Document Title	Acceptance Test Specification of Global Time Synchronization
<b>Document Owner</b>	AUTOSAR
<b>Document Responsibility</b>	AUTOSAR
<b>Document Identification No</b>	841
<b>Document Classification</b>	Auxiliary

Document Status	Final
Part of AUTOSAR Standard	Acceptance Tests for Classic Platform
Part of Standard Release	1.2.0

Document Change History				
Date	Release	Changed by	Change Description	
2016-12-15	1.2.0	AUTOSAR Release Management	<ul> <li>Initial release, including test suites on</li> <li>RS_BRF_01660- Global Time Synchronization over CAN</li> <li>RS_BRF_01660 - Global Time Synchronization over FlexRay</li> <li>RS_BRF_01660 - Global Time Synchronization over Multiple Bus</li> </ul>	

#### **Disclaimer**

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

The material contained in this specification is protected by copyright and other types of Intellectual Property Rights. The commercial exploitation of the material contained in this specification requires a license to such Intellectual Property Rights.

This specification may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only.

For any other purpose, no part of the specification may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The AUTOSAR specifications have been developed for automotive applications only. They have neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

#### **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	1	Acror	nyms and abbreviations	6
2	I	Relat	ed Documentation	. 7
	2.1	In	put documents	. 7
3	;	Scop	e	. 8
4		•	BRF_01660 - Global Time Synchronization over CAN	
			eneral Test Objective and Approach	
			Test System	
	4	.1.2	Test Case Design	13
	4.2	С	onfiguration requirements	14
			Test Configuration	
	4		Static Configuration	
	4.3		e-usable Test Steps	
			est cases	21
	4	.4.1	[ATS_GTS_01228] Global Time Master: Setting of Global Time base by Active Customer and sending of SYNC frames	
			(CRC SUPPORTED) over CAN	21
	4	.4.2	[ATS_GTS_01266] Global Time Master: Handling of SYNC message	
		4.0	Confirmation Failures	25
	4	.4.3	[ATS_GTS_01264] Global Time Master: Setting of Global Time base by Active Customer and sending of offset frames	
			(CRC_NOT_SUPPORTED) over CAN	27
	4	.4.4	[ATS_GTS_01268] Global Time Master: Handling of Offset message	
	4	4 5	Confirmation Failures	
	4	.4.5	[ATS_GTS_01238] Global Time Master: Handling of time base using NvM (Storage and Retrieve)	
	4	.4.6		וכ
			(CRC_IGNORED) over CAN, Synchronize Local Time Base and	
	4	4 7	share the current time to active customers	33
	4	.4.7	[ATS_GTS_01271] Time Slave: Handling of SYNC message reception timeout (CanTSynGlobalTimeFollowUpTimeout)	
	4	.4.8	[ATS_GTS_01273] Time Slave: Handling of SYNC message	,,
			Sequence Mismatch failures	38
	4	.4.9	[ATS_GTS_01286] Time Slave: Reception of Offset frames	
			(CRC_VALIDATED) over CAN, Synchronize Local Time Base and share the current time to active customers.	13
	4	.4.10	[ATS_GTS_01244] Time Slave: Reception of Offset frames	τυ
			(CRC_NOT_VALIDATED) over CAN, Synchronize Local Time Base	
			and share the current time to active customer	45
	4	.4.11	[ATS_GTS_01272] Time Slave: Handling of Offset message reception timeout (CanTSynGlobalTimeFollowUpTimeout)	16
	4	.4.12	P[ATS_GTS_01274] Time Slave: Handling of Offset message	τU
			Sequence Mismatch failure	49
	4	.4.13	[ATS_GTS_01270] Time Slave: Synchronize Runnable entities to time	
			base	აპ

5	RS_	BRF_01660 - Global Time Synchronization over FlexRay	56
	5.1	Seneral Test Objective and Approach	56
		Test System Test Case Design	
	5.2	Configuration requirements	60
		Test ConfigurationStatic Configuration	
		Re-usable Test Steps est Cases	
		[ATS_GTS_01275] Global Time Master: Setting of Global Time base and user data by Active Customer and sending of SYNC frames (CRC_NOT_SUPPORTED) over FlexRay	67
		by Active Customer and sending Offset frames (CRC_SUPPORTED) over FlexRay	70
		[ATS_GTS_01249] Time Slave: Reception of SYNC frames (CRC_VALIDATED) over FlexRay, Synchronize Local Time Base and share the current time to active customers	72
		(CRC_IGNORED) over FlexRay, Synchronize Local Time Base and share the current time to active customers.	75
		[ATS_GTS_01256] Time Slave: Handling of SYNC message Sequence Mismatch failures	77
	5.4.7	Base and share the current time to active customers	
6	RS_	BRF_01660 - Global Time Synchronization over Multiple Bus	85
	6.1	Seneral Test Objective and Approach	85
		Test System Test Case Design	
	6.2 C	Configuration requirements	93
		Test ConfigurationStatic Configuration	
		Re-usable Test Steps1 est Cases1	
	6.4.1	[ATS_GTS_01281] Global Time Master over Multiple Bus(Single Time Domain): Setting of Global Time base and user data and sending of SYNC frame over CAN and FlexRay	01
	6.4.2	[ATS_GTS_01257] Global Time Master over Multiple Bus(MultipleTime Domain): Setting of Global Time base and user data	
	6.4.3	and sending of SYNC frame over CAN and FlexRay	

## AUTOSAR TC Release 1.2.0

6.4.4	[ATS_GTS_01259] Time Gateway - Time Slave on CAN and Time	
	Master on FlexRay11	5
6.4.5	[ATS_GTS_01260] Time Gateway - Time Slave on CAN (Network 1)	
	and Time Master on CAN (Network 2) 11	8

## 1 Acronyms and abbreviations

Abbreviation /	Description:
Acronym:	
AT	Acceptance Test
CAN	Controller Area Network
ECU	Electronic Control Unit
LT	Lower Tester
PCO	Point of Control and Observation
Rx	Reception
SUT	System Under Test
SWC	Software Component
TCP	Test Coordination Procedures
Tx	Transmission
UT	Upper Tester

## 2 Related Documentation

## 2.1 Input documents

- [1] Specification of Synchronized Time-Base Manager AUTOSAR\_SWS\_SynchronizedTimeBaseManager.pdf
- [2] Specification of Time Synchronization over CAN AUTOSAR\_SWS\_TimeSyncOverCAN.pdf
- [3] Specification of Time Synchronization over FlexRay AUTOSAR\_SWS\_TimeSyncOverFlexRay.pdf
- [4] Specification of Operating System AUTOSAR\_SWS\_OS.pdf
- [5] Specification of Basic Software Mode Manager AUTOSAR\_SWS\_BSWModeManager.pdf
- [6] Specification of CAN Interface AUTOSAR SWS CANInterface.pdf
- [7] Specification of FlexRay Interface AUTOSAR\_SWS\_FlexRayInterface.pdf
- [8] Specification of RTE AUTOSAR\_SWS\_RTE.pdf
- [9] Requirements on Synchronized Time-Base Manager AUTOSAR\_SRS\_SynchronizedTimeBaseManager.pdf
- [10] Requirements on Acceptance Tests AUTOSAR\_ATR\_Requirements.pdf
- [11] Requirements on AUTOSAR Features AUTOSAR\_RS\_Features.pdf
- [12] System Template
  AUTOSAR\_TPS\_SystemTemplate.pdf

## 3 Scope

The following test cases are used to verify the correct behavior of all the Global Time Synchronization features.

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.

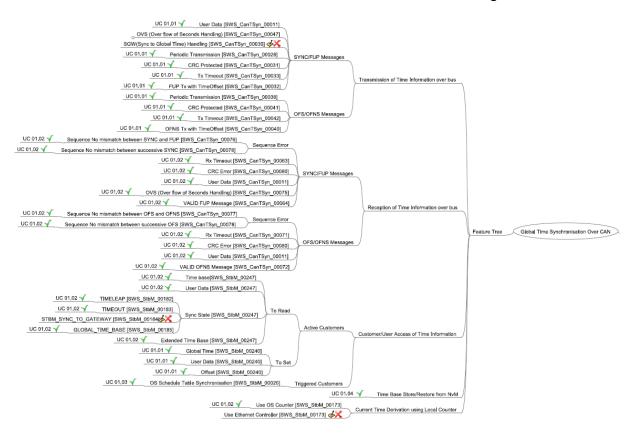
## 4 RS\_BRF\_01660 - Global Time Synchronization over CAN

## 4.1 General Test Objective and Approach

This Test Specification intends to cover the Global Time Synchronization feature of StbM and CanTSyn as described in the AUTOSAR Feature [RS\_BRF\_01660].

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:



Below Features are not tested in this test suite:

- SGW(Sync to Global Time) Handling [SWS\_CanTSyn\_00030] and STBM\_SYNC\_TO\_GATEWAY [SWS\_StbM\_00184]: Feature are tested in the test suite 'Global Time Synchronization over Multiple Bus'.
- Use Ethernet Controller [SWS\_StbM\_00173]: Testing over Ethernet bus is out of scope of this ATS.

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

## 4.1.1 Test System

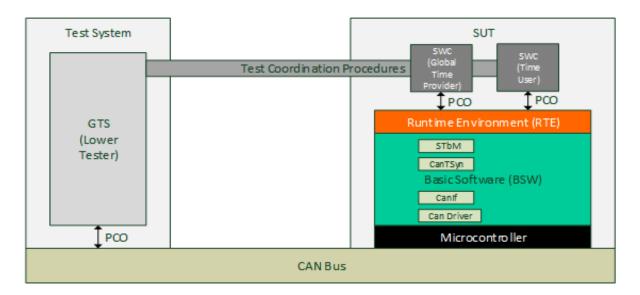
#### 4.1.1.1 Overview on Architecture

In order to cover the required features / sub-features coverage, the environment has been separated in several use cases.

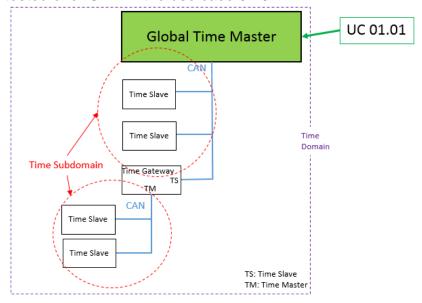
Test Cases are derived based on below use cases

#### 4.1.1.1.1 UC 01.01: Global Time Master over CAN

SUT acts as Global time Master and Sets time base, offset time base and Trigger for transmission of Synchronization over CAN bus.

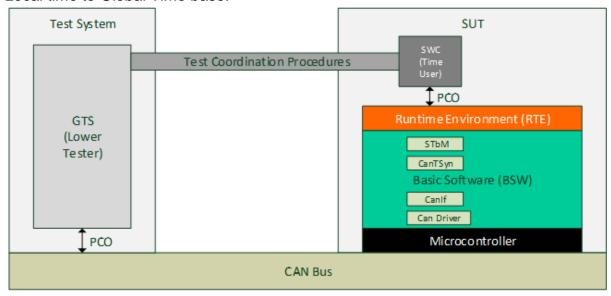


As shown in below figure, Functionalities of Global Time master of a time domain are tested over CAN in the use case 01.01

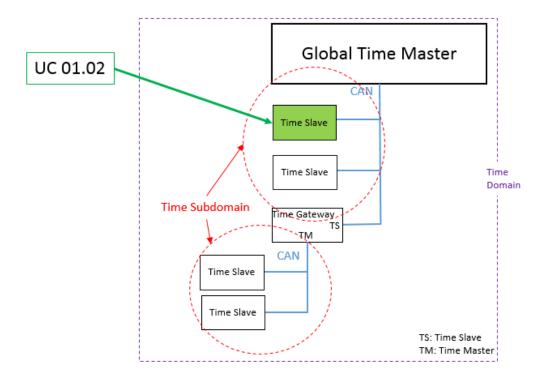


## 4.1.1.1.2 UC 01.02: Time Slave over CAN

SUT acts as Time Slave and Gets time base, offset time base and Synchronizes Local time to Global Time base.

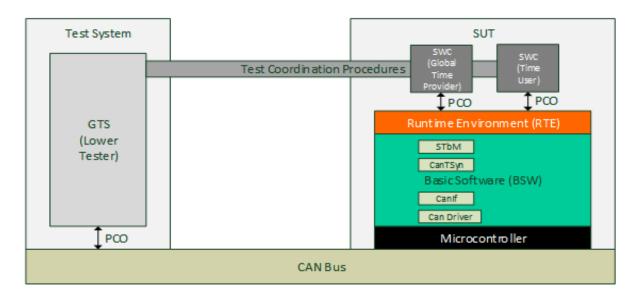


As shown in below figure, Functionalities of Time Slave of a time domain are tested over CAN in the use case 01.02



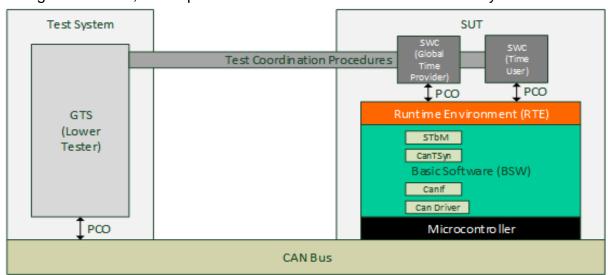
## 4.1.1.1.3 UC 01.03: Synchronization of Runnable entities to Global Time

SUT Synchronizes Runnable entities to Global Time base using OS Scheduler (Triggered Customer).



## 4.1.1.1.4 UC 01.04: Initialization of time base from value stored in Non-volatile Memory

During initialization, SUT updates time base from Non-volatile Memory.



## 4.1.1.2 Specific Requirements

Not Applicable.

## 4.1.1.3 Test Coordination Requirements

#### UC 01.01: Global Time Master over CAN

 Test System (LT <CAN>) shall read the CanTSyn CAN Frames and decode the same as per Frame Format provided in AUTOSAR\_SWS\_TimeSyncOverCAN.

## UC 01.02: Time Slave over CAN

• Test System (LT <CAN>) shall encode the CanTSyn CAN Frames as per Frame Format provided in AUTOSAR\_SWS\_TimeSyncOverCAN and transmit over bus.

## Requirements for CRC Calculation

Test System (LT <CAN>) shall use the Crc\_CalculateCRC8H2F() (Refer AUTOSAR Specification
of CRC Routines AUTOSAR\_SWS\_CRCLibrary.pdf) to calculate the CRC of the Frame. Below
are the parameters used for CRC calculation:

The CRC start value shall be 0xFF.

The CRC final XOR-value shall be 0xFF.

The CRC polynomial shall be 0x2F.

The DataIDList shall be same as provided in CanTSyn Static Configuration.

#### 4.1.2 Test Case Design

Below diagrams explain test design for different use cases

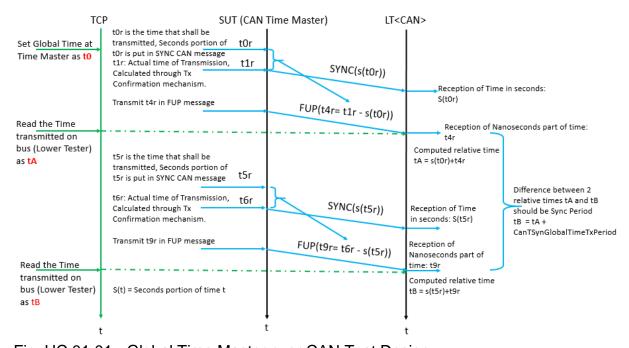


Fig: UC 01.01 - Global Time Master over CAN Test Design

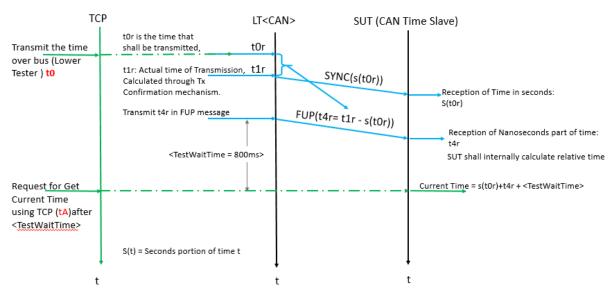


Fig: UC 01.02 - Time Slave over CAN Test Design

## 4.2 Configuration requirements

The configuration can be divided into two separate parts. The *test configuration* describes variables used to parameterize the test case. The *static configuration* describes the necessary settings of the DUT in order to allow a test case to perform.

## 4.2.1 Test Configuration

Communication data base for CanTSyn is depicted below

test configuration parameters			
I-Pdu	CAN ID	Tx ECU	Rx ECU
AT_101_lpdu	101	SUT	Test Bench
AT_102_lpdu	102	Test Bench	SUT

## 4.2.2 Static Configuration

## 4.2.2.1 Static Configuration Groups

SCG_ATS_GlobalTimeSync_Time Master	
System Description Parameters	
StbM	
SystemTemplate::GlobalTime::GlobalTimeMaster.isSystemWideGl	TRUE
obalTimeMaster	
CanTSyn	
SystemTemplate::GlobalTime::GlobalTimeMaster.syncPeriod	2000ms
SystemTemplate::GlobalTime::CAN::GlobalTimeCanMaster.syncC	80ms
onfirmationTimeout	
SystemTemplate::GlobalTime::GlobalTimeDomain.globalTimePdu	Ref. To PDU
ECU Configuration Parameters	
StbM	
StbM.StbMGeneral.StbMMainFunctionPeriod	5ms

StbM.StbMSynchronizedTimeBase.StbMLocalTimeRef	Ref. to
	OSCounter
CanTSyn	
CanTSyn.CanTSynGeneral.CanTSynMainFunctionPeriod	5ms
CanTSyn.CanTSynGlobalTimeDomain.	1
CanTSynGlobalTimeDomainId	
CanTSyn.CanTSynGlobalTimeDomain.	Ref. to StbM
CanTSynSynchronizedTimeBaseRef	time base
CanTSyn.CanTSynGlobalTimeDomain.	100ms
CanTSynGlobalTimeMaster.	
CanTSynGlobalTimeTxFollowUpOffset	
CanTSyn.CanTSynGlobalTimeDomain.	0
CanTSynGlobalTimeMaster.CanTSynGlobalTimeMasterPdu.	
CanTSynGlobalTimeMasterConfirmationHandleId	
Use Cases	
UC 01.01	

SCG_ATS_GlobalTimeSync_Time Slave	
System Description Parameters	
StbM	
SystemTemplate::GlobalTime::GlobalTimeMaster.isSystemWideGl	FALSE
obalTimeMaster	
CanTSyn	
SystemTemplate::GlobalTime::GlobalTimeMaster.syncPeriod	2000ms
SystemTemplate::GlobalTime::GlobalTimeDomain.followUpTimeou tValue	300ms
tvalue	
ECU Configuration Parameters	
StbM	
StbM.StbMGeneral.StbMMainFunctionPeriod	5ms
StbM.StbMSynchronizedTimeBase.StbMLocalTimeRef	Ref. to
	OSCounter
2 72	
CanTSyn	
CanTSyn.CanTSynGeneral.CanTSynMainFunctionPeriod	5ms
CanTSyn.CanTSynGlobalTimeDomain.	1
CanTSynGlobalTimeDomainId	
CanTSyn.CanTSynGlobalTimeDomain.	1
CanTSynGlobalTimeSequenceCounterJumpWidth	
CanTSyn.CanTSynGlobalTimeDomain.	Ref. to StbM
CanTSynSynchronizedTimeBaseRef	time base
CanTSyn.CanTSynGlobalTimeDomain.	1
CanTSynGlobalTimeSlave.CanTSynGlobalTimeSlavePdu.	
CanTSynGlobalTimeSlaveConfirmationHandleId	
CanTSyn.CanTSynGlobalTimeDomain.	Ref. to PDU
CanTSynGlobalTimeSlave.CanTSynGlobalTimeSlavePdu.	

CanTSynGlobalTimePduRef	
Use Cases	
UC 01.02	

SCG_ATS_GlobalTimeSync_Schedule table synchronization	
System Description Parameters	
StbM	
SystemTemplate::GlobalTime::GlobalTimeMaster.isSystemWideGlobalTimeMaster	TRUE
ECU Configuration Parameters	
StbM	
StbM.StbMGeneral.StbMMainFunctionPeriod	5ms
StbM.StbMTriggeredCustomer.StbMTriggeredCustomerPeriod	5ms
StbM.StbMSynchronizedTimeBase.StbMLocalTimeRef	Ref. to
	OSCounter
StbM.StbMTriggeredCustomer.StbMOSScheduleTableRef	Ref. to OS
	Schedule table
StbM.StbMTriggeredCustomer.StbMSynchronizedTimeBaseRef	Ref. to StbM
	time base
Test Cases	
UC 01.03	

SCG_ATS_GlobalTimeSync_NvM	
System Description Parameters	
StbM	
SystemTemplate::GlobalTime::GlobalTimeMaster.isSystemWideGl	TRUE
obalTimeMaster	
ECU Configuration Parameters	
StbM	
StbM.StbMGeneral.StbMMainFunctionPeriod	5ms
StbM.StbMSynchronizedTimeBase.StbMLocalTimeRef	Ref. to
	OSCounter
StbM.StbMSynchronizedTimeBase.StbMStoreTimebaseNonVolatil	STORAGE_AT_
е	SHUTDOWN
Use Cases	
UC 01.04	

## 4.2.2.2 Required System Description

Refer section 3.2.2.1

## 4.2.2.3 Required ECU Configuration

Refer section 3.2.2.1

## 4.2.2.4 Required Software Components

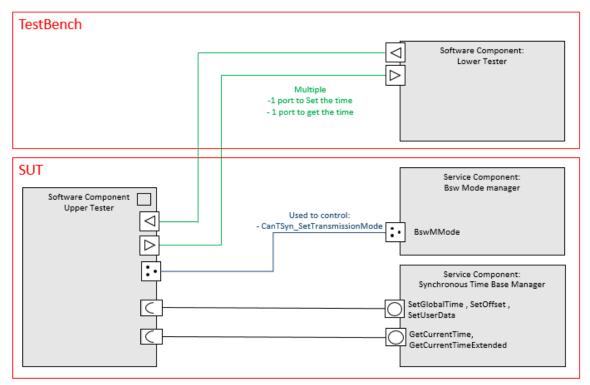


Fig: UC 01.01 - Global Time Master over CAN SWC Overview

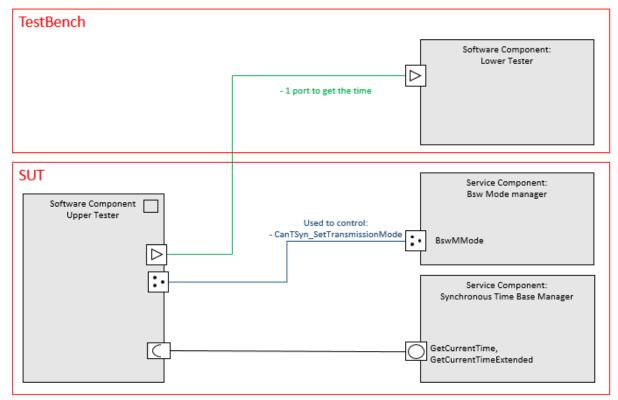


Fig: UC 01.02 - Time Slave over CAN SWC Overview

## 4.2.2.4.1 SWC Client GlobalTime Provider

SWC Name	GlobalTime_Provider		
	Name	Client_SetGloba	alTime
	Туре	<b>Type</b> RPortPrototype	
	Interface	GlobalTime_Master_Interface	
	Requirements		
	Name	Client_SetUser[	Data
PORTS	Туре	RPortPrototype	
	Interface	GlobalTime_Ma	ster_Interface
	Requirements		
	Name	Client_SetOffse	t
	Туре	RPortPrototype	
	Interface	GlobalTime_Ma	ster_Interface
	Requirements		
	Name	RUN_GlobalTim	neProvider
	Requirements	Runnable shall	be invoked by TCP
		Name	sscp_GlobalTimeProvider
		Туре	SynchronousServerCallPoint
RUNNABLE ENTITIES	ServerCallPoint	Access to	Client_SetGlobalTime (Write operation) Client_SetUserData (Write operation) Client_SetOffset (Write
		Requirements	operation)

## 4.2.2.4.2 SWC Client Time User

SWC Name	Time_User	
	Name	Client_GetCurrentTime

	Туре	RPortPrototype	
	Interface	GlobalTime_Slave_Interface	
	D		
	Requirements		
PORTS			
	Name	Client_GetCurre	entTimeExtended
	Туре	RPortPrototype	
	Interface	GlobalTime_Sla	ave_Interface
	Requirements		
	Name	RUN_TimeUser	
	Requirements	Runnable shall be invoked by TCP	
		Name	sscp_TimeUser
		Туре	SynchronousServerCallPoint
RUNNABLE ENTITIES		Access to	Client_GetCurrentTime (Read operation)
	ServerCallPoint		Client_GetCurrentTimeExtended (Read operation)
		Requirements	

## 4.2.2.4.3 SWC Server StbM

	O OCI VCI OLDINI		
SWC Name	StbM		
	Name	Server_SetGlobalTime  PPortPrototype	
	Туре		
	Interface	GlobalTime_Master_Interface	
	Requirements		
	Name	Server_SetOffset	
	Туре	PPortPrototype	
	Interface	GlobalTime_Master_Interface	
	Requirements		

	Name	Server_SetUser	Data
	Туре	PPortPrototype	
PORTS	Interface	GlobalTime_Master_Interface	
	Requirements		
	Name	Server_GetCurr	entTime
	Туре	PPortPrototype	
	Interface	GlobalTime_Sla	ve_Interface
	Requirements		
	Name	_	entTimeExtended
	Туре	PPortPrototype	
	Interface	GlobalTime_Sla	ve_Interface
	Requirements		
	Nome	CthM CotClobo	ITim o
	Name	StbM_SetGlobalTime	
	Requirements		
		Name	OIE_ SetGlobalTime
	Started by Event	Туре	OperationInvokedEvent Port:Server_SetGlobalTime Operation: Read
		Requirements	Operation: Nead
	Name	StbM_SetOffset	
	Requirements		
RUNNABLE		Name	OIE_SetOffset
ENTITIES	ServerCallPoint	Туре	OperationInvokedEvent Port: Server_SetOffset Operation: Read
		Requirements	
	Name	StbM_SetUserD	lata
	INAILIE	JUNIA JEIOSEID	vala

Requirements		
	Name	OIE_ SetUserData
Started by Event	Туре	OperationInvokedEvent
	, ,	Port: Server_SetUserData
		Operation: Read
	Requirements	
Name	StbM_GetCurre	ntTime
Requirements		
		,
	Name	OIE_ GetCurrentTime
Ctarted by Event	True	On a ration law also dEvant
Started by Event	Туре	OperationInvokedEvent Port: Server_GetCurrentTime
		Operation: Read
	Requirements	Operation: Nead
Name	StbM_GetCurre	ntTimeExtended
-		
Requirements		
	Name	OIE_GetCurrentTimeExtended
Started by Event	Туре	OperationInvokedEvent
	-	Port:
		Server_GetCurrentTimeExtende
		Operation: Read
	Requirements	

## 4.3 Re-usable Test Steps

Not applicable

## 4.4 Test cases

# 4.4.1 [ATS\_GTS\_01228] Global Time Master: Setting of Global Time base by Active Customer and sending of SYNC frames (CRC\_SUPPORTED) over CAN

	Global Time Master: Setting of G of SYNC frames (CRC_SUPPOR	lobal Time base by Active Customer and sending RTED) over CAN
ID	ATS_GTS_01228	AUTOSAR 4.2.1 4.2.2

	<u> </u>	Dalassas	
A 66	0.114 0 70	Releases	
Affected Modules	StbM, CanTSyn	State	reviewed
	ATR: ATR_ATR_00131		
	ATR: ATR_ATR_00132		
on Acceptance Test Document	ATR: ATR_ATR_00133		
	SynchronizedTimeBaseManager	C/V/C C+PV	4 00240
Item	TimeSyncOverCAN: SWS CanT		W_00240
itom	TimeSyncOverCAN: SWS_CanTSyn_00028		
	TimeSyncOverCAN: SWS_CanT		
Requirements /	Use Case UC01.01		
Reference			
to Test			
Environment	0.1.14		
	StbM: StbMSynchronizedTimeBaseIder	tifior – 1	
i arameters	StbWSynchronized i infebaseider	ılıııcı — ı	
	CanTSyn:		
	CanTSynGlobalTimeTxCrcSecur		
			{0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15}
	Can   SynGlobal   ImeSynCDataiD   "R", "A", "T", "S", "G", "T", "S", "S	Listvalue in " "V" "N"\	ASCII = {"A", "U", "T", "O", "S", "A",
			0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15}
	CanTSynGlobalTimeFupDataIDL	istValue in /	ASCII = {"A", "U", "T", "O", "S", "A", "R",
	"A", "T", "S", "G", "T", "S", "F", "U	", "P"}	•
Summary	Aim is to:		
	Verify that StbM accepts the global time base from Upper Tester using client-server Interface.		
	Verify that CanTSyn shall Transmit the global time base to time slave periodically via SYNC and FUP message with CRC secured.		
Needed	Not Applicable		
Adaptation to	Not Applicable		
other Releases			
Pre-conditions	SUT shall be initialized.		
Main Test Exec	ution		
Test Steps			Pass Criteria
Step 1	[CP]		
	Start RUN_ GlobalTimeProvider		
Step 2	[RUN <run_ globaltimeprovid<="" th=""><th>er&gt;]</th><th>[RUN<run_ globaltimeprovider="">]</run_></th></run_>	er>]	[RUN <run_ globaltimeprovider="">]</run_>
	Execute Rte_Call_ Client_SetGlo and Rte_Call_ Client_SetUserDa below values:		Rte_Call returns RTE_E_OK
	timeBaseId = 1		
	StbM_TimeStampType.nanoseco 0x00000000	onds =	
	StbM_TimeStampType.seconds	=	

	0×00000=10	<del>                                     </del>
	0x00000E10	
	StbM_TimeStampType.secondsHi = 0x0000	
	StbM_UserDataType.User Data Byte 0 = 0xAA	
Step 3	[CP]	
	Wait 100ms	
Step 4	[CP]	
	Start RUN_TimeUser	
Step 5	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time received from Time user should be as mentioned below:
		StbM_TimeStampType.nanoseconds = 0x00000000 + <testwaittime =<br="">100ms&gt;.</testwaittime>
		StbM_TimeStampType.seconds = 0x00000E10(3600d)
		StbM_TimeStampType.secondsHi = 0x0000
		StbM_UserDataType.User Data Byte 0 = 0xAA
Step 6	[LT <can>]</can>	[LT]
	Receives the SYNC message with CRC validation as per test co-ordination requirement for CRC	Receives SYNC message in the format mentioned below:
	ioquirement for cive	Byte 0: Type = 0x20
		Byte 1: CRC
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: User Byte 0 = 0xAA
		Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds
Step 7	[LT <can>]</can>	[LT]
	Receives the FUP message with CRC validation as per test co-ordination requirement for CRC	Receives FUP message in the format mentioned below:

-		
		Byte 0: Type = 0x28
		Byte 1: CRC
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: reserved (Bit 7 to Bit 3) = 0
		SGW (Bit 2) = 0
		OVS = Overflow of seconds (Bit 1 to Bit 0)
		Byte 4-7: SyncTimeNSec = StbM_TimeStampType.nanoseconds
Step 8	[LT <can>]</can>	[[LT]
	Receives Global time base.	Get the time stamp values as below:
	Store the time as base for next periodic message processing	StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 9	[LT <can>]</can>	[[LT]
	Receives the next periodic SYNC and FUP message with CRC validation as per test coordination requirement for CRC and	
	calculate the CRC value	Time stamp values shall be as mentioned below:
		StbM_TimeStampType.nanoseconds = tB.Nano
		StbM_TimeStampType.seconds = tB.Sec
		StbM_TimeStampType.secondsHi = tB.SecHi
		StbM_UserDataType.User Data Byte 0 = 0xAA
		The difference between two time base shall be
		tB – tA = CanTSynGlobalTimeTxPeriod (2 second) + CanTSynGlobalTimeTxFollowUpOffset.

Post-	None
conditions	

## 4.4.2 [ATS\_GTS\_01266] Global Time Master: Handling of SYNC message Confirmation Failures

Test Objective	Global Time Master: Handling of SYNC message Confirmation Failures		
ID	ATS_GTS_01266 AUTOSAR 4.2.1 4.2.2		
	K10_010_01200	Releases	7.2.1 4.2.2
Affected Modules	CanTSyn	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00133		
Trace to SWS Item	TimeSyncOverCAN: SWS_CanT	Syn_00033	
Requirements / Reference to Test Environment	Use Case UC01.01		
	StbM: StbMSynchronizedTimeBaseIdel	ntifier = 1	
	CanTSyn: CanTSynGlobalTimeTxCrcSecured= CRC_NOT_SUPPORTED		
	Canlf: Canlf.CanlfInitCfg.CanlfTxPduCf	g.CanIfTxP	duUserTxConfirmationUL = CDD.
	Aim is to:  Verify that CanTSyn shall send the SYNC message and on confirmation timeout		
	'CanTSynMasterConfirmationTimeout', transmission request shall be revoked and no FUP message shall be sent.		
Needed Adaptation to other Releases	Not Applicable		
	SUT shall be initialized. StbM shall be initialized with base time: StbM_TimeStampType.nanoseconds = 0x00000000 StbM_TimeStampType.seconds = 0x00000000 StbM_TimeStampType.secondsHi = 0x0000		
Main Test Exec	ution		
Test Steps			Pass Criteria
Step 1	[SUT]		[LT <can>]</can>
	CanTSyn transmit SYNC messa	ge	Receive SYNC message with format
			Byte 0: Type = 0x10
			Byte 1: User Byte 1 = 0x00

Post- conditions	None	
		(Since parameter CanIfTxPduUserTxConfirmationUL for CanTSyn PDU = CDD, confirmation shall not reach StbM.This causes transmit confirmation timeout at CanTSyn)
	Waits for FUP message	No FUP message received.
Step 4	[LT <can>]</can>	+ CanTSynMasterConfirmationTimeout.  [LT <can>]</can>
		Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds + CanTSynGlobalTimeTxPeriod (2 second)
		Byte 3: User Byte 0 = 0x00
		SC = 0x0
		Byte 2: D = 0x1
		Byte 1: User Byte 1 = 0x00
	message	Byte 0: Type = 0x10
Olop o	CanTSyn transmit next periodic SYNC	Receive SYNC message with format
Step 3	[SUT]	CanTSyn) [LT <can>]</can>
		Since parameter CanlfTxPduUserTxConfirmationUL for CanTSyn PDU = CDD, confirmation shall not reach StbM.This causes ransmit confirmation timeout at
	Waits for FUP message	No FUP message received.
Step 2	[LT <can>]</can>	StbM_TimeStampType.seconds [LT <can>]</can>
		Byte 4-7: SyncTimeSec =
		Byte 3: User Byte 0 = 0x00
		SC = 0x0
		Byte 2: D = 0x1

# 4.4.3 [ATS\_GTS\_01264] Global Time Master: Setting of Global Time base by Active Customer and sending of offset frames (CRC\_NOT\_SUPPORTED) over CAN

Test Objective	Global Time Master: Setting of Global Time base by Active Customer and sending		
•	of offset frames (CRC_NOT_SUPPORTED) over CAN		
ID	ATS_GTS_01264	AUTOSAR Releases	4.2.1 4.2.2
Affected Modules	StbM, CanTSyn	State	reviewed
	ATR: ATR_ATR_00131		
	ATR: ATR_ATR_00132 ATR: ATR ATR 00133		
Test Document			
Trace to SWS Item	SynchronizedTimeBaseManager: TimeSyncOverCAN: SWS_CanT; TimeSyncOverCAN: SWS_CanT; TimeSyncOverCAN: SWS_CanT;	Syn_00038 Syn_00040	M_00240
	Use Case UC01.01		
Reference			
to Test Environment			
Configuration	StbM:		
Parameters	StbMSynchronizedTimeBaseIden	tifier = 16.	
	0		
	CanTSyn: CanTSynGlobalTimeTxCrcSecure	ed= CRC N	OT SUPPORTED
Summary	Aim is to:	<u> </u>	01_0011 01(12)
,			
	Verify that StbM accepts the global time base from Upper Tester using client-server		
	Interface.		
	Verify that CanTSyn shall Transmit the offset time base to time slave periodically via OFS and OFNS message.		
Needed	Not Applicable		
Adaptation to	, , , , , , , , , , , , , , , , , , ,		
other Releases			
	SUT shall be initialized.		
Main Test Execu	ution		<b>1</b> -
Test Steps	kon		Pass Criteria
Step 1	[CP]		
	Start RUN_ GlobalTimeProvider		
Step 2	[RUN <run_ globaltimeprovident<="" th=""><th>er&gt;1</th><th>[RUN<run_ globaltimeprovider="">]</run_></th></run_>	er>1	[RUN <run_ globaltimeprovider="">]</run_>
•	_	•	
	Execute Rte_Call_ Client_ SetGlowith below values:	balTime	Rte_Call returns RTE_E_OK
	timeBaseId = 1		
	StbM_TimeStampType.nanoseco 0x00000000	nds =	
	StbM_TimeStampType.seconds =	=	

	0x00000E10	
Step 3	StbM_TimeStampType.secondsHi = 0x0000 [CP]	
Cton 4	Wait 100ms	TOUR DUN Clab all in a Provider 1
Step 4	[RUN <run_ globaltimeprovider="">]</run_>	[RUN <run_ globaltimeprovider="">]</run_>
	Execute Rte_Call_ Client_ SetOffset with below values:	Rte_Call returns RTE_E_OK
	timeBaseId = 16	
	StbM_TimeStampType.nanoseconds = 0x00000000	
	StbM_TimeStampType.seconds = 0x00000064	
	StbM_TimeStampType.secondsHi = 0x0000	
Step 5	[CP]	
	Wait 100ms	
Step 6	[CP]	
	Start RUN_TimeUser	
Step 7	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
	Time base = 16	Time received from Time user shall be as mentioned below:
		StbM_TimeStampType.nanoseconds = 0x000000000 + <testwaittime =<br="">100ms&gt; + <testwaittime 100ms="" ==""></testwaittime></testwaittime>
		StbM_TimeStampType.seconds = 0x00000E10 + 0x00000064
		StbM_TimeStampType.secondsHi = 0x0000
Step 8	[LT <can>]</can>	[LT]
	Receives the OFS message	Receives OFS message with format
		Byte 0: Type = 0x30
		Byte 1: reserved = 0x00
		Byte 2: D = 0x0
		SC = 0x0

		Byte 3: OfsTimeSecLsbHi = <stbm_timestamptype.secondshi (lsb)="0x00">  Byte 4-7: OfsTimeSecLsbLo = <stbm_timestamptype.seconds 0x00000064="" ==""></stbm_timestamptype.seconds></stbm_timestamptype.secondshi>
Step 9	[LT <can>]</can>	[LT]
	Receives the OFNS message	Receives OFNS message with format
		Byte 0: Type = 0x38
		Byte 1: reserved = 0x00
		Byte 2: D = 0x0
		SC = 0x0
		Byte 3: OfsTimeSecMsbHi = <stbm_timestamptype.secondshi (MSB) = 0x00&gt;</stbm_timestamptype.secondshi 
		Byte 4-7: OfsTimeNSec = <stbm_timestamptype.nanoseconds = 0x00000000&gt;</stbm_timestamptype.nanoseconds 
Post- conditions	None	

## 4.4.4 [ATS\_GTS\_01268] Global Time Master: Handling of Offset message Confirmation Failures

Test Objective	Global Time Master: Handling of Offset message Confirmation Failures		
		AUTOSAR Releases	
Affected Modules	CanTSyn	State	reviewed
Trace to Requirement on Acceptance Test Document			
Trace to SWS Item	TimeSyncOverCAN: SWS_CanTSyn_00042		
Requirements / Reference to Test Environment	Use Case UC01.01		
Parameters	StbM: StbMSynchronizedTimeBaseIdentifier = 16  CanTSyn: CanTSynGlobalTimeTxCrcSecured= CRC_NOT_SUPPORTED		

	1		
	CanIf: CanIf.CanIfInitCfg.CanIfTxPduCfg.CanIfTxPduUserTxConfirmationUL = CDD.		
Summary	Aim is to:		
	Verify that CanTSyn shall send the OFS message and on confirmation timeout 'CanTSynMasterConfirmationTimeout', transmission request shall be revoked and no OFNS message shall be sent.		
Needed	Not Applicable		
Adaptation to other Releases			
	SUT shall be initialized.		
r re-conditions	StbM shall be initialized with base time: StbM_TimeStampType.nanoseconds = 0x0 StbM_TimeStampType.seconds = 0x00000 StbM_TimeStampType.secondsHi = 0x0000	000	
Main Test Exec	ution		
Test Steps		Pass Criteria	
Step 1	[SUT]	[LT <can>]</can>	
	CanTSyn transmit OFS message with	Receive OFS message with format	
	TimeBaseId = 16	Byte 0: Type = 0x30	
	OfsTimeSecLsbLo = 0x00000002	Byte 1: reserved = 0x00	
		Byte 2: D = 0x0	
		SC = 0x0	
		Byte 3: OfsTimeSecLsbHi = 8 Bit offset time stamp (LSB) from secondsHi	
		Byte 4-7: OfsTimeSecLsbLo = StbM_TimeStampType.seconds	
Step 2	[LT <can>]</can>	[LT <can>]</can>	
	We're to OFNO access	N. OFNO	
	Waits for OFNS message	No OFNS message received.	
		(Since parameter CanIfTxPduUserTxConfirmationUL for CanTSyn PDU = CDD, Confirmation shall not reach StbM. This cause transmit confirmation timeout at CanTSyn)	
Step 3	[SUT]	[LT <can>]</can>	
	CanTSyn transmit next periodic OFS message	Receive OFS message with format	
		Byte 0: Type = 0x30	
		Byte 1: reserved = 0x00	
		Byte 2: D = 0x0	

		<u> </u>
		SC = 0x1
		Byte 3: OfsTimeSecLsbHi = 8 Bit offset
		time stamp (LSB) from secondsHi
		стантр (===) поли станта
		Byte 4-7: OfsTimeSecLsbLo =
		StbM_TimeStampType.seconds +
		CanTSynGlobalTimeTxPeriod (2
		second)
		+ CanTSynMasterConfirmationTimeout.
Step 4	[LT <can>]</can>	[LT <can>]</can>
Step 4	[LICANS]	[LICOANS]
	Maita (a. OFNO accessor	N. OFNO
	Waits for OFNS message	No OFNS message received.
		(Cin an argumentar
		(Since parameter CanIfTxPduUserTxConfirmationUL for
		CanTSyn PDU = CDD, Confirmation
		shall not reach StbM. This cause
		transmit confirmation timeout at
		CanTSyn)
Post-	None	
conditions		

## 4.4.5 [ATS\_GTS\_01238] Global Time Master: Handling of time base using NvM (Storage and Retrieve)

Test Objective	Global Time Master: Handling of time base using NvM (Storage and Retrieve)		
ID		AUTOSAR Releases	4.2.1 4.2.2
Affected Modules	StbM	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00131 ATR: ATR_ATR_00132		
Trace to SWS Item	SynchronizedTimeBaseManager: SynchronizedTimeBaseManager:		
Requirements / Reference to Test Environment	Use Case UC01.04		
Configuration Parameters	StbM: StbMMainFunctionPeriod 5ms StbMIsSystemWideGlobalTimeMaster = TRUE StbMStoreTimebaseNonVolatile = STORAGE_AT_SHUTDOWN		
Summary	Aim is to Verify that during Initialization StbM shall load the Time base value from NvM and store the Time base to NvM at shutdown.		
Needed Adaptation to other Releases	Not Applicable		
Pre-conditions	SUT shall be initialized.		
Main Test Execu	Main Test Execution		
Test Steps			Pass Criteria

Step 1	[CP]	
	Start RUN_ GlobalTimeProvider	
Step 2	[RUN <run_ globaltimeprovider="">]</run_>	[RUN <run_ globaltimeprovider="">]</run_>
	Execute Rte_Call_ Client_SetGlobalTime and Rte_Call_ Client_SetUserData with below values:	Rte_Call returns RTE_E_OK
	timeBaseId = 1	
	StbM_TimeStampType.nanoseconds = 0x000000000	
	StbM_TimeStampType.seconds = 0x00000E10	
	StbM_TimeStampType.secondsHi = 0x0000	
	StbM_UserDataType.User Data Byte 0 = xAA	
Step 3	[CP]	
	Wait 100ms	
Step 4	[CP]	
	Start RUN_TimeUser	
Step 5	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time received from Time user shall be as mentioned below:
		StbM_TimeStampType.nanoseconds = 0x00000000 + <testwaittime =<br="">100ms&gt;</testwaittime>
		StbM_TimeStampType.seconds = 0x00000E10
		StbM_TimeStampType.secondsHi = 0x0000
		StbM_UserDataType.User Data Byte 0 = 0xAA
Step 6	[SUT]	[SUT]
	ECU shutdown	Time base shall be stored into NvM during shutdown
Step 7	[SUT]	[SUT]
	Restart the ECU	Time base shall be retrieved from NvM during initialization.

Ctom 0	ICDI	1
Step 8	[CP]	
	Wait 100ms	
Step 9	[CP]	
	Start RUN_TimeUser	
Step 10	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = 0x000000000 + <testwaittime =<br="">100ms&gt; + <testwaittime 100ms="" ==""></testwaittime></testwaittime>
		StbM_TimeStampType.seconds = 0x00000E10
		StbM_TimeStampType.secondsHi = 0x0000
		StbM_UserDataType.User Data Byte 0 = 0xAA
Post- conditions	None	

# 4.4.6 [ATS\_GTS\_01242] Time Slave: Reception of SYNC frames (CRC\_IGNORED) over CAN, Synchronize Local Time Base and share the current time to active customers

Test Objective	Time Slave: Reception of SYNC frames (CRC_IGNORED) over CAN, Synchronize Local Time Base and share the current time to active customers		
ID	ATS_GTS_01242	AUTOSAR Releases	
Affected Modules	StbM, CanTSyn	State	reviewed
Requirement	ATR: ATR_ATR_00131 ATR: ATR_ATR_00132 ATR: ATR_ATR_00133		
Trace to SWS Item	SynchronizedTimeBaseManager: SWS_StbM_00247 TimeSyncOverCAN: SWS_CanTSyn_00011		
Requirements / Reference to Test Environment	Use Case UC01.02		
Configuration Parameters	StbM: StbMSynchronizedTimeBaseIdentifier = 1 CanTSyn:		
	CanTSynRxCrcValidated = CRC_IGNORED		

Summary	Verify that CanTSyn shall call SthM to undate	a global time base on recention of	
Cammary	Verify that CanTSyn shall call StbM to update global time base on reception of SYNC and FUP message even with invalid CRC when CanTSynRxCrcValidated =		
	CRC_IGNORED.	·	
Needed	Not Applicable		
Adaptation to other Releases			
Pre-conditions	SUT shall be initialized.		
Main Test Execu	ution		
Test Steps		Pass Criteria	
Step 1	[LT <can>]</can>	[SUT]	
	Transmit SYNC message with	Receives SYNC message ignoring CRC value (if there) in below format	
	timeBaseId = 1	Byte 0: Type = 0x10	
	StbM_TimeStampType.seconds = 0x00000E10	Byte 1: User Byte 1 = 0xBB	
	StbM_TimeStampType.secondsHi = 0x0000	Byte 2: D = 0x1	
	StbM_UserDataType.User Data Byte 0 = 0xAA	SC = 0x0	
	StbM_UserDataType.User Data Byte 1 =	Byte 3: User Byte 0 = 0xAA	
	0xBB	Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds	
Step 2	[LT <can>]</can>	[SUT]	
	Transmit FUP message with	Receives FUP message ignoring CRC value (if there) in below format	
	StbM_TimeStampType.nanoseconds = t4r	Byte 0: Type = 0x18	
		Byte 1: User Byte 2 = 0x00	
		Byte 2: D = 0x1	
		SC = 0x0	
		Byte 3: reserved (Bit 7 to Bit 3) = 0	
		SGW (Bit 2) = 0	
		OVS = Overflow of seconds (Bit 1 to Bit 0)	
		Byte 4-7: SyncTimeNSec = StbM_TimeStampType.nanoseconds	
Step 3		[SUT]	
		StbM updates its time base with values	
		StbM_TimeStampType.nanoseconds	

		= tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
		Time updated in StbM, tA = tA + ToleranceTime_CanTSyn.
		ToleranceTime_CanTSyn = Max one main function of CanTSyn as reception is asynchronous.
Step 4	[CP]	
	Wait 800ms	
Step 5	[CP]	
	Start RUN_TimeUser	
Step 6	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime =<br="">800ms&gt;</testwaittime>
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Post-	None	
conditions		

## 4.4.7 [ATS\_GTS\_01271] Time Slave: Handling of SYNC message reception timeout (CanTSynGlobalTimeFollowUpTimeout)

	Time Slave: Handling of SYNC message reception timeout (CanTSynGlobalTimeFollowUpTimeout)		
ID		AUTOSAR Releases	4.2.1 4.2.2
Affected Modules	StbM, CanTSyn	State	reviewed
Requirement	ATR: ATR_ATR_00131 ATR: ATR_ATR_00132 ATR: ATR_ATR_00133		
	SynchronizedTimeBaseManager: SWS_StbM_00183 SynchronizedTimeBaseManager: SWS_StbM_00247		

	TimeSyncOverCAN: SWS_CanTSyn_00063			
Requirements / Reference to Test Environment	Use Case UC01.02			
3	StbM: StbMSynchronizedTimeBaseIdentifier = 1			
	CanTSyn: CanTSynRxCrcValidated = CRC_IGNORED			
Summary	To verify that if FUP message are not recieved within 'CanTSynGlobalTimeFollowUpTimeout' CanTSyn shall reset the sequence and wait for new SYNC message.			
	To verify StbM shall set the bit TIMEOUT in Sync state (when UT requests for current time) if StbM_BusSetGlobalTime is not invoked within StbMSyncLossTimeout and Clear the bit on invocation of StbM_BusSetGlobalTime.			
Needed Adaptation to other Releases	Not Applicable			
Pre-conditions	SUT shall be initialized.			
Main Test Execu	ution			
Test Steps		Pass Criteria		
Step 1	[LT <can>]</can>	[SUT]		
	Transmit SYNC message with	Receives SYNC message with format		
	TimeBaseId = 1	Byte 0: Type = 0x10		
	StbM_TimeStampType.seconds = 0x00000E10	Byte 1: User Byte 1 = 0xBB		
	StbM_TimeStampType.secondsHi = 0x0000	Byte 2: D = 0x1		
	StbM_UserDataType.User Data Byte 0 =	SC = 0x0		
	0xAA StbM_UserDataType.User Data Byte 1 =	Byte 3: User Byte 0 = 0xAA		
	0xBB	Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds		
Step 2	[LT <can>]</can>	[SUT]		
	Transmit FUP message with	Receives FUP message with format		
	StbM_TimeStampType.nanoseconds = t4r	Byte 0: Type = 0x18		
		Byte 1: Reserved		
		Byte 2: D = 0x1		
		SC = 0x0		
		Byte 3: reserved (Bit 7 to Bit 3) = 0		

	1	Ja
		SGW (Bit 2) = 0
		OVS = Overflow of seconds (Bit 1 to Bit 0)
		Byte 4-7: SyncTimeNSec = StbM_TimeStampType.nanoseconds
Step 3		[SUT]
		CanTSyn shall update the time base of StbM
		StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
		Time updated in StbM, tA = tA + ToleranceTime_CanTSyn.
		ToleranceTime_CanTSyn = Max one main function of CanTSyn as reception is asynchronous.
Step 4	[CP]	
	Wait 800ms	
Step 5	[CP]	
	Ota d BUN Taxalla a	
Step 6	Start RUN_TimeUser [RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
Ciop C	[	[
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 800ms="" ==""></testwaittime>
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 7	[LT <can>]</can>	[SUT]
	Transmit SYNC message with	Receives SYNC message with format
	TimeBaseId = 1	Byte 0: Type = 0x10
	StbM_TimeStampType.seconds =	Byte 1: User Byte 1 = 0xDD
	1	

	0x00001C20	Byte 2: D = 0x1
	0,00001020	Byto 2. B = 0x1
	StbM_TimeStampType.secondsHi = 0x0000	SC = 0x1
	StbM_UserDataType.User Data Byte 0 = 0xCC	Byte 3: User Byte 0 = 0xCC
	StbM_UserDataType.User Data Byte 1 = 0xDD	Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds
Step 8	[LT <can>]</can>	[SUT]
	Do not transmit FUP message.	Do not receives FUP message
	Reception timeout occurs after 300ms (CanTSynGlobalTimeFollowUpTimeout = 300ms) the sequence is reset and waits for a	StbM_TimeBaseStatusType.TIMEOUT is set to 1.
	new SYNC message.	CanTSyn shall not update the time base of StbM
Step 9	[CP]	
	Wait 800ms	
Step 10	[CP]	
_	Start RUN_TimeUser	
Step 11	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 800ms="" ==""></testwaittime>
		StbM_TimeStampType.seconds = tA .Sec + CanTSynGlobalTimeTxPeriod (2 second)
		StbM_TimeStampType.secondsHi = tA .SecHi
Post- conditions	None	

## 4.4.8 [ATS\_GTS\_01273] Time Slave: Handling of SYNC message Sequence Mismatch failures

Test Objective	Time Slave: Handling of SYNC message Sequence Mismatch failures			
ID	ATS_GTS_01273			
Affected Modules	StbM, CanTSyn State reviewed			

	ATR: ATR_ATR_00131		
	ATR: ATR_ATR_00132		
	ATR: ATR_ATR_00133		
Acceptance			
Test			
Document			
Trace to	SynchronizedTimeBaseManager: S	WS StbM 00247	
SWS Item	TimeSyncOverCAN: SWS_CanTSy		
	TimeSyncOverCAN: SWS_CanTSyn_00078		
Doguiromont	Use Case UC01.02	1_00070	
	Use Case UC01.02		
s/			
Reference to Test			
Environment			
Configuratio			
n	StbMSynchronizedTimeBaseIdentifi	er = 1	
Parameters			
	CanTSyn:		
	CanTSynRxCrcValidated = CRC_IG	NORED	
Summary	To verify that if Sequence Counter of	of SYNC and FUP message are not matching,	
_		SYNC message and shall ignore the received FUP	
	message.	, ,	
	To verify Sequence Counter, Jump V	Vidth between two SYNC messages greater than	
		nterJumpWidth, the messages will be ignored.	
Needed		nerodinipvilatii, tile inlessages wiii be ignorea.	
	Not Applicable		
Adaptation			
to other			
Dalagasa			
Releases			
Pre-	SUT shall be initialized.		
Pre- conditions			
Pre-			
Pre- conditions		Pass Criteria	
Pre- conditions Main Test Ex Test Steps	ecution		
Pre- conditions Main Test Ex		Pass Criteria [SUT]	
Pre- conditions Main Test Ex Test Steps	ecution [LT <can>]</can>	[SUT]	
Pre- conditions Main Test Ex Test Steps	ecution		
Pre- conditions Main Test Ex Test Steps	[LT <can>] Transmit SYNC message with</can>	[SUT] Receives SYNC message with format	
Pre- conditions Main Test Ex Test Steps	ecution [LT <can>]</can>	[SUT]	
Pre- conditions Main Test Ex Test Steps	[LT <can>]  Transmit SYNC message with  TimeBaseId = 1</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10	
Pre- conditions Main Test Ex Test Steps	[LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds =</can>	[SUT] Receives SYNC message with format	
Pre- conditions Main Test Ex Test Steps	[LT <can>]  Transmit SYNC message with  TimeBaseId = 1</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB	
Pre- conditions Main Test Ex Test Steps	[LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10	
Pre- conditions Main Test Ex Test Steps	ECUTION  [LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB	
Pre- conditions Main Test Ex Test Steps	[LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB  Byte 2: D = 0x1	
Pre- conditions Main Test Ex Test Steps	ECUTION  [LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB	
Pre- conditions Main Test Ex Test Steps	ECUTION  [LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB  Byte 2: D = 0x1  SC = 0x0	
Pre- conditions Main Test Ex Test Steps	ECUTION  [LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000  StbM_UserDataType.User Data</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB  Byte 2: D = 0x1	
Pre- conditions Main Test Ex Test Steps	ECUTION  [LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB  Byte 2: D = 0x1  SC = 0x0  Byte 3: User Byte 0 = 0xAA	
Pre- conditions Main Test Ex Test Steps	Ecution  [LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000  StbM_UserDataType.User Data Byte 0 = 0xAA</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB  Byte 2: D = 0x1  SC = 0x0  Byte 3: User Byte 0 = 0xAA  Byte 4-7: SyncTimeSec =	
Pre- conditions Main Test Ex Test Steps	Ecution  [LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000  StbM_UserDataType.User Data Byte 0 = 0xAA  StbM_UserDataType.User Data</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB  Byte 2: D = 0x1  SC = 0x0  Byte 3: User Byte 0 = 0xAA	
Pre- conditions Main Test Ex Test Steps Step 1	ECUTION  [LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000  StbM_UserDataType.User Data Byte 0 = 0xAA  StbM_UserDataType.User Data Byte 1 = 0xBB</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB  Byte 2: D = 0x1  SC = 0x0  Byte 3: User Byte 0 = 0xAA  Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds	
Pre- conditions Main Test Ex Test Steps Step 1	Ecution  [LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000  StbM_UserDataType.User Data Byte 0 = 0xAA  StbM_UserDataType.User Data</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB  Byte 2: D = 0x1  SC = 0x0  Byte 3: User Byte 0 = 0xAA  Byte 4-7: SyncTimeSec =	
Pre- conditions Main Test Ex Test Steps Step 1	[LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000  StbM_UserDataType.User Data Byte 0 = 0xAA  StbM_UserDataType.User Data Byte 1 = 0xBB  [LT<can>]</can></can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB  Byte 2: D = 0x1  SC = 0x0  Byte 3: User Byte 0 = 0xAA  Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds  [SUT]	
Pre- conditions Main Test Ex Test Steps Step 1	ECUTION  [LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000  StbM_UserDataType.User Data Byte 0 = 0xAA  StbM_UserDataType.User Data Byte 1 = 0xBB</can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB  Byte 2: D = 0x1  SC = 0x0  Byte 3: User Byte 0 = 0xAA  Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds	
Pre- conditions Main Test Ex Test Steps Step 1	[LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000  StbM_UserDataType.User Data Byte 0 = 0xAA  StbM_UserDataType.User Data Byte 1 = 0xBB  [LT<can>]</can></can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB  Byte 2: D = 0x1  SC = 0x0  Byte 3: User Byte 0 = 0xAA  Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds  [SUT]	
Pre- conditions  Main Test Ex Test Steps  Step 1	[LT <can>]  Transmit SYNC message with  TimeBaseId = 1  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000  StbM_UserDataType.User Data Byte 0 = 0xAA  StbM_UserDataType.User Data Byte 1 = 0xBB  [LT<can>]</can></can>	[SUT]  Receives SYNC message with format  Byte 0: Type = 0x10  Byte 1: User Byte 1 = 0xBB  Byte 2: D = 0x1  SC = 0x0  Byte 3: User Byte 0 = 0xAA  Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds  [SUT]	

	lata 44.	D. 40 O. T 0.40
	ds = t4r	Byte 0: Type = 0x18
		Byte 1: Reserved
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: reserved (Bit 7 to Bit 3) = 0
		SGW (Bit 2) = 0
		OVS = Overflow of seconds (Bit 1 to Bit 0)
		Byte 4-7: SyncTimeNSec = StbM_TimeStampType.nanoseconds
Step 3		[SUT]
		CanTSyn shall update the time base of StbM
		StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
		Time updated in StbM, tA = tA + ToleranceTime_CanTSyn.
		ToleranceTime_CanTSyn = Max one main function of CanTSyn as reception is asynchronous.
Step 4	[CP]	
	Wait 800ms	
Step 5	[CP]	
	Start RUN_TimeUser	
Step 6	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_	Rte_Call returns RTE_E_OK.
	Client_GetCurrentTime	Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 800ms="" =="">.</testwaittime>
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
		StbM_TimeBaseStatusType.TIMEOUT is set to 0.
Step 7	[LT <can>]</can>	[SUT]
	Transmit next periodic SYNC	

	<b>.</b>	T
	message with	Receives SYNC message with format
	TimeBaseId = 1	Byte 0: Type = 0x10
	StbM_TimeStampType.seconds = 0x00001C20	Byte 1: User Byte 1 = 0xDD
	StbM_TimeStampType.secondsHi	Byte 2: D = 0x1
	= 0x0000	SC = 0x1
	StbM_UserDataType.User Data Byte 0 = 0xCC	Byte 3: User Byte 0 = 0xCC
	StbM_UserDataType.User Data Byte 1 = 0xDD	Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds
Step 8	[LT <can>]</can>	[SUT]
	Transmit FUP message with:	Receives FUP message with format
	StbM_TimeStampType.nanosecon ds = t9r	Byte 0: Type = 0x18
	Different sequence number	Byte 1: Reserved
	Emoronic ocquerios numbor	Byte 2: D = 0x1
		SC = 0x2
		Byte 3: reserved (Bit 7 to Bit 3) = 0
		SGW (Bit 2) = 0
		OVS = Overflow of seconds (Bit 1 to Bit 0)
		Byte 4-7: SyncTimeNSec = StbM_TimeStampType.nanoseconds
		CanTSyn shall not update the time base of StbM
Step 9	[CP]	
	Wait 800ms	
Step 10	[CP]	
	Start RUN_TimeUser	
Step 11	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 800ms="" =="">.</testwaittime>
		StbM_TimeStampType.seconds = tA .Sec +

		CanTSynGlobalTimeTxPeriod (2 second)
		StbM_TimeStampType.secondsHi = tA .SecHi
		StbM_TimeBaseStatusType.TIMEOUT is set to 1.
Step 12	[LT <can>]</can>	[SUT]
	Transmit next periodic SYNC message with	Time base is not updated as per incoming time base as sequence counter value is greater than CanTSynGlobalTimeSequenceCounterJumpW
	timeBaseId = 1	idth.
	StbM_TimeStampType.seconds = 0x00002328	
	StbM_TimeStampType.secondsHi = 0x0000	
	StbM_UserDataType.User Data Byte 0 = 0xEE	
	StbM_UserDataType.User Data Byte 1 = 0xFF	
	Sequence Counter = 3	
Step 13	[CP]	
	Wait 800ms	
Step 14	[CP]	
	Start RUN_TimeUser	
Step 15	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
·	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
	onone_ootounonkriinio	Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 800ms="" ==""> + <testwaittime =<br="">800ms&gt;.</testwaittime></testwaittime>
		StbM_TimeStampType.seconds = tA .Sec + CanTSynGlobalTimeTxPeriod (2 second) + CanTSynGlobalTimeTxPeriod (2 second).
		StbM_TimeStampType.secondsHi = tA .SecHi
		StbM_TimeBaseStatusType.TIMEOUT is set to 1.
Post- conditions	None	

# 4.4.9 [ATS\_GTS\_01286] Time Slave: Reception of Offset frames (CRC\_VALIDATED) over CAN, Synchronize Local Time Base and share the current time to active customers.

Test Objective	Time Slave: Reception of Offset frames (CRC_VALIDATED) over CAN, Synchronize Local Time Base and share the current time to active customers.		
ID		AUTOSAR Releases	4.2.1 4.2.2
Affected Modules	StbM, CanTSyn	State	reviewed
Requirement	ATR: ATR_ATR_00131 ATR: ATR_ATR_00132 ATR: ATR_ATR_00133		
Trace to SWS Item	SynchronizedTimeBaseManager: TimeSyncOverCAN: SWS_CanTS		1_00247
Requirements / Reference to Test Environment	Use Case UC01.02		
Configuration Parameters	StbM: StbMSynchronizedTimeBaseIdent	ifier = 17.	
	$\label{eq:canTSyn:} CanTSyn: \\ CanTSynRxCrcValidated = CRC_VALIDATED \\ CanTSynGlobalTimeOfsDatalDListIndex = \{0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15\} \\ CanTSynGlobalTimeOfsDatalDListValue in ASCII = {"A", "U", "T", "O", "S", "A", "R", "A", "T", "S", "G", "T", "S", "O", "F", "S"} \\ CanTSynGlobalTimeOfnsDatalDListIndex = {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15} \\ CanTSynGlobalTimeOfnsDatalDListValue in ASCII = {"A", "U", "T", "O", "S", "A", "R", "A", "T", "S", "G", "T", "O", "F", "N", "S"} \\$		
Summary	Aim is to:  Verify that CanTSyn shall Receive the offset time periodically via OFS and OFNS message respectively with CRC validation.  Verify that StbM shall synchronize its local offset time base on reception of Time Base from CanTSyn Verify that UT shall get the valid current time, offset time, current time in extended format, user data using Client-Server Interface.		
Needed Adaptation to other Releases	Not Applicable		
Pre-conditions	SUT shall be initialized.		
Main Test Execu	ution		
Test Steps	Pass Criteria		
Step 1	[LT <can>]</can>		[SUT]
	Transmit SYNC and FUP message	e with:	Get the time stamp values with CRC validation as below:
	TimeBaseId = 1 StbM_TimeStampType.nanosecor 0x00000000	nds =	StbM_TimeStampType.nanoseconds = tA.Nano
	StbM_TimeStampType.seconds =		StbM_TimeStampType.seconds = tA

	0x00000E10 (3600d)	.Sec
	StbM_TimeStampType.secondsHi = 0x0000	StbM_TimeStampType.secondsHi = tA .SecHi
		Time updated in StbM, tA = tA + ToleranceTime_CanTSyn.
		ToleranceTime_CanTSyn = Max one main function of CanTSyn as reception is asynchronous.
Step 2	[CP]	
	Wait 800ms	
Step 3	[LT <can>]</can>	[SUT]
	Transmit OFS and OFNS message with:	Get the offset time stamp values with CRC validation as below:
	TimeBaseId = 17	StbM_TimeStampType.nanoseconds
	StbM_TimeStampType.nanoseconds = 0x00000000	= tA.Nano
	StbM_TimeStampType.seconds = 0x00000010 (16d)	StbM_TimeStampType.seconds = tA .Sec
	StbM_TimeStampType.secondsHi = 0x0000	StbM_TimeStampType.secondsHi = tA .SecHi
Step 4	[CP]	
	Wait 800ms	
Step 5	[CP]	
	Start RUN_TimeUser	
Step 6	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime =<br="">800ms&gt; + <testwaittime 800ms="" ==""></testwaittime></testwaittime>
		StbM_TimeStampType.seconds = tA .Sec + <offset time="0x00000010"></offset>
		StbM_TimeStampType.secondsHi = tA .SecHi
Post- conditions	None	

# 4.4.10 [ATS\_GTS\_01244] Time Slave: Reception of Offset frames (CRC\_NOT\_VALIDATED) over CAN, Synchronize Local Time Base and share the current time to active customer

Test Objective	Time Slave: Reception of Offset frames (CRC_NOT_VALIDATED) over CAN, Synchronize Local Time Base and share the current time to active customer		
ID	ATS_GTS_01244 AI		4.2.1 4.2.2
Affected Modules	StbM, CanTSyn	ate	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00131 ATR: ATR_ATR_00132 ATR: ATR_ATR_00133		
Trace to SWS	SynchronizedTimeBaseManager: S TimeSyncOverCAN: SWS_CanTSy		I_00247
Requirements / Reference to Test Environment	Use Case UC01.02		
Configuration Parameters	StbM: StbMSynchronizedTimeBaseIdentifi	er = 16	
	CanTSyn: CanTSynRxCrcValidated = CRC_N	OT_VALII	DATED
Summary	Aim is to:		
	Verify that CanTSyn shall Recieve the offset time periodically via OFS and OFNS message respectively.  Verify that StbM shall synchronize its local offset time base on receiption of Time Base from CanTSyn  Verify that UT shall shall get the valid current time, offset time, current time in		
Needed Adaptation to other Releases	extended format, user data using Client-Server Interface. Not Applicable		
Pre-conditions	SUT shall be initialized.		
Main Test Exec	ution		
Test Steps	b. =		Pass Criteria
Step 1	[LT <can>]</can>		[SUT]
	Transmit SYNC and FUP message	with:	Get the time stamp values as below:
	TimeBaseId = 1		StbM_TimeStampType.nanoseconds = tA.Nano
	StbM_TimeStampType.nanosecond 0x00000000	ls =	StbM_TimeStampType.seconds = tA .Sec
	StbM_TimeStampType.seconds = 0x00000E10 (3600d)		StbM_TimeStampType.secondsHi = tA .SecHi
	StbM_TimeStampType.secondsHi =	= 0x0000	Time updated in StbM, tA = tA +

## AUTOSAR TC Release 1.2.0

		ToleranceTime_CanTSyn.
		ToleranceTime_CanTSyn = Max one main function of CanTSyn as reception is asynchronous.
Step 2	[CP]	, ,
	Wait 800ms	
Step 3	[LT <can>]</can>	[SUT]
	Transmit OFS and OFNS message with:	Get the offset time stamp values as below:
	TimeBaseId = 16	
	StbM_TimeStampType.nanoseconds =	StbM_TimeStampType.nanoseconds = tA.Nano
	0x00000000	StbM_TimeStampType.seconds = tA
	StbM_TimeStampType.seconds = 0x0000000A (10d)	.Sec
	, ,	StbM_TimeStampType.secondsHi =
	OtbW_11110Otamp1ype.seconds111 = 0x0000	tA .SecHi
Step 4	[CP]	
	Wait 800ms	
Step 5	[CP]	
	Stort DUN Timel lear	
Step 6	Start RUN_TimeUser [RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
Step 0		
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime =<br="">800ms&gt; + <testwaittime 800ms="" ==""></testwaittime></testwaittime>
		StbM_TimeStampType.seconds = tA .Sec + <offset time="0x0000000A"></offset>
		StbM_TimeStampType.secondsHi = tA .SecHi
Post- conditions	None	

## 4.4.11 [ATS\_GTS\_01272] Time Slave: Handling of Offset message reception timeout (CanTSynGlobalTimeFollowUpTimeout)

	Time Slave: Handling of Offset message reception timeout (CanTSynGlobalTimeFollowUpTimeout)		
ID	ATS_GTS_01272 AUTOSAR 4.2.1 4.2.2 Releases		

Affected	StbM, CanTSyn	tate	reviewed
Modules	ATD, ATD, ATD, COACA		
Trace to Requirement	ATR: ATR_ATR_00131 ATR: ATR_ATR_00132		
	ATR: ATR_ATR_00132 ATR: ATR_ATR_00133		
Test Document			
Trace to SWS Item	SynchronizedTimeBaseManager: S TimeSyncOverCAN: SWS_CanTS		I_00247
	Use Case UC01.02		
Reference to Test			
Environment			
Configuration	StbM:		
Parameters	StbMSynchronizedTimeBaseIdenti	fier = 16	
	CanTSyn: CanTSynRxCrcValidated = CRC_l	GNORFD	
Summary	To verify that if OFNS message are		ved within
- anninary			Syn shall reset the sequence and wait
	for new OFS message.		·
		FOUT: 5	Consider A to a 19 <del>-</del>
	To verify StbM shall set the bit TIM current time) if StbM_BusSetGloba		
			nvocation of StbM_BusSetGlobalTime.
Needed	Not Applicable		
Adaptation to			
other Releases			
Pre-conditions	SUT shall be initialized.		
	StbM shall be initialized with base time: StbM_TimeStampType.nanoseconds = 0x00000000		
	StbM_TimeStampType.seconds = 0x00000000		
	StbM_TimeStampType.secondsHi	= 0x0000	
Main Test Exec	ution		
Test Steps			Pass Criteria
Step 1	[LT <can>]</can>		[SUT]
	Transmit OFS message with		Receives OFS message with format
	Transmit Of O message with		1.0001103 Of O message with format
	TimeBaseId = 16		Byte 0: Type = 0x30
	StbM_TimeStampType.seconds		Byte 1: Reserved
	=0x00000014		Byto 2: D = 0v0
	Cth M. Time o Ctame n Time a case on del li		Byte 2: D = 0x0
	StDIVE TIMEStamb Evbe.seconos Hi	- UNUUNN •	
	StbM_TimeStampType.secondsHi	_ 0,0000	SC = 0x0
	Stow_1 meStamp1ype.secondsHi		
	Stolvi_TimeStampType.secondsHi		Byte 3: OfsTimeSecLsbHi = 8 Bit
	Stolvi_1 imeStamp1 ype.secondsHi		
	Stolvi_1 imeStamp1 ype.secondsHi		Byte 3: OfsTimeSecLsbHi = 8 Bit offset time stamp (LSB) from
	Stolvi_1 imeStamp1 ype.secondsHi		Byte 3: OfsTimeSecLsbHi = 8 Bit offset time stamp (LSB) from secondsHi  Byte 4-7: OfsTimeSecLsbLo =
			Byte 3: OfsTimeSecLsbHi = 8 Bit offset time stamp (LSB) from secondsHi  Byte 4-7: OfsTimeSecLsbLo = StbM_TimeStampType.seconds
Step 2	[LT <can>]</can>		Byte 3: OfsTimeSecLsbHi = 8 Bit offset time stamp (LSB) from secondsHi  Byte 4-7: OfsTimeSecLsbLo =

	Transmit OFNS message with	Receives OFNS message with format
	StbM_TimeStampType.nanoseconds = t4r	Byte 0: Type = 0x38
		Byte 1: Reserved
		Byte 2: D = 0x0
		SC = 0x0
		Byte 3: OfsTimeSecMsbHi = 8 Bit offset time stamp (MSB) from secondsHi
		Byte 4-7: OfsTimeNSec = StbM_TimeStampType.nanoseconds
Step 3		[SUT]
		CanTSyn shall update the time base of StbM
		StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 4	[CP]	
Step 5	Wait 800ms [CP]	
Сібр		
Store C	Start RUN_TimeUser	IDUN DUN Timelleen 1
Step 6	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 800ms="" ==""></testwaittime>
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 7	[LT]	[SUT]
	Transmit OFS message with	Receives OFS message with format

	TimeBaseId = 16	Byte 0: Type = 0x30
	StbM_TimeStampType.seconds =0x00000032	Byte 1: Reserved
	StbM_TimeStampType.secondsHi = 0x0000	Byte 2: D = 0x0
	Statin_TimeStampType.SecondsTill = 0x0000	SC = 0x1
		Byte 3: OfsTimeSecLsbHi = 8 Bit offset time stamp (LSB) from secondsHi
		Byte 4-7: OfsTimeSecLsbLo = StbM_TimeStampType.seconds
Step 8	[LT]	[SUT]
	Do not transmit OFNS message	No OFNS message received
	Reception timeout occurs after 300ms (CanTSynGlobalTimeFollowUpTimeout = 300ms) the sequence is reset and waits for a	StbM_TimeBaseStatusType.TIMEOUT is set to 1.
	new OFS message.	CanTSyn shall not update the time base of StbM
Step 9	[CP]	
	Wait 800ms	
Step 10	[CP]	
	Start RUN_TimeUser	
Step 11		[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 800ms="" ==""></testwaittime>
		StbM_TimeStampType.seconds = tA .Sec + CanTSynGlobalTimeTxPeriod (2 second)
		StbM_TimeStampType.secondsHi = tA .SecHi
Post- conditions	None	

## 4.4.12 [ATS\_GTS\_01274] Time Slave: Handling of Offset message Sequence Mismatch failure

Test	Time Slave: Handling of Offset message Sequence Mismatch failure
Objective	

ID	ATS_GTS_01274	AUTOSA	4.2.1 4.2.2
	, <b>5_5</b>	R	,
A 66	0.111.0 = 70	Releases	
Affected Modules	StbM, CanTSyn	State	reviewed
Trace to	ATR: ATR_ATR_00131		
	ATR: ATR_ATR_00132		
on Acceptance	ATR: ATR_ATR_00133		
Test			
Document	0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	0144	0.00 M. 000 47
Irace to SWS	SynchronizedTimeBaseMa TimeSyncOverCAN: SWS_		
	TimeSyncOverCAN: SWS_		
	Use Case UC01.02		
s / Reference to Test			
Environment			
	StbM:		
n Parameters	StbMSynchronizedTimeBa	seldentifier	= 16
	CanTSyn:		
	CanTSynRxCrcValidated =		
Summary			OFS and OFNS message are not matching,
	can i Syn shall discard the message.	received O	FS message and shall ignore the received OFNS
	incoodge.		
	To verify Sequence Counter Jump Width between two OFS messages greater than		
Needed	CanTSynGlobalTimeSequenceCounterJumpWidth, the messages will be ignored.  Not Applicable		
Adaptation to			
other			
Releases Pre-	CLIT abolt be initialized		
conditions	SUT shall be initialized. StbM shall be initialized with base time:		
	StbM_TimeStampType.nanoseconds = 0x00000000		
	StbM_TimeStampType.sed StbM_TimeStampType.sed		
Main Test Exe		7011d0111 — C	7,0000
Test Steps			Pass Criteria
Step 1	[LT <can>]</can>		[SUT]
	Transmit OFS message wi	th	Receives OFS message with format
	Transmit of o message wi	ui	Receives of a message with format
	TimeBaseId = 16		Byte 0: Type = 0x30
	StbM_TimeStampType.sed 0x00000014	conds =	Byte 1: Reserved
	0/1 M T' 0: T	,	Byte 2: D = 0x0
	StbM_TimeStampType.sed 0x0000	condsHi =	SC 0v0
	0.0000		SC = 0x0
			Byte 3: OfsTimeSecLsbHi = 8 Bit offset time
			stamp (LSB) from secondsHi

		b
		Byte 4-7: OfsTimeSecLsbLo = StbM_TimeStampType.seconds
Step 2	[LT <can>]</can>	[SUT]
	Transmit OFNS message with	Receives OFNS message with format
	StbM_TimeStampType.nanoseconds = t4r	Byte 0: Type = 0x38
	- (4)	Byte 1: Reserved
		Byte 2: D = 0x0
		SC = 0x0
		Byte 3: OfsTimeSecMsbHi = 8 Bit offset time stamp (MSB) from secondsHi
		Byte 4-7: OfsTimeNSec = StbM_TimeStampType.nanoseconds
Step 3		[SUT]
		CanTSyn shall update the offset time base of StbM
		StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 4	[CP]	
	Wait 800ms	
Step 5	[CP]	
	Start RUN_TimeUser	
Step 6	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
	Ollett_OetOutrentTime	Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 800ms="" =="">.</testwaittime>
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeBaseStatusType.TIMEOUT is set to 0.
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 7	[LT <can>]</can>	[SUT]
	Transmit next periodic OFS message	Receives OFS message with format
		I .

	u sith	Duta O. Tura Ov20
	with	Byte 0: Type = 0x30
	TimeBaseld = 16	Byte 1: Reserved
	StbM_TimeStampType.seconds = 0x00000032	Byte 2: D = 0x0
	StbM_TimeStampType.secondsHi =	SC = 0x1
	0x0000	Byte 3: OfsTimeSecLsbHi = 8 Bit offset time stamp (LSB) from secondsHi
		StbM_TimeStampType.seconds
Step 8	[LT <can>]</can>	[SUT]
	Transmit OFNS message with different sequence number	Receives OFNS message with format
		Byte 0: Type = 0x38
		Byte 1: Reserved
		Byte 2: D = 0x0
		SC = 0x2
		Byte 3: OfsTimeSecMsbHi = 8 Bit offset time stamp (MSB) from secondsHi
		Byte 4-7: OfsTimeNSec = t4
		CanTSyn shall not update the offset time base of StbM
Step 9	[CP]	
	Wait 800ms	
Step 10	[CP]	
	Start RUN_TimeUser	
Step 11	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
	Onent_Oetourentrime	Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 800ms="" =="">.</testwaittime>
		StbM_TimeStampType.seconds = tA .Sec + CanTSynGlobalTimeTxPeriod (2 second)
		StbM_TimeStampType.secondsHi = tA .SecHi
		StbM_TimeBaseStatusType.TIMEOUT is set to 1.

Step 12	[LT]	[SUT]
	Transmit next periodic OFS message with	Time base is not updated as per new incoming time base as sequence counter value is greater than
	TimeBaseId = 16	CanTSynGlobalTimeSequenceCounterJumpWidt h
	StbM_TimeStampType.seconds = 0x00000050	
	StbM_TimeStampType.secondsHi = 0x0000	
	Sequence Counter = 3	
Step 13	[CP]	
	Wait 800ms	
Step 14	[CP]	
	Start RUN_TimeUser	
Step 15	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 800ms="" ==""> + <testwaittime =<br="">800ms&gt;.</testwaittime></testwaittime>
		StbM_TimeStampType.seconds = tA .Sec + CanTSynGlobalTimeTxPeriod (2 second) + CanTSynGlobalTimeTxPeriod (2 second
		StbM_TimeStampType.secondsHi = tA .SecHi
		StbM_TimeBaseStatusType.TIMEOUT is set to 1.
Post- conditions	None	

## 4.4.13 [ATS\_GTS\_01270] Time Slave: Synchronize Runnable entities to time base

Test Objective	Time Slave: Synchronize Runnable entities to time base		
ID		AUTOSAR Releases	4.2.1 4.2.2
		Neicases	
Affected	StbM, CanTSyn	State	reviewed
Modules	,		
Trace to	ATR: ATR ATR 00132	-	
Requirement			
on Acceptance			

	T		
Test Document			
Trace to SWS	SynchronizedTimeBaseManager: SWS_StbM_00020		
Item	SynchronizedTimeBaseManager: SWS_StbN	1_00247	
<u>-</u>	Use Case UC01.03		
Reference to Test			
Environment			
	C4b N A		
Configuration Parameters	StbM: StbMMainFunctionPeriod 5ms		
i arameters	StbMSynchronizedTimeBaseIdentifier = 1		
	StbMTriggeredCustomerPeriod 10ms		
	StbMOSScheduleTableRef = Reference to O	S ScheduleTable	
	StbMSynchronizedTimeBaseRef = Reference	to StbMSynchronizedTimeBase	
	CanTSyn:		
	CanTSynRxCrcValidated = CRC_IGNORED		
	OS:		
	ScheduleTableID = 1		
Summary	Aim is to verify that StbM shall synchronize th	e OS schedule tables	
· · · · · · · · · · · · · · · · · · ·	(StbMOSScheduleTableRef) with updated tim		
Needed	Not Applicable		
Adaptation to			
other Releases			
Pre-conditions	SUT shall be initialized.		
Main Test Execu	ution		
Test Steps		Pass Criteria	
Step 1	[LT]	[SUT]	
•	-	· ·	
	Transmit SYNC message with	Receives SYNC message with format	
	TimeBaseId = 1	Byte 0: Type = 0x10	
	StbM_TimeStampType.seconds =	Byte 1: User Byte 1 = 0xBB	
	0x00000E10		
	Cth M. Time Ctamen Time accorded to 0x0000	Byte 2: D = 0x1	
	StbM_TimeStampType.secondsHi = 0x0000	00 00	
	CthM LlaarDataTypa Llaar Data Byta 0	SC = 0x0	
	StbM_UserDataType.User Data Byte 0 = 0xAA	Dito 2: Hoor Dito 0 Ov A A	
		Byte 3: User Byte 0 = 0xAA	
	StbM_UserDataType.User Data Byte 1 =	Byte 4-7: SyncTimeSec =	
	0xBB	StbM_TimeStampType.seconds	
Step 2	[LT]	ISUT1	
0.10p _	i 1	[66.]	
	Transmit FUP message with	Receives FUP message with format	
	Ĭ	Ü	
	StbM_TimeStampType.nanoseconds =	Byte 0: Type = 0x18	
	0x0000000		
		Byte 1: Reserved	
		Byte 1: Reserved Byte 2: D = 0x1	
		Byte 2: D = 0x1	

## AUTOSAR TC Release 1.2.0

		Byte 3: reserved (Bit 7 to Bit 3) = 0
		SGW (Bit 2) = 0
		, ,
		OVS = Overflow of seconds (Bit 1 to Bit 0)
		Byte 4-7: SyncTimeNSec = t4r
Step 3		[SUT]
		StbM updates its time base in 500ms with values
		StbM_TimeStampType.nanoseconds = tA.Nano + 500ms
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
		Time updated in StbM, tA = tA + ToleranceTime_CanTSyn.
		ToleranceTime_CanTSyn = Max one main function of CanTSyn as reception is asynchronous.
Step 4	[SUT]	[SUT]
	StbM synchronizes OS schedule tables	StbM shall invoke SyncScheduleTable(
		<scheduletableid 1="" =="">, <tick_value>)</tick_value></scheduletableid>
Post- conditions	None	

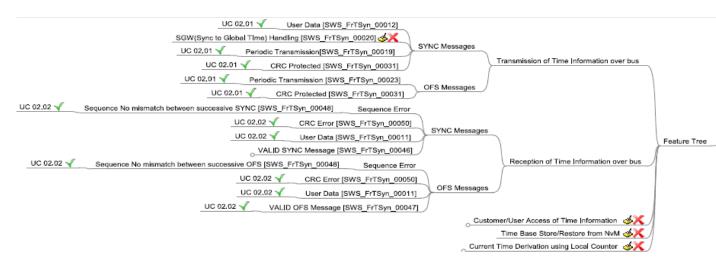
## 5 RS\_BRF\_01660 - Global Time Synchronization over FlexRay

#### 5.1 General Test Objective and Approach

This Test Specification intends to cover the Global Time Synchronization feature of StbM and FrTSyn as described in the AUTOSAR Feature [RS\_BRF\_01660].

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:



Below Features are not tested in this test suite:

- SGW(Sync to Global Time) Handling [SWS\_FrTSyn\_00020]: Feature is tested in the test suite 'Global Time Synchronization over Multiple Bus'.
- Customer/User Access of Time Information, Time Base Store/Restore from NvM, Current Time Derivation using Local Counter: These features are tested in test suite 'Global Time Synchronization over CAN'.

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

#### 5.1.1 Test System

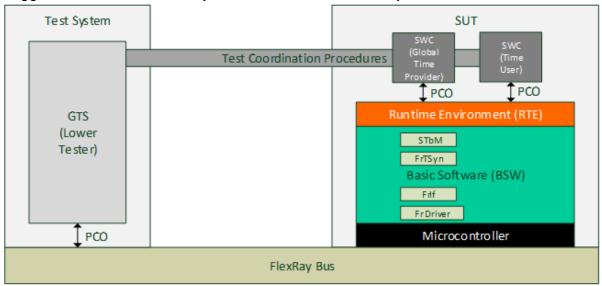
#### **5.1.1.1 Overview on Architecture**

In order to cover the required features / sub-features coverage, the environment has been separated in several use cases.

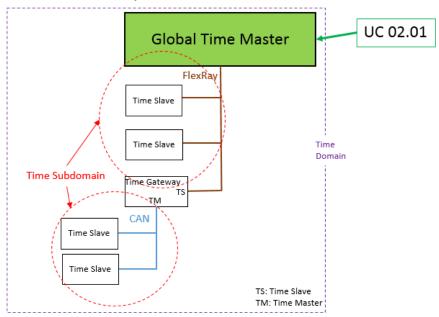
Test Cases are derived based on below use cases

#### 5.1.1.1.1 UC 02.01: Global Time Master over FlexRay

SUT acts as Global time Master and Sets time base, offset time base, User Data and Trigger for transmission of Synchronization over FlexRay bus.

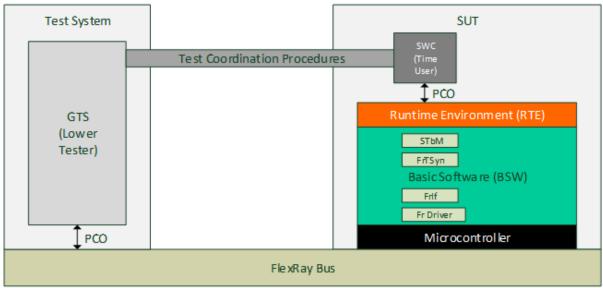


As shown in below figure, Functionalities of Global Time master of a time domain are tested over FlexRay in the use case 02.01

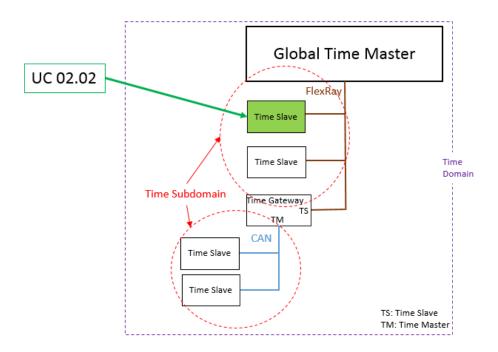


#### 5.1.1.1.2 UC 02.02: Time Slave over FlexRay

SUT acts as Time Slave and Gets time base, offset time base and Synchronizes Local time to Global Time base.



As shown in below figure, Functionalities of Time Slave of a time domain are tested over FlexRay in the use case 02.02



#### 5.1.1.2 Specific Requirements

Not Applicable.

#### **5.1.1.3 Test Coordination Requirements**

UC 01.01: Global Time Master over FlexRay

 Test System (LT <FlexRay>) shall read the FrTSyn FlexRay Frames and decode the same as per Frame Format provided in AUTOSAR\_SWS\_TimeSyncOverFlexRay.

#### UC 01.02: Time Slave over FlexRay

 Test System (LT <FlexRay>) shall encode the FrTSyn FlexRay Frames as per Frame Format provided in AUTOSAR\_SWS\_TimeSyncOver FlexRay and transmit over bus.

#### Requirements for CRC Calculation

 Test System (LT < FlexRay >) shall use the Crc\_CalculateCRC8H2F() (Refer AUTOSAR Specification of CRC Routines AUTOSAR\_SWS\_CRCLibrary.pdf) to calculate the CRC of the Frame. Below are the parameters used for CRC calculation:

The CRC start value shall be 0xFF.

The CRC final XOR-value shall be 0xFF.

The CRC polynomial shall be 0x2F.

The DataIDList shall be same as provided in FrTSyn Configuration.

#### 5.1.2 Test Case Design

Below diagrams explain test design for different use cases

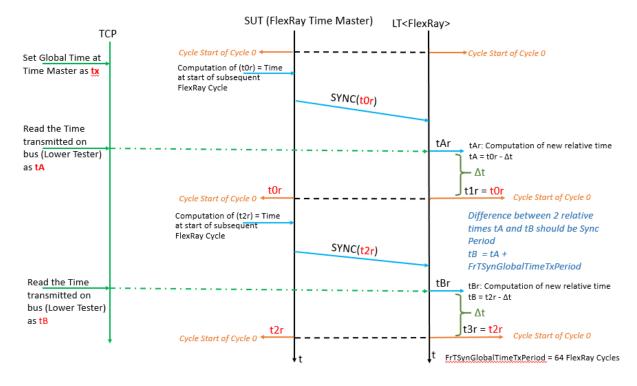


Fig: UC 02.01 - Global Time Master over FlexRay Test Design

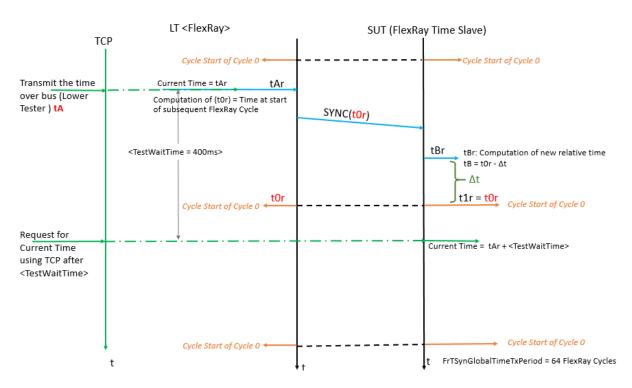


Fig: UC 02.02 - Time Slave over FlexRay Test Design

#### **5.2 Configuration requirements**

The configuration can be divided into two separate parts. The *test configuration* describes variables used to parameterize the test case. The *static configuration* describes the necessary settings of the DUT in order to allow a test case to perform.

#### 5.2.1 Test Configuration

Communication data base for FrTSyn is depicted below

test configuration parameters		
Parameter	Value	
Frlf Tx Pdu	AT_201_lpdu201	
FrlfTxPdu. Frlflmmediate	FALSE	
FrlfFrameTriggering.FrlfBaseCycle	0	
FrlfFrameTriggering.FrlfCycleRepetition	64	
FrlfFrameTriggering.FrlfSlotId	2	
Frlf Rx Pdu	AT_202_lpdu202	

#### **5.2.2 Static Configuration**

#### **5.2.2.1 Static Configuration Groups**

SCG_ATS_GlobalTimeSync_Time Master	
System Description Parameters	
StbM	
SystemTemplate::GlobalTime::GlobalTimeMaster.isSystemWideGlobalTimeMaster	TRUE

FrTSyn	
SystemTemplate::GlobalTime::GlobalTimeDomain.domainId	1
SystemTemplate::GlobalTime::GlobalTimeMaster.syncPeriod	320ms
SystemTemplate::GlobalTime::GlobalTimeDomain.subDomain	Ref. To PDU
ECU Configuration Parameters	
StbM	
StbM.StbMGeneral.StbMMainFunctionPeriod	5ms
StbM.StbMSynchronizedTimeBase.StbMLocalTimeRef	Ref. to
	OSCounter
FrTSyn	
FrTSyn.FrTSynGeneral.FrTSynMainFunctionPeriod	5ms
FrTSyn.FrTSynGlobalTimeDomain.	Ref. to StbM
FrTSynSynchronizedTimeBaseRef	time base
FrTSyn.CanTSynGlobalTimeDomain.FrTSynGlobalTimeMaster.	0
FrTSynGlobalTimeMasterPdu.	
FrTSynGlobalTimeMasterConfirmationHandleId	
Test Cases	
UC 02.01	

SCG_ATS_GlobalTimeSync_Time Slave	
System Description Parameters	
StbM	
SystemTemplate::GlobalTime::GlobalTimeMaster.isSystemWideGl	FALSE
obalTimeMaster	
FrTSyn	
SystemTemplate::GlobalTime::GlobalTimeDomain.subDomain	Ref. To PDU
ECU Configuration Parameters	
StbM	
StbM.StbMGeneral.StbMMainFunctionPeriod	5ms
StbM.StbMSynchronizedTimeBase.StbMLocalTimeRef	Ref. to
	OSCounter
FrTSyn	
FrTSyn.FrTSynGeneral.FrTSynMainFunctionPeriod	5ms
FrTSyn.FrTSynGlobalTimeDomain.FrTSynGlobalTimeDomainId	1
FrTSyn.FrTSynGlobalTimeDomain.	1
FrTSynGlobalTimeSequenceCounterJumpWidth	
FrTSyn.FrTSynGlobalTimeDomain.	Ref. to StbM
FrTSynSynchronizedTimeBaseRef	time base
FrTSyn.CanTSynGlobalTimeDomain.FrTSynGlobalTimeSlave.	1
FrTSynGlobalTimeSlavePdu.FrTSynGlobalTimeSlaveHandleId	
Test Cases	

UC 02.02

#### 5.2.2.2 Required System Description

Refer Section 4.2.2.1

#### 5.2.2.3 Required ECU Configuration

Refer Section 4.2.2.1

#### 5.2.2.4 Required Software Components

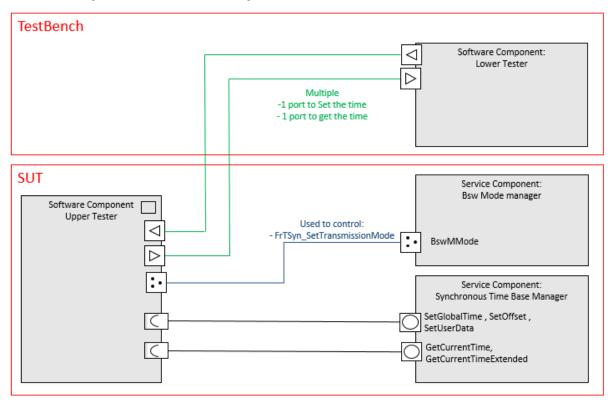


Fig: UC 02.01 - Global Time Master over FlexRay SWC Overview

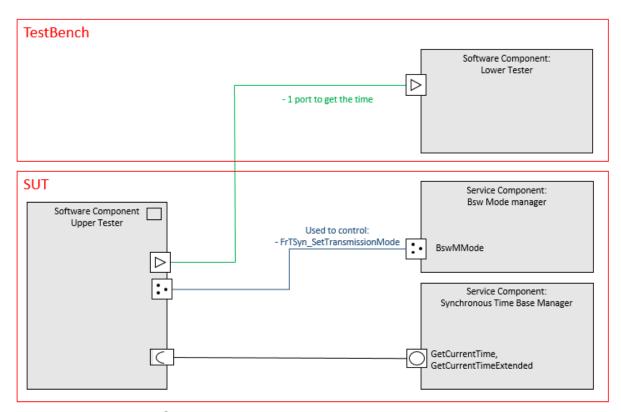


Fig: UC 02.02 - Time Slave over FlexRay SWC Overview

#### 5.2.2.4.1 SWC Client GlobalTime Provider

SWC Name	GlobalTime_Provider		
	Name	Client_SetGlobalTime	
	Туре	RPortPrototype	
	Interface	GlobalTime_Master_Interface	
	Requirements		
	Name	Client_SetUserData	
PORTS	Туре	RPortPrototype	
	Interface	GlobalTime_Master_Interface	
	Requirements		
	Name	Client_SetOffset	
	Туре	RPortPrototype	
	Interface	GlobalTime_Master_Interface	
	Requirements		

	Name	Name RUN_GlobalTimeProvider	
	Requirements	Runnable shall I	pe invoked by TCP
		Name	sscp_GlobalTimeProvider
		Туре	SynchronousServerCallPoint
RUNNABLE ENTITIES		Access to	Client_SetGlobalTime (Write operation)
	ServerCallPoint		Client_SetUserData (Write operation)
			Client_SetOffset (Write operation)
		Requirements	

#### 5.2.2.4.2 SWC Client Time\_User

SWC Name	Time_User			
	Name	Client_GetCurrentTime		
	Туре	RPortPrototype		
	Interface	GlobalTime_Slave_Interface		
	Requirements			
PORTS				
	Name	Client_GetCurre	entTimeExtended	
	Туре	RPortPrototype		
	Interface	GlobalTime_Sla	GlobalTime_Slave_Interface	
	Requirements			
	Name	RUN_TimeUser	•	
	Requirements	Runnable shall	be invoked by TCP	
		Name sscp_TimeUser		
		Туре	SynchronousServerCallPoint	
RUNNABLE ENTITIES	ServerCallPoint	Access to	Client_GetCurrentTime (Read operation) Client_GetCurrentTimeExtended (Read operation)	

	Requirements	

#### 5.2.2.4.3 SWC Server StbM

SWC Name	StbM			
	Name	Server_SetGlobalTime		
	Туре	PPortPrototype		
	Interface	GlobalTime_Master_Interface		
	Requirements			
	Name	Server_SetOffset		
	Туре	PPortPrototype		
	Interface	GlobalTime_Master_Interface		
	Requirements			
	Name	Server_SetUserData		
PORTS	Туре	PPortPrototype		
	Interface	GlobalTime_Master_Interface		
	Requirements			
	Name	Server_GetCurrentTime		
	Туре	PPortPrototype		
	Interface	GlobalTime_Slave_Interface		
	Requirements			
	Name	Server_GetCurrentTimeExtended		
	Туре	PPortPrototype		
	Interface	GlobalTime_Slave_Interface		
	Requirements			
	Name	SthM SatClahalTima		
	ivame	StbM_SetGlobalTime		
	Requirements			
		Name OIE_ SetGlobalTime		
65 of 122		Document ID 841: AUTOSAR_ATS_GlobalTimeSynchronization		

	Started by Event	Туре	OperationInvokedEvent Port:Server_SetGlobalTime
	•	_	Operation: Read
		Requirements	
	Name	StbM_SetOffset	
	Requirements		
RUNNABLE		Name	OIE_SetOffset
ENTITIES	ServerCallPoint	Туре	OperationInvokedEvent
			Port: Server_SetOffset
		Requirements	Operation: Read
	Name	StbM_SetUserData	
	Requirements		
		Name	OIE_ SetUserData
	Started by Event	Туре	OperationInvokedEvent
			Port: Server_SetUserData
		Requirements	Operation: Read
		0.114	
	Name	StbM_GetCurre	nt i ime
	Requirements		
		Name	OIE_ GetCurrentTime
	Started by Event	Туре	OperationInvokedEvent
			Port: Server_GetCurrentTime
		Requirements	Operation: Read
		Requirements	
	Name	StbM_GetCurre	ntTimeExtended
	Requirements		
		Name	OIE_GetCurrentTimeExtended

Started by Event	Туре	OperationInvokedEvent
		Port:
		Server_GetCurrentTimeExtended
		Operation: Read
	Requirements	
	•	

#### 5.3 Re-usable Test Steps

Not Applicable

#### 5.4 Test Cases

# 5.4.1 [ATS\_GTS\_01275] Global Time Master: Setting of Global Time base and user data by Active Customer and sending of SYNC frames (CRC\_NOT\_SUPPORTED) over FlexRay

(*****	101_0011 01(12b) 01c11		
	Global Time Master: Setting of Global Time base and user data by Active Customer and sending of SYNC frames (CRC_NOT_SUPPORTED) over FlexRay		
ID	ATS_GTS_01275	AUTOSAR Releases	4.2.1 4.2.2
Affected Modules	StbM, FrTSyn	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00134		
Trace to SWS Item	TimeSyncOverFlexRay: SWS_Fr	TSyn_00023	3
Requirements / Reference to Test Environment	Use Case UC02.01		
Configuration Parameters	StbM: StbMSynchronizedTimeBaseIdentifier = 1 FrTSyn:		
	FrTSynGlobalTimeTxCrcSecured = CRC_NOT_SUPPORTED CanTSynGlobalTimeSyncDatalDListIndex = {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15} CanTSynGlobalTimeSyncDatalDListValue in ASCII = {"A", "U", "T", "O", "S", "A", "R", "A", "T", "S", "G", "T", "S", "S", "Y", "N"} CanTSynGlobalTimeFupDatalDListIndex = {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15}		
	CanTSynGlobalTimeFupDataIDListValue in ASCII = {"A", "U", "T", "O", "S", "A", "R", "A", "T", "S", "G", "T", "S", "F", "U", "P"}		
Summary	Aim is to:  Verify that StbM accepts the global time base and user data from SWC Global Time Provider using client-server Interface.  Verify that FrTSyn shall transmit the global time base to time slave periodically via		
Needed Adaptation to	SYNC message Not Applicable		

other Releases						
	Pre-conditions SUT shall be initialized.					
	Main Test Execution					
Test Steps		Pass Criteria				
Step 1	[CP]					
	O DUNI OLI IT. D					
	Start RUN_ GlobalTimeProvider.	TOUR DUR ClabelTime President				
Step 2	[RUN <run_ globaltimeprovider="">]</run_>	[RUN <run_ globaltimeprovider="">]</run_>				
	Execute Rte_Call_ Client_SetGlobalTime and Rte_Call_ Client_SetUserData with below values:	Rte_Call returns RTE_E_OK.				
	timeBaseId = 1					
	StbM_TimeStampType.nanoseconds = 0x00000000					
	StbM_TimeStampType.seconds = 0x00000E10					
	StbM_TimeStampType.secondsHi = 0x0000					
StbM_UserDataType.User Data Byte 0 = 0xAA  StbM_UserDataType.User Data Byte 1 = 0xBB						
	StbM_UserDataType.User Data Byte 2 = 0xCC					
Step 3	[CP]					
	Wait 100ms.					
Step 4 [CP]						
Start RUN_TimeUser.						
		[RUN <run_timeuser>]</run_timeuser>				
	Execute Rte_Call_ Client_GetCurrentTime.	Rte_Call returns RTE_E_OK.				
		Time received from Time user shall be as mentioned below:				
		StbM_TimeStampType.nanoseconds = 0x000000000 + <testwaittime =<br="">100ms&gt;</testwaittime>				
		StbM_TimeStampType.seconds = 0x00000E10				
		StbM_TimeStampType.secondsHi = 0x0000				

		•
		StbM_UserDataType.User Data Byte 0 = 0xAA
		StbM_UserDataType.User Data Byte 1 = 0xBB
		StbM_UserDataType.User Data Byte 2 = 0xCC
Step 6	[LT <flexray>]</flexray>	[LT]
	Receives the SYNC message without CRC validation.	Receives frame format as mentioned below:
		Byte 0: Type = 0x10
		Byte 1: UserByte 2
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: FCNT= FlexRay Cycle Counter 0 (Bit 7 to Bit 2)
		SGW (Bit 1)
		SyncToSubDomain = 1
		reserved (Bit 0), default: 0
		Byte 4: User Byte 0
		Byte 5: User Byte 1
		Byte 6-11: SyncTimeSec = 48 Bit time stamp in seconds
		Byte 12-15: SyncTimeNSec = 32 Bit time stamp in nanoseconds
Step 7	[LT <flexray>]</flexray>	[LT]
	Receives Global time base.	Get the time stamp values as below:
	Store the time as base for next periodic message processing	StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 8	[LT <flexray>]</flexray>	[LT]
	Receives the next periodic SYNC message without CRC validation	Time stamp values shall be as

		mentioned below:
		StbM_TimeStampType.nanoseconds = tB.Nano
		StbM_TimeStampType.seconds = tB.Sec
		StbM_TimeStampType.secondsHi = tB.SecHi
		StbM_UserDataType.User Data Byte 0 = 0xAA
		StbM_UserDataType.User Data Byte 1 = 0xBB
		StbM_UserDataType.User Data Byte 2 = 0xCC
		The difference between two time base shall be
		tB – tA = FrTSynGlobalTimeTxPeriod (320ms).
Post- conditions	None	

# 5.4.2 [ATS\_GTS\_01287] Global Time Master: Setting of Offset Time base by Active Customer and sending Offset frames (CRC\_SUPPORTED) over FlexRay

	Global Time Master: Setting of Offset Time base by Active Customer and sending Offset frames (CRC_SUPPORTED) over FlexRay		
ID	ATS_GTS_01287	AUTOSAR Releases	4.2.1 4.2.2
Affected Modules	StbM, FrTSyn	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00134		
Trace to SWS Item	TimeSyncOverFlexRay: SWS_FrTSyn_00031		
Requirements / Reference to Test Environment	Use Case UC02.01		
Parameters	StbM: StbMSynchronizedTimeBaseIdentifier = 17  FrTSyn: FrTSynGlobalTimeTxCrcSecured = CRC_SUPPORTED		

	I		
Summary	Aim is to:		
	Verify that SthM accents the global time base	Verify that SthM accepts the global time been from Upper Tester using client corner	
	Verify that StbM accepts the global time base from Upper Tester using client-server Interface.		
	Verify that FrTSyn shall transmit the offset time base to time slave periodically via		
Needed	OFS message with CRC secured.  Not Applicable		
Adaptation to	Not Applicable		
other Releases			
	SUT shall be initialized.		
Main Test Exec	ution		
Test Steps		Pass Criteria	
Step 1	[CP]		
	Start RUN_ GlobalTimeProvider		
Step 2	[RUN <run_ globaltimeprovider="">]</run_>	[RUN <run_ globaltimeprovider="">]</run_>	
	Execute Rte_Call_ Client_ SetGlobalTime	Rte_Call returns RTE_E_OK	
	with below values:		
	TimeBaseId = 1		
	TimeBaseia = 1		
	StbM_TimeStampType.nanoseconds =		
	0x00000000		
	StbM_TimeStampType.seconds =		
	0x00000E10 (3600d)		
<b>.</b>	StbM_TimeStampType.secondsHi = 0x0000		
Step 3	[CP]		
	Wait 100ms		
Step 4	[RUN <run_ globaltimeprovider="">]</run_>	[RUN <run_ globaltimeprovider="">]</run_>	
	Execute Rte_Call_ Client_ SetOffset with	Rte_Call returns RTE_E_OK	
	below values:		
	timeBaseld = 17		
	StbM_TimeStampType.nanoseconds =		
	0x00000000		
	StbM_TimeStampType.seconds =		
	0x00000064		
	Oth M. Time Others of the second of the second		
Ston E	StbM_TimeStampType.secondsHi = 0x0000		
Step 5	[CP]		
	Wait 100ms		
Step 6	[CP]		
	L		
	Start RUN_TimeUser		

Step 7	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
	Time base = 17	Time received from Time user shall be as mentioned below:
		StbM_TimeStampType.nanoseconds = 0x000000000 + <testwaittime =<br="">100ms&gt; + <testwaittime 100ms="" ==""></testwaittime></testwaittime>
		StbM_TimeStampType.seconds = 0x00000E10 + 0x00000064
		StbM_TimeStampType.secondsHi = 0x0000
Step 8	[LT <flexray>]</flexray>	[LT]
	Receives the OFS message with CRC as per test co-ordination requirement for CRC.	Receives OFS message with format
		Byte 0: Type = 0x40
		Byte 1: CRC
		Byte 2: D = Time Domain 16 (Bit 7 to Bit 4)
		SC = <sequence 0x01="" counter="">(Bit 3 to Bit 0)</sequence>
		Byte 3: User Byte 0
		Byte 4: User Byte 1
		Byte 5: User Byte 2
		Byte 6-11: OfsTimeSec = <stbm_timestamptype.seconds 0x000000000064&gt;</stbm_timestamptype.seconds 
		Byte 12-15: OfsTimeNSec = <stbm_timestamptype.nanoseconds = 0x000000000&gt;</stbm_timestamptype.nanoseconds 
Post- conditions	None	

# 5.4.3 [ATS\_GTS\_01249] Time Slave: Reception of SYNC frames (CRC\_VALIDATED) over FlexRay, Synchronize Local Time Base and share the current time to active customers

	Time Slave: Reception of SYNC frames (CRC_VALIDATED) over FlexRay, Synchronize Local Time Base and share the current time to active customers		
ID	ATS_GTS_01249	AUTOSAR	4.2.1 4.2.2

	I	Releases	1
Affected	StbM, FrTSyn	State	reviewed
Modules			
	ATR: ATR_ATR_00134		
Requirement			
on Acceptance Test Document			
Trace to SWS	TimeSyncOverFlexRay: SWS_FrTSyn_00050		
Item	I i i i i i i i i i i i i i i i i i i i		
Requirements /	Use Case UC02.02		
Reference			
to Test Environment			
Configuration	StbM:		
Parameters	StbMSynchronizedTimeBaseIden	ntifier = 1	
	•		
	FrTSyn:	ALIDATED	
Summary	FrTSynRxCrcValidated = CRC_V Verify that FrTSyn shall receive S		ion frames after validating CPC
Summary	l verify that F11 Syll Shall receive S	griciiionizat	ion frames after validating CNC.
	Verify that StbM shall synchronize	e its local tim	ne base on reception of Time Base
	from FrTSyn.		·
	L		
	Verity that Upper Tester shall get format, user data using Client-Se		rrent time, current time in extended
Needed	Not Applicable	iver interiac	o.
Adaptation to	, tet, ipplicable		
other Releases			
Pre-conditions	SUT shall be initialized.		
Pre-conditions Main Test Exect			
Pre-conditions Main Test Execu Test Steps	ution		Pass Criteria
Pre-conditions Main Test Exect			Pass Criteria [SUT]
Pre-conditions Main Test Execu Test Steps	[LT <flexray>]</flexray>		[SUT]
Pre-conditions Main Test Execu Test Steps	ution		
Pre-conditions Main Test Execu Test Steps	[LT <flexray>]</flexray>		[SUT] Receives SYNC message with CRC value in below format
Pre-conditions Main Test Execu Test Steps	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1</flexray>		[SUT] Receives SYNC message with CRC
Pre-conditions Main Test Execu Test Steps	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1  StbM_UserDataType.User Data E</flexray>	3yte 0 =	[SUT]  Receives SYNC message with CRC value in below format  Byte 0: Type = 0x20
Pre-conditions Main Test Execu Test Steps	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1</flexray>	3yte 0 =	[SUT] Receives SYNC message with CRC value in below format
Pre-conditions Main Test Execu Test Steps	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1  StbM_UserDataType.User Data E</flexray>	3yte 0 =	[SUT]  Receives SYNC message with CRC value in below format  Byte 0: Type = 0x20
Pre-conditions Main Test Execu Test Steps Step 1	[LT <flexray>] Transmit SYNC message with timeBaseId = 1 StbM_UserDataType.User Data E</flexray>	3yte 0 =	[SUT]  Receives SYNC message with CRC value in below format  Byte 0: Type = 0x20  Byte 1: CRC
Pre-conditions Main Test Execu Test Steps Step 1	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1  StbM_UserDataType.User Data E 0xAA  StbM_UserDataType.User Data E 0xBB</flexray>	3yte 0 =	[SUT]  Receives SYNC message with CRC value in below format  Byte 0: Type = 0x20  Byte 1: CRC  Byte 2: D = <time domain="1"> (Bit 7 to Bit 4)</time>
Pre-conditions Main Test Execu Test Steps Step 1	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1  StbM_UserDataType.User Data E 0xAA  StbM_UserDataType.User Data E</flexray>	3yte 0 =	[SUT]  Receives SYNC message with CRC value in below format  Byte 0: Type = 0x20  Byte 1: CRC  Byte 2: D = <time domain="1"> (Bit 7 to Bit 4)  SC = <sequence counter="0"> (Bit 3</sequence></time>
Pre-conditions Main Test Execu Test Steps Step 1	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1  StbM_UserDataType.User Data E 0xAA  StbM_UserDataType.User Data E 0xBB  Byte CRC = Calculated CRC</flexray>	3yte 0 =	[SUT]  Receives SYNC message with CRC value in below format  Byte 0: Type = 0x20  Byte 1: CRC  Byte 2: D = <time domain="1"> (Bit 7 to Bit 4)</time>
Pre-conditions Main Test Execu Test Steps Step 1	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1  StbM_UserDataType.User Data E 0xAA  StbM_UserDataType.User Data E 0xBB</flexray>	3yte 0 =	[SUT]  Receives SYNC message with CRC value in below format  Byte 0: Type = 0x20  Byte 1: CRC  Byte 2: D = <time domain="1"> (Bit 7 to Bit 4)  SC = <sequence counter="0"> (Bit 3</sequence></time>
Pre-conditions Main Test Execu Test Steps Step 1	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1  StbM_UserDataType.User Data E 0xAA  StbM_UserDataType.User Data E 0xBB  Byte CRC = Calculated CRC  And time value as T0  Where T0 = TSYNC + (Macrotick</flexray>	Byte 0 = Byte 1 = sPerCycle *	[SUT]  Receives SYNC message with CRC value in below format  Byte 0: Type = 0x20  Byte 1: CRC  Byte 2: D = <time domain="1"> (Bit 7 to Bit 4)  SC = <sequence counter="0"> (Bit 3 to Bit 0)</sequence></time>
Pre-conditions Main Test Execu Test Steps Step 1	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1  StbM_UserDataType.User Data E 0xAA  StbM_UserDataType.User Data E 0xBB  Byte CRC = Calculated CRC  And time value as T0  Where T0 = TSYNC + (Macrotick (64 - currentCycle) - currentMacrotick (64 - currentCycle) - currentMacrotick</flexray>	Byte 0 = Byte 1 = sPerCycle *	[SUT]  Receives SYNC message with CRC value in below format  Byte 0: Type = 0x20  Byte 1: CRC  Byte 2: D = <time domain="1"> (Bit 7 to Bit 4)  SC = <sequence counter="0"> (Bit 3 to Bit 0)  Byte 3: FCNT= <flexray counter="0" cycle=""> (Bit 7 to Bit 2)</flexray></sequence></time>
Pre-conditions Main Test Execu Test Steps Step 1	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1  StbM_UserDataType.User Data E 0xAA  StbM_UserDataType.User Data E 0xBB  Byte CRC = Calculated CRC  And time value as T0  Where T0 = TSYNC + (Macrotick</flexray>	Byte 0 = Byte 1 = sPerCycle *	[SUT]  Receives SYNC message with CRC value in below format  Byte 0: Type = 0x20  Byte 1: CRC  Byte 2: D = <time domain="1"> (Bit 7 to Bit 4)  SC = <sequence counter="0"> (Bit 3 to Bit 0)  Byte 3: FCNT= <flexray cycle<="" th=""></flexray></sequence></time>
Pre-conditions Main Test Execu Test Steps Step 1	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1  StbM_UserDataType.User Data E 0xAA  StbM_UserDataType.User Data E 0xBB  Byte CRC = Calculated CRC  And time value as T0  Where T0 = TSYNC + (Macrotick (64 - currentCycle) - currentMacrotickDuration</flexray>	Byte 0 = Byte 1 = sPerCycle * oticks) *	[SUT]  Receives SYNC message with CRC value in below format  Byte 0: Type = 0x20  Byte 1: CRC  Byte 2: D = <time domain="1"> (Bit 7 to Bit 4)  SC = <sequence counter="0"> (Bit 3 to Bit 0)  Byte 3: FCNT= <flexray counter="0" cycle=""> (Bit 7 to Bit 2)  SGW (Bit 1)</flexray></sequence></time>
Pre-conditions Main Test Execu Test Steps Step 1	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1  StbM_UserDataType.User Data E 0xAA  StbM_UserDataType.User Data E 0xBB  Byte CRC = Calculated CRC  And time value as T0  Where T0 = TSYNC + (Macrotick (64 - currentCycle) - currentMacrotick (64 - currentCycle) - currentMacrotick</flexray>	Byte 0 = Byte 1 = sPerCycle * oticks) *	[SUT]  Receives SYNC message with CRC value in below format  Byte 0: Type = 0x20  Byte 1: CRC  Byte 2: D = <time domain="1"> (Bit 7 to Bit 4)  SC = <sequence counter="0"> (Bit 3 to Bit 0)  Byte 3: FCNT= <flexray counter="0" cycle=""> (Bit 7 to Bit 2)</flexray></sequence></time>
Pre-conditions Main Test Execu Test Steps Step 1	[LT <flexray>]  Transmit SYNC message with timeBaseId = 1  StbM_UserDataType.User Data E 0xAA  StbM_UserDataType.User Data E 0xBB  Byte CRC = Calculated CRC  And time value as T0  Where T0 = TSYNC + (Macrotick (64 - currentCycle) - currentMacrotickDuration</flexray>	Byte 0 =  Byte 1 =  sPerCycle * oticks) *	[SUT]  Receives SYNC message with CRC value in below format  Byte 0: Type = 0x20  Byte 1: CRC  Byte 2: D = <time domain="1"> (Bit 7 to Bit 4)  SC = <sequence counter="0"> (Bit 3 to Bit 0)  Byte 3: FCNT= <flexray counter="0" cycle=""> (Bit 7 to Bit 2)  SGW (Bit 1)</flexray></sequence></time>

	0x00000E10	Byte 4: User Byte 0
	StbM_TimeStampType.secondsHi = 0x0000	Byte 5: User Byte 1
	StbM_TimeStampType.nanoseconds = 0x00000000	Byte 6-11: SyncTimeSec = T0.secondsHi and T0.seconds
		Byte 12-15: SyncTimeNSec = T0.nanoseconds
Step 2		[SUT]
		StbM updates its time base with values
		StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
		Time updated in StbM, tA = tA + ToleranceTime_FrTSyn.
		ToleranceTime_FrTSyn = Max one main function of FrTSyn as reception is asynchronous.
Step 3	[CP]	
	Wait 200ms	
Step 4	[CP]	
	Start RUN_TimeUser	
Step 5	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime =<br="">200ms&gt;</testwaittime>
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Post- conditions	None	

# 5.4.4 [ATS\_GTS\_01250] Time Slave: Reception of SYNC frames (CRC\_IGNORED) over FlexRay, Synchronize Local Time Base and share the current time to active customers.

Test Objective	- 0 5 4 60006	(0.50	10110DED) EL D	
	Time Slave: Reception of SYNC frames (CRC_IGNORED) over FlexRay, Synchronize Local Time Base and share the current time to active customers.			
ID	ATS_GTS_01250		4.2.1 4.2.2	
Affected Modules	StbM, FrTSyn	State	reviewed	
Trace to Requirement on Acceptance Test Document				
Trace to SWS Item	TimeSyncOverFlexRay: SWS_FrT	Syn_00046	6	
Requirements / Reference to Test Environment	Use Case UC02.02			
Configuration Parameters	StbM: StbMSynchronizedTimeBaseIdentifier = 1  FrTSyn: FrTSynRxCrcValidated = CRC_IGNORED			
Summary	Aim is to Verify that FrTSyn shall call StbM to update time base on reception of SYNC message even with invalid CRC when FrTSynRxCrcValidated = CRC_IGNORED.			
Needed Adaptation to other Releases	Not Applicable			
Pre-conditions	SUT shall be initialized.			
	Main Test Execution			
Test Steps	-		Pass Criteria	
Step 1	[LT <flexray>]</flexray>		ICLITI	
			[SUT]	
	Transmit SYNC message with		Receives SYNC message ignoring CRC value (if there) in below format	
	timeBaseId = 1	0	Receives SYNC message ignoring	
		/te 0 =	Receives SYNC message ignoring CRC value (if there) in below format	
	timeBaseId = 1 StbM_UserDataType.User Data By		Receives SYNC message ignoring CRC value (if there) in below format  Byte 0: Type = 0x10	
	timeBaseId = 1 StbM_UserDataType.User Data By 0xAA StbM_UserDataType.User Data By		Receives SYNC message ignoring CRC value (if there) in below format  Byte 0: Type = 0x10  Byte 1: User Byte 2  Byte 2: D = <time domain="1"> (Bit 7)</time>	
	timeBaseId = 1  StbM_UserDataType.User Data By 0xAA  StbM_UserDataType.User Data By 0xBB	/te 1 = PerCycle *	Receives SYNC message ignoring CRC value (if there) in below format  Byte 0: Type = 0x10  Byte 1: User Byte 2  Byte 2: D = <time domain="1"> (Bit 7 to Bit 4)  SC = <sequence counter="0"> (Bit 3</sequence></time>	
	timeBaseId = 1  StbM_UserDataType.User Data By 0xAA  StbM_UserDataType.User Data By 0xBB  And time value as T0  Where T0 = TSYNC + (Macroticks) (64 - currentCycle) - currentMacrotic	/te 1 = PerCycle * icks) *	Receives SYNC message ignoring CRC value (if there) in below format  Byte 0: Type = 0x10  Byte 1: User Byte 2  Byte 2: D = <time domain="1"> (Bit 7 to Bit 4)  SC = <sequence counter="0"> (Bit 3 to Bit 0)  Byte 3: FCNT= <flexray cycle<="" th=""></flexray></sequence></time>	

		1
	0x00000E10	SyncToSubDomain = 1
	StbM_TimeStampType.secondsHi = 0x0000	reserved (Bit 0)
	StbM_TimeStampType.nanoseconds = 0x00000000	Byte 4: User Byte 0
		Byte 5: User Byte 1
		Byte 6-11: SyncTimeSec = T0.secondsHi and T0.seconds
		Byte 12-15: SyncTimeNSec = T0.nanoseconds
Step 2		[SUT]
		StbM updates its time base with values
		StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
		Time updated in StbM, tA = tA + ToleranceTime_FrTSyn.
		ToleranceTime_FrTSyn = Max one main function of FrTSyn as reception is asynchronous.
Step 3	[CP]	
	Wait 200ms	
Step 4	[CP]	
	Start RUN_TimeUser	
Step 5	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime =<br="">200ms&gt;</testwaittime>
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Post-	None	

conditions	

# 5.4.5 [ATS\_GTS\_01256] Time Slave: Handling of SYNC message Sequence Mismatch failures

Test	Time Slave: Handling of SYNC message Sequence Mismatch failures				
Objective					
ID	ATS_GTS_01256	AUTOSA	4.2.1 4.2.2		
		R	'		
		Releases			
Affected Modules	StbM, FrTSyn	State	reviewed		
	ATR: ATR_ATR_00134				
Requirement	/				
on <sup>'</sup>					
Acceptance					
Test					
Document	<del></del>		20042		
Trace to SWS Item	TimeSyncOverFlexRay: S	WS_FriSy	/n_00048		
	Use Case UC02.02				
s / Reference					
to Test					
Environment					
Configuratio	StbM:				
n	StbMSynchronizedTimeBa	aseldentifie	er = 1		
Parameters	FrTSyn:				
	FrTSynRxCrcValidated = CRC_IGNORED				
Summary	Aim is to verify that if Sequence Counter Jump Width between two SYNC messages is				
,	greater than FrTSynGlobalTimeSequenceCounterJumpWidth, then messages will be				
	ignored.				
	Not Applicable				
Adaptation					
to other Releases					
Pre-	SUT shall be initialized.				
conditions	oo i orian bo iiii.anzoai				
Main Test Exe	ecution				
Test Steps			Pass Criteria		
Step 1	[LT <flexray>]</flexray>		[SUT]		
	Transmit SYNC message	with	Receives SYNC message ignoring CRC value in		
	TimeBaseId = 1		below format		
	Timedaseid = T	= 1 Byte 0: Type = 0x10			
	StbM_UserDataType.User	· Data	byte of Type - ox to		
	Byte $0 = 0xAA$	Data	Byte 1: User Byte 2		
	StbM_UserDataType.User Byte 1 = 0xBB	· Data	Byte 2: D = <time domain="1"> (Bit 7 to Bit 4)</time>		
			SC = <sequence counter="0"> (Bit 3 to Bit 0)</sequence>		
	l .				

	And time value as T0a	Byte 3: FCNT= <flexray counter="0" cycle=""> (Bit 7 to Bit 2)</flexray>
	Where T0a = TSYNC + (MacroticksPerCycle * (64 - currentCycle) - currentMacroticks) *	SGW (Bit 1)
	MacrotickDuration	SyncToSubDomain = 1
	And TSYNC value as given below:	reserved (Bit 0)
	StbM_TimeStampType.seconds = 0x00000E10	Byte 4: User Byte 0
	StbM_TimeStampType.secondsHi =	Byte 5: User Byte 1
	0x0000 StbM_TimeStampType.nanosecond	Byte 6-11: SyncTimeSec = T0a.secondsHi and T0a.seconds
	s = 0x00000000	Byte 12-15: SyncTimeNSec = T0a.nanoseconds
Step 2		[SUT]
		FrTSyn shall update the time base of StbM
		StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
		Time updated in StbM, tA = tA + ToleranceTime_FrTSyn.
		ToleranceTime_FrTSyn = Max one main function of FrTSyn as reception is asynchronous.
Step 3	[CP]	
	Wait 200ms	
Step 4	[CP]	
Otan F	Start RUN_TimeUser	FRUM RUN Timelleen 1
Step 5	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 200ms="" =="">.</testwaittime>
		StbM_TimeStampType.seconds = tA .Sec

		StbM_TimeStampType.secondsHi = tA .SecHi
		StbM_TimeBaseStatusType.TIMEOUT is set to 0.
Step 6	[LT <flexray>]</flexray>	[SUT]
	Transmit next periodic SYNC message with timeBaseId = 1	Time base is not updated as per incoming time base as sequence counter value is greater than FrTSynGlobalTimeSequenceCounterJumpWidth
	StbM_UserDataType.User Data Byte 0 = 0xEE	
	StbM_UserDataType.User Data Byte 1 = 0xFF	
	And time value as T0b	
	Where T0b = TSYNC + (MacroticksPerCycle * (64 - currentCycle) - currentMacroticks) * MacrotickDuration	
	And TSYNC value as given below:	
	StbM_TimeStampType.seconds = 0x00000E10	
	StbM_TimeStampType.secondsHi = 0x0000	
	StbM_TimeStampType.nanosecond s = 0x00000000	
	Sequence counter = 3	
Step 7	[CP]	
	Wait 200ms	
	[CP]	
	Start RUN_TimeUser	
Step 9	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_	Rte_Call returns RTE_E_OK.
	Client_GetCurrentTime	INTE_Call returns NTE_E_ON.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 200ms="" ==""> + <testwaittime =<br="">200ms&gt;.</testwaittime></testwaittime>
		StbM_TimeStampType.seconds = tA .Sec + FrTSynGlobalTimeTxPeriod (320ms)

		StbM_TimeStampType.secondsHi = tA .SecHi
		StbM_TimeBaseStatusType.TIMEOUT is set to 1.
Post-	None	
conditions		

# 5.4.6 [ATS\_GTS\_01277] Time Slave: Reception of Offset frames (CRC\_NOT\_VALIDATED) over FlexRay, Synchronize Local Time Base and share the current time to active customers.

Test Objective	Time Slave: Reception of Offset frames (CRC_NOT_VALIDATED) over FlexRay,		
-	Synchronize Local Time Base and share the current time to active customers.		
ID	ATS_GTS_01277	AUTOSAR Releases	4.2.1 4.2.2
Affected Modules	StbM, FrTSyn	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00134		
Trace to SWS Item	TimeSyncOverFlexRay: SWS_Fr TimeSyncOverFlexRay: SWS_Fr		
Requirements / Reference to Test Environment	Use Case UC02.02		
Configuration Parameters  Summary	StbM: StbMSynchronizedTimeBaseIdentifier = 16  FrTSyn: FrTSynRxCrcValidated = CRC_NOT_VALIDATED CanTSynGlobalTimeOfsDataIDListIndex = {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15} CanTSynGlobalTimeOfsDataIDListValue in ASCII = {"A", "U", "T", "O", "S", "A", "R", "A", "T", "S", "G", "T", "S", "O", "F", "S"} CanTSynGlobalTimeOfnsDataIDListIndex = {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15} CanTSynGlobalTimeOfnsDataIDListValue in ASCII = {"A", "U", "T", "O", "S", "A", "R", "A", "T", "S", "G", "T", "O", "F", "N", "S"} Aim is to:  Verify that FrTSyn shall Recieve the offset time base periodically via offset message.		
	Verify that StbM shall synchronize its local offset time base on receiption of Time Base from FrTSyn.		
Needed Adaptation to other Releases	Not Applicable		
Pre-conditions	SUT shall be initialized.		
Main Test Execu	ution		
Test Steps			Pass Criteria
Step 1	[LT <flexray>]</flexray>		[SUT]

	Transmit SYNC message with:	Get the time stamp values as below:
	TimeBaseId = 1	StbM_TimeStampType.nanoseconds = tA.Nano
	And time value as T0	StbM_TimeStampType.seconds = tA
	Where T0 = TSYNC + (MacroticksPerCycle * (64 - currentCycle) - currentMacroticks) * MacrotickDuration	.Sec StbM_TimeStampType.secondsHi = tA .SecHi
	And TSYNC value as given below:	Time updated in StbM, tA = tA +
	StbM_TimeStampType.seconds = 0x00000E10	ToleranceTime_FrTSyn.
	StbM_TimeStampType.secondsHi = 0x0000	ToleranceTime_FrTSyn = Max one main function of FrTSyn as reception is asynchronous.
	StbM_TimeStampType.nanoseconds = 0x00000000	
Step 2	[CP]	
	Wait 200ms	
Step 3	[LT <flexray>]</flexray>	[SUT]
	Transmit OFS message with:	Get the offset time stamp values as below:
	TimeBaseId = 16	SthM TimeStampType papersonds
	StbM_TimeStampType.nanoseconds = 0x00000000	StbM_TimeStampType.nanoseconds = tA.Nano
	StbM_TimeStampType.seconds = 0x0000000A (10d)	StbM_TimeStampType.seconds = tA .Sec
	StbM_TimeStampType.secondsHi = 0x0000	StbM_TimeStampType.secondsHi = tA .SecHi
Step 4	[CP]	
Ctop 1	   1	
	Wait 200ms	
Step 5	[CP]	
	Start RUN_TimeUser	
Step 6	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime =<br="">200ms&gt; + <testwaittime 200ms="" ==""></testwaittime></testwaittime>
		StbM_TimeStampType.seconds = tA .Sec + <offset time="0x0000000A"></offset>

		StbM_TimeStampType.secondsHi = tA .SecHi
Post- conditions	None	

## 5.4.7 [ATS\_GTS\_01280] Time Slave: Handling of Offset message Sequence Mismatch failures

Test	Time Slave: Handling of Offset message Sequence Mismatch failures			
Objective				
ID		AUTOSA R Releases	4.2.1 4.2.2	
Affected Modules	StbM, FrTSyn	State	reviewed	
Requirement on Acceptance Test Document	ATR: ATR_ATR_00134			
Trace to SWS Item	TimeSyncOverFlexRay: SV	VS_FrTSyn	_00048	
s / Reference to Test Environment	Use Case UC02.02			
	StbM: StbMSynchronizedTimeBaseIdentifier = 16  FrTSyn: FrTSynRxCrcValidated = CRC_IGNORED			
Summary	Aim is to verify that if Sequence Counter Jump Width between two OFS messages is greater than FrTSynGlobalTimeSequenceCounterJumpWidth, then messages will be ignored.			
Needed Adaptation to other Releases	Not Applicable			
conditions	SUT shall be initialized. StbM shall be initialized with base time: StbM_TimeStampType.nanoseconds = 0x00000000 StbM_TimeStampType.seconds = 0x00000000 StbM_TimeStampType.secondsHi = 0x0000			
	Main Test Execution			
Test Steps			Pass Criteria	
	[LT <flexray>]  Transmit OFS message wit  TimeBaseId = 16  StbM_TimeStampType.nan</flexray>		[SUT]  Receives OFS message with format  Byte 0: Type = 0x30	

	= 0x00000000	Byte 1: User Byte 3
	StbM_TimeStampType.seconds = 0x00000014	Byte 2: D = <time domain="16"> (Bit 7 to Bit 4)</time>
	StbM_TimeStampType.secondsHi =	SC = <sequence counter="0"> (Bit 3 to Bit 0)</sequence>
	0x0000	Byte 3: User Byte 0
		Byte 4: User Byte 1
		Byte 5: User Byte 2
		Byte 6-11: OfsTimeSec = 0x000000000014
		Byte 12-15: OfsTimeNSec = 0x00000000
Step 2		[SUT]
		FrTSyn shall update the offset time base of StbM
		StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 3	[CP]	
	Wait 200ms	
Step 4	[CP]	
	Start RUN_TimeUser	
Step 5	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client GetCurrentTime.	Rte_Call returns RTE_E_OK.
	Olione_GetGallentTille.	Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 200ms="" =="">.</testwaittime>
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
		StbM_TimeBaseStatusType.TIMEOUT is set to 0.
Step 6	[LT]	[SUT]
	Transmit next periodic OFS message with	Time base is not updated as per new incoming time base as sequence counter value is greater than
	TimeBaseId = 16	FrTSynGlobalTimeSequenceCounterJumpWidth
		<b> </b> .

	0x00000050	
	StbM_TimeStampType.secondsHi = 0x0000	
	Sequence number = 3	
Step 7	[CP]	
	Wait 200ms.	
Step 8	[CP]	
	Start RUN_TimeUser	
Step 9	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client GetCurrentTime	Rte_Call returns RTE_E_OK.
	Cheff Cotton Chit in the	Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime 200ms="" ==""> + <testwaittime 200ms="" =="">.</testwaittime></testwaittime>
		StbM_TimeStampType.seconds = tA .Sec + FrTSynGlobalTimeTxPeriod (320ms) + FrTSynGlobalTimeTxPeriod (320ms)
		StbM_TimeStampType.secondsHi = tA .SecHi
		StbM_TimeBaseStatusType.TIMEOUT is set to 1.
Post- conditions	None	

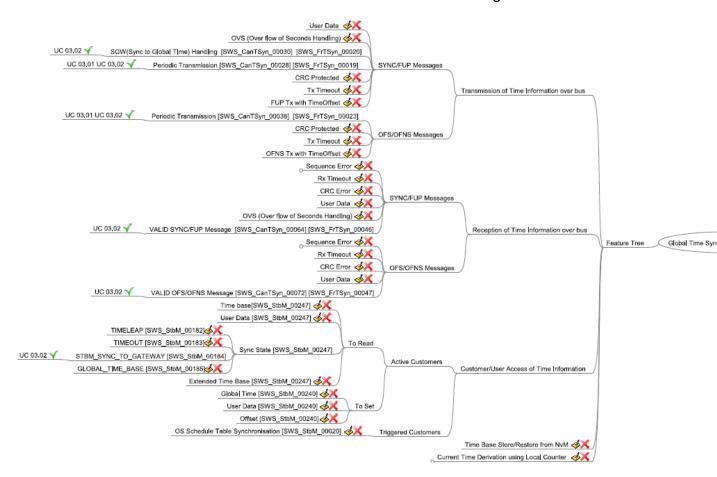
# 6 RS\_BRF\_01660 - Global Time Synchronization over Multiple Bus

#### 6.1 General Test Objective and Approach

This Test Specification intends to cover the Global Time Synchronization feature of StbM CanTSync, and FrTSync as described in the AUTOSAR Feature [RS\_BRF\_01660].

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:



Features not tested in this test suite are either tested in 'Global Time Synchronization over CAN' or 'Global Time Synchronization over FlexRay'.

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

#### 6.1.1 Test System

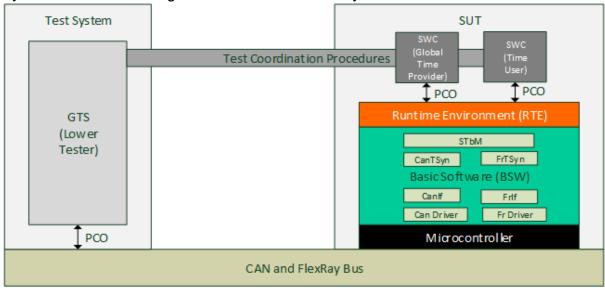
#### **6.1.1.1 Overview on Architecture**

In order to cover the required features / sub-features coverage, the environment has been separated in several use cases.

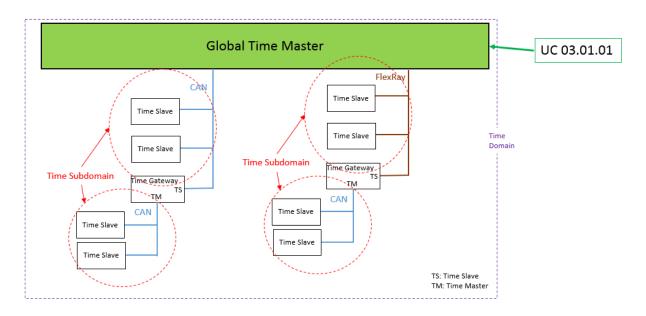
Test Cases are derived based on below use cases

## 6.1.1.1.1 UC 03.01.01 Global Time Master (Single Time Domain) over both CAN and FlexRay

SUT acts as Global time Master over Single time domain and Transmits Time Synchronization messages over CAN and FlexRay bus.

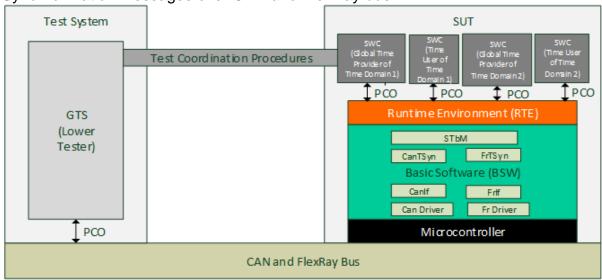


As shown in below figure, Functionalities of Global Time master for single time domain are tested over CAN and FlexRay in the use case 03.01.01

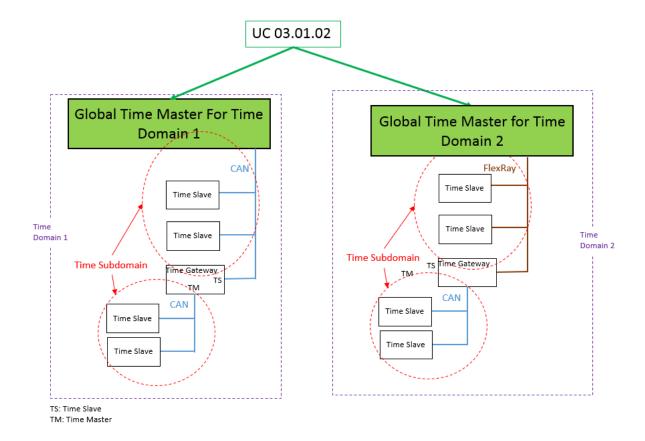


## 6.1.1.1.2 UC 03.01.02 Global Time Master (Multiple Time Domain) over both CAN and FlexRay

SUT acts as Global time Master over multiple time domain and Transmits Time Synchronization messages over CAN and FlexRay bus.

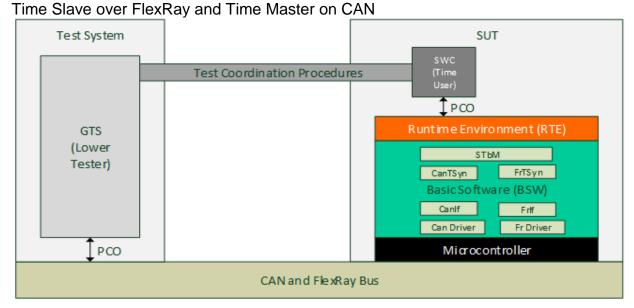


As shown in below figure, Functionalities of Global Time master for multiple time domain are tested over CAN and FlexRay in the use case 03.01.02

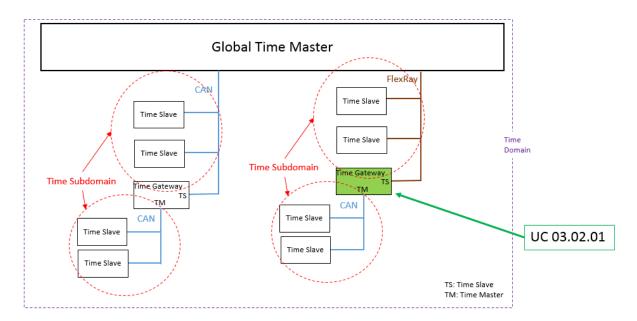


### 6.1.1.1.3 UC 03.02.01 Time Gateway: Time Slave over FlexRay and time master over CAN

SUT acts as Time Gateway. It receives time base, offset time base and Synchronizes Local time to Global Time base and transmits the received time base to its slaves.

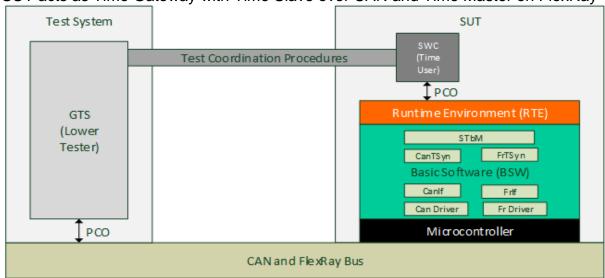


As shown in below figure, Functionalities of Time Gateway - Time Slave over FlexRay and Time Master over CAN in the use case 03.02.01

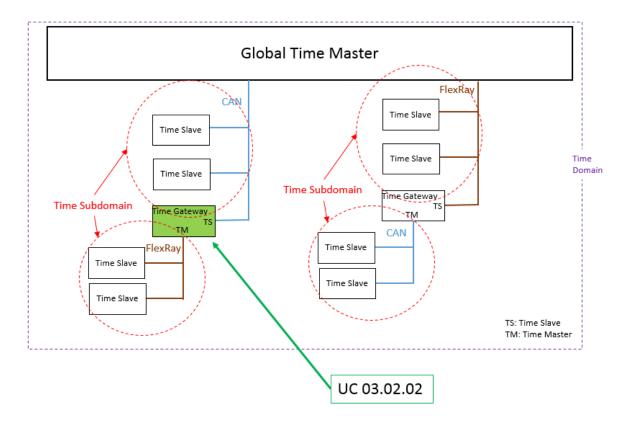


## 6.1.1.1.4 UC 03.02.02 Time Gateway: Time Slave over CAN and time master over FlexRay

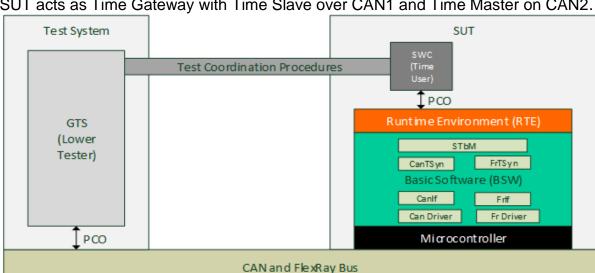
SUT acts as Time Gateway with Time Slave over CAN and Time Master on FlexRay



As shown in below figure, Functionalities of Time Gateway - Time Slave over CAN and Time Master over FlexRay in the use case 03.02.02

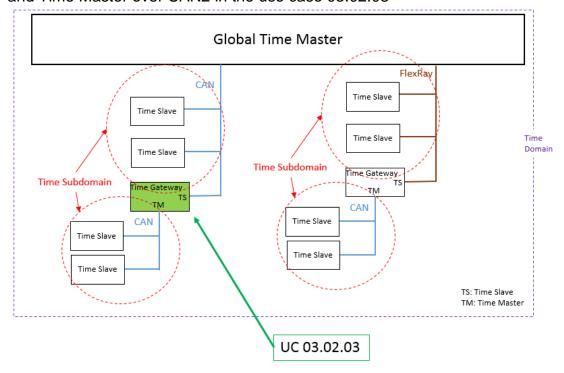


# 6.1.1.1.5 UC 03.02.03 Time Gateway: Time Slave over CAN (Network 1) and time master over CAN (Network 2)



SUT acts as Time Gateway with Time Slave over CAN1 and Time Master on CAN2.

As shown in below figure, Functionalities of Time Gateway - Time Slave over CAN1 and Time Master over CAN2 in the use case 03.02.03



#### 6.1.1.2 Specific Requirements

Not Applicable.

#### **6.1.1.3 Test Coordination Requirements**

UC 03.01, UC 03.02: Time Master over CAN and/or FlexRay

- Test System (LT <CAN>) shall read the CanTSyn CAN Frames and decode the same as per Frame Format provided in AUTOSAR\_SWS\_TimeSyncOverCAN.
- Test System (LT <FlexRay>) shall read the FrTSyn FlexRay Frames and decode the same as per Frame Format provided in AUTOSAR SWS TimeSyncOverFlexRay.

#### UC 03.01, UC 03.02: Time Slave over CAN and/or FlexRay

- Test System (LT <CAN>) shall encode the CanTSyn CAN Frames as per Frame Format provided in AUTOSAR SWS TimeSyncOverCAN and transmit over bus.
- Test System (LT <FlexRay>) shall encode the FrTSyn FlexRay Frames as per Frame Format provided in AUTOSAR\_SWS\_TimeSyncOver FlexRay and transmit over bus.

#### Requirements for CRC Calculation

Test System (LT <CAN>) shall use the Crc\_CalculateCRC8H2F() (Refer AUTOSAR Specification
of CRC Routines AUTOSAR\_SWS\_CRCLibrary.pdf) to calculate the CRC of the Frame. Below
are the parameters used for CRC calculation:

The CRC start value shall be 0xFF.

The CRC final XOR-value shall be 0xFF.

The CRC polynomial shall be 0x2F.

The DataIDList shall be same as provided in CanTSyn Static Configuration.

#### 6.1.2 Test Case Design

- Global Time Master over Multiple Bus(Single Time Domain)
   The Time domain for both CAN and FlexRay will be same. And Synchronization frames shall be sent over bus as per respective TSyn module configuration
- Global Time Master over Multiple Bus(Single Time Domain)
   The Time domain for both CAN and FlexRay will be different. And Synchronization frames shall be sent over bus with respective time domain information and respective TSyn module configuration

Below figures provides detailed info about how the time base information is validated by the tester.

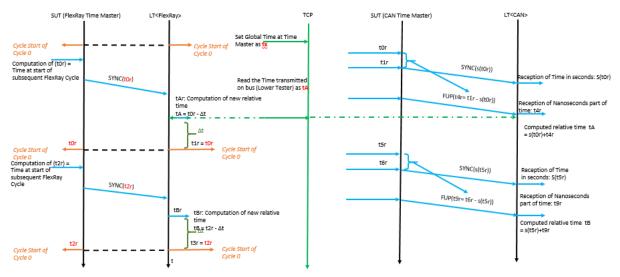


Fig: Global Time Master over Multiple Bus (Single/Multiple Time Domain) Test Design

Time Gateway - Time Slave on FlexRay and Time Master on CAN

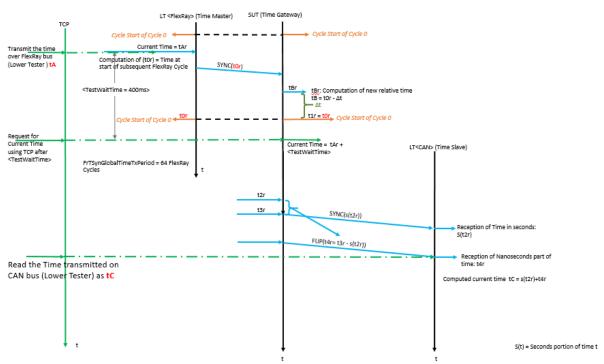


Fig: Time Gateway - Time Slave on FlexRay and Time Master on CAN Test Design

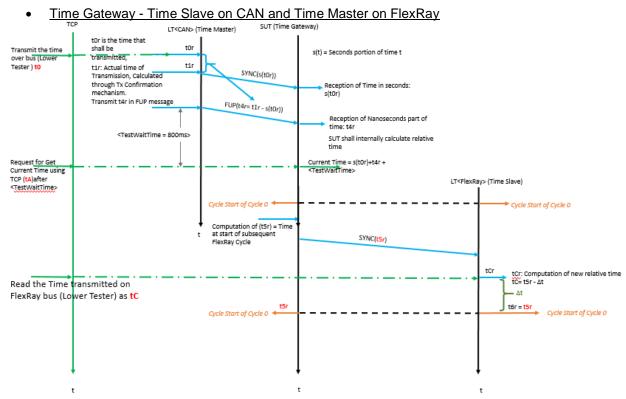


Fig: Time Gateway - Time Slave on CAN and Time Master on FlexRay Test Design

Time Gateway - Time Slave on CAN (Network 1) and Time Master on CAN (Network 2)

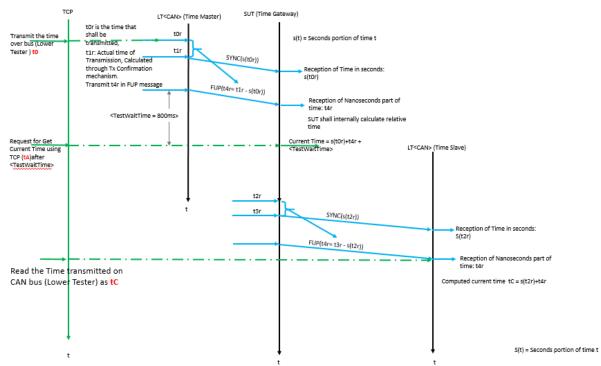


Fig: Time Gateway - Time Slave on CAN Network 1 and Time Master on CAN network 2 Test Design

### 6.2 Configuration requirements

The configuration can be divided into two separate parts. The *test configuration* describes variables used to parameterize the test case. The *static configuration* describes the necessary settings of the DUT in order to allow a test case to perform.

#### 6.2.1 Test Configuration

Communication data base for CanTSyn is depicted below

test configuration parameters				
I-Pdu CAN ID Tx ECU Rx ECU				
AT_101_lpdu	101	SUT	Test Bench	
AT_102_lpdu	102	Test Bench	SUT	

Communication data base for FrTSyn is depicted below

test configuration parameters			
Parameter	Value		
Frlf Tx Pdu	AT_201_lpdu201		
FrlfTxPdu. Frlflmmediate	FALSE		
FrlfFrameTriggering.FrlfBaseCycle	0		
FrlfFrameTriggering.FrlfCycleRepetition	64		
FrlfFrameTriggering.FrlfSlotId	2		
Frlf Rx Pdu	AT_202_lpdu202		

### 6.2.2 Static Configuration

### **6.2.2.1 Static Configuration Groups**

SCG_ATS_GlobalTimeSync_Single and Multiple time Domain	
System Description Parameters	
StbM	
SystemTemplate::GlobalTime::GlobalTimeMaster.isSystemWideGl	TRUE
obalTimeMaster	
CanTSyn	
SystemTemplate::GlobalTime::GlobalTimeMaster.syncPeriod	2000ms
SystemTemplate::GlobalTime::CAN::GlobalTimeCanMaster.syncC	80ms
onfirmationTimeout	
SystemTemplate::GlobalTime::GlobalTimeDomain.globalTimePdu	Ref. To PDU
FrTSyn	
SystemTemplate::GlobalTime::GlobalTimeDomain.domainId	1
SystemTemplate::GlobalTime::GlobalTimeMaster.syncPeriod	2000ms
SystemTemplate::GlobalTime::GlobalTimeDomain.subDomain	Ref. To PDU
ECU Configuration Parameters	
StbM	
StbM.StbMGeneral.StbMMainFunctionPeriod	5ms
StbM.StbMSynchronizedTimeBase.StbMLocalTimeRef	Ref. to
	OSCounter
CanTSyn	
CanTSyn.CanTSynGeneral.CanTSynMainFunctionPeriod	5ms
CanTSyn.CanTSynGlobalTimeDomain.	1
CanTSynGlobalTimeDomainId	
CanTSyn.CanTSynGlobalTimeDomain.	Ref. to StbM
CanTSynSynchronizedTimeBaseRef	time base
CanTSyn.CanTSynGlobalTimeDomain.	100ms
CanTSynGlobalTimeMaster.	
CanTSynGlobalTimeTxFollowUpOffset	
CanTSyn.CanTSynGlobalTimeDomain.	1
CanTSynGlobalTimeMaster. CanTSynGlobalTimeMasterPdu.	
O TO OLITE NA 4 O C	
CanTSynGlobalTimeMasterConfirmationHandleId	
FrTSyn	
FrTSyn FrTSynGeneral.FrTSynMainFunctionPeriod	5ms
FrTSyn FrTSyn.FrTSynGeneral.FrTSynMainFunctionPeriod FrTSyn.FrTSynGlobalTimeDomain.	Ref. to StbM
FrTSyn FrTSyn.FrTSynGeneral.FrTSynMainFunctionPeriod FrTSyn.FrTSynGlobalTimeDomain. FrTSynSynchronizedTimeBaseRef	Ref. to StbM time base
FrTSyn FrTSyn.FrTSynGeneral.FrTSynMainFunctionPeriod FrTSyn.FrTSynGlobalTimeDomain. FrTSynSynchronizedTimeBaseRef FrTSyn.CanTSynGlobalTimeDomain.FrTSynGlobalTimeMaster.	Ref. to StbM
FrTSyn FrTSyn.FrTSynGeneral.FrTSynMainFunctionPeriod FrTSyn.FrTSynGlobalTimeDomain. FrTSynSynchronizedTimeBaseRef FrTSyn.CanTSynGlobalTimeDomain.FrTSynGlobalTimeMaster. FrTSynGlobalTimeMasterPdu.	Ref. to StbM time base
FrTSyn FrTSyn.FrTSynGeneral.FrTSynMainFunctionPeriod FrTSyn.FrTSynGlobalTimeDomain. FrTSynSynchronizedTimeBaseRef FrTSyn.CanTSynGlobalTimeDomain.FrTSynGlobalTimeMaster.	Ref. to StbM time base

COO ATC ClabaltimaComa Tima Catavasu	
SCG_ATS_GlobalTimeSync_Time Gateway	
System Description Parameters	
StbM	
SystemTemplate::GlobalTime::GlobalTimeMaster.isSystemWideGl	FALSE
obalTimeMaster	
CanTSyn	
SystemTemplate::GlobalTime::GlobalTimeDomain.globalTimePdu	Ref. To PDU
FrTSyn	
SystemTemplate::GlobalTime::GlobalTimeDomain.subDomain	Ref. To PDU
ECU Configuration Parameters	
StbM	
StbM.StbMGeneral.StbMMainFunctionPeriod	5ms
StbM.StbMSynchronizedTimeBase.StbMLocalTimeRef	Ref. to
	OSCounter
CanTSyn	
CanTSyn.CanTSynGeneral.CanTSynMainFunctionPeriod	5ms
CanTSyn.CanTSynGlobalTimeDomain.	0
CanTSynGlobalTimeMaster.CanTSynGlobalTimeMasterPdu.	
CanTSynGlobalTimeMasterConfirmationHandleId	
•	
FrTSyn	
FrTSyn.FrTSynGeneral.FrTSynMainFunctionPeriod	5ms
FrTSyn.CanTSynGlobalTimeDomain.FrTSynGlobalTimeMaster.	0
FrTSynGlobalTimeMasterPdu.	
FrTSynGlobalTimeMasterConfirmationHandleId	
Test Cases	
UC 03.02	
00 00:02	

### 6.2.2.2 Required System Description

Refer Section 5.2.2.1

### 6.2.2.3 Required ECU Configuration

Refer Section 5.2.2.1

#### 6.2.2.4 Required Software Components

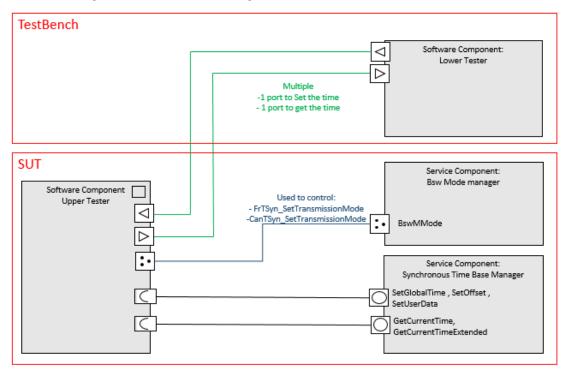


Fig: UC 03.01.01 - Global Time Master over Multiple Bus(Single Time Domain) SWC Overview

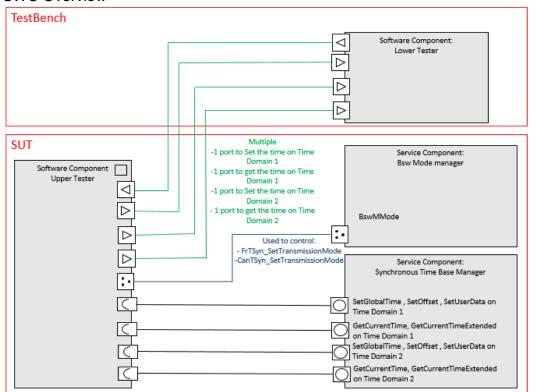


Fig: UC 03.01.02 - Global Time Master over Multiple Bus (Multiple Time Domain) SWC Overview

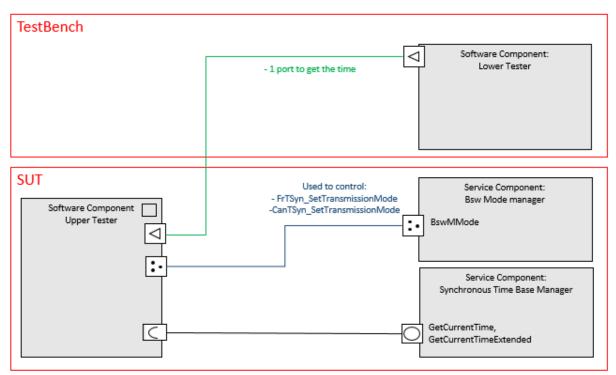


Fig: UC 03.02.01, UC 03.02.02 and UC 03.02.03 - Time Gateway SWC Overview

#### 6.2.2.4.1 SWC Client GlobalTime Provider

SWC Name	GlobalTime_Provider		
	Name Client_SetGlobalTime		
		_	
	Туре	RPortPrototype	
	Interface	GlobalTime_Master_Interface	
	Requirements		
	Name	Client_SetUserData	
PORTS	Туре	RPortPrototype	
	Interface	GlobalTime_Master_Interface	
	Requirements		
	Name	Client_SetOffset	
	Туре	RPortPrototype	
	Interface	GlobalTime_Master_Interface	
	Requirements		

Name		RUN_GlobalTimeProvider	
RUNNABLE ENTITIES	Requirements	Runnable shall be invoked by TCP	
	ServerCallPoint	Name	sscp_GlobalTimeProvider
		Туре	SynchronousServerCallPoint
		Access to	Client_SetGlobalTime (Write operation) Client_SetUserData (Write operation) Client_SetOffset (Write operation)
		Requirements	

### 6.2.2.4.2 SWC Client Time\_User

SWC Name	Time_User			
	A.1	0 0 .0	. <del></del>	
	Name	Client_GetCurrentTime		
	Туре	RPortPrototype		
	Interface	GlobalTime_Sla	ve_Interface	
	Requirements			
PORTS				
	Name	Client_GetCurre	entTimeExtended	
	Туре	RPortPrototype		
	Interface	GlobalTime_Slave_Interface		
	Requirements			
	Name	RUN_TimeUser		
	Requirements	Runnable shall be invoked by TCP		
		Name	sscp_TimeUser	
		Туре	SynchronousServerCallPoint	
RUNNABLE ENTITIES		Access to Client_GetCurrentTime (Read		
LITTILO	ServerCallPoint		operation) Client_GetCurrentTimeExtended	

		(Read operation)	
		Requirements	

#### 6 2 2 4 3 SWC Server StbM

	VC Server StbM		
SWC Name	StbM		
	Name	Server_SetGlobalTime	
	Туре	PPortPrototype	
	Interface	GlobalTime_Master_Interface	
	Requirements		
	Name	Server_SetOffset	
	Туре	PPortPrototype	
	Interface	GlobalTime_Master_Interface	
	Requirements		
	Name	Server_SetUserData	
PORTS	Туре	PPortPrototype	
	Interface	GlobalTime_Master_Interface	
	Requirements		
	Name	Server_GetCurrentTime	
	Туре	PPortPrototype	
	Interface	GlobalTime_Slave_Interface	
	Requirements		
	Name	Server_GetCurrentTimeExtended	
	Туре	PPortPrototype	
	Interface	GlobalTime_Slave_Interface	
Requirements			
	Name	StbM_SetGlobalTime	
	Requirements		

		Name	OIE_ SetGlobalTime
			_
		Type	OperationInvokedEvent
	Started by Event		Port:Server_SetGlobalTime
			Operation: Read
		Requirements	
	Marra	CthM CotOffort	
	Name	StbM_SetOffset	
	Requirements		
	- roquironionio		
DUNINIA DI E		Name	OIE_SetOffset
RUNNABLE ENTITIES			
ENTITIES	ServerCallPoint	Type	OperationInvokedEvent
			Port: Server_SetOffset
		D	Operation: Read
		Requirements	
	Name	SthM SatlicarD	)oto
	Name	StbM_SetUserData	
	Requirements		
	•		
		Name	OIE_ SetUserData
	Started by Event		0 1 15
		Туре	OperationInvokedEvent
			Port: Server_SetUserData Operation: Read
		Requirements	Operation. Nead
	Name	StbM_GetCurre	ntTime
	Requirements		
		Mana	OIE CotCurrontTime
		Name	OIE_ GetCurrentTime
	Started by Event	Туре	OperationInvokedEvent
		.,,,,,	Port: Server_GetCurrentTime
			Operation: Read
		Requirements	
	Name StbM_GetCurrentTimeExtended		ntTimeExtended
D			
	Requirements		

	Name	OIE_GetCurrentTimeExtended
Started by Event	Туре	OperationInvokedEvent
		Port:
		Server_GetCurrentTimeExtended
		Operation: Read
	Requirements	

### 6.3 Re-usable Test Steps

Not Applicable

#### 6.4 Test Cases

# 6.4.1 [ATS\_GTS\_01281] Global Time Master over Multiple Bus(Single Time Domain): Setting of Global Time base and user data and sending of SYNC frame over CAN and FlexRay

	-		
	Global Time Master over Multiple Bus(Single Time Domain): Setting of Global Time base and user data and sending of SYNC frame over CAN and FlexRay		
ID		AUTOSAR Releases	4.2.1 4.2.2
Affected Modules	StbM, CanTSyn, FrTSyn	State	reviewed
	ATR: ATR_ATR_00133 ATR: ATR_ATR_00134		
Trace to SWS Item	TimeSyncOverCAN: SWS_CanTSyn_00028 TimeSyncOverCAN: SWS_CanTSyn_00030 TimeSyncOverFlexRay: SWS_FrTSyn_00019 TimeSyncOverFlexRay: SWS_FrTSyn_00020		
Requirements / Reference to Test Environment	Use Case UC03.01		
Parameters	StbM: StbMSynchronizedTimeBaseIdentifier = 1  CanTSyn: CanTSynGlobalTimeTxCrcSecured= CRC_NOT_SUPPORTED  FrTSyn: FrTSynGlobalTimeTxCrcSecured = CRC_NOT_SUPPORTED		
	Aim is to test the functionality of global time master a time domain and Transmission of Synchronization message over FlexRay and CAN Bus.  Verify that StbM accepts the global time base from Upper Tester using client-server		

	Interface.		
	Verify that CanTSyn shall transmit the global time base to time slave periodically via SYNC and FUP message.		
	Verify that FrTSyn shall transmit the global time base to time slave periodically via SYNC message.		
Needed Adaptation to other Releases	Not Applicable.		
Pre-conditions	SUT shall be initialized.		
Main Test Exec	ution		
Test Steps		Pass Criteria	
Step 1	[CP]		
	Start RUN_ GlobalTimeProvider.		
Step 2	_	[DIIN DIIN ClobalTimoProvidors]	
Step 2	[RUN <run_ globaltimeprovider="">]</run_>	[RUN <run_ globaltimeprovider="">]</run_>	
	Execute Rte_Call_ Client_SetGlobalTime and Rte_Call_ Client_SetUserData with below values:	Rte_Call returns RTE_E_OK.	
	timeBaseId = 1		
	StbM_TimeStampType.nanoseconds = 0x00000000		
	StbM_TimeStampType.seconds = 0x00000E10		
	StbM_TimeStampType.secondsHi = 0x0000		
	StbM_UserDataType.User Data Byte 0 = 0xAA		
	StbM_UserDataType.User Data Byte 1 = 0xBB		
	StbM_UserDataType.User Data Byte 2 = 0xCC		
Step 3	[CP]		
	Wait 100ms.		
Step 4	[CP]		
	Stort DIIN Time! leer		
Stop 5	Start RUN_TimeUser.	[DUN-DUN TimeUser-1	
Step 5	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>	
	Execute Rte_Call_ Client_GetCurrentTime.	Rte_Call returns RTE_E_OK.	
		Time received from Time user shall be	

		no montioned holous
		as mentioned below:
		StbM_TimeStampType.nanoseconds = 0x00000000 + <testwaittime =<br="">100ms&gt;</testwaittime>
		StbM_TimeStampType.seconds = 0x00000E10
		StbM_TimeStampType.secondsHi = 0x0000
		StbM_UserDataType.User Data Byte 0 = 0xAA
		StbM_UserDataType.User Data Byte 1 = 0xBB
		StbM_UserDataType.User Data Byte 2 = 0xCC
Step 6	[LT <can>]</can>	[LT]
	Receives the SYNC message.	Receives frame format as mentioned below:
		Byte 0: Type = 0x10
		Byte 1: User Byte 1 = 0xBB
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: User Byte 0 = 0xAA
		Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds
Step 7	[LT <can>]</can>	[LT]
	Receives the FUP message	Receives frame format mentioned below:
		Byte 0: Type = 0x18
		Byte 1: User Byte 2 = 0xCC
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: reserved (Bit 7 to Bit 3) = 0
		SGW (Bit 2) = 0
		OVS = Overflow of seconds (Bit 1 to Bit

		1
		0)
		Byte 4-7: SyncTimeNSec = StbM_TimeStampType.nanoseconds + <test time="100ms" wait=""></test>
Step 8	[LT <can>]</can>	[LT]
	Receives Global time base.	Get the time stamp values as below:
	Store the time as base for next periodic message processing	StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 9	[LT <can>]</can>	[LT]
	Receives the next periodic SYNC and FUP message	Time stamp values shall be as mentioned below:
		StbM_TimeStampType.nanoseconds = tB.Nano
		StbM_TimeStampType.seconds = tB.Sec
		StbM_TimeStampType.secondsHi = tB.SecHi
		StbM_UserDataType.User Data Byte 0 = 0xAA
		The difference between two time base shall be
		tB - tA = CanTSynGlobalTimeTxPeriod (2 second) + CanTSynGlobalTimeTxFollowUpOffset.
Step 10	[LT <flexray>]</flexray>	[LT]
	Receives the SYNC message	Receives frame format as mentioned below:
		Byte 0: Type = 0x10
		Byte 1: UserByte 2 = 0xCC
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: FCNT= FlexRay Cycle Counter

		0 (Bit 7 to Bit 2)
		SGW (Bit 1)
		SyncToGTM = 0
		reserved (Bit 0), default: 0
		Byte 4: User Byte 0 = 0xAA
		Byte 5: User Byte 1 = 0xBB
		Byte 6-11: SyncTimeSec = 48 Bit time stamp in seconds
		Byte 12-15: SyncTimeNSec = 32 Bit time stamp in nanoseconds
Step 11	[LT <flexray>]</flexray>	[LT]
	Receives Global time base.	Get the time stamp values as below:
	Store the time as base for next periodic message processing	StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 12	[LT <flexray>]</flexray>	[LT]
	Receives the next periodic SYNC message	Time stamp values shall be as mentioned below:
		StbM_TimeStampType.nanoseconds = tB.Nano
		StbM_TimeStampType.seconds = tB.Sec
		StbM_TimeStampType.secondsHi = tB.SecHi
		StbM_UserDataType.User Data Byte 0 = 0xAA
		StbM_UserDataType.User Data Byte 1 = 0xBB
		StbM_UserDataType.User Data Byte 2 = 0xCC
		The difference between two time base shall be

		tB – tA = FrTSynGlobalTimeTxPeriod (320ms).
Post- conditions	None	

# 6.4.2 [ATS\_GTS\_01257] Global Time Master over Multiple Bus(MultipleTime Domain): Setting of Global Time base and user data and sending of SYNC frame over CAN and FlexRay

	Global Time Master over Multiple Bus(MultipleTime Domain): Setting of Global Time base and user data and sending of SYNC frame over CAN and FlexRay		
	ATS_GTS_01257	AUTOSAR Releases	
Affected Modules	StbM, CanTSyn, FrTSyn	State	reviewed
	ATR: ATR_ATR_00133 ATR: ATR_ATR_00134		
Trace to SWS Item	TimeSyncOverCAN: SWS_CanTSyn_00038 TimeSyncOverFlexRay: SWS_FrTSyn_00023		
Requirements / Reference to Test Environment	Use Case UC03.01		
Configuration Parameters	StbM: StbMSynchronizedTimeBaseIdentifier = 1, 2		
	CanTSyn: CanTSynGlobalTimeTxCrcSecured= CRC_NOT_SUPPORTED  FrTSyn: FrTSynGlobalTimeTxCrcSecured = CRC_NOT_SUPPORTED		
Summary	Aim is to test the functionality of global time master on Multiple Time Domain and Transmission of Synchronization Messages over Multiple Bus		
	Verify that StbM accepts the global time base from Upper Tester using client-server Interface for Time Domain 1 and Time Domain 2		
	Verify that CanTSyn as Global time master shall transmit the Global time base to time slaves periodically via SYNC and FUP message.		
	Verify that FrTSyn as Global time master shall transmit the Global Time Base to time slave periodically via SYNC message.		
Needed Adaptation to other Releases	Not Applicable.		
Pre-conditions	SUT shall be initialized.		
Main Test Exec	ution		
Test Steps			Pass Criteria
Step 1	[CP] Start RUN_ GlobalTimeProvider.		
106 of 122	otati (1014_ Global i liller lovidel.		nument ID 941: AUTOSAP ATS ClohalTimeSynchronization

Step 2	[RUN <run_ globaltimeprovider="">]</run_>	[RUN <run_ globaltimeprovider="">]</run_>
	Execute Rte_Call_ Client_SetGlobalTime and Rte_Call_ Client_SetUserData with below values:	Rte_Call returns RTE_E_OK.
	timeBaseId = 1	
	StbM_TimeStampType.nanoseconds = 0x00000000	
	StbM_TimeStampType.seconds = 0x00000E10	
	StbM_TimeStampType.secondsHi = 0x0000	
	StbM_UserDataType.User Data Byte 0 = 0xAA	
	StbM_UserDataType.User Data Byte 1 = 0xBB	
Step 3	[CP]	
	Wait 100ms.	
Step 4	[CP]	
	Start RUN_TimeUser.	
	_	
Step 5	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
Step 5	-	[RUN <run_timeuser>] Rte_Call returns RTE_E_OK.</run_timeuser>
Step 5	Execute Rte_Call_ Client_GetCurrentTime.	
Step 5	Execute Rte_Call_ Client_GetCurrentTime.	Rte_Call returns RTE_E_OK.  Time received from Time user shall be
Step 5	Execute Rte_Call_ Client_GetCurrentTime.	Rte_Call returns RTE_E_OK.  Time received from Time user shall be as mentioned below:  StbM_TimeStampType.nanoseconds = 0x000000000 + <testwaittime =<="" th=""></testwaittime>
Step 5	Execute Rte_Call_ Client_GetCurrentTime.	Rte_Call returns RTE_E_OK.  Time received from Time user shall be as mentioned below:  StbM_TimeStampType.nanoseconds = 0x000000000 + <testwaittime 100ms="" =="">  StbM_TimeStampType.seconds =</testwaittime>
Step 5	Execute Rte_Call_ Client_GetCurrentTime.	Rte_Call returns RTE_E_OK.  Time received from Time user shall be as mentioned below:  StbM_TimeStampType.nanoseconds = 0x000000000 + <testwaittime 100ms="" =="">  StbM_TimeStampType.seconds = 0x000000E10  StbM_TimeStampType.secondsHi =</testwaittime>
Step 5	Execute Rte_Call_ Client_GetCurrentTime.	Rte_Call returns RTE_E_OK.  Time received from Time user shall be as mentioned below:  StbM_TimeStampType.nanoseconds = 0x000000000 + <testwaittime 100ms="" =="">  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000  StbM_UserDataType.User Data Byte 0</testwaittime>
Step 5	Execute Rte_Call_ Client_GetCurrentTime.	Rte_Call returns RTE_E_OK.  Time received from Time user shall be as mentioned below:  StbM_TimeStampType.nanoseconds = 0x000000000 + <testwaittime 100ms="" =="">  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000  StbM_UserDataType.User Data Byte 0 = 0xAA  StbM_UserDataType.User Data Byte 1</testwaittime>
	Execute Rte_Call_ Client_GetCurrentTime.  [LT <can>]  Receives the SYNC message</can>	Rte_Call returns RTE_E_OK.  Time received from Time user shall be as mentioned below:  StbM_TimeStampType.nanoseconds = 0x000000000 + <testwaittime 100ms="" =="">  StbM_TimeStampType.seconds = 0x00000E10  StbM_TimeStampType.secondsHi = 0x0000  StbM_UserDataType.User Data Byte 0 = 0xAA  StbM_UserDataType.User Data Byte 1 = 0xBB</testwaittime>

		Byte 0: Type = 0x10
		Byte 1: User Byte 1 = 0xBB
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: User Byte 0 = 0xAA
		Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds
Step 7	[LT <can>]</can>	[LT]
	Receives the FUP message	Receives frame format mentioned below:
		Byte 0: Type = 0x18
		Byte 1: User Byte 2 = 0x00
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: reserved (Bit 7 to Bit 3) = 0
		SGW (Bit 2) = 0
		OVS = Overflow of seconds (Bit 1 to Bit 0)
		Byte 4-7: SyncTimeNSec = StbM_TimeStampType.nanoseconds + <test time="100ms" wait=""></test>
Step 8	[LT <can>]</can>	[LT]
	Receives Global time base.	Get the time stamp values as below:
	Store the time as base for next periodic message processing	StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 9	[LT <can>]</can>	[LT]
	Receives the next periodic SYNC and FUP message	Time stamp values shall be as mentioned below:
		StbM_TimeStampType.nanoseconds = tB.Nano
		I

	_	
		StbM_TimeStampType.seconds = tB.Sec
		StbM_TimeStampType.secondsHi = tB.SecHi
		StbM_UserDataType.User Data Byte 0 = 0xAA
		The difference between two time base shall be
		tB – tA = CanTSynGlobalTimeTxPeriod (2 second) + CanTSynGlobalTimeTxFollowUpOffset.
Step 10	[CP]	
	<b>,</b>	
	Start RUN_ GlobalTimeProvider.	
Step 11	[RUN <run_ globaltimeprovider="">]</run_>	[RUN <run_ globaltimeprovider="">]</run_>
	Execute Rte_Call_ Client_SetGlobalTime and Rte_Call_ Client_SetUserData with below values:	Rte_Call returns RTE_E_OK.
	timeBaseId = 2	
	StbM_TimeStampType.nanoseconds = 0x00000000	
	StbM_TimeStampType.seconds = 0x00001C20	
	StbM_TimeStampType.secondsHi = 0x0000	
	StbM_UserDataType.User Data Byte 0 = 0xAA	
	StbM_UserDataType.User Data Byte 1 = 0xBB	
	StbM_UserDataType.User Data Byte 2 = 0xCC	
Step 12	[CP]	
	N. 7. 400	
01 40	Wait 100ms.	
Step 13	[CP]	
	Start RUN_TimeUser.	
Step 14	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime for timeBaseId = 2	Rte_Call returns RTE_E_OK.
		Time received from Time user shall be

		as mentioned below:
		StbM_TimeStampType.nanoseconds = 0x000000000 + <testwaittime 100ms="" ==""></testwaittime>
		StbM_TimeStampType.seconds = 0x00001C20
		StbM_TimeStampType.secondsHi = 0x0000
		StbM_UserDataType.User Data Byte 0 = 0xAA
		StbM_UserDataType.User Data Byte 1 = 0xBB
		StbM_UserDataType.User Data Byte 2 = 0xCC
Step 15	[LT <flexray>]</flexray>	[LT]
	Receives the SYNC message	Receives frame format as mentioned below:
		Byte 0: Type = 0x10
		Byte 1: UserByte 2 = 0xCC
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: FCNT= FlexRay Cycle Counter 0 (Bit 7 to Bit 2)
		SGW (Bit 1)
		SyncToGTM = 0
		reserved (Bit 0), default: 0
		Byte 4: User Byte 0 = 0xAA
		Byte 5: User Byte 1 = 0xBB
		Byte 6-11: SyncTimeSec = 48 Bit time stamp in seconds
		Byte 12-15: SyncTimeNSec = 32 Bit time stamp in nanoseconds + <test Wait Time = 100ms&gt;</test 
Step 16	[LT <flexray>]</flexray>	[LT]
		1

	Receives Global time base.	Get the time stamp values as below:
	Store the time as base for next periodic message processing	StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 17	[LT <flexray>]</flexray>	[LT]
	Receives the next periodic SYNC message	Time stamp values shall be as mentioned below:
		StbM_TimeStampType.nanoseconds = tB.Nano
		StbM_TimeStampType.seconds = tB.Sec
		StbM_TimeStampType.secondsHi = tB.SecHi
		StbM_UserDataType.User Data Byte 0 = 0xAA
		StbM_UserDataType.User Data Byte 1 = 0xBB
		StbM_UserDataType.User Data Byte 2 = 0xCC
		The difference between two time base shall be
		tB – tA = FrTSynGlobalTimeTxPeriod (320ms).
Post- conditions	None	

## 6.4.3 [ATS\_GTS\_01258] Time Gateway- Time Slave on FlexRay and Time Master on CAN

Test Objective	Time Gateway- Time Slave on FlexRay and Time Master on CAN		
ID	ATS_GTS_01258		
Affected Modules	StbM, CanTSyn, FrTSyn	State	reviewed
	ATR: ATR_ATR_00133 ATR: ATR_ATR_00134		

	L	
Trace to SWS	TimeSyncOverCAN: SWS_CanTSyn_00028	
Item	TimeSyncOverCAN: SWS_CanTSyn_00030 TimeSyncOverFlexRay: SWS_FrTSyn_00046	3
Requirements /	Use Case UC03.02	,
Reference	Use Case UCU3.02	
to Test		
Environment		
Configuration	StbM:	
Parameters	StbMlsSystemWideGlobalTimeMaster = FALS	SE
	StbMSynchronizedTimeBaseIdentifier = 1	
	CanTSyn:	
	CanTSynGlobalTimeDomainId = 1	
	CanTSynSynchronizedTimeBaseRef = Reference	
	CanTSynGlobalTimeTxCrcSecured= CRC_N	
	CanTSynGlobalTimeTxFollowUpOffset = 100 CanTSynGlobalTimeTxPeriod = 2000ms	THS
	CanTSynMasterConfirmationTimeout = 80ms	3
	FrTSyn:	ATED
	FrTSynRxCrcValidated = CRC_NOT_VALIDA FrTSynGlobalTimeDomainId = 1	4 I ED
Summary	Aim is to verify the functionality of time gatew	av
Odiffinal y	Aim is to verify the functionality of time gatew	ay
	Verify that FrTSyn as time slave shall receive	the global time base periodically via
	SYNC message.	
	L	
	Verify that StbM synchronizes the local time a	
	from FrTSyn (using API StbM_BusSetGlobal	i me)
	Verify that CanTSyn as time master shall transmit the global time base received	
	from global time domain to time subdomain p	
	message respectively.	
	(	. 0 0
	Verify that UT shall get the valid current time	using Client-Server Interface.
Needed Adaptation to	Not Applicable.	
other Releases		
Pre-conditions	SUT shall be initialized.	
Main Test Execu		
Test Steps		Pass Criteria
Step 1	[LT <flexray>]</flexray>	[SUT]
·	·	[ -
	Transmit SYNC message with	Receives SYNC message ignoring
		CRC value(if there) in below format
	timeBaseId = 1	D. 45 O. T. 75 O. 40
	CthM HeerDeteTime Heer Dete Dite 0	Byte 0: Type = 0x10
	StbM_UserDataType.User Data Byte 0 = 0xAA	Byte 1: User Byte 2
		2,10 1. 000. 2,10 Z
	StbM_UserDataType.User Data Byte 1 =	Byte 2: D = <time domain="1"> (Bit 7</time>
	0xBB	to Bit 4)
	l	
	And time value as T0	SC = <sequence counter="0"> (Bit 3</sequence>
	Where TO = TSVNC + (MagretickePerCycle *	to Bit 0)
	Where T0 = TSYNC + (MacroticksPerCycle *	

	(64 - currentCycle) - currentMacroticks) * MacrotickDuration	Byte 3: FCNT= <flexray cycle<br="">Counter = 0&gt; (Bit 7 to Bit 2)</flexray>
	And TSYNC value as given below:	SGW (Bit 1)
	StbM_TimeStampType.seconds = 0x00000E10	SyncToSubDomain = 1
	StbM_TimeStampType.secondsHi = 0x0000	reserved (Bit 0)
	StbM_TimeStampType.nanoseconds =	Byte 4: User Byte 0
	0x00000000	Byte 5: User Byte 1
		Byte 6-11: SyncTimeSec = T0.secondsHi and T0.seconds
		Byte 12-15: SyncTimeNSec = T0.nanoseconds
Step 2		[SUT]
		StbM updates its time base with values
		StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 3	[CP]	
	Wait 200ms	
Step 4	[CP]	
Otep 4	[0.1]	
	Start RUN_TimeUser	
Step 5	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime =<br="">200ms&gt;</testwaittime>
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 6	[LT <can>]</can>	[LT]
	Receives the SYNC message	Receives SYNC message in the

		format mentioned below:
		Byte 0: Type = 0x10
		Byte 1: User Byte 1 = 0xBB
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: User Byte 0 = 0xAA
		Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds
Step 7	[LT <can>]</can>	[LT]
	Receives the FUP message	Receives FUP message in the format mentioned below:
		Byte 0: Type = 0x18
		Byte 1: User Byte 2 = 0x00
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: reserved (Bit 7 to Bit 3) = 0
		SGW (Bit 2) = 0
		OVS = Overflow of seconds (Bit 1 to Bit 0)
21 2	71.7. OAN 1	Byte 4-7: SyncTimeNSec = StbM_TimeStampType.nanoseconds
Step 8	[LT <can>]</can>	[LT]
	Receives Global time base.	Get the time stamp values as below:
	Store the time as base for next periodic message processing	StbM_TimeStampType.nanoseconds = tC.Nano
		StbM_TimeStampType.seconds = tC .Sec
		StbM_TimeStampType.secondsHi = tC .SecHi
		Time Received by LT <can> should be tC (Time base transmitted by LT<flexray> + ToleranceTime_FrTSyn + ToleranceTime_CanTSyn)</flexray></can>

		ToleranceTime_FrTSyn = Max one main function of FrTSyn as reception is asynchronous
		ToleranceTime_CanTSyn = Max one main function of CanTSyn as gateway is asynchronous.
Post- conditions	None	

## 6.4.4 [ATS\_GTS\_01259] Time Gateway - Time Slave on CAN and Time Master on FlexRay

	Time Gateway - Time Slave on CAN and Time Master on FlexRay		
ID	ATS_GTS_01259	AUTOSAR Releases	4.2.1 4.2.2
Affected Modules	StbM, CanTSyn, FrTSyn	State	reviewed
	ATR: ATR_ATR_00133 ATR: ATR_ATR_00134		
Trace to SWS Item	SynchronizedTimeBaseManager: SWS_StbM_00184 TimeSyncOverCAN: SWS_CanTSyn_00064 TimeSyncOverCAN: SWS_CanTSyn_00072 TimeSyncOverFlexRay: SWS_FrTSyn_00023		
Requirements / Reference to Test Environment	Use Case UC03.02		
Configuration Parameters	StbM: StbMlsSystemWideGlobalTimeMaster = FALSE StbMSynchronizedTimeBaseIdentifier = 1  CanTSyn: CanTSynGlobalTimeDomainId = 1 CanTSynGlobalTimeFollowUpTimeout = 300ms CanTSynGlobalTimeSequenceCounterJumpWidth = 2 CanTSynSynchronizedTimeBaseRef = Reference to StbMSynchronizedTimeBase CanTSynRxCrcValidated = CRC_IGNORED CanTSynGlobalTimeSlaveHandIeId = 0  FrTSyn: FrTSynGlobalTimeDomainId = 1 FrTSynGlobalTimeSequenceCounterJumpWidth = 2 FrTSynSynchronizedTimeBaseRef = Reference to StbMSynchronizedTimeBase FrTSynGlobalTimeTxCrcSecured = CRC_NOT_SUPPORTED FrTSynGlobalTimeTxPeriod = 320ms		
Summary	FrTSynGlobalTimePduRef Refere Aim is to verify the functionality of Verify that CanTSyn as time slave via SYNCand FUP message.	f time gatew	ay ve the global offset time periodically

	Verify that StbM synchronizes the offset time as per the received global time base from CanTSyn (using API StbM_SetOffset)		
	Verify that FrTSyn as time master shall transmit the offset time base received from global time domain to time sub-domain periodically via OFS message respectively.		
	Verify that UT shall get the valid current time using Client-Server Interface.		
Needed	Not Applicable.		
Adaptation to other Releases	пот Арріісавіе.		
Pre-conditions	SUT shall be initialized.		
Main Test Execu	ıtion		
Test Steps		Pass Criteria	
-	[LT <can>]</can>	[SUT]	
		[55.]	
	Transmit SYNC message with	Receives SYNC message ignoring CRC value in below format	
	timeBaseId = 1	Byte 0: Type = 0x10	
	StbM_TimeStampType.seconds = 0x00000E10	Byte 1: User Byte 1 = 0xBB	
	StbM_TimeStampType.secondsHi = 0x0000	Byte 2: D = 0x1	
	StbM_UserDataType.User Data Byte 0 = 0xAA	SC = 0x0	
	StbM_UserDataType.User Data Byte 1 = 0xBB	Byte 3: User Byte 0 = 0xAA	
		Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds	
Step 2	[LT <can>]</can>	[SUT]	
	Transmit FUP message with	Receives FUP message in below format	
	StbM_TimeStampType.nanoseconds = t4r	Byte 0: Type = 0x18	
		Byte 1: User Byte 2 = 0x00	
		Byte 2: D = 0x1	
		SC = 0x0	
		Byte 3: reserved (Bit 7 to Bit 3) = 0	
		SGW (Bit 2) = 0	
		OVS = Overflow of seconds (Bit 1 to Bit 0)	
		Byte 4-7: SyncTimeNSec = StbM_TimeStampType.nanoseconds	
Step 3		[SUT]	

Step 4	[CP]	StbM updates its time base with values  StbM_TimeStampType.nanoseconds = tA.Nano  StbM_TimeStampType.seconds = tA .Sec  StbM_TimeStampType.secondsHi = tA .SecHi
	Wait 800ms	
Step 5	[CP]	
	Start RUN_TimeUser	
Step 6	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime =<br="">800ms&gt;</testwaittime>
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 7	[LT <flexray>]</flexray>	[LT]
	Receives the SYNC message without CRC validation.	Receives frame format as mentioned below:
		Byte 0: Type = 0x10
		Byte 1: UserByte 2
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: FCNT= FlexRay Cycle Counter 0 (Bit 7 to Bit 2)
		SGW (Bit 1)
		SyncToSubDomain = 1
		reserved (Bit 0), default: 0

		Byte 4: User Byte 0
		Byte 5: User Byte 1
		Byte 6-11: SyncTimeSec = 48 Bit time stamp in seconds
		Byte 12-15: SyncTimeNSec = 32 Bit time stamp in nanoseconds
Step 8	[LT <flexray>]</flexray>	[LT]
	Receives Global time base.	Get the time stamp values as below:
	Store the time as base for next periodic message processing	StbM_TimeStampType.nanoseconds = tC.Nano
		StbM_TimeStampType.seconds = tC .Sec
		StbM_TimeStampType.secondsHi = tC .SecHi
		Time Received by LT <flexray> should be tC (Time base transmitted by LT<can> + ToleranceTime_CanTSyn + ToleranceTime_FrTSyn )</can></flexray>
		ToleranceTime_CanTSyn = Max one main function of CanTSyn as reception is asynchronous
		ToleranceTime_FrTSyn = Max one main function of FrTSyn as gateway is asynchronous
Post- conditions	None	

## 6.4.5 [ATS\_GTS\_01260] Time Gateway - Time Slave on CAN (Network 1) and Time Master on CAN (Network 2)

	Time Gateway - Time Slave on CAN (Network 1) and Time Master on CAN (Network 2)		
ID		AUTOSAR Releases	4.2.1 4.2.2
Affected Modules	StbM, CanTSyn	State	reviewed
Trace to Requirement on Acceptance Test Document	ATR: ATR_ATR_00133		
Trace to SWS Item	SynchronizedTimeBaseManager: SWS_StbM_00184 TimeSyncOverCAN: SWS_CanTSyn_00028		

	TimeSynaOyerCAN; SWS CanTSyn 00020			
	TimeSyncOverCAN: SWS_CanTSyn_00030 TimeSyncOverCAN: SWS_CanTSyn_00064			
Requirements / Reference to Test Environment	Use Case UC03.02			
	C+b M.			
Configuration Parameters	StbM: StbMIsSystemWideGlobalTimeMaster = FALSE StbMSynchronizedTimeBaseIdentifier = 1			
	CanTSyn for Master: CanTSynGlobalTimeDomainId = 1 CanTSynGlobalTimeSequenceCounterJumpWidth = 2 CanTSynSynchronizedTimeBaseRef = Reference to StbMSynchronizedTimeBase CanTSynGlobalTimeTxCrcSecured= CRC_NOT_SUPPORTED CanTSynGlobalTimeTxFollowUpOffset = 100ms CanTSynGlobalTimeTxPeriod = 2000ms CanTSynMasterConfirmationTimeout = 80ms  CanTSyn for slave:			
	CanTSynGlobalTimeFollowUpTimeout = 300ms CanTSynRxCrcValidated = CRC_IGNORED			
Summary	Aim is to verify the functionality of time gateway			
	Verify that CanTSyn as time slave(CAN 1) shall receive the global time base periodically via SYNC and FUP message.			
	Verify that StbM synchronizes the local time as per the received global time base from CanTSyn (using API StbM_BusSetGlobalTime)  Verify that CanTSyn as time master(CAN 2) shall transmit the global time base received from global time domain to time sub-domain periodically via SYNC and FUP message.			
	Verify that UT shall get the valid current time using Client-Server Interface.			
Needed Adaptation to other Releases	Not Applicable.			
Pre-conditions	SUT shall be initialized.			
Main Test Execu				
Test Steps		Pass Criteria		
Step 1	T	[SUT]		
otop i		Receives SYNC message ignoring CRC value in below format		
	timeBaseId = 1	Byte 0: Type = 0x10		
	StbM_TimeStampType.seconds = 0x00000E10	Byte 1: User Byte 1 = 0xBB		
	StbM_TimeStampType.secondsHi = 0x0000	Byte 2: D = 0x1		
	StbM_UserDataType.User Data Byte 0 = 0xAA	SC = 0x0		
	StbM_UserDataType.User Data Byte 1 =	Byte 3: User Byte 0 = 0xAA		

	0xBB	Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds
Step 2	[LT <can>]</can>	[SUT]
	Transmit FUP message with	Receives FUP message ignoring CRC value in below format
	StbM_TimeStampType.nanoseconds = t4r	Byte 0: Type = 0x18
	Byte CRC = Invalid CRC	Byte 1: User Byte 2 =0x00
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: reserved (Bit 7 to Bit 3) = 0
		SGW (Bit 2) = 0
		OVS = Overflow of seconds (Bit 1 to Bit 0)
		Byte 4-7: SyncTimeNSec = StbM_TimeStampType.nanoseconds
Step 3		[SUT]
		StbM updates its time base with values
		StbM_TimeStampType.nanoseconds = tA.Nano
		StbM_TimeStampType.seconds = tA .Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 4	[CP]	
	Wait 800ms	
Step 5	[CP]	
	Start RUN_TimeUser	
Step 6	[RUN <run_timeuser>]</run_timeuser>	[RUN <run_timeuser>]</run_timeuser>
	Execute Rte_Call_ Client_GetCurrentTime	Rte_Call returns RTE_E_OK.
		Time read at Time user shall be:
		StbM_TimeStampType.nanoseconds = tA.Nano + <testwaittime =<br="">800ms&gt;</testwaittime>
		StbM_TimeStampType.seconds = tA

		.Sec
		.Sec
		StbM_TimeStampType.secondsHi = tA .SecHi
Step 7	[LT <can>]</can>	[LT]
	Receives the SYNC message	Receives SYNC message in the format mentioned below:
		Byte 0: Type = 0x10
		Byte 1: User Byte 1 = 0xBB
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: User Byte 0 = 0xAA
		Byte 4-7: SyncTimeSec = StbM_TimeStampType.seconds
Step 8	[LT <can>]</can>	[LT]
	Receives the FUP message	Receives FUP message in the format mentioned below:
		Byte 0: Type = 0x18
		Byte 1: User Byte 2 = 0x00
		Byte 2: D = 0x1
		SC = 0x0
		Byte 3: reserved (Bit 7 to Bit 3) = 0
		SGW (Bit 2) = 0
		OVS = Overflow of seconds (Bit 1 to Bit 0)
		Byte 4-7: SyncTimeNSec = StbM_TimeStampType.nanoseconds
Step 9	[LT <can>]</can>	[LT]
	Receives Global time base.	Get the time stamp values as below:
	Store the time as base for next periodic message processing	StbM_TimeStampType.nanoseconds = tC.Nano
		StbM_TimeStampType.seconds = tC .Sec
		StbM_TimeStampType.secondsHi =

## AUTOSAR TC Release 1.2.0

		tC .SecHi
		Time Received by LT <can> should be tC (Time base transmitted by LT<flexray> + ToleranceTime_CanTSyn + ToleranceTime_CanTSyn).</flexray></can>
		ToleranceTime_CanTSyn = Max one main function of CanTSyn as reception is asynchronous.
		ToleranceTime_CanTSyn = Max one main function of CanTSyn as gateway is asynchronous.
Post- conditions	None	