Document Title	Acceptance Test Specification of Communication on CAN bus
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	632
Document Classification	Standard
Document Status	Final
Part of AUTOSAR Release	1.0.0

Document Change History		
Release	**Changed by**	**Change Description**
1.0.0	AUTOSAR Release Management	Initial release, including test suites on • RS_BRF_01592 – Data Transfer • RS_BRF_01648 – Large Data Type • RS_BRF_01707 – Can Bus Off handling

Disclaimer

Advice for users

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

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

Table of Contents

1 Acronyms and Abbreviations

Abbreviation / Acronym:	***Description:***
AT	Acceptance Test
CAN	Controller Area Network
ECU	Electronic Control Unit
LT	Lower Tester
NM	Network Management
PCO	Point of Control and Observation
PDU	Protocol Data Unit
RfC	Request for Change
Rx	Reception
SUT	System Under Test
SWC	Software Component
TCP	Test Coordination Procedures
Tx	Transmission
UT	Upper Tester

2 Scope

The following test cases are used to verify the correct behavior of all the communication features which are dependent on the CAN bus.

Each test case documents for which releases of the AUTOSAR software specification it can be used:

- When test cases are known to be applicable for a release, this is mentioned in the “AUTOSAR Releases” field of the test case specifications.
  You can find a summary of the applicability of all test cases to the software specification releases in the “AUTOSAR_TR_ATSReleaseApplicability” document.
- When test cases are known to require adaptations (in their configuration requirements or test sequences), this is mentioned in the “Needed Adaptation to other Releases” field of the test case specifications.

3 RS_BRF_01592 - Data Transfer

3.1 General Test Objective and Approach

This Test Specification intends to cover the Data Transfer feature of the Com as described in the AUTOSAR Feature [RS_BRF_01592].

The tests use a test bench environment and Embedded Software Components that use the feature.

This test case document has been established to cover the following features:

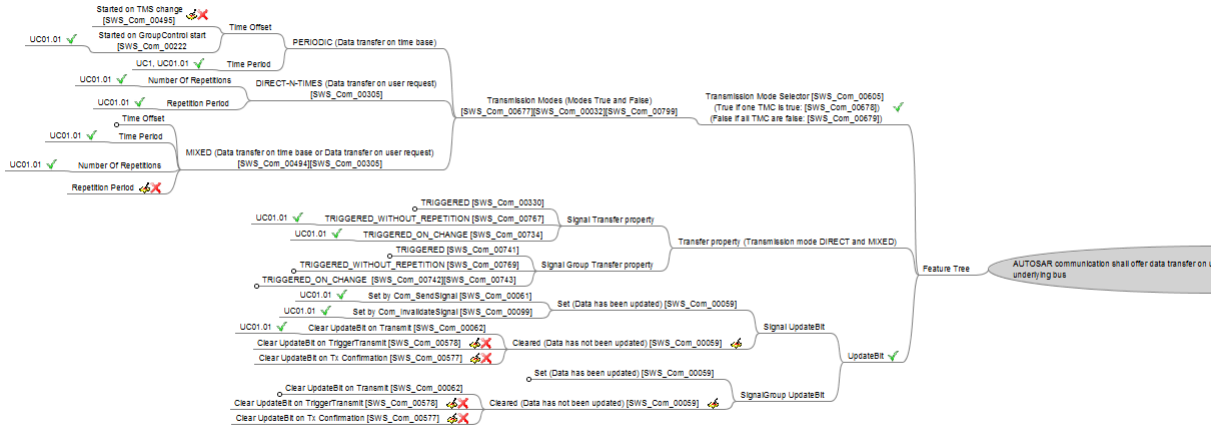

Fig A: Requirement on Data Transfer.

This specification gives the description of required tests environments (test bench, uses case, arxml files) and detailed tests cases for executing tests.

3.1.1 Test System

3.1.1.1 Overview on Architecture

In order to cover the required features / sub-features, the different uses cases are created.

3.1.1.1.1 Use case 01.01: CAN Bus

For this use case, the aim is to test the data transfer on CAN bus, In this architecture, COM focus will be on signals with 1Byte, 2 Bytes and 4 Bytes:

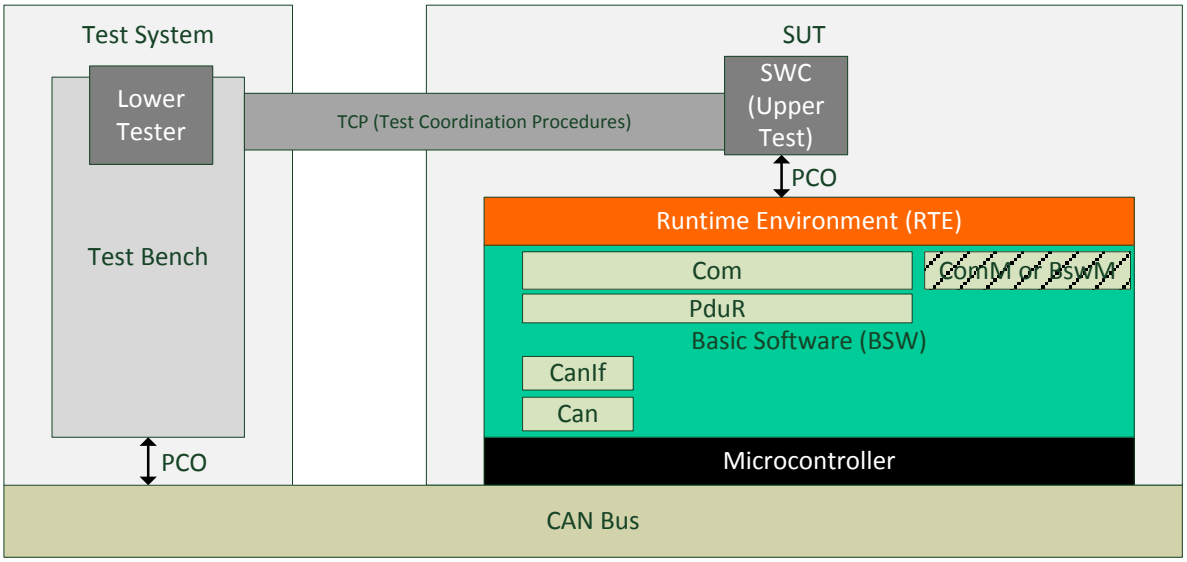

Fig B: Test System Architecture.

The test system architecture consists of Test Bench that executes only test sequencer and gives actions request through Test coordination Procedures to embedded SWC.

3.1.1.2 Specific Requirements

Not Applicable.

3.1.1.3 Test Coordination Requirements

Not Applicable.

3.1.2 Test Configuration

This section describes sets of requirements on configuration.
These sets are later referenced by test cases.
No configuration files are provided. They need to be developed when the test suite is implemented.

3.1.2.1 Required ECU Extract of System Description Files

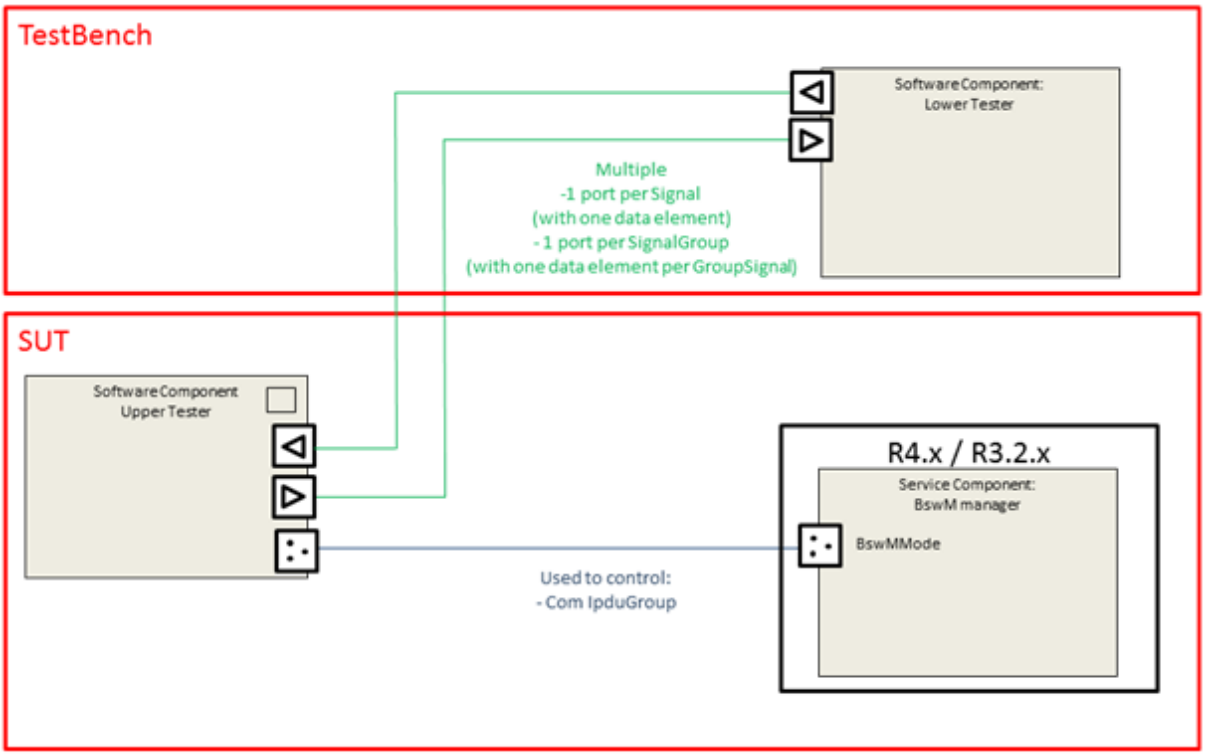

Fig C: SWC Overview.

A Mode-Switch Interface IF_AT_SwC_ActionsBswM must be created. The SWC Upper Tester should trigger BSW action's and BswM read the state through BswMMode Port. BswM shall launch actions according to following table (check 3.3 Test Cases for details):

ModeDeclaration	BswM Actions
IPDU_ACTIVATED	OnEntry: -Start IpduGroup

For the Software Component point of view, for each test case, the communication interfaces are defined as follow:

Port name	Data element type	Dataelement	Mapping	Type
<TestCaseName>_<SignalName>	uint8	<SignalName>	<SignalName>	Signal

<TestCaseName>_<SignalGroupName>	Struct
	{
	uint8: GroupSignal1;
	…
	uint8: GroupSignalx;
	}
		GroupSignal	GroupSignal1->
			<Signal1Name>
			GroupSignal2->
			<Signal2Name>
			<PortName>->
			<SignalGroupName>
				Signal Group

Table 1:

Therefore ports and signals names change according to Test Case Name, but the building rule is the same.

3.1.2.1.1 Use Case 01.01: CAN Bus

The communication database is depicted below:

ComIPduGroup	I-Pdu	SignalGroup	Signal	Tx ECU	Rx ECU
AT_208_IpduGroup	AT_208_Ipdu		AT_208_Sg1	SUT	TestBench
AT_209_IpduGroup	AT_209_Ipdu	AT_209_SgGr1	AT_209_GrSg1	SUT	TestBench
			AT_209_GrSg2		
AT_210_IpduGroup	AT_210_Ipdu		AT_210_Sg1	SUT	TestBench
			AT_210_Sg2		
AT_211_IpduGroup	AT_211_Ipdu	AT_211_SgGr1	AT_211_GrSg1	SUT	TestBench
			AT_211_GrSg2		
		AT_211_SgGr2	AT_211_GrSg3		
			AT_211_GrSg4		
			AT_211_GrSg5		
AT_213_IpduGroup	AT_213_Ipdu		AT_213_Sg1	SUT	TestBench
AT_214_IpduGroup	AT_214_Ipdu	AT_214_SgGr1	AT_214_GrSg1	SUT	TestBench
			AT_214_GrSg2		

Table 2:

3.1.2.2 Required ECU Configuration Description Files

The section describes the common EcuC parameters between test cases that are required by the implementer of the test cases.

No specific configuration requirements for ECU Configuration files as they can be derived from EcuExtract.

3.1.2.3 Required Software Component Description Files

The section describes the SWC-D that are required by the implementer of the test cases.

Refer to Fig C.

3.1.2.4 Mandatory vs. Customizable Parts

Mandatory parameters are listed in Tests Cases (see 3.3 Test Cases).

Customizable parameters are (these values are test case independent):

- ComSignalType (ISignal.networkRepresentationProps.swBaseType), ComSignalLength (baseTypeSize) and ComBitSize (ISignal.length) => must be consistent to associated dataElement
- ComSignalInitValue (ISignal.initValue)
- PduLength (Pdu.length)
- ComBitPosition (ISignalToIPduMapping.startPosition) and ComUpdateBitPosition (ISignalToIPduMapping.updateIndicationBitPosition) values => the location of these elements in the pdu
- CAN frames identifiers

NOTE: ComSignalInitValue and ComSignalDataInvalidValue are specific to test implementer and signal type.

3.1.3 Test Case Design

Not Applicable.

3.2 Re-usable Test Steps

Not Applicable.

3.3 Test Cases

3.3.1 [ATS_COMCAN_00208] Signal on Time Base frame (PERIODIC)

Test Objective	Signal on Time Base frame (PERIODIC)		
ID	ATS_COMCAN_00208	**AUTOSAR Releases**	3.2.1|3.2.2|4.0.3|4.1.1
Affected Modules	Com, PduR, CanIf, Can	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00115 ATR: ATR_ATR_00116		
Trace to R4.1.1 Item	COM: SWS_Com_00059 COM: SWS_Com_00061 COM: SWS_Com_00062 COM: SWS_Com_00099 COM: SWS_Com_00222		
Requirements / Reference to Test Environment	Use Case UC01.01		
Configuration Parameters	ComIpdu(SignalIPdu): AT_208_Ipdu1 (Mapped on CAN Frame => CanTopology) - ComIPduDirection(CommConnectorPort.communicationDirection) = SEND - ComTxModeTrue (IPduTiming.TransmissionModeDeclaration.transmissionModeTrueTiming) -- PERIODIC (CyclicTiming) --- timeOffset > 0 --- timePeriod > 0 (different from timeOffset) - ComTxIPduClearUpdateBit(no upstream template parameter) = Transmit ComSignal(ISignalToPduMapping): Sg1 - updateIndicationBitPosition is configured - ComSignalInitValue(ISignal.initValue) = Sg1_Value_Init - ComSignalDataInvalidValue (SwDataDefProps.invalidValue) = Sg1_Value_Invalid		
Summary	Aim: - Check that send signal and invalidate signal are taken into account in the periodic frame		

	Sequence: 1) Action: Start ComIPduGroup - Result: I-PDU is sent out after Offset Time [SWS_Com_00222] - Result: Frames are sent out periodically (see ComTxModeTimePeriod)" - Result: Signal value is initial value - Result: Signal update bit is 0 2) Action: Update signal - Result: Periodic Time is not changed (value is Period Time) - Result: UpdateBit is set to 1 for the first message after update/invalidation, only. [SWS_Com_00059][SWS_Com_00061][SWS_Com_00062] - Result: After successful transmission the UpdateBit is cleared. - Result: Signal value is changed for all new Tx frame occurrences 3) Action: Invalidate signal - Result: Periodic Time is not changed - Result: UpdateBit is set to 1 for the first message after update/invalidation, only.[SWS_Com_00059][SWS_Com_00099][SWS_Com_00062] -Result: After successful transmission the UpdateBit is cleared. - Result: Signal value is the invalid value for all new Tx frame occurrences	
Needed Adaptation to other Releases	None	
Pre-conditions	Com stack is initialized, but ComIPduGroup are not running	
Main Test Execution		
Test Steps		**Pass Criteria**
Step 1	***SWC:*** Request ModeSwitch (call to BswMModeRequest port) to IPDU_ACTIVATED (start ComIPduGroup AT_208_IpduGroup)	***BUS:*** AT_208_Ipdu is sent out after Offset Time. Next AT_208_Ipdu are sent out every Period Time AT_208_Sg1 value is AT_208_Sg1_Value_Init AT_208_Sg1 update bit is 0
Step 2	***SWC:*** Update signal AT_208_Sg1 (call Rte_Write() API for Port AT_208_Sg1) with AT_208_Sg1_Value_1	***BUS:*** AT_208_Ipdu Periodic Time is not changed (value is Period Time) AT_208_Sg1 UpdateBit is set to 1 in the first send, after that it is 0 AT_208_Sg1 value is now AT_208_Sg1_Value_1
Step 3	***SWC:*** Invalidate signal AT_208_Sg1 (by calling API Rte_Invalidate())	***BUS:*** AT_208_Ipdu Periodic Time is not changed (value is Period Time) AT_208_Sg1 UpdateBit is set to 1 in the first send, after that it is 0 AT_208_Sg1 value is now AT_208_Sg1_Value_Invalid
Post-conditions	Not applicable	

3.3.2 [ATS_COMCAN_00209] SignalGroup on Time Base frame (PERIODIC)

Test Objective	SignalGroup on Time Base frame (PERIODIC)		
ID	ATS_COMCAN_00209	**AUTOSAR Releases**	3.2.1|3.2.2|4.0.3|4.1.1
Affected Modules	Com, PduR, CanIf, Can	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00115 ATR: ATR_ATR_00116		
Trace to R4.1.1 Item	COM: SWS_Com_00059 COM: SWS_Com_00062 COM: SWS_Com_00222 COM: SWS_Com_00286 COM: SWS_Com_00801		
Requirements / Reference to Test Environment	Use Case UC01.01		
Configuration Parameters	ComIpdu(SignalIPdu): AT_209_Ipdu1 (Mapped on CAN Frame => CanTopology) - ComIPduDirection(CommConnectorPort.communicationDirection) = SEND - ComTxModeTrue (IPduTiming.TransmissionModeDeclaration.transmissionModeTrueTiming) -- PERIODIC (CyclicTiming) --- timeOffset > 0 --- timePeriod > 0 (different from timeOffset) - ComTxIPduClearUpdateBit(no upstream template parameter) = Transmit ComSignalGroup(ISignalToPduMapping): SgGr1 - updateIndicationBitPosition is configured - ComGroupSignal(ISignalToPduMapping): GrSg1 -- ComSignalInitValue(ISignal.initValue) = GrSg1_Value_Init -- ComSignalDataInvalidValue (SwDataDefProps.invalidValue) = GrSg1_Value_Invalid - ComGroupSignal(ISignalToPduMapping): GrSg2 -- ComSignalInitValue(ISignal.initValue) = GrSg2_Value_Init -- ComSignalDataInvalidValue (SwDataDefProps.invalidValue) = GrSg2_Value_Invalid		
Summary	Aim: - Check that send signal group and invalidate signal group are taken into account in the periodic frame Sequence: 1) Action: Start ComIPduGroup - Result: I-PDU is sent out after Offset Time [SWS_Com_00222] - Result: Frames are sent out periodically (see ComTxModeTimePeriod) - Result: Group Signal values are initial value 2) Action: Update group signal - Result: Periodic Time is not changed - Result: SignalGroup UpdateBit is set to 1 for the first message after update/invalidation, only. [SWS_Com_00059][SWS_Com_00801][SWS_Com_00062] - Result: After successful transmission the UpdateBit is cleared.		

	- Result: Group Signal values are changed for all new Tx frame occurrences 3) Action: Invalidate signal group - Result: Periodic Time is not changed - Result: SignalGroup UpdateBit is set to 1, only in the first send after step 3. After it is 0. [SWS_Com_00059][SWS_Com_00286][SWS_Com_00062] - Result: All Group Signal values are the invalid values for all new Tx frame occurrences	
Needed Adaptation to other Releases		
Pre-conditions	Com stack is initialized, but ComIPduGroup are not running	
Main Test Execution		
Test Steps		**Pass Criteria**
Step 1	***SWC:*** Request ModeSwitch (call to BswMModeRequest port) to IPDU_ACTIVATED (start ComIPduGroup AT_209_IpduGroup)	***BUS:*** AT_209_Ipdu is sent out after Offset Time Next AT_209_Ipdu are sent out every Period Time AT_209_GrSg1 value is AT_209_GrSg1_Value_Init AT_209_GrSg2 value is AT_209_GrSg2_Value_Init
Step 2	***SWC:*** AT_209_SgGr1.AT_209_GrSg1=AT_209_GrSg1_Value_1 AT_209_SgGr1.AT_209_GrSg2=AT_209_GrSg2_Value_Init Call Rte_Write() for Port AT_209_SgGr1	***BUS:*** AT_209_Ipdu Periodic Time is not changed AT_209_SgGr1 UpdateBit is set to 1 in the first send, after that, it is 0. AT_209_GrSg1 value is now AT_209_GrSg1_Value_1 AT_209_GrSg2 value is kept to AT_209_GrSg2 AT_209_GrSg2_Value_Init
Step 3	***SWC:*** Invalidate signal group AT_209_SgGr1 by calling Rte_Invalidate() API	***BUS:*** AT_209_Ipdu Periodic Time is not changed AT_209_SgGr1 UpdateBit is set to 1 in the first send, after that it is 0 AT_209_GrSg1 value is now AT_209_GrSg1_Value_Invalid AT_209_GrSg2 value is now AT_209_GrSg2_Value_Invalid
Post-conditions	Not Applicable	

3.3.3 [ATS_COMCAN_00210] Signal on User Request frame (DIRECT-N-TIMES)

Test Objective	Signal on User Request frame (DIRECT-N-TIMES)		
ID	ATS_COMCAN_00210	**AUTOSAR Releases**	3.2.1|3.2.2|4.0.3|4.1.1

Affected Modules	Com, PduR, CanIf, Can	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00115 ATR: ATR_ATR_00116		
Trace to R4.1.1 Item	COM: SWS_Com_00305 COM: SWS_Com_00767		
Requirements / Reference to Test Environment	Use Case UC01.01		
Configuration Parameters	ComIpdu(SignalIPdu): AT_210_Ipdu1 (Mapped on CAN Frame => CanTopology) - ComIPduDirection(CommConnectorPort.communicationDirection) = SEND - ComTxModeTrue (IPduTiming.TransmissionModeDeclaration.transmissionModeTrueTiming) -- DIRECT (EventControlledTiming) --- NumberOfRepetitions = 2 --- RepetitionPeriod = 100ms (This value can be changed by the test implementer) ComSignal(ISignalToPduMapping): Sg1 - ComTransferProperty (transferProperty) = TRIGGERED - ComSignalInitValue(ISignal.initValue) = Sg1_Value_Init ComSignal(ISignalToPduMapping): Sg2 - ComTransferProperty (transferProperty) = TRIGGERED_WITHOUT_REPETITION - ComSignalInitValue(ISignal.initValue) = Sg2_Value_Init - ComSignalDataInvalidValue (SwDataDefProps.invalidValue) = Sg2_Value_Invalid		
Summary	Aim: - Check that send signal and invalidate signal are taken into account in the direct frame Sequence: 1) Action: Start ComIPduGroup - Result: I-PDU is not sent out 2) Action: Update signal 1 (triggered) [SWS_Com_00305] - Result: I-PDU is sent two times (interval is Repetition Period) - Result: Signal 1 value is changed for the 2 occurrences of the Tx frame 3) Action: Invalidate signal 2 (Triggered without repetition) [SWS_Com_00767] - Result: I-PDU is sent only one time - Result: Signal 2 value is the invalid value		
Needed Adaptation to other Releases			
Pre-conditions	Com stack is initialized, but ComIPduGroup are not running		
Main Test Execution			
Test Steps		**Pass Criteria**	
Step 1	***SWC:*** Request ModeSwitch (call to BswMModeRequest port) to IPDU_ACTIVATED (start ComIPduGroup AT_210_IpduGroup)	***BUS:*** AT_210_Ipdu is not sent out	

Step 2	***SWC:*** Update signal AT_210_Sg1 by calling Rte_Write() API for Port AT_210_Sg1 (triggered) with AT_210_Sg1_Value_1	***BUS:*** AT_210_Ipdu is sent two times (interval is Repetition Period) First AT_210_Ipdu sent is immediate Signal AT_210_Sg1 value is AT_210_Sg1_Value_1 Signal AT_210_Sg2 value is AT_210_Sg2_Value_Init
Step 3	***SWC:*** Invalidate signal AT_210_Sg2 by calling Rte_Invalidate() API (triggered without repetition)	***BUS:*** AT_210_Ipdu is sent only one time Signal AT_210_Sg1 value AT_210_Sg1_Value_1 Signal AT_210_Sg2 value AT_210_Sg2_Value_Invalid
Post-conditions	Not Applicable	

3.3.4 [ATS_COMCAN_00211] SignalGroup on User Request frame (DIRECT-N-TIMES)

Test Objective	SignalGroup on User Request frame (DIRECT-N-TIMES)		
ID	ATS_COMCAN_00211	**AUTOSAR Releases**	3.2.1|3.2.2|4.0.3|4.1.1
Affected Modules	Com, PduR, CanIf, Can	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00115 ATR: ATR_ATR_00116		
Trace to R4.1.1 Item	COM: SWS_Com_00741 COM: SWS_Com_00769		
Requirements / Reference to Test Environment	Use Case UC01.01		
Configuration Parameters	ComIpdu(SignalIPdu): AT_211_Ipdu1 (Mapped on CAN Frame => CanTopology) - ComIPduDirection(CommConnectorPort.communicationDirection) = SEND - ComTxModeTrue (IPduTiming.TransmissionModeDeclaration.transmissionModeTrueTiming) -- DIRECT (EventControlledTiming) --- NumberOfRepetitions = 2 --- RepetitionPeriod = 100ms (This value can be changed by the test implementer) ComSignalGroup(ISignalToPduMapping): SgGr1 - ComTransferProperty (transferProperty) = TRIGGERED - ComGroupSignal(ISignalToPduMapping): GrSg1/GrSg2 -- ComSignalInitValue(ISignal.initValue) = GrSg1_Value_Init ComSignalGroup(ISignalToPduMapping): SgGr2 - ComTransferProperty (transferProperty) =		

	TRIGGERED_WITHOUT_REPETITION - ComGroupSignal(ISignalToPduMapping): GrSg3/GrSg4/GrSg5 -- ComSignalInitValue(ISignal.initValue) = GrSg2_Value_Init -- ComSignalDataInvalidValue (SwDataDefProps.invalidValue) = GrSg2_Value_Invalid
Summary	Aim: - Check that send signal group and invalidate group signal are taken into account in the direct frame Sequence: 1) Action: Start ComIPduGroup - Result: I-PDU is not sent out 2) Action: Send signal group 1 (triggered) without group signals Initial values - Result: I-PDU is sent two times (interval is Repetition Period) - Result: Group Signal values of Signal Group 1 are the initial values 3a) Action: Invalidate a group signal contained in signal group 2 (Triggered without repetition) [SWS_Com_00769] 3b) Action: Send signal group 2 (Triggered without repetition) [SWS_Com_00769] - Result: I-PDU is sent only one time - Result: All group signal of signal group 2 have invalid value
Needed Adaptation to other Releases	None
Pre-conditions	Com stack is initialized, but ComIPduGroup are not running

Main Test Execution		
Test Steps		**Pass Criteria**
Step 1	***SWC:*** Request ModeSwitch (call to BswMModeRequest port) to IPDU_ACTIVATED (start ComIPduGroup AT_211_IpduGroup)	***BUS:*** AT_211_Ipdu is not sent out
Step 2	***SWC:*** AT_211_SgGr1.AT_211_GrSg1=AT_211_GrSg1_Value_Init AT_211_SgGr1.AT_211_GrSg2=AT_211_GrSg2_Value_Init Call Rte_Write() for Port AT_211_SgGr1	***BUS:*** AT_211_Ipdu is sent two times (interval is Repetition Period) First AT_211_Ipdu sent is immediate AT_211_GrSg1 value is AT_211_GrSg1_Value_Init AT_211_GrSg2 value is AT_211_GrSg2_Value_Init All Group Signals of AT_211_SgGr2 are set to Value_Init
Step 3	***SWC:*** Invalidate group signal AT_211_SgGr2 by calling Rte_Invalidate() API (triggered without repetition)	***BUS:*** AT_211_Ipdu is sent only one time SgGr1 is unchanged: - AT_211_GrSg1 value is AT_211_GrSg1_Value_Init - AT_211_GrSg2 value is AT_211_GrSg2_Value_Init SgGr2 is Invalid: AT_211_GrSg3 value is AT_211_GrSg3_Value_Invalid AT_211_GrSg4 value is AT_211_GrSg4_Value_Invalid

		AT_211_GrSg5 value is AT_211_GrSg5_Value_Invalid
Post-conditions	Not Applicable	

3.3.5 [ATS_COMCAN_00213] Signal on Time Base and User Request frame (MIXED)

Test Objective	Signal on Time Base and User Request frame (MIXED)		
ID	ATS_COMCAN_00213	**AUTOSAR Releases**	3.2.1|3.2.2|4.0.3|4.1.1
Affected Modules	Com, PduR, CanIf, Can	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00115 ATR: ATR_ATR_00116		
Trace to R4.1.1 Item	COM: SWS_Com_00222 COM: SWS_Com_00734		
Requirements / Reference to Test Environment	Use Case UC01.01		
Configuration Parameters	ComIpdu(SignalIPdu): AT_213_Ipdu1 (Mapped on CAN Frame => CanTopology) - ComIPduDirection(CommConnectorPort.communicationDirection) = SEND - ComTxModeTrue (IPduTiming.TransmissionModeDeclaration.transmissionModeTrueTiming) -- MIXED (EventControlledTiming and CyclicTiming) --- NumberOfRepetitions = 1 --- timeOffset != timePeriod (Different from 0) ComSignal(ISignalToPduMapping): Sg1 - ComTransferProperty (transferProperty) = TRIGGERED_ON_CHANGE - ComSignalInitValue(ISignal.initValue) = Sg1_Value_Init		
Summary	Aim: - Check that send signal is taken into account in the mixed frame Sequence: 1) Action: Start ComIPduGroup - Result: I-PDU is sent out after Offset Time [SWS_Com_00222] - Result: Next frames are sent out every Period Time - Result: Signal value is initial value 2) Action: Update signal (triggered on change) with a new value [SWS_Com_00734] - Result: an I-PDU sent out event is added between two I-PDU sent out period - Result: Signal value is the new value 3) Action: Update signal (triggered on change) with the same value [SWS_Com_00734] - Result: I-PDU send out period is not change (event I-PDU was not sent) - Result: Signal value is the same value		
Needed	None.		

Adaptation to other Releases		
Pre-conditions	Com stack is initialized, but ComIPduGroup are not running	
Main Test Execution		
Test Steps		**Pass Criteria**
Step 1	***SWC:*** Request ModeSwitch (call Rte_Switch associated to BswMMode port) to IPDU_ACTIVATED (start ComIPduGroupAT_213_IpduGroup)	***BUS:*** AT_213_Ipdu is sent out after Offset Time. Next AT_213_Ipdu sent out are every Period Time AT_213_Sg1 value is AT_213_Sg1_Value_Init
Step 2	***SWC:*** Update signal AT_213_Sg1 by calling Rte_Write() API for Port AT_213_Sg1 (triggered on change) with AT_213_Sg1_Value_1	***BUS:*** AT_213_Ipdu sent out event is added between two AT_213_Ipdu sent out period Signal AT_213_Sg1 value is AT_213_Sg1_Value_1
Step 3	***SWC:*** Update signal AT_213_Sg1 (call Rte_Write() API for Port AT_213_Sg1) (triggered on change) with the same value AT_213_Sg1_Value_1	***BUS:*** AT_213_Ipdu send out period is not change (event I-PDU was not sent) Signal AT_213_Sg1 value is AT_213_Sg1_Value_1
Post-conditions	Not Applicable	

3.3.6 [ATS_COMCAN_00214] Signal Group on Time Base and User Request frame (MIXED)

Test Objective	Signal Group on Time Base and User Request frame (MIXED)		
ID	ATS_COMCAN_00214	**AUTOSAR Releases**	3.2.1|3.2.2|4.0.3|4.1.1
Affected Modules	Com, PduR, CanIf, Can	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00115 ATR: ATR_ATR_00116		
Trace to R4.1.1 Item	COM: SWS_Com_00222 COM: SWS_Com_00742 COM: SWS_Com_00743		
Requirements / Reference to Test Environment	Use Case UC01.01		
Configuration Parameters	ComIpdu(SignalIPdu): AT_214_Ipdu1 (Mapped on CAN Frame => CanTopology) - ComIPduDirection(CommConnectorPort.communicationDirection) = SEND		

	- ComTxModeTrue (IPduTiming.TransmissionModeDeclaration.transmissionModeTrueTiming) -- MIXED (EventControlledTiming and CyclicTiming) --- NumberOfRepetitions = 1 --- timeOffset != timePeriod (Different from 0) ComSignalGroup(ISignalToPduMapping): SgGr1 - ComTransferProperty (transferProperty) = TRIGGERED_ON_CHANGE - ComGroupSignal(ISignalToPduMapping): GrSg1 -- ComSignalInitValue(ISignal.initValue) = GrSg1_Value_Init - ComGroupSignal(ISignalToPduMapping): GrSg2 -- ComSignalInitValue(ISignal.initValue) = GrSg2_Value_Init
Summary	Aim: - Check that send signal group is taken into account in the mixed frame Sequence: 1) Action: Start ComIPduGroup - Result: I-PDU is sent out after Offset Time [SWS_Com_00222] - Result: Next frames are sent out every Period Time - Result: Group Signal values are initial values 2a) Action: Update group signal (triggered on change) with the initial value 2b) Action: Send signal group (triggered on change) [SWS_Com_00743][SWS_Com_00742] - Result: I-PDU send out period is not changed (event I-PDU was not sent) - Result: Group Signal values are initial values 3a) Action: Update group signal (triggered on change) with a new value 3b) Action: Send signal group (triggered on change) [SWS_Com_00743][SWS_Com_00742] - Result: an I-PDU send out event is added between two I-PDU sent out period - Result: Group Signal value is the new value
Needed Adaptation to other Releases	None
Pre-conditions	Com stack is initialized, but ComIPduGroup are not running

Main Test Execution

Test Steps		**Pass Criteria**
Step 1	***SWC:*** Request ModeSwitch (call Rte_Switch associated to BswMMode port) to IPDU_ACTIVATED (start ComIPduGroupAT_214_IpduGroup)	***BUS:*** AT_214_Ipdu is sent out after Offset Time Next AT_214_Ipdu sent out are every Period Time Group Signal AT_214_GrSg1 value is AT_214_GrSg1_Value_Init Group Signal AT_214_GrSg2 value is AT_214_GrSg2_Value_Init
Step 2	***SWC:*** Call Rte_Write() API for Port AT_214_SgGr1 with AT_214_SgGr1 structure value {AT_214_GrSg1_Value_Init ; AT_214_GrSg2_Value_Init} (Rte will Send group signal AT_214_GrSg1 with AT_214_GrSg1_Value_Init Send group signal AT_214_GrSg2 with	***BUS:*** AT_214_Ipdu send out period is not changed (event I-PDU was not sent) AT_214_GrSg1 value is AT_214_GrSg1_Value_Init AT_214_GrSg1 value is AT_214_GrSg2_Value_Init

	AT_214_GrSg2_Value_Init Send signal group AT_214_SgGr1 (triggered on change))	
Step 3	***SWC:*** Call Rte_Write() API for Port AT_214_SgGr1 with AT_214_SgGr1 structure value {AT_214_GrSg1_Value_1 ; AT_214_GrSg2_Value_Init} (Rte will Send group signal AT_214_GrSg1 with a new value AT_214_GrSg1_Value_1 Send group signal AT_214_GrSg2 with AT_214_GrSg2_Value_Init Send signal group AT_214_SgGr1 (triggered on change))	***BUS:*** AT_214_Ipdu send out event is added between two AT_214_Ipdu sent out period Group signal AT_214_GrSg1 value is AT_214_GrSg1_Value_1 Group signal AT_214_GrSg2 value is AT_214_GrSg2_Value_Init
Post-conditions	Not Applicable	

4 RS_BRF_01648 - Large Data Type

4.1 General Test Objective and Approach

This Test Specification intends to cover the communication transfer of data sizes larger than the maximum transmission unit of the underlying bus as described in the AUTOSAR Feature [RS_BRF_01648].

The tests use a test bench environment and Embedded Software Components that use the feature.

This test case document has been established to cover the following features:

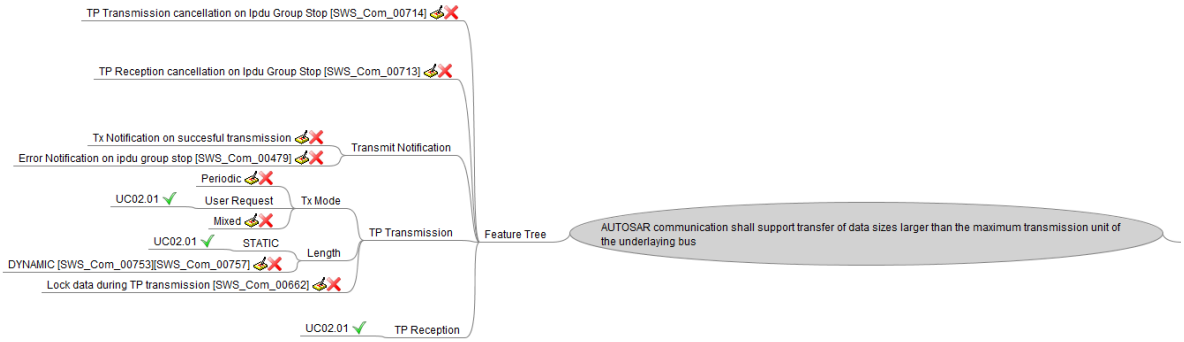

Fig D: Requirement on Large Data Type.

This specification gives the description of required tests environments (test bench, uses case, arxml files) and detailed tests cases for executing tests.

4.1.1 Test System

4.1.1.1 Overview on Architecture

In order to cover the required features / sub-features, the different uses cases are created.

4.1.1.1.1 Use case 02.01: CAN Bus

For this use case, the aim is to test the large data type transfer on CAN bus:

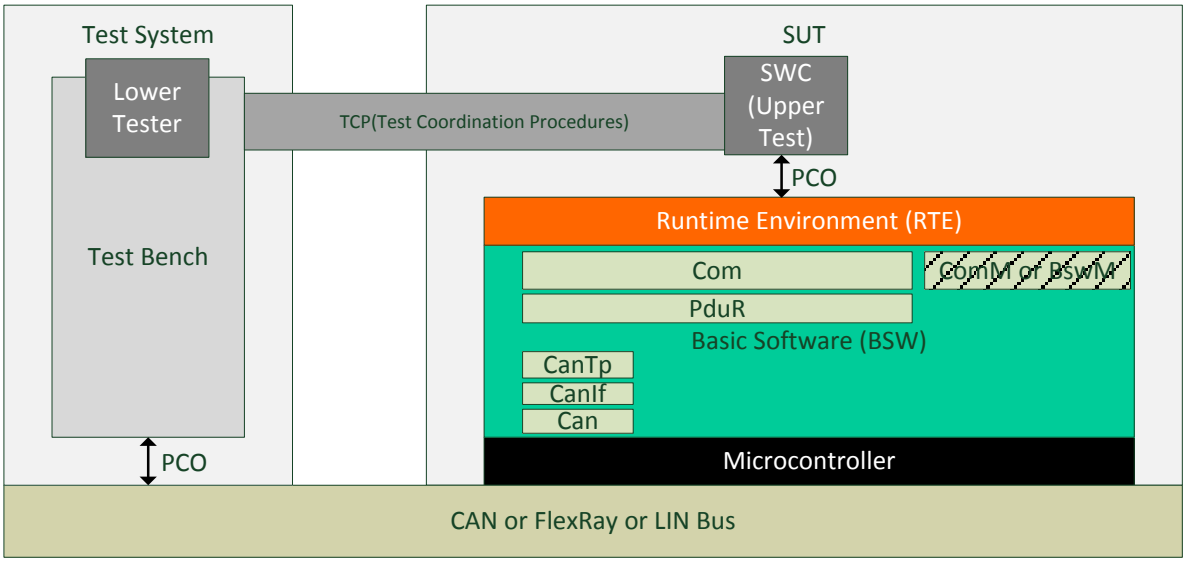

Fig E: Test System Architecture.

The test system architecture consists of Test Bench that executes only test sequencer and gives actions request through Test coordination Procedures to embedded SWC.

4.1.1.2 Specific Requirements

Not Applicable.

4.1.1.3 Test Coordination Requirements

Not Applicable.

4.1.2 Test Configuration

This section describes sets of requirements on configuration.
These sets are later referenced by test cases.
No configuration files are provided. They need to be developed when the test suite is implemented.

4.1.2.1 Required ECU Extract of System Description Files

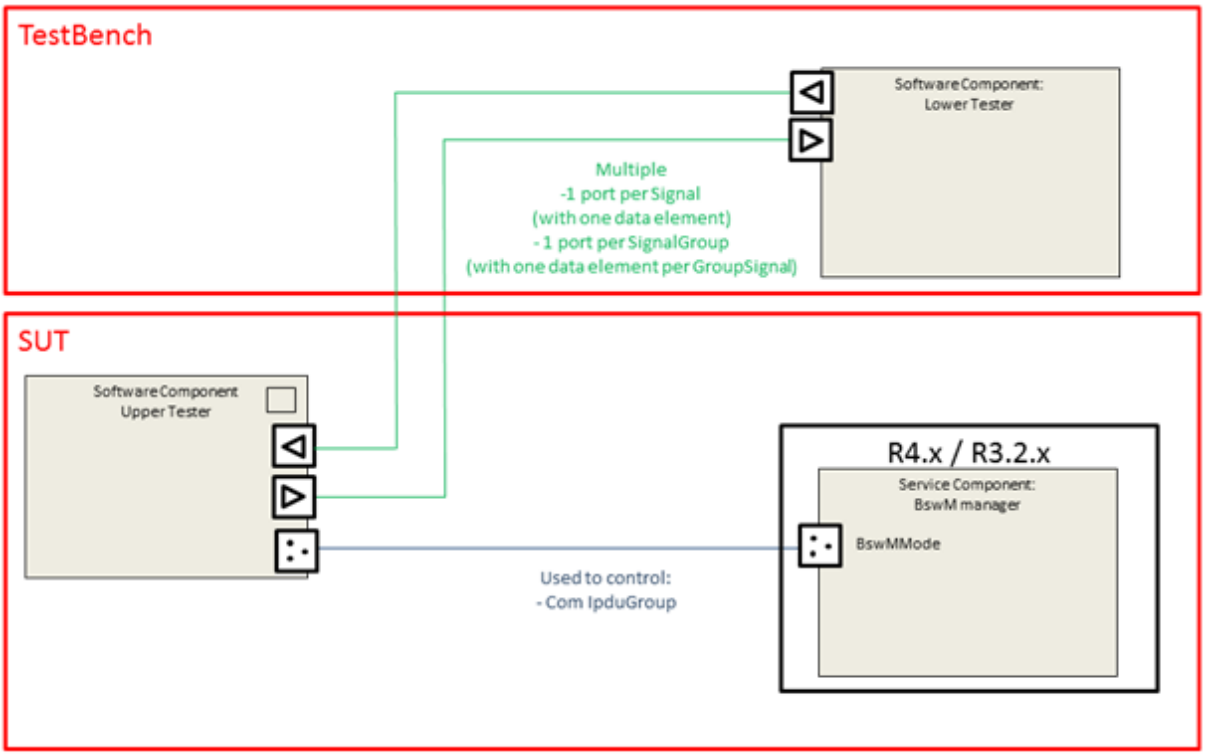

Fig F: SWC Overview on Large Data Type.

A Mode-Switch Interface IF_AT_SwC_ActionsBswM must be created. The SWC Upper Tester is the owner of this state machine and BswM read the state through BswMMode Port. BswM shall launch actions according to following table (check 4.3 Test Cases for details):

ModeDeclaration	BswM Actions
IPDU_ACTIVATED	OnEntry: -Start IpduGroup

For the Software Component point of view, for each test case, the communication interfaces are defined as follow:

Port name	Data element type	Dataelement	Mapping	Type
<TestCaseName>_<signalname>	uint8	<signalname>	<Signalname>	Signal

<TestCaseName>_<signalgroupname>	Struct
	{
	uint8: groupsignal1;
	…
	uint8: groupsignalx;
	}
		Groupsignal	Groupsignal1->
			<signal1name>
			Groupsignal2->
			<signal2name>
			<PortName>->
			<signalgroupname>
				Signal Group

Table 3:

Therefore ports and signals names change according to Test Case number, but the building rule is the same.

4.1.2.1.1 Use Case 02.01: CAN Bus

The communication database is depicted below:

IPduGroup	IPdu	SignalGroup	Signal	Tx ECU	Rx ECU
AT_239_IpduGroup	AT_239_Ipdu		AT_239_Sg1	SUT	TestBench
AT_276_IpduGroup	AT_276_Ipdu		AT_276_Sg1	TestBench	SUT

Table 4:

4.1.2.2 Required ECU Configuration Description Files

The section describes the common EcuC parameters between test cases that are required by the implementer of the test cases.

No specific configuration requirements for ECU Configuration files as they can be derived from EcuExtract.

4.1.2.3 Required Software Component Description Files

The section describes the SWC-D that are required by the implementer of the test cases.

Refer to Fig F.

4.1.2.4 Mandatory vs. Customizable Parts

Mandatory parameters are listed in Tests Cases (see 4.3 Test Cases).

Customizable parameters are (these values are test case independent):

- ComSignalType (ISignal.networkRepresentationProps.swBaseType), ComSignalLength (baseTypeSize) and ComBitSize (ISignal.length) => must be consistent to associated dataElement
- ComSignalInitValue (ISignal.initValue)
- PduLength (Pdu.length)
- ComBitPosition (ISignalToIPduMapping.startPosition) values => the location of these elements in the pdu
- CAN frames identifiers

NOTE: ComSignalInitValue and ComSignalDataInvalidValue are specific to test implementer and signal type.

4.1.3 Test Case Design

Not Applicable.

4.2 Re-usable Test Steps

Not Applicable.

4.3 Test Cases

4.3.1 [ATS_COMCAN_00239] Large Data TP transmission on CAN (>= 8 bytes)

Test Objective	Large Data TP transmission on CAN (>= 8 bytes)		
ID	ATS_COMCAN_00239	**AUTOSAR Releases**	4.0.3|4.1.1
Affected Modules	Com, PduR, CanTp, CanIf, Can	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00118		
Trace to R4.1.1 Item	COM: ECUC_Com_00761		
Requirements / Reference to Test Environment	Use Case UC02.01		
Configuration Parameters	ComIpdu(SignalIPdu): AT_239_Ipdu1 (large I-PDU) - length = 9 (large, greater than a Single Frame) - ComIPduType = TP(TpConfig.TpConnection) - ComIPduDirection(CommConnectorPort.communicationDirection) = SEND - ComTxModeTrue (IPduTiming.TransmissionModeDeclaration.transmissionModeTrueTiming) -- DIRECT(EventControlledTiming) --- NumberOfRepetitions = 1 ComSignal(ISignalToPduMapping): Sg1 - dataElement with queued swImplPolicy - DataSendCompletedEvent mapped on signal transmission (ComNotification is configured) - ComTransferProperty (transferProperty) = TRIGGERED PduRRoutingPath: - Routing path for ComIpdu with PduRSrcBswModuleRef = BswMod_Com - PduRDestPdu with PduRDestBswModuleRef = BswMod_CanTP		
Summary	Aim: - Check that Application layer can initiate a TP transmission greater than or equal to 8 bytes on CAN bus		
Needed Adaptation to other Releases	Configuration: [n/a] Test Steps: [n/a]	`Large data types and TP for regular COM is not possible in R3.x.` This test case is not applicable for R3.x.	
Pre-conditions	Com stack is initialized AT_239_IpduGroup is running		
Main Test Execution			
Test Steps		**Pass Criteria**	

Step 1	***SWC:*** Call Rte_Send() for Port AT_239_Sg1 with AT_239_Sg1_Value_1 (Send Signal AT_239_Sg1 with AT_239_Sg1_Value_1 (this will initiate a TP transmission with 9 bytes))	***BUS:*** First Frame is received Frame length is 8 byte, FF_DL is 9 bytes
Step 2	***BUS:*** Send Flow Control Clear to Send (BlockSize = 0, STMin = 0). 3 bytes length if PADDING is not activated, 8 bytes otherwise.	***BUS:*** One Consecutive Frame is received (4 bytes length if PADDING is not activated) AT_239_Sg1 value is AT_239_Sg1_Value_1
Post-conditions	Not Applicable	

4.3.2 [ATS_COMCAN_00276] Large Data TP reception on CAN (>= 8 bytes)

Test Objective	Large Data TP reception on CAN (>= 8 bytes)		
ID	ATS_COMCAN_00276	**AUTOSAR Releases**	4.0.3|4.1.1
Affected Modules	Com, PduR, CanTp, CanIf, Can	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00118		
Trace to R4.1.1 Item	COM: ECUC_Com_00761		
Requirements / Reference to Test Environment	Use Case UC02.01		
Configuration Parameters	ComIpdu(SignalIPdu): AT_276_Ipdu1 (large I-PDU) - length = 9 (large, greater than a Single Frame) - ComIPduType = TP(TpConfig.TpConnection) - ComIPduDirection(CommConnectorPort.communicationDirection) = RECEIVE ComSignal(ISignalToPduMapping): Sg1 - dataElement with queued swImplPolicy - DataReceivedEvent mapped on signal reception (ComNotification is configured) PduRRoutingPath: - Routing path for ComIpdu with PduRSrcBswModuleRef = BswMod_CanTP - PduRDestPdu with PduRDestBswModuleRef = BswMod_Com		
Summary	Aim: - Check that Application layer can receive a TP Data greater or equal than 8 bytes on CAN bus		
Needed Adaptation to	Configuration: [n/a]	`Large data types and TP for regular COM is`	

other Releases	Test Steps: [n/a]	`not possible in R3.x.` This test case is not applicable for R3.x.
Pre-conditions	Com stack is initialized AT_276_IpduGroup is running	
Main Test Execution		
Test Steps		**Pass Criteria**
Step 1	***Lower Tester:*** Send Signal AT_276_Sg1 with AT_276_Sg1_Value_1 (this will initiate a TP transmission with 9 bytes)	***BUS:*** First Frame is sent Frame length is 8 byte, FF_DL is 9 bytes
Step 2	***BUS:*** Wait reception of Flow Control Clear to Send	***BUS:*** Flow Control Clear to Send is received
Step 3	***Lower Tester:*** Send Consecutive Frame with last data bytes (4 bytes length if PADDING is not activated)	***BUS:*** One Consecutive Frame is received (4 bytes length if PADDING is not activated)
Step 4	***TCP:*** Wait DataReceivedEvent	***SWC:*** DataReceivedEvent is activated
Step 5	***SWC:*** Call Rte_Receive() for AT_276_Sg1	***SWC:*** AT_276_Sg1 value is AT_276_Sg1_Value_1 Return Value of Rte_Receive is RTE_E_OK
Post-conditions	Not Applicable	

5 RS_BRF_01707 – CAN Bus Off handling

5.1 General Test Objective and Approach

The “CAN Bus-Off” feature is tested by setting the conditions which should trigger Bus-Off, transitions between internal states, Bus-Off recovery and then checking whether the transitions are performed correctly, following the right timing constraints.

5.1.1 Test System

5.1.1.1 Overview on Architecture

The basic test setup is depicted in the following figure:

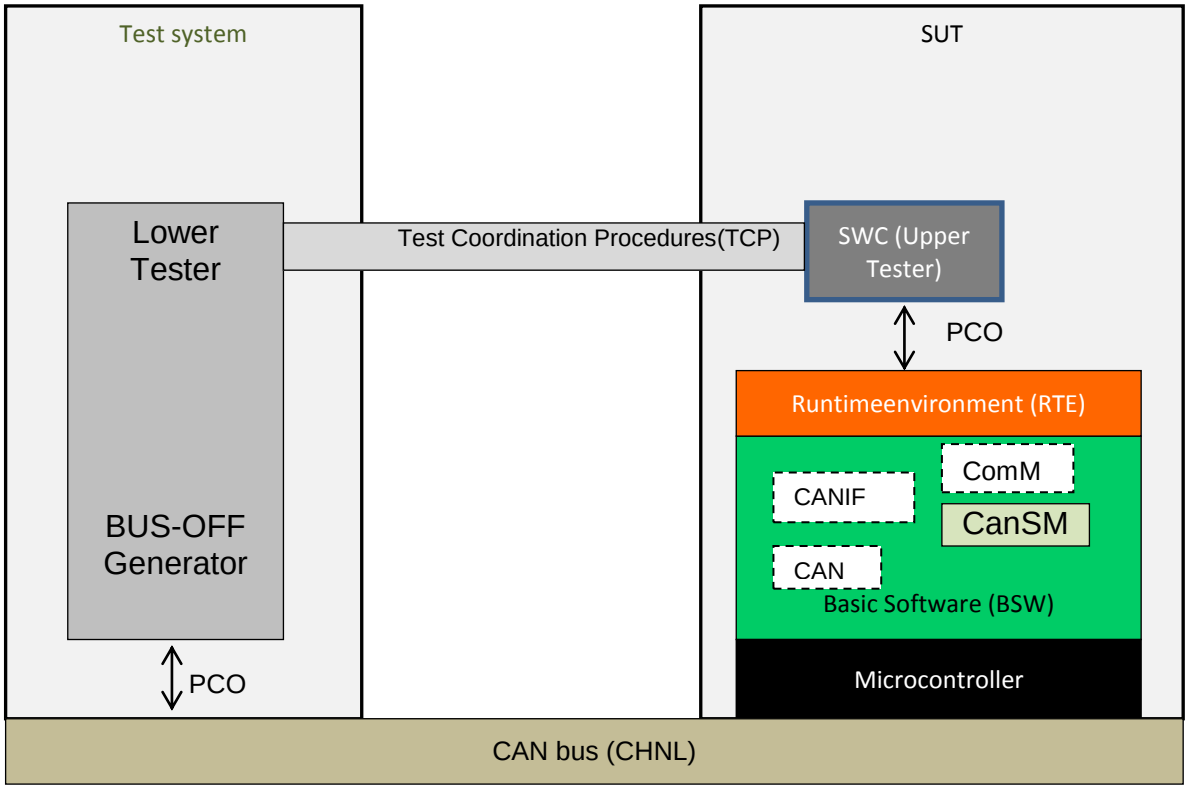

Fig G: Test System Architecture for BUS-OFF.

Figure1: Test Architecture

5.1.1.2 Specific Requirements

Lower Tester simulates Bus-Off with the help of a custom made tool which can generate Bus-Off in CHNL either by

a. Creating a short circuit and then by transmitting a message from the SUT
b. A disturbance in the CAN messages. Sent by the SUT

It is up to the test system designer/ implementer to decide how to generate Bus-Off.

Internal states of SUT shall be verified by checking the invocation of Rte_Mode API. A tester shall be connected to the test ECU CHNL for reading the logged Events.

An internal timer TMR-1 shall be used to verify the timing requirements.

5.1.1.3 Test Coordination Requirements

"Test Coordination Procedures" are needed to collect the test results of the SWC and the Remaining bus simulation at one central place, in order to derive the test verdict. It is up to the test system designer/implementer to define that "central place" and to design/implement the test coordination functionality.

5.1.2 Test Configuration

This section describes sets of requirements on configuration.
These sets are later referenced by test cases.
No configuration files are provided. They need to be developed when the test suite is implemented.

5.1.2.1 Required ECU Extract of System Description Files

The section describes the common EcuC parameters between test cases that are required by the implementer of the test cases.

No specific configuration requirements for ECU Configuration files as they can be derived from EcuExtract.

5.1.2.2 Required ECU Configuration Description Files

Each test case requires some configuration parameters of the SUT to be set with a specific value or within a given range. The test cases below are then provided along with 2 configuration sets (BUSOFF_PS001, BUSOFF_PS002). Each test case description includes a field mentioning the configuration sets which are applicable to that test case among these proposals.

5.1.2.2.1 Parameter Set [BUSOFF_PS001]

SUT configuration parameters	

Parameter name	Value
CanSMMainFunctionTimePeriod	10 msec
CanSMBorCounterL1ToL2	20
CanSMBorTimeL1	2000 msec
CanSMBorTimeL2	2000 msec
CanSMBorTimeTxEnsured	1500 msec
CanSMBorTxConfirmationPolling	false

Table 5:

5.1.2.2.2 Parameter Set [BUSOFF_PS002]

SUT configuration parameters	

Parameter name	Value
CanSMMainFunctionTimePeriod	10 msec
CanSMBorCounterL1ToL2	2
CanSMBorTimeL1	4000 msec
CanSMBorTimeL2	8000 msec

	CanSMBorTimeTxEnsured	1500 msec	
	CanSMBorTxConfirmationPolling	true	

Table 6:

5.1.2.3 Mandatory vs. Customizable Parts

Timing's (CanSMBorTimeL1, CanSMBorTimeL2 and CanSMBorTimeTxEnsured) and counter (CanSMBorCounterL1ToL2) values may be changed to the user's requirements or typical values.

5.1.3 Test Case Design

The test cases check that the SUT follows the state transitions defined in CanSM SWS with the required behavior. States and behavior can only be observed indirectly because of the ICC1 approach of acceptance testing. Thus the behavior on the bus, on the RTE and the diagnostic modules will be observed. State changes can be triggered only from outside of the SUT thus the bus has to be disturbed directly.
The test cases cover

- state change transitions triggered by Bus-Off generation and release
- behavior to the bus
- behavior to the RTE
- behavior to Events (Behaviour related to DEM Event)
- timing behaviour (Behaviour related to Configurable timing parameters – Ref sec 5.1.2.2)

5.2 Re-usable Test Steps

Creation of BUS-OFF scenario can be re-used in all the test cases.

5.3 Test Cases

5.3.1 [ATS_COMCAN_00269] Switching of communication mode during Bus-Off

Test Objective	Switching of communication mode during Bus-Off		
ID	ATS_COMCAN_00269	**AUTOSAR Releases**	4.0.3|4.1.1
Affected Modules	CanSM, ComM	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00101 ATR: ATR_ATR_00104		
Trace to R4.1.1 Item	CANStateManager: SWS_CanSM_00496 CANStateManager: SWS_CanSM_00498 CANStateManager: SWS_CanSM_00521 CANStateManager: SWS_CanSM_00522 COMManager: SWS_ComM_00091 COMManager: SWS_ComM_00778		
Requirements / Reference to Test	Test environment shall be able to generate a Bus-Off in the Test ECU		

Environment		
Configuration Parameters	See [BUSOFF_PS001]	
Summary	Test whether CanSM is able to perform state transition based on Bus-Off notification and release. Bus-Off mode switch and its release is observed by • events retrieved through diagnostic interface • the Rte_Mode API • the CAN bus under test itself	
Needed Adaptation to other Releases	**Needed Adaptation for Release [3.2.2]** Configuration: none Test Steps: low Same requirements on configuration DEM events for bus off have fixed name in R3.2 Same test step sequence	
Pre-conditions	All the communication channels are initialized	
Main Test Execution		
Test Steps		**Pass Criteria**
Step 1	SWC requests ComM to switch to COMM_FULL_COMMUNICATION using Rte_Call_comRequest_RequestComMode	Call to Rte_Call_comRequest_RequestComMode returns E_OK Rte_Mode API is invoked with mode as RTE_MODE_ComMode_COMM_FULL_COMMUNICATION
Step 2	Trigger a communication sequence in the SUT- Example -ComIPduGroup start	Valid frames are observed in the Bus
Step 3	WAIT till (CanSMBorTimeTxEnsured + 1s) time WHILE WAITING, DO nothing Note: This delay is to provide enough time for the SUT to log the Events.	-
Step 4	Check the Events	Event CANSM_E_BUS_OFF is logged with status as DEM_EVENT_STATUS_PASSED [SWS_CanSM_00496][SWS_CanSM_00498]
Step 5	Generate Bus-Off in the Test ECU	-
Step 6	WAIT till (CanSMBorTimeL1 / 2)	-

	WHILE WAITING, DO nothing	
Step 7	Trigger a communication sequence in the SUT- Example -IPDU group start	No valid frames are observed in the Bus
Step 8	Check the Events	Event CANSM_E_BUS_OFF is logged with status as DEM_EVENT_STATUS_PREFAILED [SWS_CanSM_00522]
Step 9	Check whether the SUT is in COMM_SILENT_COMMUNICATION	Rte_Mode API is invoked with mode as RTE_MODE_ComMode_COMM_SILENT_COMMUNICATION [SWS_CanSM_00521][SWS_ComM_00091][SWS_ComM_00778]
Step 10	End generation of Bus-Off in the Test ECU	-
Step 11	WAIT till (CanSMBorTimeL1 + 1s) WHILE WAITING, DO nothing	-
Step 12	Check the Events	Event CANSM_E_BUS_OFF is logged with status as DEM_EVENT_STATUS_PASSED [SWS_CanSM_00498]
Step 13	Check whether the SUT is in COMM_FULL_COMMUNICATION	Rte_Mode API is invoked with mode as RTE_MODE_ComMode_COMM_FULL_COMMUNICATION [SWS_ComM_00091][SWS_ComM_00778]
Step 14	BUS-OFF should be re-covered	Valid frames are observed in the Bus
Post-conditions	-	

5.3.2 [ATS_COMCAN_00270] Retaining FULL com in case of no BusOff with disabled CanSMBorTxConfirmationPolling

Test Objective	Retaining FULL com in case of no BusOff with disabled CanSMBorTxConfirmationPolling		
ID	ATS_COMCAN_00270	**AUTOSAR Releases**	4.0.3|4.1.1
Affected Modules	CanSM	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00101		

Trace to R4.1.1 Item	CANStateManager: SWS_CanSM_00496 CANStateManager: SWS_CanSM_00498	
Requirements / Reference to Test Environment	-	
Configuration Parameters	See [BUSOFF_PS001]	
Summary	This test cases tests the ability of retaining in the FULL communication mode in case of no Bus-Off event when CanSMBorTxConfirmationPolling is disabled Test whether CanSM is able to enter and stay in FULL communication mode in case of no Bus-Off event. The CAN-Bus under test and Events are observed.	
Needed Adaptation to other Releases	**Needed Adaptation for Release [3.2.2]** Configuration: none Test Steps: low Same requirements on configuration DEM events for bus off have fixed name in R3.2 Same test step sequence	
Pre-conditions	All the communication channels are initialized	
Main Test Execution		
Test Steps		**Pass Criteria**
Step 1	SWC requests ComM to switch to COMM_FULL_COMMUNICATION using Rte_Call_comRequest_RequestComMode	Call to Rte_Call_comRequest_RequestComMode returns E_OK Rte_Mode API is invoked with mode as RTE_MODE_ComMode_COMM_FULL_COMMUNICATION
Step 2	Trigger a communication sequence in the SUT- Example -ComIPduGroup start	Valid frames are observed in the Bus Note: To ensure that the channel is in COMM_FULL_COMMUNICATION
Step 3	WAIT till (CanSMBorTimeTxEnsured + 1s) time WHILE WAITING, DO nothing	-
Step 4	Check the Events	Event CANSM_E_BUS_OFF is logged with status as DEM_EVENT_STATUS_PASSED [SWS_CanSM_00496][SWS_CanSM_00498]
Step 5	WAIT for 1 sec WHILE WAITING, DO: Check the invocation of Rte_Mode API	Rte_Mode API is not invoked

		Note: This ensures that no mode switches are observed during the test cycle.
Post-conditions	-	

5.3.3 [ATS_COMCAN_00271] Retaining FULL com in case of no BusOff with enabled CanSMBorTxConfirmationPolling

Test Objective	Retaining FULL com in case of no BusOff with enabled CanSMBorTxConfirmationPolling		
ID	ATS_COMCAN_00271	**AUTOSAR Releases**	4.0.3|4.1.1
Affected Modules	CanSM	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00101		
Trace to R4.1.1 Item	CANStateManager: SWS_CanSM_00497 CANStateManager: SWS_CanSM_00498		
Requirements / Reference to Test Environment	-		
Configuration Parameters	See [BUSOFF_PS002]		
Summary	This test case tests the ability of retaining in the FULL communication mode in case of no Bus-Off event when CanSMBorTxConfirmationPolling is enabled Test whether CanSM is able to enter and stay in FULL communication mode in case of no Bus-Off event. The CAN-Bus under test and Events are observed.		
Needed Adaptation to other Releases	**Needed Adaptation for Release [3.2.2]** Configuration: none Same requirements on configuration Test Steps: low DEM events for bus off have fixed name in R3.2 Same test step sequence		
Pre-conditions	All the communication channels in SUT are initialized		
Main Test Execution			
Test Steps			**Pass Criteria**
Step 1	SWC requests ComM to switch to COMM_FULL_COMMUNICATION using Rte_Call_comRequest_RequestComMode		Call to Rte_Call_comRequest_RequestComMode returns E_OK

		Rte_Mode API is invoked with mode as RTE_MODE_ComMode_COMM_FULL_COMMUNICATION
Step 2	Trigger a communication sequence in the SUT- Example -ComIPduGroup start	Valid frames are observed in the Bus Note: To ensure that the channel is in COMM_FULL_COMMUNICATION
Step 3	Check the Events	Event CANSM_E_BUS_OFF is logged with status as DEM_EVENT_STATUS_PASSED [SWS_CanSM_00497][SWS_CanSM_00498]
Step 4	WAIT for 1 sec WHILE WAITING, DO: Check the invocation of Rte_Mode API	Rte_Mode API is not invoked Note: This ensures that no mode switches are observed during the test cycle.
Post-conditions	-	

5.3.4 [ATS_COMCAN_00272] Behavior of SUT during short recovery time

Test Objective	Behavior of SUT during short recovery time		
ID	ATS_COMCAN_00272	**AUTOSAR Releases**	4.0.3|4.1.1
Affected Modules	CanSM	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00101 ATR: ATR_ATR_00104		
Trace to R4.1.1 Item	CANStateManager: SWS_CanSM_00375 CANStateManager: SWS_CanSM_00498 CANStateManager: SWS_CanSM_00522		
Requirements / Reference to Test Environment	Test environment shall be able to generate a Bus-Off in the Test ECU		
Configuration Parameters	See [BUSOFF_PS002]		
Summary	This test cases tests the behavior of the SUT during short recovery time.		

	Test the behavior of SUT in: a. handling the application requests during the short recovery time b. handling the received messages during the Bus-Off recovery cycle The test procedure generates Bus-Off, waits short Bus-Off recovery time and releases Bus-Off again. The correct behavior in the respective states is observed • on the bus (transmission of frames and acknowledgement of received frames) • on the RTE (application requests) • by events retrieved through diagnostic interface
Needed Adaptation to other Releases	**Needed Adaptation for Release [3.2.2]** Configuration: none Same requirements on configuration Test Steps: low DEM events for bus off have fixed name in R3.2 Same test step sequence
Pre-conditions	All the communication channels are initialized

Main Test Execution		
Test Steps		**Pass Criteria**
Step 1	SWC requests ComM to switch to COMM_FULL_COMMUNICATION using Rte_Call_comRequest_RequestComMode	Call to Rte_Call_comRequest_RequestComMode returns E_OK Rte_Mode API is invoked with mode as RTE_MODE_ComMode_COMM_FULL_COMMUNICATION
Step 2	Generate Bus-Off in the Test ECU	No valid frames are observed in the Bus
Step 3	Check the Events	Event CANSM_E_BUS_OFF is logged with status as DEM_EVENT_STATUS_PREFAILED [SWS_CanSM_00522]
Step 4	SWC requests ComM to switch to COMM_NO_COMMUNICATION using Rte_Call_comRequest_RequestComMode	Call to Rte_Call_comRequest_RequestComMode returns E_NOT_OK [SWS_CanSM_00375]
Step 5	SWC requests ComM to switch to COMM_FULL_COMMUNICATION using Rte_Call_comRequest_RequestComMode	Call to Rte_Call_comRequest_RequestComMode returns E_NOT_OK [SWS_CanSM_00375]
Step 6	WAIT till (CanSMBorTimeL1 / 2).	-

	WHILE WAITING, DO nothing	
Step 7	Trigger a communication sequence in the SUT- Example -ComIPduGroup start	No valid frames are observed in the Bus
Step 8	End generation of Bus-Off in the Test ECU	-
Step 9	Send one message from the Tester to the test ECU	Test ECU acknowledges the message and no error frames are observed in the Bus
Step 10	WAIT till (CanSMBorTimeL1 + 1s) WHILE WAITING, DO nothing	-
Step 11	Check the Events	Event CANSM_E_BUS_OFF is logged with status as DEM_EVENT_STATUS_PASSED [SWS_CanSM_00498]
Step 12	Check whether the SUT is in COMM_FULL_COMMUNICATION	Rte_Mode API is invoked with mode as RTE_MODE_ComMode_COMM_FULL_COMMUNICATION
Step 13	BUS-OFF should be re-covered	Valid frames are observed in the Bus
Post-conditions	-	

5.3.5 [ATS_COMCAN_00273] Behavior of SUT during long recovery time

Test Objective	Behavior of SUT during long recovery time		
ID	ATS_COMCAN_00273	**AUTOSAR Releases**	4.0.3|4.1.1
Affected Modules	CanSM, ComM	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00101 ATR: ATR_ATR_00104		
Trace to R4.1.1 Item	CANStateManager: SWS_CanSM_00376 CANStateManager: SWS_CanSM_00498 CANStateManager: SWS_CanSM_00522 CANStateManager: SWS_CanSM_00515 CANStateManager: SWS_CanSM_00518 COMManager: SWS_ComM_00091 COMManager: SWS_ComM_00778		
Requirements / Reference to Test Environment	Test environment shall be able to generate a Bus-Off in the Test ECU		
Configuration Parameters	See [BUSOFF_PS002]		
Summary	This test case tests the behavior of the SUT during long recovery time.		

	Test the behavior of SUT in: a. handling the application requests during the long recovery time b. handling the received messages during the Bus-Off recovery cycle The test procedure generates Bus-Off, waits short plus long Bus-Off recovery time and releases Bus-Off again. The correct behavior in the respective states is observed • on the bus (transmission of frames and acknowledgement of received frames) • on the RTE (application requests) • Events retrieved through diagnostic interface
Needed Adaptation to other Releases	**Needed Adaptation for Release [3.2.2]** Configuration: none — Same requirements on configuration DEM events for bus off have fixed name in R3.2 Test Steps: low — Same test step sequence
Pre-conditions	All the communication channels are initialized

Main Test Execution		
Test Steps		**Pass Criteria**
Step 1	SWC requests ComM to switch to COMM_FULL_COMMUNICATION using Rte_Call_comRequest_RequestComMode	Call to Rte_Call_comRequest_RequestComMode returns E_OK Rte_Mode API is invoked with mode as RTE_MODE_ComMode_COMM_FULL_COMMUNICATION
Step 2	Generate Bus-Off in the Test ECU	-
Step 3	WAIT till (CanSMBorTimeL1 + 1s) WHILE WAITING, DO check frames on the bus	No valid frames are observed in the Bus
Step 4	Check the Events	Event CANSM_E_BUS_OFF is logged with status as DEM_EVENT_STATUS_PREFAILED [SWS_CanSM_00522]
Step 5	SWC requests ComM to switch to COMM_NO_COMMUNICATION using Rte_Call_comRequest_RequestComMode	Call to Rte_Call_comRequest_RequestComMode returns E_NOT_OK [CANSM376]
Step 6	SWC requests ComM to switch to COMM_FULL_COMMUNICATION using Rte_Call_comRequest_RequestComMode	Call to Rte_Call_comRequest_RequestCom

		Mode returns E_NOT_OK [SWS_CanSM_00376]
Step 7	WAIT till (CanSMBorTimeL2 / 2) WHILE WAITING, DO nothing	-
Step 8	Trigger a communication sequence in the SUT- Example -ComIPduGroup start	No valid frames are observed in the Bus
Step 9	End generation of Bus-Off in the Test ECU	-
Step 10	Send one message from the Tester to the test ECU	Test ECU acknowledges the message and no error frames are observed in the Bus
Step 11	WAIT till (CanSMBorTimeL2 + 1s) WHILE WAITING, DO nothing	-
Step 12	Check the Events	Event CANSM_E_BUS_OFF is logged with status as DEM_EVENT_STATUS_PASSED [CANSM498]
Step 13	Check whether the SUT is in COMM_FULL_COMMUNICATION	Rte_Mode API is invoked with mode as RTE_MODE_ComMode_COMM_FULL_COMMUNICATION [SWS_CanSM_00515][SWS_CanSM_00518][SWS_ComM_00091][SWS_ComM_00778]
Step 14	BUS-OFF should be re-covered	Valid frames are observed in the Bus
Post-conditions	-	

5.3.6 [ATS_COMCAN_00274] Ensure the correct duration of Bus-Off recovery delay time

Test Objective	Ensure the correct duration of Bus-Off recovery delay time		
ID	ATS_COMCAN_00274	**AUTOSAR Releases**	4.0.3|4.1.1
Affected Modules	CanSM, ComM	**State**	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00101 ATR: ATR_ATR_00104		
Trace to R4.1.1 Item	CANStateManager: SWS_CanSM_00375 CANStateManager: SWS_CanSM_00376 CANStateManager: SWS_CanSM_00498 CANStateManager: SWS_CanSM_00521 CANStateManager: SWS_CanSM_00522		

	CANStateManager: SWS_CanSM_00514 CANStateManager: SWS_CanSM_00518 COMManager: SWS_ComM_00091 COMManager: SWS_ComM_00778
Requirements / Reference to Test Environment	Test environment shall be able to generate a Bus-Off in the Test ECU
Configuration Parameters	See [BUSOFF_PS002] ComIpdu(SignalIPdu): AT_274_Ipdu1 (Mapped on CAN Frame => CanTopology) - ComIPduDirection(CommConnectorPort.communicationDirection) = SEND - ComTxModeTrue (IPduTiming.TransmissionModeDeclaration.transmissionModeTrueTiming) -- PERIODIC (CyclicTiming) --- timePeriod > 0
Summary	This test case tests the correct duration of Bus-Off recovery delay time. Test whether the correct time is ensured for short and long recovery time. The test procedure generates bus off, releases Bus-Off and waits for valid messages. The time between Bus-Off generation and messages on the bus shall be the short respectively long bus off recovery time. RTE callbacks and Events are observed additionally.
Needed Adaptation to other Releases	**Needed Adaptation for Release [3.2.2]** Configuration: none Same requirements on configuration Test Steps: low DEM events for bus off have fixed name in R3.2 Same test step sequence
Pre-conditions	All the communication channels are initialized
Main Test Execution	
Test Steps	**Pass Criteria**
Step 1	SWC requests ComM to switch to COMM_FULL_COMMUNICATION using Rte_Call_comRequest_RequestComMode Call to Rte_Call_comRequest_RequestComMode returns E_OK Rte_Mode API is invoked with mode as RTE_MODE_ComMode_COMM_FULL_COMMUNICATION
Step 2	Generate Bus-Off in the Test ECU -
Step 3	Start the measurement timer TMR-1 -
Step 4	WAIT till (CanSMBorTimeL1 / 2) WHILE WAITING, DO check frames on the bus No valid frames are observed in the Bus
Step 5	Check whether the SUT is in COMM_SILENT_COMMUNICATION Rte_Mode API is invoked with mode as RTE_MODE_ComMode_COMM_SIL

		ENT_COMMUNICATION [SWS_CanSM_00521][SWS_ComM_00091][SWS_ComM_00778]
Step 6	End generation of Bus-Off in the Test ECU	-
Step 7	BUS-OFF should be re-covered	Valid frames are observed in the Bus
Step 8	LT: Stop the measurement timer TMR-1 with the first reception of I-PDUs	-
Step 9	Check the calculated elapsed time from the timer TMR-1	The Elapsed time is within the permissible range of CanSMBorTimeL1 [SWS_CanSM_00375][ECUC_CanSM_00128] Note: Time base of calculated Elapsed time should be in-line with CanSMMainFunctionTimePeriod
Step 10	Check whether the SUT is in COMM_FULL_COMMUNICATION	Rte_Mode API is invoked with mode as RTE_MODE_ComMode_COMM_FULL_COMMUNICATION [SWS_CanSM_00514][SWS_CanSM_00518][SWS_ComM_00091][SWS_ComM_00778]
Step 11	Check the Events	Event CANSM_E_BUS_OFF is logged with status as DEM_EVENT_STATUS_PASSED [SWS_CanSM_00498]
Step 12	Generate Bus-Off in the Test ECU	-
Step 13	WAIT till CanSMBorTimeL1 WHILE WAITING, DO check frames on the bus	No valid frames are observed in the Bus
Step 14	Check the Events	Event CANSM_E_BUS_OFF is logged with status as DEM_EVENT_STATUS_PREFAILED [SWS_CanSM_00522]
Step 15	Start the measurement timer TMR-1	-
Step 16	WAIT till (CanSMBorTimeL1 / 2) WHILE WAITING, DO check frames on the bus	No valid frames are observed in the Bus
Step 17	End generation of Bus-Off in the Test ECU	-
Step 18	BUS-OFF should be re-covered	Valid frames are observed in the Bus
Step 19	LT: Stop the measurement timer TMR-1 with	-

	the first reception of I-PDUs	
Step 20	Check the calculated elapsed time from the timer TMR-1	The Elapsed time is within the permissible range of CanSMBorTimeL2 [SWS_CanSM_00376][ECUC_CanSM_00129] Note: Time base of calculated Elapsed time should be in-line with CanSMMainFunctionTimePeriod
Post-conditions	-	