, SWS_Os_00463, SWS_StbM_00015, SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00030, SWS_StbM_00031, SWS_StbM_00032, SWS_StbM_00033, SWS_StbM_00034, SWS_StbM_00035, SWS_StbM_00036, SWS_StbM_00050, SWS_StbM_00077, SWS_StbM_00080, SWS_StbM_00081, SWS_StbM_00083		
SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083	**RS_BRF_00126**	SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083, EventTriggeringConstraint, LatencyTimingConstraint, SynchronizationTimingConstraint, ExecutionOrderConstraint
SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083	**RS_BRF_00127**	SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00025, SWS_StbM_00026, SWS_StbM_00028, SWS_StbM_00029, SWS_StbM_00037, SWS_StbM_00038, SWS_StbM_00077, SWS_StbM_00082, SWS_StbM_00083, SWS_StbM_00084, SWS_StbM_00085
SWS_Os_00013, SWS_Os_00199, SWS_Os_00201, SWS_Os_00206, SWS_Os_00227, SWS_Os_00415, SWS_Os_00416, SWS_Os_00417, SWS_Os_00418, SWS_Os_00419, SWS_Os_00420, SWS_Os_00421, SWS_Os_00422, SWS_Os_00429, SWS_Os_00430, SWS_Os_00431, SWS_Os_00435, SWS_Os_00436, SWS_Os_00437, SWS_Os_00438, SWS_Os_00462, SWS_Os_00463, SWS_StbM_00015, SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00030, SWS_StbM_00031, SWS_StbM_00032, SWS_StbM_00033, SWS_StbM_00034, SWS_StbM_00035, SWS_StbM_00036, SWS_StbM_00050, SWS_StbM_00077, SWS_StbM_00080, SWS_StbM_00081, SWS_StbM_00083	**RS_BRF_00278** *This two BRF are currently covered by the same requirements*	SWS_Os_00013, SWS_Os_00199, SWS_Os_00201, SWS_Os_00206, SWS_Os_00227, SWS_Os_00415, SWS_Os_00416, SWS_Os_00417, SWS_Os_00418, SWS_Os_00419, SWS_Os_00420, SWS_Os_00421, SWS_Os_00422, SWS_Os_00429, SWS_Os_00430, SWS_Os_00431, SWS_Os_00435, SWS_Os_00436, SWS_Os_00437, SWS_Os_00438, SWS_Os_00462, SWS_Os_00463, SWS_StbM_00015, SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00030, SWS_StbM_00031, SWS_StbM_00032, SWS_StbM_00033, SWS_StbM_00034, SWS_StbM_00035, SWS_StbM_00036, SWS_StbM_00050, SWS_StbM_00077, SWS_StbM_00080, SWS_StbM_00081, SWS_StbM_00083

- **RS_BRF_00121**

SWS requirement	*Covers the BRF*	*Related SWS requirement*
SWS_Os_00028, SWS_Os_00033, SWS_Os_00037, SWS_Os_00048, SWS_Os_00064, SWS_Os_00089, SWS_Os_00465, SWS_Os_00469, SWS_Os_00470, SWS_Os_00471, SWS_Os_00472, SWS_Os_00473, SWS_Os_00474, rte_sws_2697, sws_rte_7800, sws_rte_7802	**RS_BRF_00121**	To themselves
rte_sws_2697	**RS_BRF_00122**	SWS_Os_00001, SWS_Os_00002, SWS_Os_00007, rte_sws_1120, rte_sws_1122, rte_sws_1123, rte_sws_1131, rte_sws_1133, rte_sws_1135, rte_sws_1137, rte_sws_1166, rte_sws_1359, rte_sws_2203, rte_sws_2512, rte_sws_2697, rte_sws_2740, rte_sws_2741, rte_sws_2743, rte_sws_2744, rte_sws_2745, rte_sws_2746, rte_sws_3500, rte_sws_3515, rte_sws_3520, rte_sws_3523, rte_sws_3524, rte_sws_3526, rte_sws_3527, rte_sws_3530, rte_sws_3531, rte_sws_3532, rte_sws_7023, rte_sws_7024, rte_sws_7025, rte_sws_7026, rte_sws_7027, rte_sws_7177, rte_sws_7178, rte_sws_7207, rte_sws_7208, rte_sws_7250, rte_sws_7251, rte_sws_7252, rte_sws_7253, rte_sws_7254, rte_sws_7379, rte_sws_7403, rte_sws_7515, rte_sws_7522, rte_sws_7523, rte_sws_7524, rte_sws_7575, rte_sws_7578, rte_sws_7579, TimingDescriptionEvent, TimingDescriptionEventChain, EventTriggeringConstraint, LatencyTimingConstraint, SynchronizationTimingConstraint, ExecutionOrderConstraint

- **RS_BRF_00122**

SWS requirement	*Covers the BRF*	*Related SWS requirement*
SWS_Os_00001, SWS_Os_00002, SWS_Os_00007, rte_sws_1120, rte_sws_1122, rte_sws_1123, rte_sws_1131, rte_sws_1133, rte_sws_1135, rte_sws_1137, rte_sws_1166, rte_sws_1359, rte_sws_2203, rte_sws_2512, rte_sws_2697, rte_sws_2740, rte_sws_2741, rte_sws_2743, rte_sws_2744, rte_sws_2745, rte_sws_2746, rte_sws_3500, rte_sws_3515, rte_sws_3520, rte_sws_3523, rte_sws_3524, rte_sws_3526, rte_sws_3527, rte_sws_3530, rte_sws_3531, rte_sws_3532, rte_sws_7023,	**RS_BRF_00122**	To themselves

rte_sws_7024, rte_sws_7025, rte_sws_7026, rte_sws_7027, rte_sws_7177, rte_sws_7178, rte_sws_7207, rte_sws_7208, rte_sws_7250, rte_sws_7251, rte_sws_7252, rte_sws_7253, rte_sws_7254, rte_sws_7379, rte_sws_7403, rte_sws_7515, rte_sws_7522, rte_sws_7523, rte_sws_7524, rte_sws_7575, rte_sws_7578, rte_sws_7579, TimingDescriptionEvent, TimingDescriptionEventChain, EventTriggeringConstraint, LatencyTimingConstraint, SynchronizationTimingConstraint, ExecutionOrderConstraint		
rte_sws_2697	**RS_BRF_00121**	SWS_Os_00028, SWS_Os_00033, SWS_Os_00037, SWS_Os_00048, SWS_Os_00064, SWS_Os_00089, SWS_Os_00465, SWS_Os_00469, SWS_Os_00470, SWS_Os_00471, SWS_Os_00472, SWS_Os_00473, SWS_Os_00474, rte_sws_2697, sws_rte_7800, sws_rte_7802
rte_sws_7207, rte_sws_7208	**RS_BRF_00123**	rte_sws_7200, rte_sws_7201, rte_sws_7203, rte_sws_7204, rte_sws_7207, rte_sws_7208, rte_sws_7212, rte_sws_7213, rte_sws_7214, rte_sws_7215, rte_sws_7216, rte_sws_7218, rte_sws_7220, rte_sws_7221, rte_sws_7223, rte_sws_7224, rte_sws_7226, rte_sws_7227, rte_sws_7228, rte_sws_7229, rte_sws_7282, rte_sws_7283, rte_sws_7514, rte_sws_7542, rte_sws_7543, rte_sws_7544, rte_sws_7545, rte_sws_7546, rte_sws_7548, rte_sws_7549, rte_sws_7555
EventTriggeringConstraint, LatencyTimingConstraint, SynchronizationTimingConstraint, ExecutionOrderConstraint	**RS_BRF_00126**	SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083, EventTriggeringConstraint, LatencyTimingConstraint, SynchronizationTimingConstraint, ExecutionOrderConstraint

- **RS_BRF_00123**

SWS requirement	***Covers the BRF***	***Related SWS requirement***
rte_sws_7200, rte_sws_7201, rte_sws_7203, rte_sws_7204, rte_sws_7207, rte_sws_7208, rte_sws_7212, rte_sws_7213, rte_sws_7214, rte_sws_7215, rte_sws_7216, rte_sws_7218,	**RS_BRF_00123**	To themselves

rte_sws_7220, rte_sws_7221, rte_sws_7223, rte_sws_7224, rte_sws_7226, rte_sws_7227, rte_sws_7228, rte_sws_7229, rte_sws_7282, rte_sws_7283, rte_sws_7514, rte_sws_7542, rte_sws_7543, rte_sws_7544, rte_sws_7545, rte_sws_7546, rte_sws_7548, rte_sws_7549, rte_sws_7555		
rte_sws_7207, rte_sws_7208	**RS_BRF_00122**	SWS_Os_00001, SWS_Os_00002, SWS_Os_00007, rte_sws_1120, rte_sws_1122, rte_sws_1123, rte_sws_1131, rte_sws_1133, rte_sws_1135, rte_sws_1137, rte_sws_1166, rte_sws_1359, rte_sws_2203, rte_sws_2512, rte_sws_2697, rte_sws_2740, rte_sws_2741, rte_sws_2743, rte_sws_2744, rte_sws_2745, rte_sws_2746, rte_sws_3500, rte_sws_3515, rte_sws_3520, rte_sws_3523, rte_sws_3524, rte_sws_3526, rte_sws_3527, rte_sws_3530, rte_sws_3531, rte_sws_3532, rte_sws_7023, rte_sws_7024, rte_sws_7025, rte_sws_7026, rte_sws_7027, rte_sws_7177, rte_sws_7178, rte_sws_7207, rte_sws_7208, rte_sws_7250, rte_sws_7251, rte_sws_7252, rte_sws_7253, rte_sws_7254, rte_sws_7379, rte_sws_7403, rte_sws_7515, rte_sws_7522, rte_sws_7523, rte_sws_7524, rte_sws_7575, rte_sws_7578, rte_sws_7579, TimingDescriptionEvent, TimingDescriptionEventChain, EventTriggeringConstraint, LatencyTimingConstraint, SynchronizationTimingConstraint, ExecutionOrderConstraint

- **RS_BRF_00126**

SWS requirement	***Covers the BRF***	***Related SWS requirement***
SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083, EventTriggeringConstraint, LatencyTimingConstraint, SynchronizationTimingConstraint, ExecutionOrderConstraint	**RS_BRF_00126**	To themselves
SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083	**RS_BRF_00120**	SWS_Os_00013, SWS_Os_00199, SWS_Os_00201, SWS_Os_00206, SWS_Os_00227, SWS_Os_00415, SWS_Os_00416, SWS_Os_00417, SWS_Os_00418, SWS_Os_00419, SWS_Os_00420, SWS_Os_00421, SWS_Os_00422, SWS_Os_00429, SWS_Os_00430,

		SWS_Os_00431, SWS_Os_00435, SWS_Os_00436, SWS_Os_00437, SWS_Os_00438, SWS_Os_00462, SWS_Os_00463, SWS_StbM_00015, SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00030, SWS_StbM_00031, SWS_StbM_00032, SWS_StbM_00033, SWS_StbM_00034, SWS_StbM_00035, SWS_StbM_00036, SWS_StbM_00050, SWS_StbM_00077, SWS_StbM_00080, SWS_StbM_00081, SWS_StbM_00083
EventTriggeringConstraint, LatencyTimingConstraint, SynchronizationTimingConstraint, ExecutionOrderConstraint	**RS_BRF_00122**	SWS_Os_00001, SWS_Os_00002, SWS_Os_00007, rte_sws_1120, rte_sws_1122, rte_sws_1123, rte_sws_1131, rte_sws_1133, rte_sws_1135, rte_sws_1137, rte_sws_1166, rte_sws_1359, rte_sws_2203, rte_sws_2512, rte_sws_2697, rte_sws_2740, rte_sws_2741, rte_sws_2743, rte_sws_2744, rte_sws_2745, rte_sws_2746, rte_sws_3500, rte_sws_3515, rte_sws_3520, rte_sws_3523, rte_sws_3524, rte_sws_3526, rte_sws_3527, rte_sws_3530, rte_sws_3531, rte_sws_3532, rte_sws_7023, rte_sws_7024, rte_sws_7025, rte_sws_7026, rte_sws_7027, rte_sws_7177, rte_sws_7178, rte_sws_7207, rte_sws_7208, rte_sws_7250, rte_sws_7251, rte_sws_7252, rte_sws_7253, rte_sws_7254, rte_sws_7379, rte_sws_7403, rte_sws_7515, rte_sws_7522, rte_sws_7523, rte_sws_7524, rte_sws_7575, rte_sws_7578, rte_sws_7579, TimingDescriptionEvent, TimingDescriptionEventChain, EventTriggeringConstraint, LatencyTimingConstraint, SynchronizationTimingConstraint, ExecutionOrderConstraint
SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083	**RS_BRF_00127**	SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00025, SWS_StbM_00026, SWS_StbM_00028, SWS_StbM_00029, SWS_StbM_00037, SWS_StbM_00038, SWS_StbM_00077, SWS_StbM_00082, SWS_StbM_00083, SWS_StbM_00084,

		SWS_StbM_00085
SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083	**RS_BRF_00278**	SWS_Os_00013, SWS_Os_00199, SWS_Os_00201, SWS_Os_00206, SWS_Os_00227, SWS_Os_00415, SWS_Os_00416, SWS_Os_00417, SWS_Os_00418, SWS_Os_00419, SWS_Os_00420, SWS_Os_00421, SWS_Os_00422, SWS_Os_00429, SWS_Os_00430, SWS_Os_00431, SWS_Os_00435, SWS_Os_00436, SWS_Os_00437, SWS_Os_00438, SWS_Os_00462, SWS_Os_00463, SWS_StbM_00015, SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00030, SWS_StbM_00031, SWS_StbM_00032, SWS_StbM_00033, SWS_StbM_00034, SWS_StbM_00035, SWS_StbM_00036, SWS_StbM_00050, SWS_StbM_00077, SWS_StbM_00080, SWS_StbM_00081, SWS_StbM_00083

- **RS_BRF_00127**

SWS requirement	***Covers the BRF***	***Related SWS requirement***
SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00025, SWS_StbM_00026, SWS_StbM_00028, SWS_StbM_00029, SWS_StbM_00037, SWS_StbM_00038, SWS_StbM_00077, SWS_StbM_00082, SWS_StbM_00083, SWS_StbM_00084, SWS_StbM_00085	**RS_BRF_00127**	To themselves
SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083	**RS_BRF_00120**	SWS_Os_00013, SWS_Os_00199, SWS_Os_00201, SWS_Os_00206, SWS_Os_00227, SWS_Os_00415, SWS_Os_00416, SWS_Os_00417, SWS_Os_00418, SWS_Os_00419, SWS_Os_00420, SWS_Os_00421, SWS_Os_00422, SWS_Os_00429, SWS_Os_00430, SWS_Os_00431, SWS_Os_00435, SWS_Os_00436, SWS_Os_00437, SWS_Os_00438, SWS_Os_00462, SWS_Os_00463, SWS_StbM_00015, SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00030, SWS_StbM_00031, SWS_StbM_00032, SWS_StbM_00033, SWS_StbM_00034, SWS_StbM_00035, SWS_StbM_00036, SWS_StbM_00050, SWS_StbM_00077, SWS_StbM_00080,

		SWS_StbM_00081, SWS_StbM_00083
SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083	**RS_BRF_00126**	SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083, EventTriggeringConstraint, LatencyTimingConstraint, SynchronizationTimingConstraint, ExecutionOrderConstraint
SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083	**RS_BRF_00278**	SWS_Os_00013, SWS_Os_00199, SWS_Os_00201, SWS_Os_00206, SWS_Os_00227, SWS_Os_00415, SWS_Os_00416, SWS_Os_00417, SWS_Os_00418, SWS_Os_00419, SWS_Os_00420, SWS_Os_00421, SWS_Os_00422, SWS_Os_00429, SWS_Os_00430, SWS_Os_00431, SWS_Os_00435, SWS_Os_00436, SWS_Os_00437, SWS_Os_00438, SWS_Os_00462, SWS_Os_00463, SWS_StbM_00015, SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00030, SWS_StbM_00031, SWS_StbM_00032, SWS_StbM_00033, SWS_StbM_00034, SWS_StbM_00035, SWS_StbM_00036, SWS_StbM_00050, SWS_StbM_00077, SWS_StbM_00080, SWS_StbM_00081, SWS_StbM_00083

- **RS_BRF_00278**

SWS requirement	***Covers the BRF***	***Related SWS requirement***
SWS_Os_00013, SWS_Os_00199, SWS_Os_00201, SWS_Os_00206, SWS_Os_00227, SWS_Os_00415, SWS_Os_00416, SWS_Os_00417, SWS_Os_00418, SWS_Os_00419, SWS_Os_00420, SWS_Os_00421, SWS_Os_00422, SWS_Os_00429, SWS_Os_00430, SWS_Os_00431, SWS_Os_00435, SWS_Os_00436, SWS_Os_00437, SWS_Os_00438, SWS_Os_00462, SWS_Os_00463, SWS_StbM_00015, SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00030, SWS_StbM_00031, SWS_StbM_00032, SWS_StbM_00033, SWS_StbM_00034, SWS_StbM_00035, SWS_StbM_00036, SWS_StbM_00050, SWS_StbM_00077, SWS_StbM_00080, SWS_StbM_00081, SWS_StbM_00083	**RS_BRF_00278**	To themselves
SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083	**RS_BRF_00120**	SWS_Os_00013, SWS_Os_00199, SWS_Os_00201, SWS_Os_00206, SWS_Os_00227, SWS_Os_00415, SWS_Os_00416, SWS_Os_00417, SWS_Os_00418,

		SWS_Os_00419, SWS_Os_00420, SWS_Os_00421, SWS_Os_00422, SWS_Os_00429, SWS_Os_00430, SWS_Os_00431, SWS_Os_00435, SWS_Os_00436, SWS_Os_00437, SWS_Os_00438, SWS_Os_00462, SWS_Os_00463, SWS_StbM_00015, SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00030, SWS_StbM_00031, SWS_StbM_00032, SWS_StbM_00033, SWS_StbM_00034, SWS_StbM_00035, SWS_StbM_00036, SWS_StbM_00050, SWS_StbM_00077, SWS_StbM_00080, SWS_StbM_00081, SWS_StbM_00083
SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083	**RS_BRF_00126**	SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083, EventTriggeringConstraint, LatencyTimingConstraint, SynchronizationTimingConstraint, ExecutionOrderConstraint
SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00077, SWS_StbM_00083	**RS_BRF_00127**	SWS_StbM_00020, SWS_StbM_00022, SWS_StbM_00025, SWS_StbM_00026, SWS_StbM_00028, SWS_StbM_00029, SWS_StbM_00037, SWS_StbM_00038, SWS_StbM_00077, SWS_StbM_00082, SWS_StbM_00083, SWS_StbM_00084, SWS_StbM_00085