Document Title	Specification of Time Synchronization over FlexRay
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	675

Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R24-11

Document Change History

Date	Release	Changed by	Description
2024-11-27	R24-11	AUTOSAR Release Managemen	• Offset Time Domains removed • Changed lower layer from FrIf to LSduR
2023-11-23	R23-11	AUTOSAR Release Management	• Clarification of / refinement of sequence counter validation • Clarification of / refinement of Timesync message transmission and debouncing behavior • Incorporation of validation findings for "Secured Time Synchronization"
2022-11-24	R22-11	AUTOSAR Release Management	• Support for "Secured Time Synchronization" added • Minor content changes, clarifications
2021-11-25	R21-11	AUTOSAR Release Management	• Hysteresis added for sequence counter validation • Small enhancement to improve precision of Global Time • Bugfix for Time Validation
2020-11-30	R20-11	AUTOSAR Release Management	• Time Validation updated for Time Gateways • Post build variant value corrected for FrTSynGlobalTimeMasterHandleId and FrTSynGlobalTimeSlaveHandleId

▽

△

2019-11-28	R19-11	AUTOSAR Release Management	• Time Validation (draft) • Clarification regarding messages with stuck sequence counter • Clarification regarding cyclic operation entry after timebase startup • Clarification regarding transmission and reception of User Bytes • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	• Modifications to enhance the precision of Global Time Synchronization • Additional minor corrections / clarifications / editorial changes; For details please refer to the Change Documentation
2016-11-30	4.3.0	AUTOSAR Release Management	• Offset message formats changed • Immediate Time Synchronization message transmission • Various enhancements and corrections
2015-07-31	4.2.2	AUTOSAR Release Management	• Error code FRTSYN_E_INVALID_PDU_SDU_ID replaced by FRTSYN_E_INVALID_PDUID • FlexRay communication state handling simplified (FrIf_GetPOCStatus replaced by FrIf_GetState)
2014-10-31	4.2.1	AUTOSAR Release Management	• Initial Release

Disclaimer

This work (specification and/or software implementation) 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 work.

The material contained in this work is protected by copyright and other types of intellectual property rights. The commercial exploitation of the material contained in this work requires a license to such intellectual property rights.

This work 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 work may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The work has been developed for automotive applications only. It has neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Contents

1 Introduction and functional overview

The FrTSyn module handles the distribution of time information over FlexRay buses.

The FlexRay mechanism is much simpler than the mechanism for CAN since it is based on the fact, that FlexRay nodes are synchronized to each other, otherwise no messages can be transmitted on FlexRay.

Both, Time Master and Time Slaves have the same view on the FlexRay global time. It is therefore just necessary to define the same point in (FlexRay) time and to transmit the time information, which will be valid at that point in (FlexRay) time.

Although this same point in (FlexRay) time could be in theory any FlexRay macrotick within a FlexRay cycle, the start of a FlexRay cycle simplifies this mechanism. In addition, the mechanism does not just use any cycle start but uses the cycle start of the subsequent cycle with cycle counter value 0, i.e. the Time Master transmits time information located in the future.

On FlexRay only one Time Synchronization message is needed.

The Time Master uses its current FlexRay time, i.e. macrotick counter and cycle counter, and the current time, which shall be distributed and calculates the resulting time at the start of the next cycle 0. Once this resulting time has been calculated, it is neither very time critical, when exactly the FlexRay frame is transmitted, nor when it is received and processed.

Every Time Slave receiving the transmitted time information will use it in combination with the current FlexRay macrotick counter and cycle counter to determine the actual master time and set its slave time.

Figure 1.1 illustrates the Time Synchronization mechanism on FlexRay.

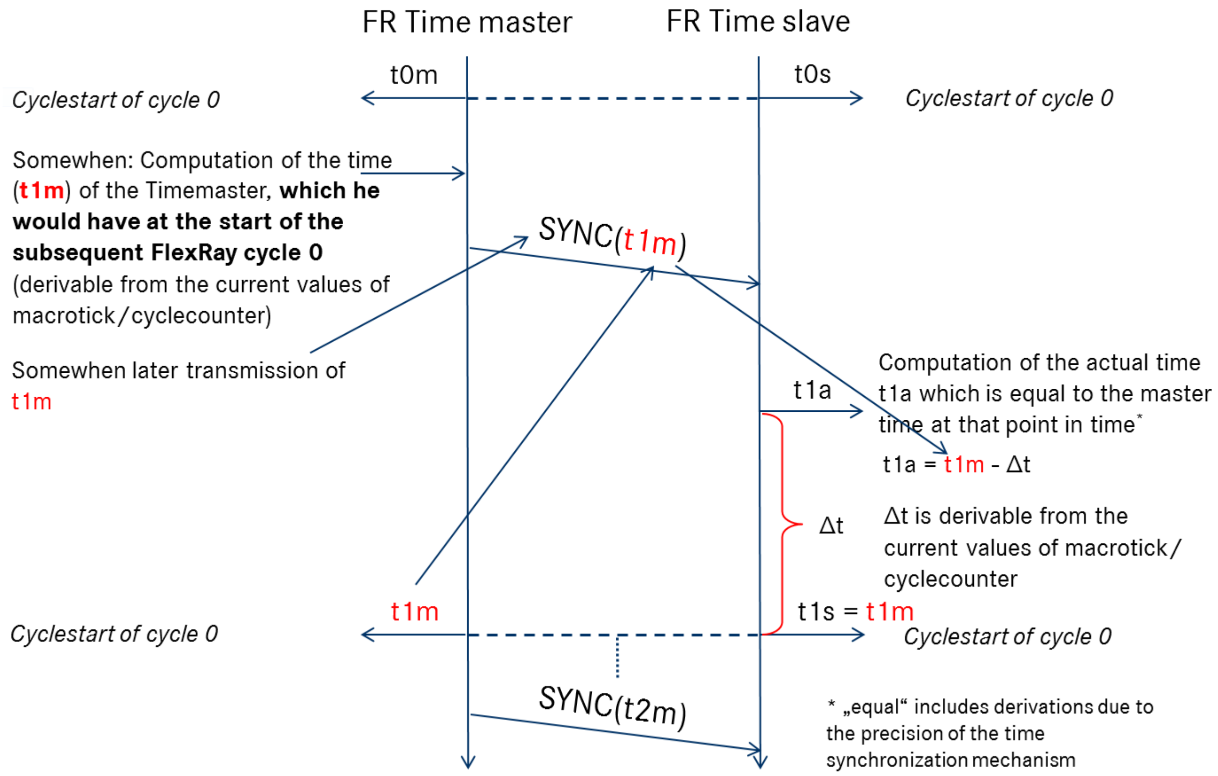

FR Time master
FR Time slave
t0m
t0s
Cyclestart of cycle 0
Cyclestart of cycle 0
Somewhen: Computation of the time (t1m) of the Timemaster, which he would have at the start of the subsequent FlexRay cycle 0 (derivable from the current values of macrotick/cyclecounter)
SYNC(t1m)
Somewhen later transmission of t1m
t1a
Computation of the actual time t1a which is equal to the master time at that point in time*
t1a = t1m - Δt
Δt
Δt is derivable from the current values of macrotick/cyclecounter
t1m
t1s = t1m
Cyclestart of cycle 0
Cyclestart of cycle 0
SYNC(t2m)
* „equal“ includes derivations due to the precision of the time synchronization mechanism

Figure 1.1: FlexRay Time Synchronization Mechanism

The FrTSyn also supports securing the global time messages on the FlexRay communication bus. The figure below shows the time provider modules interface with the security modules in the AUTOSAR Layered Architecture.

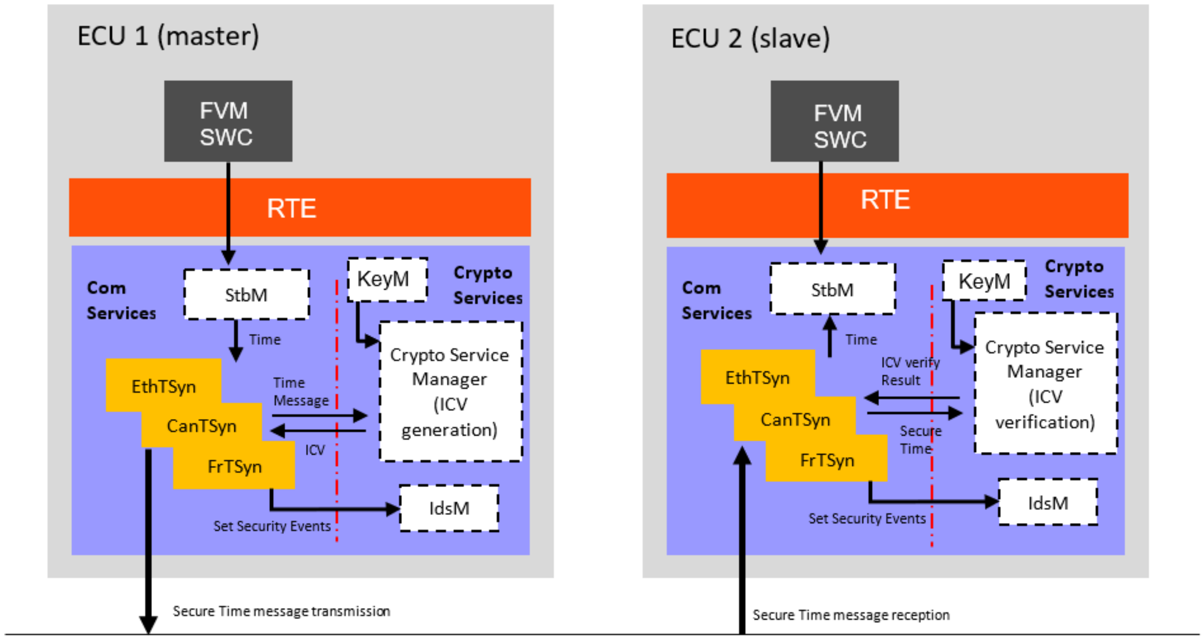


Figure 1.2: Timesync modules interface with security modules in the AUTOSAR Layered Architecture

2 Acronyms, Abbreviations, and Definitions

This section lists module local abbreviations and definitions. For additional Time Synchronization related abbreviations and definitions refer to chapter 3 in the RS Time Synchronization [1]. For general terms and abbreviations refer to the AUTOSAR Glossary [2].

Abbreviation	Description
GTM	Global Time Master
BswM	BSW Mode Manager module
<Bus>TSyn	Bus specific Time Synchronization module
CRC	Cyclic Redundancy Checksum
CSM	Crypto Service Manager
Debounce Time	Minimum gap between two Tx messages with the same PDU
Det	Default Error Tracer module
FCNT	FlexRay Cycle Counter
FR	FlexRay
FrIf	FlexRay interface module
FrTSyn	Time Synchronization over FlexRay module
FV	Freshness Value
FVL	Freshness Value Length
FVM	Freshness Value Manager
ICV	Integrity Check Value
ICVL	Integrity Check Value Length
IdsM	Intrusion Detection System Manager module
LSduR	L-SDU Router module
MAC	Message Authentication Code
SC	Sequence Counter in Time Synchronization messages
SGW	"Synchronized to Gateway" state of Time Synchronization
StbM	Synchronized Time-Base Manager
SYNC message	Time Synchronization message
Timesync	Time Synchronization

3 Related documentation

3.1 Input documents & related standards and norms

[1] Requirements on Time Synchronization
AUTOSAR_FO_RS_TimeSync

[2] Glossary
AUTOSAR_FO_TR_Glossary

[3] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral

[4] General Requirements on Basic Software Modules
AUTOSAR_CP_RS_BSWGeneral

[5] Specification of Synchronized Time-Base Manager
AUTOSAR_CP_SWS_SynchronizedTimeBaseManager

[6] Specification of CRC Routines
AUTOSAR_CP_SWS_CRCLibrary

[7] Specification of Crypto Service Manager
AUTOSAR_CP_SWS_CryptoServiceManager

[8] Specification of Intrusion Detection System Manager
AUTOSAR_CP_SWS_IntrusionDetectionSystemManager

3.2 Related specification

AUTOSAR provides a General Specification on Basic Software modules [3, SWS BSW General], which is also valid for FrTSyn.

Thus, the specification SWS BSW General shall be considered as additional and required specification for FrTSyn.

4 Constraints and assumptions

4.1 Limitations

Time Masters, Time Gateways and Time Slaves shall work with a Time Base reference clock with a worst-case accuracy of 2μs.

4.2 Applicability to car domains

Automotive systems requiring a common Time Base for ECUs regardless of which bus system the ECUs are connected to.

5 Dependencies to other modules

The Time Synchronization over FlexRay (FrTSyn) has interfaces towards the Synchronized Time-Base Manager (StbM), the FlexRay Interface (FrIf), the L-SDU Router (LSduR), the BSW Mode Manager (BswM), the Default Error Tracer (Det), the Crypto Service Manager (CSM), and the Intrusion Detection System Manager (IdsM).

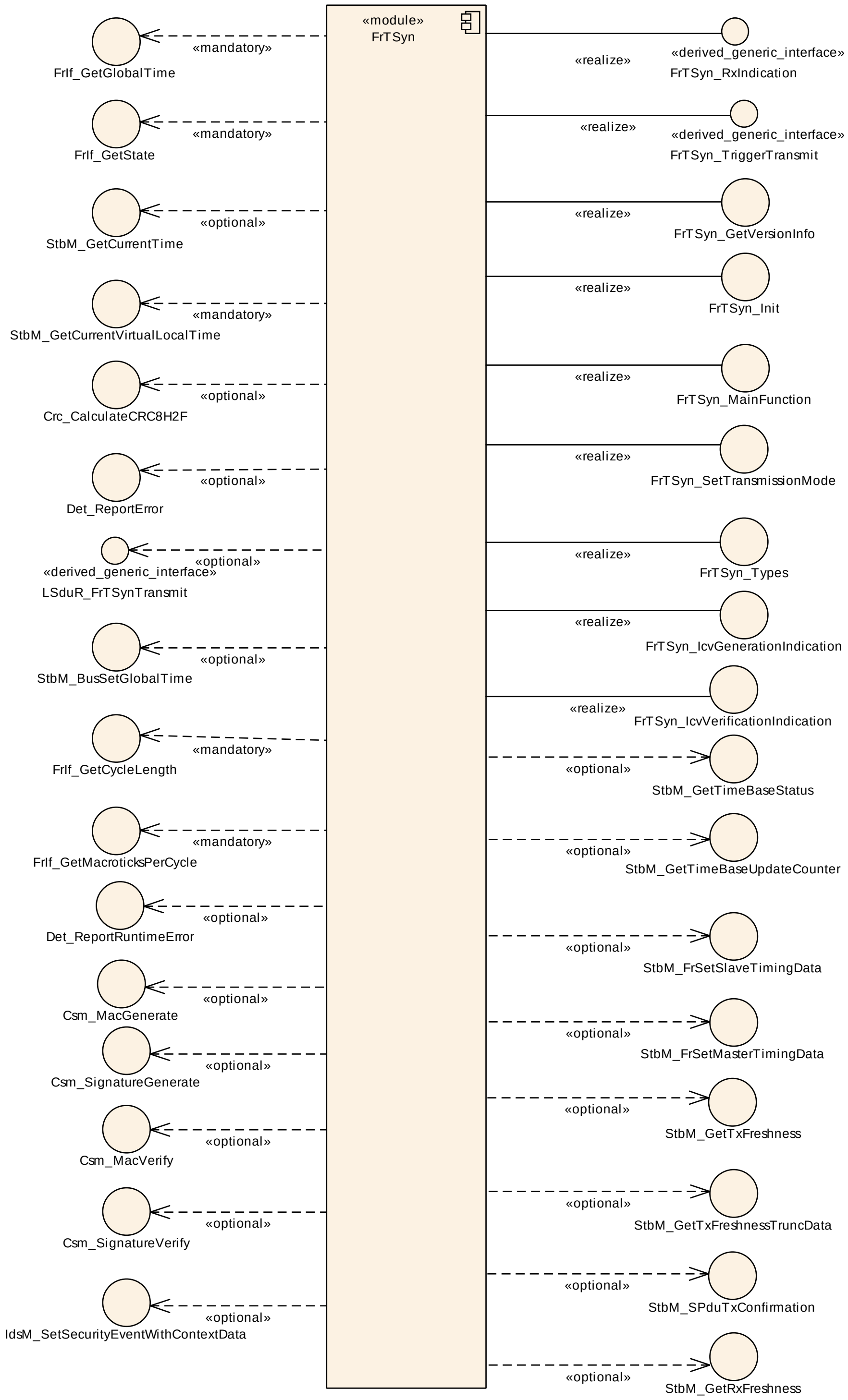


Figure 5.1: Module dependencies of the FrTSyn module

- StbM -
  - Get and set the current time value
  - Get FV from FVM
- LSduR - Receiving and transmitting messages
- FrIf - Accessing the clock unit in the FlexRay controller
- BswM - Coordination of network access (via FrTSyn_SetTransmissionMode)
- Det - Reporting of development errors
- IdsM - Reporting of Security Events
- CSM -
  - Generation of ICV for Time Master
  - Verification of ICV for Time Slave

5.1 File structure

5.1.1 Code file structure

For details, refer to the section 5.1.6 "Code file structure" of the SWS BSW General [3].

5.1.2 Header file structure

For details, refer to the section 5.1.7 "Header file structure" of the SWS BSW General [3].

6 Requirements Tracing

The following tables reference the requirements specified in [1, RS TimeSync] and [4, SRS BSWGeneral] and links to the fulfillment of these. Please note that if column "Satisfied by" is empty for a specific requirement this means that this requirement is not fulfilled by this document.

Requirement	Description	Satisfied by
[RS_Ids_00810]	Basic SW security events	[SWS_FrTSyn_00103] [SWS_FrTSyn_92000] [SWS_FrTSyn_92001] [SWS_FrTSyn_92002]
[RS_TS_00003]	The TS shall initialize the Local Time Base with a configurable startup value	[SWS_FrTSyn_00003]
[RS_TS_00004]	The Implementation of Time Synchronization shall initialize the Global Time Base with a configurable startup value.	[SWS_FrTSyn_00003]
[RS_TS_00034]	The Implementation of Time Synchronization shall provide measurement data to the application	[SWS_FrTSyn_00092] [SWS_FrTSyn_00096] [SWS_FrTSyn_00097] [SWS_FrTSyn_00098] [SWS_FrTSyn_00099] [SWS_FrTSyn_00100] [SWS_FrTSyn_00101]
[RS_TS_20039]	The Timesync over FlexRay module shall trigger Time Base Synchronization transmission	[SWS_FrTSyn_00019] [SWS_FrTSyn_00026] [SWS_FrTSyn_00027] [SWS_FrTSyn_00085] [SWS_FrTSyn_00086] [SWS_FrTSyn_00087] [SWS_FrTSyn_00088] [SWS_FrTSyn_00090] [SWS_FrTSyn_00091] [SWS_FrTSyn_00093] [SWS_FrTSyn_00161] [SWS_FrTSyn_00169] [SWS_FrTSyn_00170] [SWS_FrTSyn_00173]
[RS_TS_20040]	The Timesync over FlexRay module shall provide a Time Base after reception of a valid protocol information	[SWS_FrTSyn_00041] [SWS_FrTSyn_00078] [SWS_FrTSyn_00094]
[RS_TS_20041]	The Timesync over FlexRay module shall support means to protect the Time Synchronization protocol	[SWS_FrTSyn_00006] [SWS_FrTSyn_00014] [SWS_FrTSyn_00015] [SWS_FrTSyn_00021] [SWS_FrTSyn_00030] [SWS_FrTSyn_00031] [SWS_FrTSyn_00035] [SWS_FrTSyn_00036] [SWS_FrTSyn_00078] [SWS_FrTSyn_00106] [SWS_FrTSyn_00107] [SWS_FrTSyn_00112] [SWS_FrTSyn_00162] [SWS_FrTSyn_00163] [SWS_FrTSyn_00164] [SWS_FrTSyn_00165] [SWS_FrTSyn_00166] [SWS_FrTSyn_00167] [SWS_FrTSyn_00168]
[RS_TS_20042]	The Timesync over FlexRay module shall detect and handle timeout and integrity errors in the Time Synchronization protocol	[SWS_FrTSyn_00015] [SWS_FrTSyn_00038] [SWS_FrTSyn_00041] [SWS_FrTSyn_00050] [SWS_FrTSyn_00054] [SWS_FrTSyn_00055] [SWS_FrTSyn_00057] [SWS_FrTSyn_00058] [SWS_FrTSyn_00081] [SWS_FrTSyn_00094] [SWS_FrTSyn_00107] [SWS_FrTSyn_00150] [SWS_FrTSyn_00162] [SWS_FrTSyn_00163] [SWS_FrTSyn_00164] [SWS_FrTSyn_00165] [SWS_FrTSyn_00166] [SWS_FrTSyn_00167] [SWS_FrTSyn_00168]

▽

△

Requirement	Description	Satisfied by
[RS_TS_20043]	The Timesync over FlexRay module shall support a protocol for precise time measurement and synchronization over FlexRay	[SWS_FrTSyn_00007] [SWS_FrTSyn_00009] [SWS_FrTSyn_00010] [SWS_FrTSyn_00014] [SWS_FrTSyn_00015] [SWS_FrTSyn_00018] [SWS_FrTSyn_00019] [SWS_FrTSyn_00020] [SWS_FrTSyn_00021] [SWS_FrTSyn_00026] [SWS_FrTSyn_00027] [SWS_FrTSyn_00028] [SWS_FrTSyn_00030] [SWS_FrTSyn_00031] [SWS_FrTSyn_00035] [SWS_FrTSyn_00036] [SWS_FrTSyn_00037] [SWS_FrTSyn_00038] [SWS_FrTSyn_00039] [SWS_FrTSyn_00040] [SWS_FrTSyn_00041] [SWS_FrTSyn_00046] [SWS_FrTSyn_00050] [SWS_FrTSyn_00054] [SWS_FrTSyn_00055] [SWS_FrTSyn_00056] [SWS_FrTSyn_00057] [SWS_FrTSyn_00060] [SWS_FrTSyn_00061] [SWS_FrTSyn_00062] [SWS_FrTSyn_00063] [SWS_FrTSyn_00064] [SWS_FrTSyn_00065] [SWS_FrTSyn_00066] [SWS_FrTSyn_00069] [SWS_FrTSyn_00071] [SWS_FrTSyn_00072] [SWS_FrTSyn_00074] [SWS_FrTSyn_00075] [SWS_FrTSyn_00081] [SWS_FrTSyn_00106] [SWS_FrTSyn_00107] [SWS_FrTSyn_00112] [SWS_FrTSyn_00136] [SWS_FrTSyn_00150]
[RS_TS_20045]	The Timesync over FlexRay module shall support user specific data within the time measurement and synchronization protocol	[SWS_FrTSyn_00010] [SWS_FrTSyn_00011] [SWS_FrTSyn_00012] [SWS_FrTSyn_00013]
[RS_TS_20046]	The configuration for Time synchronization over FlexRay shall allow the FlexRay Time Synchronization module to support different roles for a Time Base	[SWS_FrTSyn_00077]
[RS_TS_20074]	The Timesync over FlexRay module shall support means to secure the Time Synchronization protocol	[SWS_FrTSyn_00009] [SWS_FrTSyn_00037] [SWS_FrTSyn_00106] [SWS_FrTSyn_00107] [SWS_FrTSyn_00110] [SWS_FrTSyn_00111] [SWS_FrTSyn_00112] [SWS_FrTSyn_00114] [SWS_FrTSyn_00115] [SWS_FrTSyn_00116] [SWS_FrTSyn_00117] [SWS_FrTSyn_00118] [SWS_FrTSyn_00119] [SWS_FrTSyn_00120] [SWS_FrTSyn_00121] [SWS_FrTSyn_00122] [SWS_FrTSyn_00123] [SWS_FrTSyn_00124] [SWS_FrTSyn_00125] [SWS_FrTSyn_00126] [SWS_FrTSyn_00127] [SWS_FrTSyn_00128] [SWS_FrTSyn_00129] [SWS_FrTSyn_00130] [SWS_FrTSyn_00131] [SWS_FrTSyn_00136] [SWS_FrTSyn_00137] [SWS_FrTSyn_00138] [SWS_FrTSyn_00139] [SWS_FrTSyn_00140] [SWS_FrTSyn_00142] [SWS_FrTSyn_00143] [SWS_FrTSyn_00144] [SWS_FrTSyn_00145] [SWS_FrTSyn_00146] [SWS_FrTSyn_00147] [SWS_FrTSyn_00148] [SWS_FrTSyn_00149] [SWS_FrTSyn_00153] [SWS_FrTSyn_00154] [SWS_FrTSyn_00155] [SWS_FrTSyn_00156] [SWS_FrTSyn_00157] [SWS_FrTSyn_00158] [SWS_FrTSyn_00159] [SWS_FrTSyn_00160] [SWS_FrTSyn_00171] [SWS_FrTSyn_00172] [SWS_FrTSyn_91001] [SWS_FrTSyn_91002]
[SRS_BSW_00323]	All AUTOSAR Basic Software Modules shall check passed API parameters for validity	[SWS_FrTSyn_00058] [SWS_FrTSyn_00067] [SWS_FrTSyn_00070] [SWS_FrTSyn_00095] [SWS_FrTSyn_00151] [SWS_FrTSyn_00152]

▽

△

Requirement	Description	Satisfied by
[SRS_BSW_00337]	Classification of development errors	[SWS_FrTSyn_00067] [SWS_FrTSyn_00070] [SWS_FrTSyn_00095] [SWS_FrTSyn_00151] [SWS_FrTSyn_00152]
[SRS_BSW_00385]	List possible error notifications	[SWS_FrTSyn_00059] [SWS_FrTSyn_91000]
[SRS_BSW_00489]	Reporting of security events	[SWS_FrTSyn_00105]

Table 6.1: Requirements Tracing

7 Functional specification

This chapter defines the behavior of the Time Synchronization over FlexRay. The API of the module is defined in chapter 8, while the configuration is defined in chapter 10.

7.1 Overview

The Time Synchronization over FlexRay is responsible to ensure the collection and distribution of Synchronized Time information across the FlexRay network. It interacts with the StbM and provides all FlexRay specific functions to the StbM.

Time Synchronization principles and common wording is described in [5] and [1].

7.2 Module Handling

This section contains description of auxiliary functionality of the Time Synchronization over FlexRay.

7.2.1 Initialization

The Time Synchronization over FlexRay is initialized via FrTSyn_Init. Except for FrTSyn_GetVersionInfo and FrTSyn_Init, the API functions of the Time Synchronization over FlexRay may only be called when the module has been properly initialized.

[SWS_FrTSyn_00003]

Upstream requirements: RS_TS_00003, RS_TS_00004

⌈A call to FrTSyn_Init initializes all internal variables and sets the Time Synchronization over FlexRay to the initialized state.
⌋

[SWS_FrTSyn_00006]

Upstream requirements: RS_TS_20041

⌈The Sequence Counter (SC) shall be initialized with 0.
⌋

7.2.2 FlexRay Interface

[SWS_FrTSyn_00078]

Upstream requirements: RS_TS_20040, RS_TS_20041

⌈The FrTSyn module shall call FrIf_GetGlobalTime only if FrIf_GetState returns FRIF_STATE_ONLINE. This is to ensure that FrIf_GetGlobalTime returns valid time information, i.e. that the FlexRay communication controller is synchronous to the FlexRay global time.⌋

7.2.3 Error Handling

[SWS_FrTSyn_00058]

Upstream requirements: RS_TS_20042, SRS_BSW_00323

⌈On errors and exceptions, the FrTSyn module shall not modify its current module state but shall simply report the error event.⌋

7.3 Message Format

SYNC messages may share the same FR PDU by using a multiplexed signal group. The multiplexer is located in byte 0, named Type.

For different Time Domains the same FR PDU may be used if Time Synchronization messages are sent by the same Time Master or Time Gateway.

For different Time Domains different FR PDUs shall be used if Time Synchronization messages are sent by different Time Masters or Time Gateways.

The usage of CRC is optional. To ensure a great variability between several time observing units, the configuration decides of how to handle CRC protected time synchronization messages if the receiver does not support the CRC calculation. Hence it might be possible, that a receiver is just using the given Time Base value, without evaluating the CRC.

SYNC messages can be ICV secured. This provides the integrity and authenticity protection of these messages.

The usage of a ICV is optional. To ensure a great variability between several time observing units, the configuration decides on how to handle ICV secured Time Synchronization messages if the receiver does not support the ICV calculation. Hence it might be possible, that a receiver is just using the given Time Base value without evaluating the ICV.

[SWS_FrTSyn_00007]

Upstream requirements: RS_TS_20043

⌈The byte order for time values inside Time Synchronization messages is "Big Endian".⌋

[SWS_FrTSyn_00009]

Upstream requirements: RS_TS_20043, RS_TS_20074

⌈If the message type is 0x10 or 0x20 (not `ICV` secured `SYNC` messages), the Payload Length shall be 16.⌋

The length of `ICV` secured messages depends on the length of the variable `FV` and `ICV` fields.

[SWS_FrTSyn_00136]

Upstream requirements: RS_TS_20043, RS_TS_20074

⌈If the message type is 0x50 or 0x60 (`ICV` secured `SYNC` messages), the Payload Length shall be 18 .. 254, depending on the configured `FV` length (`StbMFreshnessValueLength`) and the `ICV` length (`FrTSynIcvTxLength`).⌋

[SWS_FrTSyn_00110]

Upstream requirements: RS_TS_20074

⌈For `SYNC` messages, if the `FVL` field is 0, there shall be no `FV` field, i.e., the `ICV` field shall follow immediately, starting with byte 18.⌋

[SWS_FrTSyn_00111]

Upstream requirements: RS_TS_20074

⌈For `SYNC` messages, if the `ICVL` field is 0, there shall be no `ICV` field, i.e., the `SYNC` message shall end with byte 17.⌋

[SWS_FrTSyn_00010]

Upstream requirements: RS_TS_20043, RS_TS_20045

⌈Time Synchronization messages contain User Data according to the given message format.⌋

[SWS_FrTSyn_00011]

Upstream requirements: RS_TS_20045

⌈User Data shall be read consistently from the incoming Time Synchronization messages.⌋

[SWS_FrTSyn_00012]

Upstream requirements: RS_TS_20045

⌈User Data shall be written consistently to outgoing Time Synchronization messages.

If the number of User Data Fields in a Time Synchronization message is greater than the number of User Data Bytes provided by the StbM, the remaining User Data Fields shall be set to 0 (default value).⌋

[SWS_FrTSyn_00013]

Upstream requirements: RS_TS_20045

⌈User Data shall be mapped to the `StbM_UserDataType`, where the byte number given in the message and by the `StbM_UserDataType` shall match (User Byte 0 mapped to `StbM_UserDataType.userByte0` etc.). `StbM_UserDataType.userDataLength` shall be set to the Time Synchronization message type specific number of User Bytes.⌋

7.3.1 SYNC message

The message layout of the SYNC messages is defined by the following requirements:

- [SWS_FrTSyn_00014]
- [SWS_FrTSyn_00015]
- [SWS_FrTSyn_00106]
- [SWS_FrTSyn_00107]

depending on whether the payload is `CRC` protected and/or `ICV` secured or not.

[SWS_FrTSyn_00014] SYNC message format - not CRC protected, not ICV secured

Upstream requirements: RS_TS_20041, RS_TS_20043

⌈

Byte	Bit Position	Field Name	Field Value Range	Description
0		`Type`	0x10	Message Type
1		User Byte 2	default: 0	
2	7..4	`D`	0..15	Time Domain Id
	3..0	`SC`	0..15	Sequence Counter
3	7..2	`FCNT`	0..63	FlexRay Cycle Counter
	1	`SGW`	0 = `SyncToGTM` 1 = `SyncToSubDomain`	
	0	reserved	0	
4		User Byte 0	default: 0	
5		User Byte 1	default: 0	

6..11		SyncTimeSec		48 bit time value in seconds
12..15		SyncTimeNSec		32 bit time value in nanoseconds

⌋

[SWS_FrTSyn_00015] SYNC message format - CRC protected, not ICV secured

Upstream requirements: RS_TS_20041, RS_TS_20042, RS_TS_20043

⌈

Byte	Bit Position	Field Name	Field Value Range	Description
0		Type	0x20	Message Type
1		CRC	0..255	Checksum
2	7..4	D	0..15	Time Domain Id
	3..0	SC	0..15	Sequence Counter
3	7..2	FCNT	0..63	FlexRay Cycle Counter
	1	SGW	0 = SyncToGTM 1 = SyncToSubDomain	
	0	reserved	0	
4		User Byte 0	default: 0	
5		User Byte 1	default: 0	
6..11		SyncTimeSec		48 bit time value in seconds
12..15		SyncTimeNSec		32 bit time value in nanoseconds

⌋

[SWS_FrTSyn_00106] SYNC message format - not CRC protected, ICV secured

Upstream requirements: RS_TS_20041, RS_TS_20043, RS_TS_20074

⌈

Byte	Bit Position	Field Name	Field Value Range	Description
0		Type	0x50	Message Type
1		User Byte 2	default: 0	
2	7..4	D	0..15	Time Domain Id
	3..0	SC	0..15	Sequence Counter
3	7..2	FCNT	0..63	FlexRay Cycle Counter
	1	SGW	0 = SyncToGTM 1 = SyncToSubDomain	
	0	reserved	0	
4		User Byte 0	default: 0	
5		User Byte 1	default: 0	
6..11		SyncTimeSec		48 bit time value in seconds
12..15		SyncTimeNSec		32 bit time value in nanoseconds
16	7	reserved	0	
	6..0	FVL	0..64	FV Length in bits
17		ICVL	0..236	ICV Length in bytes
18		FV		FV
18+FVL in bytes		ICV		ICV

⌋

[SWS_FrTSyn_00107] SYNC message format - CRC protected, ICV secured

Upstream requirements: RS_TS_20041, RS_TS_20042, RS_TS_20043, RS_TS_20074

⌈

Byte	Bit Position	Field Name	Field Value Range	Description
0		Type	0x60	Message Type
1		CRC	0..255	Checksum
2	7..4	D	0..15	Time Domain Id
	3..0	SC	0..15	Sequence Counter
3	7..2	FCNT	0..63	FlexRay Cycle Counter
	1	SGW	0 = SyncToGTM 1 = SyncToSubDomain	
	0	reserved	0	
4		User Byte 0	default: 0	
5		User Byte 1	default: 0	
6..11		SyncTimeSec		48 bit time value in seconds
12..15		SyncTimeNSec		32 bit time value in nanoseconds
16	7	reserved	0	
	6..0	FVL	0..64	FV Length in bits
17		ICVL	0..236	ICV Length in bytes
18		FV		FV
18+FVL in bytes		ICV		ICV

⌋

7.4 Acting as Time Master

A Time Master is an entity which is the master for a certain Time Base and which propagates this Time Base to a set of Time Slaves within a certain segment of a communication network, being a source for this Time Base.

If a Time Master is also the owner of the Global Time Base, the Time Base from which all further Time Bases are derived from, then it is the Global Time Master. A Time Gateway typically consists of one Time Master port which is connected to one or more Time Slaves. When mapping time entities to real ECUs it has to be noted, that an ECU could be Time Master (or even Global Time Master) for one Time Base and Time Slave for another Time Base.

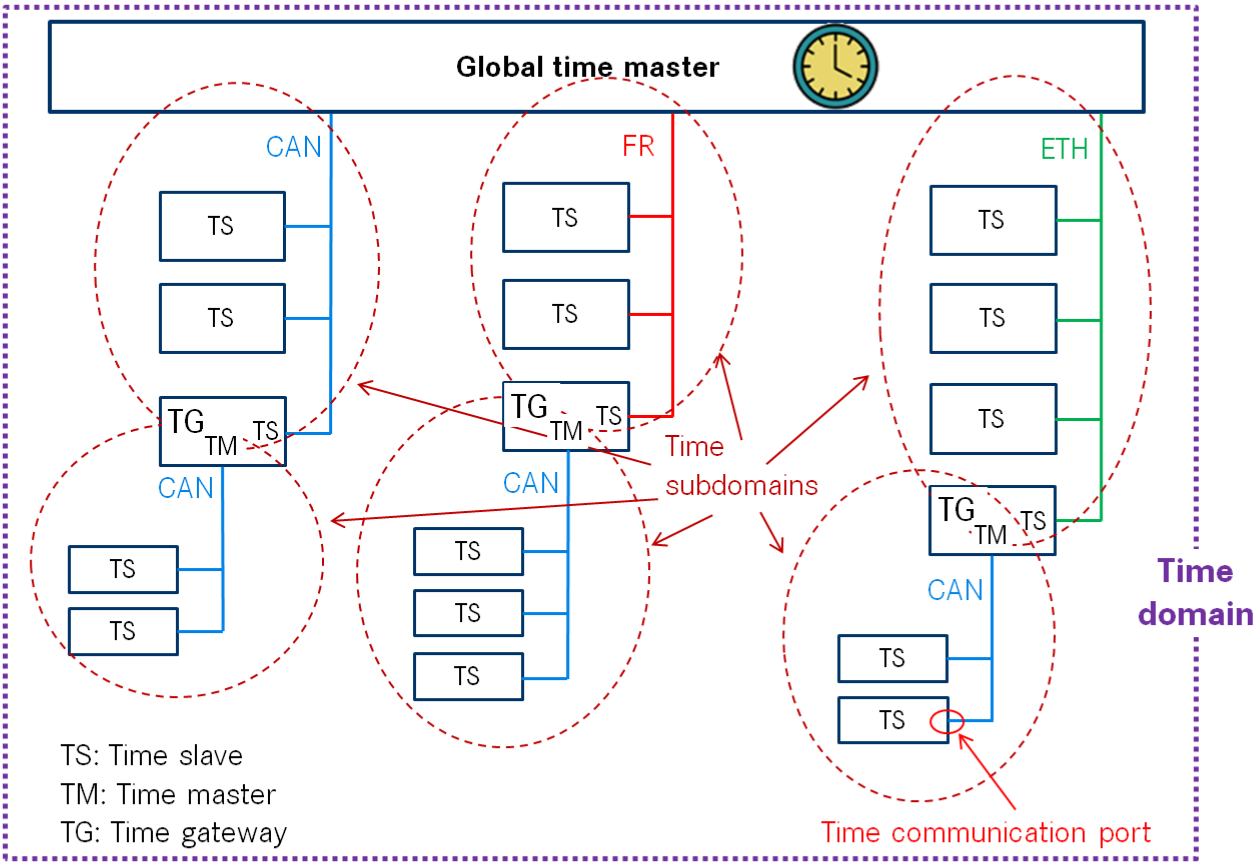


Figure 7.1: Terminology Example

If the FrTSyn is configured as a Time Master for Time Domain, the FrTSyn module checks on each `FrTSyn_MainFunction` call the necessity for a Timesync message transmission for that Time Domain.

Figure 7.2 illustrates how FrTSyn determines if (immediate and cyclic) message transmission of a Timesync message is required.

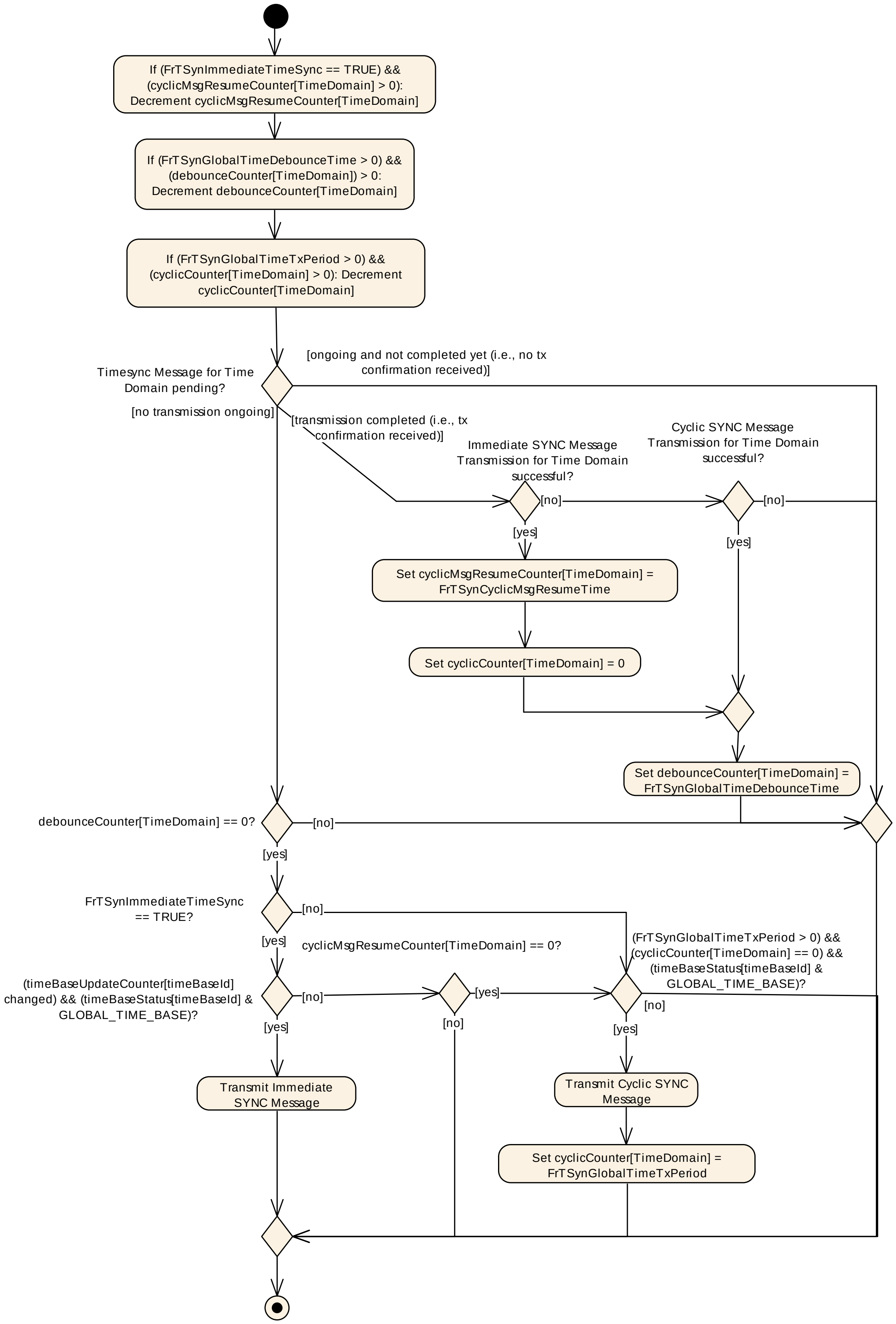

If (FrTSynImmediateTimeSync == TRUE) && (cyclicMsgResumeCounter[TimeDomain] > 0): Decrement cyclicMsgResumeCounter[TimeDomain]
If (FrTSynGlobalTimeDebounceTime > 0) && (debounceCounter[TimeDomain]) > 0: Decrement debounceCounter[TimeDomain]
If (FrTSynGlobalTimeTxPeriod > 0) && (cyclicCounter[TimeDomain] > 0): Decrement cyclicCounter[TimeDomain]
Timesync Message for Time Domain pending?
[ongoing and not completed yet (i.e., no tx confirmation received)]
[no transmission ongoing]
[transmission completed (i.e., tx confirmation received)]
Immediate SYNC Message Transmission for Time Domain successful?
Cyclic SYNC Message Transmission for Time Domain successful?
[no]
[no]
[yes]
[yes]
Set cyclicMsgResumeCounter[TimeDomain] = FrTSynCyclicMsgResumeTime
Set cyclicCounter[TimeDomain] = 0
Set debounceCounter[TimeDomain] = FrTSynGlobalTimeDebounceTime
debounceCounter[TimeDomain] == 0?
[no]
[yes]
FrTSynImmediateTimeSync == TRUE?
[no]
[yes]
cyclicMsgResumeCounter[TimeDomain] == 0?
(FrTSynGlobalTimeTxPeriod > 0) && (cyclicCounter[TimeDomain] == 0) && (timeBaseStatus[timeBaseId] & GLOBAL_TIME_BASE)?
(timeBaseUpdateCounter[timeBaseId] changed) && (timeBaseStatus[timeBaseId] & GLOBAL_TIME_BASE)?
[no]
[yes]
[no]
[no]
[yes]
[yes]
Transmit Immediate SYNC Message
Transmit Cyclic SYNC Message
Set cyclicCounter[TimeDomain] = FrTSynGlobalTimeTxPeriod

Figure 7.2: Timesync Message Transmission

[SWS_FrTSyn_00173] SYNC message transmission by LSduR_FrTSynTransmit

Upstream requirements: RS_TS_20039

⌈A Time Master shall transmit SYNC messages by calling LSduR_FrTSynTransmit with the PduId derived via FrTSynGlobalTimePduRef of the corresponding Time Domain.⌋

7.4.1 SYNC message processing

[SWS_FrTSyn_00018]

Upstream requirements: RS_TS_20043

⌈A Time Synchronization message sequence consists of a SYNC message per Time Domain.⌋

Note: Refer to figure 9.1 for the sequence diagram of a Time Master.

[SWS_FrTSyn_00019]

Upstream requirements: RS_TS_20039, RS_TS_20043

⌈For each configured Time Master (refer to FrTSynGlobalTimeMaster) if

- the GLOBAL_TIME_BASE bit within the timeBaseStatus is set
- and FrTSynGlobalTimeTxPeriod is unequal to 0
- and the associated cyclicMsgResumeCounter is equal or less than 0,

then the FrTSyn module shall periodically transmit SYNC messages with the cycle FrTSynGlobalTimeTxPeriod including the time value, which will be valid at the start of the next FlexRay cycle 0 and User Data.

The cyclic transmission shall be started in the earliest possible FrTSyn_MainFunction call once the requirements above are fulfilled.
⌋

Note: "earliest possible" means:

- In the next FrTSyn_MainFunction, because GLOBAL_TIME_BASE is set outside the FrTSyn_MainFunction.
- In the current FrTSyn_MainFunction, when switching from immediate to cyclic transmission (because this decision is made inside the FrTSyn_MainFunction). For details on immediate transmission refer to chapter 7.4.4.

[SWS_FrTSyn_00021]

Upstream requirements: RS_TS_20041, RS_TS_20043

⌈If

- FrTSynGlobalTimeTxIcvSecured is set to ICV_NOT_SUPPORTED
- FrTSynGlobalTimeTxCrcSecured is set to CRC_NOT_SUPPORTED

then the message type of the SYNC message shall be 0x10 (i.e., SYNC message not CRC protected and not ICV authenticated).

If

- FrTSynGlobalTimeTxIcvSecured is set to ICV_NOT_SUPPORTED
- FrTSynGlobalTimeTxCrcSecured is set to CRC_SUPPORTED

then the message type of the SYNC message shall be 0x20 (i.e., SYNC message CRC protected and not ICV authenticated).⌋

[SWS_FrTSyn_00112]

Upstream requirements: RS_TS_20041, RS_TS_20043, RS_TS_20074

⌈If

- FrTSynGlobalTimeTxIcvSecured is set to ICV_SUPPORTED
- FrTSynGlobalTimeTxCrcSecured is set to CRC_NOT_SUPPORTED

then the message type of the SYNC message shall be 0x50 (i.e., SYNC message not CRC protected, but ICV authenticated).

If

- FrTSynGlobalTimeTxIcvSecured is set to ICV_SUPPORTED
- FrTSynGlobalTimeTxCrcSecured is set to CRC_SUPPORTED

then the message type of the SYNC message shall be 0x60 (i.e., SYNC message CRC protected and ICV authenticated).⌋

7.4.2 Transmission mode

[SWS_FrTSyn_00026]

Upstream requirements: RS_TS_20039, RS_TS_20043

⌈If FrTSyn_SetTransmissionMode(Controller, Mode) is called and parameter Mode equals FRTSYN_TX_OFF, all transmit requests from FrTSyn shall be omitted on this FlexRay channel.⌋

[SWS_FrTSyn_00027]

Upstream requirements: RS_TS_20039, RS_TS_20043

⌈If FrTSyn_SetTransmissionMode(Controller, Mode) is called and parameter Mode equals FRTSYN_TX_ON, all transmit requests from FrTSyn on this FlexRay channel shall be able to be transmitted.⌋

7.4.3 Debounce Time

The FrTSyn debounces FlexRay Tx PDUs of a Time Master to avoid bursts of Timesync messages on the bus (e.g. if immediate transmission is enabled).

For each Tx PDU the FrTSyn maintains a debounce counter debounceCounter. On each transmission of a Timesync message the debounceCounter is (re-)loaded by the configured debounce time FrTSynGlobalTimeDebounceTime. The debounceCounter is decremented in each FrTSyn main cycle. Transmission of the same PDU can only be triggered, if the debounceCounter has reached the value 0. Refer also to the overall sequence for the Timesync message transmission in Figure 7.2.

The FrTSyn does not support sharing of PDUs across domains and busses, i.e. same PDU ID shall not be used for different time domains.

[SWS_FrTSyn_00085]

Upstream requirements: RS_TS_20039

⌈If for a Time Domain

- FrTSynGlobalTimeDebounceTime is greater than 0
- and the corresponding Timesync PDU has been successfully sent,

then the FrTSyn shall set the PDU specific debounceCounter to FrTSynGlobalTimeDebounceTime.⌋

Note: A Timesync PDU is considered to be successfully sent, if

- for a Tx Pdu, which is configured for immediate transmission, LSduR_FrTSynTransmit returns E_OK,
- and for a Tx Pdu, which is configured for decoupled transmission, FrTSyn_TriggerTransmit returns E_OK.

[SWS_FrTSyn_00169]

Upstream requirements: RS_TS_20039

⌈If for a Time Domain

- FrTSynGlobalTimeDebounceTime is greater than 0

- and the debounceCounter for the corresponding Timesync PDU is greater than 0,

then the FrTSyn shall decrement the debounceCounter value by FrTSynMainFunctionPeriod on each invocation of FrTSyn_MainFunction.⌋

[SWS_FrTSyn_00086]

Upstream requirements: RS_TS_20039

⌈If for a Time Domain

- FrTSynGlobalTimeDebounceTime is greater than 0
- and the debounceCounter for the corresponding Timesync PDU is greater than 0
- and a transmission of a TimeSync message is requested,

then FrTSyn shall defer the actual transmission of the Timesync message until debounceCounter is equal or less than 0⌋

Rationale: While debouncing a new transmission request should not get lost.

[SWS_FrTSyn_00170]

Upstream requirements: RS_TS_20039

⌈If for a Time Domain

- FrTSynGlobalTimeDebounceTime is greater than 0
- and a deferred SYNC message transmission request is pending
- and a new immediate or cyclic transmission of a SYNC message is requested,

then the FrTSyn shall discard the pending request for that Time Domain.⌋

Rationale: While debouncing there is no queuing of multiple transmission requests. The latest request is the best one.

7.4.4 Immediate Time Synchronization

In addition to the cyclic Timesync message transmission an immediate message transmission might be required. Refer also to the overall sequence for the Timesync message transmission in Figure 7.2.

[SWS_FrTSyn_00087]

Upstream requirements: RS_TS_20039

⌈If FrTSynImmediateTimeSync is set to TRUE for a Time Base, FrTSyn shall check on each FrTSyn_MainFunction call by calling StbM_GetTimeBaseUpdateCounter, if the timeBaseUpdateCounter of the corresponding Time Base has changed.⌋

[SWS_FrTSyn_00088]

Upstream requirements: RS_TS_20039

⌈If

- FrTSynImmediateTimeSync is set to TRUE
- and the timeBaseUpdateCounter of a Time Base has changed
- and the GLOBAL_TIME_BASE bit of the timeBaseStatus is set,

then FrTSyn shall trigger an immediate transmission of Time Synchronization messages for the corresponding Time Base.⌋

Note: timeBaseStatus can be obtained by StbM_GetTimeBaseStatus or StbM_GetCurrentTime.

Note: The debounceCounter as described in 7.4.3 shall always be considered.

In addition to the actual trigger condition for an immediate transmission (refer to [SWS_FrTSyn_00088] above) the parameter FrTSynCyclicMsgResumeTime needs to be considered for immediate transmission. Refer also to the trigger condition for cyclic Timesync message transmissions (refer to [SWS_FrTSyn_00019]) .

Two main scenarios are relevant for configuration of FrTSynCyclicMsgResumeTime

- With FrTSynCyclicMsgResumeTime and FrTSynGlobalTimeTxPeriod both being configured as zero, a single shot mode is achieved that is solely triggered by the change of the timeBaseUpdateCounter.
- With FrTSynCyclicMsgResumeTime greater than FrTSynGlobalTimeTxPeriod a hold-over scenario in a Time Gateway can be configured:
  - While Timesync messages are received from the Time Master side, the Timesync messages on the sub-busses are only triggered by immediate transmission (cyclic transmission is suspended while cyclicMsgResumeCounter is running).
  - If no Timesync messages from the Time Master side are received anymore and a timeout is detected, cyclic transmission takes over (cyclic transmission no longer suspended because cyclicMsgResumeCounter has elapsed)
  - If reception of Timesync messages from the Time Master side resumes, the Timesync messages on the sub-busses are again triggered by im-

mediate transmission (cyclic transmission is again suspended by running cyclicMsgResumeCounter)

[SWS_FrTSyn_00090]

Upstream requirements: RS_TS_20039

⌈If for a Time Domain

- FrTSynImmediateTimeSync is set to TRUE
- and FrTSynCyclicMsgResumeTime is greater than 0
- and an immediate SYNC message (refer to [SWS_FrTSyn_00088]) is sent,

then the FrTSyn shall set the counter cyclicMsgResumeCounter to FrTSynCyclicMsgResumeTime for the corresponding Time Domain.⌋

[SWS_FrTSyn_00093]

Upstream requirements: RS_TS_20039

⌈If for a Time Domain the cyclicMsgResumeCounter is greater than 0, then the FrTSyn shall discard cyclic Timesync message transmission requests for that Time Domain.⌋

[SWS_FrTSyn_00161]

Upstream requirements: RS_TS_20039

⌈If for a Time Domain the cyclicMsgResumeCounter is greater than 0, then the FrTSyn shall decrement the cyclicMsgResumeCounter of the corresponding Time Domain by FrTSynMainFunctionPeriod on each invocation of FrTSyn_MainFunction.⌋

[SWS_FrTSyn_00091]

Upstream requirements: RS_TS_20039

⌈If the cyclicMsgResumeCounter is decremented to 0 or below, then the FrTSyn shall resume cyclic Timesync message transmission within the FrTSyn_MainFunction call by requesting a SYNC message transmission.⌋

Note: [SWS_FrTSyn_00091] is to ensure, that the first cyclic transmission is requested in the same main function call in which also cyclicMsgResumeCounter reaches 0 (refer to term "earliest possible" main function in [SWS_FrTSyn_00019]). If the message is actually transmitted also depends on the debounceCounter.

7.4.5 Calculation and Assembling of Time Synchronization Messages

This chapter describes the workflow, how the items of a Time Synchronization message will be calculated (1st step) and how the message will be assembled (2nd step).

7.4.5.1 Global Time Calculation

[SWS_FrTSyn_00028]

Upstream requirements: RS_TS_20043

⌈The transmitter of a Synchronized Time Base (Time Master) shall perform the following steps to distribute the Synchronized Time Base:

1. Retrieve current Synchronized Time Base's Time Tuple as [T_{SYNC};$T0_{VLT}$] via StbM_GetCurrentTime
2. Protect the following two steps against interruptions:
   (a) Get currentCycle and currentMacroticks via FrIf_GetGlobalTime
   (b) Retrieve current Virtual Local Time value as $T1_{VLT}$ via StbM_GetCurrentVirtualLocalTime
3. Calculate the (future) time value of the Time Base at the start of the next FlexRay cycle by T0 = T_{SYNC} + ($T1_{VLT}$ - $T0_{VLT}$) + (64 - currentCycle) * CycleLength - (CycleLength / MacroticksPerCycle) * currentMacroticks
4. Calculate SyncTimeSec (second portion of T0) and SyncTimeNSec (nanosecond portion of T0)

⌋

Note: Refer to figure 9.1 for the Time Master sequence of actions.

Note: It is inevitable to retrieve currentCycle and currentMacroticks of the FlexRay time and $T1_{VLT}$ of the Virtual Local Time in an atomic way, otherwise any delay between them will worsen the precision by the amount of the delay.

Note: If the calculation is done on an integer basis the rounding error of the term (CycleLength / MacroticksPerCycle) needs to be minimized. This can be done in multiple ways, e.g., by calculating

- ((CycleLength * currentMacroticks) / MacroticksPerCycle) on 64 bit architectures or
- ((((CycleLength * 256) / MacroticksPerCycle) * currentMacroticks) / 256) on 32 bit architectures (multiplication by 256 is acceptable for any possible FlexRay parameter configuration)

CycleLength and MacroticksPerCycle are retrieved via FrIf_GetCycleLength and FrIf_GetMacroticksPerCycle.

7.4.5.2 SGW Calculation

[SWS_FrTSyn_00020]

Upstream requirements: RS_TS_20043

⌈The SGW value (Time Gateway synchronization status) shall be retrieved from the Time Base synchronization status. If the SYNC_TO_GATEWAY bit within timeBaseStatus is not set the SGW value shall be SyncToGTM. Otherwise the SGW value shall be set to SyncToSubDomain.⌋

7.4.5.3 Sequence Counter Calculation

[SWS_FrTSyn_00030]

Upstream requirements: RS_TS_20041, RS_TS_20043

⌈A Sequence Counter (SC) of 4 bit is representing numbers from 0 to 15 per Time Domain. The Sequence Counter shall be independent between SYNC messages and shall be incremented by 1 on every transmission request of a SYNC message. It shall wrap around at 15 to 0 again.⌋

7.4.5.4 CRC Calculation

[SWS_FrTSyn_00031]

Upstream requirements: RS_TS_20041, RS_TS_20043

⌈The function Crc_CalculateCRC8H2F as defined in [6] shall be used to calculate the CRC, if configured. ⌋

[SWS_FrTSyn_00035]

Upstream requirements: RS_TS_20041, RS_TS_20043

⌈The DataID shall be calculated as DataID = DataIDList[SC], where DataIDList is given by configuration for each message type (refer to FrTSynGlobalTimeSyncDataIDList).⌋

Note: A specific DataID out of a predefined DataIDList ensures the identification of data elements of Time Synchronization messages.

[SWS_FrTSyn_00036]

Upstream requirements: RS_TS_20041, RS_TS_20043

⌈The CRC shall be calculated over Time Synchronization message byte 2 to byte 15 and DataID, where byte 2 is applied first, followed by the other bytes in ascending order, and DataID last. ⌋

7.4.5.5 ICV Calculation

Refer to the chapter 7.3.13 in StbM [5] for the configuration details of FV referenced in each Time Domain.

[SWS_FrTSyn_00114]

Upstream requirements: RS_TS_20074

⌈When:

- the FV is referenced (refer FrTSynIcvGenerationFvIdRef),
- and the configured truncated FV length (StbMFreshnessValueTruncLength) is equal to the FV length (StbMFreshnessValueLength),

the Time Master shall call StbM_GetTxFreshness to obtain the FV by using the StbMFreshnessValueId (referenced via the FrTSynIcvGenerationFvIdRef parameter).⌋

[SWS_FrTSyn_00115]

Upstream requirements: RS_TS_20074

⌈When:

- the FV is referenced (refer FrTSynIcvGenerationFvIdRef),
- and the configured truncated FV length (StbMFreshnessValueTruncLength) < FV length (StbMFreshnessValueLength),

the Time Master shall call StbM_GetTxFreshnessTruncData to obtain the FV and the truncated FV by using the StbMFreshnessValueId (referenced via the FrTSynIcvGenerationFvIdRef parameter).⌋

Note: Having the configured truncated FV length (StbMFreshnessValueTruncLength) > FV length (StbMFreshnessValueLength) in StbM is not a valid configuration.

[SWS_FrTSyn_00116]

Upstream requirements: RS_TS_20074

⌈When the FV is not referenced (refer FrTSynIcvGenerationFvIdRef), the Time Master shall not include the FV in the ICV generation and neither in the SYNC message.⌋

[SWS_FrTSyn_00117]

Upstream requirements: RS_TS_20074

⌈If StbM_GetTxFreshness returns E_OK, the Time Master shall construct the SYNC message with FV and use the full FV in the ICV generation.⌋

[SWS_FrTSyn_00118]

Upstream requirements: RS_TS_20074

⌈If StbM_GetTxFreshnessTruncData returns E_OK, the Time Master shall construct the SYNC message with truncated FV and use the full FV in the ICV generation.⌋

[SWS_FrTSyn_00119]

Upstream requirements: RS_TS_20074

⌈When StbM_GetTxFreshness or StbM_GetTxFreshnessTruncData return a non-recoverable error code (i.e, E_NOT_OK), then the Time Master shall:

- stop the ICV generation and set the FVL and the ICVL to 0 in the SYNC message,
- call Det_ReportRuntimeError with parameter ErrorId set to FRTSYN_E_FRESHNESSFAILURE (refer [SWS_FrTSyn_91000]),
- call IdsM_SetSecurityEventWithContextData with parameter EventId set to SEV_TSYN_FR_FRESHNESS_NOT_AVAILABLE.

⌋

Refer to the chapter 10.2.5 in [7] for the configuration details of CSM job used for ICV generation.

[SWS_FrTSyn_00120]

Upstream requirements: RS_TS_20074

⌈If FrTSynIcvGenerationBase for the Time Domain is configured to ICV_MAC, the Time Master shall call Csm_MacGenerate to generate the ICV value.⌋

[SWS_FrTSyn_00121]

Upstream requirements: RS_TS_20074

⌈If FrTSynIcvGenerationBase for the Time Domain is configured to ICV_SIGNATURE, the Time Master shall call Csm_SignatureGenerate to generate the ICV value.⌋

Note: The mode parameter is intentionally left open for the implementer to choose (i.e. CRYPTO_OPERATIONMODE_SINGLECALL would possibly be the best option since it does not require further calls to CSM).

The CSM job used to generate the ICV can be configured to synchronous or asynchronous behavior. The ICV generation timeout observation should be disabled, when the CSM job used to generate the ICV, is configured in synchronous behavior.

[SWS_FrTSyn_00122]

Upstream requirements: RS_TS_20074

⌈If FrTSynIcvGenerationTimeout is set to 0, the Time Master shall not do ICV generation timeout monitoring.⌋

[SWS_FrTSyn_00123]

Upstream requirements: RS_TS_20074

⌈If:

- FrTSynIcvGenerationTimeout is set to any value greater than 0,
- and Csm_MacGenerate or Csm_SignatureGenerate returns E_OK,

the Time Master shall start the FrTSynIcvGenerationTimeout.⌋

[SWS_FrTSyn_00124]

Upstream requirements: RS_TS_20074

⌈If:

- FrTSynIcvGenerationTimeout is set to any value greater than 0,
- the callback FrTSyn_IcvGenerationIndication is called,

the Time Master shall stop the FrTSynIcvGenerationTimeout.⌋

[SWS_FrTSyn_00125]

Upstream requirements: RS_TS_20074

⌈If one of the following conditions is true:

- the authentication build counter has reached the configuration value FrTSynTxAuthenticationBuildAttempts,

- the verification of the ICV has returned a non-recoverable error such as returning E_NOT_OK or KEY_FAILURE,
- FrTSynIcvGenerationTimeout expires before the notification of the FrTSyn_IcvGenerationIndication callback

then the Time Master shall

- stop the ICV generation and set the FVL and the ICVL to 0 in the SYNC message,
- and call IdsM_SetSecurityEventWithContextData with parameter EventId set to SEV_TSYN_FR_ICV_GENERATION_FAILED.

⌋

[SWS_FrTSyn_00126]

Upstream requirements: RS_TS_20074

⌈With the notification of the FrTSyn_IcvGenerationIndication callback, the Time Master shall add the generated ICV to the SYNC message and transmit it.⌋

[SWS_FrTSyn_00127]

Upstream requirements: RS_TS_20074

⌈When the FV is referenced (refer FrTSynIcvGenerationFvIdRef), then the Time Master shall notify the successful transmission of the SYNC message to FVM by calling StbM_SPduTxConfirmation.⌋

[SWS_FrTSyn_00153]

Status: DRAFT
Upstream requirements: RS_TS_20074

⌈For every transmission of messages of type 0x50 and 0x60, the Time Master shall shall maintain an authentication build counter (refer to FrTSynTxAuthenticationBuildAttempts).⌋

[SWS_FrTSyn_00154]

Status: DRAFT
Upstream requirements: RS_TS_20074

⌈Upon the initial processing of messages of type 0x50 and 0x60 (i.e., upon the first attempt of a freshness value and ICV generation for each received message) the Time Master shall set the authentication build counter to 0.⌋

[SWS_FrTSyn_00155]

Status: DRAFT
Upstream requirements: RS_TS_20074

⌈If StbM_GetTxFreshness or StbM_GetTxFreshnessTruncData return a recoverable error code (e.g., STBM_E_BUSY), then the Time Master shall increment the authentication build counter.⌋

[SWS_FrTSyn_00156]

Status: DRAFT
Upstream requirements: RS_TS_20074

⌈If Csm_MacGenerate or Csm_SignatureGenerate return a recoverable error code (e.g., E_BUSY, QUEUE_FULL), then the Time Master shall increment the authentication build counter.⌋

[SWS_FrTSyn_00171]

Status: DRAFT
Upstream requirements: RS_TS_20074

⌈If

- the generation of the authenticated message has failed
- and the authentication build counter has not yet reached the configuration value FrTSynTxAuthenticationBuildAttempts,

then the Time Master shall retry the freshness attempt and the ICV calculation in the next call of FrTSyn_MainFunction.⌋

7.4.5.6 Message Assembling

[SWS_FrTSyn_00037]

Upstream requirements: RS_TS_20043, RS_TS_20074

⌈For each transmission of a Time Synchronization message the FrTSyn module shall assemble the message as follows:

- Calculate SC
- Copy currentCycle (refer to [SWS_FrTSyn_00028]) to FCNT (for SYNC message)
- Calculate SGW
- Copy all data to the appropriate position within the related message
- Calculate CRC (configuration dependent)

- Fetch the FV (configuration dependent) and append the FVL, ICVL and FV in the appropriate position within the related message
- Calculate the ICV (configuration dependent) and append it in the appropriate position within the related message

⌋

7.5 Acting as Time Slave

A Time Slave is an entity, which is the recipient for a certain Time Base within a certain segment of a communication network, being a consumer for this Time Base.

7.5.1 SYNC message processing

[SWS_FrTSyn_00038]

Upstream requirements: RS_TS_20042, RS_TS_20043

⌈The FrTSyn shall only accept a SYNC message with Type equal to 0x20 / 0x60 and a correct CRC value if FrTSynRxCrcValidated is configured to CRC_VALIDATED.
⌋

[SWS_FrTSyn_00039]

Upstream requirements: RS_TS_20043

⌈The FrTSyn shall only accept a SYNC message with Type equal to 0x10 / 0x50 if FrTSynRxCrcValidated is configured to CRC_NOT_VALIDATED.⌋

[SWS_FrTSyn_00040]

Upstream requirements: RS_TS_20043

⌈The FrTSyn shall only accept a SYNC message with Type equal to 0x10 / 0x50 or 0x20 / 0x60 if FrTSynRxCrcValidated is configured to CRC_IGNORED.⌋

[SWS_FrTSyn_00081]

Upstream requirements: RS_TS_20042, RS_TS_20043

⌈The FrTSyn shall only accept a SYNC message with Type equal to 0x10 / 0x50 or a SYNC message with Type equal to 0x20 / 0x60 and a correct CRC value if FrTSynRxCrcValidated is configured to CRC_OPTIONAL.
⌋

[SWS_FrTSyn_00128]

Upstream requirements: RS_TS_20074

⌈If FrTSynRxIcvVerificationType is configured to ICV_VERIFIED, FrTSyn shall perform ICV verification for SYNC messages with ICV value (Message type: 0x50, 0x60).

The FrTSyn shall consider ICV verification as failed for SYNC messages without ICV (Message type: 0x10, 0x20).⌋

[SWS_FrTSyn_00129]

Upstream requirements: RS_TS_20074

⌈If FrTSynRxIcvVerificationType is configured to ICV_NOT_VERIFIED, the FrTSyn shall not perform the ICV verification and the SYNC messages shall not contain an ICV value (Message type: 0x10, 0x20).

The FrTSyn shall consider ICV verification as failed for SYNC messages with ICV (Message type: 0x50, 0x60).⌋

[SWS_FrTSyn_00130]

Upstream requirements: RS_TS_20074

⌈If FrTSynRxIcvVerificationType is configured to ICV_IGNORED, FrTSyn shall not perform the ICV verification.

The FrTSyn shall ignore the ICV in SYNC messages with ICV (Message type: 0x50, 0x60).⌋

[SWS_FrTSyn_00131]

Upstream requirements: RS_TS_20074

⌈If FrTSynRxIcvVerificationType is configured to ICV_OPTIONAL, the FrTSyn shall perform ICV verification for SYNC messages with ICV (Message type: 0x50, 0x60).

The FrTSyn shall not perform ICV verification for SYNC messages without ICV (Message type: 0x10, 0x20).⌋

[SWS_FrTSyn_00041]

Upstream requirements: RS_TS_20040, RS_TS_20042, RS_TS_20043

⌈For valid SYNC messages a new Time Tuple, consisting of the Global Time value and the associated value of the Virtual Local Time, shall be calculated and forwarded to the StbM module via StbM_BusSetGlobalTime. ⌋

7.5.2 Validation and Disassembling of Time Synchronization Messages

This chapter describes the workflow how the items of a Time Synchronization message will be validated (1st step) and how the message will be disassembled (2nd step).

7.5.2.1 Global Time Calculation

[SWS_FrTSyn_00046]

Upstream requirements: RS_TS_20043

⌈The receiver of a Synchronized Time Base shall perform the following steps to assemble the Synchronized Time Base:

1. On SYNC message RX indication (or in the subsequent MainFunction call) store received time value T0 (SyncTimeSec, SyncTimeNSec)
2. Protect the following two steps against interruptions:
   (a) Get currentCycle and currentMacroticks via FrIf_GetGlobalTime
   (b) Retrieve current Virtual Local Time value as $T1_{VLT}$ via StbM_GetCurrentVirtualLocalTime
3. Calculate Time Tuple [T1; $T1_{VLT}$] to update the Time Slave's local instance of the Time Base:
   (a) T1 = T0 + (CycleLength * currentCycle) + ((CycleLength / MacrotickPerCycle) * currentMacroticks)
   (b) If currentCycle is greater or equal than the retrieved FCNT value from the transmitter (Time Master), then the calculated value T1 shall be subtracted by 64 times the FlexRay cycle duration: T1 = T1 - (CycleLength * 64)

⌋

Note: Refer to figure 9.2 for the Time Slave sequence of actions.

Note: It is inevitable to retrieve currentCycle and currentMacroticks of the FlexRay time and $T1_{VLT}$ of the Virtual Local Time atomic, otherwise any delay between them will worsen the precision by the amount of the delay.

Note: In order to minimize rounding errors for the term (CycleLength / MacrotickPerCycle) in case of integer calculation refer to note below [SWS_FrTSyn_00028].

7.5.2.2 SGW Calculation

[SWS_FrTSyn_00094]

Upstream requirements: RS_TS_20040, RS_TS_20042

⌈If the SGW value of a SYNC message is set to SyncToSubDomain, the SYNC_TO_GATEWAY bit within timeBaseStatus shall be set to TRUE. Otherwise, it shall be set to FALSE.⌋

7.5.2.3 Sequence Counter Validation

Figure 7.3 illustrates the Sequence Counter validation of a Time Slave for SYNC messages.

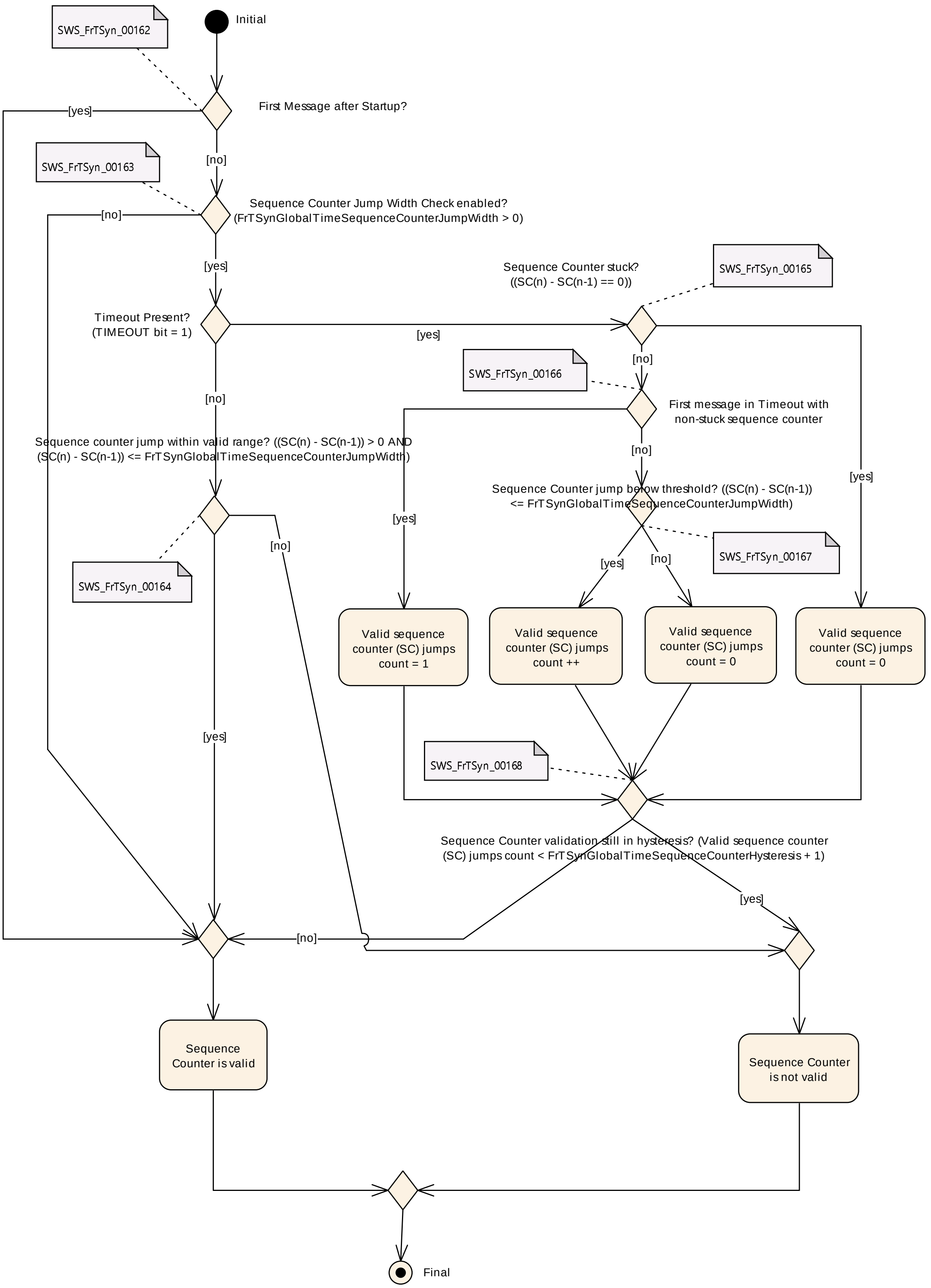

SWS_FrTSyn_00162
Initial
First Message after Startup?
[yes]
[no]
SWS_FrTSyn_00163
Sequence Counter Jump Width Check enabled?
(FrTSynGlobalTimeSequenceCounterJumpWidth > 0)
[no]
[yes]
Sequence Counter stuck?
((SC(n) - SC(n-1) == 0))
SWS_FrTSyn_00165
Timeout Present?
(TIMEOUT bit = 1)
[yes]
[no]
SWS_FrTSyn_00166
First message in Timeout with
non-stuck sequence counter
[no]
Sequence counter jump within valid range? ((SC(n) - SC(n-1)) > 0 AND
(SC(n) - SC(n-1)) <= FrTSynGlobalTimeSequenceCounterJumpWidth)
[no]
[yes]
Sequence Counter jump below threshold? ((SC(n) - SC(n-1))
<= FrTSynGlobalTimeSequenceCounterJumpWidth)
[yes]
[no]
[yes]
SWS_FrTSyn_00167
[no]
SWS_FrTSyn_00164
Valid sequence counter (SC) jumps count = 1
Valid sequence counter (SC) jumps count ++
Valid sequence counter (SC) jumps count = 0
Valid sequence counter (SC) jumps count = 0
[yes]
SWS_FrTSyn_00168
Sequence Counter validation still in hysteresis? (Valid sequence counter
(SC) jumps count < FrTSynGlobalTimeSequenceCounterHysteresis + 1)
[yes]
[no]
Sequence Counter is valid
Sequence Counter is not valid
Final

Figure 7.3: Sequence Counter Validation

[SWS_FrTSyn_00162]

Status: DRAFT
Upstream requirements: RS_TS_20041, RS_TS_20042

⌈Upon reception of a SYNC message, if the message is the first SYNC message after startup, then a Time Slave shall consider the `Sequence Counter value` as valid.⌋

Rationale: After startup it makes sense to skip the Sequence Counter check and to allow the Sequence Counter of the Time Slave to synchronize to the one of the Time Master.

[SWS_FrTSyn_00163]

Status: DRAFT
Upstream requirements: RS_TS_20041, RS_TS_20042

⌈Upon reception of a SYNC message, if the Sequence Counter check is disabled for SYNC messages (i.e., `FrTSynGlobalTimeSequenceCounterJumpWidth` == 0), then a Time Slave shall consider the `Sequence Counter value` of the SYNC message as valid.⌋

[SWS_FrTSyn_00164]

Status: DRAFT
Upstream requirements: RS_TS_20041, RS_TS_20042

⌈Upon reception of a SYNC message, if

- the message is not the first SYNC message after startup
- and Sequence Counter check is enabled (i.e., `FrTSynGlobalTimeSequenceCounterJumpWidth` > 0)
- and the Time Domain is not in timeout (i.e., `TIMEOUT` bit not set in Time Base synchronization status `timeBaseStatus`)

then a Time Slave shall check the difference value between the Sequence Counter of the current message and the Sequence Counter of the previous SYNC message.

If the difference value is greater than 0 and less or equal than `FrTSynGlobalTimeSequenceCounterJumpWidth` , a Time Slave shall consider the Sequence Counter value as valid, else as invalid.⌋

7.5.2.3.1 Sequence Counter Hysteresis

This chapter specifies how to apply an optional hysteresis (`FrTSynGlobalTimeSequenceCounterHysteresis`, refer to [SWS_FrTSyn_00168]) to check if the `Sequence Counter` **`value`** is valid, i.e., if the Sequence Counter check is actually successful.

This requires that a number of consecutive Sequence Counter **jump**s are valid. Requirements [SWS_FrTSyn_00165], [SWS_FrTSyn_00166] and [SWS_FrTSyn_00167] specify when an individual Sequence Counter jump is considered to be valid.

The hysteresis improves robustness against a scenario with a buggy master implementation or injection of invalid master messages, i.e., when the Sequence Counter increments by more than FrTSynGlobalTimeSequenceCounterJumpWidth. In such a scenario (without any hysteresis) a message with any (also invalid) Sequence Counter value would cause the Time Slave to leave the Timeout state although the Sequence Counter is not incremented correctly. A hysteresis avoids this.

[SWS_FrTSyn_00165]

Status: DRAFT
Upstream requirements: RS_TS_20041, RS_TS_20042

⌈Upon reception of a SYNC message, if

- Sequence Counter check is enabled (i.e., FrTSynGlobalTimeSequenceCounterJumpWidth > 0)
- and the Time Domain is in timeout (i.e., TIMEOUT bit set in Time Base synchronization status timeBaseStatus)
- and the Sequence Counter is stuck, i.e., the value of the difference between the Sequence Counter of the current message and the Sequence Counter of the previous SYNC message is 0,

then a Time Slave shall consider the Sequence Counter jump as invalid.⌋

[SWS_FrTSyn_00166]

Status: DRAFT
Upstream requirements: RS_TS_20041, RS_TS_20042

⌈Upon reception of a SYNC message, if

- Sequence Counter check is enabled (i.e., FrTSynGlobalTimeSequenceCounterJumpWidth > 0)
- and the Time Domain is in timeout (i.e., TIMEOUT bit set in Time Base synchronization status timeBaseStatus)
- and the message is the first SYNC message in Timeout for which the Sequence Counter is not stuck,

then a Time Slave shall consider the Sequence Counter jump as valid.⌋

Rationale: After a Timeout (e.g., due to a reset or disconnect of the Time Master) it is very likely that the Sequence Counter of the first received Timesync message is out of sync, i.e., the Sequence Counter difference exceeds FrTSynGlobalTimeSequenceCounterJumpWidth . To allow for faster re-synchronization of the Sequence Counter to the Time Master, the Sequence Counter of the first Timesync message is not checked for FrTSynGlobalTimeSequenceCounterJumpWidth.

However, a stuck Sequence Counter will always, i.e., also in this situation, be considered as invalid (refer to [SWS_FrTSyn_00165]).

[SWS_FrTSyn_00167]

Status: DRAFT
Upstream requirements: RS_TS_20041, RS_TS_20042

⌈Upon reception of a SYNC message, if

- Sequence Counter check is enabled (i.e., FrTSynGlobalTimeSequenceCounterJumpWidth > 0)
- and the Time Domain is in timeout (i.e., TIMEOUT bit set in Time Base synchronization status timeBaseStatus)
- and the Sequence Counter is not stuck, i.e., the value of the difference between the Sequence Counter of the current message and the Sequence Counter of of the previous message is not 0
- and the message is not the first SYNC message in Timeout for which the Sequence Counter is not stuck,

then a Time Slave shall check if the value of the difference between the Sequence Counter of the current message and the Sequence Counter of the previous SYNC message exceeds the threshold FrTSynGlobalTimeSequenceCounterJumpWidth .

If the difference value exceeds the threshold FrTSynGlobalTimeSequenceCounterJumpWidth, a Time Slave shall consider the Sequence Counter jump as invalid, else as valid.⌋

[SWS_FrTSyn_00168]

Status: DRAFT
Upstream requirements: RS_TS_20041, RS_TS_20042

⌈Upon reception of a SYNC message, if

- Sequence counter check is enabled (i.e., FrTSynGlobalTimeSequenceCounterJumpWidth > 0)
- and the Time Domain is in timeout (i.e., TIMEOUT bit set in Time Base synchronization status timeBaseStatus),

then a Time Slave shall check the number of consecutive valid Sequence Counter jumps (refer to requirements [SWS_FrTSyn_00165], [SWS_FrTSyn_00166] and [SWS_FrTSyn_00167])

If the number of consecutive valid Sequence Counter jumps exceeds the value FrTSynGlobalTimeSequenceCounterHysteresis, a Time Slave shall consider the Sequence Counter value as valid, else as invalid.⌋

7.5.2.4 CRC Validation

[SWS_FrTSyn_00050]

Upstream requirements: RS_TS_20042, RS_TS_20043

⌈The function Crc_CalculateCRC8H2F as defined in [6] shall be used to validate the CRC, if configured.⌋

[SWS_FrTSyn_00054]

Upstream requirements: RS_TS_20042, RS_TS_20043

⌈The DataID shall be calculated as DataID = DataIDList[SC], where DataIDList is given by configuration for each message Type.⌋

Note: A specific DataID out of a predefined DataIDList ensures the identification of data elements of Time Synchronization messages.

[SWS_FrTSyn_00055]

Upstream requirements: RS_TS_20042, RS_TS_20043

⌈The CRC shall be calculated over Time Synchronization message byte 2 to byte 15 and DataID, where byte 2 is applied first, followed by the other bytes in ascending order, and DataID last.⌋

7.5.2.5 ICV Verification

Refer to the chapter 7.3.13 in StbM [5] for the configuration details of FV referenced in each Time Domain.

[SWS_FrTSyn_00137]

Upstream requirements: RS_TS_20074

⌈When the FV is referenced (refer FrTSynIcvVerificationFvIdRef) and the FVL of the received SYNC message is greater than 0, the Time Slave shall call the StbM_GetRxFreshness Api to obtain the FV by using:

- the StbMFreshnessValueId from the reference FrTSynIcvVerificationFvIdRef,

- the StbMTruncatedFreshnessValue as received in the FV field of the SYNC message,
- the StbMTruncatedFreshnessValueLength as received in the FVL field of the SYNC message,
- the StbMFreshnessValueLength from the reference FrTSynIcvVerificationFvIdRef,
- the StbMAuthVerifyAttempts as the number of failed verification attempts for the current message (ICV verification attempt counter).

⌋

[SWS_FrTSyn_00138]

Upstream requirements: RS_TS_20074

⌈When the FVL of the received SYNC message is equal to 0, the Time Slave shall not include the FV in the ICV verification.⌋

[SWS_FrTSyn_00139]

Upstream requirements: RS_TS_20074

⌈When the FV is not referenced (refer FrTSynIcvVerificationFvIdRef) and the FVL of the received SYNC message is greater than 0, the Time Slave shall stop the ICV verification and consider the ICV verification as failed.⌋

[SWS_FrTSyn_00140]

Upstream requirements: RS_TS_20074

⌈If StbM_GetRxFreshness returns E_OK, the Time Slave shall use the FV in ICV verification.⌋

[SWS_FrTSyn_00142]

Upstream requirements: RS_TS_20074

⌈If StbM_GetRxFreshness returns a non-recoverable error code (i.e, E_NOT_OK), the Time Slave shall

- consider the ICV verification of the received SYNC message as failed,
- stop the ICV verification,
- call Det_ReportRuntimeError with parameter ErrorId set to FRTSYN_E_FRESHNESSFAILURE (refer to [SWS_FrTSyn_91000])
- and call IdsM_SetSecurityEventWithContextData with parameter EventId set to SEV_TSYN_FR_FRESHNESS_NOT_AVAILABLE

⌋

Refer to the chapter 10.2.5 in [7] for the configuration details of CSM job used for ICV verification.

[SWS_FrTSyn_00143]

Upstream requirements: RS_TS_20074

⌈If FrTSynIcvVerificationBase for the Time Domain is configured to ICV_MAC, the Time Slave shall call Csm_MacVerify to verify the ICV value, using as many bytes as specified in the ICVL.⌋

[SWS_FrTSyn_00144]

Upstream requirements: RS_TS_20074

⌈If FrTSynIcvVerificationBase for the Time Domain is configured to ICV_SIGNATURE, the Time Slave shall call Csm_SignatureVerify to verify the ICV value, using as many bytes as specified in the ICVL.⌋

Note: The mode parameter is intentionally left open for the implementer to choose (i.e. CRYPTO_OPERATIONMODE_SINGLECALL would possibly be the best option since it does not require further calls to CSM).

The CSM job used to generate the ICV can be configured to synchronous or asynchronous behavior.

[SWS_FrTSyn_00145]

Upstream requirements: RS_TS_20074

⌈If FrTSynIcvVerificationTimeout is set to 0, then the Time Slave shall not do ICV verification timeout monitoring.⌋

[SWS_FrTSyn_00146]

Upstream requirements: RS_TS_20074

⌈If Csm_MacVerify or Csm_SignatureVerify return a recoverable error code (e.g., CRYPTO_E_BUSY or CRYPTO_QUEUE_FULL), then the Time Slave shall

- consider the verification of the received SYNC message as failed
- and increment the ICV authentication build counter for this SYNC message.

⌋

[SWS_FrTSyn_00147]

Upstream requirements: RS_TS_20074

⌈If:

- FrTSynIcvVerificationTimeout is set to any value greater than 0,

- and Csm_MacVerify or Csm_SignatureVerify returns E_OK,

the Time Slave shall start the FrTSynIcvVerificationTimeout.⌋

[SWS_FrTSyn_00148]

Upstream requirements: RS_TS_20074

⌈If:

- FrTSynIcvVerificationTimeout is set to any value greater than 0,
- and the FrTSyn_IcvVerificationIndication callback is called,

the Time Slave shall stop the FrTSynIcvVerificationTimeout.⌋

[SWS_FrTSyn_00149]

Upstream requirements: RS_TS_20074

⌈If one of the following conditions is true:

- the authentication build counter has reached the configuration value FrTSynRxAuthenticationBuildAttempts,
- the ICV verification attempt counter has reached the configuration value FrTSynIcvVerificationAttempts,
- the verification of the ICV has returned a non-recoverable error such as returning E_NOT_OK, or KEY_FAILURE,
- the ICVL is 0 in the received SYNC message,
- FrTSynIcvVerificationTimeout expires before the notification of the FrTSyn_IcvVerificationIndication callback

then the Time Slave shall

- stop the ICV verification and consider the ICV verification as failed
- and call IdsM_SetSecurityEventWithContextData with parameter EventId set to SEV_TSYN_FR_ICV_VERIFICATION_FAILED.

⌋

[SWS_FrTSyn_00157]

Status: DRAFT
Upstream requirements: RS_TS_20074

⌈For every reception of messages that require ICV verification the Time Slave shall maintain an authentication build counter (refer FrTSynRxAuthenticationBuildAttempts).⌋

[SWS_FrTSyn_00158]

Status: DRAFT
Upstream requirements: RS_TS_20074

⌈Upon the initial processing of messages that require ICV verification (i.e., upon the first attempt of a freshness value and ICV verification for each received message) the Time Slave shall set the authentication build counter to 0.⌋

[SWS_FrTSyn_00159]

Status: DRAFT
Upstream requirements: RS_TS_20074

⌈If `StbM_GetRxFreshness` returns a recoverable error code (e.g., `STBM_E_BUSY`), then the Time Slave shall

- increment the authentication build counter
- and not do ICV verification.

⌋

[SWS_FrTSyn_00160]

Status: DRAFT
Upstream requirements: RS_TS_20074

⌈If

- verification of the authenticated message has failed
- and the authentication build counter has not yet reached the configuration value `FrTSynRxAuthenticationBuildAttempts`,

then the Time Slave shall retry the freshness attempt and ICV verification in the next call of `FrTSyn_MainFunction`.⌋

[SWS_FrTSyn_00172]

Status: DRAFT
Upstream requirements: RS_TS_20074

⌈If the verification of the ICV could be successfully executed but the verification failed (e.g. the MAC verification has failed or the key was invalid), then the Time Slave shall

- increment the ICV verification attempt counter
- and set the authentication build counter to 0.

⌋

Note: Resetting the authentication build counter will prevent dropping the authentication process too early even though ICV verification attempts are still possible.

7.5.2.6 Message Disassembling

[SWS_FrTSyn_00056]

Upstream requirements: RS_TS_20043

⌈For each received Time Synchronization message the FrTSyn shall validate the message as follows (all conditions must match):

1. Type matches depending on the FrTSynRxCrcValidated parameter
2. SC value is valid (refer to requirements [SWS_FrTSyn_00162] to [SWS_FrTSyn_00168])
3. D matches to the defined Time Domain range for each Type
4. D matches to one of the configured Time Domains
5. SyncTimeNSec matches the defined range of StbM_TimeStampType.nanoseconds.
6. CRC (including DataID) matches depending on the FrTSynRxCrcValidated parameter.
7. ICV verification is successful, depending on the FrTSynRxIcvVerificationType parameter.

⌋

[SWS_FrTSyn_00057]

Upstream requirements: RS_TS_20042, RS_TS_20043

⌈If the validation of received Time Synchronization message is successful (refer to [SWS_FrTSyn_00056]), the FrTSyn shall disassemble the message and forward the global time via StbM_BusSetGlobalTime to StbM.⌋

[SWS_FrTSyn_00150]

Upstream requirements: RS_TS_20042, RS_TS_20043

⌈If the validation of the received Time Synchronization message has failed(refer to [SWS_FrTSyn_00056]), the FrTSyn shall discard the message.⌋

7.6 Time Recording

7.6.1 Global Time Measurement Support

[SWS_FrTSyn_00092]

Upstream requirements: RS_TS_00034

⌈On an invocation of StbM_BusSetGlobalTime the member pathDelay of the measureDataPtr structure shall be set to 0.⌋

7.6.2 Time Validation

[SWS_FrTSyn_00096]

Upstream requirements: RS_TS_00034

⌈The FrTSyn shall support Time Validation, if FrTSynTimeValidationSupport set to TRUE.⌋

[SWS_FrTSyn_00097]

Upstream requirements: RS_TS_00034

⌈
If

- FrTSynTimeValidationSupport is enabled and
- FrTSynEnableTimeValidation for the Time Domain is enabled

FrTSyn shall do time recording for Time Validation for that Time Domain.
⌋

[SWS_FrTSyn_00098]

Upstream requirements: RS_TS_00034

⌈
If

- time recording for Time Validation is enabled for a Time Domain (refer to [SWS_FrTSyn_00096] and [SWS_FrTSyn_00097]) and
- FrTSyn is configured as Time Slave for that Time Domain,

FrTSyn shall call StbM_FrSetSlaveTimingData upon successful reception of a SYNC message.

StbM_FrSetSlaveTimingData shall be called after StbM_BusSetGlobalTime.⌋

Note: StbM_BusSetGlobalTime shall be called first, because it updates the Sync-local Time Tuple (refer to [5]), which is required by StbM_FrSetSlaveTimingData. Refer to figure 9.2 for the overall sequence of API calls for a Time Slave.

[SWS_FrTSyn_00099]

Upstream requirements: RS_TS_00034

⌈Upon invocation of StbM_FrSetSlaveTimingData FrTSyn shall pass following values

- the Sequence Counter as received in the SYNC message,
- the segment id of the physical channel on which the SYNC message has been received (refer to parameter FrTSynGlobalTimeNetworkSegmentId)
- currentCycle and currentMacroticks and FCNT as read upon reception of the SYNC message (refer to step 2 in [SWS_FrTSyn_00046]),
- CycleLength and MacrotickDuration
- the Sync ingress timestamp $T1_{VLT}$ as retrieved in step 1 in [SWS_FrTSyn_00046])
- T0 as received in the SYNC message (refer to step 1 in [SWS_FrTSyn_00046]),

by the parameter measureDataPtr.

Struct members

- measureDataPtr→referenceLocalTimestamp and
- measureDataPtr→referenceGlobalTimestampSec

shall be passed as 0.⌋

Note: MacrotickDuration is calculated as CycleLength / MacroticksPerCycle

Note: The FrTSyn passes 0 to avoid undefined values. The structure members referenceLocalTimestamp and referenceGlobalTimestampSec will be set by the StbM StbM_FrSetSlaveTimingData internally (refer to [SWS_StbM_00471] in [5]).

[SWS_FrTSyn_00100]

Upstream requirements: RS_TS_00034

⌈
If

- time recording for Time Validation is enabled for a Time Domain (refer to [SWS_FrTSyn_00096] and [SWS_FrTSyn_00097]) and
- FrTSyn is configured as Time Master for that Time Domain,

FrTSyn shall call StbM_FrSetMasterTimingData upon successful transmission of a SYNC message.⌋

Note: Refer to figure 9.1 for the overall sequence of API calls for a Time Master.

[SWS_FrTSyn_00101]

Upstream requirements: RS_TS_00034

⌈Upon invocation of StbM_FrSetMasterTimingData FrTSyn shall pass the following data

- the Sequence Counter as sent in the SYNC message
- the segment id of the physical channel on which the SYNC message has been sent (refer to parameter FrTSynGlobalTimeNetworkSegmentId)
- currentCycle and currentMacroticks read upon construction of the Sync message (refer to step 2 in [SWS_FrTSyn_00028]),
- cycleLength and macrotickDuration
- the reference timestamp $T1_{VLT}$ (refer to step 2 In [SWS_FrTSyn_00028]),
- T0 as sent in the SYNC message (refer to step 3 In [SWS_FrTSyn_00028]),

by the parameter measureDataPtr.⌋

7.7 Security Events

[SWS_FrTSyn_00105]

Upstream requirements: SRS_BSW_00489

⌈If security event reporting has been enabled for the FrTSyn module (FrTSynEnableSecurityEventReporting = true) the respective security events shall be reported to the IdsM [8] via the interfaces defined in BSWGeneral [3].⌋

The following table lists the security events which are standardized for the FrTSyn together with their trigger conditions.

[SWS_FrTSyn_00103] Security events for FrTSyn

Status: DRAFT
Upstream requirements: RS_Ids_00810

⌈

Name	***Description***	***ID***
SEV_TSYN_FR_ICV_GENERATION_FAILED	ICV generation for a Sync message has failed.	70
SEV_TSYN_FR_ICV_VERIFICATION_FAILED	ICV verification of a received Sync message has failed.	71
SEV_TSYN_FR_FRESHNESS_NOT_AVAILABLE	Failed to get freshness value from FvM.	72

⌋

The following tables specify the context data which shall be reported for the respective security events.

[SWS_FrTSyn_92000] Security event context data definition: SEV_TSYN_FR_ICV_GENERATION_FAILED

Status: DRAFT
Upstream requirements: RS_Ids_00810

⌈

SEV Name	***SEV_TSYN_FR_ICV_GENERATION_FAILED***	
ID	70	
Description	ICV generation for a Sync message has failed.	
Context Data Version	1	
Context Data	***Data Type***	***Allowed Values***
GlobalTimeDomainId	uint8	

⌋

[SWS_FrTSyn_92001] Security event context data definition: SEV_TSYN_FR_ICV_VERIFICATION_FAILED

Status: DRAFT
Upstream requirements: RS_Ids_00810

⌈

SEV Name	***SEV_TSYN_FR_ICV_VERIFICATION_FAILED***	
ID	71	
Description	ICV verification of a received Sync message has failed.	
Context Data Version	1	
Context Data	***Data Type***	***Allowed Values***
GlobalTimeDomainId	uint8	

⌋

[SWS_FrTSyn_92002] Security event context data definition: SEV_TSYN_FR_FRESHNESS_NOT_AVAILABLE

Status: DRAFT
Upstream requirements: RS_Ids_00810

⌈

SEV Name	***SEV_TSYN_FR_FRESHNESS_NOT_AVAILABLE***	
ID	72	
Description	Failed to get freshness value from FvM.	
Context Data Version	1	
Context Data	***Data Type***	***Allowed Values***
GlobalTimeDomainId	uint8	

⌋

7.8 Error Classification

Section 7.2 "Error Handling" of the document "General Specification of Basic Software Modules" [3] describes the error handling of the Basic Software in detail. Above all, it constitutes a classification scheme consisting of five error types which may occur in BSW modules.

Based on this foundation, the following section specifies particular errors arranged in the respective subsections below.

7.8.1 Development Errors

[SWS_FrTSyn_00059] Definiton of development errors in module FrTSyn

Upstream requirements: SRS_BSW_00385

⌈

Type of error	***Related error code***	***Error value***
API service called with wrong PDU or SDU.	FRTSYN_E_INVALID_PDUID	0x01
API service used in un-initialized state	FRTSYN_E_UNINIT	0x20
A pointer is invalid	FRTSYN_E_NULL_POINTER	0x21
FrTSyn initialization failed	FRTSYN_E_INIT_FAILED	0x22
API called with invalid parameter	FRTSYN_E_PARAM	0x23
Invalid Controller index	FRTSYN_E_INV_CTRL_IDX	0x24

⌋

7.8.2 Runtime Errors

[SWS_FrTSyn_91000] Definiton of runtime errors in module FrTSyn

Upstream requirements: SRS_BSW_00385

⌈

Type of error	***Related error code***	***Error value***
No FV available from the FVM	FRTSYN_E_FRESHNESSFAILURE	0x01

⌋

7.8.3 Production Errors

There are no production errors.

7.8.4 Extended Production Errors

There are no extended production errors.

8 API specification

8.1 API

8.1.1 Imported types

In this section all types included from the following modules are listed:

[SWS_FrTSyn_00060] Definition of imported datatypes of module FrTSyn

Upstream requirements: RS_TS_20043

⌈

Module	***Header File***	***Imported Type***
Comtype	ComStack_Types.h	PduIdType
	ComStack_Types.h	PduInfoType
	ComStack_Types.h	PduLengthType
Csm	Rte_Csm_Type.h	Crypto_OperationModeType
	Rte_Csm_Type.h	Crypto_ResultType
	Rte_Csm_Type.h	Crypto_VerifyResultType
Eth	Eth.h	Eth_RateDeviationStatusType (draft)
	Eth.h	Eth_RateDeviationType (draft)
FrIf	FrIf.h	FrIf_StateType
IdsM	IdsM_Types.h	IdsM_SecurityEventIdType
StbM	Rte_StbM_Type.h	StbM_FrTimeMasterMeasurementType
	Rte_StbM_Type.h	StbM_FrTimeSlaveMeasurementType
	Rte_StbM_Type.h	StbM_SynchronizedTimeBaseType
	Rte_StbM_Type.h	StbM_TimeBaseStatusType
	Rte_StbM_Type.h	StbM_TimeStampShortType
	Rte_StbM_Type.h	StbM_TimeStampType
	Rte_StbM_Type.h	StbM_TimeTupleType
	Rte_StbM_Type.h	StbM_UserDataType
	StbM.h	StbM_MeasurementType
	StbM.h	StbM_VirtualLocalTimeType
Std	Std_Types.h	Std_ReturnType
	Std_Types.h	Std_VersionInfoType

⌋

8.1.2 Type definitions

8.1.2.1 FrTSyn_ConfigType

[SWS_FrTSyn_00061] Definition of datatype FrTSyn_ConfigType

Upstream requirements: RS_TS_20043

⌈

Name	FrTSyn_ConfigType	
Kind	Structure	
Elements	implementation specific	
	Type	–
	Comment	–
Description	This is the base type for the configuration of the Time Synchronization over FlexRay. A pointer to an instance of this structure will be used in the initialization of the Time Synchronization over FlexRay. The content of this structure is defined in chapter 10 Configuration specification.	
Available via	FrTSyn.h	

⌋

8.1.2.2 FrTSyn_TransmissionModeType

[SWS_FrTSyn_00062] Definition of datatype FrTSyn_TransmissionModeType

Upstream requirements: RS_TS_20043

⌈

Name	FrTSyn_TransmissionModeType		
Kind	Enumeration		
Range	FRTSYN_TX_OFF	–	Transmission Disabled
	FRTSYN_TX_ON	–	Transmission Enabled
Description	Handles the enabling and disabling of the transmission mode		
Available via	FrTSyn.h		

⌋

8.1.3 Function definitions

8.1.3.1 FrTSyn_Init

[SWS_FrTSyn_00063] Definition of API function FrTSyn_Init

Upstream requirements: RS_TS_20043

⌈

Service Name	FrTSyn_Init	

Syntax	`void FrTSyn_Init (`
	`  const FrTSyn_ConfigType* configPtr`
	`)`	

Service ID [hex]	0x01	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	configPtr	Pointer to selected configuration structure
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This function initializes the Time Synchronization over FlexRay.	
Available via	FrTSyn.h	

⌋

See section 7.2.1 for details.

8.1.3.2 FrTSyn_GetVersionInfo

[SWS_FrTSyn_00064] Definition of API function FrTSyn_GetVersionInfo

Upstream requirements: RS_TS_20043

⌈

Service Name	FrTSyn_GetVersionInfo	

Syntax	`void FrTSyn_GetVersionInfo (`
	`  Std_VersionInfoType* versioninfo`
	`)`	

Service ID [hex]	0x02	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	versioninfo	Pointer to where to store the version information of this module.
Return value	None	

▽

△

Description	Returns the version information of this module.
Available via	FrTSyn.h

⌋

8.1.3.3 FrTSyn_SetTransmissionMode

[SWS_FrTSyn_00065] Definition of API function FrTSyn_SetTransmissionMode

Upstream requirements: RS_TS_20043

⌈

Service Name	FrTSyn_SetTransmissionMode	

Syntax	`void FrTSyn_SetTransmissionMode (`
	`  uint8 CtrlIdx,`
	`  FrTSyn_TransmissionModeType Mode`
	`)`	

Service ID [hex]	0x03	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the FlexRay channel
	Mode	FRTSYN_TX_OFF FRTSYN_TX_ON
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This API is used to turn on and off the TX capabilities of the FrTSyn.	
Available via	FrTSyn.h	

⌋

[SWS_FrTSyn_00095]

Upstream requirements: SRS_BSW_00323, SRS_BSW_00337

⌈The function `FrTSyn_SetTransmissionMode` shall inform the `Det`, if development error detection is enabled (`FrTSynDevErrorDetect` is set to `TRUE`) and if function call has failed because of the following reasons:

- Invalid CtrlIdx (`FRTSYN_E_INV_CTRL_IDX`)
- Invalid Mode (`FRTSYN_E_PARAM`)

⌋

8.1.4 Call-back notifications

This is a list of functions provided for other modules.

8.1.4.1 FrTSyn_RxIndication

[SWS_FrTSyn_00066] Definition of callback function FrTSyn_RxIndication

Upstream requirements: RS_TS_20043

⌈

Service Name	FrTSyn_RxIndication	

Syntax	`void FrTSyn_RxIndication (`
	`  PduIdType RxPduId,`
	`  const PduInfoType* PduInfoPtr`
	`)`	

Service ID [hex]	0x42	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	RxPduId	ID of the received PDU.
	PduInfoPtr	Contains the length (SduLength) of the received PDU, a pointer to a buffer (SduDataPtr) containing the PDU, and the MetaData related to this PDU.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Indication of a received PDU from a lower layer communication interface module.	
Available via	FrTSyn.h	

⌋

Note: The callback function `FrTSyn_RxIndication` called by the `FrIf` module and implemented by the `FrTSyn` module. It is called in case of a receive indication event of the `FR` Driver.

[SWS_FrTSyn_00067]

Upstream requirements: SRS_BSW_00323, SRS_BSW_00337

⌈The callback function `FrTSyn_RxIndication` shall inform the `Det`, if development error detection is enabled `FrTSynDevErrorDetect` is set to `TRUE`) and if function call has failed because of the following reasons:

- Invalid `RxPduId` (`FRTSYN_E_INVALID_PDUID`)
- `PduInfoPtr` or `SduDataPtr` equals `NULL_PTR` (`FRTSYN_E_NULL_POINTER`)

⌋

Caveats of FrTSyn_RxIndication

- The FrTSyn module is initialized correctly.

8.1.4.2 FrTSyn_TriggerTransmit

[SWS_FrTSyn_00069] Definition of callback function FrTSyn_TriggerTransmit

Upstream requirements: RS_TS_20043

⌈

Service Name	FrTSyn_TriggerTransmit	

Syntax	`Std_ReturnType FrTSyn_TriggerTransmit (`
	`  PduIdType TxPduId,`
	`  PduInfoType* PduInfoPtr`
	`)`	

Service ID [hex]	0x41	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	TxPduId	ID of the SDU that is requested to be transmitted.
Parameters (inout)	PduInfoPtr	Contains a pointer to a buffer (SduDataPtr) to where the SDU data shall be copied, and the available buffer size in SduLengh. On return, the service will indicate the length of the copied SDU data in SduLength.
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: SDU has been copied and SduLength indicates the number of copied bytes. E_NOT_OK: No SDU data has been copied. PduInfoPtr must not be used since it may contain a NULL pointer or point to invalid data.
Description	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr.	
Available via	FrTSyn.h	

⌋

Note: The function FrTSyn_TriggerTransmit might be called by the FrTSyn module's environment in an interrupt context.

[SWS_FrTSyn_00070]

Upstream requirements: SRS_BSW_00323, SRS_BSW_00337

⌈The callback function FrTSyn_TriggerTransmit shall inform the Det, if development error detection is enabled FrTSynDevErrorDetect is set to TRUE) and if function call has failed because of the following reasons:

- Invalid TxPduId (FRTSYN_E_INVALID_PDUID)
- PduInfoPtr or SduDataPtr equals NULL_PTR (FRTSYN_E_NULL_POINTER)

⌋

8.1.4.3 FrTSyn_IcvGenerationIndication

[SWS_FrTSyn_91001] Definition of API function FrTSyn_IcvGenerationIndication

Upstream requirements: RS_TS_20074

⌈

Service Name	FrTSyn_IcvGenerationIndication	

Syntax	`void FrTSyn_IcvGenerationIndication (`
	`  uint32 jobId,`
	`  Crypto_ResultType result`
	`)`	

Service ID [hex]	0x5	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	jobId	JobID of the operation that caused the callback.
	result	Contains the result of the cryptographic operation.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	By this API service the FrTSyn gets an indication and the result of ICV generation.	
Available via	FrTSyn.h	

⌋

[SWS_FrTSyn_00151]

Upstream requirements: SRS_BSW_00323, SRS_BSW_00337

⌈The function `FrTSyn_IcvGenerationIndication` shall inform the `Det`, if development error detection is enabled (`FrTSynDevErrorDetect` is set to `TRUE`) and if the function call has failed because of the following reasons:

- jobId is invalid (`FRTSYN_E_PARAM`)

⌋

8.1.4.4 FrTSyn_IcvVerificationIndication

[SWS_FrTSyn_91002] Definition of API function FrTSyn_IcvVerificationIndication

Upstream requirements: RS_TS_20074

⌈

Service Name	FrTSyn_IcvVerificationIndication	

Syntax	`void FrTSyn_IcvVerificationIndication (`
	`  uint32 jobId,`
	`  Crypto_ResultType result`
	`)`	

Service ID [hex]	0x6	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	jobId	JobID of the operation that caused the callback.
	result	Contains the result of the cryptographic operation.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	By this API service the FrTSyn gets an indication and the result of ICV verification.	
Available via	FrTSyn.h	

⌋

[SWS_FrTSyn_00152]

Upstream requirements: SRS_BSW_00323, SRS_BSW_00337

⌈The function `FrTSyn_IcvVerificationIndication` shall inform the `Det`, if development error detection is enabled (`FrTSynDevErrorDetect` is set to `TRUE`) and if the function call has failed because of the following reasons:

- jobId is invalid (`FRTSYN_E_PARAM`)

⌋

8.1.5 Scheduled functions

These functions are directly called by the Basic Software Scheduler. The following functions shall have no return value and no parameters. All functions shall be non-reentrant.

8.1.5.1 FrTSyn_MainFunction

[SWS_FrTSyn_00071] Definition of scheduled function FrTSyn_MainFunction

Upstream requirements: RS_TS_20043

⌈

Service Name	FrTSyn_MainFunction

Syntax	`void FrTSyn_MainFunction (`
	`  void`
	`)`

Service ID [hex]	0x04
Description	Main function for cyclic call / resp. Timesync message transmission
Available via	FrTSyn_SchM.h

⌋

[SWS_FrTSyn_00072]

Upstream requirements: RS_TS_20043

⌈The frequency of invocations of `FrTSyn_MainFunction` is determined by the configuration parameter `FrTSynMainFunctionPeriod`.
⌋

8.1.6 Expected Interfaces

In this section, all interfaces required by other modules are listed.

8.1.6.1 Mandatory Interfaces

This section defines all interfaces that are required to fulfill a mandatory functionality of the module.

[SWS_FrTSyn_00074] Definition of mandatory interfaces required by module Fr TSyn

Upstream requirements: RS_TS_20043

⌈

API Function	***Header File***	***Description***
FrIf_GetCycleLength	FrIf.h	This API returns the configured time of the configuration parameter "GdCycle" in nanoseconds for the FlexRay controller with index FrIf_CtrlIdx.

▽

△

API Function	Header File	Description
FrIf_GetGlobalTime	FrIf.h	Wraps the FlexRay Driver API function Fr_GetGlobalTime(). Important Note: FrIf_GetGlobalTime may be called within an exclusive area.
FrIf_GetMacroticksPerCycle	FrIf.h	Retrieves the amount of Macroticks per Cycle
FrIf_GetState	FrIf.h	Get current FrIf state.
StbM_GetCurrentVirtualLocalTime	StbM.h	Returns the Virtual Local Time of the referenced Time Base.

⌋

8.1.6.2 Optional Interfaces

This section defines all interfaces that are required to fulfill an optional functionality of the module.

[SWS_FrTSyn_00075] Definition of optional interfaces requested by module FrTSyn

Upstream requirements: RS_TS_20043

⌈

API Function	Header File	Description
Crc_CalculateCRC8H2F	Crc.h	This service makes a CRC8 calculation with the Polynomial 0x2F on Crc_Length
Csm_MacGenerate	Csm.h	Uses the given data to perform a MAC generation and stores the MAC in the memory location pointed to by the MAC pointer.
Csm_MacVerify	Csm.h	Verifies the given MAC by comparing if the MAC is generated with the given data.
Csm_SignatureGenerate	Csm.h	Uses the given data to perform the signature calculation and stores the signature in the memory location pointed by the result pointer.
Csm_SignatureVerify	Csm.h	Verifies the given signature by checking if it was generated with the given data.
Det_ReportError	Det.h	Service to report development errors.
Det_ReportRuntimeError	Det.h	Service to report runtime errors. If a callout has been configured then this callout shall be called.
IdsM_SetSecurityEventWithContextData (obsolete)	IdsM.h	This API is the application interface to report security events with context data to the IdsM. **Tags:** atp.Status=obsolete
LSduR_FrTSynTransmit (draft)	LSduR_FrTSyn.h	Requests transmission of a PDU.
StbM_BusSetGlobalTime	StbM.h	Allows the Time Base Provider Modules to forward the Rx Time Tuple to the StbM.

▽

△

API Function	***Header File***	***Description***
StbM_FrSetMasterTimingData (draft)	StbM_FrTSyn.h	Provides Flexray Timesyn module specific data for a Time Master to the StbM. **Tags:** atp.Status=draft
StbM_FrSetSlaveTimingData (draft)	StbM_FrTSyn.h	Allows the FrTSyn Module to forward Flexray specific details to the StbM. **Tags:** atp.Status=draft
StbM_GetCurrentTime	StbM.h	Returns a time tuple (Local time, Global time and Timebase status) and user data details Note: This API shall be called with locked interrupts / within an Exclusive Area to prevent interruption (i.e., the risk that the time stamp is outdated on return of the function call).
StbM_GetRxFreshness	StbM.h	This interface is used by the StbM to query the current freshness value.
StbM_GetTimeBaseStatus	StbM.h	Returns detailed status information for a Synchronized (or Pure Local) Time Base.
StbM_GetTimeBaseUpdateCounter	StbM.h	Allows the Timesync Modules to detect, whether a Time Base should be transmitted immediately in the subsequent <Bus>TSyn_MainFunction() cycle.
StbM_GetTxFreshness	StbM.h	This API returns the freshness value from the Most Significant Bits in the first byte, of the Freshness array, in big endian format.
StbM_GetTxFreshnessTruncData	StbM.h	This interface is used by the StbM to obtain the current freshness value. The interface function provides also the truncated freshness transmitted in the secured time sync message.
StbM_SPduTxConfirmation	StbM.h	This interface is used by the StbM to indicate that the Secured Time Synchronization Message has been initiated for transmission.

⌋

9 Sequence diagrams

9.1 FlexRay Time Synchronization (Time Master)

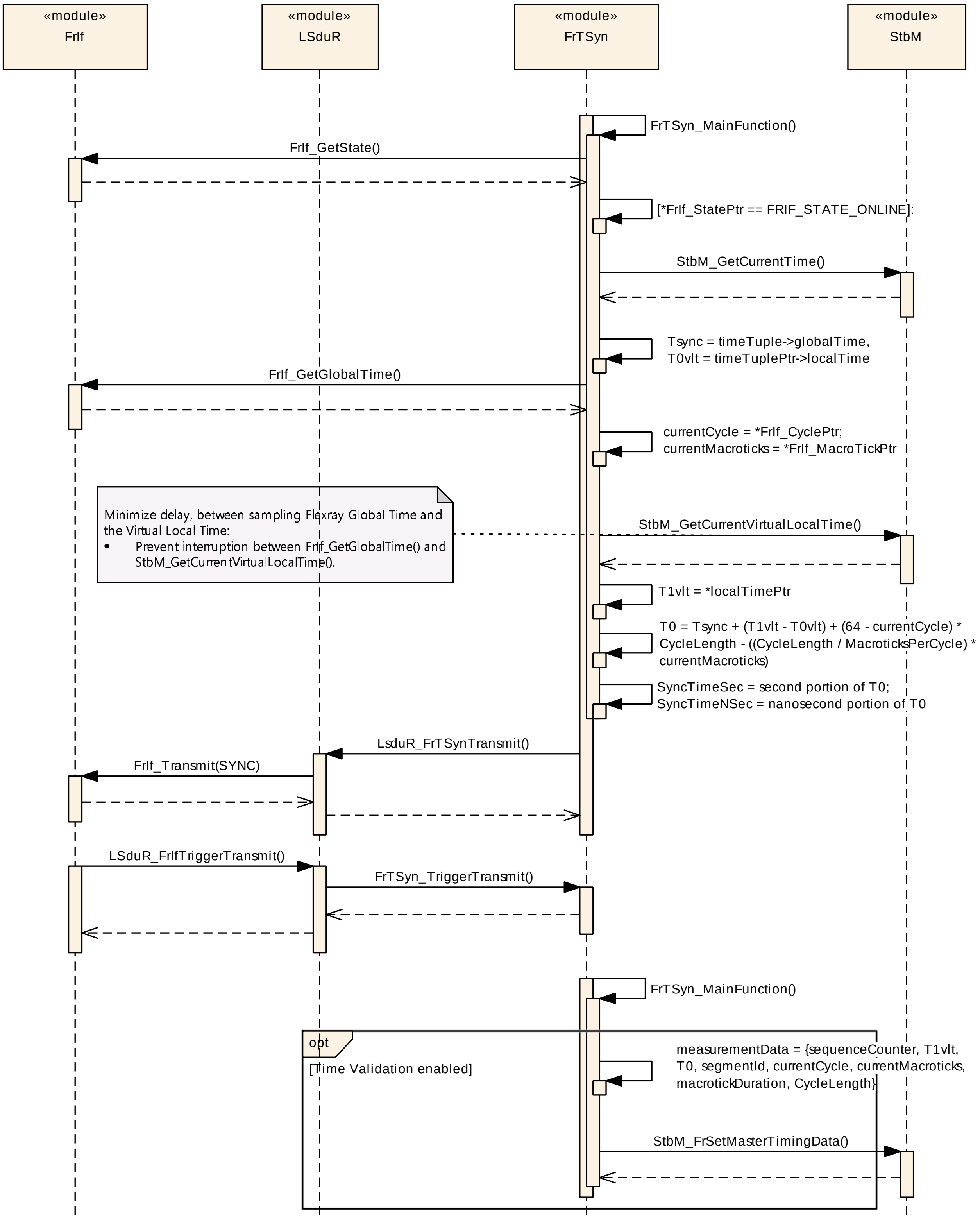
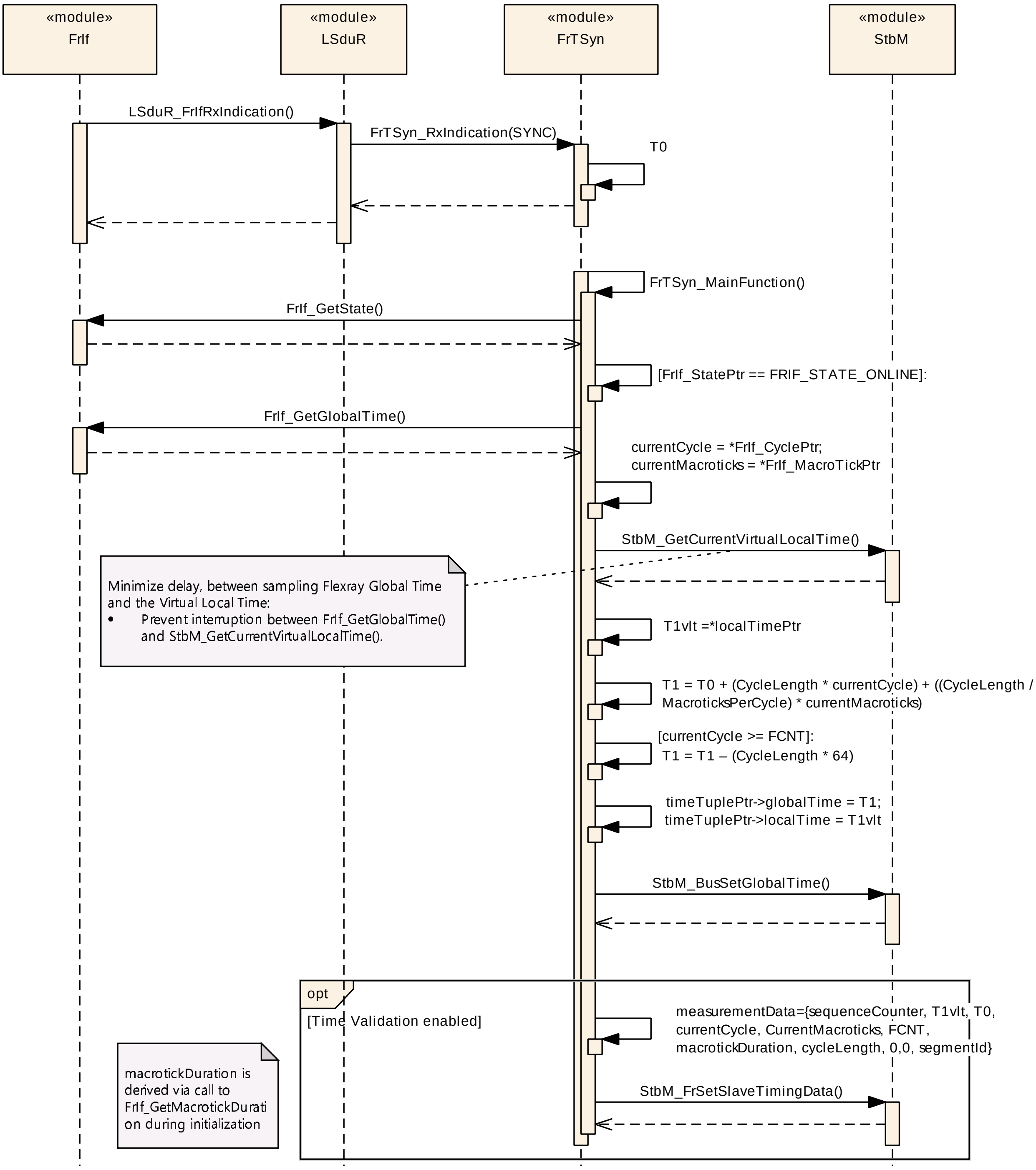

«module»
FrIf
«module»
LSduR
«module»
FrTSyn
«module»
StbM
FrTSyn_MainFunction()
FrIf_GetState()
[*FrIf_StatePtr == FRIF_STATE_ONLINE]:
StbM_GetCurrentTime()
Tsync = timeTuple->globalTime,
T0vlt = timeTuplePtr->localTime
FrIf_GetGlobalTime()
currentCycle = *FrIf_CyclePtr;
currentMacroticks = *FrIf_MacroTickPtr
Minimize delay, between sampling Flexray Global Time and the Virtual Local Time:
• Prevent interruption between FrIf_GetGlobalTime() and StbM_GetCurrentVirtualLocalTime().
StbM_GetCurrentVirtualLocalTime()
T1vlt = *localTimePtr
T0 = Tsync + (T1vlt - T0vlt) + (64 - currentCycle) * CycleLength - ((CycleLength / MacroticksPerCycle) * currentMacroticks)
SyncTimeSec = second portion of T0;
SyncTimeNSec = nanosecond portion of T0
LsduR_FrTSynTransmit()
FrIf_Transmit(SYNC)
LSduR_FrIfTriggerTransmit()
FrTSyn_TriggerTransmit()
FrTSyn_MainFunction()
opt
[Time Validation enabled]
measurementData = {sequenceCounter, T1vlt, T0, segmentId, currentCycle, currentMacroticks, macrotickDuration, CycleLength}
StbM_FrSetMasterTimingData()

Figure 9.1: FlexRay Time Synchronization (Time Master)

9.2 FlexRay Time Synchronization (Time Slave)


«module»
FrIf
«module»
LSduR
«module»
FrTSyn
«module»
StbM
LSduR_FrIfRxIndication()
FrTSyn_RxIndication(SYNC)
T0
FrTSyn_MainFunction()
FrIf_GetState()
[FrIf_StatePtr == FRIF_STATE_ONLINE]:
FrIf_GetGlobalTime()
currentCycle = *FrIf_CyclePtr;
currentMacroticks = *FrIf_MacroTickPtr
StbM_GetCurrentVirtualLocalTime()
T1vlt =*localTimePtr
T1 = T0 + (CycleLength * currentCycle) + ((CycleLength / MacroticksPerCycle) * currentMacroticks)
[currentCycle >= FCNT]:
T1 = T1 – (CycleLength * 64)
timeTuplePtr->globalTime = T1;
timeTuplePtr->localTime = T1vlt
StbM_BusSetGlobalTime()
opt
[Time Validation enabled]
measurementData={sequenceCounter, T1vlt, T0, currentCycle, CurrentMacroticks, FCNT, macrotickDuration, cycleLength, 0,0, segmentId}
StbM_FrSetSlaveTimingData()

Minimize delay, between sampling Flexray Global Time and the Virtual Local Time:

- Prevent interruption between FrIf_GetGlobalTime() and StbM_GetCurrentVirtualLocalTime().

macrotickDuration is derived via call to FrIf_GetMacrotickDurati on during initialization

Figure 9.2: FlexRay Time Synchronization (Time Slave)

9.3 FlexRay Secure Time Synchronization Sequence

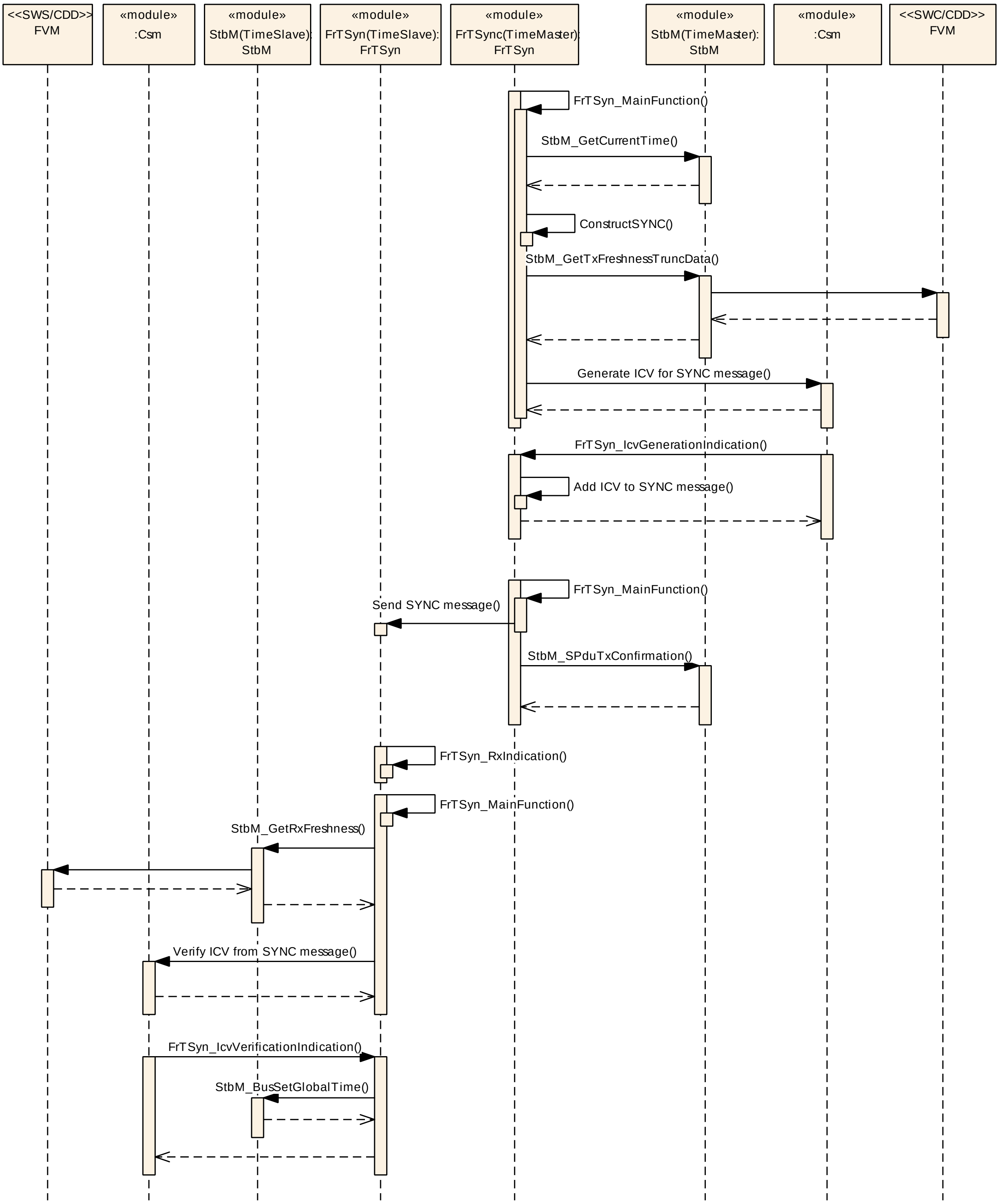

<<SWS/CDD>> FVM
«module» :Csm
«module» StbM(TimeSlave): StbM
«module» FrTSyn(TimeSlave): FrTSyn
«module» FrTSync(TimeMaster): FrTSyn
«module» StbM(TimeMaster): StbM
«module» :Csm
<<SWC/CDD>> FVM
FrTSyn_MainFunction()
StbM_GetCurrentTime()
ConstructSYNC()
StbM_GetTxFreshnessTruncData()
Generate ICV for SYNC message()
FrTSyn_IcvGenerationIndication()
Add ICV to SYNC message()
FrTSyn_MainFunction()
Send SYNC message()
StbM_SPduTxConfirmation()
FrTSyn_RxIndication()
FrTSyn_MainFunction()
StbM_GetRxFreshness()
Verify ICV from SYNC message()
FrTSyn_IcvVerificationIndication()
StbM_BusSetGlobalTime()

Figure 9.3: FlexRay Secure Time Synchronization Sequence

10 Configuration specification

In general, this chapter defines configuration parameters and their clustering into containers. In order to support the specification section 10.1 describes fundamentals. It also specifies a template (table) you shall use for the parameter specification. We intend to leave section 10.1 in the specification to guarantee comprehension.

Section 10.2 specifies the structure (containers) and the parameters of the Time Synchronization over FlexRay.

Section 10.4 specifies published information of the Time Synchronization over FlexRay.

10.1 How to read this chapter

For details, refer to the chapter 10.1 "Introduction to configuration specification" in SWS BSW General [3].

10.2 Containers and configuration parameters

The following sections summarize all configuration parameters of the Time Synchronization over FlexRay. The detailed meaning of the parameters is described in chapters 7 and 8.

10.2.1 Variants

[SWS_FrTSyn_00077]

Upstream requirements: RS_TS_20046

⌈The Time Synchronization over FlexRay shall support the configuration for Time Master, Time Slave and Time Gateway.⌋

The module supports different post-build variants (previously known as post-build selectable configuration sets), but not post-build loadable configuration.

10.2.2 FrTSyn

[ECUC_FrTSyn_00001] Definition of EcucModuleDef FrTSyn ⌈

Module Name	FrTSyn
Description	This represents the specific configuration variant for the TSyn on Flexray.
Post-Build Variant Support	true
Supported Config Variants	VARIANT-PRE-COMPILE

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
FrTSynGeneral	1	This container holds the general parameters of the Flexray-specific Synchronized Time-base Manager
FrTSynGlobalTimeDomain	1..*	This represents the existence of a global time domain on Flexray. The FrTSyn module can administrate several global time domains at the same time that in itself form a hierarchy of domains and sub-domains. If the FrTSyn exists it is assumed that at least one global time domain exists.

⌋

10.2.3 FrTSynGeneral

[ECUC_FrTSyn_00003] Definition of EcucParamConfContainerDef FrTSynGeneral ⌈

Container Name	FrTSynGeneral
Parent Container	FrTSyn
Description	This container holds the general parameters of the Flexray-specific Synchronized Time-base Manager
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrTSynDevErrorDetect	1	[ECUC_FrTSyn_00002]
FrTSynEnableSecurityEventReporting	1	[ECUC_FrTSyn_00044]
FrTSynMainFunctionPeriod	1	[ECUC_FrTSyn_00016]
FrTSynTimeValidationSupport	1	[ECUC_FrTSyn_00040]
FrTSynVersionInfoApi	1	[ECUC_FrTSyn_00019]

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
FrTSynSecurityEventRefs	0..1	Container for the references to IdsMEvent elements representing the security events that the FrTSyn module shall report to the Ids M in case the coresponding security related event occurs (and if FrTSynEnableSecurityEventReporting is set to true). The standardized security events in this container can be extended by vendor-specific security events.

⌋

[ECUC_FrTSyn_00002] Definition of EcucBooleanParamDef FrTSynDevErrorDetect ⌈

Parameter Name	FrTSynDevErrorDetect		
Parent Container	FrTSynGeneral		
Description	Switches the development error detection and notification on or off. • true: detection and notification is enabled. • false: detection and notification is disabled.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00044] Definition of EcucBooleanParamDef FrTSynEnableSecurityEventReporting ⌈

Parameter Name	FrTSynEnableSecurityEventReporting		
Parent Container	FrTSynGeneral		
Description	Switches the reporting of security events to the IdsM: - true: reporting is enabled. - false: reporting is disabled.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00016] Definition of EcucFloatParamDef FrTSynMainFunctionPeriod ⌈

Parameter Name	FrTSynMainFunctionPeriod	
Parent Container	FrTSynGeneral	
Description	Schedule period of the main function FrTSyn_MainFunction. Unit: [s].	
Multiplicity	1	
Type	EcucFloatParamDef	
Range	]0 .. INF[	

▽

△

Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00040] Definition of EcucBooleanParamDef FrTSynTimeValidationSupport ⌈

Parameter Name	FrTSynTimeValidationSupport		
Parent Container	FrTSynGeneral		
Description	Switches support for Time Validation on or off. • true: Time Validation is enabled. • false:Time Validation is disabled.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00019] Definition of EcucBooleanParamDef FrTSynVersionInfoApi ⌈

Parameter Name	FrTSynVersionInfoApi		
Parent Container	FrTSynGeneral		
Description	Activate/Deactivate the version information API (FrTSyn_GetVersionInfo). True: version information API activated False: version information API deactivated.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

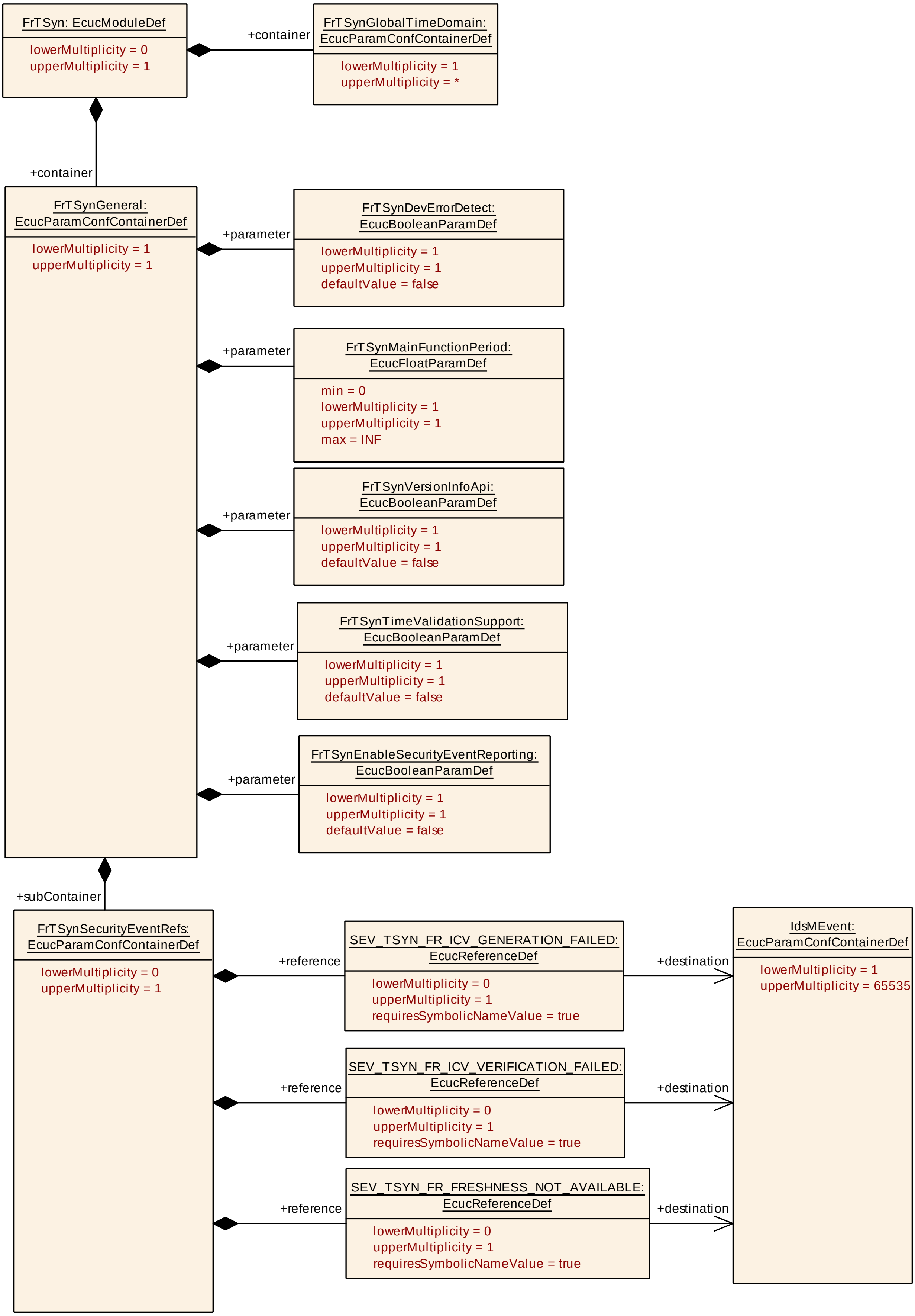


Figure 10.1: FrTSynGeneral

10.2.4 FrTSynSecurityEventRefs

[ECUC_FrTSyn_00063] Definition of EcucParamConfContainerDef FrTSynSecurityEventRefs ⌈

Container Name	FrTSynSecurityEventRefs		
Parent Container	FrTSynGeneral		
Description	Container for the references to IdsMEvent elements representing the security events that the FrTSyn module shall report to the IdsM in case the coresponding security related event occurs (and if FrTSynEnableSecurityEventReporting is set to true). The standardized security events in this container can be extended by vendor-specific security events.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
SEV_TSYN_FR_FRESHNESS_NOT_AVAILABLE	0..1	[ECUC_FrTSyn_00047]
SEV_TSYN_FR_ICV_GENERATION_FAILED	0..1	[ECUC_FrTSyn_00045]
SEV_TSYN_FR_ICV_VERIFICATION_FAILED	0..1	[ECUC_FrTSyn_00046]

No Included Containers

⌋

[ECUC_FrTSyn_00047] Definition of EcucReferenceDef SEV_TSYN_FR_FRESHNESS_NOT_AVAILABLE ⌈

Parameter Name	SEV_TSYN_FR_FRESHNESS_NOT_AVAILABLE		
Parent Container	FrTSynSecurityEventRefs		
Description	FV not available from FVM. Context data provides the respective domain ID.		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00045] Definition of EcucReferenceDef SEV_TSYN_FR_ICV_GENERATION_FAILED ⌈

Parameter Name	SEV_TSYN_FR_ICV_GENERATION_FAILED		
Parent Container	FrTSynSecurityEventRefs		
Description	ICV generation for SYNC message failed. Context data provides the respective domain ID		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00046] Definition of EcucReferenceDef SEV_TSYN_FR_ICV_VERIFICATION_FAILED ⌈

Parameter Name	SEV_TSYN_FR_ICV_VERIFICATION_FAILED		
Parent Container	FrTSynSecurityEventRefs		
Description	ICV verification for SYNC message failed. Context data provides the respective domain ID.		
Multiplicity	0..1		
Type	Symbolic name reference to IdsMEvent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

10.2.5 FrTSynGlobalTimeDomain

[ECUC_FrTSyn_00004] Definition of EcucParamConfContainerDef FrTSynGlobalTimeDomain ⌈

Container Name	FrTSynGlobalTimeDomain
Parent Container	FrTSyn
Description	This represents the existence of a global time domain on Flexray. The FrTSyn module can administrate several global time domains at the same time that in itself form a hierarchy of domains and sub-domains. If the FrTSyn exists it is assumed that at least one global time domain exists.
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrTSynEnableTimeValidation	0..1	[ECUC_FrTSyn_00041]
FrTSynGlobalTimeDomainId	1	[ECUC_FrTSyn_00005]
FrTSynGlobalTimeNetworkSegmentId	0..1	[ECUC_FrTSyn_00042]
FrTSynSynchronizedTimeBaseRef	1	[ECUC_FrTSyn_00018]

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
FrTSynGlobalTimeMaster	0..1	Configuration of the global time master. Each global time domain is required to have exactly one global time master. This master may or may not exist on the configured ECU.
FrTSynGlobalTimeSlave	0..1	This represents the time slave for the enclosing global time domain.
FrTSynGlobalTimeSyncDataIDList	0..1	The DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation and message authentication process.

⌋

[ECUC_FrTSyn_00041] Definition of EcucBooleanParamDef FrTSynEnableTimeValidation ⌈

Parameter Name	FrTSynEnableTimeValidation		
Parent Container	FrTSynGlobalTimeDomain		
Description	Enables/disables time recording for Time Validation for a specific Time Domain.		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: Only valid if FrTSynTimeValidationSupport is TRUE. Value set according to parameter StbMEnableTimeValidation of the referenced Time Base in the StbM.		

⌋

[ECUC_FrTSyn_00005] Definition of EcucIntegerParamDef FrTSynGlobalTimeDomainId ⌈

Parameter Name	FrTSynGlobalTimeDomainId		
Parent Container	FrTSynGlobalTimeDomain		
Description	The global time domain ID.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 15		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00042] Definition of EcucIntegerParamDef FrTSynGlobalTimeNetworkSegmentId ⌈

Parameter Name	FrTSynGlobalTimeNetworkSegmentId		
Parent Container	FrTSynGlobalTimeDomain		
Description	This represents the numerical identifier of the network on system level scope where this Global Time has been communicated on.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00018] Definition of EcucReferenceDef FrTSynSynchronizedTimeBaseRef ⌈

Parameter Name	FrTSynSynchronizedTimeBaseRef
Parent Container	FrTSynGlobalTimeDomain
Description	Mandatory reference to the required synchronized time-base.

▽

△

Multiplicity	1		
Type	Symbolic name reference to StbMSynchronizedTimeBase		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

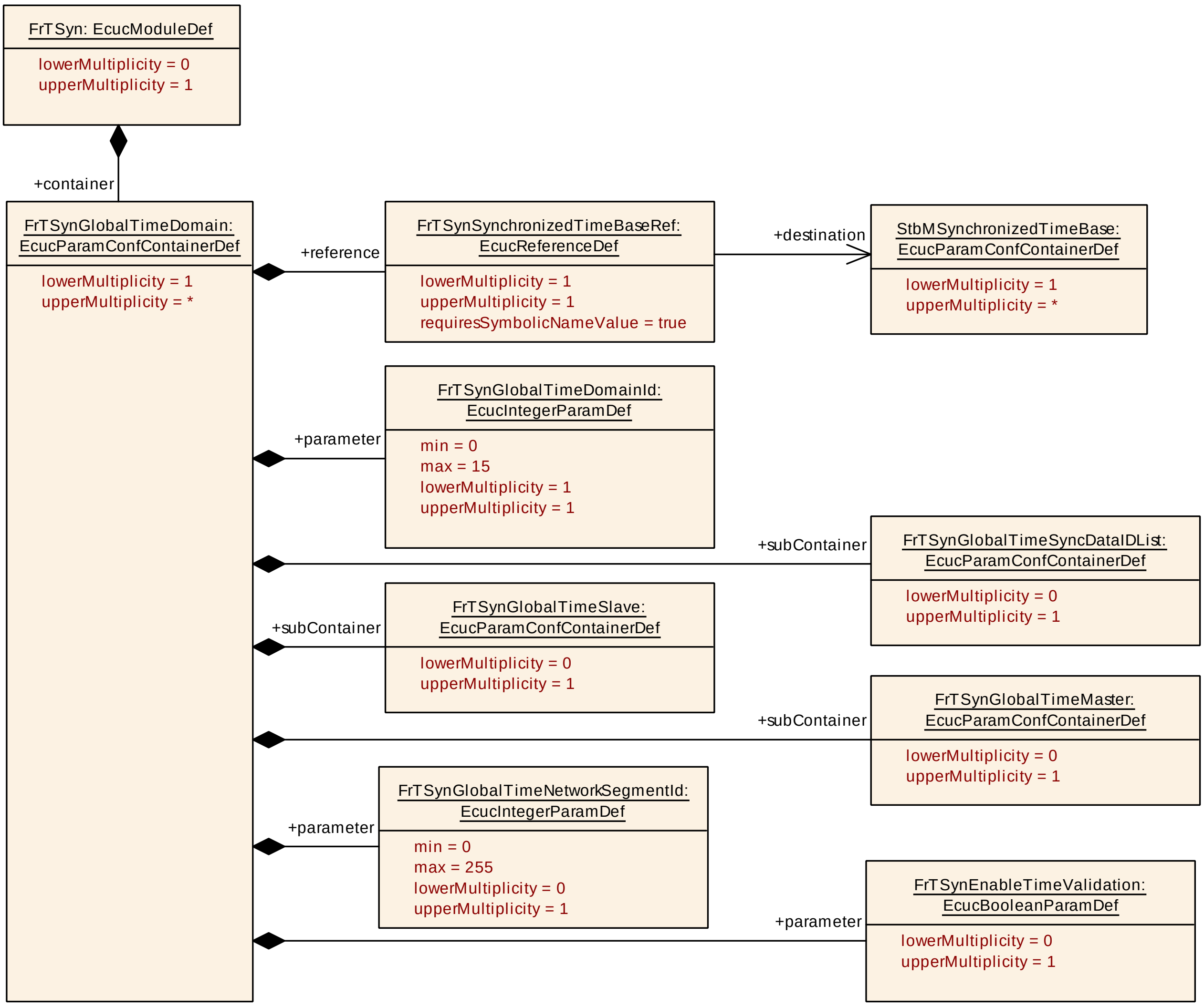


Figure 10.2: FrTSynGlobalTimeDomain

10.2.6 FrTSynGlobalTimeSyncDataIDList

[ECUC_FrTSyn_00023] Definition of EcucParamConfContainerDef FrTSynGlobalTimeSyncDataIDList ⌈

Container Name	FrTSynGlobalTimeSyncDataIDList		
Parent Container	FrTSynGlobalTimeDomain		
Description	The DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation and message authentication process.		
Post-Build Variant Multiplicity	true		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

No Included Parameters

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
FrTSynGlobalTimeSyncDataIDListElement	16	Element of the DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation and message authentication process.

⌋

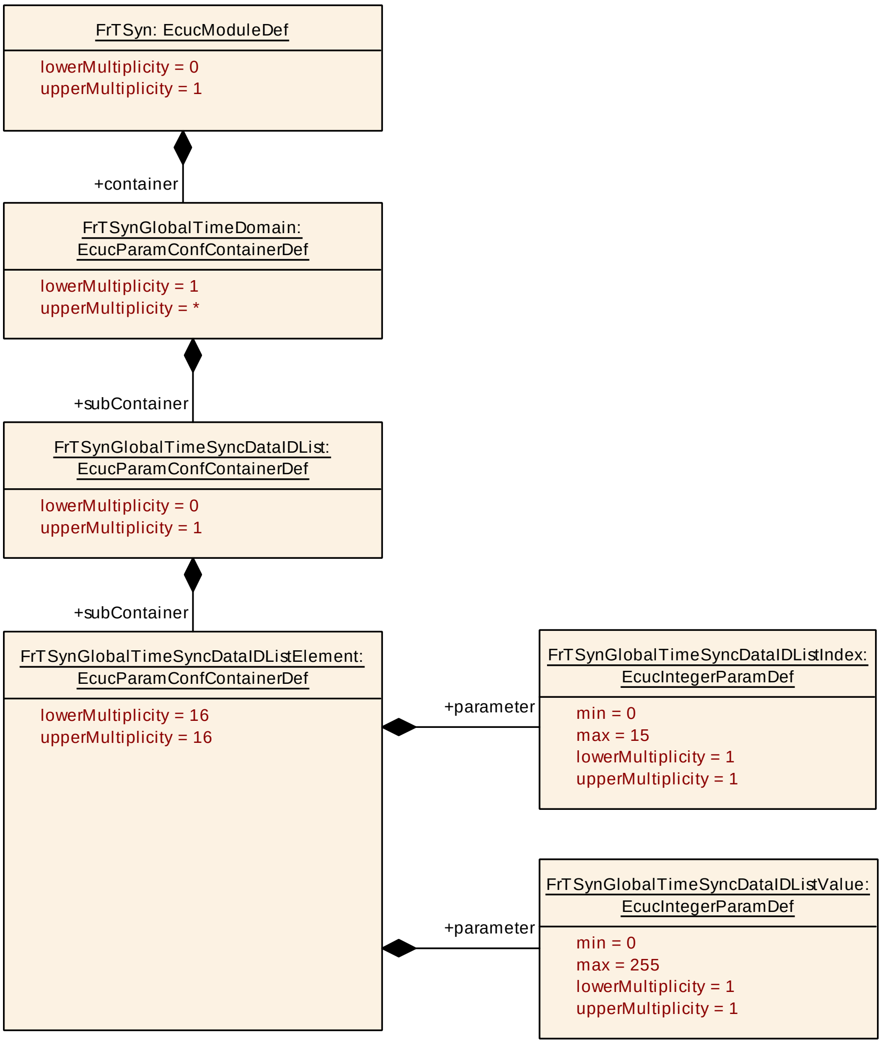


Figure 10.3: FrTSynGlobalTimeSyncDataIDList

10.2.7 FrTSynGlobalTimeSyncDataIDListElement

[ECUC_FrTSyn_00025] Definition of EcucParamConfContainerDef FrTSynGlobalTimeSyncDataIDListElement ⌈

Container Name	FrTSynGlobalTimeSyncDataIDListElement
Parent Container	FrTSynGlobalTimeSyncDataIDList
Description	Element of the DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation and message authentication process.
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrTSynGlobalTimeSyncDataIDListIndex	1	[ECUC_FrTSyn_00026]
FrTSynGlobalTimeSyncDataIDListValue	1	[ECUC_FrTSyn_00027]

No Included Containers

⌋

[ECUC_FrTSyn_00026] Definition of EcucIntegerParamDef FrTSynGlobalTimeSyncDataIDListIndex ⌈

Parameter Name	FrTSynGlobalTimeSyncDataIDListIndex		
Parent Container	FrTSynGlobalTimeSyncDataIDListElement		
Description	Index of the DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation and message authentication process.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 15		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00027] Definition of EcucIntegerParamDef FrTSynGlobalTimeSyncDataIDListValue ⌈

Parameter Name	FrTSynGlobalTimeSyncDataIDListValue		
Parent Container	FrTSynGlobalTimeSyncDataIDListElement		
Description	Value of the DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation and message authentication process.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

10.2.8 FrTSynGlobalTimeMaster

[ECUC_FrTSyn_00006] Definition of EcucParamConfContainerDef FrTSynGlobalTimeMaster ⌈

Container Name	FrTSynGlobalTimeMaster		
Parent Container	FrTSynGlobalTimeDomain		
Description	Configuration of the global time master. Each global time domain is required to have exactly one global time master. This master may or may not exist on the configured ECU.		
Post-Build Variant Multiplicity	true		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrTSynCyclicMsgResumeTime	1	[ECUC_FrTSyn_00032]
FrTSynGlobalTimeDebounceTime	1	[ECUC_FrTSyn_00033]
FrTSynGlobalTimeTxCrcSecured	1	[ECUC_FrTSyn_00013]
FrTSynGlobalTimeTxIcvSecured	1	[ECUC_FrTSyn_00048]
FrTSynGlobalTimeTxPeriod	1	[ECUC_FrTSyn_00014]
FrTSynImmediateTimeSync	1	[ECUC_FrTSyn_00031]

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
FrTSynGlobalTimeMasterPdu	1	This container carries all properties required to configure the PDU sent by the global time master for the given global time domain.
FrTSynGlobalTimeTxIcvGeneration	0..1	This container collects configuration that shall be used for ICV generation.

⌋

[ECUC_FrTSyn_00032] Definition of EcucFloatParamDef FrTSynCyclicMsgResumeTime ⌈

Parameter Name	FrTSynCyclicMsgResumeTime	
Parent Container	FrTSynGlobalTimeMaster	
Description	Defines the time where the 1st regular cycle time based message transmission takes place, after an immediate transmission before. Unit: seconds	
Multiplicity	1	
Type	EcucFloatParamDef	
Range	[0 .. INF[	
Default value	–	
Post-Build Variant Value	true	

▽

△

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00033] Definition of EcucFloatParamDef FrTSynGlobalTimeDebounceTime ⌈

Parameter Name	FrTSynGlobalTimeDebounceTime		
Parent Container	FrTSynGlobalTimeMaster		
Description	This represents the configuration of a TX debounce time for SYNC messages compared to a message before with the same PDU. Unit: seconds		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. 4]		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00013] Definition of EcucEnumerationParamDef FrTSynGlobalTimeTxCrcSecured ⌈

Parameter Name	FrTSynGlobalTimeTxCrcSecured		
Parent Container	FrTSynGlobalTimeMaster		
Description	This represents the configuration of whether or not CRC is supported.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRC_NOT_SUPPORTED	This represents a configuration where CRC is not supported.	
	CRC_SUPPORTED	This represents a configuration where CRC is supported.	
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00048] Definition of EcucEnumerationParamDef FrTSynGlobalTimeTxIcvSecured ⌈

Parameter Name	FrTSynGlobalTimeTxIcvSecured		
Parent Container	FrTSynGlobalTimeMaster		
Description	This parameter controls whether or not ICV generation shall be supported.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	ICV_NOT_SUPPORTED	The Timesync module shall not generate the ICV.	
	ICV_SUPPORTED	The Timesync module shall generate the ICV.	
Default value	ICV_NOT_SUPPORTED		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00014] Definition of EcucFloatParamDef FrTSynGlobalTimeTxPeriod ⌈

Parameter Name	FrTSynGlobalTimeTxPeriod		
Parent Container	FrTSynGlobalTimeMaster		
Description	This represents the TX period. Unit: seconds		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. INF]		
Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00031] Definition of EcucBooleanParamDef FrTSynImmediateTimeSync ⌈

Parameter Name	FrTSynImmediateTimeSync
Parent Container	FrTSynGlobalTimeMaster
Description	Enables/Disables the cyclic polling of StbM_GetTimeBaseUpdateCounter() within FrTSyn_MainFunction().
Multiplicity	1
Type	EcucBooleanParamDef

▽

△

Default value	–		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

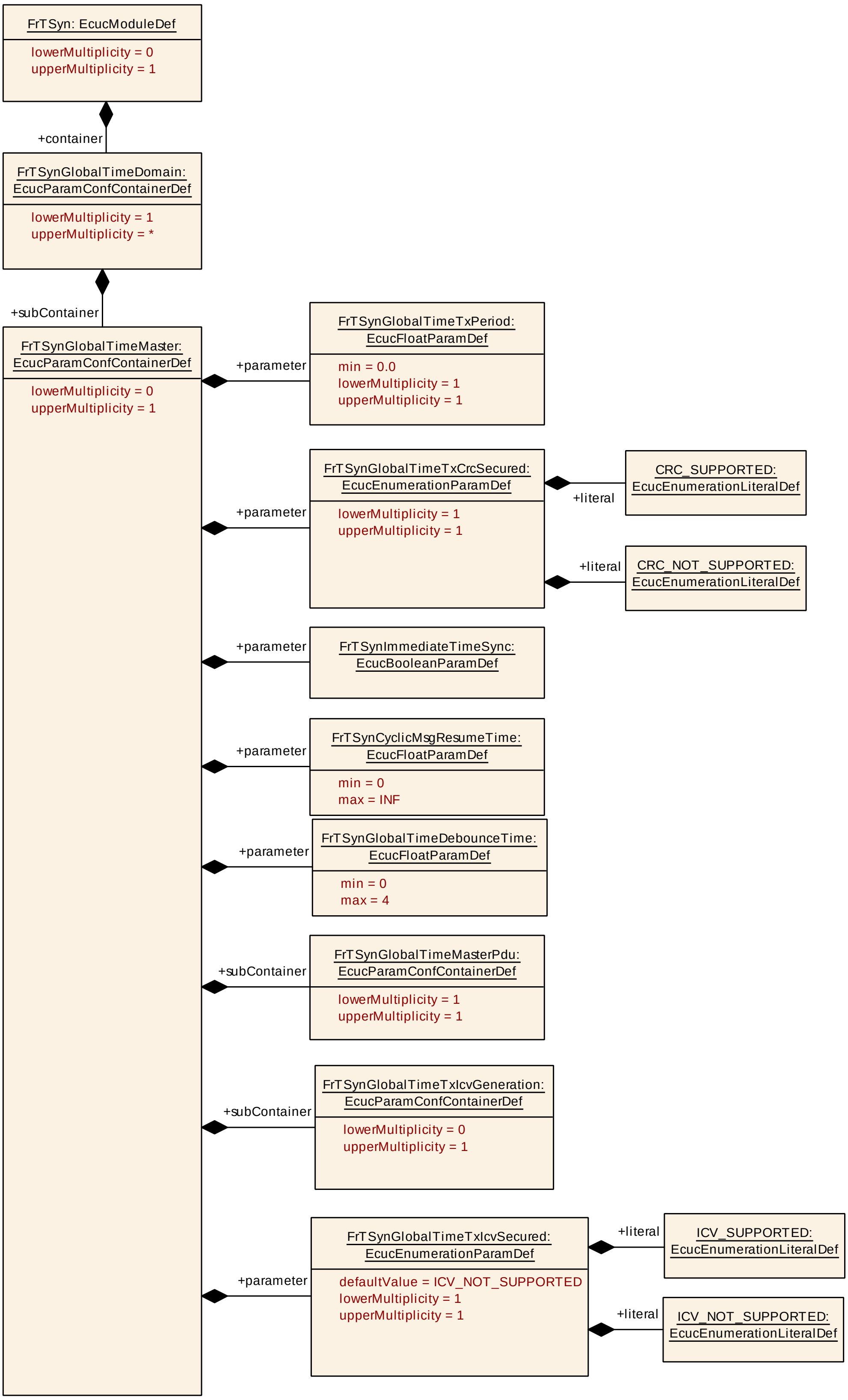

FrTSyn: EcucModuleDef
lowerMultiplicity = 0
upperMultiplicity = 1
+container
FrTSynGlobalTimeDomain: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
+subContainer
FrTSynGlobalTimeMaster: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
FrTSynGlobalTimeTxPeriod: EcucFloatParamDef
min = 0.0
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
FrTSynGlobalTimeTxCrcSecured: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRC_SUPPORTED: EcucEnumerationLiteralDef
+literal
CRC_NOT_SUPPORTED: EcucEnumerationLiteralDef
+parameter
FrTSynImmediateTimeSync: EcucBooleanParamDef
+parameter
FrTSynCyclicMsgResumeTime: EcucFloatParamDef
min = 0
max = INF
+parameter
FrTSynGlobalTimeDebounceTime: EcucFloatParamDef
min = 0
max = 4
+subContainer
FrTSynGlobalTimeMasterPdu: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+subContainer
FrTSynGlobalTimeTxIcvGeneration: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
FrTSynGlobalTimeTxIcvSecured: EcucEnumerationParamDef
defaultValue = ICV_NOT_SUPPORTED
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
ICV_SUPPORTED: EcucEnumerationLiteralDef
+literal
ICV_NOT_SUPPORTED: EcucEnumerationLiteralDef

Figure 10.4: FrTSynGlobalTimeMaster

10.2.9 FrTSynGlobalTimeMasterPdu

[ECUC_FrTSyn_00008] Definition of EcucParamConfContainerDef FrTSynGlobalTimeMasterPdu ⌈

Container Name	FrTSynGlobalTimeMasterPdu
Parent Container	FrTSynGlobalTimeMaster
Description	This container carries all properties required to configure the PDU sent by the global time master for the given global time domain.
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrTSynGlobalTimeMasterHandleId	1	[ECUC_FrTSyn_00007]
FrTSynGlobalTimePduRef	1	[ECUC_FrTSyn_00020]

No Included Containers

⌋

[ECUC_FrTSyn_00007] Definition of EcucIntegerParamDef FrTSynGlobalTimeMasterHandleId ⌈

Parameter Name	FrTSynGlobalTimeMasterHandleId		
Parent Container	FrTSynGlobalTimeMasterPdu		
Description	This represents the handle ID of the PDU that contains the global time information.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local withAuto = true		

⌋

[ECUC_FrTSyn_00020] Definition of EcucReferenceDef FrTSynGlobalTimePduRef ⌈

Parameter Name	FrTSynGlobalTimePduRef		
Parent Container	FrTSynGlobalTimeMasterPdu, FrTSynGlobalTimeSlavePdu		
Description	This represents the reference to the Pdu taken to transmit the global time information. The global time master of a global time domain acts as the sender of the Pdu while all the time slaves are supposed to receive the Pdu.		
Multiplicity	1		
Type	Reference to Pdu		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

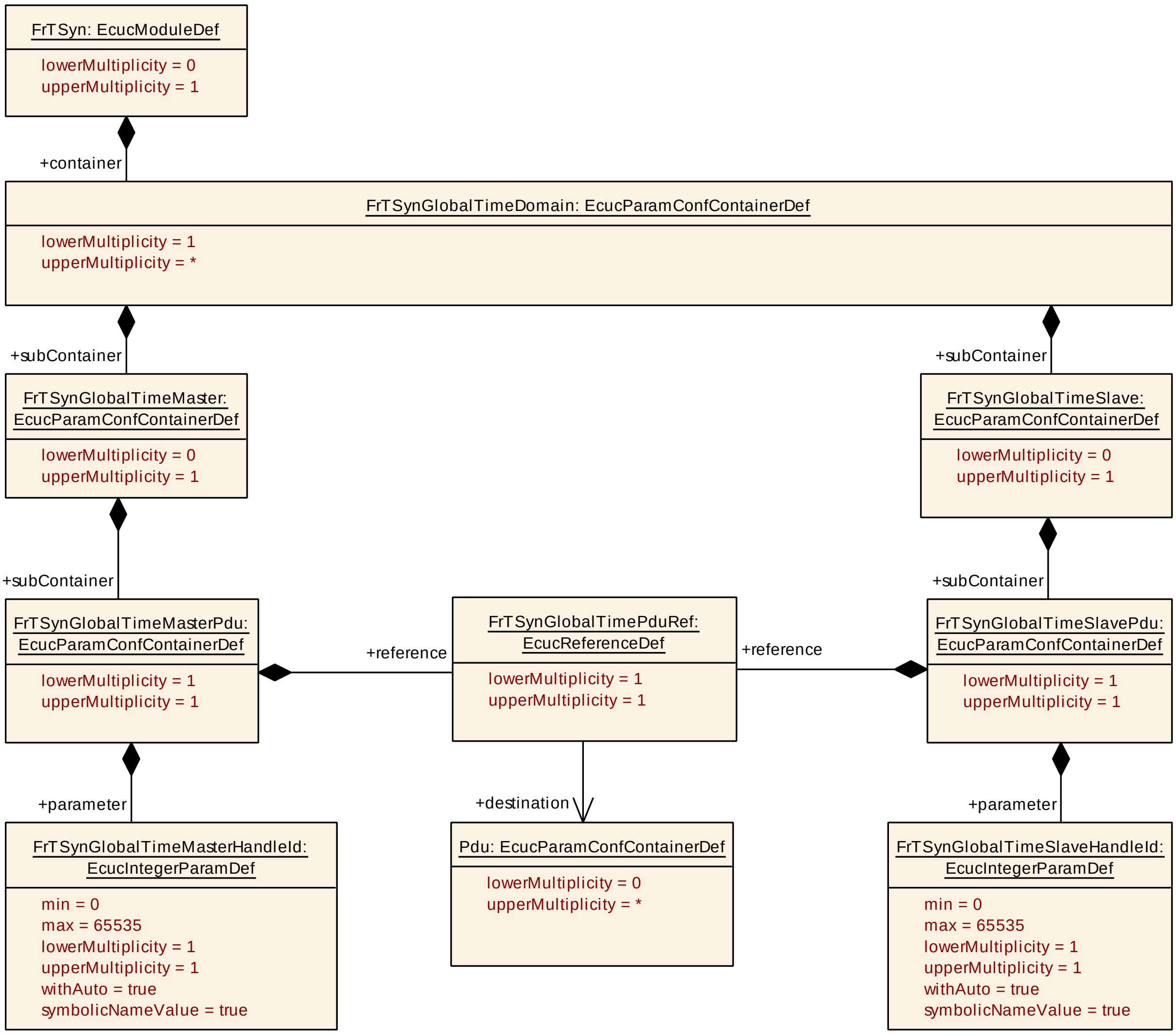


Figure 10.5: FrTSynGlobalTimePdu

10.2.10 FrTSynGlobalTimeTxIcvGeneration

[ECUC_FrTSyn_00049] Definition of EcucParamConfContainerDef FrTSynGlobalTimeTxIcvGeneration ⌈

Container Name	FrTSynGlobalTimeTxIcvGeneration		
Parent Container	FrTSynGlobalTimeMaster		
Description	This container collects configuration that shall be used for ICV generation.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrTSynIcvGenerationBase	1	[ECUC_FrTSyn_00051]
FrTSynIcvGenerationTimeout	1	[ECUC_FrTSyn_00054]
FrTSynIcvTxLength	1	[ECUC_FrTSyn_00052]
FrTSynTxAuthenticationBuildAttempts	1	[ECUC_FrTSyn_00065]
FrTSynIcvGenerationFvIdRef	0..1	[ECUC_FrTSyn_00050]
FrTSynIcvGenerationJobRef	1	[ECUC_FrTSyn_00053]

No Included Containers

⌋

[ECUC_FrTSyn_00051] Definition of EcucEnumerationParamDef FrTSynIcvGenerationBase ⌈

Parameter Name	FrTSynIcvGenerationBase		
Parent Container	FrTSynGlobalTimeTxIcvGeneration		
Description	Symmetric or asymmetric cryptography selection for the ICV generation		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	ICV_MAC	Symmetric cryptography selection for the ICV generation.	
	ICV_SIGNATURE	Asymmetric cryptography selection for the ICV generation.	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00054] Definition of EcucFloatParamDef FrTSynIcvGenerationTimeout ⌈

Parameter Name	FrTSynIcvGenerationTimeout		
Parent Container	FrTSynGlobalTimeTxIcvGeneration		
Description	Timeout of ICV generation (respective CSM job completion in asynchronous behavior). A value of 0 disables the ICV timeout monitoring. Unit: Seconds		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[0 .. INF[		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00052] Definition of EcucIntegerParamDef FrTSynIcvTxLength ⌈

Parameter Name	FrTSynIcvTxLength		
Parent Container	FrTSynGlobalTimeTxIcvGeneration		
Description	Length of ICV to be transmitted within Sync Message on the bus.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 236		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00065] Definition of EcucIntegerParamDef FrTSynTxAuthenticationBuildAttempts ⌈

Parameter Name	FrTSynTxAuthenticationBuildAttempts	
Parent Container	FrTSynGlobalTimeTxIcvGeneration	
Description	This parameter specifies the number of authentication build attempts that are to be carried out when the generation of the ICV failed for a given SYNC message. If zero is set, then only one ICV generation attempt is done.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 65535	

▽

△

Default value	0		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00050] Definition of EcucReferenceDef FrTSynIcvGenerationFvIdRef ⌈

Parameter Name	FrTSynIcvGenerationFvIdRef		
Parent Container	FrTSynGlobalTimeTxIcvGeneration		
Description	This represents the reference to the FV taken to generate the ICV generation.		
Multiplicity	0..1		
Type	Symbolic name reference to StbMFreshnessValue		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00053] Definition of EcucReferenceDef FrTSynIcvGenerationJobRef ⌈

Parameter Name	FrTSynIcvGenerationJobRef		
Parent Container	FrTSynGlobalTimeTxIcvGeneration		
Description	This represents the reference to the CSM job to fetch the CSM job ID.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

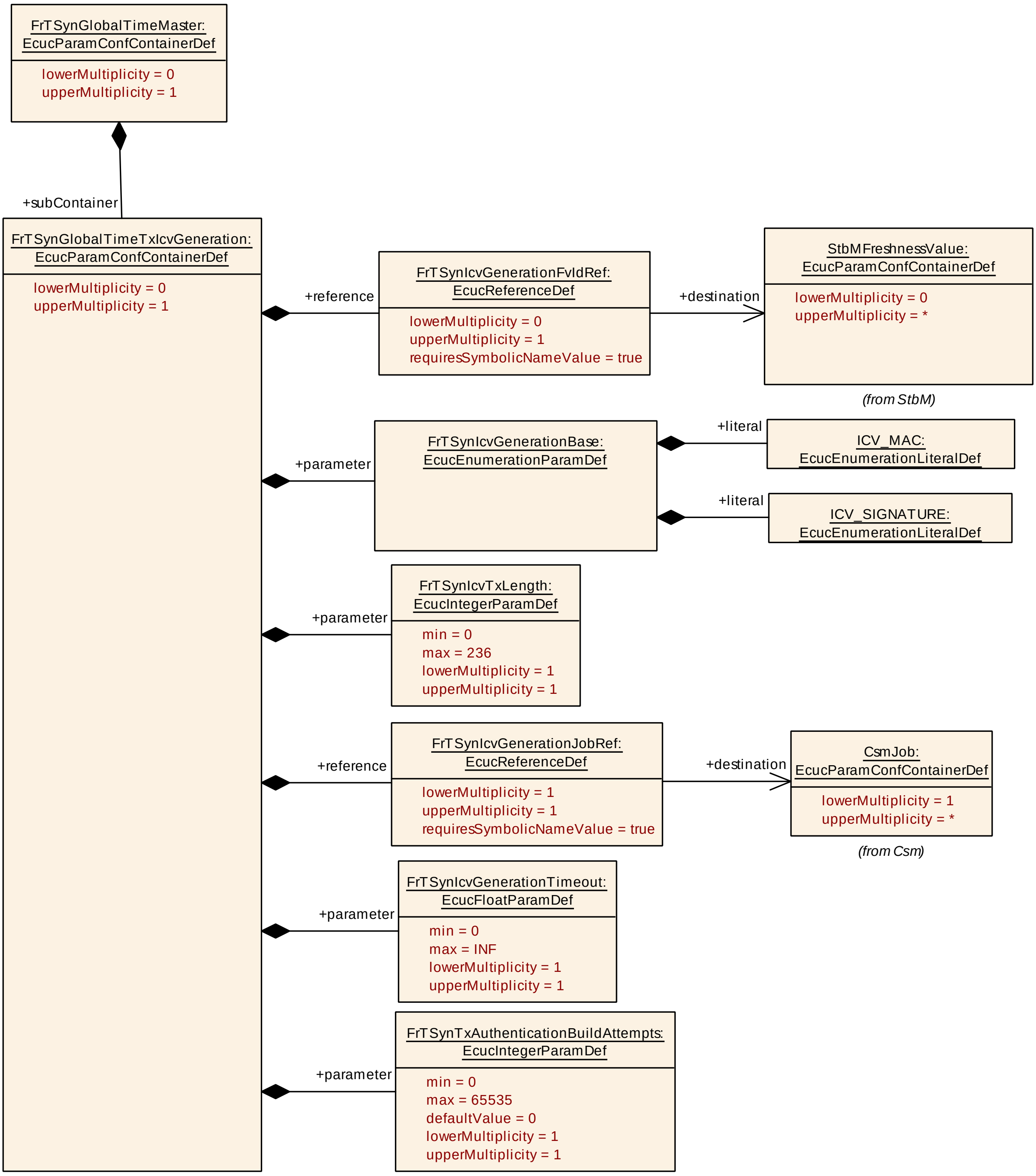

FrTSynGlobalTimeMaster: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
FrTSynGlobalTimeTxIcvGeneration: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+reference
FrTSynIcvGenerationFvIdRef: EcucReferenceDef
lowerMultiplicity = 0
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
StbMFreshnessValue: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = *
(from StbM)
+parameter
FrTSynIcvGenerationBase: EcucEnumerationParamDef
+literal
ICV_MAC: EcucEnumerationLiteralDef
+literal
ICV_SIGNATURE: EcucEnumerationLiteralDef
+parameter
FrTSynIcvTxLength: EcucIntegerParamDef
min = 0
max = 236
lowerMultiplicity = 1
upperMultiplicity = 1
+reference
FrTSynIcvGenerationJobRef: EcucReferenceDef
lowerMultiplicity = 1
upperMultiplicity = 1
requiresSymbolicNameValue = true
+destination
CsmJob: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
(from Csm)
+parameter
FrTSynIcvGenerationTimeout: EcucFloatParamDef
min = 0
max = INF
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
FrTSynTxAuthenticationBuildAttempts: EcucIntegerParamDef
min = 0
max = 65535
defaultValue = 0
lowerMultiplicity = 1
upperMultiplicity = 1

Figure 10.6: FrTSynGlobalTimeTxIcvGeneration

10.2.11 FrTSynGlobalTimeSlave

[ECUC_FrTSyn_00010] Definition of EcucParamConfContainerDef FrTSynGlobalTimeSlave ⌈

Container Name	FrTSynGlobalTimeSlave		
Parent Container	FrTSynGlobalTimeDomain		
Description	This represents the time slave for the enclosing global time domain.		
Post-Build Variant Multiplicity	true		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrTSynGlobalTimeSequenceCounterHysteresis	1	[ECUC_FrTSyn_00043]
FrTSynGlobalTimeSequenceCounterJumpWidth	1	[ECUC_FrTSyn_00022]
FrTSynRxCrcValidated	1	[ECUC_FrTSyn_00017]
FrTSynRxIcvVerificationType	1	[ECUC_FrTSyn_00055]

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
FrTSynGlobalTimeRxIcvVerification	0..1	This container collects configuration required for ICV verification.
FrTSynGlobalTimeSlavePdu	1	This container carries all properties required to configure the PDU received by the time slave for the given global time domain. Supported MetaData entry: TIMETUPLE_TYPE_PTR

⌋

[ECUC_FrTSyn_00043] Definition of EcucIntegerParamDef FrTSynGlobalTimeSequenceCounterHysteresis ⌈

Parameter Name	FrTSynGlobalTimeSequenceCounterHysteresis		
Parent Container	FrTSynGlobalTimeSlave		
Description	FrTSynGlobalTimeSequenceCounterHysteresis specifies the number of consecutive valid SYNC messages that are required by the Time Slave while being in Timeout state until a Time Tuple is forwarded to the StbM.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 15		
Default value	0		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00022] Definition of EcucIntegerParamDef FrTSynGlobalTimeSequenceCounterJumpWidth ⌈

Parameter Name	FrTSynGlobalTimeSequenceCounterJumpWidth		
Parent Container	FrTSynGlobalTimeSlave		
Description	The SequenceCounterJumpWidth specifies the maximum allowed gap of the Sequence Counter between two SYNC messages.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 15		
Default value	0		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00017] Definition of EcucEnumerationParamDef FrTSynRxCrcValidated ⌈

Parameter Name	FrTSynRxCrcValidated		
Parent Container	FrTSynGlobalTimeSlave		
Description	This parameter controls whether or not CRC validation shall be supported.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	CRC_IGNORED	The Timesync module accepts Time Synchronization messages, which are CRC secured (without actually validating the CRC) and those, which are not CRC secured. That means, the Timesync module ignores the CRC.	
	CRC_NOT_VALIDATED	The Timesync module accepts only Time Synchronization messages, which are not CRC secured. All other Time Synchronization messages are ignored.	
	CRC_OPTIONAL	The Timesync module accepts only Time Synchronization messages which are not CRC secured and Time Synchronization messages which are CRC secured and have the correct CRC. All other Time Synchronization messages are ignored.	
	CRC_VALIDATED	The Timesync module accepts only Time Synchronization messages, which are CRC secured and have the correct CRC. All other Time Synchronization messages are ignored.	
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00055] Definition of EcucEnumerationParamDef FrTSynRxIcvVerificationType ⌈

Parameter Name	FrTSynRxIcvVerificationType		
Parent Container	FrTSynGlobalTimeSlave		
Description	This parameter controls whether or not ICV verification shall be supported.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	ICV_IGNORED	The Timesync module accepts Time Synchronization messages, which are ICV secured (without actually validating the ICV) and those which are not ICV secured. That means, the Timesync module ignores the ICV.	
	ICV_NOT_VERIFIED	The Timesync module accepts only Time Synchronization messages, which are not ICV secured. All other Time Synchronization messages are ignored.	
	ICV_OPTIONAL	The Timesync module accepts only Time Synchronization messages which are not ICV secured and Time Synchronization messages which are ICV secured and have the correct ICV. All other Time Synchronization messages are ignored.	
	ICV_VERIFIED	The Timesync module accepts only Time Synchronization messages, which are ICV secured and have the correct ICV. All other Time Synchronization messages are ignored.	
Default value	ICV_IGNORED		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

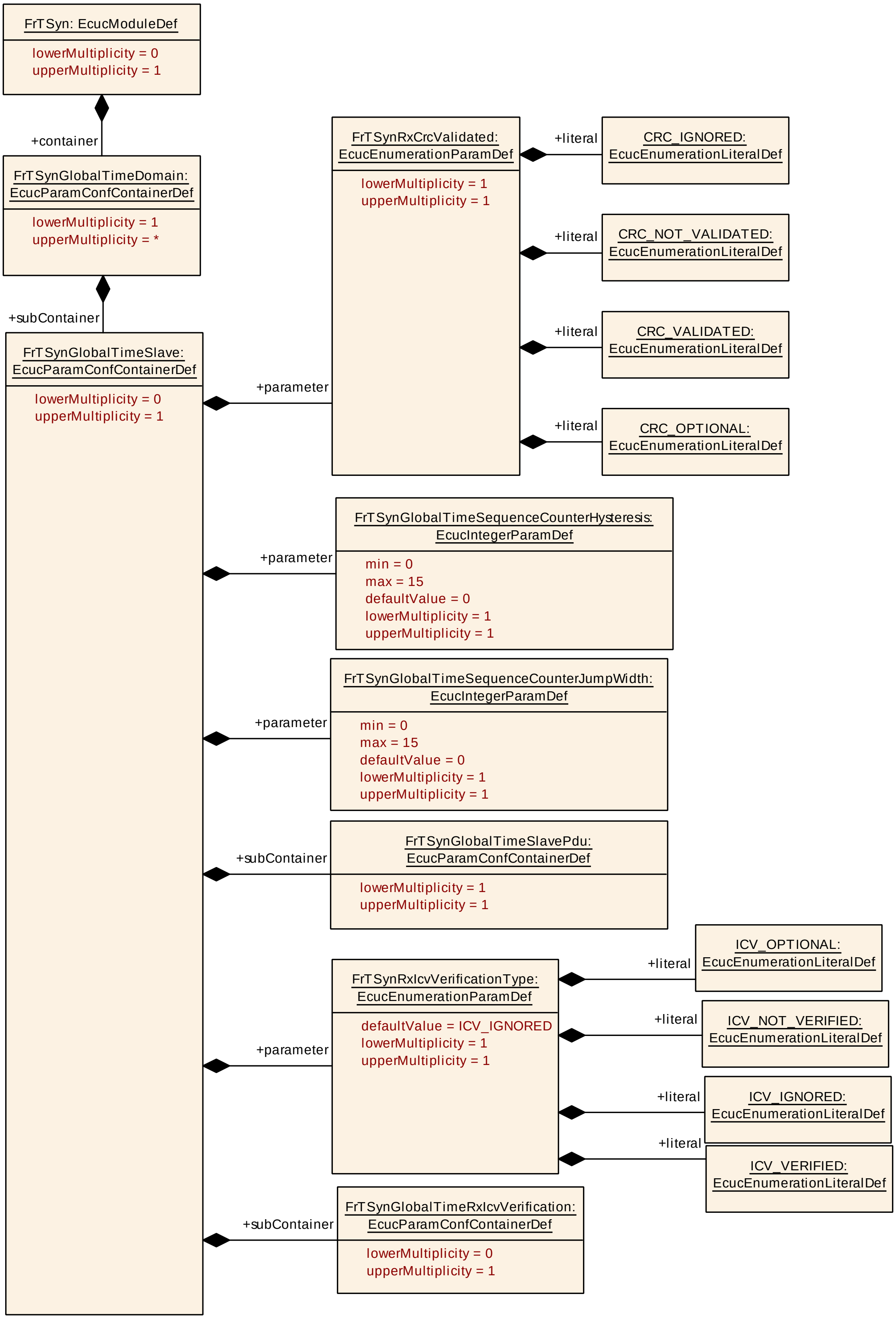

FrTSyn: EcucModuleDef
lowerMultiplicity = 0
upperMultiplicity = 1
+container
FrTSynGlobalTimeDomain: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
+subContainer
FrTSynGlobalTimeSlave: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
FrTSynRxCrcValidated: EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRC_IGNORED: EcucEnumerationLiteralDef
+literal
CRC_NOT_VALIDATED: EcucEnumerationLiteralDef
+literal
CRC_VALIDATED: EcucEnumerationLiteralDef
+literal
CRC_OPTIONAL: EcucEnumerationLiteralDef
+parameter
FrTSynGlobalTimeSequenceCounterHysteresis: EcucIntegerParamDef
min = 0
max = 15
defaultValue = 0
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
FrTSynGlobalTimeSequenceCounterJumpWidth: EcucIntegerParamDef
min = 0
max = 15
defaultValue = 0
lowerMultiplicity = 1
upperMultiplicity = 1
+subContainer
FrTSynGlobalTimeSlavePdu: EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = 1
+parameter
FrTSynRxIcvVerificationType: EcucEnumerationParamDef
defaultValue = ICV_IGNORED
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
ICV_OPTIONAL: EcucEnumerationLiteralDef
+literal
ICV_NOT_VERIFIED: EcucEnumerationLiteralDef
+literal
ICV_IGNORED: EcucEnumerationLiteralDef
+literal
ICV_VERIFIED: EcucEnumerationLiteralDef
+subContainer
FrTSynGlobalTimeRxIcvVerification: EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1

Figure 10.7: FrTSynGlobalTimeSlave

10.2.12 FrTSynGlobalTimeSlavePdu

[ECUC_FrTSyn_00012] Definition of EcucParamConfContainerDef FrTSynGlobalTimeSlavePdu ⌈

Container Name	FrTSynGlobalTimeSlavePdu
Parent Container	FrTSynGlobalTimeSlave
Description	This container carries all properties required to configure the PDU received by the time slave for the given global time domain. Supported MetaData entry: TIMETUPLE_TYPE_PTR
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrTSynGlobalTimeSlaveHandleId	1	[ECUC_FrTSyn_00011]
FrTSynGlobalTimePduRef	1	[ECUC_FrTSyn_00020]

No Included Containers

⌋

[ECUC_FrTSyn_00011] Definition of EcucIntegerParamDef FrTSynGlobalTimeSlaveHandleId ⌈

Parameter Name	FrTSynGlobalTimeSlaveHandleId		
Parent Container	FrTSynGlobalTimeSlavePdu		
Description	This represents the handle ID of the PDU that contains the global time information.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local withAuto = true		

⌋

For parameter table [ECUC_FrTSyn_00020] FrTSynGlobalTimePduRef, see definition below container FrTSynGlobalTimeMasterPdu.

10.2.13 FrTSynGlobalTimeRxIcvVerification

[ECUC_FrTSyn_00056] Definition of EcucParamConfContainerDef FrTSynGlobalTimeRxIcvVerification ⌈

Container Name	FrTSynGlobalTimeRxIcvVerification		
Parent Container	FrTSynGlobalTimeSlave		
Description	This container collects configuration required for ICV verification.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
FrTSynIcvRxLength	1	[ECUC_FrTSyn_00059]
FrTSynIcvVerificationAttempts	1	[ECUC_FrTSyn_00062]
FrTSynIcvVerificationBase	1	[ECUC_FrTSyn_00058]
FrTSynIcvVerificationTimeout	1	[ECUC_FrTSyn_00061]
FrTSynRxAuthenticationBuildAttempts	1	[ECUC_FrTSyn_00064]
FrTSynIcvVerificationFvIdRef	0..1	[ECUC_FrTSyn_00057]
FrTSynIcvVerificationJobRef	1	[ECUC_FrTSyn_00060]

No Included Containers

⌋

[ECUC_FrTSyn_00059] Definition of EcucIntegerParamDef FrTSynIcvRxLength ⌈

Parameter Name	FrTSynIcvRxLength		
Parent Container	FrTSynGlobalTimeRxIcvVerification		
Description	Length of ICV to be used for verification of received ICV within Sync Message.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 236		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00062] Definition of EcucIntegerParamDef FrTSynIcvVerificationAttempts ⌈

Parameter Name	FrTSynIcvVerificationAttempts		
Parent Container	FrTSynGlobalTimeRxIcvVerification		
Description	This parameter specifies the number of ICV verification attempts that are to be carried out when the verification of the ICV failed for a given secured SYNC message. If zero is set, then only one ICV verification attempt is done.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	0		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00058] Definition of EcucEnumerationParamDef FrTSynIcvVerificationBase ⌈

Parameter Name	FrTSynIcvVerificationBase		
Parent Container	FrTSynGlobalTimeRxIcvVerification		
Description	Symmetric or asymmetric cryptography selection for the ICV verification.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	ICV_MAC	Symmetric cryptography selection for the ICV verification.	
	ICV_SIGNATURE	Asymmetric cryptography selection for the ICV verification.	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00061] Definition of EcucFloatParamDef FrTSynIcvVerificationTimeout ⌈

Parameter Name	FrTSynIcvVerificationTimeout
Parent Container	FrTSynGlobalTimeRxIcvVerification
Description	Timeout of ICV verification (respective CSM job completion in asynchronous behaviour). A value of 0 disables the ICV timeout monitoring. Unit: Seconds
Multiplicity	1

▽

△

Type	EcucFloatParamDef		
Range	[0 .. INF[		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00064] Definition of EcucIntegerParamDef FrTSynRxAuthenticationBuildAttempts ⌈

Parameter Name	FrTSynRxAuthenticationBuildAttempts		
Parent Container	FrTSynGlobalTimeRxIcvVerification		
Description	This parameter specifies the number of authentication build attempts that are to be carried out when the verification of the ICV failed for a given SYNC message. If zero is set, then only one ICV verification attempt is done.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	0		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00057] Definition of EcucReferenceDef FrTSynIcvVerificationFvIdRef ⌈

Parameter Name	FrTSynIcvVerificationFvIdRef		
Parent Container	FrTSynGlobalTimeRxIcvVerification		
Description	This represents the reference to the FV taken to generate the ICV generation.		
Multiplicity	0..1		
Type	Symbolic name reference to StbMFreshnessValue		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	

▽

△

	Post-build time	–	
Scope / Dependency	scope: local		

⌋

[ECUC_FrTSyn_00060] Definition of EcucReferenceDef FrTSynIcvVerificationJobRef ⌈

Parameter Name	FrTSynIcvVerificationJobRef		
Parent Container	FrTSynGlobalTimeRxIcvVerification		
Description	This represents the reference to the CSM job to fetch the CSM job ID.		
Multiplicity	1		
Type	Symbolic name reference to CsmJob		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

⌋

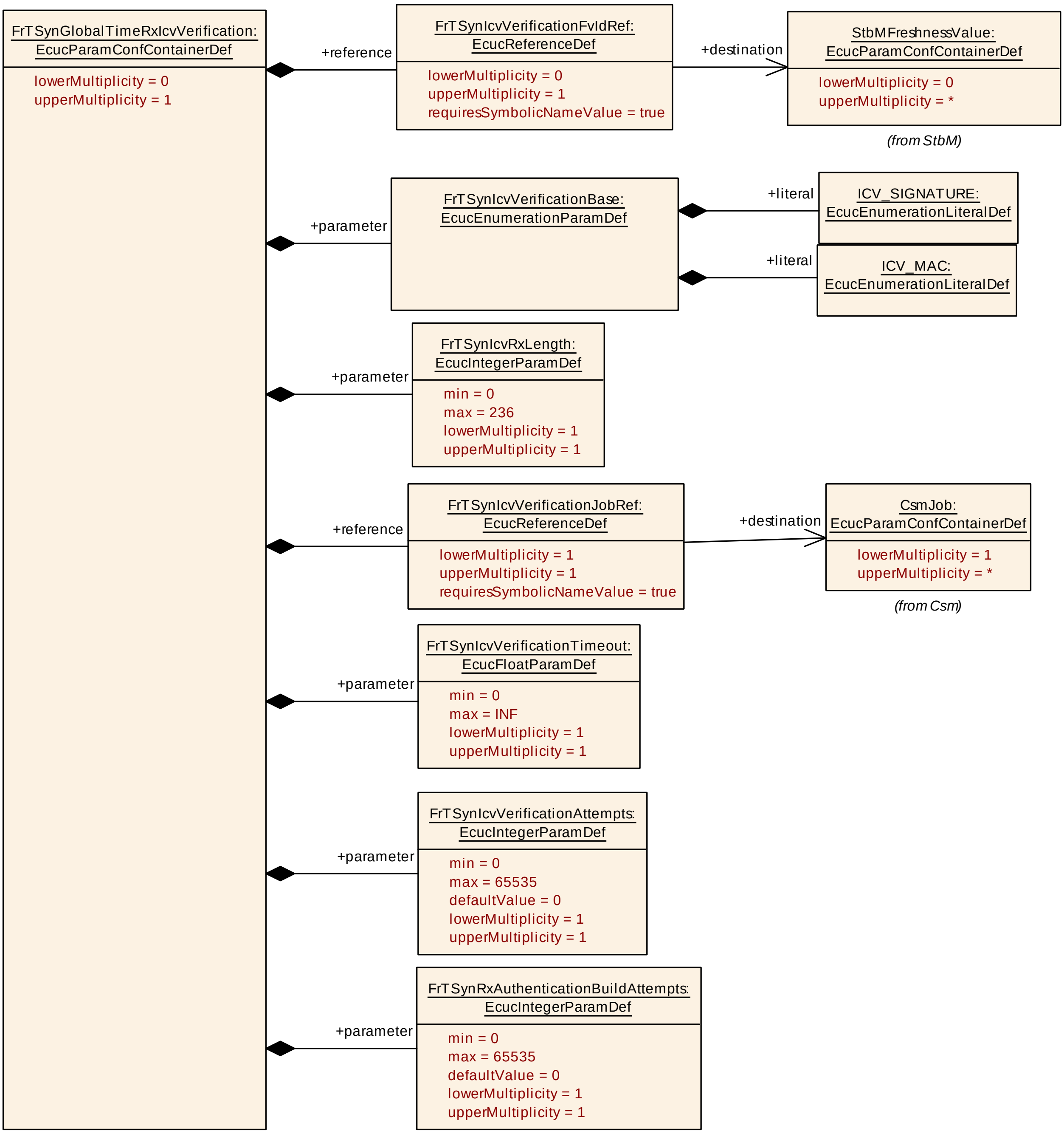


Figure 10.8: FrTSynGlobalTimeRxIcvVerification

10.3 Constraints

[SWS_FrTSyn_CONSTR_00001] ⌈If the CSM job used to verify the ICV is configured in synchronous behaviour, then the FrTSynIcvVerificationTimeout shall be set to 0.⌋

[SWS_FrTSyn_CONSTR_00002]

Status: DRAFT

⌈If the CSM job used to generate the ICV is configured in synchronous behavior, then the FrTSynIcvGenerationTimeout shall be set to 0.⌋

[SWS_FrTSyn_CONSTR_00003] FrTSynGlobalTimeMasterPdu and FrTSynGlobalTimeSlavePdu constraint for keeping the local buffer

Status: DRAFT

⌈The configuration of FrTSynGlobalTimeMasterPdu and FrTSynGlobalTimeSlavePdu shall refer to PDUs where KeepLocalPduBuffer is set to FALSE.

Otherwise the configuration shall be rejected as invalid.⌋

10.4 Published Information

For details refer to the chapter 10.3 “Published Information” in the SWS BSW General [3].

A Not applicable requirements

[SWS_FrTSyn_NA_00999]

Upstream requirements: RS_TS_00002, RS_TS_00005, RS_TS_00006, RS_TS_00007, RS_TS_00008, RS_TS_00009, RS_TS_00010, RS_TS_00011, RS_TS_00014, RS_TS_00015, RS_TS_00016, RS_TS_00017, RS_TS_00018, RS_TS_00019, RS_TS_00021, RS_TS_00024, RS_TS_00025, RS_TS_00026, RS_TS_00027, RS_TS_00029, RS_TS_00030, RS_TS_00031, RS_TS_00032, RS_TS_00033, RS_TS_00035, RS_TS_00036, RS_TS_00037, RS_TS_00038, RS_TS_00039, RS_TS_20031, RS_TS_20032, RS_TS_20033, RS_TS_20034, RS_TS_20035, RS_TS_20037, RS_TS_20038, RS_TS_20047, RS_TS_20048, RS_TS_20051, RS_TS_20052, RS_TS_20053, RS_TS_20054, RS_TS_20058, RS_TS_20059, RS_TS_20060, RS_TS_20061, RS_TS_20062, RS_TS_20066, RS_TS_20068, RS_TS_20069, RS_TS_20070

⌈These requirements on Time Synchronization from the RS Time Synchronization [1] are not applicable to FrTSyn, because they refer either to network types other than FlexRay or to the Time Base Manager module⌋

B Change history of AUTOSAR traceable items

Please note that the lists in this chapter also include constraints and specification items that have been removed from the specification in a later version. These constraints and specification items do not appear as hyperlinks in the document.

B.1 Traceable item history of this document according to AUTOSAR Release R24-11

B.1.1 Added Specification Items in R24-11

[SWS_FrTSyn_00173] [SWS_FrTSyn_92000] [SWS_FrTSyn_92001] [SWS_FrTSyn_92002]

B.1.2 Changed Specification Items in R24-11

[ECUC_FrTSyn_00010] [ECUC_FrTSyn_00012] [SWS_FrTSyn_00009] [SWS_FrTSyn_00030] [SWS_FrTSyn_00035] [SWS_FrTSyn_00056] [SWS_FrTSyn_00075] [SWS_FrTSyn_00090] [SWS_FrTSyn_00091] [SWS_FrTSyn_00094] [SWS_FrTSyn_00110] [SWS_FrTSyn_00111] [SWS_FrTSyn_00116] [SWS_FrTSyn_00117] [SWS_FrTSyn_00118] [SWS_FrTSyn_00119] [SWS_FrTSyn_00125] [SWS_FrTSyn_00126] [SWS_FrTSyn_00127] [SWS_FrTSyn_00136] [SWS_FrTSyn_00137] [SWS_FrTSyn_00138] [SWS_FrTSyn_00139] [SWS_FrTSyn_00142] [SWS_FrTSyn_00146] [SWS_FrTSyn_00149] [SWS_FrTSyn_00153] [SWS_FrTSyn_00154] [SWS_FrTSyn_00162] [SWS_FrTSyn_00163] [SWS_FrTSyn_00164] [SWS_FrTSyn_00165] [SWS_FrTSyn_00166] [SWS_FrTSyn_00167] [SWS_FrTSyn_00168] [SWS_FrTSyn_00170]

B.1.3 Deleted Specification Items in R24-11

[ECUC_FrTSyn_00024] [ECUC_FrTSyn_00028] [ECUC_FrTSyn_00029] [ECUC_FrTSyn_00030] [SWS_FrTSyn_00022] [SWS_FrTSyn_00023] [SWS_FrTSyn_00025] [SWS_FrTSyn_00029] [SWS_FrTSyn_00042] [SWS_FrTSyn_00043] [SWS_FrTSyn_00044] [SWS_FrTSyn_00045] [SWS_FrTSyn_00047] [SWS_FrTSyn_00048] [SWS_FrTSyn_00049] [SWS_FrTSyn_00079] [SWS_FrTSyn_00080] [SWS_FrTSyn_00082] [SWS_FrTSyn_00084] [SWS_FrTSyn_00089] [SWS_FrTSyn_00102] [SWS_FrTSyn_00104] [SWS_FrTSyn_00108] [SWS_FrTSyn_00109] [SWS_FrTSyn_00113] [SWS_FrTSyn_00132] [SWS_FrTSyn_00133] [SWS_FrTSyn_00134] [SWS_FrTSyn_00135]

B.1.4 Added Constraints in R24-11

[SWS_FrTSyn_CONSTR_00003]

B.1.5 Changed Constraints in R24-11

none

B.1.6 Deleted Constraints in R24-11

none

B.2 Traceable item history of this document according to AUTOSAR Release R23-11

B.2.1 Added Specification Items in R23-11

[SWS_FrTSyn_00153] [SWS_FrTSyn_00154] [SWS_FrTSyn_00155] [SWS_FrTSyn_00156] [SWS_FrTSyn_00157] [SWS_FrTSyn_00158] [SWS_FrTSyn_00159] [SWS_FrTSyn_00160] [SWS_FrTSyn_00161] [SWS_FrTSyn_00162] [SWS_FrTSyn_00163] [SWS_FrTSyn_00164] [SWS_FrTSyn_00165] [SWS_FrTSyn_00166] [SWS_FrTSyn_00167] [SWS_FrTSyn_00168] [SWS_FrTSyn_00169] [SWS_FrTSyn_00170] [SWS_FrTSyn_00171] [SWS_FrTSyn_00172]

B.2.2 Changed Specification Items in R23-11

[SWS_FrTSyn_00014] [SWS_FrTSyn_00015] [SWS_FrTSyn_00019] [SWS_FrTSyn_00021] [SWS_FrTSyn_00023] [SWS_FrTSyn_00025] [SWS_FrTSyn_00048] [SWS_FrTSyn_00049] [SWS_FrTSyn_00056] [SWS_FrTSyn_00060] [SWS_FrTSyn_00079] [SWS_FrTSyn_00080] [SWS_FrTSyn_00084] [SWS_FrTSyn_00085] [SWS_FrTSyn_00086] [SWS_FrTSyn_00089] [SWS_FrTSyn_00090] [SWS_FrTSyn_00091] [SWS_FrTSyn_00093] [SWS_FrTSyn_00102] [SWS_FrTSyn_00103] [SWS_FrTSyn_00104] [SWS_FrTSyn_00106] [SWS_FrTSyn_00107] [SWS_FrTSyn_00108] [SWS_FrTSyn_00109] [SWS_FrTSyn_00112] [SWS_FrTSyn_00113] [SWS_FrTSyn_00119] [SWS_FrTSyn_00125] [SWS_FrTSyn_00127] [SWS_FrTSyn_00142] [SWS_FrTSyn_00145] [SWS_FrTSyn_00146] [SWS_FrTSyn_00149]

B.2.3 Deleted Specification Items in R23-11

[SWS_FrTSyn_00141]

B.2.4 Added Constraints in R23-11

[SWS_FrTSyn_CONSTR_00002]

B.2.5 Changed Constraints in R23-11

none

B.2.6 Deleted Constraints in R23-11

none

B.3 Traceable item history of this document according to AUTOSAR Release R22-11

B.3.1 Added Specification Items in R22-11

[SWS_FrTSyn_00103] [SWS_FrTSyn_00104] [SWS_FrTSyn_00105] [SWS_FrTSyn_00106] [SWS_FrTSyn_00107] [SWS_FrTSyn_00108] [SWS_FrTSyn_00109] [SWS_FrTSyn_00110] [SWS_FrTSyn_00111] [SWS_FrTSyn_00112] [SWS_FrTSyn_00113] [SWS_FrTSyn_00114] [SWS_FrTSyn_00115] [SWS_FrTSyn_00116] [SWS_FrTSyn_00117] [SWS_FrTSyn_00118] [SWS_FrTSyn_00119] [SWS_FrTSyn_00120] [SWS_FrTSyn_00121] [SWS_FrTSyn_00122] [SWS_FrTSyn_00123] [SWS_FrTSyn_00124] [SWS_FrTSyn_00125] [SWS_FrTSyn_00126] [SWS_FrTSyn_00127] [SWS_FrTSyn_00128] [SWS_FrTSyn_00129] [SWS_FrTSyn_00130] [SWS_FrTSyn_00131] [SWS_FrTSyn_00132] [SWS_FrTSyn_00133] [SWS_FrTSyn_00134] [SWS_FrTSyn_00135] [SWS_FrTSyn_00136] [SWS_FrTSyn_00137] [SWS_FrTSyn_00138] [SWS_FrTSyn_00139] [SWS_FrTSyn_00140] [SWS_FrTSyn_00141] [SWS_FrTSyn_00142] [SWS_FrTSyn_00143] [SWS_FrTSyn_00144] [SWS_FrTSyn_00145] [SWS_FrTSyn_00146] [SWS_FrTSyn_00147] [SWS_FrTSyn_00148] [SWS_FrTSyn_00149] [SWS_FrTSyn_00150] [SWS_FrTSyn_00151] [SWS_FrTSyn_00152] [SWS_FrTSyn_91000] [SWS_FrTSyn_91001] [SWS_FrTSyn_91002] [SWS_FrTSyn_CONSTR_00001] [SWS_FrTSyn_NA_00999]

B.3.2 Changed Specification Items in R22-11

[SWS_FrTSyn_00009] [SWS_FrTSyn_00014] [SWS_FrTSyn_00015] [SWS_FrTSyn_00018] [SWS_FrTSyn_00019] [SWS_FrTSyn_00021] [SWS_FrTSyn_00022] [SWS_FrTSyn_00023] [SWS_FrTSyn_00025] [SWS_FrTSyn_00028] [SWS_FrTSyn_00036] [SWS_FrTSyn_00037] [SWS_FrTSyn_00038] [SWS_FrTSyn_00039] [SWS_FrTSyn_00040] [SWS_FrTSyn_00042] [SWS_FrTSyn_00043] [SWS_FrTSyn_00044] [SWS_FrTSyn_00055] [SWS_FrTSyn_00056] [SWS_FrTSyn_00057] [SWS_FrTSyn_00059] [SWS_FrTSyn_00060] [SWS_FrTSyn_00061] [SWS_FrTSyn_00062] [SWS_FrTSyn_00063] [SWS_FrTSyn_00064] [SWS_FrTSyn_00065] [SWS_FrTSyn_00066] [SWS_FrTSyn_00067] [SWS_FrTSyn_00069] [SWS_FrTSyn_00070] [SWS_FrTSyn_00071]

[SWS_FrTSyn_00074] [SWS_FrTSyn_00075] [SWS_FrTSyn_00079] [SWS_FrTSyn_00080] [SWS_FrTSyn_00081] [SWS_FrTSyn_00082] [SWS_FrTSyn_00095]

B.3.3 Deleted Specification Items in R22-11

[SWS_FrTSyn_00999]

B.3.4 Added Constraints in R22-11

none

B.3.5 Changed Constraints in R22-11

none

B.3.6 Deleted Constraints in R22-11

none