Document Title	Requirements on FlexRay
Document Owner	AUTOSAR GbR
Document Responsibility	AUTOSAR GbR
Document Identification No	005
Document Classification	Auxiliary

Document Version	2.0.3
Document Status	Final
Part of Release	3.0
Revision	0001

Document Change History

Date	Version	Changed by	Change Description
31.10.2007	2.0.3	AUTOSAR Administration	• Document meta information extended • Small layout adaptations made
24.01.2007	2.0.2	AUTOSAR Administration	• “Advice for users” revised • “Revision Information” added
05.12.2006	2.0.1	AUTOSAR Administration	Legal disclaimer revised
12.04.2006	2.0.0	AUTOSAR Administration	• Major rework of most requirements to enhance clarity (e.g. ensured expressive requirement "descriptions" and only brief "short descriptions") • Closed all open requirements, rejected some requirements, and created some new requirements • Chapter "4.1.1 Functional Overview": "Support of the FlexRay Protocol 2.1 specification" instead of "Support of the FlexRay Protocol 2.0 specification" • Added transceiver requirements chapter 4.7 • Changed the allowed retry mechanism in [BSW05083]
23.06.2005	1.0.0	AUTOSAR Administration	Initial Release

Page left intentionally blank

Disclaimer

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 this document

The goal of this document is to define to what extent elements of the basic software have to be configurable and what preliminaries they have to comply with to meet the tailoring requirements.

As far as possible, the set of basic software elements should consist of already existing elements of modules of automotive software. Only in case of "good reasons" new elements of basic software should be part of the set.

If such the definition of these new elements is not part of this work package. Nevertheless the information about basic software elements additionally required has to be given to related work groups.

Constraints and restrictions

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

Data transmission accesses to the FlexRay Interface while FlexRay Communication Controllers are out of sync or in shutdown state shall not return an error.

The FlexRay Transport Layer shall only support half-duplex communication sessions (If both half-duplex and full-duplex session were possible in one implementation, the need for resources and runtime would be increased significantly).

Redundancy (in the time or spatial domain) is not supported by the FlexRay Interface; redundancy has to be explicitly modeled on a higher BSW layer.

There are functional restrictions imposed by the FlexRay Protocol Specification [FR_PROT_SPEC] concerning Wake-Up and Start-up. In both cases, it is not guaranteed by the FlexRay Protocol Specification that the intended result will be reached, i.e. in case of:

- Start-up: It is not guaranteed that the start-up process will be successful, i.e. after initiating a start-up, it is not guaranteed that the FlexRay Communication Controller will successfully establish a communication on the Cluster, or integrate into an ongoing communication, resp.

- Wake-up: It is not guaranteed that the wake-up process will be successful, i.e. after the FlexRay Communication Controller has sent a wake-up pattern on a specific FlexRay Channel, it is not guaranteed that all other FlexRay Communication Controllers connected to this Channel actually do wake up.

A higher AUTOSAR BSW module has to provide a supervision mechanism to ensure that the intended result of these operations will be reached. This mechanism is supported by features requested by the FlexRay Software Requirements Specification documents in hand.

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 Requirement 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
- Dependencies on other SW modules (e.g. Description Templates, Tooling, ...)
- ...

Not each requirement in this SRS does necessarily map to exactly one API call of the respective AUTOSAR FlexRay SW module. It might be possible that one API call fulfills multiple requirements.
It might also be possible that some requirements address internal features of the AUTOSAR FlexRay SW modules and do not require an API call at all in order to be fulfilled.

Implementations of the AUTOSAR FlexRay SW modules used in safety-critical environments might only be allowed to fulfill a subset of the requirements described herein.

Apart from this safety-critical case, every requirement listed in this document is mandatory, i.e. each implementation of an AUTOSAR FlexRay SW module **MUST** fulfill all respective requirements in this document so if that SW module is being used with a hardware supporting the full feature set, the full functionality of the AUTOSAR FlexRay SW module is available.

However, if the underlying hardware does NOT support a specific feature, the AUTOSAR FlexRay SW modules do NOT have to emulate this functionality if that requirement is explicitly marked as "**if supported by the FlexRay CC**".

If a higher layer BSW module calls an API of the AUTOSAR FlexRay SW modules that operates on an optional feature of the FlexRay Specification, and this feature is not supported in the system in question, the respective FlexRay SW modules shall raise an error in debug mode.

In general, the FlexRay Interface has no knowledge of the origin of a PDU passed to it in an API call.

Therefore, throughout this document, unless stated otherwise, the term "PDU" is being used for PDUs originating or sent to:

- AUTOSAR COM (I-PDU), or
- FlexRay TP (N-PDU), or
- FlexRay NM

Throughout this document the term "**configuration**" is being used in several requirements. Unless otherwise noted, this term refers to the configuration of the respective software module that affects the **runtime behavior** but ***not*** the generated **executable code** of this software module. The most prominent case of such a configuration is the FlexRay communication schedule. Enabling the development error handling in the executable code is explicitly not meant.

Accordingly, the term "configuration data" in general refers to pure data (representing a certain configuration) and not to any executable code.

Throughout this document, several scenarios for **changing configuration data** are mentioned. They are being used as follows:

- "**pre compile time**" = carried out *before* compiling the code of the FlexRay Interface/Driver, since the code generation might or will depend on this setting.
- "**at system configuration time**" = static configuration parameters stored in the FlexRay Interface/Driver; may be defined *after* compilation of the code of the FlexRay Interface/Driver, but have to be defined *before* the first execution of the FlexRay Interface/Driver code.
- "**by a flashing process**" = data (not code!) manipulation carried out in a *flashing process* of a flashable memory (in general a Flash-EEPROM) e.g. in a garage, but *not* while the car is being driven. Usually used to *replace* a static configuration *already stored* in the ECU, or a part thereof.
- "**during runtime**" = *dynamically switching* (in normal active mode of the FlexRay CC, if supported by the FlexRay CC) between different configuration parameter sets *statically stored* in the FlexRay Interface.

3 Acronyms and abbreviations

The following acronyms and abbreviations are being used throughout this document

Acronym:	***Description:***
SW	Software
BSW	Basic Software
DEM	Diagnostic Event Manager BSW module
ComM	CommunicationManager BSW module
CC	(FlexRay) Communication Controller

Abbreviation:	***Description:***
i.e.	[lat.] id est = [eng.] that is
e.g.	[lat.] exempli gratia = [eng.] for example

4 Requirement Specification

4.1 Common Requirements (FlexRay Interface and FlexRay Driver)

4.1.1 Functional Overview

- Support of the FlexRay Protocol 2.1 specification
- Abstraction of FlexRay specific features
- Abstraction of FlexRay Controller type specific features
- Support of at least 4 (possibly different) FlexRay Controllers per ECU with up to 2 Channels each and data rates up to 10 MBit/s
- Connect the communication interface to the specific FlexRay Controller
- Transmit and receive FlexRay Frames in the static and in the dynamic segment
- Support of multiple FlexRay Clusters
- Support of (with respect to the FlexRay global time) asynchronous and synchronous operation of the communication stack
- Packing and unpacking of multiple PDU into/out of one or more FlexRay Frames
- Interface: PDU-based API to upper software layer
- Single/multiple sender Cycle Multiplexing for scalable efficient bandwidth exploitation
- Configuration of the complete FlexRay system at system configuration time
- Run time reconfiguration of FlexRay Controllers
- Provision of basic network management functions: Transmit/detect wake-up patterns; start-up, shutdown, sleep mode, start and stop communication
- Provision of the FlexRay Controller state e.g. for usage by FlexRay NM
- Provision of FlexRay bus errors

For an overview of the structure of AUTOSAR software modules please see the current AUTOSAR Layered Software Architecture Specification [LSA].

4.1.2 Functional Requirements

4.1.2.1 General requirements

4.1.2.1.1 [BSW05000] Support of Synchronous SW Modules

Initiator:	BMW
Date:	23.07.2004
Short Description:	Support of applications (or software modules) running synchronously to the FlexRay global time.

Type:	New
Importance:	High
Description:	The FlexRay-related BSW modules of the AUTOSAR communication stack shall support applications (or software modules) running synchronously to the FlexRay global time.
Rationale:	An application (or software module) running synchronously to the FlexRay global time (i.e. it is driven by the FlexRay global time) shall be able to efficiently communicate via FlexRay, since an application (or software module) running synchronously to the FlexRay global time allows an optimal adaptation of the data processing schedule to the schedule of the data transmission via FlexRay.
Use Case:	For some applications (e.g. chassis function as regulator) it is advantageous to run the application synchronously to sensors and actuators connected to the FlexRay bus, hence to the FlexRay global time.
Dependencies:	[BSW05001] Support of Asynchronous SW Modules
Conflicts:	--
Supporting Material:	--

4.1.2.1.2 [BSW05001] Support of Asynchronous SW Modules

Initiator:	BMW
Date:	23.07.2004
Short Description:	Support of applications (or software modules) running asynchronously to the FlexRay global time.
Type:	New
Importance:	High
Description:	The FlexRay-related BSW modules of the AUTOSAR communication stack shall support applications (or software modules) running asynchronously to the FlexRay global time.
Rationale:	An application (or software module) running asynchronously to the FlexRay global time (i.e. it has no "knowledge" about the FlexRay global time) shall be able to communicate via FlexRay, since an application (or software module) running asynchronously to the FlexRay global time allows a flexible software design.
Use Case:	For some applications (e.g. body or gateway) it is advantageous to run the application event-triggered, hence asynchronously to the FlexRay global time.
Dependencies:	[BSW05000] Support of Synchronous SW Modules
Conflicts:	--
Supporting Material:	--

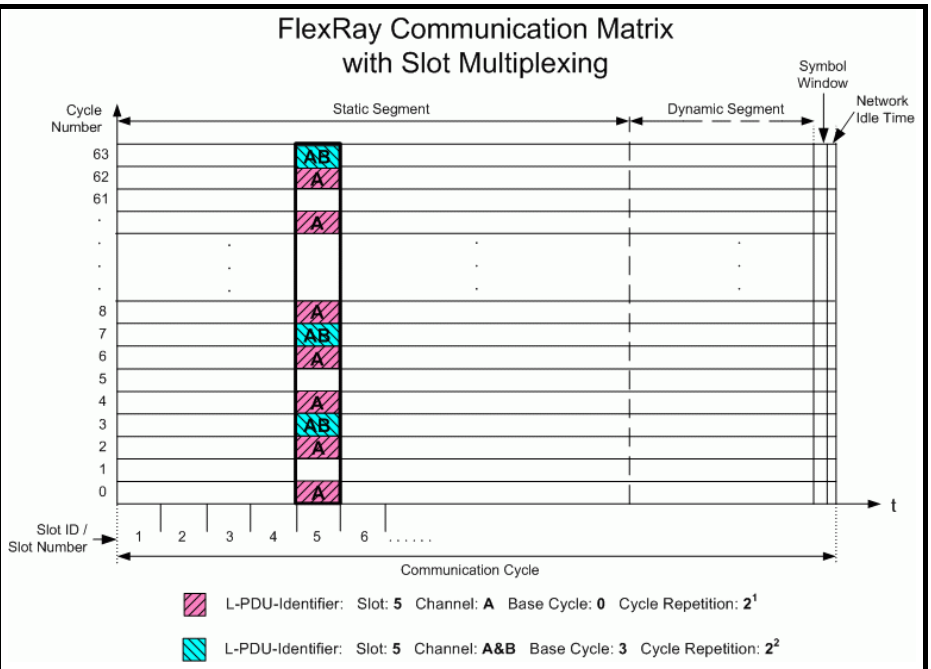
4.1.2.1.3 [BSW05002] FlexRay Interface and FlexRay Driver as Only Necessarily Synchronous SW Modules

Initiator:	BMW
Date:	23.07.2004
Short Description:	Support of completely asynchronous usage of the synchronously running FlexRay-related BSW modules by all upper-layer SW modules.
Type:	New
Importance:	High
Description:	The FlexRay-related BSW modules of the AUTOSAR communication stack shall encapsulate all synchronous services to enable a completely asynchronous usage of the FlexRay-related BSW modules by all upper-layer

	SW modules. Therefore, the FlexRay Interface shall operate synchronized to the FlexRay global time(s) of all involved FlexRay Communication Controllers by using services of the respective FlexRay Driver.
Rationale:	The FlexRay Interface and the FlexRay Driver shall be synchronous to the FlexRay bus to support synchronous communication and to avoid buffering and queuing. Also, each upper-layer SW module shall have the possibility to run asynchronously to the FlexRay global time. If [BSW05001] applies to all other SW modules (i.e. all upper-layer software modules run completely asynchronously to any FlexRay global time), the FlexRay Interface and the FlexRay Driver(s) it controls are the only synchronous SW modules. Note that (for example in case of asynchronous gateways) the global time of multiple FlexRay Controllers in a single ECU might be different, since they are connected to different FlexRay Clusters. In this case, the respective operations for each single Communication Controller are to be performed synchronously to this Communication Controller's global time.
Use Case:	An ECU with all SW modules running asynchronously to the FlexRay global time.
Dependencies:	[BSW05000] Support of Synchronous SW Modules [BSW05001] Support of Asynchronous SW Modules
Conflicts:	--
Supporting Material:	--

4.1.2.1.4 [BSW05003] Support of Slot/Cycle Multiplexing

Initiator:	BMW
Date:	23.07.2004
Short Description:	Support of the FlexRay Slot/Cycle Multiplexing mechanism.
Type:	New
Importance:	High
Description:	The FlexRay-related BSW modules of the AUTOSAR communication stack shall support the FlexRay Slot/Cycle Multiplexing mechanism which is used to use a FlexRay Slot on a Channel more efficiently by alternating the Frames being sent in this Slot from Cycle to Cycle. In the static segment, the FlexRay Slot/Cycle Multiplexing mechanism allows an ECU to transmit different Frame contents in the same Slot in different Cycles. This is called "Single Sender Slot Multiplexing". In the dynamic segment, the FlexRay Cycle Multiplexing mechanism allows several ECUs to share the same Slot by uniquely assigning certain Cycles (to be accurate: FlexRay Cells of the same Slot) to each ECU. This is called "Multiple Sender Slot Multiplexing". Both kinds of Slot/Cycle Multiplexing mechanisms shall be supported by the FlexRay-related BSW modules.
Rationale:	Optimal uses of the FlexRay bandwidth can only be reached with an extensive and effective use of the Cycle Multiplexing mechanism.
Use Case:	In the following example, an ECU sends two different Frame contents in the same Slot and thus effectively uses this resource.

	
Dependencies:	--
Conflicts:	--
Supporting Material:	[FR_PROT_SPEC]

4.1.2.1.5 [BSW05169] Avoid Timer Interrupts during Start-up

Initiator:	BMW
Date:	13.09.2004
Short Description:	Avoid FlexRay Communication Controller timer interrupts during the start-up phase of the ECU.
Type:	New
Importance:	High
Description:	The FlexRay-related BSW modules of the AUTOSAR communication stack shall ensure that no timer interrupt raised from a FlexRay Communication Controller will be forwarded to other software modules during the start-up phase of the ECU.
Rationale:	Interrupts occurring before the entire ECU software has been initialized may cause problems.
Use Case:	--
Dependencies:	[BSW05055] Avoid Timer Interrupts during Shutdown
Conflicts:	--
Supporting Material:	--

4.1.2.1.6 [BSW05055] Avoid Timer Interrupts during Shutdown

Initiator:	BMW
Date:	29.07.2004
Short Description:	Avoid FlexRay Communication Controller timer interrupts during the shutdown phase of the ECU.
Type:	New
Importance:	High
Description:	The FlexRay-related BSW modules of the AUTOSAR communication stack

	shall ensure that no timer interrupt raised from a FlexRay Communication Controller will be forwarded to other software modules during the shutdown phase of the ECU.
Rationale:	Interrupts occurring while the ECU software is being shut down may cause problems.
Use Case:	--
Dependencies:	[BSW05169] Avoid Timer Interrupts during Start-up
Conflicts:	--
Supporting Material:	--

4.2 FlexRay Interface

4.2.1 Functional Overview

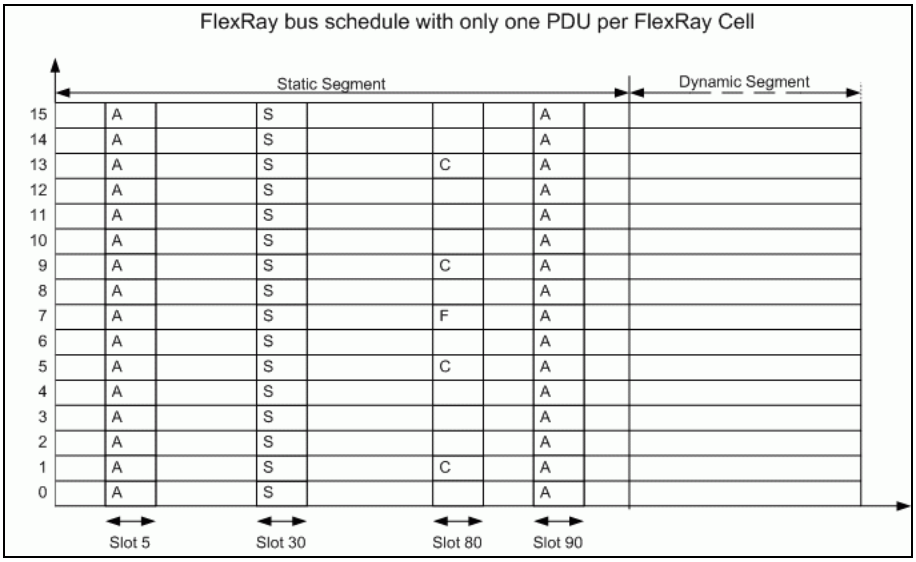
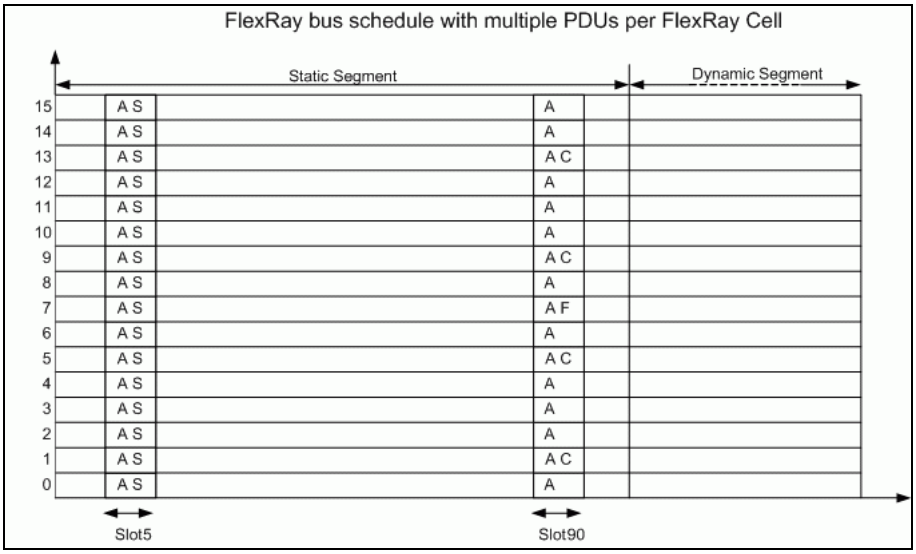
The FlexRay Interface shall provide a standardized interface to access the FlexRay Communication system/hardware. The FlexRay Interface shall be independent of the specific FlexRay CC(s) being used and its/their access through the FlexRay Driver(s). The FlexRay Interface shall give access to one or several FlexRay Drivers via one uniform interface.

4.2.2 Functional Requirements

4.2.2.1 General Requirement

4.2.2.1.1 [BSW05004] PDU-Based Data API

Initiator:	BMW
Date:	23.07.2004
Short Description:	PDU-based data API to all upper layers.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a PDU-based data API to all upper layers.
Rationale:	A PDU-based data API is needed in order to fulfill [BSW05003] in an optimal way. In order to fulfill [BSW05002], the FlexRay Interface needs to abstract the synchronous features of the FlexRay protocol. In order to fulfill [BSW05003], a FlexRay Slot-based API is not possible. On the other hand, a FlexRay Cell-based API needs a lot of resources (one local memory space per FlexRay Cell) and performance (e.g. for the identification of the most recently received value). Moreover, in the static segment of the FlexRay Cycle, all FlexRay Cells always have the same length, independent of the Cell contents. To obtain an optimal use of the FlexRay bandwidth, the FlexRay Interface shall allow transferring PDUs with different transmission periods in the same FlexRay Cell.
Use Case:	The following series-relevant example: An ECU sends the following PDUs using a 5ms Cycle: • Acceleration (A) with a 2,5ms period • Status (S) with a 5ms period

	• Controller Status (C) with a 20ms period • Fault Status (F) with a 80ms period   Conclusion: Only half the bandwidth is required if multiple PDUs are scheduled per FlexRay Cell.
Dependencies:	[BSW05002] FlexRay Modules as Only Necessarily Synchronous SW Modules [BSW05003] Support of Slot/Cycle Multiplexing
Conflicts:	--
Supporting Material:	--

4.2.2.1.2 [BSW05010] Unique PDU-ID

Initiator:	BMW
Date:	23.07.2004
Short Description:	Identify each PDU with a unique PDU-ID.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall identify each PDU with a unique PDU-ID, based on the operation to be carried out with this PDU.
Rationale:	Unique PDU-IDs are being used in each BSW module throughout the

	AUTOSAR Com stack to identify each single PDU and to decide on the handling of the respective PDU. Within the FlexRay Interface, independent sets of PDU-IDs may be used for independent types of operation (i.e. sending & receiving). Unlike with other communication systems – as for example CAN – the Frame-ID is not sufficient for the identification of an PDU
Use Case:	Interaction of the FlexRay Interface with higher layer BSW modules, e.g. the PDU-Router.
Dependencies:	[BSW05004] PDU-Based API
Conflicts:	--
Supporting Material:	--

4.2.2.1.3 [BSW05126] PDU Update/Valid Information

Initiator:	BMW
Date:	16.02.2005
Short Description:	Generate, transmit, and (upon reception) handle PDU-based update/valid information where necessary.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall be configurable at system configuration time to generate, transmit, and (upon reception) handle PDU-based update/valid information where necessary. Per PDU, this information shall not consume more than one bit in the FlexRay Frame. This feature shall be configurable at system configuration time independently for each FlexRay Frame.
Rationale:	In certain configurations of the FlexRay Interface and the FlexRay CC, this information is necessary in order for the receiving FlexRay Interface to be able to decide whether a specific PDU contained in a received FlexRay Frame is up-to-date or outdated.
Use Case:	Assume two PDUs to be packed into the same FlexRay Frame. Assume further that only one of the two PDUs has been updated before the scheduled sending of this Frame, and the other PDU has not. The receiver needs to be able to recognize which of the PDUs is valid (i.e. contains updated data) and which is not (i.e. contains invalid or old data). Therefore, this update information has to be generated by the sending FlexRay Interface and transmitted via the FlexRay bus to the receiving FlexRay Interface which evaluates it and acts according to this evaluation.
Dependencies:	--
Conflicts:	--
Supporting Material:	Signal update information of COM

4.2.2.1.4 [BSW05097] Number of FlexRay Drivers per FlexRay Interface

Initiator:	DECOMSYS
Date:	20.01.2005
Short Description:	Communicate with at least four FlexRay Drivers.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall be able to communicate with at least four FlexRay Drivers. The actual number of FlexRay Drivers serviced by the FlexRay Interface shall be defined at system configuration time and shall be less or equal to the number of FlexRay Drivers this FlexRay Interface

	supports.
Rationale:	Required in order to fulfill [BSW05007] for four different types of FlexRay Communication Controllers, each of them requiring their own FlexRay Driver.
Use Case:	--
Dependencies:	[BSW05007] Number of FlexRay CCs per Interface [BSW05065] Number of FlexRay CCs per Driver
Conflicts:	--
Supporting Material:	--

4.2.2.1.5 [BSW05007] Number of FlexRay CCs per Interface

Initiator:	BMW
Date:	23.07.2004
Short Description:	Communicate with at least four FlexRay CCs.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall be able to communicate with at least four FlexRay CCs via the appropriate FlexRay Driver(s). The actual number of CCs operated by the FlexRay Interface shall be defined at system configuration time and shall be less or equal to the number of CCs this FlexRay Interface supports.
Rationale:	Future network topologies might demand more than one FlexRay CC on one ECU.
Use Case:	An ECU connected to two FlexRay Clusters.
Dependencies:	[BSW05065] Number of FlexRay CCs per Driver [BSW05097] Number of FlexRay Drivers per FlexRay Interface
Conflicts:	--
Supporting Material:	--

4.2.2.1.6 [BSW05130] Transmit Request Queuing

Initiator:	BMW
Date:	20.07.2005
Short Description:	Support of PDU transmission buffer queues located in a higher BSW module.
Type:	General
Importance:	High
Description:	The FlexRay Interface shall support PDU transmission buffer queues (FIFOs) of configurable size located in a higher BSW module by queuing the transmit requests originating from this BSW module.
Rationale:	A higher BSW module might use a FIFO to queue PDUs to be transmitted, but might forward the transmission request to the FlexRay Interface faster (within a limited time interval) than this PDU can be transmitted via FlexRay.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.2.2.2 Configuration

4.2.2.2.1 [BSW05060] Scheduling of Copy Operation into/from FlexRay CC

Initiator:	BMW
Date:	15.09.2004
Short Description:	Copy the data into/from the FlexRay CC's transmit/receive buffers at the desired point in the FlexRay global time.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall be configured at system configuration time to copy, by means of the corresponding FlexRay Driver, the data into/from the FlexRay CC's transmit/receive buffers at the desired point in the FlexRay global time. This copy operation shall be carried out in a task configured (i.e. scheduled) at system configuration time. This task has to be synchronous to the communication schedule of the involved FlexRay CC. Together with the copy operation, a reconfiguration of a transmit/receive buffer during normal active mode may be carried out, if supported by the FlexRay CC. The configuration of the copy operation (i.e. the schedule) shall be changeable by a flashing process.
Rationale:	FlexRay Controllers have to be serviced synchronously to the FlexRay communication schedule.
Use Case:	--
Dependencies:	[BSW05058] Configuration of FlexRay Driver at System Configuration Time [BSW05034] Configuration Modifiable by a Flashing Process
Conflicts:	--
Supporting Material:	--

4.2.2.2.2 [BSW05096] Assignment of Drivers to Controllers

Initiator:	DECOMSYS
Date:	20.01.2005
Short Description:	Assignment of FlexRay Drivers to FlexRay Controllers.
Type:	New
Importance:	High
Description:	The configuration of the FlexRay Interface shall specify which FlexRay Communication Controller is to be served by which FlexRay Driver. Thus a mapping between used FlexRay Drivers and available FlexRay Controllers is made though the interface configuration at system configuration time.
Rationale:	This information has to be available to the FlexRay Interface in order to invoke the correct Driver for a given FlexRay Controller on an ECU.
Use Case:	--
Dependencies:	[BSW05056] Configuration of the FlexRay Interface at System Configuration Time
Conflicts:	--
Supporting Material:	--

4.2.2.3 Initialization

4.2.2.3.1 [BSW05013] Initialize Local Memory Space

Initiator:	BMW
Date:	26.07.2004
Short Description:	Initialize the FlexRay Interface's local memory space.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to initialize the local memory space used to store the PDU data and their corresponding properties.
Rationale:	The initialization of the local memory space is necessary to store the transferred data from the FlexRay Controller. Usually, variables (e.g. pointers) have to be initialized before they are being used during runtime.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.2.2.3.2 [BSW05031] Initialization of a FlexRay CC

Initiator:	BMW
Date:	29.07.2004
Short Description:	Initialization and configuration of a specific FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to initialize and configure a specific FlexRay CC. Thereby, the transmit/receive buffers and the low level parameters of the FlexRay CC shall be configured.
Rationale:	--
Use Case:	Initial configuration.
Dependencies:	[BSW05116] Initialization of FlexRay CC [BSW05078] Initialization of a FlexRay Cluster [BSW05011] Initialize Low-Level Parameters [BSW05012] Initialize FlexRay CC Transmit/Receive Buffers
Conflicts:	--
Supporting Material:	--

4.2.2.3.3 [BSW05078] Initialization of a FlexRay Cluster

Initiator:	BMW
Date:	02.02.2005
Short Description:	Initialization and configuration of all FlexRay CCs connected to a specific FlexRay Cluster.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to initialize and configure all FlexRay CCs connected to a specific FlexRay Cluster. Thereby, the transmit/receive buffers and the low level parameters of these FlexRay Controllers shall be configured.
Rationale:	--

Use Case:	Initial configuration.
Dependencies:	[BSW05031] Initialization of a FlexRay CC
Conflicts:	--
Supporting Material:	--

4.2.2.3.4 [BSW05034] Configuration Modifiable by a Flashing Process

Initiator:	BMW
Date:	16.08.2004
Short Description:	Modification of configuration data by a flashing process.
Type:	New
Importance:	High
Description:	All configuration data of the FlexRay Interface defined at system configuration time shall be modifiable by a flashing process. These modifiable configuration data encompass (amongst other configuration items) the scheduling of the copy operation into/from the FlexRay CCs' transmit/receive buffers as well as the configuration of those buffers and the FlexRay Interface's local memory space. The modification shall be carried out in a specific mode (e.g. in the garage) and not during run-time (i.e. while the car is being driven).
Rationale:	Re-flashing of a changed communication schedule.
Use Case:	If e.g. the only modification in a configuration is the Frame-ID of a specific PDU (i.e. the FlexRay Frame in which this PDU is being sent), this shall not automatically implicate a new compilation of the complete ECU code! Only the altered parameters need to be modified.
Dependencies:	[BSW05060] Scheduling of Copy Operation into/from FlexRay CC [BSW05012] Initialize FlexRay CC Transmit/Receive Buffers [BSW05013] Initialize Local Memory Space
Conflicts:	--
Supporting Material:	--

4.2.2.3.5 [BSW05042] Switch Configuration in Normal Active Mode

Initiator:	BMW
Date:	30.08.2004
Short Description:	Switching the configuration of a specific FlexRay Communication Controller's transmit/receive buffers and the local memory space during runtime (i.e. in normal active mode), if supported by this FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall allow switching from one configuration of a specific FlexRay Communication Controller's transmit/receive buffers and the local memory space to another one, both stored in the ECU's memory space. This switching shall be possible during runtime, i.e. while the CC is in normal active mode, if supported by this FlexRay CC. It shall be ensured that none of the selectable configurations violate the rules for Slot Multiplexing set forth in [FR_PROT_SPEC]. The switching of the configuration includes a reconfiguration of the FlexRay Communication Controller's transmit/receive buffers. Therefore the following restrictions apply:

	• In the dynamic segment any FlexRay Communication Controller's transmit/receive buffer shall be used maximally once per Cycle. This limits the reconfiguration possibilities and thus restricts the number of transmittable (sent and received) Frames per dynamic segment to the accumulated number (over all CCs on one ECU) of transmit/receive buffers connected to one Cluster. The FlexRay Interface shall support at least 4 different configurations. The actual number of configurations stored in the ECU's memory space shall be defined at system configuration time and shall be less or equal to the number of configurations this FlexRay Interface supports. For safety reasons it shall be possible to completely deactivate this feature pre compile time. For safety reason such a switching operation shall not be carried out while the car is being driven, but only in the garage in a specific mode.
Rationale:	--
Use Case:	Specific flash mode for the gateway: In order to increase the throughput of the gateway, and thus speed up the flashing process, transmit/receive buffers normally used for application data transmission could temporarily be used for the flashing process.
Dependencies:	[BSW05012] Initialize FlexRay CC Transmit/Receive Buffers [BSW05013] Initialize Local Memory Space
Conflicts:	--
Supporting Material:	--

4.2.2.4 Start-up Operation

4.2.2.4.1 [BSW05015] Start-up of a FlexRay CC

Initiator:	BMW
Date:	29.07.2004
Short Description:	Start-up of a specific FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to start-up a specific FlexRay CC.
Rationale:	Start-up of a specific FlexRay Controller after proper configuration.
Use Case:	--
Dependencies:	[BSW05109] Start-up of a FlexRay CC [BSW05031] Initialization of a FlexRay CC
Conflicts:	--
Supporting Material:	--

4.2.2.4.2 [BSW05101] Start-up of a FlexRay Cluster

Initiator:	BMW
Date:	02.02.2005
Short Description:	Start-up of all FlexRay CCs connected to a specific FlexRay Cluster.
Type:	New
Importance:	High
Description:	The Interface shall provide a software interface to start-up all FlexRay CCs connected to a specific FlexRay Cluster.

Rationale:	--
Use Case:	Start-up of a specific FlexRay Cluster after proper configuration.
Dependencies:	[BSW05015] Start-up of a FlexRay CC
Conflicts:	--
Supporting Material:	--

4.2.2.4.3 [BSW05018] Sending of a Wake-Up Pattern

Initiator:	BMW
Date:	29.07.2004
Short Description:	Sending of a wake-up pattern on a specific FlexRay Channel of a specific FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to send a wake-up pattern on a specific FlexRay Channel of a specific FlexRay CC. The FlexRay Specification allows only a wake-up on one Channel at a time. However, the FlexRay Interface shall support the sending of a wake-up pattern on any one of the two FlexRay Channels. The FlexRay Interface shall observe the restrictions set forth in the FlexRay Protocol Specification concerning reception of a wake-up pattern or a Frame header during own wake-up pattern sending attempts.
Rationale:	Enable waking up a FlexRay Cluster.
Use Case:	--
Dependencies:	[BSW05117] Sending of Wake-Up Pattern
Conflicts:	--
Supporting Material:	[FR_PROT_SPEC]

4.2.2.4.4 [BSW05158] Get FlexRay Transceiver Wake-up Reason

Initiator:	VW
Date:	24.01.2006
Short Description:	Get the wake-up reason of a specific FlexRay Transceiver device.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to get the wake-up reason of a specific FlexRay Transceiver device.
Rationale:	Provide EcuM with a possibility to find out the wake-up reason.
Use Case:	--
Dependencies:	[BSW05144] Get FlexRay Transceiver Wake-up Reason
Conflicts:	--
Supporting Material:	[FR_EPL_SPEC]

4.2.2.4.5 [BSW05159] Enable FlexRay Transceiver Wake-up Indication

Initiator:	VW
Date:	24.01.2006
Short Description:	Enable the wake-up indication of a specific FlexRay Transceiver device.
Type:	New

Importance:	High
Description:	The FlexRay Interface shall provide a software interface to enable the wake-up indication of a specific FlexRay Transceiver device.
Rationale:	EcuM might be interested in a wake-up event via FlexRay.
Use Case:	--
Dependencies:	[BSW05160] Disable FlexRay Transceiver Wake-up Indication [BSW05163] Cluster-wide Enable FlexRay Transceiver Wake-up Indication [BSW05164] Cluster-wide Disable FlexRay Transceiver Wake-up Indication
Conflicts:	--
Supporting Material:	[FR_EPL_SPEC]

4.2.2.4.6 [BSW05163] Cluster-wide Enable FlexRay Transceiver Wake-up Indication

Initiator:	VW
Date:	24.01.2006
Short Description:	Enable the wake-up indication of all FlexRay Transceiver devices connected to a specific FlexRay Cluster.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to enable the wake-up indication of all FlexRay Transceiver devices connected to a specific FlexRay Cluster.
Rationale:	EcuM might be interested in wake-up events via FlexRay.
Use Case:	--
Dependencies:	[BSW05159] Enable FlexRay Transceiver Wake-up Indication [BSW05160] Disable FlexRay Transceiver Wake-up Indication [BSW05164] Cluster-wide Disable FlexRay Transceiver Wake-up Indication
Conflicts:	--
Supporting Material:	[FR_EPL_SPEC]

4.2.2.4.7 [BSW05160] Disable FlexRay Transceiver Wake-up Indication

Initiator:	VW
Date:	24.01.2006
Short Description:	Disable the wake-up indication of a specific FlexRay Transceiver device.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to disable the wake-up indication of a specific FlexRay Transceiver device.
Rationale:	EcuM might not be interested in a wake-up event via FlexRay.
Use Case:	--
Dependencies:	[BSW05159] Enable FlexRay Transceiver Wake-up Indication [BSW05163] Cluster-wide Enable FlexRay Transceiver Wake-up Indication [BSW05164] Cluster-wide Disable FlexRay Transceiver Wake-up Indication
Conflicts:	--
Supporting Material:	[FR_EPL_SPEC]

4.2.2.4.8 [BSW05164] Cluster-wide Disable FlexRay Transceiver Wake-up Indication

Initiator:	VW
Date:	24.01.2006
Short Description:	Disable the wake-up indication of all FlexRay Transceiver devices connected to a specific FlexRay Cluster.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to disable the wake-up indication of all FlexRay Transceiver devices connected to a specific FlexRay Cluster.
Rationale:	EcuM might not be interested in wake-up events via FlexRay.
Use Case:	--
Dependencies:	[BSW05159] Enable FlexRay Transceiver Wake-up Indication [BSW05160] Disable FlexRay Transceiver Wake-up Indication [BSW05163] Cluster-wide Enable FlexRay Transceiver Wake-up Indication
Conflicts:	--
Supporting Material:	[FR_EPL_SPEC]

4.2.2.4.9 [BSW05161] Clear FlexRay Transceiver Wake-up Events

Initiator:	VW
Date:	24.01.2006
Short Description:	Clear all pending wake-up events of a specific FlexRay Transceiver device.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to clear all pending wake-up events of a specific FlexRay Transceiver device.
Rationale:	EcuM needs to reset the wake-up events of FlexRay Transceiver devices.
Use Case:	--
Dependencies:	[BSW05165] Cluster-wide Clear FlexRay Transceiver Wake-up Events
Conflicts:	--
Supporting Material:	[FR_EPL_SPEC]

4.2.2.4.10 [BSW05165] Cluster-wide Clear FlexRay Transceiver Wake-up Events

Initiator:	VW
Date:	24.01.2006
Short Description:	Clear all pending wake-up events of all FlexRay Transceiver devices connected to a specific FlexRay Cluster.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to clear all pending wake-up events of all FlexRay Transceiver devices connected to a specific FlexRay Cluster.
Rationale:	EcuM needs to reset the wake-up events of FlexRay Transceiver devices.
Use Case:	--
Dependencies:	[BSW05161] Clear FlexRay Transceiver Wake-up Events
Conflicts:	--
Supporting Material:	[FR_EPL_SPEC]

4.2.2.5 FlexRay Specific Information

This section includes all the FlexRay-specific requirements.

4.2.2.5.1 [BSW05067] Set FlexRay Cluster Offline Mode

Initiator:	BMW
Date:	17.12.2004
Short Description:	Set a specific FlexRay Cluster into communication mode "cluster offline".
Type:	New
Importance:	Medium
Description:	The FlexRay Interface shall provide a software interface to set a specific Cluster into communication mode "cluster offline" which is achieved by setting all FlexRay Communication Controllers connected to this Cluster into mode "controller offline". If a FlexRay Communication Controller is in communication mode "controller offline", it shall be ensured that no PDU data is being copied from/to this CC's transmit/receive buffers, and that neither a PDU reception indication nor a PDU transmit confirmation is forwarded to a higher layer BSW module. Note: This does not necessarily result in an immediate termination of data transmission via this FlexRay CC, since already committed transmit buffers might still send their buffer contents. However, further filling and committing of transmit buffers is prohibited.
Rationale:	The communication mode "cluster offline" is only a software possibility to (temporarily, e.g. due to a temporary error condition) stop the data transfer for a specific FlexRay Cluster, i.e. for all FlexRay Communication Controllers connected to this Cluster. It has no impact on the hardware, i.e. these FlexRay CCs do NOT lose synchronicity to the FlexRay global time, and higher-layer SW modules may access the FlexRay Interface like in communication mode "cluster online".
Use Case:	FlexRay communication on a specific Cluster has to be stopped temporarily, but the involved FlexRay CC(s) should remain synchronous to this Cluster, since the resynchronization to the FlexRay global time would take too much time.
Dependencies:	[BSW05068] Set FlexRay Cluster Online Mode [BSW05069] Get FlexRay Cluster Mode [BSW05154] Set FlexRay Controller Offline Mode [BSW05155] Set FlexRay Controller Online Mode [BSW05153] Get FlexRay Controller Mode
Conflicts:	--
Supporting Material:	--

4.2.2.5.2 [BSW05154] Set FlexRay Controller Offline Mode

Initiator:	BMW
Date:	16.01.2006
Short Description:	Set a specific FlexRay Communication Controller into communication mode "controller offline".
Type:	New
Importance:	Medium
Description:	The FlexRay Interface shall provide a software interface to set a specific

	FlexRay Communication Controller into mode "controller offline". If a FlexRay Communication Controller is in communication mode "controller offline", it shall be ensured that no PDU data is being copied from/to this CC's transmit/receive buffers, and that neither a PDU reception indication nor a PDU transmit confirmation is forwarded to a higher layer BSW module. Note: This does not necessarily result in an immediate termination of data transmission via this FlexRay CC, since already committed transmit buffers might still send their buffer contents. However, further filling and committing of transmit buffers is prohibited.
Rationale:	The communication mode "controller offline" is only a software possibility to (temporarily, e.g. due to a temporary error condition) stop the data transfer for a specific FlexRay Communication Controller. It has no impact on the hardware, i.e. this FlexRay CC does NOT lose synchronicity to the FlexRay global time, and higher-layer SW modules may access the FlexRay Interface like in communication mode "controller online".
Use Case:	FlexRay communication via a specific FlexRay CC has to be stopped temporarily, but the FlexRay CC should remain synchronous to the Cluster, since the resynchronization to the FlexRay global time would take too much time.
Dependencies:	[BSW05067] Set FlexRay Cluster Offline Mode [BSW05068] Set FlexRay Cluster Online Mode [BSW05069] Get FlexRay Cluster Mode [BSW05155] Set FlexRay Controller Online Mode [BSW05153] Get FlexRay Controller Mode
Conflicts:	--
Supporting Material:	--

4.2.2.5.3 [BSW05068] Set FlexRay Cluster Online Mode

Initiator:	BMW
Date:	17.12.2004
Short Description:	Set a specific FlexRay Cluster into communication mode "cluster online".
Type:	New
Importance:	Medium
Description:	The FlexRay Interface shall provide a software interface to set a specific Cluster into communication mode "cluster online" which is achieved by setting all FlexRay Communication Controllers connected to this Cluster into mode "controller online". If a FlexRay Communication Controller is in communication mode "controller online", it shall be ensured that PDU data is being copied from/to this CC's transmit/receive buffers according to the FlexRay communication schedule.
Rationale:	The communication mode "cluster online" is the counterpart of the communication mode "cluster offline". If a FlexRay Cluster has been set to mode "cluster offline" due to a temporary error condition, it can be set back to mode "cluster online" once the error condition is removed.
Use Case:	--
Dependencies:	[BSW05067] Set FlexRay Cluster Offline Mode [BSW05069] Get FlexRay Cluster Mode [BSW05154] Set FlexRay Controller Offline Mode [BSW05155] Set FlexRay Controller Online Mode [BSW05153] Get FlexRay Controller Mode
Conflicts:	--

Supporting Material:	--

4.2.2.5.4 [BSW05155] Set FlexRay Controller Online Mode

Initiator:	BMW
Date:	16.01.2006
Short Description:	Set a specific FlexRay Communication Controller into communication mode "controller online".
Type:	New
Importance:	Medium
Description:	The FlexRay Interface shall provide a software interface to set a specific FlexRay Communication Controller into communication mode "controller online". If a FlexRay Communication Controller is in communication mode "controller online", it shall be ensured that PDU data is being copied from/to this CC's transmit/receive buffers according to the FlexRay communication schedule.
Rationale:	The communication mode "controller online" is the counterpart of the communication mode "controller offline". If a FlexRay Communication Controller has been set to mode "controller offline" due to a temporary error condition, it can be set back to mode "controller online" once the error condition is removed.
Use Case:	--
Dependencies:	[BSW05067] Set FlexRay Cluster Offline Mode [BSW05068] Set FlexRay Cluster Online Mode [BSW05069] Get FlexRay Cluster Mode [BSW05154] Set FlexRay Controller Offline Mode [BSW05153] Get FlexRay Controller Mode
Conflicts:	--
Supporting Material:	--

4.2.2.5.5 [BSW05069] Get FlexRay Cluster Mode

Initiator:	BMW
Date:	21.01.2005
Short Description:	Get the current communication mode ("cluster online"/"cluster offline") of a specific FlexRay Cluster.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to get the current communication mode ("cluster online"/"cluster offline") of a specific FlexRay Cluster. If more than one FlexRay Communication Controller is connected to a specific Cluster, the state "cluster offline" shall be yielded unless all FlexRay Communication Controllers connected to this Cluster are in state "controller online".
Description:	--
Rationale:	The upper-layer software shall have the possibility to get the current mode ("cluster online"/"cluster offline") of a specific FlexRay Cluster.
Use Case:	--
Dependencies:	[BSW05067] Set FlexRay Cluster Offline Mode [BSW05068] Set FlexRay Cluster Online Mode [BSW05154] Set FlexRay Controller Offline Mode [BSW05155] Set FlexRay Controller Online Mode [BSW05153] Get FlexRay Controller Mode

Conflicts:	--
Supporting Material:	--

4.2.2.5.6 [BSW05153] Get FlexRay Controller Mode

Initiator:	BMW
Date:	16.01.2006
Short Description:	Get the current communication mode ("controller online"/"controller offline") of a specific FlexRay Communication Controller.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to get the current communication mode ("controller online"/"controller offline") of a specific FlexRay Communication Controller.
Description:	--
Rationale:	The upper-layer software shall have the possibility to get the current mode ("controller online"/"controller offline") of a specific FlexRay Controller.
Use Case:	--
Dependencies:	[BSW05067] Set FlexRay Cluster Offline Mode [BSW05068] Set FlexRay Cluster Online Mode [BSW05069] Get FlexRay Cluster Mode [BSW05154] Set FlexRay Controller Offline Mode [BSW05155] Set FlexRay Controller Online Mode
Conflicts:	--
Supporting Material:	--

4.2.2.5.7 [BSW05022] Get FlexRay CC POC Status

Initiator:	BMW
Date:	29.07.2004
Short Description:	Get the CC POC status of a specific FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to get the CC POC status of a specific FlexRay CC. All information contained in the vPOC structure as defined in chapter "2.2.1.3 POC status" of the [FR_PROT_SPEC] shall be returned.
Rationale:	Software modules like Mode Manager, Network Management, or DCM might be interested in the FlexRay CC POC status.
Use Case:	--
Dependencies:	[BSW05120] Get FlexRay CC POC Status
Conflicts:	--
Supporting Material:	--

4.2.2.5.8 [BSW05023] Get FlexRay CC Sync State

Initiator:	BMW
Date:	29.07.2004
Short Description:	Get the sync state of a specific FlexRay CC.
Type:	New
Importance:	High

Description:	The FlexRay Interface shall provide a software interface to get the sync state ("controller synchronous"/"controller asynchronous") of a specific FlexRay CC.
Rationale:	In order to be able to communicate via FlexRay, it is crucial that the FlexRay CC be synchronous to the FlexRay Cluster's global time. Software modules like Mode Manager, Network Management, or DCM might be interested in the sync state of a specific FlexRay CC.
Use Case:	A distributed control might only be started once the communication via the FlexRay bus is possible. Therefore, knowledge about the sync state of the FlexRay CC is crucial.
Dependencies:	[BSW05121] Get FlexRay CC Sync State
Conflicts:	--
Supporting Material:	--

4.2.2.5.9 [BSW05035] MTS Sending

Initiator:	BMW / Freescale
Date:	16.08.2004
Short Description:	Sending of a Media Access Test Symbol.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to send a Media Access Test Symbol (MTS) on a specific FlexRay Channel of a specific FlexRay CC.
Rationale:	The sending and the resulting reception status of a Media Access Test Symbol are used to decide whether the Bus Guardian is working correctly in the Symbol Window.
Use Case:	--
Dependencies:	[BSW05107] MTS Sending [BSW05038] Get MTS Reception Status [BSW05111] Get MTS Reception Status
Conflicts:	--
Supporting Material:	--

4.2.2.5.10 [BSW05038] Get MTS Reception Status

Initiator:	BMW
Date:	20.08.2004
Short Description:	Get the reception status of a Media Access Test Symbol.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to get the reception status (according to table 9-1 of [FR_PROT_SPEC]) of a Media Access Test Symbol.
Rationale:	The sending and the resulting reception status of a Media Access Test Symbol are used to decide whether the Bus Guardian is working correctly in the Symbol Window.
Use Case:	--
Dependencies:	[BSW05111] Get MTS Reception Status [BSW05035] MTS Sending [BSW05107] MTS Sending
Conflicts:	--
Supporting Material:	--

4.2.2.5.11 [BSW05039] Set FlexRay Transceiver Operation Mode

Initiator:	BMW
Date:	20.08.2004
Short Description:	Set the operation mode of a specific FlexRay Transceiver device.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to set the operation mode of a specific FlexRay Transceiver device. According to [FR_EPL_SPEC], at least these operation modes shall be supported: • BD_Normal • BD_Standby • BD_Receive_only • BD_Sleep
Rationale:	Provide ComM with a possibility to set the operation mode of a specific FlexRay Transceiver device.
Use Case:	--
Dependencies:	[BSW05166] Set FlexRay Transceiver Operation Mode [BSW05162] Set Cluster-wide FlexRay Transceiver Operation Mode [BSW05157] Get FlexRay Transceiver Operation Mode
Conflicts:	--
Supporting Material:	[FR_EPL_SPEC]

4.2.2.5.12 [BSW05162] Set Cluster-wide FlexRay Transceiver Operation Mode

Initiator:	BMW
Date:	20.08.2004
Short Description:	Set the operation mode of all FlexRay Transceiver devices connected to a specific FlexRay Cluster.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to set the operation mode of all FlexRay Transceiver devices connected to a specific FlexRay Cluster. According to [FR_EPL_SPEC], at least these operation modes shall be supported: • BD_Normal • BD_Standby • BD_Receive_only • BD_Sleep
Rationale:	Provide ComM with a possibility to set the operation mode of all FlexRay Transceivers connected to a specific Cluster.
Use Case:	--
Dependencies:	[BSW05166] Set FlexRay Transceiver Operation Mode [BSW05039] Set FlexRay Transceiver Operation Mode [BSW05157] Get FlexRay Transceiver Operation Mode
Conflicts:	--
Supporting Material:	[FR_EPL_SPEC]

4.2.2.5.13 [BSW05157] Get FlexRay Transceiver Operation Mode

Initiator:	VW
Date:	24.01.2006
Short Description:	Get the operation mode of a specific FlexRay Transceiver device.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to get the operation mode of a specific FlexRay Transceiver device. According to [FR_EPL_SPEC], at least these operation modes shall be supported: • BD_Normal • BD_Standby • BD_Receive_only • BD_Sleep
Rationale:	Provide ComM with a possibility to get the operation mode of a specific FlexRay Transceiver device.
Use Case:	--
Dependencies:	[BSW05167] Get FlexRay Transceiver Operation Mode [BSW05039] Set FlexRay Transceiver Operation Mode [BSW05162] Set Cluster-wide FlexRay Transceiver Operation Mode
Conflicts:	--
Supporting Material:	[FR_EPL_SPEC]

4.2.2.5.14 [BSW05174] Interrupt Handling

Initiator:	BMW
Date:	01.02.2006
Short Description:	The FlexRay Interface shall provide services to handle interrupts of a FlexRay Communication Controller.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide services to handle interrupts of a FlexRay Communication Controller. At least the following functionality shall be provided: • get interrupt source • service interrupt • enable interrupt • disable interrupt • acknowledge interrupt.
Rationale:	The FlexRay Interface might use an alarm (absolute timer interrupt) of a FlexRay Communication Controller to run synchronously to the FlexRay global time. This interrupt has to be serviced.
Use Case:	--
Dependencies:	[BSW05125] Interrupt Handling
Conflicts:	--
Supporting Material:	--

4.2.2.6 Normal Operation

4.2.2.6.1 [BSW05170] Receive PDU

Initiator:	BMW
Date:	29.01.2006
Short Description:	Reception of PDUs via the FlexRay communication system.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall retrieve PDUs received via the FlexRay communication system and indicate the reception to the appropriate higher-layer BSW module, allowing this module to retrieve the PDU data. Depending on static configuration of the FlexRay Interface, it might be possible that one FlexRay Frame contains more than one PDU. In this case, the FlexRay Interface shall disassemble the FlexRay Frame and indicate the reception of each PDU to the appropriate higher BSW module. If the use of PDU-based update/valid information has been configured for a FlexRay Frame, the reception indication shall consider this information.
Rationale:	As explained in the general requirement, the FlexRay Interface API shall be PDU-based. The access via the PDU is the only possibility to access the FlexRay data.
Use Case:	An application needs to receive information from the FlexRay network.
Dependencies:	[BSW05004] PDU-Based API [BSW05010] Unique PDU-ID [BSW05126] PDU Update/Valid Information
Conflicts:	--
Supporting Material:	--

4.2.2.6.2 [BSW05027] Transmit PDU

Initiator:	BMW
Date:	29.07.2004
Short Description:	Transmission of PDUs via the FlexRay communication system.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to transmit a PDU via the FlexRay communication system Depending on static configuration of the FlexRay Interface, it might be possible that one FlexRay Frame contains more than one PDU. In this case, the FlexRay Interface shall assemble the FlexRay Frame of the corresponding PDUs before the Frame is transmitted. If the use of PDU-based update/valid information has been configured for a FlexRay Frame, the FlexRay Interface shall correctly generate this information.
Rationale:	As explained in the general requirement the FlexRay Interface API shall be PDU-based. The access via the PDU is the only possibility to access the FlexRay data.
Use Case:	An applications needs to send information over the FlexRay network.
Dependencies:	[BSW05004] PDU-Based API [BSW05010] Unique PDU-ID [BSW05126] PDU Update/Valid Information

Conflicts:	--
Supporting Material:	--

4.2.2.6.3 [BSW05171] Provide PDU Transmit Confirmation

Initiator:	BMW
Date:	30.07.2004
Short Description:	Provide the transmit status of a PDU.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall be configurable to provide the appropriate higher-layer BSW module with a transmit confirmation for a PDU this BSW module has requested to be sent. At system configuration time it shall be configurable per PDU whether a transmit confirmation shall be provided.
Rationale:	In the dynamic segment, the transmission of a Frame is not guaranteed. Therefore, the application needs information whether a PDU has been sent.
Use Case:	--
Dependencies:	[BSW05004] PDU-Based API
Conflicts:	--
Supporting Material:	--

4.2.2.7 Shutdown Operation

4.2.2.7.1 [BSW05016] Abortion of a FlexRay CC Communication

Initiator:	BMW
Date:	29.07.2004
Short Description:	Immediate abortion of the communication of a specific FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to immediately abort the communication of a specific FlexRay CC by setting this FlexRay CC into the "HALT" state. Thus, the FlexRay CC will stop participating in the communication and lose synchronicity with the FlexRay global time.
Rationale:	--
Use Case:	Error confinement
Dependencies:	[BSW05114] Abortion of FlexRay CC Communication [BSW05113] Abortion of a FlexRay Cluster Communication
Conflicts:	--
Supporting Material:	--

4.2.2.7.2 [BSW05113] Abortion of a FlexRay Cluster Communication

Initiator:	BMW
Date:	02.02.2005
Short Description:	Immediate abortion of the communication of a specific FlexRay Cluster.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to immediately abort the communication of a specific FlexRay Cluster by setting all FlexRay CCs connected to this FlexRay Cluster into the "HALT" state. Thus, these

	FlexRay CCs will stop participating in the communication and lose synchronicity with the FlexRay global time.
Rationale:	--
Use Case:	Error confinement
Dependencies:	[BSW05016] Abortion of a FlexRay CC Communication
Conflicts:	--
Supporting Material:	--

4.2.2.7.3 [BSW05063] Halt of a FlexRay CC Communication

Initiator:	BMW
Date:	29.07.2004
Short Description:	Halting of the communication of a specific FlexRay CC at the end of the current Communication Cycle.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to halt the communication of a specific FlexRay CC at the end of the current Communication Cycle by setting this FlexRay CC into the "HALT" state at the end of the current Communication Cycle. Thus, the FlexRay CC will stop participating in the communication and lose synchronicity with the FlexRay global time.
Rationale:	--
Use Case:	Error confinement
Dependencies:	[BSW05115] Halt of FlexRay CC Communication [BSW05016] Abortion of a FlexRay CC Communication
Conflicts:	--
Supporting Material:	[FR_PROT_SPEC]

4.2.2.7.4 [BSW05102] Halt of a FlexRay Cluster Communication

Initiator:	BMW
Date:	02.02.2005
Short Description:	Halting of the communication of a specific FlexRay Cluster at the end of the current Communication Cycle.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide a software interface to halt the communication of a specific FlexRay Cluster at the end of the current Communication Cycle by setting all FlexRay CC connected to this FlexRay Cluster into the "HALT" state at the end of the current Communication Cycle. Thus, these FlexRay CC will stop participating in the communication and lose synchronicity with the FlexRay global time.
Rationale:	--
Use Case:	Error confinement
Dependencies:	[BSW05063] Halt of a FlexRay CC Communication
Conflicts:	--
Supporting Material:	--

4.2.2.8 Fault Operation

4.2.2.8.1 [BSW05175] Provide Error Information

Initiator:	BMW
Date:	31.01.2006
Short Description:	Provide error-related information.
Type:	New
Importance:	High
Description:	The FlexRay Interface shall provide the DEM with error-related information that is only known to the FlexRay Interface.
Rationale:	Some error-related information is only known to the FlexRay Interface.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.2.3 Non-Functional Requirements (Qualities)

4.2.3.1 Abstraction Requirements

4.2.3.1.1 [BSW05006] Abstraction of FlexRay-Specific Features

Initiator:	BMW
Date:	23.07.2004
Short Description:	Abstraction from the specific features of the FlexRay communication system.
Type:	New
Importance:	High
Description:	For data reception and sending, the FlexRay Interface shall provide to the upper software layers an abstraction from the specific features of the FlexRay communication system as well as a PDU-based API.
Rationale:	All bus interfaces supported by the AUTOSAR BSW (e.g. CAN, FlexRay, LIN) should provide the same API semantics/signature.
Use Case:	Several bus interfaces (e.g. CAN, FlexRay, and LIN) on one ECU.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.2.3.2 Resource Usage

4.2.3.2.1 [BSW05009] Local Memory Space Usage

Initiator:	BMW
Date:	23.07.2004
Short Description:	Local Memory Space Usage
Type:	New
Importance:	High
Description:	The FlexRay Interface shall allocate the needed memory space only once for a PDU sent multiple times in the FlexRay matrix.
Rationale:	The use of only one local memory space for each PDU, instead of one local memory space for each occurrence of the PDU in the FlexRay matrix,

	implicates a much better performance.
Use Case:	--
Dependencies:	[BSW05004] PDU-Based API
Conflicts:	--
Supporting Material:	--

4.2.3.3 Configuration

4.2.3.3.1 [BSW05056] Configuration of the FlexRay Interface at System Configuration Time

Initiator:	BMW
Date:	19.10.2004
Short Description:	Configuration at system configuration time.
Type:	New
Importance:	High
Description:	The FlexRay Interface configuration shall be done at system configuration time, i.e. the configuration parameters have to be defined before the execution of the FlexRay Interface code (even prior to executing the initialization code).
Rationale:	The purpose of the FlexRay Interface is to execute data processing that has been configured at system configuration time, i.e. before the execution of the FlexRay Interface code. It shall not decide or calculate during runtime any communication parameters like the assignment of Slots to ECU or the packaging of PDUs into FlexRay Frames.
Use Case:	--
Dependencies:	[BSW05042] Switch Configuration during Runtime
Conflicts:	--
Supporting Material:	--

4.3 FlexRay Driver

4.3.1 Functional Overview

The FlexRay Driver module shall offer to the FlexRay Interface Module, the user of the API, a uniform interface to access a number of FlexRay Communication Controllers, usually of the same type. The FlexRay Driver is a software layer which maps abstract functional requests to sequences of CC specific hardware accesses. The hardware implementation of a CC will be hidden from the FlexRay Interface; e. g. the number of transmit/receive buffers will not be visible outside a FlexRay Driver.

4.3.2 Functional Requirements

4.3.2.1 General Requirements

4.3.2.1.1 [BSW05064] Abstraction of FlexRay CC-specific Implementation

Initiator:	BMW
Date:	16.12.2004
Short Description:	Abstraction from the FlexRay CC-specific implementation.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall provide an abstraction from the FlexRay CC-specific implementation as well as a generic FlexRay Driver API to the upper software layer (FlexRay Interface).
Rationale:	All FlexRay Drivers should provide the same API semantics/signature. Upper layers should not depend on the concrete implementation of specific features (e.g. optional features) of the CC. They need to be hardware-independent.
Use Case:	Several (different) FlexRay CCs (e.g. from different manufacturers) handled by one FlexRay Interface.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.3.2.1.2 [BSW05065] Number of FlexRay CCs per Driver

Initiator:	BMW
Date:	16.12.2004
Short Description:	Communicate with at least four FlexRay CCs of the same type.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall be able to communicate with at least four FlexRay CCs of the same type. The actual number of CCs operated by one FlexRay Driver shall be defined at system configuration time and shall be less or equal to the number of CCs this FlexRay Driver supports.
Rationale:	Future network topologies might demand more than one FlexRay CC on one ECU.
Use Case:	An ECU connected to two FlexRay Clusters.
Dependencies:	[BSW05007] Number of FlexRay CCs per Interface [BSW05097] Number of FlexRay Drivers per FlexRay Interface

Conflicts:	--
Supporting Material:	--

4.3.2.1.3 [BSW05005] Support of Hardware FIFO Mechanism

Initiator:	BMW
Date:	23.07.2004
Short Description:	Support the optional CC hardware FIFO mechanism, if supported by the FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall support the CC hardware FIFO mechanism, if supported by the FlexRay CC, and abstract its usage because this mechanism is optional.
Rationale:	In the future, a FlexRay CC could have a low number of transmit/receive buffers to reduce its cost. Then the high number of different Slot-IDs – with slow communication requirements – in the dynamic segment requires a FIFO on the reception side.
Use Case:	The hardware FIFO could be configured for the dynamic segment and / or the static segment of the FlexRay Communication Cycle.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.3.2.1.4 [BSW05024] Abstraction from CC Buffer Configuration

Initiator:	BMW
Date:	29.07.2004
Short Description:	Abstraction from the FlexRay CC buffer configuration.
Type:	New
Importance:	High
Description:	The software interface of the FlexRay Driver shall be independent of the FlexRay CC transmit/receive buffers' configuration (transmit, receive, filtering …) and their access.
Rationale:	The FlexRay transmit/receive buffer configuration and properties shall not have any influence on the upper-layer software.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.3.2.1.5 [BSW05066] L-SDU-Based API

Initiator:	BMW
Date:	16.12.2004
Short Description:	Provide an L-SDU-based data handling API to all upper layers.
Type:	New
Importance:	High
Description:-	The FlexRay Driver shall provide an L-SDU-based data handling to all upper layers. Based on the unique L-SDU identifier (this is NOT the Frame-ID according to [FR_PROT_SPEC]) and according to rules defined at system

	configuration time, the FlexRay Interface assembles L-SDU for sending or extracts them for reception, resp. Therefore, the FlexRay Driver shall provide an L-SDU-based data handling to all upper layers.
Rationale:	Payload contained in L-SDU is usually transferred into/out of a FlexRay Cell via transmit/receive buffers allocated in the FlexRay Controller.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.3.2.2 Configuration

4.3.2.2.1 [BSW05058] Configuration of FlexRay Driver at System Configuration Time

Initiator:	BMW
Date:	15.09.2004
Short Description:	The configuration of the FlexRay Driver shall be defined at system configuration time.
Type:	New
Importance:	High
Description:	The configuration of the FlexRay Driver shall be defined at system configuration time, i.e. before the execution of the FlexRay Driver code.
Rationale:	The purpose of the FlexRay Driver is to carry out data processing that has been configured at system configuration time, i.e. before the execution of the FlexRay Driver code. It shall not decide or calculate during runtime any communication parameters like the assignment of Slots to ECU or the packaging of PDUs into FlexRay Frames.
Use Case:	--
Dependencies:	[BSW05034] Configuration Modifiable by a Flashing Process
Conflicts:	--
Supporting Material:	--

4.3.2.2.2 [BSW05059] Transmit/Receive Buffer Configuration

Initiator:	BMW
Date:	15.09.2004
Short Description:	Transmit/Receive Buffer Configuration.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall configure the FlexRay CC's transmit/receive buffers according to the configuration of the FlexRay Driver defined at system configuration time.
Rationale:	The FlexRay CC's transmit/receive buffers need to be configured before the FlexRay CC can be used for communication.
Use Case:	--
Dependencies:	[BSW05058] Configuration of FlexRay Driver at System Configuration Time
Conflicts:	--
Supporting Material:	--

4.3.2.3 Initialization

4.3.2.3.1 [BSW05116] Initialization of FlexRay CC

Initiator:	BMW
Date:	02.02.2005
Short Description:	Initialization and configuration of a specific FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall provide a software interface to initialize and configure a specific FlexRay CC. Thereby, the transmit/receive buffers and the low level parameters of the FlexRay CC shall be configured.
Rationale:	--
Use Case:	Initial configuration.
Dependencies:	[BSW05031] Initialization of a FlexRay CC [BSW05011] Initialize Low-Level Parameters [BSW05012] Initialize FlexRay CC Transmit/Receive Buffers
Conflicts:	--
Supporting Material:	--

4.3.2.3.2 [BSW05011] Initialize Low-Level Parameters

Initiator:	BMW
Date:	26.07.2004
Short Description:	Initialization of the FlexRay low-level parameters.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall provide a software interface to initialize the FlexRay low-level parameters. The low-level parameters cover all the configurable parameters defined in the [FR_PROT_SPEC] and also the FlexRay CC-specific parameters or CC-specific services (e.g. concerning interrupts).
Rationale:	The initialization of the low-level parameters is necessary to communicate on the bus with a FlexRay Controller. The set of CC parameters is FlexRay CC-specific.
Use Case:	Initialization of the FlexRay communication system.
Dependencies:	[BSW05116] Initialization of FlexRay CC
Conflicts:	--
Supporting Material:	--

4.3.2.3.3 [BSW05012] Initialize FlexRay CC Transmit/Receive Buffers

Initiator:	BMW
Date:	26.07.2004
Short Description:	Initialize the transmit/receive buffers of a specific FlexRay CC with a default configuration.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall provide a software interface to initialize the transmit/receive buffers of a specific FlexRay CC with a default configuration.
Rationale:	The initialization of the transmit/receive buffers is necessary to transfer data with a FlexRay Controller. The configuration of the transmit/receive buffers

	and filtering mechanisms is FlexRay CC-specific.
Use Case:	--
Dependencies:	[BSW05116] Initialization of FlexRay CC
Conflicts:	--
Supporting Material:	--

4.3.2.4 Start-up Operation

4.3.2.4.1 [BSW05109] Start-up of a FlexRay CC

Initiator:	BMW
Date:	29.07.2004
Short Description:	Start-up of a specific FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall provide a software interface to start-up a specific FlexRay CC.
Rationale:	Start-up of a specific FlexRay Controller after proper configuration.
Use Case:	--
Dependencies:	[BSW05015] Start-up of a FlexRay CC [BSW05114] Abortion of FlexRay CC Communication [BSW05115] Halt of FlexRay CC Communication
Conflicts:	--
Supporting Material:	--

4.3.2.4.2 [BSW05117] Sending of Wake-Up Pattern

Initiator:	DECOMSYS
Date:	02.02.2005
Short Description:	Sending of a wake-up pattern on a specific FlexRay Channel of a specific FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall provide a software interface to send a wake-up pattern on a specific FlexRay Channel of a specific FlexRay CC. The FlexRay Specification allows only a wake-up on one Channel at a time. However, the FlexRay Driver shall support the sending of a wake-up pattern on any one of the two FlexRay Channels. The FlexRay Driver shall observe the restrictions set forth in the FlexRay Protocol Specification concerning reception of a wake-up pattern or a Frame header during own wake-up pattern sending attempts.
Rationale:	Enable waking up a FlexRay Cluster.
Use Case:	--
Dependencies:	[BSW05018] Sending of a Wake-Up Pattern
Conflicts:	
Supporting Material:	[FR_PROT_SPEC]

4.3.2.5 FlexRay Specific Information

This section includes all the FlexRay-specific status information needed by the upper-layer Software e.g. synchronization information, etc...

4.3.2.5.1 [BSW05120] Get FlexRay CC POC Status

Initiator:	DECOMSYS
Date:	02.02.2005
Short Description:	Get the CC POC status of a specific FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall provide a software interface to get the CC POC status of a specific FlexRay CC. All information contained in the vPOC structure as defined in chapter "2.2.1.3 POC status" of the [FR_PROT_SPEC] shall be returned.
Rationale:	Software modules like Mode Manager, Network Management, or DCM might be interested in the FlexRay CC POC status.
Use Case:	--
Dependencies:	[BSW05022] Get FlexRay CC POC Status
Conflicts:	--
Supporting Material:	--

4.3.2.5.2 [BSW05121] Get FlexRay CC Sync State

Initiator:	DECOMSYS
Date:	02.02.2005
Short Description:	Get the sync state of a specific FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall provide a software interface to get the sync state ("controller synchronous"/"controller asynchronous") of a specific FlexRay CC.
Rationale:	In order to be able to communicate via FlexRay, it is crucial that the FlexRay CC be synchronous to the FlexRay Cluster's global time. Software modules like Mode Manager, Network Management, or DCM might be interested in the sync state of a specific FlexRay CC.
Use Case:	A distributed control might only be started once the communication via the FlexRay bus in possible. Therefore, knowledge about the sync state of the FlexRay CC is crucial
Dependencies:	[BSW05023] Get FlexRay CC Sync State
Conflicts:	--
Supporting Material:	--

4.3.2.6 Normal Operation

4.3.2.6.1 [BSW05106] Buffer Reconfiguration in Normal Active Mode

Initiator:	BMW
Date:	26.07.2004
Short Description:	Reconfiguration of a specific transmit/receive buffer of a specific FlexRay

	Communication Controller in normal active mode, if supported by the FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall be able to reconfigure a specific transmit/receive buffer of a specific FlexRay Communication Controller in normal active mode, if supported by the FlexRay CC. The reconfiguration of a transmit/receive buffer of a FlexRay CC in normal active mode allows providing to the FlexRay Interface a higher number of (virtual) transmit/receive buffers than this FlexRay CC actually has. The configuration tool of the FlexRay Driver shall schedule these reconfiguration actions whenever it is necessary before/after the FlexRay Interface accesses a transmit/receive buffer of a FlexRay CC. The reconfiguration of a transmit/receive buffer of a FlexRay CC in normal active mode means that the same buffer is being used for different buffer configurations (i.e. Frame-ID and filter configuration, transmitting/sending, Channel A/B) without leaving normal active mode (i.e. without setting the FlexRay CC into configuration mode). In the dynamic segment, each transmit/receive buffer of a FlexRay CC shall be used maximally once per Cycle. This limits the reconfiguration possibilities and thus restricts the number of transmittable (sent and received) Frames per dynamic segment to the accumulated number (over all CCs on one ECU) of transmit/receive buffers connected to one Cluster. For safety reasons it shall be possible to completely deactivate this feature pre compile time.
Rationale:	The number of transmit/receive buffers affects the HW cost, therefore, the supplier offers low-cost controllers with a small number of transmit/receive buffers. And by using only dedicate transmit/receive buffer configurations, each small application would need a lot of buffer. Therefore, the buffer reconfiguration during normal active mode shall be supported by the FlexRay Driver.
Use Case:	For example, this can be used to send many PDUs with slow communication requirements in the dynamic segment. Non-safety-critical use, in gateways, etc. Another example (in the static segment): PDU with a period of 1.25ms in a 2.5 ms Cycle length scheduled in the Static Slot 5 and 45. The same buffer should be use to send (or receive) the Slot with the Frame-ID 5 and 45 with the buffer reconfiguration in normal active mode.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.3.2.6.2 [BSW05107] MTS Sending

Initiator:	BMW / Freescale
Date:	16.08.2004
Short Description:	Sending of a Media Access Test Symbol.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall provide a software interface to send a Media

	Access Test Symbol (MTS) on a specific FlexRay Channel of a specific FlexRay CC.
Rationale:	The sending and the resulting reception status of a Media Access Test Symbol are used to decide whether the Bus Guardian is working correctly in the Symbol Window.
Use Case:	--
Dependencies:	[BSW05035] MTS Sending [BSW05038] Get MTS Reception Status [BSW05111] Get MTS Reception Status
Conflicts:	--
Supporting Material:	--

4.3.2.6.3 [BSW05111] Get MTS Reception Status

Initiator:	BMW
Date:	20.08.2004
Short Description:	Get the reception status of a Media Access Test Symbol.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall provide a software interface to get the reception status (according to table 9-1 of [FR_PROT_SPEC]) of a Media Access Test Symbol.
Rationale:	The sending and the resulting reception status of a Media Access Test Symbol are used to decide whether the Bus Guardian is working correctly in the Symbol Window.
Use Case:	--
Dependencies:	[BSW05038] Get MTS Reception Status [BSW05035] MTS Sending [BSW05107] MTS Sending
Conflicts:	--
Supporting Material:	--

4.3.2.6.4 [BSW05125] Interrupt Handling

Initiator:	BMW
Date:	16.02.2005
Short Description:	The FlexRay Driver shall provide services to handle interrupts of a FlexRay Communication Controller.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall provide services to handle interrupts of a FlexRay Communication Controller. At least the following functionality shall be provided: • get interrupt source • service interrupt • enable interrupt • disable interrupt • acknowledge interrupt.
Rationale:	Only the FlexRay Driver has knowledge on how to handle (e.g. enable/disable) a CC-specific interrupt.
Use Case:	--
Dependencies:	[BSW05174] Interrupt Handling
Conflicts:	--

Supporting Material:	--

4.3.2.7 Shutdown Operation

4.3.2.7.1 [BSW05114] Abortion of FlexRay CC Communication

Initiator:	BMW
Date:	02.02.2005
Short Description:	Immediate abortion of the communication of a specific FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall provide a software interface to immediately abort of the communication of a specific FlexRay CC by setting this FlexRay CC into the "HALT" state. Thus, the FlexRay CC will stop participating in the communication and lose synchronicity with the FlexRay global time.
Rationale:	--
Use Case:	Error confinement
Dependencies:	[BSW05016] Abortion of a FlexRay CC Communication
Conflicts:	--
Supporting Material:	--

4.3.2.7.2 [BSW05115] Halt of FlexRay CC Communication

Initiator:	BMW
Date:	02.02.2005
Short Description:	Halting of the communication of a specific FlexRay CC at the end of the current Communication Cycle.
Type:	New
Importance:	High
Description:	The FlexRay Driver shall provide a software interface to halt the communication of a specific FlexRay CC at the end of the current Communication Cycle by setting this FlexRay CC into the "HALT" state at the end of the current Communication Cycle. Thus, this FlexRay CC will stop participating in the communication and lose synchronicity with the FlexRay global time.
Rationale:	--
Use Case:	Error confinement
Dependencies:	[BSW05063] Halt of a FlexRay CC Communication
Conflicts:	--
Supporting Material:	--

4.4 Transport Layer FlexRay

4.4.1 Functional Overview

The FlexRay Transport Layer provides support for segmented acknowledged and unacknowledged 1:1 communication as well as segmented unacknowledged 1:n communication using physical and functional addressing.
Basically, this Layer is compatible to ISO-TP for CAN [ISO 15765-2/4] but it is configurable to support additional features (PDUs longer than 4095 byte, acknowledgement with or without retry of 1:1 PDUs, segmentation of 1:n PDUs).

4.4.2 Non-functional requirements

4.4.2.1 [BSW05073] Usage of ISO 15765-2 and ISO 15765-4 Specifications

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	Compliance with ISO 15765-2 and with ISO 15765-4.
Type:	New
Importance:	High
Description:	It shall be possible to configure the AUTOSAR FlexRay Transport Layer at system configuration time to be compliant with the ISO 15765-2 and the ISO 15765-4 specification. It shall be possible to configure this independently for each channel (i.e. for each N-SDU).
Rationale:	Reuse of existing standards for AUTOSAR BSW. The ISO 15765-2 and 15765-4 specifications are the mostly used Transport Layer in automotive area. Although ISO 15765-2 and 15765-4 have several drawbacks from the functional point of view (e.g., no acknowledgement on the last block of data), ISO compliance is an important issue for the sake of interoperability with legacy ECUs. – Thus the AUTOSAR FlexRay Transport Layer shall at least provide an ISO compliant mode that can be enforced at system configuration time via a configuration switch.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	[ISO 15765-2] and [ISO 15765-4] specification.

4.4.2.2 [BSW05074] FlexRay Transport Layer Interfaces

Initiator:	BMW
Date:	20.01.2005
Short Description:	Location of the FlexRay Transport Layer software module.
Type:	New
Importance:	High
Description:	The FlexRay Transport Layer software module shall be located between the PDU Router and the FlexRay Interface for FlexRay PDUs requiring Transport Protocol functionalities. The FlexRay Transport Layer is used by the PDU Router to transmit and receive FlexRay PDUs coming from the Diagnostic Communication Manager (or from a SW-Component).
Rationale:	Design of AUTOSAR Software Architecture.

Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	AUTOSAR Layered Software Architecture [LSA]

4.4.2.3 [BSW05075] Independence of the Network Configuration

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	Independence of the Network Configuration.
Type:	New
Importance:	High
Description:	The FlexRay Transport Layer implementation shall be independent of the network configuration represented by the FlexRay communication matrix. The API just deals with universal identifiers and data units (N-SDU) properties.
Rationale:	Design of AUTOSAR Software Architecture.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	AUTOSAR Layered Software Architecture [LSA]

4.4.2.4 [BSW05123] Configuration Modifiable by a Flashing Process

Initiator:	BMW
Date:	15.02.2005
Short Description:	Modification of configuration data by a flashing process.
Type:	New
Importance:	High
Description:	All configuration data of the FlexRay Transport Layer defined at system configuration time shall be modifiable by a flashing process. This encompasses (amongst other configuration items) all connection-specific parameters and also all global parameters required for the configuration of the FlexRay Transport Layer.
Rationale:	Change of FlexRay Transport Layer parameters without recompiling of the FlexRay Transport Layer software.
Use Case:	--
Dependencies:	[BSW05034] Configuration Modifiable by a Flashing Process
Conflicts:	--
Supporting Material:	--

4.4.2.5 [BSW05076] Multiple Logical FlexRay Transport Layer Channels

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	Support of at least 32 logical FlexRay Transport Layer Channels being used concurrently.
Type:	New
Importance:	High
Description:	The FlexRay Transport Layer shall support at least 32 logical FlexRay

	Transport Layer Channels being used concurrently. The number of Channels shall be configurable at system configuration time.
Rationale:	Enable multiple concurrent FlexRay Transport Layer connections.
Use Case:	The FlexRay Transport Layer configuration may be specific for each Transport Layer connection.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.4.3 Functional Requirements

4.4.3.1 Configuration

4.4.3.1.1 [BSW05077] Unique Identifier of N-SDU

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	Identification of each N-SDU with a unique identifier.
Type:	New
Importance:	High
Description:	The FlexRay Transport Layer shall identify each N-SDU with a unique identifier, so the upper layer can address an N-SDU without any assumption on the addressing mode configuration of the FlexRay Transport Layer. Furthermore, a symbolic name may be assigned for each N-SDU identifier value to simplify usage of the API.
Rationale:	Independence of upper layer with the FlexRay Transport Layer addressing particularities, optimization. The PDU-Router routes all N-SDUs (FlexRay, CAN and LIN) regardless of the addressing mode particularities of the underlying protocols.
Use Case:	FlexRay – CAN TP gateway
Dependencies:	--
Conflicts:	--
Supporting Material:	[ISO 15765-2] and [ISO 15765-4] specification.

4.4.3.1.2 [BSW05079] Transport Connection Properties

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	Individual properties of each N-SDU.
Type:	New
Importance:	High
Description:	The FlexRay Transport Layer configuration shall be defined at system configuration time and shall define the following properties individually for each N-SDU (i.e. for each connection): • Its unique identifier • Associated N-PDU identifiers • Size of the associated N-PDU(s) (equal for all N-PDUs of this N-SDU) • Acknowledge type • Support segmentation of 1 to n communication • Transmit cancellation ON/OFF • Default block size • Default value for STmin

	• Target address • Source address • Connection type (1:1 or 1:n) • Timeout parameters according to ISO 15765-2 • Maximum PDU length: ▪ Option 1: compliant with ISO 15765-2: PDU length < 2^12 byte = 4096 byte, Frame length up to 7 byte payload ▪ Option 2: PDU length < 2^24 byte = 16 MB, Frame length according to available space in the associated N-PDU(s) ▪ Option 3: PDU length < 2^32 byte = 4 GB, Frame length according to available space in the associated N-PDU(s)
Rationale:	At runtime the FlexRay Transport module must have all the information required to manage a Transport Layer connection.
Use Case:	This information can be used at generation time to check the network configuration with a FlexRay Transport Layer point of view.
Dependencies:	[BSW05129] Mismatch of Service Call and Connection Properties
Conflicts:	--
Supporting Material:	[ISO 15765-2] and [ISO 15765-4] specification.

4.4.3.1.3 [BSW05082] Acknowledgement without Retry

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	Support acknowledgement without retry.
Type:	New
Importance:	High
Description:	The FlexRay Transport Layer shall be configurable at system configuration time to support acknowledgement of segmented and unsegmented PDUs. In case of a 1:1 connection, for an unsegmented transfer an acknowledgement Frame shall be sent back to the sender. For a segmented transfer an acknowledgement Frame shall be sent to the sender after the last consecutive Frame. In the case of a 1:n connection, no acknowledgement is supported. This configuration parameter shall be set per channel.
Rationale:	Provide a simple mechanism to confirm the successful reception of a block.
Use Case:	This mechanism provides more secure transmission.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.4.3.1.4 [BSW05083] Acknowledgement with Retry

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	Support acknowledgement with retry.
Type:	New
Importance:	High
Description:	**Alternatively to acknowledgement without retry**, an acknowledgement with retry shall be configurable at system configuration time for segmented and unsegmented PDUs. In case of a 1:1 connection, for an unsegmented transfer an acknowledge-ment Frame shall be sent back to the sender. If it is a negative acknowledge, a retransfer of the PDU shall take place. For segmented PDUs, the acknowledgment shall be done on a "per-block-basis" (BS parameter of ISO

	15765-2). The following retry mechanism shall be implemented: Starting with the erroneous Frame, the rest of the block that has just been transmitted shall be re-transmitted. In the case of a 1:n connection, no acknowledgement is supported. Generally, a maximum number for retries shall be defined. This configuration parameter shall be set per channel.
Rationale:	Comfortable and safe way to transmit PDUs
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	[ISO 15765-2] and [ISO 15765-4] specification.

4.4.3.1.5 [BSW05084] PDU Length

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	Maximum length of PDUs
Type:	New
Importance:	High
Description:	The maximum length of FlexRay Transport Layer PDUs shall be configurable at system configuration time to the following values: • Option 1: compliant with ISO 15765-2: PDU length < 2^12 byte = 4096 byte, Frame length up to 7 byte payload • Option 2: PDU length < 2^24 byte = 16 MB, Frame length according to available space in the associated N-PDU(s) • Option 3: PDU length < 2^32 byte = 4 GB, Frame length according to available space in the associated N-PDU(s)
Rationale:	Support of larger PDUs for Inter-ECU communication.
Use Case:	Sending a picture to a display.
Dependencies:	--
Conflicts:	--
Supporting Material:	[ISO 15765-2] and [ISO 15765-4] specification.

4.4.3.1.6 [BSW05085] Segmented 1:n Connections without Flow Control

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	Support of segmented 1:n connections without flow control.
Type:	New
Importance:	High
Description:	The FlexRay Transport Layer shall support segmented 1:n connections without flow control.
Rationale:	Enabling of segmented 1:n connections.
Use Case:	Transferring the same long text to multiple displays.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.4.3.1.7 [BSW05104] Default Separation Time

Initiator:	DECOMSYS
Date:	20.01.2005
Short Description:	The default separation time (STmin) shall be configured at system configuration time.
Type:	New
Importance:	High
Description:	The default separation time of the FlexRay Transport Layer (i.e., the separation time used if not specified otherwise by higher layers via an API) shall be configurable at system configuration time.
Rationale:	In order to relieve higher layers from explicitly specifying the separation time via an API, a default separation time (which is used if no specific separation time has been defined via an API call) shall be configurable at system configuration time.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.4.3.2 Initialization

4.4.3.2.1 [BSW05088] FlexRay Transport Layer Initialization

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	Interface for initialization.
Type:	New
Importance:	High
Description:	The FlexRay Transport Layer shall implement an interface for initialization in order to initialize all global variables of the FlexRay Transport Layer module and set all Transport Layer connections in a default state (Idle).
Rationale:	Basic functionality.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.4.3.2.2 [BSW05089] FlexRay Transport Layer Availability

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	The FlexRay Transport Layer services shall not be operational before initializing the module.
Type:	New
Importance:	Medium
Description:	Before using the transmission capabilities of the Flex-Ray Transport Layer, it shall be initialized. If this is not the case, the services have to raise an error in development mode.
Rationale:	Basic functionality.
Use Case:	To avoid usage of the module without a complete initialization this could cause the transmission of corrupted Frames.

Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.4.3.3 Normal Operation

4.4.3.3.1 [BSW05090] Support of Optional ISO 15765-2 Service

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	Support of the Change Parameter Service according to ISO 15765-2.
Type:	New
Importance:	High
Description:	The FlexRay Transport Layer shall support per connection (i.e. per N-SDU) the optional ISO 15765-2 service N_ChangeParameter. Thus, it is possible to adapt the STmin to the receiver's needs.
Rationale:	In the case of Inter-ECU communication it could be necessary to adapt the BS and STmin parameter to the individual needs of the receiver.
Use Case:	--
Dependencies:	--
Caveats:	In contrast to ISO 15765-2, the BS parameter is not changeable. This makes no sense in combination with the AUTOSAR buffering concept.
Supporting Material:	[ISO 15765-2] and [ISO 15765-4] specification.

4.4.3.3.2 [BSW05093] Transmit Cancellation

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	Provide a transmit cancellation service to the sender and the receiver.
Type:	New
Importance:	High
Description:	The FlexRay Transport Layer shall provide to the sender a cancellation service of transmission at any point in time and to the receiver (in 1:1 connection only) after each block. It shall be configurable at system configuration time whether this feature is provided by the FlexRay Transport Layer.
Rationale:	Cancellation of pending transmission.
Use Case:	The higher layer can cancel a pending transmission by using this service.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.4.3.3.3 [BSW05095] Bandwidth Control

Initiator:	VW / AUDI
Date:	20.01.2005
Short Description:	Dynamic control of used bandwidth.
Type:	Changed
Importance:	High
Description:	The FlexRay Transport Layer shall support the dynamic bandwidth control mechanism (by means of STmin definable per connection) as described in

	the ISO 15765-2.
Rationale	Adapting the bandwidth control parameters as used in ISO-TP to FlexRay circumstances.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	[ISO 15765-2] specification.

4.4.3.4 Fault Operation

4.4.3.4.1 [BSW05129] Mismatch of Service Call and Connection Properties

Initiator:	VW / AUDI
Date:	15.03.2005
Short Description:	Detection of mismatch of service call and connection properties.
Type:	New
Importance:	High
Description:	The FlexRay Transport Layer shall detect the mismatch of a service call by the upper layer and the corresponding connection properties (e.g. a transmit request which would cause segmentation for a not appropriately configured 1:n connection, e.g. an ISO compliant connection) and shall indicate this to the caller.
Rationale:	The connection specific configuration parameters have impact on the validity range of the parameters passed to the FlexRay Transport Layer API during runtime. Since these ranges cannot be checked at system configuration time, this check has to take place during runtime.
Use Case:	--
Dependencies:	[BSW05079] Transport Connection Properties
Conflicts:	--
Supporting Material:	--

4.5 FlexRay Time Service

4.5.1 Functional Overview

Provides FlexRay time services, synchronization to FlexRay global time, interrupt services based on FlexRay global time, and synchronization of FlexRay Clusters to external time references

4.5.2 Functional Requirements

4.5.2.1 General Requirements

4.5.2.1.1 [BSW05033] Tick Conversion

Initiator:	BMW
Date:	09.08.2004
Short Description:	Conversion between macroticks and nanoseconds.
Type:	New
Importance:	High
Description:	The FlexRay software modules shall provide a software interface to carry out

	a conversion between FlexRay macroticks and nanoseconds in both directions.
Rationale:	The upper-layer SW module does not know the duration of the FlexRay specific macrotick. Therefore, the FlexRay software modules shall provide an API to carry out the conversion.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.5.2.1.2 [BSW05053] Cluster External Clock Synchronization

Initiator:	BMW
Date:	02.06.2004
Short Description:	Apply rate and offset correction terms to a specific FlexRay Cluster.
Type:	New
Importance:	High
Description:	The FlexRay software modules shall provide a software interface to apply rate and offset correction terms (according to chapter 8.6.5 of [FR_PROT_SPEC]) to a specific FlexRay Cluster, i.e. to all FlexRay Communication Controllers connected to this Cluster.
Rationale:	--
Use Case:	An application may need to maintain the synchronicity between two FlexRay Clusters.
Dependencies:	[BSW05156] Controller External Clock Synchronization
Conflicts:	--
Supporting Material:	--

4.5.2.1.3 [BSW05156] Controller External Clock Synchronization

Initiator:	BMW
Date:	16.01.2006
Short Description:	Apply rate and offset correction terms to a specific FlexRay Communication Controller.
Type:	New
Importance:	High
Description:	The FlexRay software modules shall provide a software interface to apply rate and offset correction terms (according to chapter 8.6.5 of [FR_PROT_SPEC]) to a specific FlexRay Communication Controller.
Rationale:	--
Use Case:	An application may need to maintain the synchronicity between two FlexRay Clusters.
Dependencies:	[BSW05053] Cluster External Clock Synchronization
Conflicts:	--
Supporting Material:	--

4.5.2.2 Configuration

4.5.2.2.1 [BSW05044] Set Absolute Timer

Initiator:	BMW
Date:	13.07.2004
Short Description:	Set the absolute timer of a FlexRay Communication Controller.
Type:	New
Importance:	High
Description:	The FlexRay software modules shall provide a software interface to set a FlexRay Communication Controller's absolute timer in order to program a timer interrupt ("alarm") at a defined point in (the FlexRay global) time specified by a pair of cycle time (in units of macroticks) and cycle count.
Rationale:	--
Use Case:	Absolute alarms can be used to synchronize software modules to the global time of a FlexRay node. The scheduler of a time driven operating system can be triggered by an absolute alarm, thereby synchronizing the operating system to the global time of a FlexRay node.
Dependencies:	The interrupt service routine for the alarm has to be registered with an operating system.
Conflicts:	--
Supporting Material:	--

4.5.2.2.2 [BSW05045] Set Relative Timer

Initiator:	BMW
Date:	13.07.2004
Short Description:	Set the relative timer of a FlexRay Communication Controller, if supported by the FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay software modules shall provide a software interface to set a FlexRay Communication Controller's relative timer, if supported by the FlexRay CC, in order to program a timer interrupt ("alarm") after a defined time period (in units of macroticks of the FlexRay global time) relative to the current point in time.
Rationale:	--
Use Case:	A relative alarm can be used to synchronize software modules to the global time of a FlexRay node.
Dependencies:	The interrupt service routine for the alarm has to be registered with an operating system.
Conflicts:	--
Supporting Material:	--

4.5.2.3 Normal Operation

4.5.2.3.1 [BSW05046] Enable Absolute Alarms

Initiator:	BMW
Date:	13.07.2004
Short Description:	Enable the absolute alarm(s) of a FlexRay Communication Controller.

Type:	New
Importance:	High
Description:	The FlexRay software modules shall provide a software interface to enable the absolute alarm(s) of a FlexRay Communication Controller.
Rationale:	--
Use Case:	Higher software layers may want to (re)-enable an absolute alarm.
Dependencies:	The operating system has to provide a service to enable interrupts.
Conflicts:	--
Supporting Material:	--

4.5.2.3.2 [BSW05047] Disable Absolute Alarms

Initiator:	BMW
Date:	13.07.2004
Short Description:	Disable the absolute alarm(s) of a FlexRay Communication Controller.
Type:	New
Importance:	High
Description:	The FlexRay software modules shall provide a software interface to disable the absolute alarm(s) of a FlexRay Communication Controller.
Rationale:	--
Use Case:	Higher software layers may want to disable an absolute alarm temporarily.
Dependencies:	The operating system has to provide a service to disable interrupts.
Conflicts:	--
Supporting Material:	--

4.5.2.3.3 [BSW05048] Acknowledge Absolute Alarms

Initiator:	BMW
Date:	13.07.2004
Short Description:	Acknowledge the absolute alarm(s) of a FlexRay Communication Controller.
Type:	New
Importance:	High
Description:	The FlexRay software modules shall provide a software interface to acknowledge the absolute alarm(s) of a FlexRay Communication Controller, i.e. to clear the interrupt condition.
Rationale:	If in case of level-sensitive interrupts, an alarm is not acknowledged within the ISR (i.e. the interrupt request flag is not being reset) an interrupt service will be requested immediately again after leaving the ISR.
Use Case:	Higher software layers may use this function to reset the absolute alarm condition.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.5.2.3.4 [BSW05049] Enable Relative Alarms

Initiator:	BMW
Date:	13.07.2004
Short Description:	Enable the relative alarm(s) of a FlexRay Communication Controller, if supported by the FlexRay CC.
Type:	New

Importance:	High
Description:	The FlexRay software modules shall provide a software interface to enable the relative alarm(s) of a FlexRay Communication Controller, if supported by the FlexRay CC.
Rationale:	--
Use Case:	Higher software layers may want to (re)-enable a relative alarm.
Dependencies:	The operating system has to provide a service to enable interrupts.
Conflicts:	--
Supporting Material:	--

4.5.2.3.5 [BSW05050] Disable Relative Alarms

Initiator:	BMW
Date:	13.07.2004
Short Description:	Disable the relative alarm(s) of a FlexRay Communication Controller, if supported by the FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay software modules shall provide a software interface to disable the relative alarm(s) of a FlexRay Communication Controller, if supported by the FlexRay CC.
Rationale:	--
Use Case:	Higher software layers may want to disable a relative alarm temporarily.
Dependencies:	The operating system has to provide a service to disable interrupts.
Conflicts:	--
Supporting Material:	--

4.5.2.3.6 [BSW05051] Acknowledge Relative Alarms

Initiator:	BMW
Date:	13.07.2004
Short Description:	Acknowledge the relative alarm(s) of a FlexRay Communication Controller, if supported by the FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay software modules shall provide a software interface to acknowledge the relative alarm(s) of a FlexRay Communication Controller, if supported by the FlexRay CC, i.e. to clear the interrupt condition.
Rationale:	If in case of level-sensitive interrupts, an alarm is not acknowledged within the ISR (i.e. the interrupt request flag is not being reset) an interrupt service will be requested immediately again after leaving the ISR.
Use Case:	Higher software layers may use this function to reset the relative alarm condition.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.5.2.3.7 [BSW05052] Get Cycle Length in Macroticks

Initiator:	BMW
Date:	02.06.2004

Short Description:	Get the FlexRay Cycle length of a specific FlexRay Cluster.
Type:	New
Importance:	High
Description:	The FlexRay software modules shall provide a software interface to get the length of a Communication Cycle (in macroticks) of a specific FlexRay Cluster.
Rationale:	--
Use Case:	This is for example required to set up a (2^n Cycles) repetitive relative alarm with the repetition rate of one Cycle.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.5.2.3.8 [BSW05019] Get FlexRay Global Time

Initiator:	BMW
Date:	29.07.2004
Short Description:	Get the FlexRay global time from a specific FlexRay CC.
Type:	New
Importance:	High
Description:	The FlexRay software modules shall provide a software interface to get the FlexRay global time from a specific FlexRay CC, specified by a pair of cycle time (in units of macroticks) and cycle count.
Rationale:	The FlexRay global time may be used for the synchronization of the upper-layer software. It can also be read by an application running synchronously or asynchronously to the FlexRay global time.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.5.2.4 Fault Operation

4.5.2.4.1 [BSW05071] FlexRay Time Services Access if CC is in Halt

Initiator:	BMW
Date:	26.01.2005
Short Description:	Error on time services if CC is in "HALT" state.
Type:	New
Importance:	High
Description:	If a time services function is called for a specific FlexRay CC after the communication of this FlexRay CC has been halted or aborted (i.e. the FlexRay CC is in "HALT" state), the FlexRay Driver shall raise an error.
Rationale:	Time service functions only make sense when the FlexRay global time is available, thus when the FlexRay CC is not in "HALT" state.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.5.2.4.2 [BSW05072] FlexRay Time Services Access if CC is Out of Sync

Initiator:	BMW
Date:	26.01.2005
Short Description:	Error on time services if CC is not synchronous.
Type:	New
Importance:	High
Description:	If a time services function is called for a specific FlexRay CC which is not synchronous to its Cluster, the FlexRay Driver shall raise an error.
Rationale:	Time service functions only make sense when the FlexRay global time is available, thus when the FlexRay CC is synchronous.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.6 FlexRay Transceiver Driver

4.6.1 Functional Overview

The FlexRay Transceiver Driver is responsible to handle the FlexRay transceivers on an ECU according to the expected state of the bus specific NM.

The FlexRay Transceiver is a hardware device, which mainly transforms the logical 1/0 signals of the FlexRay Communication Controller's ports to the bus compliant electrical levels, currents and timings. Within an automotive environment, there is currently only one physical layer specified for FlexRay.

In addition, the transceivers are often able to detect electrical malfunctions like wiring issues, ground offsets or collisions. Depending on the interface, they are capable of flagging the detected errors by a single port pin or very detailed via SPI.

Some transceivers also support power supply control and wake-up via the bus. A lot of different wake-up/sleep and power supply concepts are available on the market with focus on the best cost-optimized solution for a given task.

Latest developments are so called System Basis Chips (SBC) where not only the FlexRay transceivers but also power-supply control and advanced watchdogs are implemented in one housing and are controlled via one interface (e.g. via SPI).

A typical FlexRay Transceiver device is the TJA1080 device for a 10 MBit FlexRay bus with support for wake-up via the bus.

4.6.1.1 Transceiver Operation Modes

Important transceiver operation modes are:

- **Un-powered**: Neither communication nor wake-up is possible
- **Normal**: Full communication is possible
- **Read Only**: Transmission is inhibited, reception is possible
- **Standby**: No communication is possible but ECU is still powered. In addition, a transition to Sleep is only valid from this mode. Wake-up by bus is possible, too.
- **Sleep**: No communication is possible and ECU may be un-powered. Wake-up by bus is possible, too.

4.6.1.2 Transceiver Error Status

In parallel to the transceiver operation modes, the transceiver provides status information for the bus. The status is only relevant for operation mode Normal:

- **Good**: No error, bus physics work correctly without any drawbacks.
- **Error**: The transceiver has detected a serious error on the bus which does not allow further communication.

4.6.1.3 Transceiver Wake-up Reason

The transceiver driver is able to store the local view on who has requested the wake-up:

- **Remote wake-up event**: A remote wake-up event is the reception of at least two consecutive wake-up symbols via the bus.
- **Local wake-up event**: A pulse on the FlexRay Transceiver device's WAKE pin (if present) has caused the wake-up.
- **Power on wake-up:** The FlexRay Transceiver device has become sufficiently supplied after being not powered.
- **No wake-up**: No wake-up has been detected.

4.6.1.4 Remarks to the FlexRay Transceiver Driver

FlexRay Transceivers are very different in their behavior and supported features. The range starts with very simple FlexRay Transceivers, which are "always on", includes transceivers with support for advanced limp-home-handling and error detection and ends with so called system basis chips (SBC) which internally contain multiple FlexRay Transceivers, watchdog, voltage regulators and more.

The target of this document is to specify interfaces and behavior, which are applicable to most current and future FlexRay Transceivers on the market for nearly all use cases. The target is that the "user" of the transceiver functionality, typically the AUTOSAR CommunicationManager (ComM) and the AUTOSAR ECU State Manager (EcuM), are bus independent and therefore reusable.

It will not be possible to cover all possible combinations of FlexRay Transceivers with all conceivable power concepts within one AUTOSAR implementation.

4.6.1.5 Explicitly uncovered FlexRay Transceiver Functionality

Some FlexRay Transceivers offer additional functionality to improve e.g. ECU self-test or enhanced error detection capability for diagnostics.

ECU self-test and enhanced error detection are not defined within AUTOSAR and requiring such functionality in general will lock out most currently used (and cheap) transceiver devices. Therefore, features like "ground shift detection", "selective wake-up", "slope control" and others are not supported within these requirements. A general and "open" API like IOControl() is not applicable (and accepted) within AUTOSAR due to portability and reuse.

4.6.1.6 System Basis Chip and FlexRay Transceiver Driver

A system basis chip (SBC) contains, beside the FlexRay Transceivers, additional hardware related to power control and safety (e.g. multiple voltage regulators and a watchdog) and even more features (e.g. persistent memory).

In the AUTOSAR concept, a separate manager/driver/handler (in AUTOSAR called: Interface) is responsible for each identified hardware device. Therefore, the additional manager/driver/handler covers the functionality inside a SBC beside the transceiver driver (e.g. Watchdog Manager, non-volatile memory manager, power control driver, etc.). Due to the shared communication access and the (security-related) restrictions within this communication, independent handling of each SBC-sub-functionality will not be possible.
This will lead to the situation that either a SBC could not be used within an AUTOSAR compliant ECU or (the better solution) a specialized manager/driver/handler for the SBC functionality with all APIs of each single domain has to be used.

4.6.2 Functional Requirements

4.6.2.1 Configuration

4.6.2.1.1 [BSW05131] Configuration Data for FlexRay Transceiver

Initiator:	Bosch
Date:	30.08.2005
Short Description:	The transceiver driver package shall include a description file with the basic information needed to configure the driver for a given bus and the supported notifications.
Type:	New
Importance:	Medium
Description:	This file shall clarify whether the following features are supported: • Single and/or multiple instances of a given transceiver device • Maximally supported baud rate of each bus to enable the detection of configuration errors • Wake-up by bus • Control of power supply possible by the transceiver • List of errors • Transceiver control via SPI or port pin • Which notifications are supported • Call context of the notification functions (ISR, polling) to enable detection of necessary data consistency mechanisms during configuration time • Transceiver supports bus error detection (if not, a state query will always return <good>)
Rationale:	Configuration
Use Case:	Basic functionality for transceiver configuration.
Dependencies:	ECU resource template
Conflicts:	--
Supporting Material:	--

4.6.2.1.2 [BSW05132] Support for More than One FlexRay Transceiver

Initiator:	Bosch
Date:	30.08.2005
Short Description:	The FlexRay Transceiver Driver shall support the configuration for more than one transceiver type as well as for more than one Cluster pre compile time.
Type:	New
Importance:	High

Description:	The driver shall be able to support multiple FlexRay Clusters on the ECU. It must be possible to configure the used transceiver device independently for each Cluster. This includes also mixed systems with e.g. two FlexRay Clusters using different bus physics.
Rationale:	Systems with two FlexRay Clusters.
Use Case:	Basic functionality for transceiver configuration.
Dependencies:	ECU resource template
Conflicts:	Transceiver handling depends strongly on the used device. Therefore each transceiver may needs its own implementation within the driver and only known and supported devices could be selected. A general solution for the FlexRay Transceiver Driver for all use cases might not be possible. FlexRay: By default each FlexRay controller is attached to an own Cluster and needs therefore an own transceiver. In some special cases more than one FlexRay controller is attached to the same Cluster to increase the number of buffers.
Supporting Material:	--

4.6.2.1.3 [BSW05133] Configuration of Bus Operation Mode after Initialization for Each FlexRay Transceiver

Initiator:	Bosch
Date:	30.08.2005
Short Description:	The FlexRay Transceiver Driver shall support the independent configuration of the bus operation mode for each supported Cluster.
Type:	New
Importance:	High
Description:	Due to the different startup requirements on a multiple-FlexRay-Cluster-ECU, the FlexRay Transceiver Driver support the independent pre-selection of the bus operation mode to which each FlexRay Transceiver device is set during the driver initialization.
Rationale:	Multi-Cluster systems
Use Case:	Basic functionality for transceiver configuration.
Dependencies:	See needs described for FlexRay Interface, ECU state manager, ComM and NM.
Conflicts:	--
Supporting Material:	--

4.6.2.1.4 [BSW05134] Initialization Sequence for FlexRay Transceiver Driver

Initiator:	Bosch
Date:	30.08.2005
Short Description:	The FlexRay Transceiver Driver shall support the configuration sequence of the AUTOSAR stack.
Type:	New
Importance:	High
Description:	To start the ECU from power-up or reset, a fix sequence of driver and manager initialization is necessary to reach the required startup times and to set the FlexRay stack into working state. The sequence itself depends on a lot of requirements, partly dependent on the FlexRay controller and the power supply concept.
Rationale:	Correct ECU startup behavior

Use Case:	Basic functionality for transceiver configuration.
Dependencies:	NM, ECU state manager, wake-up concepts of ECU state manager and ComManager.
Conflicts:	--
Supporting Material:	--

4.6.2.1.5 [BSW05136] Configuration "Notification for Wake-up by Bus"

Initiator:	Bosch
Date:	30.08.2005
Short Description:	The FlexRay Transceiver Driver shall support the compile time configuration of one notification to a higher layer for change notification for "wake-up by bus" events.
Type:	New
Importance:	High
Description:	One wake-up by bus event notification shall be supported to one higher layer. If a transceiver device does not support "wake-up by bus", this notification is never called for this bus.
Rationale:	Efficient coupling between FlexRay Transceiver Driver and higher layer.
Use Case:	See [BSW05147]
Dependencies:	Higher layer, i.e. one of (bus specific) NM or ECU state manager. [BSW05147]
Conflicts:	--
Supporting Material:	--

4.6.2.2 Initialization

4.6.2.2.1 [BSW05137] Initialize the FlexRay Transceiver Driver

Initiator:	Bosch
Date:	30.08.2005
Short Description:	The FlexRay Transceiver Driver shall provide an API to initialize the driver internally and set then all attached FlexRay Transceivers in their pre-selected operation modes.
Type:	New
Importance:	High
Description:	The FlexRay Transceiver Driver must be initialized during the power-up/reset sequence of the ECU. Depending on the used drivers to control the transceivers (e.g. DIO, SPI), they must be already available and working when the FlexRay Transceiver Driver is initialized. The wake-up reason has to be detected and stored during the execution of the driver initialization, too.
Rationale:	Set FlexRay Transceivers and FlexRay Transceiver Driver in a pre-defined and known state
Use Case:	Basic functionality for transceiver control.
Dependencies:	SPI and DIO driver initialization. [BSW05144] The FlexRay Transceiver Driver setup information must provide the necessary configuration data to enable the generation tool to select the appropriate control mechanism (e.g. SPI, I/O ports) and to guarantee the

	correct allocation of the necessary communication resources and initialization sequences.
Conflicts:	--
Supporting Material:	--

4.6.2.3 Normal Operation

4.6.2.3.1 [BSW05138] FlexRay Transceiver Driver API shall be Synchronous

Initiator:	Bosch
Date:	30.08.2005
Short Description:	The FlexRay Transceiver Driver API shall be synchronous.
Type:	New
Importance:	High
Description:	The FlexRay Transceiver Driver API shall execute the requested action immediately and shall deliver the result state immediately to the caller. This will ease up the implementation of wake-up and sleep concepts within the AUTOSAR BSW stack.
Rationale:	Better usage of transceiver functionality in the complex AUTOSAR BSW environment.
Use Case:	Atomic transition to other operation mode; easier and better abstraction for higher layers like the ECU state manager or ComManager. Improved testability compared to asynchronous handling.
Dependencies:	ECU state manager, NM
Conflicts:	--
Supporting Material:	--

4.6.2.3.2 [BSW05166] Set FlexRay Transceiver Operation Mode

Initiator:	BMW
Date:	02.02.2006
Short Description:	Set the operation mode of a specific FlexRay Transceiver device.
Type:	New
Importance:	High
Description:	The FlexRay Transceiver Driver shall provide a software interface to set the operation mode of a specific FlexRay Transceiver device. According to [FR_EPL_SPEC], at least these operation modes shall be supported: • BD_Normal • BD_Standby • BD_Receive_only • BD_Sleep
Rationale:	Provide ComM with a possibility to set the operation mode of a specific FlexRay Transceiver device.
Use Case:	--
Dependencies:	[BSW05039] Set FlexRay Transceiver Operation Mode [BSW05162] Set Cluster-wide FlexRay Transceiver Operation Mode
Conflicts:	--
Supporting Material:	[FR_EPL_SPEC]

4.6.2.3.3 [BSW05167] Get FlexRay Transceiver Operation Mode

Initiator:	BMW
Date:	02.02.2006
Short Description:	Get the operation mode of a specific FlexRay Transceiver device.
Type:	New
Importance:	High
Description:	The FlexRay Transceiver Driver shall provide a software interface to get the operation mode of a FlexRay Transceiver device. According to [FR_EPL_SPEC], at least these operation modes shall be supported: • BD_Normal • BD_Standby • BD_Receive_only • BD_Sleep
Rationale:	Provide ComM with a possibility to get the operation mode of a specific FlexRay Transceiver device.
Use Case:	--
Dependencies:	[BSW05157] Get FlexRay Transceiver Operation Mode
Conflicts:	--
Supporting Material:	[FR_EPL_SPEC]

4.6.2.3.4 [BSW05144] Get FlexRay Transceiver Wake-up Reason

Initiator:	Bosch
Date:	30.08.2005
Short Description:	Get the wake-up reason of a specific FlexRay Transceiver device.
Type:	New
Importance:	High
Description:	The FlexRay Transceiver Driver shall provide a software interface to get the wake-up reason of a specific FlexRay Transceiver device. The FlexRay Transceiver Driver shall be able to store the local view "who has requested the wake-up: bus or internally". • **Bus**: The bus has caused the wake-up. • **Internally**: The wake-up has been caused by an internal request to the driver. • **Sleep**: The transceiver is in operation mode sleep and no wake-up has been occurred. The wake-up reason should be "sleep" when the operation mode is not Normal and no wake-up has been occurred. When a wake-up has occurred, the API must always return the first detected wake-up reason (e.g. if a wake-up by bus occurs and than nearly at the same time an internal wake-up, the wake-up reason is "bus".). After leaving the operation mode Normal, the wake-up reason shall be set to "sleep" again.
Rationale:	Detection of wake-up reason during development and via diagnostic command. May also be used by the NM or ECU state manager.
Use Case:	--
Dependencies:	[BSW05158] Get FlexRay Transceiver Wake-up Reason
Conflicts:	--
Supporting Material:	--

4.6.2.3.5 [BSW05147] Notification for Wake-up by Bus

Initiator:	Bosch
Date:	30.08.2005
Short Description:	The FlexRay Transceiver Driver shall support a notification to inform higher layers about the wake-up by bus. It must be possible to support more than one bus within the ECU with this notification.
Type:	New
Importance:	High
Description:	The FlexRay Transceiver Driver shall call this notification when the transceiver detects a wake-up by bus. The FlexRay Transceiver Driver is notified by a notification from the underlying SPI or DIO driver in that case. The notification is executed in the context of the caller (may be interrupt context!). Due to the delay from wake-up detection till the start of the necessary actions have a large influence to the startup time of an ECU, this event shall be processed internally and transferred immediately via this notification to the next layer. The call context and the reaction time depend on the call context of the lower layer DIO or SPI. In case of interrupt it is very fast but data consistency issues must be covered in all layers, in case of polling data consistency issues are reduced but reaction time may be to slow.
Rationale:	Support wake-up by FlexRay Transceiver devices.
Use Case:	The FlexRay Transceiver detects a wake-up condition on the bus and shows this to the µC via e.g. a port pin. Further handling depends on current ECU state. Assumed the ECU is halted, the change on the port may terminate the "HALT" statement and let the processor continue its work. The assigned port interrupt will be executed and this handler is called. Now, the FlexRay Transceiver Driver will store the wake-up reason and give the call via this notification to e.g. the NM to let the NM decide how to handle the event. See [BSW05136] Configuration "Notification for Wake-up by Bus" for details, too.
Dependencies:	DIO and SPI driver for notification, one of (bus specific) NM, diagnostics or ECU state manager as client. [BSW05136]
Conflicts:	--
Supporting Material:	--

4.6.2.3.6 [BSW05148] Support for Wake-up During Sleep Transition

Initiator:	Bosch
Date:	30.08.2005
Short Description:	The FlexRay Transceiver Driver shall support situations where a wake-up by bus occurs at the same moment the transition to standby/sleep is executed by the driver.
Type:	New
Importance:	High
Description:	Wake-up by bus is always asynchronous to the internal transition to sleep. In worst case, the wake-up occurs during the transition to sleep. This situation must be covered by the design and explicitly tested for each ECU. The driver shall create a wake-up notification by bus immediately after the

	API to enter the standby/sleep mode has finished. The calling/controlling component (NM or ECU state manager) must be capable to handle the wake-up immediately after requesting the standby/sleep.
Rationale:	Safe wake-up and sleep handling.
Use Case:	All busses with a wake-up by bus are affected.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.6.2.3.7 [BSW05149] Support API to Enable/Disable and Clear Wake-up Events

Initiator:	Bosch
Date:	30.08.2005
Short Description:	The FlexRay Transceiver Driver shall support an API to enable and disable the wake-up notification for each bus separately.
Type:	New
Importance:	High
Description:	To enable higher layers to command the FlexRay Transceiver device safe into its standby and/or sleep state, an additional API to disable and enable the wake-up notification is necessary. If the notification is disabled, driver shall not perform the notification but store the event internally until the notification is enabled again. The notification shall then be processed immediately. It shall be possible to clear a pending wake-up event. If no further wake-up event occurs, no notification shall be performed after enabling the notification again. If a further wake-up event occurs it shall be notified.
Rationale:	Safe wake-up and sleep handling.
Use Case:	All busses with a wake-up by bus are affected.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.6.2.4 Shutdown Operation

4.6.2.4.1 [BSW05150] Safe System Shutdown for FlexRay Transceiver Driver

Initiator:	Bosch
Date:	30.08.2005
Short Description:	The FlexRay Transceiver Driver shall support the AUTOSAR ECU state manager in a way that a safe system startup and shutdown is possible.
Type:	New
Importance:	High
Description:	The ECU state manager is under development in parallel to this specification, therefore this requirement is added to check during implementation and system test for correct interaction between the FlexRay Transceiver Driver and the ECU state manager. In general, for startup the FlexRay Transceivers must not be enabled until the power supply is available and stable to prevent errors on the bus. Also the communication hardware and driver must not be enabled until the transceiver is configured into its normal operation mode.

	For shutdown, the communication must be stopped according to the AUTOSAR NM algorithm, the FlexRay drivers must be stopped and then the transceivers may be set to standby/sleep, too. The correct sequence depends on the wake-up/sleep concept of AUTOSAR.
Rationale:	System startup and shutdown together with ECU state manager.
Use Case:	--
Dependencies:	ECU state manager
Conflicts:	--
Supporting Material:	--

4.6.2.5 Fault Operation

4.6.2.5.1 [BSW05151] FlexRay Transceiver Driver Must Check Transceiver Control

Initiator:	Bosch
Date:	30.08.2005
Short Description:	The FlexRay Transceiver Driver shall check the control communication to the transceiver and the reaction of the transceiver for correctness.
Type:	New
Importance:	Medium
Description:	Depending on the supported transceiver device, the driver shall check the correctness of the executed control communication and the operation mode a transceiver is in.
Rationale:	Diagnostics and trouble shooting
Use Case:	1. Detection of defect or misbehaving transceiver hardware 2. Detection of corrupted SPI communication The check shall only be applied to errors within the transceiver or the transceiver control communication (ports or SPI), i.e. errors caused by malfunction of the µC, SW or a defect transceiver device. "Errors" caused by the "outer world" (e.g. disturbed communication Channels or ground offsets) are not in the scope of this API.
Dependencies:	--
Conflicts:	--
Supporting Material:	--

4.6.2.5.2 [BSW05168] Indicate FlexRay Transceiver Error State

Initiator:	Bosch
Date:	30.08.2005
Short Description:	Indicate the error state of a FlexRay Transceiver device to the DEM.
Type:	New
Importance:	High
Description:	The FlexRay Transceiver Driver shall indicate error states of a FlexRay Transceiver device to the DEM.
Rationale:	The Transceiver error state is necessary for error handling.
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	[FR_EPL_SPEC]

4.6.3 Non-Functional Requirements

The FlexRay Transceiver Driver has strong relations to the AUTOSAR NM and to the AUTOSAR ECU state manager since they control and use this BSW component. The requirements and needs of these two BSW components may have strong influence on this specification and maybe vice versa.

The FlexRay Transceiver Driver itself will use the drivers for ports (DIO) or SPI. Therefore, the FlexRay Transceiver Driver will be a user for their services as it is. If the FlexRay Transceiver Driver uses other drivers for e.g. initialization access, they must be available before the FlexRay Transceiver initialization is executed!

4.6.3.1 Timing Requirements

4.6.3.1.1 [BSW05152] Transceiver-specific Timing Requirements

Initiator:	Bosch
Date:	30.08.2005
Short Description:	The FlexRay Transceiver Driver shall handle the transceiver-specific timing requirements internally.
Type:	New
Importance:	High
Description:	The communication between the µC and the transceiver is performed via ports or SPI or both. If ports are used, applying values in a predefined sequence and with a given timing to the ports are used to communicate and change the hardware operation modes. These sequences and timings must be handled within the FlexRay Transceiver Driver. Small times may be implemented as a wait loop inside the driver. Disadvantages are that this time is lost for the other software and the wait time depends on the used µC and e.g. system clock. Large wait times (e.g. >200µs) may require an asynchronous API of the FlexRay Transceiver Driver. Disadvantage is then that the complete API and usage will be different for such a hardware device.
Rationale:	Correct handling of used transceiver
Use Case:	--
Dependencies:	--
Conflicts:	--
Supporting Material:	--

5 References

5.1 Deliverables of AUTOSAR

[GLOSSARY] Glossary,
https:/svn2.autosar.org/repos2/22_Releases
AUTOSAR_Glossary.pdf

[LSA] Layered Software Architecture,
https:/svn2.autosar.org/repos2/22_Releases
AUTOSAR_LayeredSoftwareArchitecture.pdf

[VFB] Specification of the Virtual Function Bus,
https:/svn2.autosar.org/repos2/22_Releases
AUTOSAR_VirtualFunctionBus.pdf

5.2 Related Standards and Norms

5.2.1 FlexRay

[FR_PROT_SPEC] FlexRay Communications System Protocol Specification Version 2.1 Revision A,
http://www.FlexRay.com

[FR_EPL_SPEC] FlexRay Communications System Electrical Physical Layer Specification Version 2.1 Revision A,
http://www.FlexRay.com

5.2.2 ISO

[ISO_15765_2] ISO 15765-2(2003-11-11), Road vehicles — Diagnostics on Controller Area Networks (CAN) — Part2: Network layer services
http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=33616

[ISO_15765_4] ISO 15765-4(2004-09-07), Road vehicles — Diagnostics on Controller Area Networks (CAN) — Part4: Requirements for emissions-related systems
http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=33619