Document Title	Requirements on ECU Configuration
Document Owner	AUTOSAR GbR
Document Responsibility	AUTOSAR GbR
Document Version	1.1.1
Document Status	Final
Part of Release	2.1
Revision	0014

Document Change History			
Date	**Version**	**Changed by**	**Change Description**
24.01.2007	1.1.1	AUTOSAR Administration	• "Advice for users" revised • "Release Notes" added • "Revision Information" added
05.12.2006	1.1.0	AUTOSAR Administration	• Added requirements ECUC0076 • Legal disclaimer revised
14.12.2005	1.0.0	AUTOSAR Administration	1.0.0 Initial release

Page left intentionally blank

Disclaimer

Any use of these specifications requires membership within the AUTOSAR Development Partnership or an agreement with the AUTOSAR Development Partnership. The AUTOSAR Development Partnership will not be liable for any use of these specifications.

Following the completion of the development of the AUTOSAR specifications commercial exploitation licenses will be made available to end users by way of written License Agreement only.

Advice to users of AUTOSAR Specification Documents:

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 Contents

1 Scope of Document

ECU Configuration is one activity performed during the development of an AUTOSAR ECU.
The input to the ECU Configuration is one portion of the System Configuration Description which is called ECU Extract of System Configuration. The activity of ECU Configuration is to provide configuration information for all the software within one ECU. This spans from AUTOSAR SW-Components over the RTE Configuration to the multitude of Basic Software Modules.
The Output of the ECU Configuration is the ECU Configuration Description which is used to actually generate and build the ECU Executable.

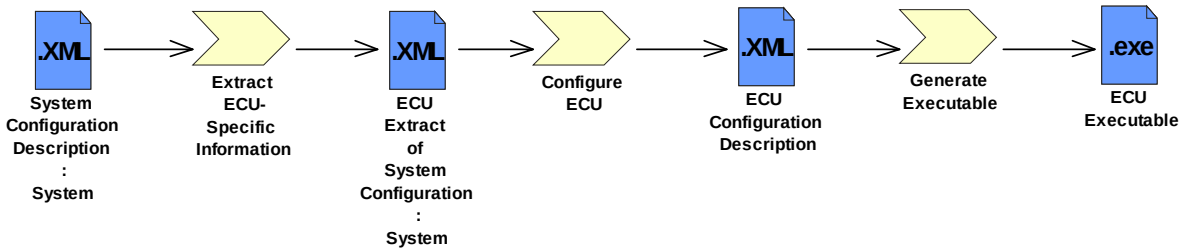

Figure 1-1: Overview AUTOSAR Methodology [9]

The main focus of this requirements document is the format of the ECU Configuration Description.

2 Conventions to be used

- In requirements, the following specific semantics shall be used (based on 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:

  - SHALL: This word means that the definition is an absolute requirement of the specification.
  - SHALL NOT: This phrase means that the definition is an absolute prohibition of the specification.
  - MUST: This word means that the definition is an absolute requirement of the specification due to legal issues.
  - MUST NOT: This phrase means that the definition is an absolute prohibition of the specification due to legal constraints.
  - 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.)

3 Related Documentation

3.1 Input Documents

The following input documents have been used in the development of these requirements:

- General Requirements on Basic Software Modules [2]
- AUTOSAR RTE Software Requirement Specification [3]
- AUTOSAR Methodology [9]
- AUTOSAR Glossary [1]
- Technical Overview [10]
- AUTOSAR Template UML Profile and Modeling Guide [5]
- AUTOSAR Model Persistence Rules for XML [6]

3.2 Specification Documents

The requirements collected in this document will be satisfied by two specification documents:

- ECU Configuration Specification [11]
  This document provides the outline of the configuration methodology and the development guidelines for the ECU Configuration Parameters.
- ECU Configuration Parameters [12]
  This document contains the specification of the AUTOSAR standardized configuration parameters of BSW, RTE, SW-Components and ECU integration.

3.3 Abbreviations

Abbreviation	***Meaning***
BSW	Basic Software
BSWMD	Basic Software Module Description
ECUC	ECU Configuration
SW-C	Software Component

4 Requirements on ECU Configuration

4.1 Functional Requirements

4.1.1 Requirements on the Template

The requirements in this section all concern how the ECU Configuration Template shall be defined.

4.1.1.1 [ECUC0032] ECU Configuration Description shall be the root for the whole configuration information of an ECU

Initiator:	WP4.1.1.2
Date:	2004-11-22
Short Description:	ECU Configuration Description shall be the root for the whole configuration information of an ECU
Type:	new
Importance:	high
Description:	The ECU Configuration Template shall become the backbone for the integration of software on a particular ECU. Any necessary configuration information shall be aggregated or referenced in this template.
Rationale:	Integration of software means basically the resolution of interdependencies among particular parts of the software. The resolution process can only be performed properly if all relevant information is accessible. Note that this requirement does not imply that all relevant information is copied into the template. References to elements in other templates may be included to avoid redundant elements in the model.
Use Case:	The ECU Configuration Template will include references to SW Components described using the SW component part of the meta model. It will also allow to specify the sequencing of runnables within task, something which is not specified by any other template.
Dependencies:	Consequence: ECU Configuration tools (editors and generators) need to be able to read parts of other templates as well (like System Template, SW Component Template, ECU Resource Template).
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.1.2 [ECUC0072] Support for referencing from dependent containers

Initiator:	WP4.1.1.2
Date:	2005-09-06
Short Description:	Support for referencing from dependent containers
Type:	new
Importance:	high
Description:	It shall be possible to define references from parameter container to other parameter definitions in the ECUC Parameter definition and to elements in other AUTOSAR templates for standardized AUTOSAR Configuration parameters.
Rationale:	If there is a reference between two containers then it is assumed that there is a dependency from the referencing container to the referenced container.
Use Case:	Examples for such dependencies are:

	a) PortPin references the driver that uses this Pin b) The BitPosition of a network signal in COM is defined by the SignalPosition in the SystemTemplate.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.1.3 [ECUC0050] Specify ECU Configuration Parameter Definition

Initiator:	WP4.1.1.2
Date:	2005-01-26
Short Description:	Specify ECU Configuration Parameter Definition
Type:	new
Importance:	high
Description:	Definition of ECU Configuration Parameters and their constraints such as configuration class, value range, multiplicities have to be captured.
Rationale:	Constraints on configuration parameters are important to be identified and verified. This also includes validity of the actual parameters themselves, like ranges and predefined values.
Use Case:	a) Clock frequency needs to be > 0; b) 0 <= Data_Length <= 8
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	BSW167 [2]

4.1.1.4 [ECUC0055] Support standardization of mandatory and optional configuration parameters

Initiator:	WP4.1.1.2
Date:	2005-03-22
Short Description:	Support standardization of mandatory and optional configuration parameters
Type:	new
Importance:	high
Description:	In the standardization of ECU Configuration Parameter Definitions it shall be possible to define if a parameter is mandatory or optional. The definition of optional and mandatory shall be as follows: A mandatory parameter must be implemented by all implementations of that module. It must always be present in a completed ECU configuration description. An optional parameter may be omitted by an implementation.
Rationale:	Parameters which are not applicable for every implementation can anyhow be standardized if it is clarified that not every implementation needs to support that parameter. Note that for a concrete implementation a parameter is either supported and must then be present in the ECU configuration or not supported and must then not be present. Thus for a concrete implementation no optionality exists anymore, only in the standardized version of the parameter definition. Note that an implementation may still choose to fix the value for a mandatory parameter (e.g. an object code delivery fixes the values of pre-compile parameters). Anyhow, the value selected must then be stated in the ECU Configuration Description [11].
Use Case:	The parameter ACTIVATE_PULLUP in the PORT Module is only needed if the hardware supports it. Thus an implementation of the PORT module may

	omit the parameter if the target HW does not support pull ups.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.1.5 [ECUC0070] Support mandatory and optional containers

Initiator:	WP4.1.1.2
Date:	2005-03-22
Short Description:	Support mandatory and optional containers
Type:	new
Importance:	high
Description:	It shall be possible to group configuration parameters into a hierarchy of containers. It shall be possible to define if a container is mandatory or optional. The definition of optional and mandatory shall be as follows: A mandatory container defined in a module must be implemented by all implementations of that module. It must always be present in a completed ECU Configuration description. All parameters and sub-containers of that container must also be present unless they are optional. An optional container may be omitted by an implementation. If an optional container is omitted, all parameters and sub-containers defined within that container are omitted as well, independent of whether they are defined optional or mandatory.
Rationale:	Containers which are not applicable for every implementation can anyhow be standardized if it is clarified that not every implementation needs to support that container.
Use Case:	An ADC module may define a set of parameters for each ADC channel for the microprocessor. If there is no ADC channel present on a specific microcontroller, none of these parameters is useful. If a channel is defined, then all mandatory parameters for that channel should be filled in. So it is better to define an optional container that includes mandatory parameters than defining several optional parameters (which may then be omitted independently of each other).
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.1.6 [ECUC0043] Duplication free description

Initiator:	WP4.1.1.2
Date:	2004-12-17
Short Description:	Duplication free description
Type:	new
Importance:	medium
Description:	ECU Configuration description shall contain each configuration information only once, even if it is required to configure several modules.
Rationale:	This helps to avoid inconsistency in the description and keeps the description compact. Note that it is still allowed to include derived information: If configuration parameter C may be in principle calculated from configuration items A and B, it is still possible to include C in the template.
Use Case:	Definition of tasks used in the ECU may be relevant for OS configuration and RTE configuration. But it should only be defined once in the ECU

	configuration template.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.1.7 [ECUC0002] Support of vendor-specific ECU Configuration Parameters

Initiator:	WP4.1.1.2
Date:	2004-11-09
Short Description:	Support of vendor-specific ECU Configuration Parameters
Type:	new
Importance:	medium
Description:	ECU Configuration shall provide means for vendor specific information in addition to the standard configuration parameters, which are defined in the SWS of the BSW module.
Rationale:	It has to be assured that really all ECU information can be stored in ECU Configuration description. So specific items can be passed to the generation tools.
Use Case:	A special attribute and some tool settings for NVRAM Manager have to be defined in the ECU Configuration Parameter Definition and the actual values stored in the ECU Configuration Description.
Dependencies:	ECUC0018 is requiring extension mechanism to support the evolution of the standard.
Conflicts:	This is diluting the standard however there is a need for such extensions.
Supporting Material:	None identified.

4.1.1.8 [ECUC0046] Support definition of configuration class

Initiator:	WP4.1.1.2
Date:	2005-01-26
Short Description:	Support definition of configuration class
Type:	new
Importance:	high
Description:	The ECU Configuration parameter definition shall support the definition of the configuration class of configuration parameters.
Rationale:	The BSW SWS allows several configuration classes: pre-compile time, link time, post-build time, post-build time selectable, post-build time loadable. The standardization of configuration parameters does not necessarily fix the configuration class of parameters. Instead it may define different configuration class implementation variants of a module. An actual BSW module implementation does fix the configuration class for each parameter. This information has to be stored in the ECU Configuration Parameter definition.
Use Case:	If the parameter XXX_DEV_ERROR_DETECT is specified to be only pre-compile configurable then the parameter value has to be identical in all configuration sets of this BSW module instance.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.1.9 [ECUC0012] One description mechanism for different configuration classes

Initiator:	WP4.1.1.2
Date:	2004-11-15
Short Description:	One description mechanism for different configuration classes
Type:	new
Importance:	high
Description:	There shall be only one configuration description mechanism for all different kinds of configuration classes. So regardless if "pre-compile-time" or "link-time" or “post-build-time” configuration is described the description format shall be the same.
Rationale:	The description of the configuration shall be independent from the BSW implementations. Additional information might be needed if the parameter class changes in the implementation, e.g. memory location for post-build-time parameters.
Use Case:	If one BSW module is configured to be "pre-compile-time" configuration - and this module is exchanged against a "link-time" configuration one - the ECU Configuration description shall not change due to this exchange.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.1.10 [ECUC0049] ECU Configuration description shall be tool process-able

Initiator:	WP4.1.1.2
Date:	2005-01-26
Short Description:	ECU Configuration description shall be tool process-able
Type:	new
Importance:	high
Description:	The ECU Configuration Descriptions shall be read- and writable by ECU Configuration tools and generators.
Rationale:	The configuration of an ECU shall be supported by tools.
Use Case:	ECU Configuration will have to have tool support.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	BSW159 [2]

4.1.1.11 [ECUC0065] Development according to the AUTOSAR Template UML Profile and Modeling Guide

Initiator:	WP4.1.1.2
Date:	2005-06-27
Short Description:	Development according to the AUTOSAR Template UML Profile and Modeling Guide
Type:	new
Importance:	high
Description:	The ECU Configuration Description shall be developed according to the AUTOSAR Template UML Profile and Modeling Guide.
Rationale:	The experience and tools already available for the AUTOSAR Modeling shall

	be reused.
Use Case:	The template for the ECU Configuration is similar to other templates already done within AUTOSAR.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	AUTOSAR Template UML Profile and Modeling Guide [5]

4.1.1.12 [ECUC0066] Transformation of ECUC model according to the AUTOSAR Model Persistence Rules for XML

Initiator:	WP4.1.1.2
Date:	2005-06-27
Short Description:	Transformation of ECUC model according to the AUTOSAR Model Persistence Rules for XML
Type:	new
Importance:	high
Description:	The ECU Configuration Template schema shall be derived using the model transformation described in the AUTOSAR Model Persistence Rules for XML.
Rationale:	The experience and tools already available for the AUTOSAR Modeling from other templates shall be reused.
Use Case:	The template for the ECU Configuration is similar to other templates already done within AUTOSAR.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	AUTOSAR Model Persistence Rules for XML [6].

4.1.1.13 [ECUC0018] Extension handling

Initiator:	WP4.1.1.2
Date:	2004-11-18
Short Description:	Extension handling
Type:	new
Importance:	high
Description:	The ECU Configuration must allow for later extensions to support the evolution of the standard.
Rationale:	It may be possible, that extensions/additional aspects must be handled in the ECU Configuration and ECU resources which are currently not part of the standard. But at some point in time these extensions shall become part of the standards next version.
Use Case:	If e.g. new type of ECU resources become available, it must be possible to easily incorporate this into the AUTOSAR architecture without changing the standard at once, since changes to the standard may take some time.
Dependencies:	ECUC0002 is specifying vendor-specific extensions that will never be part of the standard.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.1.14 [ECUC0074] Support Sequential ECU Configuration

Initiator:	WP4.1.1.2
Date:	2005-09-20
Short Description:	Support Sequential ECU Configuration
Type:	new
Importance:	high
Description:	It shall be possible to edit the different parts of the ECU Configuration Description separately.
Rationale:	The ECU Configuration Description contains the configuration of several modules whose configurations influence each other. So the dependencies have to be resolved sequentially and iteratively.
Use Case:	The RTE configuration editor can allocate an OS Task but still the OS configuration editor is able to change the properties of this OS Task.
Dependencies:	ECUC0025 implies that the tools can be used iteratively as well.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.1.15 [ECUC0025] Compatible with iterative design

Initiator:	WP4.1.1.2
Date:	2004-11-18
Short Description:	Compatible with iterative design
Type:	new
Importance:	high
Description:	Support iterative design as mandated by AUTOSAR methodology.
Rationale:	It is rare that development of any product is finished without any design changes. A design change invariably requires iteration of some portion of the design stages. AUTOSAR tools will be used iteratively and this includes ECU configuration.
Use Case:	Product design change after configuration of an ECU which requires a new ECU configuration to be developed.
Dependencies:	This is closely related with the usage of the System Constraint/Configuration Description. ECUC0074
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.1.16 [ECUC0075] Support exactly one to be configured micro-controller per ECU

Initiator:	WP4.1.1.2
Date:	20005-11-13
Short Description:	Support exactly one to be configured micro-controller per ECU
Type:	new
Importance:	high
Description:	The ECUC shall support the configuration of exactly one micro-controller per ECU.
Rationale:	This restriction is needed because the input to the ECUC is only able to cover one micro-controller per ECU as well. It is not possible to introduce support of multiple micro-controllers only in the ECUC.
Use Case:	AUTOSAR does currently not support multiple µCs per ECU. If each µC runs an own RTE they could be seen as two separate ECUs (if they do not communicate over shared memory). So the whole AUTOSAR would have to

	consider this workaround.
Dependencies:	
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.2 Requirements from the clients of the ECUC

4.1.2.1 [ECUC0047] Pre-compile time configuration of BSW

Initiator:	WP4.1.1.2
Date:	2005-01-26
Short Description:	Pre-compile time configuration of BSW
Type:	new
Importance:	high
Description:	The configuration of parameters of BSW at pre-compile time shall be possible for configuration parameters that are defined to be pre-compile time configurable.
Rationale:	Some AUTOSAR BSW module's configuration parameters are defined to be pre-compile time configurable for efficiency reasons. The ECU Configuration needs to support those parameters.
Use Case:	Support of pre-compile time configuration parameters.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	BSW00345 [2], AUTOSAR Glossary [1]

4.1.2.2 [ECUC0048] Link time configuration of BSW

Initiator:	WP4.1.1.2
Date:	2004-11-22
Short Description:	Link time configuration of BSW
Type:	new
Importance:	high
Description:	The configuration of parameters of BSW during link time shall be possible for configuration parameters that are defined to be link time configurable.
Rationale:	When a BSW module is delivered as object code it can only be configured at link time or post-build time.
Use Case:	Support of link time configuration parameters.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	BSW00344 [2], AUTOSAR Glossary [1]

4.1.2.3 [ECUC0008] Post-build time configuration of BSW

Initiator:	WP4.1.1.2
Date:	2005-01-25
Short Description:	Post-build time configuration of BSW
Type:	new
Importance:	high
Description:	It shall be possible to reconfigure configuration parameters post-build time that are defined to be post-build configurable.
Rationale:	This will allow post-build time configuration of the BSW Components in an ECU to adapt to changes in the surrounding system. The effect on the overall system must be taken into account when defining a parameter as post-build time configurable.
Use Case:	The FMC development and after sales methodology heavily depends on the possibility to reconfigure the CAN and LIN communication post-build time.

	The main configuration items are mapping of signals into frames, frame priority and frame timing.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	AUTOSAR Glossary [1]

4.1.2.4 [ECUC0039] Support configuration of BSW

Initiator:	WP4.1.1.2
Date:	2004-12-02
Short Description:	Support configuration of BSW
Type:	new
Importance:	high
Description:	ECU Configuration SHALL support all configuration parameters defined in the BSW SWS documents. The list of all supported modules can be found in the Layered Software Architecture document [13].
Rationale:	The BSW has to be configured out of the ECU Configuration description.
Use Case:	a) BSW SWS define a list of configuration parameters. These parameters need to be reflected in the ECU Configuration template. b) The OS SWS will identify configuration parameters from existing formats (such as OIL) and place them in the SWS. In that case only the content of OIL is used in the ECUC, not the actual format itself.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	Related released specifications of BSW modules [13].

4.1.2.5 [ECUC0015] Configuration of multiple instances of BSW modules

Initiator:	WP4.1.1.2
Date:	2004-11-15
Short Description:	Configuration of multiple instances of BSW modules
Type:	new
Importance:	high
Description:	It shall be possible to describe the configuration of multiple instances of one BSW module type on one ECU – if applicable.
Rationale:	Some BSW module types can be present multiple times within one ECU, using different implementations, e.g. FLASH, EEP and watchdog drivers. For some BSW modules it is not possible to have multiple instances like the OS, NVRAM-Manager, etc.
Use Case:	When two different external EEPROM chips from different vendors are present on an ECU, there is a need for two different BSW drivers to cope with these.
Dependencies:	
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.2.6 [ECUC0021] Select AUTOSAR SW Component and BSW Module implementation

Initiator:	WP4.1.1.2

Date:	2004-11-18
Short Description:	Select AUTOSAR SW Component and BSW Module implementation
Type:	new
Importance:	high
Description:	Allow selection of a specific SW Module implementation if more than one is available.
Rationale:	Multiple implementations (code) of modules may be available for incorporation into a system configuration. A specific selection may be required.
Use Case:	a) Multiple CAN drivers from different vendors. b) Multiple driver versions from a single vendor. c) Different implementations of an AUTOSAR SW Component.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.2.7 [ECUC0068] Mapping of software sections to memory

Initiator:	WP4.1.1.2
Date:	2005-10-25
Short Description:	Mapping of software sections to memory
Type:	new
Importance:	high
Description:	Parts of software (code, data) shall be map-able to specific memory areas.
Rationale:	Each software consists of several sections which are defined when developing/compiling the software. Those sections need to be placed in different memory areas on the ECU.
Use Case:	a) If data shall be post-build configurable it needs to be placed in non-volatile memory. The location the data is placed is needed for tools to be able to change the data later on. b) Certain SW variables have to be placed in a NOINIT memory area which will not be initialized during ECU start-up.
Dependencies:	Software needs to publish the memory sections that have been used while developing/compiling the software. This needs to be part of the BSWMD [4] and the SWC-T [8].
Conflicts:	None identified.
Supporting Material:	Specification of Memory Mapping [7]

4.1.2.8 [ECUC0040] Support configuration of RTE

Initiator:	WP4.1.1.2
Date:	2004-12-02
Short Description:	Support configuration of RTE
Type:	new
Importance:	high
Description:	The generation and configuration of the RTE shall be supported.
Rationale:	The RTE has to be generated and configured on the basis of the ECUC's information.
Use Case:	Analyze all configuration requirements of RTE SWS document.
Dependencies:	None identified.
Conflicts:	None identified.

Supporting Material:	None identified.

4.1.2.9 [ECUC0076] Support the configuration of which AUTOSAR Services are available on a specific ECU

Initiator:	WP4.1.1.2
Date:	2006-08-31
Short Description:	Support the configuration of which AUTOSAR Services are available on a specific ECU
Type:	New
Importance:	High
Description:	AUTOSAR defines several services which may be integrated into an ECU. It shall be possible to define which AUTOSAR Services are present on a specific ECU.
Rationale:	There may be ECUs which do not need all AUTOSAR Services, so a subset may be defined.
Use Case:	If the ECU does not use any NvRam there is no need for a NvRam Manager on this ECU.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.3 Requirements from Software Components

4.1.3.1 [ECUC0041] Support AUTOSAR SW Component Integration

Initiator:	WP4.1.1.2
Date:	2005-01-24
Short Description:	Support AUTOSAR SW-Component integration
Type:	new
Importance:	high
Description:	The integration of AUTOSAR SW Components shall be supported.
Rationale:	AUTOSAR SW Components shall be instantiated on an ECU. The relevant information has to be provided to enable their integration and execution on that ECU.
Use Case:	Usage of AUTOSAR SW Components.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.3.2 [ECUC0073] Support Service Configuration of AUTOSAR SW Components

Initiator:	WP4.1.1.2
Date:	2005-09-07
Short Description:	Support Service Configuration of AUTOSAR SW Components
Type:	new
Importance:	high
Description:	ECU Configuration shall support configuration of the interface between the AUTOSAR SW Components and the AUTOSAR Services.

Rationale:	During the configuration of an ECU, specific values for AUTOSAR SW Components service access need to be configured.
Use Case:	The SW Component requires NVRAM Blocks with symbolic IDs. The NVRAM Manager configuration is assigning specific ID values and needs to configure those values in the SW Component.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.3.3 [ECUC0016] Execution order of runnable entities within one OS task

Initiator:	WP4.1.1.2
Date:	2004-11-15
Short Description:	Execution order of runnable entities within one OS task
Type:	new
Importance:	high
Description:	It shall be possible to describe the execution order of runnable entities with OS tasks across SW-Component borders.
Rationale:	The SW-C template can only describe constraints on the execution order of runnable entities within one SW-Component. But it has to be possible to describe execution order between SW Component's runnable entities mapped to a specific OS task to establish control flow algorithms where the control flow is passed from one SW-C to the next in a defined order.
Use Case:	Three periodically executed runnable entities are mapped to one OS task. It has to be defined in which order those runnable entities are to be executed when the OS task is activated.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.4 Requirements on the Configuration Parameter Definition

4.1.4.1 [ECUC0071] Support for Generic Configuration Editor

Initiator:	WP4.1.1.2
Date:	2005-08-04
Short Description:	Support for Generic Configuration Editor
Type:	new
Importance:	high
Description:	ECU Parameter definition shall be defined in a tool processable format. The parameter definitions shall contain enough information to support generic configuration editors.
Rationale:	A Generic Configuration Editor may read in parameter definitions for standardized and HW-vendor specific parameters. It may then display all defined parameters and allows to fill a ECU Configuration Description with all defined parameters.
Use Case:	Reduce the number of configuration editors required to configure an ECU. Simple BSW modules should be configurable with generic configuration editors. This also reduces the effort of BSW vendors who do not need to write specific configuration editors for simple BSW modules.
Dependencies:	None identified.
Conflicts:	None identified.

Supporting Material:	None identified.

4.1.5 To-Dos for WP4.1.1.2

These requirements provide information what the tasks for the WP4.1.1.2 are going to be. They are only informative in the sense that they are used to communicate the actions taken by WP4.1.1.2 and will be removed at a later stage.

4.1.5.1 [ECUC0029] Identify mechanisms not criteria

Initiator:	WP4.1.1.2
Date:	2004-11-18
Short Description:	Identify mechanisms not criteria
Type:	new
Importance:	high
Description:	The WP products will only identify the methods and mechanisms that may be used to configure ECUs but will not identify any engineering trade-offs (e.g. between performance and size) that must be considered by configuration engineers.
Rationale:	The mechanisms are application independent so all potential impacts cannot be assessed. Application specific trade-offs must be considered by engineers utilizing the mechanism for an application build.
Use Case:	The configuration parameter definition will only define valid ranges, but not provide algorithms for finding the best setting.
Dependencies:	This should be part of the WP scope definition chapter of some deliverable.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.5.2 [ECUC0030] Clarify configuration terminology

Initiator:	WP4.1.1.2
Date:	2004-11-18
Short Description:	Clarify configuration terminology
Type:	new
Importance:	high
Description:	Ensure that all configuration terminology introduced into AUTOSAR is adequately explained, by glossary entries or other documentation.
Rationale:	Terms such as "pre-compile configuration", "link time configuration" and "post-build time configuration" mean different things depending on the point in development that they are applied. The work package must ensure that any terminology used is adequately clarified for each context in which it may appear.
Use Case:	Powertrain Applications Calibration Engineers interpret the meaning of “post-build time configuration” differently to Powertrain Software Engineers.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.6 External Requirements

These are requirements that the ECU Configuration Description is not able to satisfy by its own. These requirements are placed on other AUTOSAR work products.

4.1.6.1 [ECUC0057] Memory needs of BSW modules

Initiator:	WP4.1.1.2
Date:	2005-05-03
Short Description:	Memory needs of BSW modules
Type:	new
Importance:	high
Description:	BSW modules shall provide information on the memory needs for the module. This is also true when the memory need is dependent on the actual configuration.
Rationale:	The amount of memory required typically depends on the configuration of the module and is finally determined at latest after configuration.
Use Case:	Post build data for COM stack varies in size, depending on the number of frames sent and received, code size (ROM) may vary for generated code. Data structures in RAM and EEPROM may vary as well.
Dependencies:	This is a requirement on the BSW Module Description, even though the actual value might depend on the configuration.
Conflicts:	None identified.
Supporting Material:	None identified.

4.1.6.2 [ECUC0036] Identify un-initialized resources

Initiator:	WP4.1.1.2
Date:	2004-11-25
Short Description:	Identify un-initialized resources.
Type:	new
Importance:	high
Description:	ECU Configuration shall identify unused resources present in an ECU.
Rationale:	Resources not used by any BSW module (standard AUTOSAR or Complex Device Driver) are not initialized by ECU software. This may affect the robustness of an ECU and cause unexpected interrupts or result in spurious EMC issues. (Note: This is a requirement on the tool – not on the template)
Use Case:	When unused (thus un-initialized) resources are identified a user may add appropriate drivers to the ECU which at least initialize these resources (e.g. explicitly disable unused interrupts or I/O ports).
Dependencies:	BSWMD0023 [4]
Conflicts:	None identified.
Supporting Material:	This would have to be checked by the tool-chain and/or process.

4.1.6.3 [ECUC0056] Identify conflicting usage of µC registers

Initiator:	WP4.1.1.2
Date:	2005-03-22
Short Description:	Identify conflicting usage of µC registers
Type:	new
Importance:	medium
Description:	ECU Configuration tool shall identify µC registers which are configured

	and/or accessed by more than one BSW module in an inconsistent way.
Rationale:	BSW modules (AUTOSAR drivers, complex drivers etc) may directly access µC registers. When BSW modules, possibly even from different vendors, are integrated into a single ECU then these modules may try to configure the same µC registers differently. Such conflicts must be identified. (Note: This is a requirement on the tool – not on the ECUC template)
Use Case:	4 LSB of a register are used by the GPT Driver, the 4 MSB of the same register are used by the ICU. If both modules write the entire register (8 bit) then they overwrite each other's configuration setting. Once a conflict is identified, different BSW module implementations or different BSW configuration may be used to resolve the problem.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	BSWMD0009 [4]

4.1.6.4 [ECUC0062] Configuration option to initialize unused memory

Initiator:	WP4.1.1.2
Date:	2005-05-30
Short Description:	Configuration option to initialize unused memory
Type:	new
Importance:	high
Description:	The AUTOSAR build environment shall provide a configuration option to initialize unused memory to a default value.
Rationale:	Faulty ECU software may access memory locations which are normally not used. Faulty ECU software may even attempt to execute code in memory locations which are not used. Unused RAM contains random data which can cause non-deterministic behavior in case of an erroneous read-access or execution. Initialization of unused memory improves reliability. (Note: This is a requirement on the build tool/environment – not on the template)
Use Case:	Fill unused memory (especially ROM/FLASH) with HALT instructions to increase robustness against unwanted code execution. Fill RAM with default value (0 or HALT instruction) to reduce threat of non-deterministic behavior in case of improper pointer operations.
Dependencies:	None identified.
Conflicts:	None identified.
Supporting Material:	None identified.

5 References

[1] Glossary,
https://svn.autosar.org/repos/10Releases/
AUTOSAR_Glossary.pdf

[2] General Requirements on Basic Software Modules,
https://svn.autosar.org/repos/10Releases/
AUTOSAR_SRS_General.pdf

[3] Requirements on RTE Software,
https://svn.autosar.org/repos/10Releases/
AUTOSAR_SRS_RTE.pdf

[4] Requirements on Basic Software Module Description,
https://svn.autosar.org/repos/10Releases/
AUTOSAR_RS_BSW_ModuleDescription.pdf

[5] Template UML Profile and Modeling Guide,
https://svn.autosar.org/repos/10Releases/
AUTOSAR_TemplateModelingGuide.pdf

[6] Model Persistence Rules for XML,
https://svn.autosar.org/repos/10Releases/
AUTOSAR_ModelPersistenceRulesXML.pdf

[7] Specification of Memory Mapping,
https://svn.autosar.org/repos/10Releases/
AUTOSAR_SWS_MemoryMapping.pdf

[8] Software Component Template,
https://svn.autosar.org/repos/10Releases/
AUTOSAR_SoftwareComponentTemplate.pdf

[9] Methodology,
https://svn.autosar.org/repos/10Releases/
AUTOSAR_Methodology.pdf

[10] Technical Overview,
https://svn.autosar.org/repos/10Releases/
AUTOSAR_TechnicalOverview.pdf

[11] Specification of ECU Configuration,
https://svn.autosar.org/repos/10Releases/
AUTOSAR_ECU_Configuration.pdf

[12] Specification of ECU Configuration Parameters,
https://svn.autosar.org/repos/10Releases/
AUTOSAR_SWS_ECU_ConfigationParameters.pdf

[13] Layered Software Architecture,
https://svn.autosar.org/repos/10Releases/
AUTOSAR_LayeredSoftwareArchitecture.pdf