Document Title	Requirements on Memory Services
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	007
Document Classification	Auxiliary

Document Version	3.0.0
Document Status	Final
Part of Release	4.0
Revision	1

Document Change History

Date	Version	Changed by	Change description
09.12.2009	3.0.0	AUTOSAR Administration	• Added NVM safety mechanism • Added support for debugging and diagnosis • Added shared use of blocks • Legal disclaimer revised
23.06.2008	2.2.1	AUTOSAR Administration	Legal disclaimer revised
12.12.2007	2.2.0	AUTOSAR Administration	• Removal of requirements belonging to the CRC library • Document meta information extended • Small layout adaptations made
26.01.2007	2.1.0	AUTOSAR Administration	• Removed statement about conformance and mandatory requirements • Legal disclaimer revised • “Advice for users” revised • “Revision Information” added
22.02.2006	2.0.0	AUTOSAR Administration	• Requirements on Block Management Types changed • New requirements, related to RAM Block Types, error detection, reduction of write load, configuration. • Requirements on “Spreading of write accesses” an “detection of incomplete writes” removed
09.05.2005	1.0.0	AUTOSAR Administration	Initial Release

Disclaimer

Advice for users

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

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

Table of content

1 Scope of this document 5

2 How to read this document 6

2.1 Conventions used 6
2.2 Requirements structure 7

3 Acronyms and abbreviations 8

4 Requirement Specification 10

4.1 NVRAM-Manager 10
4.1.1 Functional Overview 10
4.1.2 Functional Requirements 10
4.1.2.1 Configuration 10
4.1.2.1.1 [BSW041] Declaration and allocation of application memory 10
4.1.2.1.2 [BSW08534] Classes of RAM data blocks 10
4.1.2.1.3 [BSW08528] Block management type - native 11
4.1.2.1.4 [BSW08529] Block management type - redundant 11
4.1.2.1.5 [BSW08531] Block management type – dataset 12
4.1.2.1.6 [BSW08543] Static configuration of block priority 12
4.1.2.1.7 [BSW08009] Default write protection of blocks 13
4.1.2.1.8 [BSW135] NVRAM configuration ID 13
4.1.2.1.9 [BSW08549] Automatic initialization of RAM data after Software update 13
4.1.2.1.10 [BSW125] Job notification 14
4.1.2.1.11 [BSW08000] Configurable access to multiple (different) devices 14
4.1.2.1.12 [BSW08001] Configuration of consistency check of data 14
4.1.2.1.13 [BSW08551] Configuration of maximum read and write retries. 15
4.1.2.1.14 [BSW08552] Configuration of block identifier verification 15
4.1.2.1.15 [BSW08553] Configuration of write verification for data 15
4.1.2.1.16 [BSW08538] Static configuration of NVRAM blocks being loaded during start-up 16
4.1.2.1.17 [BSW08546] Protection of RAM data blocks against data loss . 16
4.1.2.1.18 [BSW08560] Configuration of shared use of blocks 16
4.1.2.2 Initialization 17
4.1.2.2.1 [BSW08533] Load data blocks from NVRAM to RAM during startup 17
4.1.2.3 Normal Operation 17
4.1.2.3.1 [BSW027] Accessing of non volatile data 17
4.1.2.3.2 [BSW08014] RAM block allocation 18
4.1.2.3.3 [BSW013] Handling of concurrent accesses to NVRAM 18
4.1.2.3.4 [BSW016] Block-wise reading of data 18
4.1.2.3.5 [BSW017] Block-wise writing of data 19
4.1.2.3.6 [BSW08554] Retrying of read and write operations 19
4.1.2.3.7 [BSW08541] Guaranteed processing of accepted write requests 19
4.1.2.3.8 [BSW018] Block-wise restoring of default data 20

4.1.2.3.9 [BSW08548] Automatic initialization without ROM Block 20
4.1.2.3.10 [BSW08547] Distinction between invalidated and inconsistent data 20
4.1.2.3.11 [BSW08550] Marking Blocks modified/unmodified 21
4.1.2.3.12 [BSW08545] Validation of permanent RAM data blocks 21
4.1.2.3.13 [BSW08011] Invalidation of NVRAM data blocks 22
4.1.2.3.14 [BSW08544] Block-wise erasing of NVRAM data 22
4.1.2.3.15 [BSW08007] Selection of datasets 22
4.1.2.3.16 [BSW08542] Job order prioritization 23
4.1.2.3.17 [BSW020] Readout of current status of NVRAM manager operations 23
4.1.2.3.18 [BSW127] Write protect/unprotect function 24
4.1.2.3.19 [BSW030] Consistency/integrity check of data 24
4.1.2.3.20 [BSW08555] Verification of block identifier 24
4.1.2.3.21 [BSW08556] Write verification for data 25
4.1.2.3.22 [BSW034] Quasi-parallel write access 25
4.1.2.3.23 [BSW08558] Removal of all requests associated with a NVRAM block 25
4.1.2.3.24 [BSW08559] Enable shared use of blocks 26
4.1.2.4 Shutdown Operation 26
4.1.2.4.1 [BSW08535] Save data blocks from RAM to NV memory 26
4.1.2.4.2 [BSW08540] Aborting the shut down process 27
4.1.2.5 Fault Operation 27
4.1.2.5.1 [BSW038] Treatable errors shall not affect other software components 27
4.1.2.5.2 [BSW129] Automatic data repair 27
4.1.2.5.3 [BSW08010] Loading of ROM default data 28
4.1.2.5.4 [BSW08015] NV Block security of ECU reprogramming 28
4.1.3 Non-Functional Requirements (Qualities) 28
4.1.3.1 Hardware independence 28
4.1.3.1.1 [BSW011] (Memory) hardware independence 28
4.1.3.2 Usability 29
4.1.3.2.1 [BSW130] Provide information about used memory resources 29
4.1.3.3 Debugging and diagnosis 29
4.1.3.3.1 [BSW08557] Support debugging and diagnosis 29
5 References 30
5.1 Deliverables of AUTOSAR 30

1 Scope of this document

This document specifies requirements on Basic Software Modules of the following software layers:

- Service Layer

Those modules are of the following type:

- NVRAM Management
- Interfaces

The selection of modules is derived from the Basic Software Module List and the AUTOSAR Layered Software Architecture. The following modules are in scope:

- NVRAM Manager

The requirements are structured in the following way:

- General requirements on Basic Software Modules (other document)
- General requirements which apply to all modules of the NVRAM Management
- Module specific requirements

Constraints

First scope for specification of requirements on basic software modules are systems which are not safety relevant. For this reason safety requirements are assigned to medium priority.

2 How to read this document

Each requirement has its unique identifier starting with the prefix “BSW” (for “Basic Software”). For any review annotations, remarks or questions, please refer to this unique ID rather than chapter or page numbers!

2.1 Conventions used

In requirements, the following specific semantics are used (taken from Request for Comment RFC 2119 from the Internet Engineering Task Force IETF)

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119. Note that the requirement level of the document in which they are used modifies the force of these words.

- MUST: This word, or the terms "REQUIRED" or "SHALL", mean that the definition is an absolute requirement of the specification.
- MUST NOT: This phrase, or the phrase „SHALL NOT“, means that the definition is an absolute prohibition of the specification.
- SHOULD: This word, or the adjective "RECOMMENDED", mean that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
- SHOULD NOT: This phrase, or the phrase "NOT RECOMMENDED" mean that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
- MAY: This word, or the adjective „OPTIONAL“, means that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item. An implementation, which does not include a particular option, MUST be prepared to interoperate with another implementation, which does include the option, though perhaps with reduced functionality. In the same vein an implementation, which does include a particular option, MUST be prepared to interoperate with another implementation, which does not include the option (except, of course, for the feature the option provides.)

2.2 Requirements structure

Each module specific chapter contains a short functional description of the Basic Software Module. Requirements of the same kind within each chapter are grouped under the following headlines (where applicable):

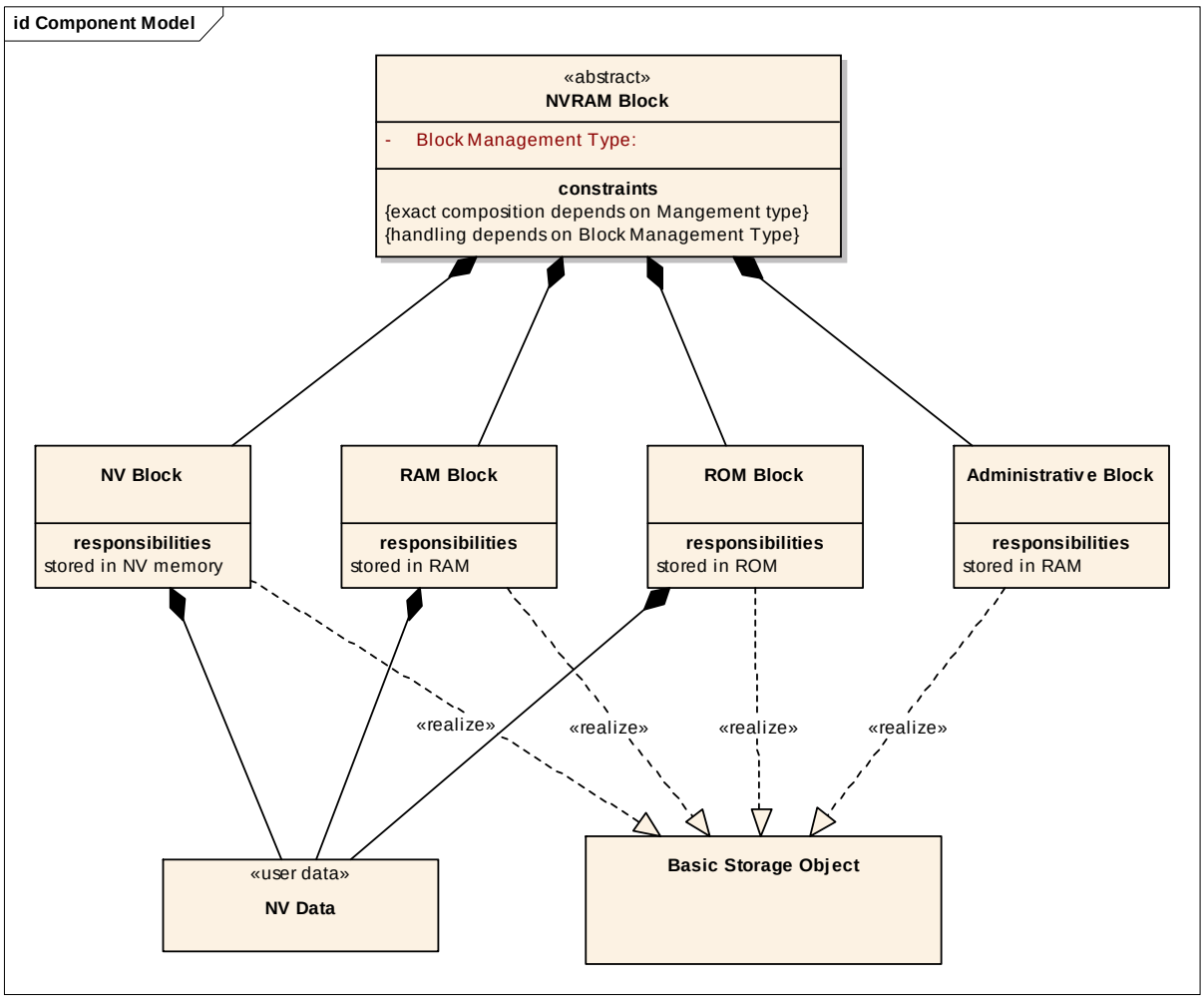
Functional Requirements:
- Configuration (which elements of the module need to be configurable)
- Initialization
- Normal Operation
- Shutdown Operation
- Fault Operation
- ...

Non-Functional Requirements:
- Timing Requirements
- Resource Usage
- Usability
- Output for other WPs (e.g. Description Templates, Tooling, ...)
- ...

3 Acronyms and abbreviations

Acronym:	*Description:*
Basic Storage Object	A “Basic Storage Object” is the smallest entity of a *NVRAM Block*. Several “Basic Storage Objects” can be used to build a *NVRAM Block*. A “Basic Storage Object” can reside in different memory locations (RAM/ROM/NV memory).
NVRAM Block	The “NVRAM Block” is the entire structure, which is needed to administrate and to store a block of *NV data*.
NV data	The data to be stored in the Non-Volatile memory.
Block Management Type	Type of the *NVRAM Block*. It depends on the (configurable) individual composition of a *NVRAM Block* in chunks of different mandatory/optional *Basic Storage Objects* and the subsequent handling of this *NVRAM block*.
NV Block Header	Additional information included in the NV Block if the mechanism “Static Block ID” is enabled.
RAM Block	The “RAM Block” is a *Basic Storage Object*. It represents the part of a *NVRAM Block*, which resides in the RAM. See [BSW08534]
ROM Block	The “ROM Block” is a *Basic Storage Object*. It represents the part of a *NVRAM Block*, which resides in the ROM. The “ROM Block” is an optional part of a *NVRAM Block*.
NV Block	The “NV Block” is a *Basic Storage Object*. It represents the part of a *NVRAM Block*, which resides in the NV memory. The “NV Block” is a mandatory part of a *NVRAM Block*.
Administrative Block	The “Administrative Block” is a *Basic Storage Object*. It resides in RAM. The Administrative Block contains any RAM data, that are necessary to manage the *NVRAM block*, for being able to perform processing on it and to deliver status information. The “Administrative Block” is a mandatory part of a *NVRAM Block*.

The following UML diagram illustrates the relationship between the acronyms defined in the table above:



Abbreviation:	*Description:*
MemHwA	Memory Hardware Abstraction see also [**AUTOSAR_SRS_MEMHW**]

4 Requirement Specification

4.1 NVRAM-Manager

4.1.1 Functional Overview

The Non-Volatile RAM Manager (NVRAM Manager) manages the storage of data in all kinds of non-volatile memory.
The NVRAM manager itself shall be hardware independent, all functionality which is directly accessing hardware, e.g. internal or external EEPROM, emulated EEPROM in internal or external flash, etc. is encapsulated in lower layers of the Basic SW. The NVRAM manager handles concurrent accesses to the non-volatile data and provides reliability mechanisms like checksum protection for single data elements. To be usable in all domains of an automotive system, the NVRAM manager needs to be highly scalable (e.g. define the number and size of request queues, support different block management types, EEPROM Emulation, …).

4.1.2 Functional Requirements

4.1.2.1 Configuration

4.1.2.1.1 [BSW041] Declaration and allocation of application memory

Initiator:	WP Architecture
Date:	12.01.2004
Short Description:	Declaration and allocation of application memory
Type:	Changed after Q&A at 25.05.2004
Importance:	High
Description:	By use of configuration techniques each application shall be enabled to declare the memory requirements at configuration time. This information shall be useable to assign memory areas and to generate the appropriate interfaces. Wrong memory assignments and conflicts in requirements (sufficient memory not available) shall be detected at configuration time.
Rationale:	Realization of higher reliability, interoperability
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.1.2 [BSW08534] Classes of RAM data blocks

Initiator:	WP Memory Service (CRC)
Date:	06.07.2005
Short Description:	Classes of RAM data blocks
Type:	New
Importance:	High
Description:	The NVRAM manager shall support two classes of RAM data blocks. These RAM data blocks are mandatory for data exchange between the NVRAM Manager and SW-Components. They have to be provided by the SW-Components or BSW modules. The assignment to a special NVRAM block can be:

	• permanent, i.e. the RAM data block is assigned to exactly one NVRAM block during configuration time • temporary, i.e. the RAM data block can be assigned to any NVRAM block during runtime
Rationale:	Reduce RAM consumption.
Use Case:	1) Some NVRAM blocks are not frequently used and their data need not to be permanently available in RAM. For example, an application wants to use one RAM block shared for all its NVRAM blocks, which are used mutual exclusive. 2) Diagnostic service wants to read out data from NV that is used by some application, without endangering application's RAM block. It uses one RAM block (must be large enough) to read out any requested block of NV data.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.1.3 [BSW08528] Block management type - native

Initiator:	WP Memory Service (CRC)
Date:	22.06.2005
Short Description:	Block management type – native
Type:	New
Importance:	High
Description:	The NVRAM manager shall allow the configuration of native Block management types. • A native NVRAM block shall provide a basic storage of data and ROM configurable ROM defaults.
Rationale:	Basic block type
Use Case:	Regular NVRAM data without special requirements.
Dependencies:	[BSW08534] Classes of RAM data blocks
Conflicts:	--
Supporting Material:	--

4.1.2.1.4 [BSW08529] Block management type - redundant

Initiator:	WP Memory Service (CRC)
Date:	22.06.2005
Short Description:	Block management type – redundant
Type:	New
Importance:	High
Description:	The NVRAM manager shall allow the configuration of redundant Block management types. A redundant NVRAM Block shall be transparent to application, and it shall be able to provide data in case of inconsistencies (e.g. incomplete write). A redundant NVRAM Block shall be configurable to include default values. • Note: Redundancy does not necessarily mean double (or multiple) storage.

Rationale:	Safety, enhanced data availability, integrity
Use Case:	For saving safety relevant data (e.g. immobilization data)
Dependencies:	[BSW038] Treatable errors shall not affect the application [BSW129] Automatic data repair [BSW08534] Classes of RAM data blocks
Conflicts:	--
Supporting Material:	--

4.1.2.1.5 [BSW08531] Block management type – dataset

Initiator:	WP Memory Service (CRC)
Date:	22.06.2005
Short Description:	NVRAM block type – dataset
Type:	New
Importance:	High
Description:	The NVRAM manager shall allow the configuration of dataset Block management type. Such NVRAM block shall provide a configurable number of selectable elements in NV. • A dataset NVRAM Block shall be configurable to include default values.
Rationale:	Make runtime selection of different datasets in ROM/NVRAM possible.
Use Case:	One ECU is used within several models/country variants of a car. Depending on the car variant, the corresponding data set in NVRAM or ROM is selected.
Dependencies:	[BSW08007] Selection of datasets [BSW08534] Classes of RAM data blocks
Conflicts:	--
Supporting Material:	--

4.1.2.1.6 [BSW08543] Static configuration of block priority

Initiator:	WP Memory Service (CRC)
Date:	09.08.2005
Short Description:	Static configuration of block priority
Type:	New
Importance:	High
Description:	The priority of each NVRAM block shall be statically configurable (pre-compile time) in different levels. One of these levels shall be “immediate”, for those blocks [BSW08542] shall apply.
Rationale:	Flexibility
Use Case:	Writing of crash data and writing of regular data
Dependencies:	[BSW08542] Job order prioritization
Conflicts:	--
Supporting Material:	--

4.1.2.1.7 [BSW08009] Default write protection of blocks

Initiator:	BMW
Date:	09.07.2004
Short Description:	Default write protection of blocks
Type:	New
Importance:	High
Description:	The NVRAM Manager shall allow a static configuration of a default write protection (on/off) for each NVRAM block.
Rationale:	Some data are read-only
Use Case:	Write protection off: data generated by the application itself (e.g. adaptive data, error memory) Write protection on: data given to the ECU from outside (e.g. EOL data, variant coding, parameters)
Dependencies:	[BSW127] Write protect/unprotect function
Conflicts:	--
Supporting Material:	--

4.1.2.1.8 [BSW135] NVRAM configuration ID

Initiator:	FMC
Date:	02.02.2004
Short Description:	The NVRAM memory layout configuration shall have a unique ID.
Type:	Changed after review in DC (06.05.2004)
Importance:	Medium
Description:	The NVRAM manager shall have a configuration identifier that is a unique property of the non-volatile memory configuration. The ID can be either statically assigned to the configuration or it can be calculated from the configuration properties. The ID must be changed if the block configuration changes, i.e. if a block is added or removed, or if its size or type is changed. The ID shall be stored separately and shall be used to determine the validity of the NVRAM contents. The mechanism shall be robust during production (e.g. power-off).
Rationale:	The NVRAM data configuration may change between different versions of the application software. There must be a way to know if the contents of the NVRAM match the configuration expected by the application.
Use Case:	--
Dependencies:	[
Conflicts:	--
Supporting Material:	--

4.1.2.1.9 [BSW08549] Automatic initialization of RAM data after Software update

Initiator:	Siemens VDO
Date:	28.09.2005
Short Description:	Automatic initialization of RAM data after Software update
Type:	Derived from former [BSW08017] (Selection of different start-up conditions for NVRAM blocks)
Importance:	Medium
Description:	The NVRAM manager shall provide functionality to automatically initialize RAM data blocks with ROM defaults, after a software update. This shall be

	configurable per NVRAM block. The whole functionality shall be statically configurable.
Rationale:	Flexibility. So there is no need to update the NV memory during ECU reprogramming.
Use Case:	After ECU reprogramming (mismatch of Configuration IDs) NVRAM blocks are initialized with (new) ROM defaults, except immobilizer data, which shall not be updated.
Dependencies:	[BSW135] NVRAM configuration ID
Conflicts:	--
Supporting Material:	--

4.1.2.1.10 [BSW125] Job notification

Initiator:	BMW
Date:	03.02.2004
Short Description:	Job notification
Type:	New
Importance:	Medium
Description:	For each block a notification shall be configurable (which is initiated at completion of read/write jobs).
Rationale:	Flexibility, integration with application software
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.1.11 [BSW08000] Configurable access to multiple (different) devices

Initiator:	BMW
Date:	04.05.2004
Short Description:	Configurable access to multiple (different) devices
Type:	New
Importance:	High
Description:	The NVRAM manager shall be able to access multiple non-volatile memory devices. The non-volatile memory devices can be of different type.
Rationale:	Flexibility, integration with application software
Use Case:	Multiple Non-volatile memories in one ECU, e.g. external EEPROM and EEPROM Emulation in internal flash
Dependencies:	[BSW011] (Memory) hardware independence
Conflicts:	--
Supporting Material:	--

4.1.2.1.12 [BSW08001] Configuration of consistency check of data

Initiator:	WP Architecture
Date:	09.06.2004
Short Description:	Configuration of Consistency check of data
Type:	New
Importance:	High
Description:	The NVRAM manager shall have a configurable consistency check for each block.
Rationale:	Consistency check might not be used for all blocks

Use Case:	Data that are read or written can be checked immediately by checksum recalculation and comparison.
Dependencies:	[BSW030] Consistency/integrity check of data [BSW023] Detection of incomplete write operations
Conflicts:	--
Supporting Material:	--

4.1.2.1.13 [BSW08551] Configuration of maximum read and write retries

Initiator:	R4.0 concept “Error Handling
Date:	04.12.2008
Short Description:	Configuration of maximum read and write retries
Type:	New
Importance:	High
Description:	For read and write operations of the NVRAM manager, the maximum number of retries in case of an error shall be separately configurable.
Rationale:	Recover from temporary error situation
Use Case:	Reading and writing of data under error conditions.
Dependencies:	[BSW08554] Retrying of read and write operations
Conflicts:	--
Supporting Material:	--

4.1.2.1.14 [BSW08552] Configuration of block identifier verification

Initiator:	R4.0 concept “Error Handling”
Date:	04.12.2008
Short Description:	Configuration of block identifier verification
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide a configurable verification of the unique block identifier when reading an NVRAM block.
Rationale:	Protection against misaddressing of blocks if required
Use Case:	Hardware addressing error can be detected if required
Dependencies:	[BSW08555] Verification of unique block identifier
Conflicts:	--
Supporting Material:	--

4.1.2.1.15 [BSW08553] Configuration of write verification for data

Initiator:	R4.0 concept “Error Handling”
Date:	04.12.2008
Short Description:	Configuration of write verification for data
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide a configurable write verification of data by reading again the previously written data and comparing it. Write verification shall be configurable per NVRAM block.
Rationale:	Verification of correctly stored data if required
Use Case:	Hardware write errors can be detected if required
Dependencies:	[BSW08556] Write verification for data
Conflicts:	--

Supporting Material:	--

4.1.2.1.16 [BSW08538] Static configuration of NVRAM blocks being loaded during start-up

Initiator:	WP Memory Service (CRC)
Date:	19.07.2005
Short Description:	Static configuration of NVRAM blocks being loaded during start-up
Type:	New
Importance:	High
Description:	It shall be statically configurable which blocks are loaded automatically during startup of the NVRAM manager.
Rationale:	Allow flexibility – what is loaded during start-up
Use Case:	Load only those blocks needed to start communication via CAN within strict timing constraints.
Dependencies:	Only applicable for NVRAM Blocks configured with permanent RAM Blocks [BSW08534] Classes of RAM data blocks
Conflicts:	--
Supporting Material:	--

4.1.2.1.17 [BSW08546] Protection of RAM data blocks against data loss

Initiator:	BMW / MEDAG
Date:	20.09.2005
Short Description:	Protection of permanent RAM data blocks against data loss due to reset.
Type:	New
Importance:	High
Description:	For each NVRAM block with permanent RAM data block it shall be a configurable option to enforce mechanisms increasing integrity of RAM data in case of resets.
Rationale:	In case of resets due to voltage drops the RAM content might remain valid. This cannot be detected (or guaranteed) by all platforms. It increases robustness, if the most recent data (from RAM) can be delivered at startup.
Use Case:	A currently moving sun-roof control can neither be fed with outdated position data after a reset occurred (see [BSW08011]) nor it is acceptable to loose the position on every voltage drop.
Dependencies:	[BSW08545] Validation of RAM data and update of integrity information
Conflicts:	--
Supporting Material:	--

4.1.2.1.18 [BSW08560] Configuration of shared use of blocks

Initiator:	R4.0 “memory related concepts”
Date:	29.04.2009
Short Description:	Enable shared use of blocks
Type:	New
Importance:	High
Description:	Each NVRAM block shall be configurable for shared access.
Rationale:	This is on a block-by-block basisto keep memory consumption low.
Use Case:	--

Dependencies:	[BSW08559] Enable shared use of blocks
Conflicts:	--
Supporting Material:	--

4.1.2.2 Initialization

4.1.2.2.1 [BSW08533] Load data blocks from NVRAM to RAM during startup

Initiator:	WP Memory Service (CRC)
Date:	22.06.2005
Short Description:	Load data blocks from NVRAM to RAM during startup
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide a service to check and load those NVRAM blocks, configured to have a permanent RAM data block to RAM. This service has to be called by the ECU state manager once during start-up before the application is running.
Rationale:	The application needs RAM blocks with valid data after ECU start-up.
Use Case:	ECU Start-up: provide NV data for application.
Dependencies:	[BSW08534] Classes of RAM data blocks [BSW08538] Static configuration of NVRAM blocks being loaded during start-up
Conflicts:	--
Supporting Material:	--

4.1.2.3 Normal Operation

Note: Only NVRAM manager shall access non-volatile memory. All other software shall exclusively use NVRAM manager to access data in non-volatile memory. However, this is not a requirement on Memory Services but on the usage. Therefore, the former requirement [BSW176] (“Only access non-volatile memory via NVRAM manager”) has been removed.

4.1.2.3.1 [BSW027] Accessing of non volatile data

Initiator:	DC
Date:	12.01.2004
Short Description:	Accessing of non volatile data
Type:	New
Importance:	High
Description:	The NVRAM manager only provides an implicit way of accessing blocks in the NVRAM and in the shared memory (RAM). This means, the NVRAM manager copies one or more blocks from NV memory to RAM block(s) and other way round. Explicit read and write access of variables inside a RAM block has to be provided by application (access macro ...).
Rationale:	Basic functionality of NVRAM manager
Use Case:	Accessing of parameters
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.3.2 [BSW08014] RAM block allocation

Initiator:	Siemens VDO
Date:	28.07.2004
Short Description:	RAM block allocation
Type:	New
Importance:	High
Description:	RAM block allocation can be defined as not continuous in the global RAM area
Rationale:	--
Use Case:	Each RAM block can be allocated without address constraint in the global RAM area
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.3.3 [BSW013] Handling of concurrent accesses to NVRAM

Initiator:	Bosch
Date:	22.12.2003
Short Description:	The NVRAM manager shall be able to handle concurrent accesses to NVRAM memory.
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide a mechanism to handle multiple, concurrent read / write orders, e.g. queuing. Implementation requirement: if queuing is used, only the read / write orders are buffered, not the data to be written!
Rationale:	The NVRAM manager handles all accesses to NVRAM in the entire SW system. Therefore concurrent accesses are most likely to occur. The NVRAM manager must process these parallel accesses in a serial manner due to resource restrictions.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.3.4 [BSW016] Block-wise reading of data

Initiator:	Bosch
Date:	27.11.2003
Short Description:	Block-wise reading of data
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide functionality to read out data associated with an NVRAM block from the non-volatile memory. The NVRAM block is referenced by a unique identifier. The NVRAM data is copied into the corresponding RAM block. The availability of this service shall be configurable.
Rationale:	Basic functionality of NVRAM manager
Use Case:	Reading of application data from NV.
Dependencies:	--

Conflicts:	--
Supporting Material:	--

4.1.2.3.5 [BSW017] Block-wise writing of data

Initiator:	Bosch
Date:	27.11.2003
Short Description:	Block-wise writing of data
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide functionality to store data associated with an NVRAM block in the non-volatile memory. The NVRAM block is referenced by a unique identifier. The data is copied from RAM block into the corresponding NV block. The availability of this service shall be configurable.
Rationale:	Basic functionality of NVRAM manager
Use Case:	Non volatile storage of data before ECU power supply is switched off
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.3.6 [BSW08554] Retrying of read and write operations

Initiator:	R4.0 concept “Error Handling”
Date:	04.12.2008
Short Description:	Retrying of read and write operations
Type:	New
Importance:	High
Description:	The NVRAM manager shall retry read and write operations on NVRAM blocks if they have not succeeded up to a configurable number of times.
Rationale:	Recover from temporary error situation
Use Case:	An electromagnetic pulse has altered the data just read and checksum check has therefore failed. Reading again the data recovers from this error situation.
Dependencies:	[BSW08551] Configuration of maximum read and write retries
Conflicts:	--
Supporting Material:	--

4.1.2.3.7 [BSW08541] Guaranteed processing of accepted write requests

Initiator:	WP Memory Service (CRC)
Date:	09.08.2005
Short Description:	Guaranteed processing of accepted write requests
Type:	New
Importance:	High
Description:	The NVRAM manager shall guarantee that an accepted write request will be processed.
Rationale:	An application issuing a write request expects the data to be written.
Use Case:	1) Queued write jobs originating from the application shall not be influenced

	by the shutdown process or it's cancellation. 2) On a minor crash the ECU remains operational, the NVRAM Manager shall behave like without this crash. Therefore the request for writing corresponding crash data shall not abort any accepted write job.
Dependencies:	[BSW017] Block-wise writing of data [BSW08535] Save data blocks from RAM to NV memory [BSW08540] Aborting the shut down process [BSW08542] Job order prioritization
Conflicts:	--
Supporting Material:	--

4.1.2.3.8 [BSW018] Block-wise restoring of default data

Initiator:	WP Architecture
Date:	03.02.2004
Short Description:	Block-wise restoring of default data
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide functionality to restore an NVRAM block's associated data from ROM defaults. The availability of this service shall be configurable.
Rationale:	--
Use Case:	Radio factory settings
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.3.9 [BSW08548] Automatic initialization without ROM Block

Initiator:	Siemens VDO
Date:	27.09.2005
Short Description:	Automatic initialization without ROM Block
Type:	New
Importance:	High
Description:	If there is no ROM Block available at configuration time, the NVRAM Manager shall request default data from the application. This shall be configurable.
Rationale:	Flexibility
Use Case:	Calibration data cannot be provided by constant data.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.3.10 [BSW08547] Distinction between invalidated and inconsistent data

Initiator:	WP Memory Service (CRC)
Date:	27.09.2005
Short Description:	Distinction between invalidated and inconsistent data
Type:	New
Importance:	High
Description:	The NVRAM Manager shall be able to distinguish between explicitly

	invalidated and inconsistent data.
Rationale:	Explicitly invalidated (or even blank) data blocks shall not denote an error condition to be reported to the DEM.
Use Case:	If the NVRAM manager detects inconsistent data, e.g. due to an aborted write operation or CRC error, it shall report an error. On invalidated data, no error will be reported.
Dependencies:	[AUTOSAR_BASIC_SW]: [BSW14014] Detection of data inconsistencies [BSW14015] Reporting of data inconsistencies [BSW14016] Don't return inconsistent data to the caller
Conflicts:	--
Supporting Material:	--

4.1.2.3.11 [BSW08550] Marking Blocks modified/unmodified

Initiator:	WP Memory Service (CRC)
Date:	27.09.2005
Short Description:	Marking Blocks modified/unmodified
Type:	New
Importance:	High
Description:	The NVRAM Manager shall provide a service for marking permanent RAM data blocks as modified/unmodified. On shutdown only data that have been marked as modified shall be saved to NV. The availability of this service shall be configurable.
Rationale:	Write cycle reduction, speed-up shutdown.
Use Case:	Application knows best when to write back data to NV memory. Using this will reduce the number of required write cycles, and thus the need for walking blocks and NV memory consumption, as well as speed up shut-down operation due to possibly slow write because of significant overhead due to underlying HW (providing pre-erased memory).
Dependencies:	[BSW08546] Protection of RAM data blocks against data loss.
Conflicts:	--
Supporting Material:	--

4.1.2.3.12 [BSW08545] Validation of permanent RAM data blocks

Initiator:	BMW/MEDAG
Date:	20.09.2005
Short Description:	Validation of permanent RAM data blocks
Type:	New
Importance:	High
Description:	The NVRAM Manager shall provide a service for marking the permanent RAM data block of an NVRAM block valid. Additionally this service shall update integrity information, if configured. The availability of this service shall be configurable.
Rationale:	Save only valid data to NV. On start-up increase the possibility to use the most recent data in RAM, rather than reload old data from NV.
Use Case:	A failure on Read may result in invalid RAM content. This content shall not be saved to NV memory. In this case the application is responsible to handle this condition, by presenting data and invoking this service to mark them as valid.
Dependencies:	[BSW08546] Protection of RAM data blocks against data loss
Conflicts:	--

Supporting Material:	--

4.1.2.3.13 [BSW08011] Invalidation of NVRAM data blocks

Initiator:	BMW
Date:	09.07.2004
Short Description:	Invalidation of NVRAM data blocks
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide a service to invalidate a block of data in the non-volatile memory. The block is referenced by a unique identifier. The availability of this service shall be configurable.
Rationale:	Avoid loading of outdated data, especially position information.
Use Case:	The position of a sunroof (incremental motor position) has to be saved to an NVRAM block after each movement. Before every new movement of the sunroof this position has to be marked as invalid. Otherwise a reset during sunroof operation would cause an invalid sunroof position (because the old position from NVRAM is loaded). This could result in a damage of the sunroof mechanics (when crashing to end position slide open) or in a serious injury.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.3.14 [BSW08544] Block-wise erasing of NVRAM data

Initiator:	WP Memory Service (CRC)
Date:	09.08.2005
Short Description:	Block-wise erasing of NVRAM data
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide a service to erase the NV block(s) associated with an NVRAM block. The NVRAM block is referenced by a unique identifier. The availability of this service shall be configurable.
Rationale:	Explicit erase to support the use cases listed below.
Use Case:	Writing of crash data: no delay due to erase-before-write, therefore pre-erase NV block must be done at application-defined point (e.g. via diagnostic service). This cannot be done automatically.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.3.15 [BSW08007] Selection of datasets

Initiator:	BMW
Date:	09.07.2004
Short Description:	Selection of datasets
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide a service for the selection of valid dataset NV blocks. This service shall associate a dataset number (ROM or

	NVRAM) to the corresponding RAM block.
Rationale:	Make runtime selection of different datasets in ROM/NVRAM possible.
Use Case:	One ECU is used within several models/country variants of a car. Depending on the car variant, the corresponding data set in NVRAM or ROM is selected.
Dependencies:	[BSW08006] Block management type – dataset
Conflicts:	--
Supporting Material:	--

4.1.2.3.16 [BSW08542] Job order prioritization

Initiator:	WP Memory Service (CRC)
Date:	09.08.2005
Short Description:	Job order prioritization
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide a prioritization for job processing order. The highest priority job shall be processed first; jobs with the same priority level shall be executed in FIFO order. This prioritization shall be configurable. If disabled, jobs shall be executed in FIFO order.
Rationale:	Some data must be written faster than other. Therefore processing them shall take precedence. Immediate data shall preempt a running lower priority operation.
Use Case:	Writing of crash data Priority based data saving like NVRAM Manager of SiemensVDO
Dependencies:	[BSW08543] Static configuration of block priority [BSW08541] Guaranteed processing of accepted write requests
Conflicts:	--
Supporting Material:	--

4.1.2.3.17 [BSW020] Readout of current status of NVRAM manager operations

Initiator:	Bosch
Date:	27.11.2003
Short Description:	Readout of current status of NVRAM manager operations
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide functionality to read out the status of read/write operations.
Rationale:	The NVRAM manager shall support a variety of non-volatile memory devices. The access time of some devices is long in comparison to the real-time requirements of the ECU. Therefore the NVRAM manager shall provide asynchronous API with a functionality to determine the current status of operations.
Use Case:	Read/write of serial EEPROM
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.3.18 [BSW127] Write protect/unprotect function

Initiator:	BMW (derived from SiemensVDO)
Date:	03.02.2004
Short Description:	Write protect/unprotect function.
Type:	Changed (adapted to NVRAM concept V0.6)
Importance:	High
Description:	The NVRAM manager shall allow enabling/disabling a write protection for each NVRAM block individually. The data area of RAM block of write-protected blocks can be changed without restrictions. If an NV block is protected, further write access jobs for this block are rejected. NVRAM blocks with enabled write protection cannot be saved to NV memory. After reset, the write protection shall be set according to the initial write protection. The availability of this service shall be configurable
Rationale:	Some data blocks are not to be changed by the application, only by special EOL (end of line)/diagnostic services
Use Case:	Coding data, EOL (end of line) data.
Dependencies:	[BSW08009] Default write protection of blocks
Conflicts:	--
Supporting Material:	--

4.1.2.3.19 [BSW030] Consistency/integrity check of data

Initiator:	DC
Date:	12.01.2004
Short Description:	The NVRAM manager shall be able to check the consistency/integrity of the data saved.
Type:	New
Importance:	High
Description:	The NVRAM manager shall implement mechanisms for consistency/integrity checks of data saved in NVRAM during operations even in case of asynchronous reset or power loss. Bit-flipping shall be covered as well. Implementation hint: Checksums are one possibility; additionally write access bytes (valid/invalid) can be used.
Rationale:	Detection of errors.
Use Case:	Signatures for the detection of Bit-flipping.
Dependencies:	[BSW08001] Configuration of consistency check of data
Conflicts:	--
Supporting Material:	--

4.1.2.3.20 [BSW08555] Verification of block identifier

Initiator:	R4.0 concept “Error Handling”
Date:	04.12.2008
Short Description:	Verification of block identifier
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide mechanisms for static verification of the

	block identifier when reading an NVRAM block.
Rationale:	Protection against misaddressing of blocks
Use Case:	Hardware addressing error can be detected
Dependencies:	[BSW030] Consistency/integrity check of data
Conflicts:	--
Supporting Material:	--

4.1.2.3.21 [BSW08556] Write verification for data

Initiator:	R4.0 concept “Error Handling”
Date:	04.12.2008
Short Description:	Write verification for data
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide a mechanism for verification of the written block data by again reading and comparing it.
Rationale:	Protection against wrongly written data
Use Case:	Hardware write errors can be detected
Dependencies:	[BSW08553] Configuration of write verification for data
Conflicts:	--
Supporting Material:	--

4.1.2.3.22 [BSW034] Quasi-parallel write access

Initiator:	DC
Date:	12.01.2004
Short Description:	NVRAM manager write accesses shall be executed quasi parallel.
Type:	New
Importance:	High
Description:	Write accesses of the NVRAM manager to persistent memory shall be executed quasi-parallel (concurrent) to normal operation of the ECU. Data consistency must not be endangered.
Rationale:	The write access of some types of memories is by order of magnitude slower compared to processor register accesses.
Use Case:	EEPROM write accesses.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.3.23 [BSW08558] Removal of all requests associated with a NVRAM block

Initiator:	R4.0 concept “Implement Stop/Restart of SW-Cs” (part of concept Error Handling)
Date:	29.04.2009
Short Description:	Removal of all requests associated with a NVRAM block
Type:	New
Importance:	High
Description:	NVRAM manager shall provide a mechanism to remove all unprocessed requests associated with a NVRAM block

Rationale:	A request originating from a partition (for example, from a SW-C belonging to that partition) prior to the termination and subsequent restart of that partition should not be processed (and therefore no completion should be reported).
Use Case:	Dealing with termination and restart of partitions.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.2.3.24 [BSW08559] Enable shared use of blocks

Initiator:	R4.0 “memory related concepts”
Date:	29.04.2009
Short Description:	Enable shared use of blocks
Type:	New
Importance:	High
Description:	NVRAM manager shall provide means to make shared access to a block possible.
Rationale:	Sharing data reduces memory footprint. NVRAM manager shall not provide an all-embracing solution (too complex, too costly with respect to resources) but provide functionality to make explicit client-sided synchronization of access possible.
Use Case:	A block is shared by different SW-Cs.
Dependencies:	[BSW08560] Configuration of shared use of blocks
Conflicts:	--
Supporting Material:	--

4.1.2.4 Shutdown Operation

4.1.2.4.1 [BSW08535] Save data blocks from RAM to NV memory

Initiator:	WP Memory Service (CRC)
Date:	07.07.2005
Short Description:	Save data blocks from RAM to NV memory
Type:	New
Importance:	High
Description:	The NVRAM manager shall provide a function, which triggers update of integrity information (recalculate a checksum, e.g. CRC) and saving of RAM data blocks to NV memory. This function can only affect permanent RAM data blocks because temporary RAM data blocks cannot be written back automatically, e.g. triggered by the ECU state manager. Information: • This function has to be called by the ECU state manager once before shutdown • This function can be called by a diagnostic service (‘Save NVRAM’)
Rationale:	Don’t let the application do the entire job.
Use Case:	ECU Shut-down: save NV data of application
Dependencies:	--
Conflicts:	--
Supporting Material:	[BSW08534] Classes of RAM data blocks

4.1.2.4.2 [BSW08540] Aborting the shut down process

Initiator:	WP Memory Service (CRC)
Date:	09.08.2005
Short Description:	Aborting the shut down process
Type:	Changed
Importance:	High
Description:	If during an ECU shutdown an ECU wake-up condition is detected, the NVRAM manager shall provide a function to abort those write jobs originating from the shutdown process. Those write requests in the queue originating from the application shall not be affected by this cancellation routine. This cancellation shall not be destructive, i.e. the NVRAM Block currently being written shall be completed.
Rationale:	Allow fast reaction to wake-up condition during ECU shut down procedure.
Use Case:	ECU Shutdown: the NVRAM manager has started saving all RAM blocks to NVRAM. During this job processing, the ECU wake-up condition occurs. The ECU has to be operable within the given time limits (e.g. 100ms). It is not acceptable that the NVRAM Manager finishes all write jobs which may take e.g. 800 ms.
Dependencies:	--
Conflicts:	Currently not used by the EcuStateManager
Supporting Material:	--

4.1.2.5 Fault Operation

4.1.2.5.1 [BSW038] Treatable errors shall not affect other software components

Initiator:	DC
Date:	12.01.2004
Short Description:	Treatable errors shall not affect other software components
Type:	Changed after review within SV (02.03.2004)
Importance:	High
Description:	The NVRAM manager shall not report treatable (healable or recoverable) errors to other software components.
Rationale:	Separation of concerns, avoidance of unnecessary degradation of functionality
Use Case:	Data is saved redundantly (within two identical blocks). If a data corruption is detected in one block, the contents of the redundant block are used. The application is not affected.
Dependencies:	[BSW08529] Block management type – redundant [BSW129] Automatic data repair
Conflicts:	--
Supporting Material:	--

4.1.2.5.2 [BSW129] Automatic data repair

Initiator:	BMW
Date:	03.02.2004
Short Description:	Automatic data repair
Type:	New
Importance:	High
Description:	The NVRAM manager shall repair data in blocks of management type

	'NVRAM redundant' if a data corruption is detected and valid data can be derived. The number of repair cycles shall be limited in order to avoid infinite loops.
Rationale:	Robustness
Use Case:	Data corruption (bit flipping) in a redundant NVRAM block
Dependencies:	[BSW08529] Block management type - redundant
Conflicts:	--
Supporting Material:	--

4.1.2.5.3 [BSW08010] Loading of ROM default data

Initiator:	BMW
Date:	09.07.2004
Short Description:	Loading of ROM default data.
Type:	New
Importance:	High
Description:	If the NVRAM manager cannot read data from NV into RAM, it shall copy the ROM default data to the data area of the corresponding RAM block.
Rationale:	Robustness
Use Case:	The calibration data set is damaged. A default calibration data set is loaded.
Dependencies:	See all Block management types.
Conflicts:	--
Supporting Material:	--

4.1.2.5.4 [BSW08015] NV Block security of ECU reprogramming

Initiator:	SiemensVDO
Date:	26.08.2004
Short Description:	NV Block security of ECU reprogramming
Type:	New
Importance:	High
Description:	After ECU reprogramming some NVRAM data has to be kept secure. This means that some of the NV Blocks in the NVRAM should never be erased nor be replaced with the default ROM data after first initialization.
Rationale:	Immobilizer code or vehicle identification number
Use Case:	--
Dependencies:	--
Conflicts:	This is a requirement on configuration / bootloader.
Supporting Material:	--

4.1.3 Non-Functional Requirements (Qualities)

4.1.3.1 Hardware independence

4.1.3.1.1 [BSW011] (Memory) hardware independence

Initiator:	Bosch
Date:	22.12.2003
Short Description:	The NVRAM manager shall be independent from its underlying memory hardware.

Type:	New
Importance:	High
Description:	Existing (standardized) interfaces shall be used by NVRAM-Manager to access the underlying memory hardware. The interfaces shall abstract the memory hardware.
Rationale:	Portability, reusability, hardware cost reduction by flexible usage of memory devices
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.1.3.2 Usability

4.1.3.2.1 [BSW130] Provide information about used memory resources

Initiator:	DC
Date:	12.01.2004
Short Description:	Provide information about used memory resources.
Type:	New
Importance:	High
Description:	The NVRAM manager configuration shall provide information how many resources of RAM, ROM and NVRAM are used. The format of this information shall be commonly used (e.g. MAP file format)
Rationale:	Get knowledge how much resources are still available
Use Case:	Mapping of AUTOSAR SW components to one ECU
Dependencies:	
Conflicts:	--
Supporting Material:	--

4.1.3.3 Debugging and diagnosis

4.1.3.3.1 [BSW08557] Support debugging and diagnosis

Initiator:	General: Implement Concept “Debugging Concept”
Date:	06-06-2008
Short Description:	Support debugging and diagnosis
Type:	New
Importance:	High
Description:	The NVRAM manager shall support AUTOSAR debugging of its variables and reading of NV data for diagnostic purposes. This qualitative property may include special functionality addressing only debugging and diagnosis.
Rationale:	Make the development process faster and more efficient. Find errors easier.
Use Case:	A bug in the behavior is observed and the error has to be located in the source code.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

5 References

5.1 Deliverables of AUTOSAR

[AUTOSAR_SW_ARCH] Layered Software Architecture
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf

[AUTOSAR_BASIC_SW] List of Basic Software Modules
AUTOSAR_TR_BSWModuleList.pdf

[AUTOSAR_SRS_MEMHW] Requirements on Memory Hardware Abstraction Layer
AUTOSAR_SRS_MemoryHWAbstractionLayer.pdf