Document Title	Requirements on Health Monitoring
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	878

Document Status	published
Part of AUTOSAR Standard	Foundation
Part of Standard Release	R23-11

Document Change History

Date	**Release**	**Changed by**	**Description**
2023-11-23	R23-11	AUTOSAR Release Management	• Added Chapter History of Constraints and Specification Items
2022-11-24	R22-11	AUTOSAR Release Management	• Merged RS_SAF_21104 with RS_HM_09125 and RS_SAF_21105 with RS_HM_09222 • Added use case for RS_HM_09125 and RS_HM_09235 • Editorial changes
2021-11-25	R21-11	AUTOSAR Release Management	• Update requirements for SystemHealthMonitoring • Add requirements for Mode Dependent Configuration
2020-11-30	R20-11	AUTOSAR Release Management	• Move AP specific requirements to RS_PlatformHealthManagement • Add requirements for SystemHealthMonitoring
2019-11-28	R19-11	AUTOSAR Release Management	• Editorial changes • Changed Document Status from Final to published
2019-03-29	1.5.1	AUTOSAR Release Management	• Editorial changes

▽

△

2018-10-31	1.5.0	AUTOSAR Release Management	• Editorial changes
2018-03-29	1.4.0	AUTOSAR Release Management	• Editorial changes
2017-12-08	1.3.0	AUTOSAR Release Management	• No content changes
2017-10-27	1.2.0	AUTOSAR Release Management	• Initial Release

Disclaimer

This work (specification and/or software implementation) and the material contained in it, as released by AUTOSAR, is for the purpose of information only. AUTOSAR and the companies that have contributed to it shall not be liable for any use of the work.

The material contained in this work is protected by copyright and other types of intellectual property rights. The commercial exploitation of the material contained in this work requires a license to such intellectual property rights.

This work may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only. For any other purpose, no part of the work may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The work has been developed for automotive applications only. It has neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Contents

1 Scope of this document 5

2 How to read this document 6

2.1 Conventions to be used . . . 6

3 Acronyms and abbreviations 7

4 Functional overview 10

5 Requirements traceability 11

6 Requirements specification 13

6.1 Functional requirements . . . 13
6.1.1 Supervision functions . . . 13
6.1.2 Interface to Supervised Entities . . . 14
6.1.3 Features related to supervision functions . . . 15
6.1.4 Features related to support for watchdogs . . . 17
6.1.5 Supported error handling mechanisms . . . 20
6.1.6 Features related to System Health Monitoring . . . 22
6.2 Non functional requirements . . . 26

7 References 27

A History of Constraints and Specification Items 28

A.1 Change History of this document according to AUTOSAR Release R23-11 . . . 28
A.1.1 Added Requirements in R23-11 . . . 28
A.1.2 Changed Requirements in R23-11 . . . 28
A.1.3 Deleted Requirements in R23-11 . . . 28
A.1.4 Added Constraints in R23-11 . . . 28
A.1.5 Changed Constraints in R23-11 . . . 28
A.1.6 Deleted Constraints in R23-11 . . . 28
A.2 Change History of this document according to AUTOSAR Release R22-11 . . . 29
A.2.1 Added Requirements in R22-11 . . . 29
A.2.2 Changed Requirements in R22-11 . . . 29
A.2.3 Deleted Requirements in R22-11 . . . 29
A.2.4 Added Constraints in R22-11 . . . 29
A.2.5 Changed Constraints in R22-11 . . . 29
A.2.6 Deleted Constraints in R22-11 . . . 29

1 Scope of this document

This document specifies requirements on the Health Monitoring.

For this release, this document applies to Adaptive Platform only: the alignment with Classic Platform will be done in a subsequent release. The "Applies to" fields in chapter 6 should be ignored. The alignment with Classic Platform will be done in a subsequent release."

Health Monitoring is required by [1] (under the terms control flow monitoring, external monitoring facility, watchdog, logical monitoring, temporal monitoring, program sequence monitoring) and this specification is supposed to address all relevant requirements from this standard.

Health monitoring has the following error detection functions:

1. Alive supervision - checking if Checkpoints happens with a correct frequency
2. Deadline supervision - checking the delta time between two Checkpoints
3. Logical supervision - checking for correct sequence of execution of Checkpoints
4. Health status supervision - checking if Health Status information is valid

Health monitoring provides also a configurable error handling mechanism in order to recover from errors detected by the previous supervision functions.

The Health Supervision is supposed to be implemented by AUTOSAR classic platform and AUTOSAR adaptive platform. It may be implemented by other platforms as well.

The Health Supervision itself is specified in [2, ASWS Health Monitoring], which specifies the implementation-independent behavior/algorithm of the four supervision functions. System health monitoring allows aggregation and forwarding of health information across several AP/CP or non-AUTOSAR platforms. The specification can be found in [2, ASWS Health Monitoring] and examples how to use them in [3, EXP System Health Monitoring]

2 How to read this document

2.1 Conventions to be used

The representation of requirements in AUTOSAR documents follows the table specified in [TPS_STDT_00078], see Standardization Template, chapter Support for Traceability [4].

The verbal forms for the expression of obligation specified in [TPS_STDT_00053] shall be used to indicate requirements, see Standardization Template, chapter Support for Traceability [4].

3 Acronyms and abbreviations

The glossary below includes acronyms and abbreviations relevant to the specification or implementation of Health Monitoring that are not included in the [5, AUTOSAR glossary].

Abbreviation:	Description:
CM	AUTOSAR Adaptive Communication Management
DM	AUTOSAR Adaptive Diagnostic Management
PHM	Platform Health Management
SE	Supervised Entity
SHM	System Health Monitor

Acronym:	Description:
Alive Counter	An independent data resource in context of a Checkpoint to track and handle its amount of Alive Indications.
Alive Indication	An indication of a Supervised Entity to signal its aliveness by calling a checkpoint used for Alive Supervision.
Alive Supervision	Mechanism to check the timing constraints of cyclic Supervised Entitys to be within the configured min and max limits.
Checkpoint	A point in the control flow of a Supervised Entity where the activity is reported.
Deadline End Checkpoint	A Checkpoint for which Deadline Supervision is configured and which is a ending point for a particular Transition. It is possible that a Checkpoint is both a Deadline Start Checkpoint and Deadline End Checkpoint - if Deadline Supervision is chained.
Deadline Start Checkpoint	A Checkpoint for which Deadline Supervision is configured and which is a starting point for a particular Transition.
Deadline Supervision	Mechanism to check that the timing constraints for execution of the transition from a Deadline Start Checkpoint to a corresponding Deadline End Checkpoint are within the configured min and max limits.
Elementary Supervision Status	Status that represents the current state of an Alive Supervision, Deadline Supervision or Logical Supervision, based on the evaluation (correct/incorrect) of the supervision.
Expired Supervision Cycle	A Supervision Cycle where the Alive Supervision has failed its two escalation steps (Alive Counter fails the expected amount of Alive Indications (including tolerances) more often than the allowed amount of failed reference cycles).

Failed Supervision Reference Cycle	A Supervision Reference Cycle that ends with a detected deviation (including tolerances) between the Alive Counter and the expected amount of Alive Indications.
Global Supervision Status	Cumulative Supervision Status. In Classic Platform, it summarizes the Local Supervision Status of all Supervised Entities. In Adaptive Platform, it is calculated based on a set of Elementary Supervision Status within a single Function Group.
Graph	A set of Checkpoints connected through Transitions, where at least one of Checkpoints is an Initial Checkpoint and there is a path (through Transitions) between any two Checkpoints of the Graph.
Health Channel	Channel providing information about the health status of a (sub)system. This might be the Global Supervision Status of an application, the result any test routine or the status reported by a (sub)system (e.g. voltage monitoring, OS kernel, ECU status, ...).
Health Channel Supervision	Kind of supervision that checks if the health indicators registered by the supervised software are within the tolerances/limits.
Health Monitoring	Supervision of the software behaviour for correct timing and sequence.
Health Status	A set of states that are relevant to the supervised software (e.g. the Global Supervision Status of an application, a Voltage State, an application state, the result of a RAM monitoring algorithm).
Logical Supervision	Kind of online supervision of software that checks if the software (Supervised Entity or set of Supervised Entities) is executed in the sequence defined by the programmer (by the developed code).
Local Supervision Status	Status that represents the current result of Alive Supervision, Deadline Supervision and Logical Supervision of a single Supervised Entity.
Platform Health Management	Health Monitoring for the Adaptive Platform
Supervised Entity	A whole or part of a software component type which is included in the supervision. A Supervised Entity denotes a collection of Checkpoints within the corresponding software component type. A software component type can include zero, one or more Supervised Entities. A Supervised Entity may be instantiated multiple times, in which case each instance is independently supervised.
Supervised Entity Identifier	An Identifier that identifies uniquely a Supervised Entity within an Application.

Supervision Counter	An independent data resource in context of a Supervised Entity which is updated during each supervision cycle and which is used by the Alive Supervision algorithm to perform the check against counted Alive Indications.
Supervision Cycle	The time period in which the cyclic Alive Supervision is performed.
Supervision Mode	An overall state of a microcontroller or virtual machine or or state of a Function Group (in case of Adaptive Platform). Modes are mutually exclusive. A mode can be e.g. Startup, Shutdown, Low-power.
Supervision Reference Cycle	The amount of Supervision Cycles to be used as reference by the Alive Supervision to perform the check of counted Alive Indications (individually for each Supervised Entity).
Local Health Monitor	Local Health Monitor gathers health information of the platform on which it is deployed.

Table 3.1: Acronyms

4 Functional overview

The Health Monitoring is intended to supervise the execution of supervised entities with respect to timing constraints (alive and deadline supervision) and with respect to the required sequence of execution (logical supervision) and with respect to their health (health supervision).

The Health Monitoring can be performed on supervised entities, which can be any software components or groups of software components or Adaptive Applications.

The supervision results, as well as the output of other monitors (e. g. Voltage monitor) can be used to create HealthIndicators, which give an overall health status for features or subsystems.

The following features are provided by the Health Monitoring:

1. Supervision of multiple individual supervised entities located on the microprocessor or virtual machine, having independent supervision constraints.

2. Support for parallel and concurrent execution of supervised entities and for multiple instantiation.

3. Support for different modes of operation, with different behavior of software components depending on mode.

4. Support for multiple hardware watchdogs.

5. Support for several error handling mechanisms.

5 Requirements traceability

The following table references the features specified in [6] and links to the fulfillments of these.

Requirement	Description	Satisfied by
[RS_Main_00001]	Real-Time System Software Platform	[RS_HM_09028] [RS_HM_09125] [RS_HM_09159] [RS_HM_09163] [RS_HM_09169] [RS_HM_09222] [RS_HM_09226] [RS_HM_09235] [RS_HM_09237] [RS_HM_09242] [RS_HM_09243] [RS_HM_09244] [RS_HM_09245] [RS_HM_09246] [RS_HM_09247] [RS_HM_09248] [RS_HM_09249] [RS_HM_09253] [RS_HM_09254] [RS_HM_09257]
[RS_Main_00010]	Safety Mechanisms	[RS_HM_09028] [RS_HM_09125] [RS_HM_09159] [RS_HM_09163] [RS_HM_09169] [RS_HM_09222] [RS_HM_09226] [RS_HM_09235] [RS_HM_09237] [RS_HM_09242] [RS_HM_09243] [RS_HM_09244] [RS_HM_09245] [RS_HM_09246] [RS_HM_09247] [RS_HM_09248] [RS_HM_09249] [RS_HM_09253] [RS_HM_09254] [RS_HM_09257] [RS_HM_09304] [RS_HM_09305]
[RS_Main_00011]	Mechanisms for Reliable Systems	[RS_HM_09028] [RS_HM_09125] [RS_HM_09159] [RS_HM_09163] [RS_HM_09169] [RS_HM_09222] [RS_HM_09226] [RS_HM_09235] [RS_HM_09237] [RS_HM_09242] [RS_HM_09243] [RS_HM_09244] [RS_HM_09245] [RS_HM_09246] [RS_HM_09247] [RS_HM_09248] [RS_HM_09249] [RS_HM_09253] [RS_HM_09254] [RS_HM_09257] [RS_HM_09302] [RS_HM_09308] [RS_HM_09309] [RS_HM_09310]
[RS_Main_00190]	Non-AUTOSAR Software Integration	[RS_HM_09306] [RS_HM_09307]
[RS_Main_00280]	Standardized Automotive Communication Protocols	[RS_HM_09300] [RS_HM_09301]
[RS_Main_00340]	AUTOSAR shall support the continuous timing requirement analysis	[RS_HM_09028] [RS_HM_09125] [RS_HM_09159] [RS_HM_09163] [RS_HM_09169] [RS_HM_09222] [RS_HM_09226] [RS_HM_09235] [RS_HM_09237] [RS_HM_09242] [RS_HM_09243] [RS_HM_09244] [RS_HM_09245] [RS_HM_09246] [RS_HM_09247] [RS_HM_09248] [RS_HM_09249] [RS_HM_09253] [RS_HM_09254] [RS_HM_09257]
[RS_Main_00435]	AUTOSAR shall support automotive microcontrollers	[RS_HM_09028] [RS_HM_09169] [RS_HM_09226] [RS_HM_09244] [RS_HM_09245] [RS_HM_09246] [RS_HM_09247] [RS_HM_09248]
[RS_Main_00460]	AUTOSAR shall standardize methods to organize mode management on Application, ECU and System level	[RS_HM_09304]
[RS_Main_00653]	Means for Functional Modeling	[RS_HM_09303]
[RS_SAF_10005]	AUTOSAR shall provide mechanisms to support safe shutdown and termination of applications, software components, basic software modules and services.	[RS_HM_09222]
[RS_SAF_10006]	AUTOSAR shall provide mechanisms to support safe transition of states in a basic software module, software component, application or service life cycle.	[RS_HM_09222]
[RS_SAF_10030]	AUTOSAR shall provide mechanisms to support safe program execution.	[RS_HM_09222]

▽

△

Requirement	Description	Satisfied by
[RS_SAF_10031]	AUTOSAR shall provide mechanisms to detect program execution time violation	[RS_HM_09125] [RS_HM_09235]

Table 5.1: RequirementsTracing

6 Requirements specification

6.1 Functional requirements

6.1.1 Supervision functions

[RS_HM_09222]{DRAFT} **Health Monitoring shall provide a Logical Supervision** ⌈

Description:	Health Monitoring shall check if the sequence of Checkpoints in a Supervised Entity at runtime is the same as the one that is specified. This shall include: • start of if/else branch (decision node): exactly one of the code branches shall be entered, the choice is runtime-specific depending on logical condition • end of if/else branch (merge node): exactly one of the branches shall be reached so that the join is performed • fork of the flow into concurrent execution (fork node): all concurrent branches shall be entered • join of the flow of concurrent execution (join node): all concurrent branches shall be reached so that the join is performed.
Rationale:	To detect if the sequence in the execution is the same as specified/designed.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	Supervision of any software components: application software components or platform components (e.g. execution manager, state manager).
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340, RS_SAF_10005, RS_SAF_10006, RS_SAF_10030)*

[RS_HM_09125]{DRAFT} **Health Monitoring shall provide an Alive Supervision** ⌈

Description:	Health Monitoring shall check if the frequency of reaching a given Checkpoint in a Supervised Entity matches specified limits.
Rationale:	To detect if a periodic function is executed periodically according to specification/design.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	A safety critical application with alive supervision get stuck at some point in time during execution. HM detects that the supervised application is not alive.
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340, RS_SAF_10031)*

[RS_HM_09235]{DRAFT} **Health Monitoring shall provide a Deadline Supervision** ⌈

Description:	Health Monitoring shall check if the elapsed time between two Checkpoints is within the specified min and max limits, including the detection if the second Checkpoint never arrives.
Rationale:	To detect timeouts or loss of deadlines.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	A safety critical application is developed to reach specific checkpoints in a defined time window and is suddenly not behaving as intended. PHM detects the violation.
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340, RS_SAF_10031)*

6.1.2 Interface to Supervised Entities

[RS_HM_09254]{DRAFT} **Health Monitoring shall provide an interface to Supervised Entities to report the currently reached Checkpoint.** ⌈

Description:	Health Monitoring shall provide an interface to Supervised Entities to report the currently reached Checkpoint by a Supervised Entity, taking into account that a given code location can be achieved from different processes, threads or executed on different cores.
Rationale:	This is the only way how an application can report its progress.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	–
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340)*

[RS_HM_09237]{DRAFT} **Health Monitoring shall provide an interface to Supervised Entities informing them about their Supervision State.** ⌈

Description:	Health Monitoring shall provide an interface informing about Supervision State, including: • which Supervised Entity failed • current Local Supervision Status of each Supervised Entity • current Global Supervision Status of microcontroller or virtual machine • reason why the last error reactions were performed • upcoming microcontroller or virtual machine reset This shall be available by notification and by polling.
Rationale:	Some applications need to know their health/state.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	Reporting of OK/Failed to Supervised Entities.
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340)*

6.1.3 Features related to supervision functions

[RS_HM_09253]{DRAFT} **Health Monitoring shall support mode-dependent behavior of Supervised Entities.** ⌈

Description:	Health Monitoring shall support supervision modes of Supervised entities, where • a Supervised Entity has possibly a different behavior in each Supervision Mode. • a Supervision Mode is shared across all Supervised Entities in case of Classic Platform. • a Supervision mode is shared across multiple Supervised Entity instances in case of Adaptive Platform. • a Supervision Mode is defined as a flat or hierarchical state machine.
Rationale:	In different modes, a Supervised Entity can have a different behavior, e.g. other execution path, other timing.
Dependencies:	–
AppliesTo:	CP, AP

▽

△

Use Case:	In "init" mode, the function init() is supervised with its Checkpoints related to the "init" mode. In "run" mode, the run() function is supervised with its Checkpoints related to the "run" mode.In AP, Supervision Modes are derived from Function Group States.
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340)*

[RS_HM_09257]{DRAFT} **Health Monitoring shall support a variable number of supervised entity occurences at runtime** ⌈

Description:	Health Monitoring shall support a varying number of supervised entity instances at runtime.
Rationale:	The number of active supervised entity instances can change depending on the active mode or processes
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	Modes or configurations can change at runtime and accordingly the number of active processes and supervised entities changes.
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340)*

[RS_HM_09242]{DRAFT} **Health Monitoring shall support the supervision within and across Supervised Entities.** ⌈

Description:	Health Monitoring shall support the supervision (logical, alive and deadline) within one Supervised Entity and across different Supervised Entities.
Rationale:	An application can contain multiple Supervised Entities from which the Global Supervision Status is calculated
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	Activity chains across several activities, where different activities belong to one or to different processes.
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340)*

[RS_HM_09243]{DRAFT} **Health Monitoring shall support the supervision of concurrent and parallel Supervised Entities.** ⌈

Description:	Health Monitoring shall support the supervision of Supervised Entities: • with parallel/concurrent execution • preempted by other Supervised Entities or by any other software • executed on multiple cores or CPUs.
Rationale:	Health Monitoring shall work also for systems with parallel and concurrent execution
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	Systems with parallel execution on multi-core processors.
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340)*

[RS_HM_09163]{DRAFT} **Health Monitoring shall provide configurable tolerances for detected errors and configurable delays of error reactions.** ⌈

Description:	Health Monitoring shall provide configurable tolerances for detected errors.
Rationale:	In case of Alive Supervision, a single failure need not trigger error reaction.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	–
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340)*

6.1.4 Features related to support for watchdogs

This section specifies requirements for support of watchdogs. A watchdog is typically a simple hardware entity that expects a simple certain information within a defined time period. It can also be realized by a more complex system, e.g. by another microcontroller.

[RS_HM_09244]{DRAFT} **Health Monitoring shall support timeout watchdogs.** ⌈

Description:	Health Monitoring shall support simple timeout watchdogs, i.e. watchdogs that require that specific value(s) are written within a defined timeout.
Rationale:	Such hardware watchdogs are broadly available. Moreover, systems exist that apply several watchdogs as a redundancy measure (with a simple timeout watchdog and a complex question-answer watchdog).
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	–
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340, RS_Main_00435)*

[RS_HM_09245]{DRAFT} **Health Monitoring shall support window watchdogs.** ⌈

Description:	Health Monitoring shall support window watchdogs, i.e. where the watchdog requires a correct value to be written within a defined min/max time window.
Rationale:	Window watchdogs are broadly used in automotive systems.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	System using a window watchdog
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340, RS_Main_00435)*

[RS_HM_09246]{DRAFT} **Health Monitoring shall support question-answer watchdogs.** ⌈

Description:	Health Monitoring shall support question-answer watchdogs, i.e. where the response provided to the watchdog depends on question from the watchdog and from the current Health Monitoring results.
Rationale:	Using systems with such a watchdog.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	The question-answer watchdog provides a random value as question, which is used as a seed to the Health Monitoring. The result of the supervision - the signature - is returned to the external watchdog as answer. Only if the answer is sent in time and matches the expected response, the external watchdog is serviced correctly and sends out the next question.

▽

△

Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340, RS_Main_00435)*

[RS_HM_09247]{DRAFT} **Health Monitoring shall support modes of the hardware watchdogs.** ⌈

Description:	Health Monitoring shall support hardware watchdog modes, where by hardware watchdog mode it is meant the set of defined hardware options like current timeout value.
Rationale:	A watchdog can provide modes like: normal, low, off, sleep.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	–
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340, RS_Main_00435)*

[RS_HM_09248]{DRAFT} **Health Monitoring shall support different watchdog realizations.** ⌈

Description:	Health Monitoring shall support different watchdog realizations, including, but not limited to: • internal hardware watchdog (in the microcontroller) • external hardware watchdog • separate dedicated chip (ASIC) • an application on a separate microcontroller
Rationale:	Different watchdog realizations already exist on the market.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	–
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340, RS_Main_00435)*

[RS_HM_09028]{DRAFT} **Health Monitoring shall support multiple watchdogs** ⌈

Description:	Health Monitoring shall support multiple watchdogs, of the same or different type, with the same or different configuration.
Rationale:	There are microprocesors including both an internal and an external watchdog for monitoring the system, as a redundancy mechanism.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	In case the internal watchdog uses the same clock as the CPU, then due to the usage of the same clock, the internal watchdog doesn't recognize the "hang-up" of a system. To achieve a higher robustness an external watchdog is used too.
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340, RS_Main_00435)*

6.1.5 Supported error handling mechanisms

[RS_HM_09159]{DRAFT} **Health Monitoring shall be able to report supervision errors.** ⌈

Description:	As a possible error reaction, Health Monitoring shall report supervision errors, providing information on what kind of error was detected.
Rationale:	Reporting of errors is needed so that they can be logged and analyzed or so that a centralized error reaction can take place.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	Reporting that a Supervised Entity violated its Alive Supervision, but still within limits. Reporting that the entire microcontroller is in such a bad state that it needs to be reset. Handling of the error reported by Health Monitoring by others.
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340)*

[RS_HM_09226]{DRAFT} **Health Monitoring shall be able to wrongly trigger the serviced watchdogs.** ⌈

Description:	As a possible error reaction, Health Monitoring shall be able to wrongly trigger the serviced watchdogs.
Rationale:	In order to provide a quick reset of the microprocessor.

▽

△

Dependencies:	–
AppliesTo:	CP, AP
Use Case:	Typical error reaction provided by hardware watchdogs is a quick reset of the microprocessor. A typical wrong triggering of watchdogs includes: • Immediate generation of a answer to a question (in case of a question-answer watchdog) • Immediate generation of a wrong trigger/notification to the watchdog (timeout watchdog and window watchdog) • Generation of no answer (timeout watchdog and window watchdog)
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340, RS_Main_00435)*

[RS_HM_09169]{DRAFT} **Health Monitoring shall be able to trigger microcontroller reset.** ⌈

Description:	As a possible error reaction, Health Monitoring shall trigger microcontroller reset, including, but not limited to: • Clean microcontroller reset (e.g. with closing all services, closing sockets) • Quick microcontroller reset.
Rationale:	Apart from wrong triggering of watchdog, this is the second main reaction that Health Monitoring can perform to recover from the faulty system state.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	Health manager requesting machine state manager to perform the reset.
Supporting Material:	–

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340, RS_Main_00435)*

6.1.6 Features related to System Health Monitoring

[RS_HM_09300]{DRAFT} **System Health Monitor shall transmit Health Indicators as standardized service events** ⌈

Description:	Health Indicator transmission shall be done in a standardized way as part of a standardized service event.
Rationale:	Health Indicators shall be provided as kind of Health of Service/Subsystem in a platform agnostic standardized way to other modules/platforms so they can be used on platform level for error recovery/degradation
Dependencies:	–
AppliesTo:	FO, CP, AP
Use Case:	E.g. feature HAD is spread over multiple platforms (AP, CP and Non-AUTOSAR). SHM determines Health Indicator and transmits it over standardized Health Indicator event to components using feature HAD.
Supporting Material:	

⌋*(RS_Main_00280)*

[RS_HM_09301]{DRAFT} **SHM shall receive relevant health information from local health monitors** ⌈

Description:	SHM shall provide an interface to receive Health Indicators and Health Information through various communication mechanisms.
Rationale:	Received information is used to determine Health Indicators on System Level, SHM needs to support information reception.
Dependencies:	–
AppliesTo:	FO, CP, AP
Use Case:	Feature HAD is spread over multiple platforms (AP, CP and Non-AUTOSAR). SHM needs health information of those platforms for Health Indicator determination. Health Information can e.g. include Supervision States determined by Platform Health Management.
Supporting Material:	

⌋*(RS_Main_00280)*

[RS_HM_09302]{DRAFT} **Communication between SHM and local health monitors shall be E2E protected** ⌈

Description:	Communication between SHM and Local Health Monitors shall be E2E protected so that it is reliable.
Rationale:	Exchanged data will be used for safety critical decisions and shall be protected against communication errors.

▽

△

Dependencies:	–
AppliesTo:	FO, CP, AP
Use Case:	Unreliable transmission of health information could trigger unnecessary degradation strategies.
Supporting Material:	

⌋(*RS_Main_00011*)

[RS_HM_09308]{DRAFT} **Communication between SHM instances shall be E2E protected** ⌈

Description:	Communication between SHM instances shall be E2E protected so that it is reliable.
Rationale:	Exchanged data will be used for safety critical decisions and shall be protected against communication errors.
Dependencies:	–
AppliesTo:	FO, CP, AP
Use Case:	Unreliable transmission of health information could trigger unnecessary degradation strategies.
Supporting Material:	

⌋(*RS_Main_00011*)

[RS_HM_09309]{DRAFT} **Cyclic communication between SHM and local health monitors shall be used for aliveness checks** ⌈

Description:	Cyclic exchange between local health monitors and SHM is necessary for aliveness determination
Rationale:	It is important to detect a failed platform or SHM instance. If communication is configured with fixed cycle times, a failed sender can be detected on the receiver side by using the regularly exchanged health information as a heartbeat signal.
Dependencies:	–
AppliesTo:	FO, CP, AP
Use Case:	
Supporting Material:	

⌋(*RS_Main_00011*)

[RS_HM_09310]{DRAFT} **Cyclic communication between SHM instances shall be used for aliveness checks** ⌈

Description:	Cyclic exchange between SHM instances is necessary for aliveness determination
Rationale:	It is important to detect a failed platform or SHM instance. If communication is configured with fixed cycle times, a failed sender can be detected on the receiver side by using the regularly exchanged health information as a heartbeat signal.
Dependencies:	–
AppliesTo:	FO, CP, AP
Use Case:	
Supporting Material:	

⌋*(RS_Main_00011)*

[RS_HM_09303]{DRAFT} **SHM shall be platform agnostic** ⌈

Description:	SHM shall be realizable on AP, CP and Non-AUTOSAR platforms.
Rationale:	Integration of SHM is project specific and shall provide maximum flexibility where to deploy SHM as different safety considerations like ASIL levels may influence this decision.
Dependencies:	–
AppliesTo:	FO, CP, AP
Use Case:	Multiple SHM instances are deployed in E/E system. Depending on safety needs (ASIL level) they may be deployed on CP,AP and Non-AUTOSAR platform.
Supporting Material:	

⌋*(RS_Main_00653)*

[RS_HM_09304]{DRAFT} **SHM shall determine Health Indicators.** ⌈

Description:	SHM shall determine Health Indicators as indicators describing whether nominal system performance is met and if system degradations are possible.
Rationale:	Health Indicators on System Level are needed for fail-degraded systems.
Dependencies:	–
AppliesTo:	FO, CP, AP
Use Case:	Automated Driving System has redundant channels. Health Indicator can be used by platforms to react on failure of one channel by activating the redundant channel.
Supporting Material:	

⌋*(RS_Main_00010, RS_Main_00460)*

[RS_HM_09305]{DRAFT} SHM should support redundancy concepts ⌈

Description:	SHM should be implemented with redundancy mechanisms
Rationale:	SHM is a single point of failure for highly safety critical functionality and therefore should be implemented in a redundant way.
Dependencies:	–
AppliesTo:	FO, CP, AP
Use Case:	Multiple SHM instances for fail-operational behavior
Supporting Material:	

⌋*(RS_Main_00010)*

[RS_HM_09306]{DRAFT} SHM shall be able to interact with Non-AUTOSAR software platforms ⌈

Description:	Information exchange with Non-AUTOSAR platforms in order to receive and provide Health Indicators is required.
Rationale:	When looking at System Health there is no rationale for restricting it to only AUTOSAR platforms and inclusion of e.g. health information from GENIVI platforms could be safety relevant
Dependencies:	–
AppliesTo:	FO, CP, AP
Use Case:	Automated Driving system uses Non-AUTOSAR platform to implement User-feedback for switching from L3 to L2 functionality. User feedback is safety relevant and needed for System Health Analysis.
Supporting Material:	

⌋*(RS_Main_00190)*

[RS_HM_09307]{DRAFT} SHM shall be configurable within Abstract Platform Description information ⌈

Description:	SHM can use abstract interface description provided by Abstract Platform Description.
Rationale:	Using abstract description of SHM is good way of modeling platform agnostic behavior with AP and CP.
Dependencies:	–
AppliesTo:	FO, CP, AP
Use Case:	E/E system using different platforms.
Supporting Material:	

⌋*(RS_Main_00190)*

6.2 Non functional requirements

[RS_HM_09249]{DRAFT} **Health Monitoring shall support building safety-related systems.** ⌈

Description:	Health Monitoring shall support building safety-related systems compliant to ISO 26262.
Rationale:	Health Monitoring shall not prevent but facilitate the implementation of safe systems compliant with ISO 26262.
Dependencies:	–
AppliesTo:	CP, AP
Use Case:	Building driving assistance systems.
Supporting Material:	[1, ISO 26262]

⌋*(RS_Main_00001, RS_Main_00010, RS_Main_00011, RS_Main_00340)*

7 References

[1] ISO 26262:2018 (all parts) – Road vehicles – Functional Safety
https://www.iso.org

[2] Specification of Health Monitoring
AUTOSAR_FO_ASWS_HealthMonitoring

[3] Explanation of System Health Monitoring
AUTOSAR_FO_EXP_SystemHealthMonitoring

[4] Standardization Template
AUTOSAR_FO_TPS_StandardizationTemplate

[5] Glossary
AUTOSAR_FO_TR_Glossary

[6] Main Requirements
AUTOSAR_FO_RS_Main

A History of Constraints and Specification Items

Please note that the lists in this chapter also include constraints and specification items that have been removed from the specification in a later version. These constraints and specification items do not appear as hyperlinks in the document.

A.1 Change History of this document according to AUTOSAR Release R23-11

A.1.1 Added Requirements in R23-11

none

A.1.2 Changed Requirements in R23-11

none

A.1.3 Deleted Requirements in R23-11

none

A.1.4 Added Constraints in R23-11

none

A.1.5 Changed Constraints in R23-11

none

A.1.6 Deleted Constraints in R23-11

none

A.2 Change History of this document according to AUTOSAR Release R22-11

A.2.1 Added Requirements in R22-11

none

A.2.2 Changed Requirements in R22-11

Number	Heading
[RS_HM_09125]	Health Monitoring shall provide an Alive Supervision
[RS_HM_09169]	Health Monitoring shall be able to trigger microcontroller reset.
[RS_HM_09222]	Health Monitoring shall provide a Logical Supervision
[RS_HM_09226]	Health Monitoring shall be able to wrongly trigger the serviced watchdogs.
[RS_HM_09235]	Health Monitoring shall provide a Deadline Supervision
[RS_HM_09249]	Health Monitoring shall support building safety-related systems.
[RS_HM_09300]	System Health Monitor shall transmit Health Indicators as standardized service events
[RS_HM_09303]	SHM shall be platform agnostic

Table A.1: Changed Requirements in R22-11

A.2.3 Deleted Requirements in R22-11

none

A.2.4 Added Constraints in R22-11

none

A.2.5 Changed Constraints in R22-11

none

A.2.6 Deleted Constraints in R22-11

none