MICROSAR Safe

Product Information

Version 1.12.2

Authors	Jonas Wolf
Status	Released

Document Information

History

Author	Date	Version	Remarks
Jonas Wolf	2015-10-31	1.0.0	Initial creation.
Jonas Wolf	2015-11-13	1.0.1	Remove GPT safety feature, add COM constraints
Jonas Wolf	2015-11-26	1.0.2	Adapt release types
Jonas Wolf	2015-12-01	1.0.3	Review of TSRs finished
Jonas Wolf	2015-12-10	1.0.4	Review comments by visrn
Jonas Wolf	2015-12-17	1.0.5	Safety manager mail contact details changed
Jonas Wolf	2016-01-19	1.1.0	Review by vismoe; Partitioning details added.
Jonas Wolf	2016-01-19	1.1.1	Fix term for SafeWDG
Jonas Wolf	2016-01-26	1.1.2	Fix typos
Jonas Wolf	2016-01-29	1.1.3	Update information from safety manual
Jonas Wolf	2016-03-08	1.2.0	Update information from safety manual Update roadmap Update Safety Case delivery Update of format Update delivery process
Jonas Wolf	2016-06-28	1.2.1	Update of lead time for production delivery
Jonas Wolf	2016-08-08	1.3.0	Update of roadmap
Jonas Wolf	2016-08-08	1.3.1	Remove constraint of PDUR
Jonas Wolf	2016-09-05	1.3.2	Update of roadmap
Jonas Wolf	2016-10-26	1.4.0	Update on XCP and AMD cluster Update on Ethernet Update on lead time for Safety Case for ASIL A/B Update of document structure
Jonas Wolf	2016-12-12	1.4.1	Update on Ethernet availability
Jonas Wolf	2017-01-10	1.5.0	Availability information revised Moved TLS to Ethernet cluster Removed constraint on COM
Jonas Wolf	2017-03-16	1.6.0	Added information that post build is not recommended Detail information on delivery dates Update to AUTOSAR 4.3 Update of availability
Rene Isau	2017-03-23	1.6.1	Corrected minor typos
Rene Isau	2017-03-27	1.6.2	Updated delivery time for Safety Case
Jonas Wolf	2017-06-20	1.7.0	Introduced new delivery type

Author	Date	Version	Remarks
			Production (Safety-ready) Update of Technical Safety Requirements
Jonas Wolf	2017-07-14	1.7.1	Added information on SOMEIPTP
Jonas Wolf	2017-08-15	1.8.0	Added information on vSENT, SafeETH, XFs, OEM SWCs Update of illustrations to fix typos
Jonas Wolf	2017-08-17	1.8.1	Added information on Safety Case
Jonas Wolf	2017-08-18	1.9.0	Simplification of delivery process and more information on issue reporting Incorporate safety manual updates
Jonas Wolf	2017-12-12	1.9.1	Make information about delivery process more precise
Jonas Wolf	2018-04-18	1.10.0	Update on availability of Crypto Services Update on availability of Transformers Update on availability of DCM Service Processors Added details on DEM More information on Safety Case Align spelling to new Vector-style
Monika Sturm	2018-05-03	1.10.1	List of Tables updated File renamed
Jonas Wolf	2018-10-01	1.10.2	Update of constraints for diagnostic components
Jonas Wolf	2018-11-16	1.11.0	Added Direkt OSEK NM Update of Technical Safety Concept
Jonas Wolf	2018-12-11	1.12.0	Added EcuM to Partitioning Solution
Jonas Wolf	2019-01-31	1.12.1	Update on availability of Crypto(vHSM)
Jonas Wolf	2019-02-08	1.12.2	Update on Crypto(Hw)

Reference Documents

No.	Source	Title	Version
[1]	ISO	ISO 26262 Road vehicles — Functional safety	2011/2012

Contents

Illustrations

Tables

1 Introduction

1.1 Purpose

This document provides product information on MICROSAR Safe. MICROSAR Safe are the ISO 26262-compliant parts of MICROSAR developed as Software Safety Element out of Context (SEooC).

This document should provide the reader with sufficient information to decide if MICROSAR Safe is an option for a specific project. This document should also give the reader a good impression of what to expect from MICROSAR Safe.

1.2 Scope

MICROSAR Safe comprises SafeBSW and SafeRTE. SafeBSW comprises further components, e.g. MICROSAR SafeOS and MICROSAR SafeWDG.

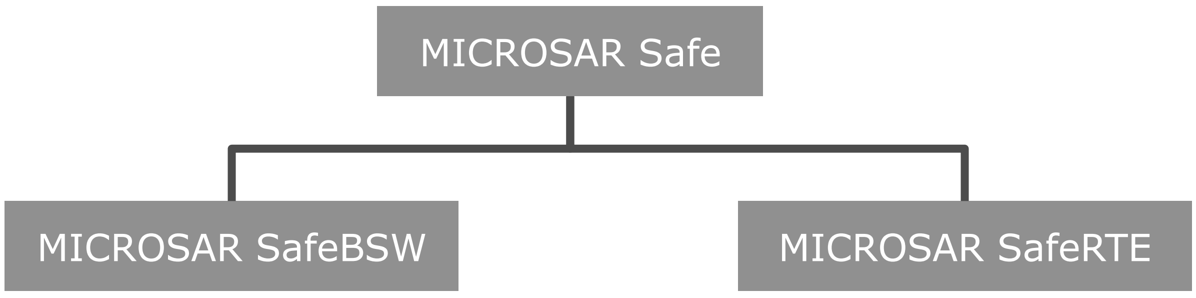

Figure 1-1 Structure of MICROSAR Safe

This document only provides information for Vector's MICROSAR product MICROSAR 4.

The information in this document is only applicable to MICROSAR Safe offers quoted by 31.10.2015 and later.

1.3 Overview

In Section 2 the safety concept is summarized. The lists of assumed requirements, components and constraints are detailed.

In Section 3 the delivery process is described.

In Section 4 the components of MICROSAR Safe and their availability is described.

In Section 5 information on the safety management is provided.

In Section 6 requirements for issue management are defined.

2 Safety Concept

2.1 Overview

The user of MICROSAR Safe is responsible for the overall safety concept.

The user of MICROSAR Safe can allocate safety requirements up to ASIL D to some MICROSAR Safe components. The assumed safety requirements and the components they affect are listed in the next section.

The user of MICROSAR Safe can use all components of MICROSAR Safe within the same partition, since they are developed according to the same ISO 26262-compliant process.

MICROSAR Safe is configurable software to serve as many use-cases as possible. The user of MICROSAR Safe is responsible for the adequate configuration of the components of MICROSAR Safe in the context of an item development.

A common safety concept involves the following set of components of MICROSAR Safe:

- MICROSAR SafeOS can be used to implement memory partitioning and achieve freedom from interference with respect to memory,
- MICROSAR SafeWDG can be used to implement deadline, alive and logic monitoring to achieve freedom from interference with respect to timing and execution,
- MICROSAR SafeE2E can be used to achieve freedom from interference with respect to exchange of information.
- MICROSAR SafeRTE can be used to communicate between application software components within in the ECU.
- MICROSAR SafeECUM can be used to initialize the ECU to a defined state.

The use of these components allows to effectively separate software components of different quality levels (QM, ASIL A-D).

Additional components from MICROSAR Safe can be used to optimize the number of partitions and thus minimize the number of partition switches or to allocate additional safety requirements to the BSW.

Vector is open to discuss your safety concept to select the components of MICROSAR Safe that best fit your needs.

2.2 Partitioning Options

There are two memory partitioning options available with MICROSAR Safe.

The first option is to run the BSW in a “non-trusted” or ”QM”-partition. This requires a MICROSAR SafeOS with Scalability Class 3 or 4 (SC3 or SC4) and a MICROSAR SafeWDG for timing supervision. This option is suitable if only a very small part of the ECU software is safety-related. The MICROSAR SafeWDG is then put in a separate “ASIL”-partition together with the safety-related application software. MICROSAR SafeRTE can be used to communicate between safety-related Software

Components (SWCs). End-to-end protection is used to ensure integrity of messages communicated between different ECUs.

MICROSAR SafeECUM provides the EcuM_Init function as ASIL software to allow initialization of the BSW to a defined state.

The second option is to run the BSW in the same partition as the rest of the safety-related application software. This option requires that all BSW components are used as MICROSAR Safe components. This option is suitable if a huge part of the ECU software is safety-related and frequently interacts with the BSW. This option still allows application software with QM level or lower ASIL. The parallel use of lower quality application software requires a MICROSAR SafeOS with Scalability Class 3 or 4 (SC3 or SC4) then.

For the scope of this document BSW also includes MCAL and EXT components.

MCAL and EXT components interfacing with the BSW (e.g. DioDrv, CanDrv or CanTrcv) need to be assigned in the same partition as the interfacing BSW. Other MCAL and EXT components not interfacing with the BSW (e.g. AdcDrv or PwmDrv) can be assigned to a different partition.

Note
The user of MICROSAR Safe should check with the MCAL provider if the MCAL is available with the required ASIL as soon as possible.

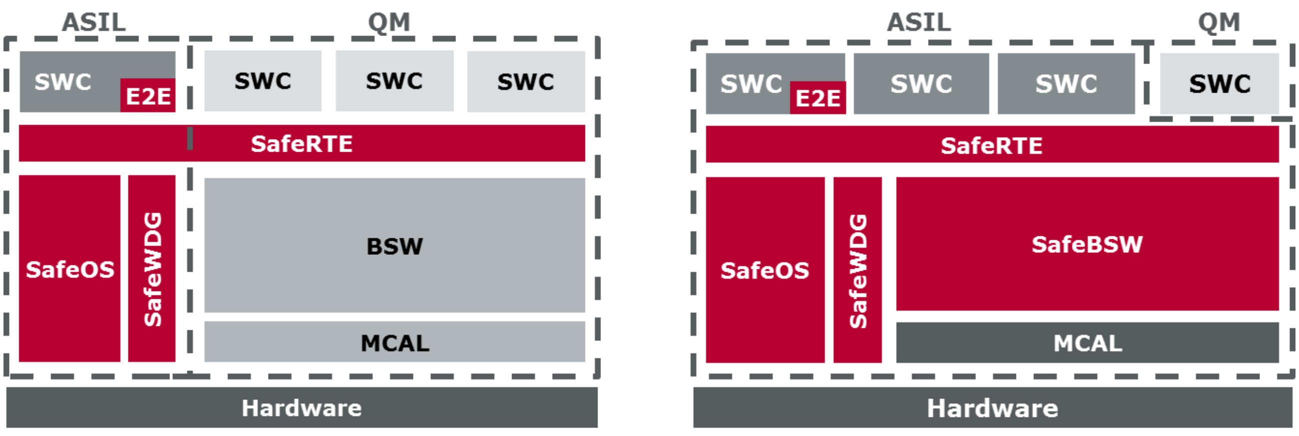

Figure 2-1 BSW in “QM”-partition (left), BSW in “ASIL”-partition (right)

2.3 Safety Concept

It is assumed that the user of MICROSAR Safe is responsible for overall system safety. MICROSAR Safe can provide parts of safety mechanisms to its user. It is also assumed that the user of MICROSAR Safe verifies the robustness and effectiveness of the safety mechanisms within its system based on the configuration of MICROSAR Safe.

2.3.1 Technical Solution

Possible hardware faults (transient or permanent) are detected and handled by the application (user of MICROSAR Safe) according to the required ASIL necessary for the project.

This also implies protection against Single Event Upsets (SEUs) in volatile and non-volatile memory, as well as faults in registers and processing logic.

The hardware manual is assumed to be complete and correct.

MICROSAR Safe does not implement mechanisms to mitigate random hardware faults. Instead, MICROSAR Safe relies on hardware mechanisms, such as ECC memory and lock-step cores. For fail-operational systems redundancy in hardware is assumed to cover hardware faults. Such hardware faults might e.g. occur during communication between the different parts of the fail-operational system.

Note that multi-core controllers with cores that provide different levels of diagnostic coverage (lock-step/non-lock-step cores) are considered during development of the MICROSAR Safe operating system.

However, they also require a detailed analysis of core interdependency by the user of MICROSAR Safe.

It is assumed that systematic hardware faults are handled on system or application level by the user of MICROSAR Safe.

Systematic faults in functionality that is allocated an assumed TSR in MICROSAR Safe are prevented by implementation of an ISO 26262-compliant development process.

It is assumed that development according to an ISO 26262-compliant development process implicitly ensures freedom from interference with respect to memory. I.e. MICROSAR Safe components only write within bounds of memory that they have ownership of.

For fail-operational systems, it is assumed that systematic faults in software in the chain of events must be avoided. Thus, the complete software in the chain of events must implement safety requirements (see e.g. TSR-111108). It is already ensured that no out-of-bounds memory accesses are performed (see above). However, the software in the chain of events must also perform its function within a bounded time (see TSR-6). Moreover, MICROSAR Safe components for fail-operational systems do not trigger hardware exceptions where the only reaction possible is a reset, e.g. by dereferencing a NULL pointer or division by zero.

MICROSAR Safe does not implement redundant (inverted) data storage to mitigate software faults.

It is assumed that software with a different integrity level is separated using a memory protection unit (MPU) (see also TSR-7).

The user of MICROSAR Safe has to ensure that write access to the same memory parts is only possible for software of the same or higher integrity level.

Depending on the requirements of the item, timing and program flow must be monitored. MICROSAR Safe provide mechanisms to implement this monitoring (see also TSR-13, TSR-14 and TSR-15).

MICROSAR Safe does not automatically monitor itself for intended timing and program flow.

For fail-operational systems, it may be required to avoid interference in time and not only to detect it using e.g. a watchdog. It is assumed that the user of MICROSAR Safe adequately ensures freedom from interference with respect to time for this use-case by configuration of MICROSAR Safe.

MICROSAR Safe components without allocated safety requirements are developed according to an ISO 26262-compliant development process to provide an argument for coexistence.

If safety requirements are implemented in components of MICROSAR Safe, Vector ensures that they are locally non-complex.

For locally non-complex components no diverse implementation of algorithms is designed.

2.3.2 Tool Confidence

MICROSAR Safe must be configured using the DaVinci tools. MICROSAR Safe must be compiled by using a compiler for the target hardware.

These tools must be evaluated with respect to their impact on functional safety by the user.

It is assumed that the user of MICROSAR Safe evaluates and qualifies the defined compiler (incl. options) and linker according to ISO 26262 Part 8 Clause 11.

The DaVinci Developer and DaVinci Configurator are assumed to have an impact on functional safety. In the first place, only a tool error detection level of two can be expected.

The resulting tool confidence level (TCL) would require a qualification of those tools.

To ensure freedom from interference with respect to memory for the BSW, Vector provides the MICROSAR Safe Silence Verifier (MSSV).

To ensure freedom from interference with respect to memory and to detect additional faults that may have been introduced by the DaVinci tools for the RTE, Vector provides the RTE Analyzer.

If MSSV and RTE Analyzer are used according to their Technical References and the Safety Manual, the tool confidence level for the DaVinci tools can be reduced to TCL1 by the user of MICROSAR Safe.

Tools by Vector do not implement mechanisms to handle hardware faults on the host development computer.

2.4 Technical Safety Requirements

The items listed in this section are the assumed technical safety requirements on the Safety Element out of Context MICROSAR Safe.

These requirements are expected to match the requirements in the actual item development.

All technical safety requirements (TSRs) are assigned an ASIL D (unless otherwise stated) to service as many projects as possible.

No fault tolerant time intervals are given. Timing depends on the used hardware and its configuration.

It is assumed that the user configures MICROSAR Safe adequately for the intended use.

No safe state is defined since MICROSAR Safe allows the user to define the desired behavior in case of a detected fault.

The Safety Manual provided with a delivery shows the details on the support of a safety feature on component level for a specific version.

2.4.1 Initialization

TSR-1 MICROSAR Safe shall provide a mechanism to initialize itself and its controlled hardware.

It is sensible for safety-related systems to start performing potentially hazardous actions only in a defined and intended state.

For example, adequate setup of clocks may be required to reach required fault tolerant times.

Components of MICROSAR Safe initialize their variables. For post build selectable and post build loadable configurations MICROSAR Safe components use a consistent and defined set of configuration data.

Hardware-specific components of MICROSAR Safe describe in their Technical References what hardware units are controlled.

The user of MICROSAR Safe shall ensure that initialization functions of MICROSAR Safe are called in the right order and at the right point in time.

The user of MICROSAR Safe is also responsible for the startup code and main function.

2.4.2 Data Consistency

TSR-101876 MICROSAR Safe shall provide mechanisms to ensure data consistency.

Concurrent access to resources (e.g. variables) from e.g. task and interrupt level, may lead to data inconsistencies.

MICROSAR Safe provides functions to enable or disable interrupts, spin-locks, resources or abstractions (i.e. exclusive areas) to enable the user of MICROSAR Safe to ensure data consistency.

MICROSAR Safe will not unintentionally enable or disable the mechanisms to ensure data consistency.

MICROSAR Safe also uses this functionality in its own components to ensure data consistency.

2.4.3 Non-volatile Memory

TSR-4 MICROSAR Safe shall provide a mechanism to store data in non-volatile memory.

Calibration or other application data may be safety-related, i.e. if there is data available from non-volatile memory it must not be corrupted by either software or hardware.

MICROSAR Safe implements an **end-to-end protection** in the Non-volatile RAM Manager (NvM) to ensure that data is neither corrupted nor masqueraded in either software or hardware.

Note that hardware may not even start storing data in non-volatile memory or loose it at any time.

Availability of data is, thus, not guaranteed. Availability may be increased by redundantly storing data in non-volatile memory.

The user of MICROSAR Safe must handle unavailability of data on application level.

TSR-5 MICROSAR Safe shall provide a mechanism to retrieve data from non-volatile memory.

Calibration or other application data may be safety-related, i.e. if there is data available from non-volatile memory it must not be corrupted by either software or hardware.

MICROSAR Safe implements an **end-to-end protection** in the Non-volatile RAM Manager (NvM) to ensure that data is neither corrupted nor masqueraded in either software or hardware.

Note that hardware may not even start storing data in non-volatile memory or loose it at any time.

Availability of data is, thus, not guaranteed. Availability may be increased by redundantly storing data in non-volatile memory.

The user of MICROSAR Safe must handle unavailability of data on application level.

Note that if data is written more than once and the latest data in the non-volatile memory gets corrupted, older (uncorrupted) data may be returned to the user of MICROSAR Safe.

2.4.4 Hard Real-time Scheduling

TSR-6 MICROSAR Safe shall provide hard real-time scheduling properties.

Hard real-time scheduling is not required for safety in the first place, since deadline violations can usually be detected and a safe state entered.

However, hard real-time scheduling may support argumentation in some cases.

For fail-operational systems hard real-time scheduling properties are essential for the software that needs to stay operational.

The operating system of MICROSAR Safe implements fixed priority scheduling and the immediate priority ceiling protocol specified by AUTOSAR to guarantee hard real-time scheduling properties.

The operating system of MICROSAR Safe also provides an upper bound for the execution time of its functionality.

Note that fail-operational requirements on a system usually require a second independent channel of control.

Note that fail-operational requirements require analyses of the worst-case execution time (WCET) of the software that needs to stay operational.

2.4.5 Memory Protection

TSR-7 MICROSAR Safe shall provide mechanisms to protect software applications from unspecified memory access.

Partitioning in software is often introduced because of different quality levels of software and different responsibilities of software development on one ECU.

Memory partitioning relies on the memory protection unit (MPU) in hardware for the effectiveness of the mechanism.

MICROSAR Safe configures exactly the configured memory access rights for each task and ISR at the hardware interface of the MPU.

Note that MICROSAR Safe does not necessarily control all available protection units (e.g. system MPUs or peripheral protection units).

The user of MICROSAR Safe is responsible for adequate configuration of the memory partitions.

2.4.6 Communication

2.4.6.1 Inter ECU Communication

TSR-10 MICROSAR Safe shall provide mechanisms to protect communication between ECUs.

Communication between ECUs may be corrupted, unintentionally replayed, lost or masked. To protect against these failure modes MICROSAR Safe provides the end-to-end (E2E) protection mechanism defined by AUTOSAR.

MICROSAR Safe detects repetition, loss, insertion, masking, reodering and corruption of messages between ECUs.

MICROSAR Safe allows the usage of cyclic redundancy checks (CRCs) and cryptographic algorithms to protect the integrity of messages between ECUs.

CRCs provide a certain hamming distance given a polynomial and maximum data block size.

Cryptographic hash functions or message authentication codes (MACs) provide a probabilistic statement on data corruption detection depending on the hash function or MAC, data block size and hash value size.

The user of MICROSAR Safe is responsible for the selection of the E2E profile or algorithm that is adequate to the requirements of the item development.

MICROSAR Safe does not reject messages with a valid CRC, hash or MAC. This is required for fail-operational systems to claim reliability of a single channel. MICROSAR Safe relies on the integrity of the underlying hardware for e.g. verification of a MAC if configured.

TSR-104422 MICROSAR Safe shall provide mechanisms to communicate between ECUs.

The use of end-to-end protection mechanisms requires additional data to be computed in software and sent over the network. For ASIL A it might be acceptable to reduce fault detection capabilities to reduce this overhead.

MICROSAR Safe detects if there are no incoming messages in a configured time frame and if messages have been corrupted on the bus.

MICROSAR Safe does not unintentionally repeat, discard, delay, insert, mask, reorder or corrupt messages in software.

MICROSAR Safe cannot detect if old, repeated or masked data was received. MICROSAR Safe cannot detect if messages are unintentionally reordered or if some messages have been lost.

MICROSAR Safe does not provide mechanisms to detect latent faults in the communication hardware, e.g. runtime check of the CRC check in the CAN controller.

The user of MICROSAR Safe is responsible to decide whether reduced fault detection capabilities are acceptable. This should include an analysis on a per signal base of the underlying system requirements of the item.

Note this TSR is only assigned an ASIL A. For higher ASILs Vector requires the usage of an end-to-end protection mechanism (see TSR-10).

TSR-111108 MICROSAR Safe shall provide mechanisms to communicate between ECUs.

End-to-end protection ensures that repetition, loss, insertion, masking, reordering and corruption of messages between ECUs can be detected. These kinds of faults can be introduced by hardware or software.

For fail-operational systems it is required to eliminate software as a source of the above-mentioned faults on sender and receiver side.

Thus, MICROSAR Safe does not unintentionally repeat, drop, insert, mask, reorder or corrupt messages. MICROSAR Safe might drop messages that have been corrupted by hardware faults.

Usage of end-to-end protection is still required to ensure protection against hardware faults.

2.4.6.2 Intra ECU Communication

TSR-16 MICROSAR Safe shall provide mechanisms to communicate within its applications.

Software components need to communicate in order to fulfill their task.

MICROSAR Safe provides mechanisms to communicate within OS applications without end-to-end protection.

MICROSAR Safe does not unintentionally repeat, delay, insert, mask, corrupt or loose data communicated within OS applications.

MICROSAR Safe assumes protection of the memory against random hardware faults by the system, e.g. via ECC RAM and lock-step mode.

TSR-12 MICROSAR Safe shall provide mechanisms to communicate between its applications.

Mixed ASIL or multi-core systems need to exchange safety-related information between applications.

MICROSAR Safe provides mechanisms to communicate between OS applications without end-to-end protection.

MICROSAR Safe does not unintentionally repeat, delay, insert, mask, corrupt or loose data communicated between OS applications.

MICROSAR Safe assumes protection of the memory against random hardware faults by the system, e.g. via ECC RAM and lock-step mode.

2.4.7 Monitoring of Software

MICROSAR Safe provides monitoring mechanisms to supervise correct execution of software. These mechanisms must be configured by the user of MICROSAR Safe according to the requirements within the context of the item development.

TSR-13 MICROSAR Safe shall provide a mechanism to detect faults in program flow.

Program flow can be corrupted by random hardware faults or software faults.

The user of MICROSAR Safe can configure a graph of the logical program flow that is then supervised by MICROSAR Safe. A transition from one checkpoint to another that is not allowed is detected.

TSR-14 MICROSAR Safe shall provide a mechanism to detect stuck software.

Alive monitoring is used to reset the software or controller in case it is unresponsive.

The user of MICROSAR Safe can configure entities that need to regularly report their alive status that is then supervised by MICROSAR Safe. Omission of a regular report is detected.

TSR-15 MICROSAR Safe shall provide a mechanism to detect deadline violations.

Deadlines are important to reach a safe state within the defined fault tolerant time interval.

The user of MICROSAR Safe can configure a tolerable time interval for each transition in a logical program flow graph that is then supervised by MICROSAR Safe. Violation of a deadline is detected.

TSR-8 MICROSAR Safe shall provide a mechanism to detect time budget violations.

Budgets for task execution times, inter-arrival times and locking times allow to identify the originator of a deadline violation.

The user of MICROSAR Safe can configure budgets for execution times of tasks and category 2 ISRs, inter-arrival times and duration of locks (e.g. resource or interrupt locks) that are then supervised by MICROSAR Safe.

Exhaustion of budgets is detected.

TSR-9 MICROSAR Safe shall provide a mechanism to terminate software applications.

Terminating a complete OS application may be used as a safety mechanisms to mitigate a software fault within this OS application. This terminated OS application typically comprises QM software.

MICROSAR Safe terminates exactly the requested OS application and only upon request.

Note that MICROSAR Safe will not terminate OS applications automatically without configuration or request.

2.4.8 Operating Modes

TSR-100551 MICROSAR Safe shall provide a mechanism to switch between operating modes.

Fault detection and mitigation often result in a degraded or safe state. These modes can be configured and switched in BswM and Rte.

MICROSAR Safe does not unintentionally switch between operating modes. MICROSAR Safe switches to exactly the requested operating mode.

2.4.9 Peripheral In- and output

TSR-17 MICROSAR Safe shall provide a mechanism to read input data from peripheral units.

Input of data from microcontroller peripheral units is required to implement almost any safety-related system.

MICROSAR Safe provides the data read from peripheral units without corruption or unintentional delay.

MICROSAR Safe does not unintentionally reorder nor masquerade data read from peripheral units.

Note that the peripheral hardware may not provide sufficient diagnostic coverage without redundant input.

For example, DIO and SPI drivers provided by Vector support the reading input data from peripheral units as safety feature.

TSR-18 MICROSAR Safe shall provide a mechanism to write output data to peripheral units.

Output of data from microcontroller peripheral units is required to implement almost any safety-related system.

MICROSAR Safe writes the data provided at the software interfaces without corruption or unintentional delay.

MICROSAR Safe does not unintentionally trigger peripheral output units.

MICROSAR Safe does not unintentionally reorder nor masquerade data read from peripheral units.

Note that the peripheral hardware may require additional mechanisms on application or system level to ensure sufficient safety.

For example, DIO and SPI drivers provided by Vector support the output of data to peripheral units as safety feature (e.g. to trigger external watchdogs or switch actuation paths).

2.4.10 Time Synchronization

TSR-111109 MICROSAR Safe shall provide a mechanism synchronize time between different ECUs.

For advanced driver assistance systems (ADAS) data from different sensor ECUs must be synchronized to perform sensor fusion. It is necessary that all participating ECUs must have a synchronized time for the sensor fusion to yield correct results. Moreover, it is required that the synchronized time correlates to the real time in the real world. If e.g. synchronized time and real time drift apart, sensor fusion will not yield correct results anymore.

MICROSAR Safe provides time synchronization using different communication busses like CAN, FlexRay and Ethernet. The used mechanisms for time synchronization differ between the different bus systems. This impacts the achievable accuracy of the synchronized time.

MICROSAR Safe can be used as time master and time slave.

MICROSAR Safe does not claim a synchronized state if the reported time is not synchronized to the master.

MICROSAR Safe relies on the integrity of the local clock source used for synchronization. MICROSAR Safe does not monitor the local clock source. Especially, drift and jumps are not monitored by MICROSAR Safe.

MICROSAR Safe does not detect a loss of accuracy resulting from faults in the local clock source (Ethernet) or unexpected interrupt latency (CAN and FlexRay).

2.5 Assumptions

2.5.1 Environment

2.5.1.1 Safety Concept

The user of MICROSAR Safe shall be responsible for the functional safety concept.

The overall functional safety concept is in the responsibility of the user of MICROSAR Safe. MICROSAR Safe can only provide parts that can be used to implement the functional safety concept of the item.

It is also the responsibility of the user of MICROSAR Safe to configure MICROSAR Safe as intended by the user's safety concept.

The safety concept shall only rely on safety features explicitly described in this document. If a component from MICROSAR Safe does not explicitly describe safety features in this safety manual, this component has been developed according to the methods for ASIL D to provide coexistence with other ASIL components.

> Example: NvM provides safety features for writing and reading of data, the lower layers, i.e. MemIf, Ea, Fee and drivers, only provide the ASIL for coexistence.

The safety concept shall **not** rely on functionality that is **not** explicitly described as safety feature in this document. This functionality may fail silently in case of a detected fault.

> Example: If a potential out-of-bounds memory access, e.g. due to invalid input or misconfiguration, is detected the requested function will not be performed. An error via DET is only reported if error reporting is enabled.

The user of MICROSAR Safe shall adequately address hardware faults.

The components of MICROSAR Safe can support in the detection and handling of some hardware faults (e.g. using watchdog).

MICROSAR Safe does not provide redundant data storage.

The user of MICROSAR Safe especially shall address faults in volatile random access memory, non-volatile memory, e.g. flash or EEPROM, and the CPU.

MICROSAR Safe relies on the adequate detection of faults in memory and the CPU by other means, e.g. hardware. Thus, Vector recommends using lock-step CPUs together with ECC memory.

The user of MICROSAR Safe shall ensure that the reset or powerless state is a safe state of the system.

Vector uses this assumption in its safety analyses and development process.

The user of MICROSAR Safe shall implement a timing monitoring using e.g. a watchdog.

The components of MICROSAR Safe do not provide mechanisms to monitor their own timing behavior.

The watchdog stack that is part of MICROSAR Safe can be used to fulfill this assumption.

If the safety concept also requires logic monitoring, the watchdog stack that is part of MICROSAR Safe can be used to implement it.

The watchdog is one way to perform timing monitoring. Today the watchdog is the most common approach. In future, there may be different approaches e.g. by monitoring using a different ECU.

The timing protection provided by the operating system can only partially replace the usage of a timing monitoring by a watchdog, because e.g. deadline violations cannot be directly detected by the OS timing protection.

The user of MICROSAR Safe shall ensure an end-to-end protection for safety-related communication between ECUs.

The communication components of MICROSAR Safe do not assume sending or receiving as a safety requirement, because considered faults can only be detected using additional information like a cycle counter. Vector always assumes that an end-to-end protection or equivalent mechanism is implemented on application level.

Considered faults in communication are:

> Failure of communication peer
> Message masquerading
> Message corruption
> Unintended message repetition
> Insertion of messages
> Re-sequencing
> Message loss
> Message delay

This requirement can be fulfilled by e.g. using the end-to-end protection wrapper for safety related communication.

The user of MICROSAR Safe shall ensure data consistency for its application.

Data consistency is not automatically provided when using MICROSAR Safe. MICROSAR Safe only provides services to support enforcement of data consistency. Their application is in the responsibility of the user of MICROSAR Safe.

To ensure data consistency in an application, critical sections need to be identified and protected.

To identify critical sections in the code, e.g. review or static code analysis can be used.

To protect critical sections, e.g. the services to disable and enable interrupts provided by the MICROSAR Safe operating system can be used.

To verify correctly implemented protection, e.g. stress testing or review can be used.

Note the AUTOSAR specification with respect to nesting and sequence of calls to interrupt enabling and disabling functions.

2.5.1.2 Use of MICROSAR Safe Components

The user of MICROSAR Safe shall adequately select the type definitions to reflect the hardware platform and compiler environment.

The user of MICROSAR Safe is responsible for selecting the correct platform types (`PlatformTypes.h`) and compiler abstraction (`Compiler.h`). Especially the size of the predefined types shall match the target environment.

Example: A `uint32` shall be mapped to an unsigned integer type with a size of 32 bits.

The user of MICROSAR Safe can use the platform types provided by Vector. Vector has created and verified the platform types mapping according to the information provided by the user of MICROSAR Safe.

The user of MICROSAR Safe shall initialize all components of MICROSAR Safe prior to using them.

This constraint is required by AUTOSAR anyway. Vector uses this assumption in its safety analyses and development process.

Correct initialization can be verified, e.g. during integration testing.

The user of MICROSAR Safe shall only pass valid pointers at all interfaces to MICROSAR Safe components.

Plausibility checks on pointers are performed by MICROSAR Safe (see also SMI-18), but they are limited. MICROSAR Safe components potentially use provided pointers to write to the location in memory.

Also the length and pointer of a buffer provided to a MICROSAR Safe component need to be consistent.

This assumption also applies to QM as well as ASIL components.

This can e.g. be verified using static code analysis tools, reviews and integration testing.

The user of MICROSAR Safe shall only retain pointers provided by MICROSAR Safe where explicitly stated.

Some MICROSAR Safe components provide call-outs to user or application code. The pointers passed to these call-outs may have limited validity.

The user of MICROSAR Safe shall ensure that pointers are not used neither for reading nor for writing after returning from the call-out. Exceptions of this rule are explicitly documented for the respective call-out.

The user of MICROSAR Safe shall enable plausibility checks for the MICROSAR Safe components.

This setting is necessary to introduce defensive programming and increase robustness at the interfaces as required by ISO 26262.

This setting does not enable error reporting to the DET component.

This setting is enforced by an MSSV plug-in.

The user of MICROSAR Safe shall configure and use the interrupt system correctly.

The user of MICROSAR Safe is responsible for a correct and consistent configuration and usage of the interrupt system.

Especially the following topics shall be verified:

> Consistent configuration of interrupt category, level and priority in OS and MCAL modules

> Correct assignment of logical channels/instances to interrupt vectors in case of MCAL modules with multiple channels/instances

> The interrupt controller is configured in a mode which processes interrupts of the same level sequentially to avoid unbounded interrupt nesting"

The user of MICROSAR Safe shall not use functionality of MICROSAR Safe components marked with a BETA-ESCAN.

Functionality marked with a BETA-ESCAN is not thoroughly tested by Vector.

Vector ensures that functionality marked with a BETA-ESCAN can be completely disabled.

The list of ESCANs (incl. BETA-ESCANs) is provided with a delivery.

2.5.1.3 Partitioning

The user of MICROSAR Safe shall ensure that for one AUTOSAR functional cluster (e.g. System Services, Operating System, CAN, COM, etc.) only components from Vector are used.

This assumption is required because of dependencies within the development process of Vector.

This assumption does not apply to the MCAL or the EXT cluster.

Vector may have requirements on MCAL or EXT components depending on the upper layers that are used and provided by Vector. For example, the watchdog driver is considered to have safety requirements allocated to its initialization and triggering services. Details are described in the component specific parts of this safety manual.

This assumption does not apply to components that are not provided by Vector.

In case the partitioning solution is used, this assumption only partially applies to the System Services cluster. Only the Watchdog Manager and Watchdog Interface need to be used from Vector then, because the Watchdog Manager and Watchdog Interface will be placed in separate memory partitions apart from the other System Services components.

The user of MICROSAR Safe shall provide an argument for coexistence for software that resides in the same partition as components from MICROSAR Safe.

Vector considers an ISO 26262-compliant development process for the software as an argument for coexistence (see [1] Part 9 Clause 6). Vector assumes that especially freedom from interference with respect to memory is provided by an ISO 26262-compliant development process.

Redundant data storage as the only measure by the other software is not considered a sufficient measure.

In general Vector components do not implement methods to interface with other software (e.g. components, hooks, callouts) in other partitions. They assume that all interfacing components reside in the same partition. Interfacing components are described in the respective technical reference.

If an argument for coexistence cannot be provided, other means of separation shall be implemented (e.g. trusted or non-trusted function calls).

If ASIL components provided by Vector are used, this requirement is fulfilled.

The user of MICROSAR Safe shall verify that the memory mapping is consistent with the partitioning concept.

The volatile data of every component shall be placed in the associated memory partition. This can be verified e.g. by review of the linker map file.

The memory sections for each component placed in RAM can be identified `<MIP>_START_SEC_VAR[_<xxx>]`, where `<MIP>` is the Module Implementation Prefix of the component.

It is assumed that the MPU is configured to not allow write access by a partition with lower integrity to data from a partition with higher integrity.

2.5.1.4 Resources

The user of MICROSAR Safe shall provide sufficient resources in RAM, ROM, stack and CPU runtime for MICROSAR Safe.

Selection of the microcontroller and memory capacities as well as dimensioning of the stacks is in the responsibility of the user of MICROSAR Safe.

If MICROSAR Safe components have specific requirements, these are documented in the respective Technical Reference document.

2.5.2 Process

The user of MICROSAR Safe shall follow the instructions of the corresponding Technical Reference of the components.

Especially deviations from AUTOSAR specifications are described in the Technical References.

If there are constraints for the implementation of an exclusive area, these are described in the Technical References.

The user of MICROSAR Safe shall verify all code that is modified during integration of MICROSAR Safe.

Code that is typically modified by the user of MICROSAR Safe during integration comprises generated templates, hooks, callouts, or similar.

This assumption also applies if interfaces between components are looped through user-defined functions.

Vector assumes that this verification also covers ISO 26262:6-9. Vector assumes that modified code that belongs to a Vector component, e.g. EcuM callouts or OS trace hooks can at least coexist with this component, because no separation in memory or time is implemented.

Example: Callouts of the EcuM are executed in the context of the EcuM.

Non-trusted functions provided by the Vector operating system can be used to implement a separation in memory in code modified by the user of MICROSAR Safe.

The user of MICROSAR Safe shall only modify source code of MICROSAR Safe that is explicitly allowed to be changed.

Usually no source code of MICROSAR Safe is allowed to be changed by the user of MICROSAR Safe.

The user of MICROSAR Safe can check if the source code was modified by e.g, comparing it to the original delivery.

The user of MICROSAR Safe shall verify generated functions according to ISO 26262:6-9.

Generated functions can be identified when searching through the generated code.

An example of generated functions is the configured rules of the Basic Software Manager (BSWM). Their correctness can only be verified by the user of MICROSAR Safe. Please note, however, that BSWM does not provide safety features.

This requirement does not apply to MICROSAR SafeRTE.

The user of MICROSAR Safe shall execute the MICROSAR Safe Silence Verifier (MSSV).

MSSV is used to detect potential out-of-bounds accesses by Vector's basic software based on inconsistent configuration.

If the report shows “Overall Check Result: Fail", please contact the Safety Manager at Vector. See Section 0 for contact details.

The user of MICROSAR Safe shall perform the integration (ISO 26262:6-10) and verification (ISO 26262:6-11) processes as required by ISO 26262.

Especially the safety mechanisms shall be verified in the final target ECU.

Vector assumes that by performing the integration and verification processes as required by ISO 26262 the generated configuration data, e.g. data tables, task priorities or PDU handles, are sufficiently checked. An additional review of the configuration data is then considered not necessary.

Integration does not apply to a MICROSAR Safe component that consists of several subcomponents. This integration is already performed by Vector. The integration of subcomponents is validated during creation of the Safety Case by Vector based on the configuration handed in by the user of MICROSAR Safe.

However, integration of all MICROSAR Safe components in the specific use-case of the user of MICROSAR Safe is the responsibility of the user of MICROSAR Safe. This also includes the hardware-software integration in the context of the target ECU.

The user of MICROSAR Safe shall ensure that a consistent set of generated configuration is used for verification and production.

Make sure that the same generated files are used for testing and production code. E.g. be aware that configuration can be changed in the DaVinci tools without generating the code again.

Make sure that all generated files have the same configuration basis, i.e. always generate the MICROSAR Safe configuration for all components for a relevant release of the ECU software.

The use of post build loadable is supported but not recommended by Vector. Post build loadable increases complexity of verification and validation processes due to increased number of variants. There is no support by Vector to qualify the post build loadable tool chain provided by Vector to create binary data without a qualified compiler.

The user of MICROSAR Safe shall verify the integrity of the delivery by Vector.

Run the `SIPModificationChecker.exe` and verify that the source code, BSWMD and safety manual files are unchanged.

The user of MICROSAR Safe shall verify the consistency of the binary downloaded into the ECU's flash memory.

This also includes re-programming of flash memory via a diagnostics service. The consistency of the downloaded binary can be checked by the bootloader or the application. MICROSAR Safe assumes a correct program image.

The user of MICROSAR Safe shall evaluate all tools (incl. compiler) that are used by the user of MICROSAR Safe according to ISO 26262:8-11.

Evaluation especially shall be performed for the compiler, linker, debugging and test tools.

Vector provides a guideline for the evaluation of the Tool Confidence Level (TCL) for the tools provided by Vector (e.g. DaVinci Configurator PRO).

Vector has evaluated the tools exclusively used by Vector during the development of MICROSAR Safe.

3 Delivery Process

3.1 Overview

Figure 3-1 provides an overview on the delivery process for series production safety projects.

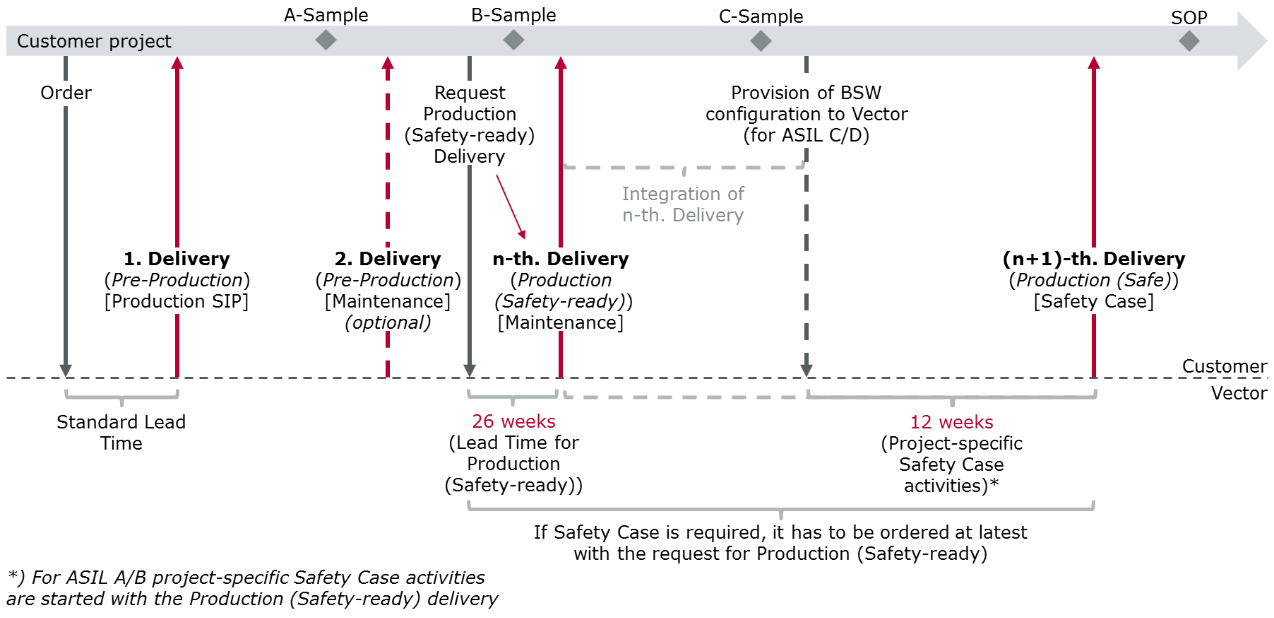
Customer project
A-Sample
B-Sample
C-Sample
SOP
Order
Request Production (Safety-ready) Delivery
Provision of BSW configuration to Vector (for ASIL C/D)
Integration of n-th. Delivery
1. Delivery (Pre-Production) [Production SIP]
2. Delivery (Pre-Production) [Maintenance] (optional)
n-th. Delivery (Production (Safety-ready)) [Maintenance]
(n+1)-th. Delivery (Production (Safe)) [Safety Case]
Customer
Vector
Standard Lead Time
26 weeks (Lead Time for Production (Safety-ready))
12 weeks (Project-specific Safety Case activities)*
If Safety Case is required, it has to be ordered at latest with the request for Production (Safety-ready)
*) For ASIL A/B project-specific Safety Case activities are started with the Production (Safety-ready) delivery

Figure 3-1 Overview of delivery process for safety projects

The first delivery is provided in standard lead time. The release type of the first delivery will be Pre-Production. The Production SIP offer position is invoiced after shipment of the Pre-Production delivery.

The Production (Safety-ready) delivery must be explicitly requested from Vector. This Production (Safety-ready) delivery is the basis for the Safety Case. It has a lead time of 26 weeks. It can be requested by providing a date in the field “Production (Safety-ready) Delivery Date” in a Questionnaire Step II.

The Production (Safe) delivery comprises the Safety Case for the delivered software. The delivery date of the Production (Safe) delivery is confirmed in the Questionnaire Step III which is sent within two weeks after the Production (Safety-ready) delivery.

Vector needs 12 weeks to create the Safety Case. This period includes the necessary time if validation of the MICROSAR Safe component test indicates retesting of components at Vector. With the delivery of the Safety Case the corresponding offer position is invoiced.

Note
If the process described above does not match the schedule of your project, please get in contact with Vector as soon as possible.

Attention
Vector requires ordering Maintenance at least until Start of Production of the project.

3.2 Deliverables

3.2.1 Safety Manual

The Safety Manual defines requirements for integration of MICROSAR Safe. The content of the Safety Manual is specific for the ASIL components for a delivery.

The Safety Manual is provided as starting from the first delivery.

The Safety Manual provided with a Development and Pre-Production delivery has draft status. The Safety Manual provided with the Production (Safety-ready) delivery has final status.

The Safety Manual with draft status may be incomplete with respect to components that are currently under development or hardware-specific.

3.2.2 Verification Tools

Vector provides additional verification tools to verify freedom from interference with respect to memory and consistent configuration.

These verification tools currently are:

- **MICROSAR Safe Silence Verifier**
  - Used to ensure freedom from interference with respect to memory of basic software
  - Enforce configuration constraints
- **RTE Analyzer**
  - Used to ensure freedom from interference with respect to memory of RTE
  - Enforce configuration constraints
  - Provide configuration feedback for integration testing
  - Provided for SafeRTE only
- **WdgVerify**
  - Check of consistent configuration of Watchdog Manager and Watchdog Interface

  - Provided for SafeWDG only
- **OS BackReader**
  - Provide configuration feedback for easier verification
  - Provided for generation six operating systems only

3.2.3 Safety Case

The Safety Case confirms to the user of MICROSAR Safe that all activities required by ISO 26262 have been completed by Vector.

The Safety Case by Vector is required to assume that the delivered software is ISO 26262-compliant ASIL software.

The Safety Case is provided with the Production (Safe) delivery. The Production (Safe) delivery is based on a Production (Safety-ready) delivery.

Attention
For projects with the highest ASIL C or D, you are required to send the configuration of the MICROSAR Safe components to Vector. The configuration is required 12 weeks before Vector can deliver a Safety Case.

You will only receive a Safety Case if you provide the configuration of the MICROSAR Safe components to Vector.

For projects with the highest ASIL A or B, the configuration does not need to be provided to Vector. The Safety Case is provided 12 weeks after the Production (Safety-ready) delivery.

The described timeline assumes that hardware and compiler (incl. options) are not changed after the Production (Safety-ready) delivery.

The configuration of the MICROSAR Safe components is required to validate the MICROSAR Safe component tests.

The Safety Case is fixed for the following characteristics:

- Set of MICROSAR Safe components (incl. their version)
- Configuration of MICROSAR Safe components (for ASIL C/D)
- Microcontroller derivative
- External hardware units (e.g. external watchdogs or SBCs)
- Compiler (incl. version and options)
- Linker (incl. version and options)

If an additional Safety Case is required, e.g. for another project with different characteristics, the Safety Case can be ordered again.

Note
There can only be one Safety Case per CBD number to allow tracking of issues and project status. If the same SIP is supposed to be used for multiple projects, a license copy must be requested.

The Safety Case also comprises the confirmation review of the completeness of the Safety Case by Vector's independent quality department. The confirmation review ensures that all work products required to argue functional safety are available at Vector.
The configuration of MICROSAR Safe is required if ASIL C or ASIL D software is provided by Vector to ensure that components with dedicated safety requirements are sufficiently tested during the component test at Vector.

The configuration is not checked for adequate configuration in the project nor for a configuration compliant to the provided Safety Manual.

The configuration is only checked for components with dedicated safety requirements. The configuration for components without dedicated safety requirements is not checked, since it is assumed that the combination of applied methods (e.g. testing and Silent Analysis) is sufficient to argue coexistence for ASIL C/D.

The parameters for which the Safety Case is valid are documented in the Safety Case. They are identified during check of the configuration.

3.3 Prerequisites

For the creation of the Safety Case for MICROSAR Safe the following information is required 20 weeks prior to the Production (Safety-Ready) delivery:

> Final versions of the microcontroller manuals and microcontroller target specification,

> Final version of the safety manual of the microcontroller,

> Compiler version and options,

> Information about potential requirements and restrictions which are required by 3rd party safety software to all other software, e.g. provided by a safety manual.

For the creation of the Safety Case for MICROSAR Safe the following information is required 12 weeks prior to Safety Case delivery

> Customer configuration of MICROSAR Safe is available (for ASIL C/D).

3.4 Prototype SIPs and Evaluation Bundles

Prototype SIPs and Evaluation Bundles cannot be used for series production. Thus, Vector does not offer a Safety Case for Prototype SIPs and Evaluation Bundles.

Prototype SIPs and Evaluation Bundles comprise the verification tools and a **draft version** of the Safety Manual.

4 Components of MICROSAR Safe

This section comprises details on safety functionality, constraints, dependencies and availability of MICROSAR Safe components.

Please note the availability of a component in the following sections. Availability of HLP components depends on the specific hardware derivative.

Features mentioned in Beta-ESCANS may not be used in safety-related projects.

Unless otherwise noted, each component requires coexistence with all interfacing software.

Attention
If a specific version of a component provides the described safety features below is described in the Safety Manual provided with the delivery.

4.1 Operating System

Short Name	Operating System
Abbreviation	Os
Safety functionality	> Memory partitioning > Immediate priority ceiling protocol and the scheduling algorithm > Timing protection > Schedule tables > Killing of applications > Inter OS application communication And safety features required by other MICROSAR Safe components.
Availability	HLP component; available for the major automotive microcontrollers
Major constraints	For SC3 and SC4 a Memory Protection Unit (MPU) in hardware is necessary. For SC2 and SC4 also a high-resolution timer in hardware is necessary.
Additional dependencies to other components	None; The OS can protect itself from other components using the MPU if configured.

Table 4-1 Component Os

Short Name	Lean Hypervisor
Abbreviation	vLhyp
Safety functionality	> Memory partitioning
Availability	HLP component
Major constraints	A central MPU that can separate between different cores is required. There can be only one partition per core. A partition can span across multiple cores. No timing partitioning is implemented.
Additional dependencies to other components	None

Table 4-2 Component vLhyp

4.2 Microcontroller Abstraction (MCAL)

Short Name	Analog/Digital Conversion Driver
Abbreviation	AdcDrv
Safety functionality	n/a
Availability	Not provided by Vector
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-3 Component AdcDrv

Short Name	CAN Driver
Abbreviation	CanDrv
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	None

Table 4-4 Component CanDrv

Short Name	Core Test
Abbreviation	CorTst
Safety functionality	n/a
Availability	Not provided by Vector
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-5 Component CorTst

Short Name	Hardware Crypto Driver
Abbreviation	Crypto(Hw)
Safety functionality	n/a
Availability	HLP component; A wrapper for AUTOSAR 4.2 to 4.3 interfaces is available as ASIL as well
Major constraints	Hardware specific
Additional dependencies to other components	None

Table 4-6 Component Crypto(Hw)

Short Name	Hardware Crypto Driver for HSM
Abbreviation	Crypto(vHsm)
Safety functionality	n/a
Availability	Available
Major constraints	This is the driver for a vHSM on host side only. The vHSM is QM software only and must be separated by system MPU or equivalent means during integration.
Additional dependencies to other components	n/a

Table 4-7 Component Crypto(vHsm)

Short Name	Digital Input/Output Driver
Abbreviation	DioDrv
Safety functionality	> Must provide I/O functionality as safety feature if used by WdgDrv.
Availability	HLP component
Major constraints	HW specific
Additional dependencies to other components	None

Table 4-8 Component DioDrv

Short Name	EEPROM Driver
Abbreviation	EepDrv
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	None

Table 4-9 Component EepDrv

Short Name	Ethernet Driver
Abbreviation	EthDrv
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	None

Table 4-10 Component EthDrv

Short Name	Ethernet Switch Driver
Abbreviation	EthSwtDrv
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	None

Table 4-11 Component EthSwtDrv

Short Name	Flash Driver
Abbreviation	FlsDrv
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	None

Table 4-12 Component FlsDrv

Short Name	Flash Test
Abbreviation	FlsTst
Safety functionality	n/a
Availability	Not provided by Vector
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-13 Component FlsTst

Short Name	FlexRay Driver
Abbreviation	FrDrv
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	None

Table 4-14 Component FrDrv

Short Name	General Purpose Timer Driver
Abbreviation	GptDrv
Safety functionality	n/a
Availability	Not provided by Vector
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-15 Component GptDrv

Short Name	Input Capture Unit Driver
Abbreviation	IcuDrv
Safety functionality	n/a
Availability	Not provided by Vector
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-16 Component IcuDrv

Short Name	I²C Driver
Abbreviation	vIICDrv
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	None

Table 4-17 Component vIICDrv

Short Name	LIN Driver
Abbreviation	LinDrv
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	None

Table 4-18 Component LinDrv

Short Name	Microcontroller Unit (MCU) Driver
Abbreviation	McuDrv
Safety functionality	> Must provide reset functionality as safety feature
Availability	Not provided by Vector
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-19 Component McuDrv

Short Name	Memory Driver
Abbreviation	vMem
Safety functionality	n/a
Availability	Not scheduled to be part of MICROSAR Safe.
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-20 Component vMem

Short Name	Output Capture Unit Driver
Abbreviation	OcuDrv
Safety functionality	n/a
Availability	Not provided by Vector
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-21 Component OcuDrv

Short Name	Port Driver
Abbreviation	PortDrv
Safety functionality	> Must provide port direction setting functionality as safety feature
Availability	Not provided by Vector
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-22 Component PortDrv

Short Name	Pulse Width Modulation Driver
Abbreviation	PwmDrv
Safety functionality	n/a
Availability	Not provided by Vector
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-23 Component PwmDrv

Short Name	RAM Test
Abbreviation	RamTst
Safety functionality	n/a; Vector considers the RAM Test as specified by AUTOSAR as not sensible to use to increase diagnostic coverage. Thus, it is not part of MICROSAR Safe solution.
Availability	Not scheduled to be part of MICROSAR Safe.
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-24 Component RamTst

Short Name	Serial Peripheral Interface (SPI) Driver
Abbreviation	SpiDrv
Safety functionality	> Sending and receiving features if used with WdgExt or vSbc
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	None

Table 4-25 Component SpiDrv

Short Name	Watchdog Driver
Abbreviation	WdgDrv
Safety functionality	> Provides triggering and mode setting functionality as safety feature
Availability	HLP component
Major constraints	n/a
Additional dependencies to other components	> Requires MICROSAR Safe WDGM and WDGIF.

Table 4-26 Component WdgDrv

Short Name	Wireless Ethernet Driver
Abbreviation	WEth
Safety functionality	n/a
Availability	Not scheduled to be part of MICROSAR Safe
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-27 Component WEth

4.3 External Components (EXT)

Short Name	CAN Transceiver
Abbreviation	CanTrcv
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	None; The CanTrcv for vSbc provides a configuration option to cross partitions in case CanTrcv and vSbc are assigned to different partitions.

Table 4-28 Component CanTrcv

Short Name	External Driver (includes AdcExt, EepExt, FlsExt, EthSwExt and WdgExt)
Abbreviation	DrvExt
Safety functionality	See corresponding MCAL driver
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	See corresponding MCAL driver In case of WdgExt: > If SpiDrv is used the sending and receiving features of the vSbc driver must be provided as safety feature. > If DioDrv is used the I/O functionality must be provided as safety feature.

Table 4-29 Component DrvExt

Short Name	Ethernet Transceiver
Abbreviation	EthTrcv
Safety functionality	n/a
Availability	Not yet available
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-30 Component EthTrcv

Short Name	FlexRay Transceiver
Abbreviation	FrTrcv
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	None

Table 4-31 Component FrTrcv

Short Name	LIN Transceiver
Abbreviation	LinTrcv
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	None; The LinTrcv for vSbc provides a configuration option to cross partitions in case LinTrcv and vSbc are assigned to different partitions.

Table 4-32 Component LinTrcv

Short Name	System Basis Chip Driver
Abbreviation	vSbc
Safety functionality	> Initialize watchdog in SBC > Set mode of watchdog in SBC > Trigger watchdog in SBC
Availability	HLP component
Major constraints	Hardware specific
Additional dependencies to other components	> If SpiDrv is used the sending and receiving features of the SpiDrv must be provided as safety feature. > If DioDrv is used the I/O functionality must be provided as safety feature.

Table 4-33 Component vSbc

Short Name	Wireless Ethernet Transceiver
Abbreviation	WEthTrcv
Safety functionality	n/a
Availability	Not scheduled to be part of MICROSAR Safe.
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-34 Component WEthTrcv

4.4 System Services

Short Name	Basic Software Manager
Abbreviation	BswM
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-35 Component BswM

Short Name	Communication Manager
Abbreviation	ComM
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-36 Component ComM

Short Name	Default Error Tracer
Abbreviation	Det
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-37 Component Det

Short Name	ECU State Manager
Abbreviation	EcuM
Safety functionality	> Validation of all postbuild configurable BSW module configuration parameters. > Set programmable interrupts during the startup phase. > Initialize drivers prior the start of the OS. > Determine the Postbuild configuration data. > Initialize drivers prior Postbuild data is available. > Re-initialize drivers during a restart. > Set the current shutdown target of the ECU. > Initiate the ECU shutdown depending on the shutdown target. > Notify Errors from the ECU management.
Availability	Available
Major constraints	> Defensive behavior is not available.
Additional dependencies to other components	> The Operating System must provide the core ID retrieval service as safety feature.

Table 4-38 Component EcuM

Short Name	Synchronized Time Base Manager
Abbreviation	StbM
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-39 Component StbM

Short Name	Time Service
Abbreviation	Tm
Safety functionality	n/a
Availability	Not scheduled to be part of MICROSAR Safe.
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-40 Component Tm

Short Name	Watchdog Interface
Abbreviation	WdgIf
Safety functionality	Based on WdgM
Availability	Available
Major constraints	Needs to be mapped to a partition with the highest ASIL of the ECU.
Additional dependencies to other components	> WdgM must have same ASIL assigned. > Watchdog driver must provide triggering and mode setting functionality as safety feature. > Watchdog driver may be an internal or external watchdog or a system basis chip (SBC).

Table 4-41 Component WdgIf

Short Name	Watchdog Manager
Abbreviation	WdgM
Safety functionality	> Alive monitoring > Deadline monitoring > Logic monitoring
Availability	Available
Major constraints	Needs to be mapped to a partition with the highest ASIL of the ECU.
Additional dependencies to other components	> WDGIF must be used with same ASIL from MICROSAR Safe.

Table 4-42 Component WdgM

4.5 Crypto Services

Short Name	Cryptographic Service Manager
Abbreviation	Csm
Safety functionality	Only for CSM in the context for AUTOSAR 4.2: > Compute hash functions > Compute MAC functions > Verify MAC functions CSM in the context of AUTOSAR 4.3 does not provide safety features.
Availability	Available
Major constraints	None
Additional dependencies to other components	> If safety functionality is used, the CRY component must provide the cryptographic algorithms as safety feature.

Table 4-43 Component Csm

Short Name	Cryptographic Interface
Abbreviation	CryIf
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-44 Component CryIf

Short Name	Cryptographic Driver (Software)
Abbreviation	Crypto(SW)
Safety functionality	> Cryptographic algorithms
Availability	Available
Major constraints	Only the following primitives are available: - AES-CMAC - AES 128 decryption / encryption - FIPS 186 random number generation - SHA-1 - SHA-2 (256, 384, 512, 512-224, 512-256) - ECDH and EdDSA with curve25519
Additional dependencies to other components	None

Table 4-45 Component Crypto(SW)

4.6 Diagnostic Services

Short Name	Diagnostic Communication Manager
Abbreviation	Dcm
Safety functionality	None
Availability	Process not yet completely implemented, but argument that component fulfills freedom from interference can be provided.
Major constraints	None
Additional dependencies to other components	None

Table 4-46 Component Dcm

Short Name	Diagnostic Event Manager
Abbreviation	Dem
Safety functionality	None; Vector considers degradation via Dem and Fim for safety-related functionality not sensible.
Availability	Process not yet completely implemented, but argument that component fulfills freedom from interference can be provided.
Major constraints	A wrapper component is provided to provide AUTOSAR 4.2 interfaces. The DEM implementation natively provides AUTOSAR 4.3 interfaces. The wrapper component is currently not available as ASIL software.
Additional dependencies to other components	None

Table 4-47 Component Dem

Short Name	Diagnostic Event Synchronizer
Abbreviation	vDes
Safety functionality	n/a
Availability	Not scheduled to be part of MICROSAR Safe.
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-48 Component vDes

Short Name	Diagnostic Request Manager
Abbreviation	vDrm
Safety functionality	n/a
Availability	Not scheduled to be part of MICROSAR Safe.
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-49 Component vDrm

Short Name	Function Inhibition Manager
Abbreviation	Fim
Safety functionality	None; Vector considers degradation via Dem and Fim for safety-related functionality not sensible.
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-50 Component Fim

Short Name	J1939 Diagnostic Communication Manager
Abbreviation	J1939Dcm
Safety functionality	To be determined
Availability	Not yet available
Major constraints	To be determined
Additional dependencies to other components	To be determined

Table 4-51 Component J1939Dcm

4.7 Memory Services

Short Name	EEPROM Abstraction
Abbreviation	Ea
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-52 Component Ea

Short Name	Flash EEPROM Emulation
Abbreviation	Fee
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-53 Component Fee

Short Name	Memory Interface
Abbreviation	MemIf
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-54 Component MemIf

Short Name	Non-volatile Memory Manager
Abbreviation	NvM
Safety functionality	> Loading of data > Storing of data see Section 2.4.3.
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-55 Component NvM

4.8 Communication

Short Name	COM
Abbreviation	Com
Safety functionality	None
Availability	Available
Major constraints	> Com TP Interface (Dynamic Length Signals) is not available > Meta Data Support is not available
Additional dependencies to other components	None

Table 4-56 Component Com

Short Name	COM-based Transformer
Abbreviation	ComXf
Safety functionality	> Transformation of sender/receiver communication
Availability	Available
Major constraints	None
Additional dependencies to other components	> MICROSAR SafeRTE must be used due to verification concept of the transformer.

Table 4-57 Component ComXf

Short Name	End-to-End (E2E) Transformer
Abbreviation	E2eXf
Safety functionality	> Protection of data for communication between ECUs > Verification of data protection for communication between ECUs
Availability	Available
Major constraints	> Supports all AUTOSAR E2E profiles if configured via System Extract
Additional dependencies to other components	> MICROSAR SafeRTE must be used due to verification concept of the transformer. > ComXf and SomeIpXf must have the same ASIL as E2eXf.

Table 4-58 Component E2eXf

Short Name	I-PDU Manager
Abbreviation	IpduM
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-59 Component IpduM

Short Name	Large Data COM
Abbreviation	LdCom
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-60 Component LdCom

Short Name	Network Manager
Abbreviation	Nm
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-61 Component Nm

Short Name	PDU Router
Abbreviation	PduR
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-62 Component PduR

Short Name	Secure On-board Communication
Abbreviation	SecOC
Safety functionality	None; Vector considers the usage of SecOC for end-to-end protection not sensible.
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-63 Component SecOC

Short Name	SOME/IP Transport Protocol
Abbreviation	SomeIpTp
Safety functionality	n/a
Availability	Not scheduled to be part of MICROSAR Safe.
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-64 Component SomeIpTp

Short Name	SOME/IP Transformer
Abbreviation	SomeIpXf
Safety functionality	> Transformation of sender/receiver communication
Availability	Available
Major constraints	None
Additional dependencies to other components	> MICROSAR SafeRTE must be used due to verification concept of the transformer.

Table 4-65 Component SomeIpXf

4.9 Advanced Measurement and Debug (AMD)

Short Name	Debug
Abbreviation	vDbg
Safety functionality	None
Availability	Currently not planned to be part of MICROSAR Safe since not used in series production software.
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-66 Component vDbg

Short Name	Debug Log and Trace
Abbreviation	Dlt
Safety functionality	None
Availability	Currently not planned to be part of MICROSAR Safe since not used in series production software.
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-67 Component Dlt

Short Name	Runtime Measurement
Abbreviation	vRtm
Safety functionality	> Disabling of all module functionality. > **If RTM is not part of series production software, it is not required to be ordered as ASIL software.**
Availability	Available
Major constraints	> RTM must be disabled according to Safety Manual during normal operation.
Additional dependencies to other components	None

Table 4-68 Component vRtm

4.10 XCP

Short Name	Universal Calibration Protocol (XCP)
Abbreviation	Xcp
Safety functionality	> Disabling of all module functionality. > **If Xcp is not part of series production software, it is not required to be ordered as ASIL software.**
Availability	Available
Major constraints	> Xcp must be disabled according to Safety Manual during normal operation.
Additional dependencies to other components	None

Table 4-69 Component Xcp

4.11 CAN

Short Name	J1939 Transport Protocol
Abbreviation	J1939Tp
Safety functionality	To be determined
Availability	Not yet available
Major constraints	To be determined
Additional dependencies to other components	To be determined

Table 4-70 Component J1939Tp

Short Name	J1939 Network Manager
Abbreviation	J1939Nm
Safety functionality	To be determined
Availability	Not yet available
Major constraints	To be determined
Additional dependencies to other components	To be determined

Table 4-71 Component J1939Nm

Short Name	J1939 Resource Manager
Abbreviation	J1939Rm
Safety functionality	To be determined
Availability	Not yet available
Major constraints	To be determined
Additional dependencies to other components	To be determined

Table 4-72 Component J1939Rm

Short Name	CAN Universal Calibration Protocol (XCP)
Abbreviation	CanXcp
Safety functionality	> Disabling of all module functionality.
Availability	Available
Major constraints	> XCP must be disabled according to Safety Manual during normal operation.
Additional dependencies to other components	None

Table 4-73 Component CanXcp

Short Name	CAN Time Synchronization
Abbreviation	CanTSyn
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-74 Component CanTSyn

Short Name	CAN Transport Protocol
Abbreviation	CanTp
Safety functionality	None
Availability	Available
Major constraints	> No unidirectional configurations are supported, i.e. at least one Rx and one Tx SDU must be configured.
Additional dependencies to other components	None

Table 4-75 Component CanTp

Short Name	CAN Interface
Abbreviation	CanIf
Safety functionality	None
Availability	Available
Major constraints	> Single channel optimization is not available > Only one CAN driver is supported > J1939 dynamic address mapping is not available
Additional dependencies to other components	None

Table 4-76 Component CanIf

Short Name	CAN Network Manager
Abbreviation	CanNm
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-77 Component CanNm

Short Name	Direkt OSEK NM
Abbreviation	NmOsek
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-78 Component Direkt OSEK NM

Short Name	CAN Station Manager
Abbreviation	CanSM
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-79 Component CanSM

4.12 LIN

Short Name	LIN Universal Calibration Protocol (XCP)
Abbreviation	vLinXcp
Safety functionality	> Disabling of all module functionality.
Availability	Available
Major constraints	> XCP must be disabled according to Safety Manual during normal operation.
Additional dependencies to other components	None

Table 4-80 Component vLinXcp

Short Name	LIN Interface and LIN Transport Protocol
Abbreviation	LinIf and LinTp
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-81 Component LinIf and LinTp

Short Name	LIN Network Manager
Abbreviation	LinNm
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-82 Component LinNm

Short Name	LIN Station Manager
Abbreviation	LinSM
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-83 Component LinSM

4.13 FlexRay

Short Name	FlexRay AUTOSAR TP
Abbreviation	FrArTp
Safety functionality	n/a
Availability	Not scheduled to be part of MICROSAR Safe.
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-84 Component FrArtp

Short Name	FlexRay Universal Calibration Protocol (XCP)
Abbreviation	FrXcp
Safety functionality	> Disabling of all module functionality.
Availability	Available
Major constraints	> XCP must be disabled according to Safety Manual during normal operation.
Additional dependencies to other components	None

Table 4-85 Component FrXcp

Short Name	FlexRay Time Synchronization
Abbreviation	FrTSyn
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-86 Component FrTSyn

Short Name	FlexRay Transport Protocol
Abbreviation	FrTp
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-87 Component FrTp

Short Name	FlexRay Interface
Abbreviation	FrIf
Safety functionality	None
Availability	Available
Major constraints	> Customized job list execution scheduling is not available > Measurement of the delay/overlapping is not available > Support of 3rd party drivers is not available
Additional dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-88 Component FrIf

Short Name	FlexRay Network Manager
Abbreviation	FrNm
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-89 Component FrNm

Short Name	FlexRay Station Manager
Abbreviation	FrSM
Safety functionality	None
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-90 Component FrSM

4.14 Ethernet

The components (EthXcp, SoAd, TcpIp, EthIf, EthSM, DoIP, UdpNm, Sd and EthTSyn) of Ethernet are currently planned for MICROSAR Safe. Availability is provided upon request. The components (vTls, vEthFw and vEtm) of Ethernet are currently not planned for MICROSAR Safe.

There is no safety functionality planned for the Ethernet components. Major constraints are not yet determined.

4.15 Libraries (LIBS)

Short Name	Cyclic Redundancy Check (CRC) Library
Abbreviation	Crc
Safety functionality	> Calculate CRCs with different polynomials.
Availability	Available
Major constraints	None
Additional dependencies to other components	None

Table 4-91 Component Crc

Short Name	End-to-end (E2E) Library
Abbreviation	E2e
Safety functionality	> Create end-to-end protection for data and verify end-to-end protection information.
Availability	Available
Major constraints	None
Additional dependencies to other components	> The CRC library from MICROSAR Safe is required.

Table 4-92 Component E2e

4.16 Audio/Video Bridging (AVB)

The components (vRtp, vAvTp, vSrp, vPtp) of AVB are currently not part of MICROSAR Safe.

4.17 Vehicle-to-Grid (V2G)

The components (vCanCcCdm, vCanCcGbt, vDns, vExi, vHttp, vScc, vXmlSecurity) of V2G are currently not part of MICROSAR Safe.

4.18 Vehicle-to-X (V2X)

The components (V2xBtp, V2xFac, V2xGn, V2xM) of V2X are currently not part of MICROSAR Safe.

4.19 Input/Output (IO)

The component vDioHwAb is template code only.

4.20 Runtime Environment (RTE)

Short Name	Runtime Environment
Abbreviation	Rte
Safety functionality	> Explicit Sender/Receiver communication with last-is-best semantic > Explicit Sender/Receiver communication with queued semantic > Implicit Sender/Receiver communication > Client/Server communication > Explicit Inter-Runnable Variables > Implicit Inter-Runnable Variables > Per-Instance Memory > Calibration parameters > NVM access > Mode management > Exclusive areas for data consistency > Scheduling of executable entities > Triggering of executable entities
Availability	Available
Major constraints	The following features of RTE are not available for SafeRTE: > Minimum Start Interval > Inter-ECU Client/Server communication > Runnable activation reason > Transformers (will be supported when they are available)
Additional dependencies to other components	The Operating System must provide the following safety features: > Interrupt enabling and disabling > Resource handling > Task handling > Alarm handling > Core ID retrieval > Spinlock handling > Event handling > Scheduling > Schedule table handling > IOC If the OS from MICROSAR Safe is used these dependencies are fulfilled If saving and loading safety-related data is required, the NVM must provide saving and loading as safety feature. If the NVM from MICROSAR Safe is used, this dependency is fulfilled.

Table 4-93 Component Rte

Short Name	E2E Protection Wrapper
Abbreviation	E2ePw
Safety functionality	> End-to-end protection of signal groups
Availability	Available for the following profiles: > Profile 1 > Profile 2 > JLR > VCC
Major constraints	None
Additional dependencies to other components	> E2ePw is using RTE. Either MICROSAR SafeRTE is used or argument for coexistence of RTE with E2EPW must be provided by the user of MICROSAR Safe.

Table 4-94 Component E2ePw

4.21 Hardware Security Module (HSM)

Short Name	Hardware Security Module (HSM)
Abbreviation	vHsm
Safety functionality	n/a
Availability	Not scheduled to be part of MICROSAR Safe.
Major constraints	n/a
Additional dependencies to other components	n/a

Table 4-95 Component vHsm

4.22 Inter-process Communication (IPC)

The components (vIpc, vIpcMem, vIpcMemIf, vIpcAd Posix) of IPC are currently not part of MICROSAR Safe.

4.23 Over-the-Air Update (OTA)

The component (vOtaDl) of OTA are currently not part of MICROSAR Safe.

4.24 OEM-specific Components

OEM-specific components for several OEMs are scheduled to be part of MICROSAR Safe. The following OEM-specific components are available:

- Toyota Motor Company (TMC)
  - Freshness Value Manager (FvM) and Key Manager (KeyM)
- Ford (Ford)
  - Cry (Ford)
- Renault (Renault)
  - Indirect OSEK NM (IndOsekNm)

Moreover, OEM-specific software components are scheduled to be part of MICROSAR Safe. Especially, General Motors (GM) Supplier Utility Modules (SUMs) are developed according to ISO 26262. However, they are not yet available.

More details and information for other OEMs is available upon request.

5 Safety Management

Vector has performed reference functional safety assessments together with an independent assessor for the MICROSAR Safe products.

The Vector-internal quality management department regularly performs functional safety audits.

If you are interested in a project specific functional safety audit and assessment, please contact the safety manager.

Vector's Safety Manager for the MICROSAR Safe solution is:

> Jonas Wolf

> SafetyManagerPES@vector.com

For reasons of efficiency, ASIL A and ASIL B are handled equally at Vector. Also ASIL C and ASIL D are handled equally.

6 Issue Management

The user of MICROSAR Safe is required to provide a contact to Vector that will receive notification about safety-related issues. The user of MICROSAR Safe is also required to notify Vector when the contact changes. The contact provided by the user of MICROSAR Safe must be available even after Start of Production of the project for which the Safety Case is provided.

Vector will notify safety-related issues for 15 years after the Safety Case is delivered.

Vector recommends establishing an email distribution list, e.g. standard-sw@customer.com. This mailbox must be checked within adequate time intervals by the responsible persons.

7 Glossary and Abbreviations

7.1 Glossary

Term	Description
Configuration data	Data that is used to adapt the MICROSAR Safe component to the specific use-case of the user of MICROSAR Safe. Configuration data typically comprises among others: feature selection, routing tables, channel tables, task priorities, memory block descriptions.
Critical section	A section of code that needs to be protected from concurrent access. A critical section may be protected by using the AUTOSAR exclusive area concept.
Generated code	Source code that is generated as a result of the configuration in DaVinci Configurator Pro
MICROSAR Safe	MICROSAR Safe comprises MICROSAR SafeBSW and MICROSAR SafeRTE as Safety Element out of Context. MICROSAR SafeBSW is a set of components, that are developed according to ISO 26262 [1], and are provided by Vector in the context of this delivery. The list of MICROSAR Safe components in this delivery can be taken from the documentation of the delivery.
Partition	A set of memory regions that is accessible by tasks and ISRs. Synonym to OSApplication.
Safety Case	The Safety Case of MICROSAR Safe can be used as an argument that the safety requirements for an item are complete and satisfied by evidence compiled from work products of the safety activities during development. The Safety Case of MICROSAR Safe is progressively compiled by the work products that are generated during the safety lifecycle of the MICROSAR Safe lifecycle. The user of MICROSAR Safe is provided a Safety Case Report that confirms the requested ASIL for the delivered software to the user of MICROSAR Safe.
Safety Manual	The Safety Manual details the requirements for the user of MICROSAR Safe for integration of MICROSAR Safe into the item development.
User of MICROSAR Safe	Integrator and user of components from MICROSAR Safe provided by Vector.

Table 7-1 Glossary

7.2 Abbreviations

Abbreviation	Description
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
BSW	Basis Software
DEM	Diagnostic Event Manager
DET	Development Error Tracer
EAD	Embedded Architecture Designer
ECC	Error Correcting Code
ECU	Electronic Control Unit
EEPROM	Eletronically Ereasable and Programmable Read-only Memory
HIS	Hersteller Initiative Software
HLP	Hardware License Package
ISR	Interrupt Service Routine
MICROSAR	Microcontroller Open System Architecture (the Vector AUTOSAR solution)
PPORT	Provide Port
QM	Quality Management
RAM	Random Access Memory
RTE	Runtime Environment
RPORT	Require Port
SLP	Software License Package
SRS	Software Requirement Specification
SWC	Software Component
SWS	Software Specification
TCL	Tool Confidence Level

Table 7-2 Abbreviations

8 Contact

Visit our website for more information on

> News
> Products
> Demo software
> Support
> Training data
> Addresses

www.vector.com