MICROSAR Safe

Product Information

Version 1.5.0

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

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.

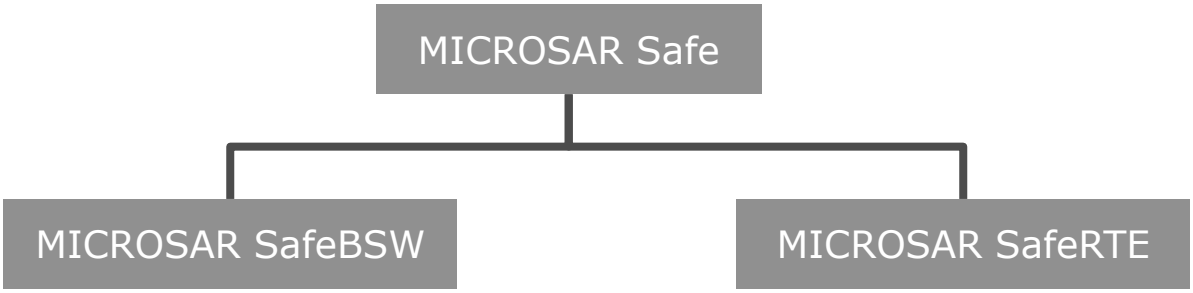

Figure 1-1 Structure of MICROSAR Safe

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.

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

Additional components from MICROSAR SafeBSW 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

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.

The second option is to run the BSW in the same partition as the rest of the safety-related 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. DIO, CANDRV or CANTRCVDRV) need to be assigned in the same partition as the interfacing BSW. Other MCAL and EXT components not interfacing with the BSW (e.g. ADC or PWM) 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.

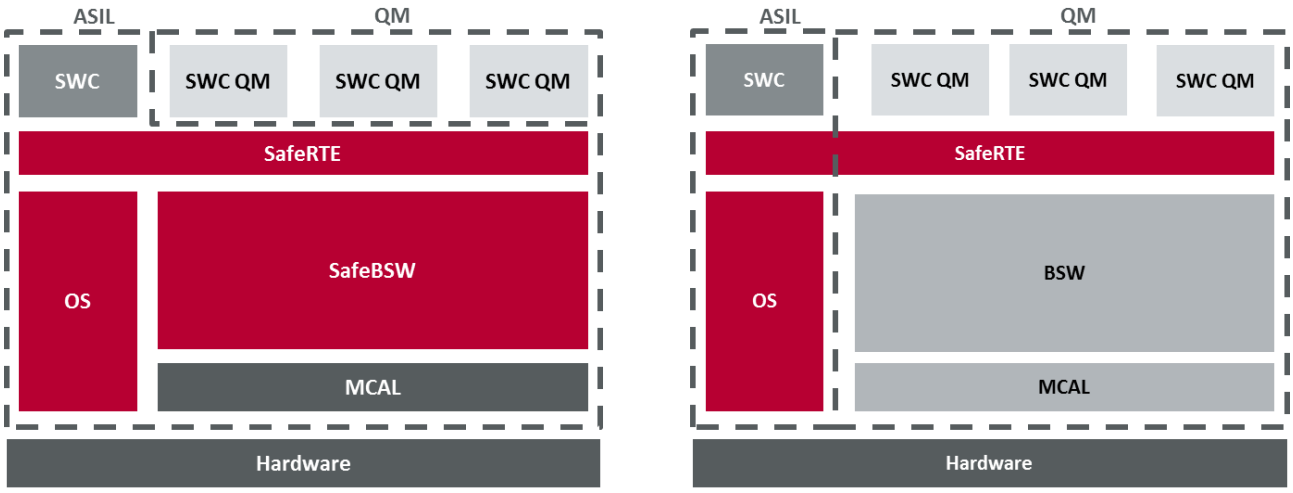

Figure 2-1 BSW in “ASIL”-partition (left), BSW in “QM”-partition (right) (MICROSAR SafeWDG is not shown)

2.3 Assumptions

2.3.1 Technical Safety Requirements

These 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 are assigned an ASIL D 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.

2.3.1.1 Initialization

ID	TSR-1
Requirements text	The system shall initialize the CPU, MPU, watchdog, and operating system.
Rationale	Initialization of the hardware, e.g. clocks, memory protection, scheduling etc. is necessary to enable the other safety requirements.
MICROSAR Safe Feature	The ECU State Manager (EcuM) is responsible for performing the configured driver initialization. After initialization the EcuM starts the operating system. The EcuM distributes the post-built loadable configuration information within the ECU. The documentation of the operating system describes which parts of the CPU it initializes. The startup code and main function is in the responsibility of the user of MICROSAR Safe.

Table 2-1 Initialization

2.3.1.2 Self-test

ID	TSR-2
Requirements text	The system shall perform self-tests based on the requirements of the system.
Rationale	It may be necessary to periodically test individual components of the system to detect latent faults.
MICROSAR Safe Feature	The operating system provides a function to self-test the effectiveness of the MPU settings. The ECU State Manager services callouts to user code that can check RAM consistency after wakeup. Other hardware self-tests are usually performed by MCAL components not developed by Vector. End-to-end protection of communication provides its own fault detection mechanisms.

Table 2-2 Self-test

2.3.1.3 Data consistency

ID	TSR-101876
Requirements text	The system shall provide functions to ensure data consistency.
Rationale	Concurrent access, e.g. from task or interrupt level, must not lead to data inconsistencies.
MICROSAR Safe Feature	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. This functionality is also used within MICROSAR Safe to ensure data consistency.

Table 2-3 Data consistency

2.3.1.4 Reset of ECU

ID	TSR-3
Requirements text	The system shall reset itself in case of a detected fault.
Rationale	Resetting the CPU of the ECU is in most cases an appropriate measure to achieve a safe state. It is assumed that the reset state of a microcontroller leads to the safe state of the ECU and system, since a reset may occur at any time due to e.g. EMC.
MICROSAR Safe Feature	MICROSAR Safe does not reset the ECU on its own (there may be exceptions for the operating system). Functionality from ECU State Manager (setting the shutdown target) can be used to reset the ECU instead, if this is the intended fault reaction.

Table 2-4 Reset of ECU

2.3.1.5 Non-volatile memory

The system must be designed in a way that in case of the absence of non‑volatile data it is still safe (e.g. safe state or degradation).

It cannot be assured that data is saved completely or at all because a reset or loss of energy might happen at any time, e.g. brown-out, black-out.

This also implies that it is in general impossible to guarantee that the latest information is available in the non-volatile memory, e.g. the system is reset before memory stack is even notified to write data to non-volatile memory.

Thus, safety-related functionality may not rely on the availability of data in non-volatile memory.

Since the availability of data in non-volatile memory cannot be guaranteed in any case, the only sensible use-case is reading safety-related calibration data.

Writing of data into non-volatile memory must be verified to assure that the information is available in non-volatile memory. Verification can only be done manually in a protected environment, e.g. at end of line, in a workshop, etc.

ECU software cannot verify if data was written, since at any time a reset could occur and the information that had to be written is lost immediately.

Reading of data does not modify data stored in non-volatile memory. Thus, reading can be used by safety-related functionality. The memory stack has to assure that the read data is identical to the data stored in non-volatile memory.

The absence of data still has to be handled by the application.

The availability may be increased by e.g. redundant storage.

2.3.1.5.1 Saving data

ID	TSR-4
Requirements text	The system shall save information in non-volatile memory.
Rationale	see text in Section 2.3.1.5.
MICROSAR Safe Feature	The intended block is written with the information provided by the application.

Table 2-5 Saving data

2.3.1.5.2 Loading data

ID	TSR-5
Requirements text	The system shall retrieve the last stored information from non-volatile memory.
Rationale	see text in Section 2.3.1.5.
MICROSAR Safe Feature	The intended block is read and the information is provided to the application. The last or any previous completely stored block in non-volatile memory is returned by memory stack. If CRC or ECC protections do not match block data an error is returned. If data is stored redundantly, the redundant information is returned. If no completely stored block is found, an error is returned.

Table 2-6 Loading data

2.3.1.6 Scheduling

2.3.1.6.1 Deterministic, hard real-time scheduling

ID	TSR-6
Requirements text	The system shall execute the specified functions within their respective hard timing limits.
Rationale	Hard real-time scheduling may be used for scheduling safety mechanisms implemented in software. This requirement is even more important for fail-operational systems, where one function may have to work, while another blocks the processor.
MICROSAR Safe Feature	The immediate priority ceiling protocol specified by AUTOSAR is capable of performing this task. The schedule tables specified by AUTOSAR may also be used on top of the scheduling algorithm. The operating system of MICROSAR Safe implements the scheduling algorithm according to the methods for ASIL D required by ISO 26262.

Table 2-7 Deterministic, hard real-time scheduling

2.3.1.7 Partitioning

2.3.1.7.1 Memory partitioning

ID	TSR-7
Requirements text	The system shall protect software applications from unspecified memory access.
Rationale	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 available MPU in hardware for the effectiveness of the mechanism.
MICROSAR Safe Feature	Memory partitioning and context switching using AUTOSAR Operating Feature System SC3 mechanisms is implemented according to the methods for ASIL D required by ISO 26262. Adequate configuration of the memory partitions is in the responsibility of the user of MICROSAR Safe.

Table 2-8 Memory partitioning

2.3.1.7.2 Time partitioning

1.1.1.1.1.1 Timing protection

ID	TSR-8
Requirements text	The system shall detect timing faults in the software.
Rationale	Relying on a watchdog for timing protection is sometimes not sufficiently robust or efficient.
MICROSAR Safe Feature	The operating system of MICROSAR Safe implements the timing protection functionality of SC4 according to the methods for ASIL D required by ISO 26262.

Table 2-9 Timing protection

1.1.1.1.2 Killing of applications

ID	TSR-9
Requirements text	The system shall terminate software applications.
Rationale	In combination with the timing protection this allows the software to continue operation in case of a software fault.
MICROSAR Safe Feature	The operating system of MICROSAR Safe implements the termination of applications according to the methods for ASIL D required by ISO 26262.

Table 2-10 Killing of applications

2.3.1.8 Communication protection

2.3.1.8.1 Inter ECU communication

1.1.1.1.3 End-to-end protection

ID	TSR-10
Requirements text	The system shall protect communication between its elements.
Rationale	Communication has to be protected against corruption, unintended replay and masquerading. The loss of a message must be detected. This can be achieved using the end-to-end (E2E) protection mechanism defined by AUTOSAR.
MICROSAR Safe Feature	MICROSAR Safe implements the E2E functionality according to the methods for ASIL D required by ISO 26262. This also includes the CRC library functionality.

Table 2-11 Communication Protection

1.1.1.1.4 Protection by cryptographic algorithms

ID	TSR-11
Requirements text	The system shall protect communication between its elements using cryptographic hash algorithms to detect accidental corruption of the communication.
Rationale	Cyclic redundancy codes (CRC) provide a specified hamming distance given a polynomial and data block size. Cryptographic hash functions provide a probabilistic statement on data corruption detection depending on the hash function, data block size and hash value size.
MICROSAR Safe Feature	The Cryptographic Service Manager of MICROSAR Safe is implemented according to the methods for ASIL D required by ISO 26262. The Cryptographic Service Manager services the main function and functions to calculate a cryptographic hash function according to AUTOSAR specification.

Table 2-12 Protection by cryptographic algorithms

2.3.1.8.2 Intra ECU communication

1.1.1.1.5 Intra OS application communication

ID	TSR-16
Requirements text	The microcontroller software shall communicate within its applications.
Rationale	Software components need to communicate. Protection of the memory against random hardware faults is expected by the system (e.g. via ECC RAM and lock-step mode).
MICROSAR Safe Feature	The RTE provides services to allow communication of software components within OS applications (intra-partition communication).

Table 2-13 Intra OS application communication

1.1.1.1.6 Inter OS application communication

ID	TSR-12
Requirements text	The microcontroller software shall communicate between its applications.
Rationale	Multi-core systems may need to exchange safety-related information between applications. Protection of the memory against random hardware faults is expected by the system (e.g. via ECC RAM and lock-step mode).
MICROSAR Safe Feature	The operating system of MICROSAR Safe implements the inter-OS application (IOS) functionality according to the methods for ASIL D required by ISO 26262. The RTE provides services to allow communication between OS applications (inter-partition communication).

Table 2-14 Inter OS application communication

2.3.1.9 Watchdog services

2.3.1.9.1 Program flow monitoring

ID	TSR-13
Requirements text	The system shall provide a mechanism to detect faults in program flow.
Rationale	Program flow can be corrupted by random hardware faults or software faults.
MICROSAR Safe Feature	MICROSAR Safe watchdog stack implements program flow monitoring functionality according to the methods for ASIL D required by ISO 26262.

Table 2-15 Program flow monitoring

2.3.1.9.2 Alive monitoring

ID	TSR-14
Requirements text	The system shall provide a mechanism to detect stuck software.
Rationale	Alive monitoring is used to reset the software or controller in case it is unresponsive.
MICROSAR Safe Feature	MICROSAR Safe watchdog stack implements alive monitoring functionality according to the methods for ASIL D required by ISO 26262.

Table 2-16 Alive monitoring

2.3.1.9.3 Deadline monitoring

ID	TSR-15
Requirements text	The system shall provide a mechanism to detect deadline violations.
Rationale	Deadline monitoring using the watchdog stack can be used to implement timing monitoring.
MICROSAR Safe Feature	MICROSAR Safe watchdog stack implements deadline monitoring functionality according to the methods for ASIL D required by ISO 26262.

Table 2-17 Deadline monitoring

2.3.2 Environment

2.3.2.1 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 has to 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 functional safety concept also requires a 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 user of MICROSAR Safe shall ensure an end-to-end protection for safety-relevant 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.

2.3.2.2 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 must match the target environment.

Example: A `uint32` must 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 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.

2.3.2.3 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.

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.

2.3.2.4 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.

2.3.3 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.

The possible implementations of exclusive areas are described in the Technical References.

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

Code that is typically changed 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 MICRSAR 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).

If the report shows “Overall Check Result: Fail", please contact the Safety Manager at Vector. See Section 5 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 must 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.

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, i.e. be aware that configuration can be changed 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 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 has to 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 safety projects.

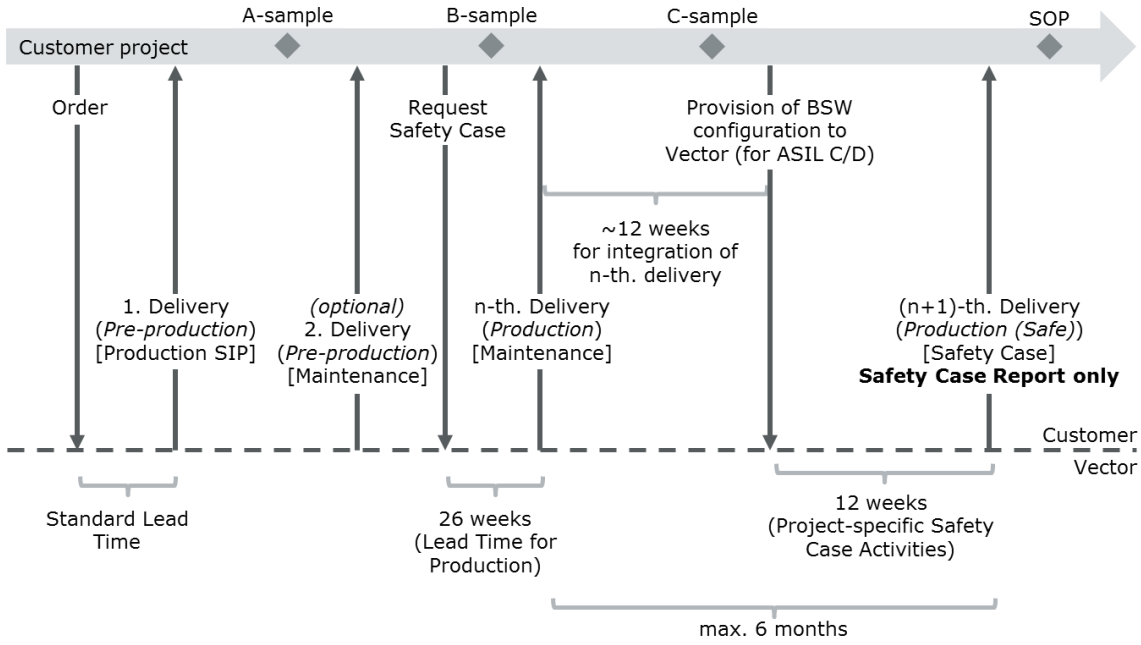

Figure 3-1 Overview of delivery process for safety projects

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

The Production delivery needs to be explicitly requested from Vector. This Production delivery is the basis for the Safety Case. It has a lead time of 26 weeks.

Vector recommends requesting a Safety Case 50 weeks prior to its necessity in the project. This assumes a 12 week integration period at the customer for the Production delivery. 12 weeks are required by Vector to create the Safety Case. This period includes the necessary time if validation of the MICROSAR Safe component test indicates retesting at Vector. With the delivery of the Safety Case the corresponding offer position is invoiced.

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

There may be additional deliveries with release type Pre-production, e.g. to update features.

3.2 Safety Manual

The Safety Manual is provided as a draft starting with the first delivery. The draft Safety Manual may not comprise the parts of components that are currently under development or hardware-specific.

The Safety Manual provided with the Production delivery has a final status.

Prototype SIPs and Evaluation Bundles do not comprise the parts of the Safety Manual of components that are currently under development or hardware-specific.

3.3 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 is provided for a Production delivery with the Production (Safe) 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 can then be provided within 12 weeks after the Production delivery.

The described timeline assumes that hardware and compiler (incl. options) are not changed after the production delivery.

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

A Safety Case is fixed for:

- 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, the safety case can be ordered again.

Attention
Vector requires that the Production delivery for which a Safety Case is requested must not be older than six month. This requirement results from the experience by Vector that there are software changes (e.g. issue fixes, additional feature implementations, changes in configuration) even in late phases of a project. Moreover, final version of (safety) manuals of used controllers and compilers are required.

The Production delivery date as indicated in the offer defines the delivery date of the first Pre-production delivery (see Figure 3-1).

3.4 Prerequisites

For the creation of the Safety Case for MICRSAR Safe the following information is required 20 weeks prior to the Production 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 MICRSAR Safe the following information is required 12 weeks prior to Safety Case delivery

- Customer configuration of MICRSAR Safe is available (for ASIL C/D).

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.

4.1 Operating System

Short Name	Operating System
Abbreviation	OS
Safety functionality	> Self-test the effectiveness of the MPU settings > Schedule tables > Immediate priority ceiling protocol and the scheduling algorithm > Memory partitioning > Timing protection > Killing of applications > Inter OS application communication And all 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.
Dependencies to other components	None

Table 4-1 Component OS

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
Dependencies to other components	n/a

Table 4-2 Component ADCDRV

Short Name	CAN Driver
Abbreviation	CANDRV
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-3 Component CANDRV

Short Name	LIN Driver
Abbreviation	LINDRV
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-4 Component LINDRV

Short Name	FlexRay Driver
Abbreviation	FRDRV
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-5 Component FRDRV

Short Name	Ethernet Driver
Abbreviation	ETHDRV
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-6 Component ETHDRV

Short Name	Ethernet Switch Driver
Abbreviation	ETHSWTDRV
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-7 Component ETHSWTDRV

Short Name	Flash Driver
Abbreviation	FLSDRV
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-8 Component FLSDRV

Short Name	EEPROM Driver
Abbreviation	EEPDRV
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-9 Component EEPDRV

Short Name	Serial Peripheral Interface (SPI) Driver
Abbreviation	SPIDRV
Safety functionality	> Sending and receiving features if used with WDGEXT or SBC
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-10 Component SPIDRV

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

Table 4-11 Component FLSTST

Short Name	I²C Driver
Abbreviation	IICDRV
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-12 Component IICDRV

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

Table 4-13 Component CORTST

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
Dependencies to other components	n/a

Table 4-14 Component RAMTST

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

Table 4-15 Component OCUDRV

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

Table 4-16 Component ICUDRV

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

Table 4-17 Component PWMDRV

Short Name	Hardware CRY Driver
Abbreviation	CRY (HW)
Safety functionality	> Hardware accelerated encryption and decryption
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	n/a

Table 4-18 Component SHEDRV

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

Table 4-19 Component MCUDRV

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

Table 4-20 Component GPTDRV

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

Table 4-21 Component PORTDRV

Short Name	Digital Input/Output Driver
Abbreviation	DIODRV
Safety functionality	> Has to provide I/O functionality as safety feature if used by WDGDRV.
Availability	HLP component
Major constraints	HW specific
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-22 Component DIODRV

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

Table 4-23 Component WDGDRV

4.3 External Components (EXT)

Short Name	CAN Transceiver
Abbreviation	CANTRCV
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-24 Component CANTRCV

Short Name	FlexRay Transceiver
Abbreviation	FRTRCV
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-25 Component FRTRCV

Short Name	LIN Transceiver
Abbreviation	LINTRCV
Safety functionality	None
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-26 Component LINTRCV

Short Name	System Basis Chip Driver
Abbreviation	SBC
Safety functionality	> Trigger watchdog
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	> All interfacing BSW components must reside in the same partition. > If SPIDRV is used the sending and receiving features of the SBC driver must be provided as safety feature. > If DIODRV is used the I/O functionality must be provided as safety feature.

Table 4-27 Component SBC

Short Name	External Driver (includes EXTADC, EEPEXT, FLSEXT, ETHSWTEXT and WDGEXT)
Abbreviation	DRVEXT
Safety functionality	See corresponding MCAL driver
Availability	HLP component
Major constraints	Hardware specific
Dependencies to other components	See corresponding MCAL driver

Table 4-28 Component DRVEXT

4.4 System Services

Short Name	Basic Software Manager
Abbreviation	BSWM
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-29 Component BSWM

Short Name	Communication Manager
Abbreviation	COMM
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-30 Component COMM

Short Name	Development Error Tracer
Abbreviation	DET
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-31 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. > Reset the ECU. > Generate a RAM Hash. > Check the RAM Hash. > 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. > Complete the ECU shutdown process.
Availability	Available
Major constraints	> Defensive behavior is not available.
Dependencies to other components	> All interfacing BSW components must reside in the same partition. > The Operating System has to provide the core ID retrieval service as safety feature.

Table 4-32 Component ECUM

Short Name	Cryptographic Service Manager
Abbreviation	CSM
Safety functionality	> Compute hash functions > Compute MAC functions > Verify MAC functions
Availability	Available
Major constraints	None
Dependencies to other components	> All interfacing BSW components must reside in the same partition. > If safety functionality is used, the CRY component must provide the cryptographic algorithms as safety feature.

Table 4-33 Component CSM

Short Name	Cryptographic Primitive Library
Abbreviation	CRY (SW)
Safety functionality	> Cryptographic algorithms
Availability	Not yet available
Major constraints	To be determined
Dependencies to other components	To be determined

Table 4-34 Component CRY

Short Name	Synchronized Time Base Manager
Abbreviation	STBM
Safety functionality	To be determined.
Availability	Not yet available
Major constraints	To be determined.
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-35 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
Dependencies to other components	n/a

Table 4-36 Component TM

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.
Dependencies to other components	WDGIF must be used with same ASIL from MICROSAR Safe.

Table 4-37 Component WDGM

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.
Dependencies to other components	> WDGM must have same ASIL assigned. > Watchdog driver has to 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 4-38 Component WDGIF

4.5 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	DCM does **only** support the following features: > DCM handles only the following diagnostic services internally: 0x10, 0x3E, 0x28, 0x14, 0x19, 0x85 > Handling of multiple diagnostic clients is supported with pseudo parallel handling (NRC 0x21 on server busy). > Request notification callbacks (OEM/SYS supplier indication/confirmation) are supported. Other diagnostic services are provided as a template that can be verified by the customer.
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-39 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	> OBD-II features are not available > WWH-OBD features are not available > J1939 features are not available
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-40 Component DEM

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	> No cyclic event evaluation > No calibration support (except post-build loadable)
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-41 Component FIM

Short Name	J1939 Diagnostic Communication Manager
Abbreviation	J1939DCM
Safety functionality	To be determined
Availability	Not yet available
Major constraints	To be determined
Dependencies to other components	To be determined

Table 4-42 Component J1939DCM

4.6 Memory Services

Short Name	Non-volatile Memory Manager
Abbreviation	NVM
Safety functionality	> Loading of data > Storing of data see Section 2.3.1.5.
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-43 Component NVM

Short Name	Memory Interface
Abbreviation	MEMIF
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-44 Component MEMIF

Short Name	Flash EEPROM Emulation
Abbreviation	FEE
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-45 Component FEE

Short Name	EEPROM Abstraction
Abbreviation	EA
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-46 Component EA

4.7 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
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-47 Component COM

Short Name	I-PDU Manager
Abbreviation	IPDUM
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-48 Component IPDUM

Short Name	Network Manager
Abbreviation	NM
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-49 Component NM

Short Name	PDU Router
Abbreviation	PDUR
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-50 Component PDUR

Short Name	COM Transformer
Abbreviation	COMXF
Safety functionality	To be determined
Availability	Not yet available
Major constraints	To be determined
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-51 Component COMXF

Short Name	SOME/IP Transformer
Abbreviation	SOMEIPXF
Safety functionality	To be determined
Availability	Not yet available
Major constraints	To be determined
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-52 Component SOMEIPXF

Short Name	End-to-End (E2E) Transformer
Abbreviation	E2EXF
Safety functionality	To be determined
Availability	Not yet available
Major constraints	To be determined
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-53 Component E2EXF

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	Not yet available
Major constraints	To be determined.
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-54 Component SECOC

Short Name	Large Data COM
Abbreviation	LDCOM
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-55 Component LDCOM

4.8 Advanced Measurement and Debug (AMD)

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

Table 4-56 Component DBG

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
Dependencies to other components	n/a

Table 4-57 Component DLT

Short Name	Runtime Measurement
Abbreviation	RTM
Safety functionality	Disabling of all module functionality.
Availability	Available
Major constraints	RTM must be disabled according to Safety Manual during normal operation.
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-58 Component RTM

4.9 XCP

Short Name	Universal Calibration Protocol (XCP)
Abbreviation	XCP
Safety functionality	Disabling of all module functionality.
Availability	Available
Major constraints	XCP must be disabled according to Safety Manual during normal operation.
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-59 Component XCP

4.10 CAN

Short Name	J1939 Transport Protocol
Abbreviation	J1939TP
Safety functionality	To be determined
Availability	Not yet available
Major constraints	To be determined
Dependencies to other components	To be determined

Table 4-60 Component J1939TP

Short Name	J1939 Network Manager
Abbreviation	J1939NM
Safety functionality	To be determined
Availability	Not yet available
Major constraints	To be determined
Dependencies to other components	To be determined

Table 4-61 Component J1939NM

Short Name	J1939 Resource Manager
Abbreviation	J1939RM
Safety functionality	To be determined
Availability	Not yet available
Major constraints	To be determined
Dependencies to other components	To be determined

Table 4-62 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.
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-63 Component CANXCP

Short Name	CAN Time Synchronization
Abbreviation	CANTSYN
Safety functionality	To be determined.
Availability	Not yet available
Major constraints	To be determined.
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-64 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.
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-65 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
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-66 Component CANIF

Short Name	CAN Network Manager
Abbreviation	CANNM
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-67 Component CANNM

Short Name	CAN Station Manager
Abbreviation	CANSM
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-68 Component CANSM

4.11 LIN

Short Name	LIN Universal Calibration Protocol (XCP)
Abbreviation	LINXCP
Safety functionality	Disabling of all module functionality.
Availability	Available
Major constraints	XCP must be disabled according to Safety Manual during normal operation.
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-69 Component LINXCP

Short Name	LIN Transport Protocol and LIN Interface
Abbreviation	LINTP and LINIF
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-70 Component LINTP and LINIF

Short Name	LIN Network Manager
Abbreviation	LINNM
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-71 Component LINNM

Short Name	LIN Station Manager
Abbreviation	LINSM
Safety functionality	None
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-72 Component LINSM

4.12 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
Dependencies to other components	n/a

Table 4-73 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.
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-74 Component FRXCP

Short Name	FlexRay Time Synchronization
Abbreviation	FRTSYN
Safety functionality	To be determined.
Availability	Not yet available
Major constraints	To be determined.
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-75 Component FRTSYN

Short Name	FlexRay Transport Protocol
Abbreviation	FRTP
Safety functionality	None
Availability	Available
Major constraints	n/a
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-76 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
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-77 Component FRIF

Short Name	FlexRay Network Manager
Abbreviation	FRNM
Safety functionality	None
Availability	Available
Major constraints	n/a
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-78 Component FRNM

Short Name	FlexRay Station Manager
Abbreviation	FRSM
Safety functionality	None
Availability	Available
Major constraints	n/a
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-79 Component FRSM

4.13 Ethernet

The components (ETHXCP, SOAD, TCPIP, ETHIF, ETHSM, DOIP, UDPNM, SD, TLS and ETHTSYN) of Ethernet are currently planned for MICROSAR Safe. Availability is provided upon request.

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

4.14 Libraries (LIBS)

Short Name	Cyclic Redundancy Check (CRC) Library
Abbreviation	CRC
Safety functionality	> Calculate CRCs with different polynomials.
Availability	Available
Major constraints	None
Dependencies to other components	All interfacing BSW components must reside in the same partition.

Table 4-80 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
Dependencies to other components	> All interfacing BSW components must reside in the same partition. > The CRC library from MICROSAR Safe is required.

Table 4-81 Component E2E

Short Name	Cryptographic Abstraction Layer / Cryptographic Primitives Library
Abbreviation	CAL/CPL
Safety functionality	To be determined.
Availability	Under development
Major constraints	To be determined.
Dependencies to other components	To be determined.

Table 4-82 Component CAL/CPL

4.15 Audio/Video Bridging (AVB)

The components (AVTP, BMCA and PTP) of AVB are currently not part of MICROSAR Safe.

4.16 V2G

The components (DNS, EXI, HTTP, XML Security and SCC) of V2G are currently not part of MICROSAR Safe.

4.17 Input/Output (IO)

The components IOHWAB and DIOHWAB are template code only. SENT and PSI5 are currently not part of MICROSAR Safe.

4.18 Runtime Environment (RTE)

Short Name	Runtime Environment
Abbreviation	RTE
Safety functionality	To be determined.
Availability	R19
Major constraints	To be determined.
Dependencies to other components	The Operating System has to 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 has to provide saving and loading as safety feature. If the NVM from MICROSAR Safe is used, this dependency is fulfilled.

Table 4-83 Component RTE

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 has to 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 installing an email distribution box, e.g. standard-sw@customer.com. This mail box has to 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