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

Document Status	Final
Part of AUTOSAR Release	4.2.2

Document Change History

Release	Changed by	Change Description
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)
4.2.1	AUTOSAR Release Management	• Initial Release

Disclaimer

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

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Advice for users

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

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

Table of Contents

1 Introduction and functional overview ... 5

2 Acronyms, Abbreviations, and Definitions ... 7

3 Related documentation ... 8

3.1 Input documents ... 8
3.2 Related specification ... 8

4 Constraints and assumptions ... 9

4.1 Limitations ... 9
4.2 Applicability to car domains ... 9

5 Dependencies to other modules ... 10

5.1 File structure ... 12
5.1.1 Code file structure ... 12
5.1.2 Header file structure ... 12

6 Requirements traceability ... 14

7 Functional specification ... 16

7.1 Overview ... 16
7.2 Module Handling ... 16
7.2.1 Initialization ... 16
7.2.2 Flexray Interface ... 17
7.3 Message Format ... 17
7.3.1 SYNC message ... 18
7.3.2 OFS message ... 19
7.4 Acting as Time Master ... 20
7.4.1 SYNC message processing ... 21
7.4.2 OFS message processing ... 21
7.4.3 Transmission mode ... 22
7.4.4 Calculation and Assembling of Time Synchronization Messages on FlexRay ... 22
7.5 Acting as Time Slave ... 24
7.5.1 SYNC message processing ... 24
7.5.2 OFS message processing ... 25
7.5.3 Validation and Disassembling of Time Synchronization Messages on FlexRay ... 26
7.6 Error Classification ... 29
7.7 Error Detection ... 29
7.8 Error Notification ... 29

8 API specification ... 30

8.1 API ... 30
8.1.1 Imported types ... 30
8.1.2 Type definitions ... 30
8.1.3 Function definitions ... 31

8.1.4 Call-back notifications 32
8.1.5 Scheduled functions 34
8.1.6 Expected Interfaces 36

9 Sequence diagrams 37

9.1 StbM_GetCurrentTime <Master FR SYNC> 37
9.2 StbM_BusSetGlobalTime <Slave FR SYNC> 38

10 Configuration specification 39

10.1 How to read this chapter 39
10.2 Containers and configuration parameters 40
10.2.1 Variants 40
10.2.2 FrTSyn 40
10.2.3 FrTSynGeneral 41
10.2.4 FrTSynGlobalTimeSyncDataIDList 42
10.2.5 FrTSynGlobalTimeSyncDataIDListElement 44
10.2.6 FrTSynGlobalTimeOfsDataIDList 45
10.2.7 FrTSynGlobalTimeOfsDataIDListElement 46
10.2.8 FrTSynGlobalTimeDomain 47
10.2.9 FrTSynGlobalTimeMaster 49
10.2.10 FrTSynGlobalTimeMasterPdu 50
10.2.11 FrTSynGlobalTimeSlave 51
10.2.12 FrTSynGlobalTimeSlavePdu 53
10.3 Published Information 55

1 Introduction and functional overview

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

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 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 is in combination with the current FlexRay macrotick counter and cycle counter to determine the actual master time and set its slave time.

The following Figure shows the time synchronization mechanism on FlexRay.

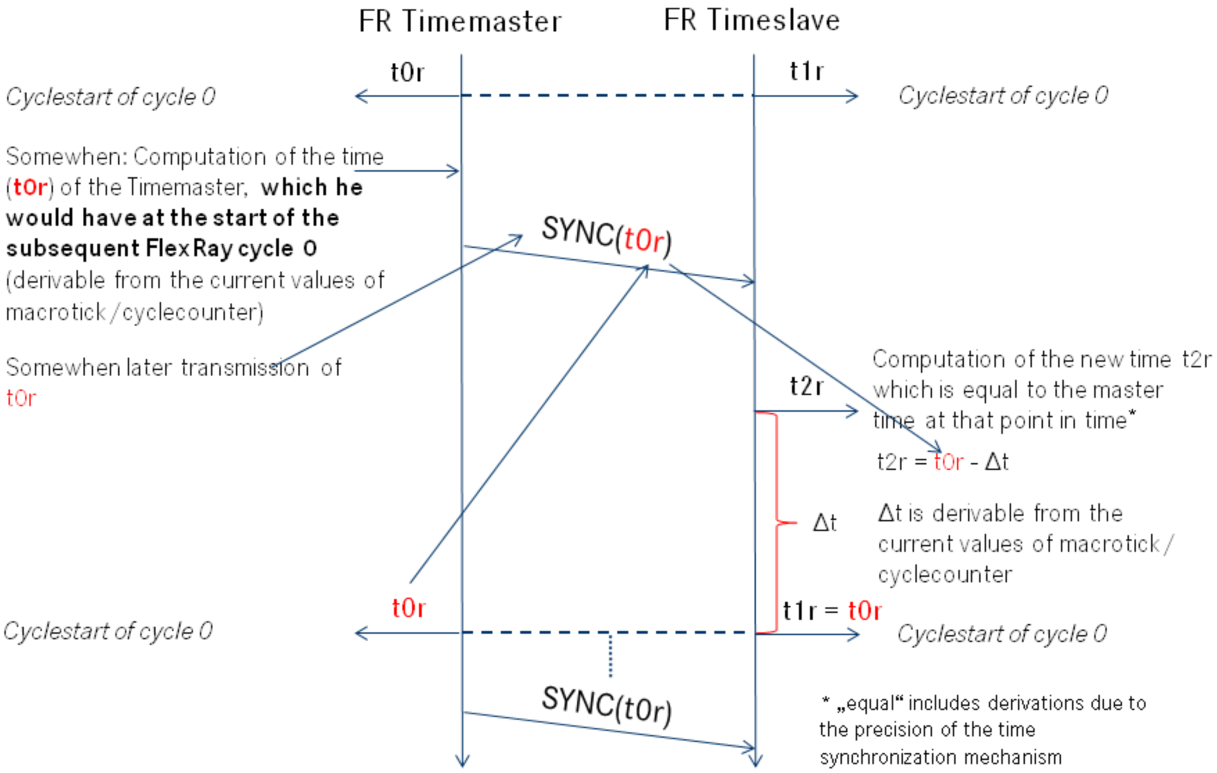

FR Timemaster
FR Timeslave
t0r
t1r
Cyclestart of cycle 0
Cyclestart of cycle 0
Somewhen: Computation of the time (t0r) of the Timemaster, which he would have at the start of the subsequent FlexRay cycle 0 (derivable from the current values of macrotick / cyclecounter)
SYNC(t0r)
Somewhen later transmission of t0r
Computation of the new time t2r which is equal to the master time at that point in time*
t2r
t2r = t0r - Δt
Δt
Δt is derivable from the current values of macrotick / cyclecounter
t0r
t1r = t0r
Cyclestart of cycle 0
Cyclestart of cycle 0
SYNC(t0r)
* „equal“ includes derivations due to the precision of the time synchronization mechanism

Figure 1: FlexRay time synchronization mechanism

2 Acronyms, Abbreviations, and Definitions

This section lists module local Abbreviations and Definitions. For a complete set of Synchronized Time Base related Abbreviations and Definitions refer to the corresponding chapter in [3].

Abbreviation / Acronym:	Description
(G)TD	(Global) Time Domain
(G)TM	(Global)Time Master
<Bus>TSyn	A bus specific Time Synchronization Provider module
CRC	Cyclic Redundancy Checksum
DEM	Diagnostic Event Manager
DET	Development Error Tracer
FR	Flexray
FUP message	Time adjustment message (Follow-Up)
OFNS message	Offset adjustment message
OFS message	Offset synchronization message
StbM	Synchronized Time-Base Manager
SYNC message	Time synchronization message
TG	Time Gateway
TS	Time Slave
TSD	Time Sub-domain

3 Related documentation

3.1 Input documents

[1] AUTOSAR Layered Software Architecture
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf

[2] AUTOSAR General Specification of Basic Software Modules
AUTOSAR_SWS_BSWGeneral.pdf

[3] AUTOSAR Synchronized Time Base Manager
AUTOSAR_SWS_SynchronizedTimeBaseManager.pdf

[4] AUTOSAR Specification of CRC Routines
AUTOSAR_SWS_CRCLibrary.pdf

3.2 Related specification

AUTOSAR provides a General Specification on Basic Software (SWS BSW General [2]) which is also valid for FrTSyn.

Thus, the General Specification on Basic Software (SWS BSW General) shall be considered additionally and as required specification for FrTSyn.

4 Constraints and assumptions

4.1 Limitations

Time Master and Time Slaves shall work with a Time Base reference clock with a worst-case accuracy of 10µs.

4.2 Applicability to car domains

Systems requiring a common Time Base to ECUs independent to which bus system the ECU is connected.

5 Dependencies to other modules

The Time Synchronization over FlexRay (FrTSyn) has interfaces towards the Synchronized Time Base Manager (StbM), the FlexRay Interface (FrIf) and the Development Error Tracer (DET).

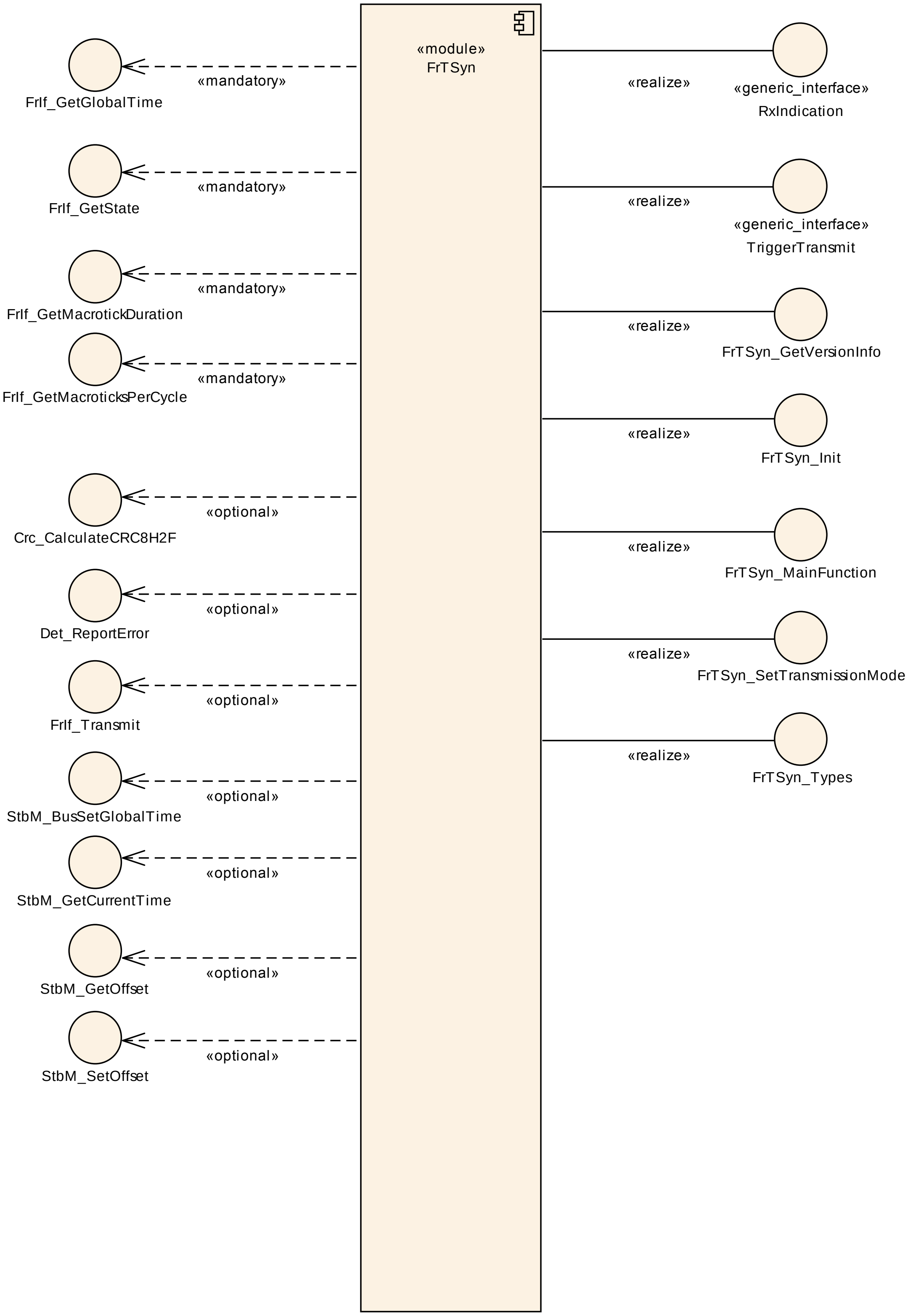


Figure 2: Module dependencies of the FrTSyn module

- StbM – Get and set the current time value

- FrIf – Receiving and transmitting messages
- BswM – Coordination of network access (via `FrTSyn_SetTransmissionMode()`)
- DET – Reporting of development errors

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 [2].

5.1.2 Header file structure

Besides the files defined in section 5.1.7 “Header file structure” of the SWS BSW General [2], the Time Synchronization over FlexRay needs to include the files defined below.

[SWS_FrTSyn_00002]
⌈The implementation header files shall include *ComStack_Types.h*.
⌋(SRS_BSW_00301, SRS_BSW_00456)

The following picture shows the include hierarchy of the Time Synchronization over FlexRay.

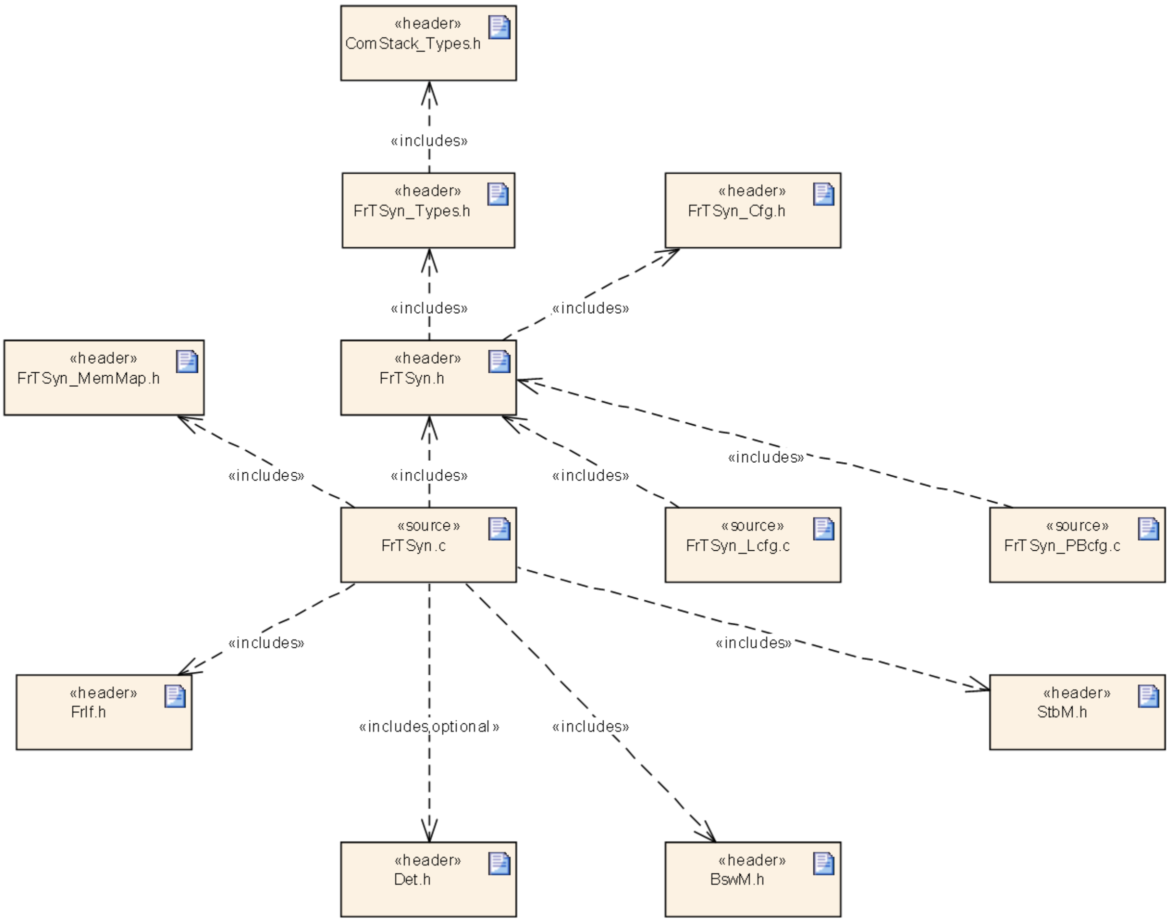


Figure 3: File structure of FrTSyn

6 Requirements traceability

Requirement	Description	Satisfied by
SRS_BSW_00301	All AUTOSAR Basic Software Modules shall only import the necessary information	SWS_FrTSyn_00002
SRS_BSW_00323	All AUTOSAR Basic Software Modules shall check passed API parameters for validity	SWS_FrTSyn_00058, SWS_FrTSyn_00067, SWS_FrTSyn_00070
SRS_BSW_00337	Classification of development errors	SWS_FrTSyn_00004, SWS_FrTSyn_00067, SWS_FrTSyn_00070, SWS_FrTSyn_00073
SRS_BSW_00385	List possible error notifications	SWS_FrTSyn_00059
SRS_BSW_00396	The Basic Software Module specifications shall specify the supported configuration classes for changing values and multiplicities for each parameter/container	SWS_FrTSyn_00076
SRS_BSW_00456	- A Header file shall be defined in order to harmonize BSW Modules	SWS_FrTSyn_00002
SRS_StbM_20017	The StbM shall initialize the local time base if configured as Time Slave	SWS_FrTSyn_00003, SWS_FrTSyn_00005
SRS_StbM_20018	The StbM shall initialize the local time base with 0 at startup if configured as Time Slave	SWS_FrTSyn_00003, SWS_FrTSyn_00005
SRS_StbM_20019	The StbM shall initialize the global time base with a configurable startup value if configured as Time Master	SWS_FrTSyn_00003, SWS_FrTSyn_00005
SRS_StbM_20035	The CAN TimeSync module shall support a protocol for precise time measurement and synchronization over CAN	SWS_FrTSyn_00074
SRS_StbM_20038	The CAN TimeSync configuration shall allow the CanTSyn to support different roles for a time base	SWS_FrTSyn_00077
SRS_StbM_20039	The FlexRay TimeSync module shall trigger time base synchronization transmission	SWS_FrTSyn_00018, SWS_FrTSyn_00019, SWS_FrTSyn_00023, SWS_FrTSyn_00026, SWS_FrTSyn_00027
SRS_StbM_20040	The FlexRay TimeSync module shall provide a time base after reception of a valid protocol information	SWS_FrTSyn_00041, SWS_FrTSyn_00045, SWS_FrTSyn_00078
SRS_StbM_20041	The FlexRay TimeSync module shall support means to protect the time synchronization	SWS_FrTSyn_00006, SWS_FrTSyn_00014, SWS_FrTSyn_00015, SWS_FrTSyn_00016, SWS_FrTSyn_00017, SWS_FrTSyn_00021,

	protocol	SWS_FrTSyn_00025, SWS_FrTSyn_00030, SWS_FrTSyn_00031, SWS_FrTSyn_00032, SWS_FrTSyn_00033, SWS_FrTSyn_00034, SWS_FrTSyn_00035, SWS_FrTSyn_00036, SWS_FrTSyn_00078
SRS_StbM_20042	The FlexRay TimeSync Module shall detect and handle timeout and integrity errors in the time synchronization protocol	SWS_FrTSyn_00015, SWS_FrTSyn_00017, SWS_FrTSyn_00038, SWS_FrTSyn_00041, SWS_FrTSyn_00042, SWS_FrTSyn_00045, SWS_FrTSyn_00048, SWS_FrTSyn_00049, SWS_FrTSyn_00050, SWS_FrTSyn_00051, SWS_FrTSyn_00052, SWS_FrTSyn_00053, SWS_FrTSyn_00054, SWS_FrTSyn_00055, SWS_FrTSyn_00057, SWS_FrTSyn_00058
SRS_StbM_20043	The FlexRay TimeSync module shall support a protocol for precise time measurement and synchronization over FlexRay	SWS_FrTSyn_00007, SWS_FrTSyn_00008, SWS_FrTSyn_00009, SWS_FrTSyn_00010, SWS_FrTSyn_00014, SWS_FrTSyn_00015, SWS_FrTSyn_00016, SWS_FrTSyn_00017, SWS_FrTSyn_00018, SWS_FrTSyn_00019, SWS_FrTSyn_00020, SWS_FrTSyn_00021, SWS_FrTSyn_00022, SWS_FrTSyn_00023, SWS_FrTSyn_00024, SWS_FrTSyn_00025, SWS_FrTSyn_00026, SWS_FrTSyn_00027, SWS_FrTSyn_00028, SWS_FrTSyn_00029, SWS_FrTSyn_00030, SWS_FrTSyn_00031, SWS_FrTSyn_00032, SWS_FrTSyn_00033, SWS_FrTSyn_00034, SWS_FrTSyn_00035, SWS_FrTSyn_00036, SWS_FrTSyn_00037, SWS_FrTSyn_00038, SWS_FrTSyn_00039, SWS_FrTSyn_00040, SWS_FrTSyn_00041, SWS_FrTSyn_00042, SWS_FrTSyn_00043, SWS_FrTSyn_00044, SWS_FrTSyn_00045, SWS_FrTSyn_00046, SWS_FrTSyn_00047, SWS_FrTSyn_00048, SWS_FrTSyn_00049, SWS_FrTSyn_00050, SWS_FrTSyn_00051, SWS_FrTSyn_00052, SWS_FrTSyn_00053, 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_00068, SWS_FrTSyn_00069, SWS_FrTSyn_00071, SWS_FrTSyn_00072, SWS_FrTSyn_00075
SRS_StbM_20044	The FlexRay TimeSync module shall use the time measurement and synchronization protocol to transmit and receive an offset value	SWS_FrTSyn_00022, SWS_FrTSyn_00023, SWS_FrTSyn_00025, SWS_FrTSyn_00042, SWS_FrTSyn_00043, SWS_FrTSyn_00044, SWS_FrTSyn_00047, SWS_FrTSyn_00049
SRS_StbM_20045	The FlexRay TimeSync module shall support user specific data within the time measurement and synchronization protocol	SWS_FrTSyn_00008, SWS_FrTSyn_00010, SWS_FrTSyn_00011, SWS_FrTSyn_00012, SWS_FrTSyn_00013

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 [3].

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]
⌈A call to `FrTSyn_Init()` initializes all internal variables and sets the Time Synchronization over FlexRay to the initialized state.
⌋(SRS_StbM_20017, SRS_StbM_20018, SRS_StbM_20019)

[SWS_FrTSyn_00004]
⌈When DET reporting is enabled (see `FrTSynDevErrorDetect`), the Time Synchronization over FlexRay shall call Det_ReportError with the error code `FRTSYN_E_NOT_INITIALIZED` when any API other than `FrTSyn_GetVersionInfo()` or `FrTSyn_Init()` is called in uninitialized state.
⌋(SRS_BSW_00337)

[SWS_FrTSyn_00005]
⌈When `FrTSyn_Init()` is called in initialized state, the Time Synchronization over FlexRay shall re-initialize its internal variables.
⌋(SRS_StbM_20017, SRS_StbM_20018, SRS_StbM_20019)

[SWS_FrTSyn_00006]
⌈Sequence Counter (*SC*) shall be initialized with 0.
⌋(SRS_StbM_20041)

7.2.2 Flexray Interface

[SWS_FrTSyn_00078]
⌈ The FrTSyn module shall call Fr_GetGlobalTime() only if FrIf_GetState() returns FRIF_STATE_ONLINE. This is to ensure that Fr_GetGlobalTime returns valid time information, i.e. that the Flexray communication controller is synchronous to the Flexray global time.⌋ (SRS_StbM_20040, SRS_StbM_20041)

7.3 Message Format

SYNC and OFS messages are assigned to a dedicated SLOT ID of the Static Segment.

SYNC and OFS messages share the same FR PDU by using a multiplexed signal group. The Multiplexer is located at Byte 0, named as "*Type*".

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

[SWS_FrTSyn_00007]
⌈The byte order for time synchronization message time stamp signals is "Big Endian".
⌋(SRS_StbM_20043)

[SWS_FrTSyn_00008]
⌈The byte order for time synchronization message User Data bytes is "Opaque".
⌋(SRS_StbM_20043, SRS_StbM_20045)

[SWS_FrTSyn_00009]
⌈The PayloadLength is 16.
⌋(SRS_StbM_20043)

[SWS_FrTSyn_00010]
⌈Each time synchronization message contains User Data according to the given message format.
⌋(SRS_StbM_20043, SRS_StbM_20045)

[SWS_FrTSyn_00011]
⌈User Data shall be read from the current incoming message consistently.
⌋(SRS_StbM_20045)

[SWS_FrTSyn_00012]
⌈User Data shall be written to the next outgoing message consistently.
⌋(SRS_StbM_20045)

[SWS_FrTSyn_00013]

⌈User Data shall be mapped to the `StbM_UserDataType`, whereas the byte number given in the message and by the `StbM_UserDataType` shall match (User Byte 0 mapped to `StbM_UserDataType.userByte0` etc.). Afterwards `StbM_UserDataType.userDataLength` shall be set accordingly.
⌋(SRS_StbM_20045)

7.3.1 SYNC message

[SWS_FrTSyn_00014]
⌈SYNC not CRC secured message format:
Byte 0: *Type* = 0x10
Byte 1: User Byte 2, default: 0
Byte 2: *D* = Time Domain 0 to 15 (Bit 7 to Bit 4)
SC = Sequence Counter (Bit 3 to Bit 0)
Byte 3: *FCNT*= FlexRay Cycle Counter from 0 to 63 (Bit 7 to Bit 2)
SGW (Bit 1)
SyncToGTM = 0
SyncToSubDomain = 1
reserved (Bit 0), default: 0
Byte 4: User Byte 0, default: 0
Byte 5: User Byte 1, default: 0
Byte 6-11: *SyncTimeSec* = 48 Bit time stamp in seconds
Byte 12-15: *SyncTimeNSec* = 32 Bit time stamp in nanoseconds
⌋(SRS_StbM_20041, SRS_StbM_20043)

[SWS_FrTSyn_00015]
⌈SYNC CRC secured message format:
Byte 0: Type = 0x20
Byte 1: *CRC*
Byte 2: *D* = Time Domain 0 to 15 (Bit 7 to Bit 4)
SC = Sequence Counter (Bit 3 to Bit 0)
Byte 3: *FCNT*= FlexRay Cycle Counter from 0 to 63 (Bit 7 to Bit 2)
SGW (Bit 1)
SyncToGTM = 0
SyncToSubDomain = 1
reserved (Bit 0), default: 0
Byte 4: User Byte 0, default: 0
Byte 5: User Byte 1, default: 0
Byte 6-11: *SyncTimeSec* = 48 Bit time stamp in seconds
Byte 12-15: *SyncTimeNSec* = 32 Bit time stamp in nanoseconds
⌋(SRS_StbM_20041, SRS_StbM_20042, SRS_StbM_20043)

7.3.2 OFS message

Offset messages are multiplexed with the SYNC message (using the same PDU, etc.).

[SWS_FrTSyn_00016]
⌈OFS not CRC secured message format:

Byte 0:	*Type* = 0x30
Byte 1:	User Byte 3, default: 0
Byte 2:	*D* = Time Domain 16 to 31 (Bit 7 to Bit 4) *SC* = Sequence Counter (Bit 3 to Bit 0)
Byte 3:	User Byte 0, default: 0
Byte 4:	User Byte 1, default: 0
Byte 5:	User Byte 2, default: 0
Byte 6-11:	*OfsTimeSec* = 48 Bit offset time stamp in seconds
Byte 12-15:	*OfsTimeNSec* = 32 Bit offset time stamp in nanoseconds

⌋(SRS_StbM_20041, SRS_StbM_20043)

[SWS_FrTSyn_00017]
⌈OFS CRC secured message format:

Byte 0:	Type = 0x40
Byte 1:	*CRC*
Byte 2:	*D* = Time Domain 16 to 31 (Bit 7 to Bit 4) *SC* = Sequence Counter (Bit 3 to Bit 0)
Byte 3:	User Byte 0, default: 0
Byte 4:	User Byte 1, default: 0
Byte 5:	User Byte 2, default: 0
Byte 6-11:	*OfsTimeSec* = 48 Bit offset time stamp in seconds
Byte 12-15:	*OfsTimeNSec* = 32 Bit offset time stamp in nanoseconds

⌋(SRS_StbM_20041, SRS_StbM_20042, SRS_StbM_20043)

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 derived, 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.

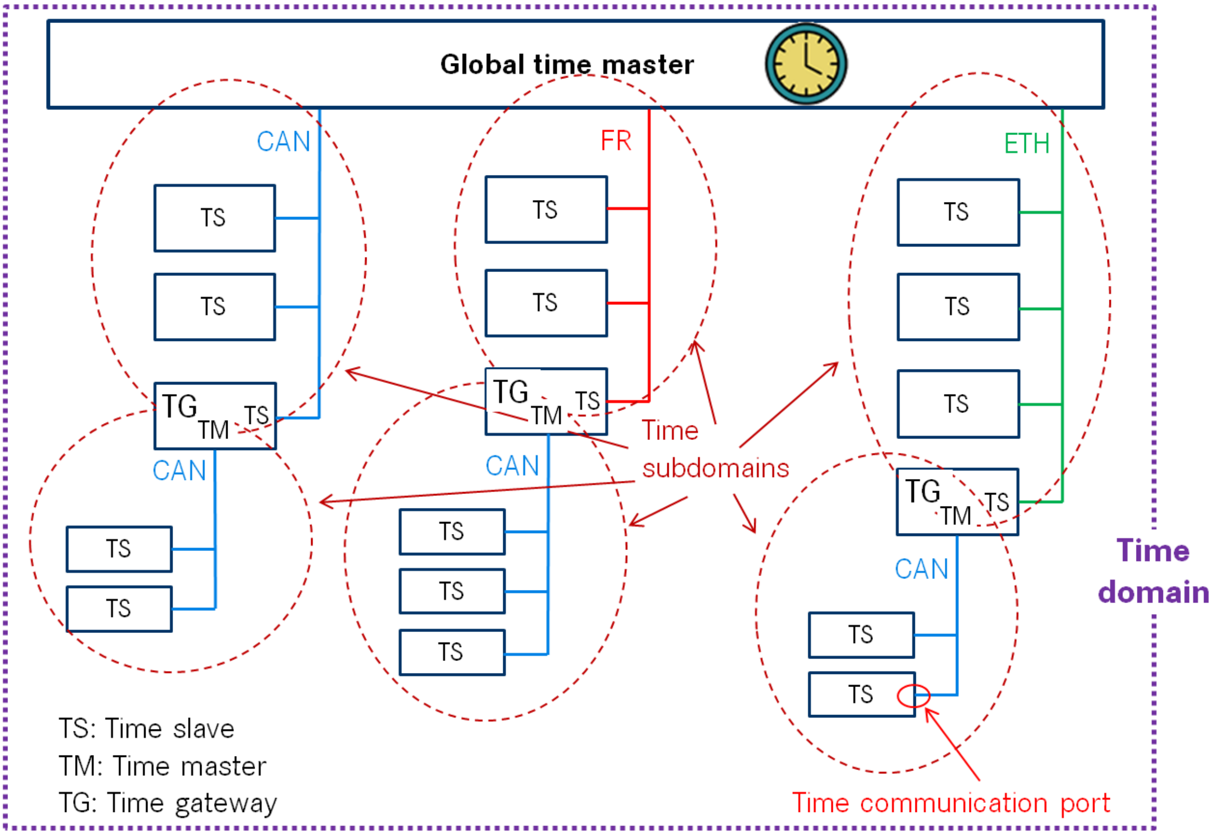


Figure 4: Terminology Example

[SWS_FrTSyn_00024]
⌈The FrTSyn module shall transmit in one FlexRay cycle either SYNC or OFS message.
⌋(SRS_StbM_20043)

7.4.1 SYNC message processing

[SWS_FrTSyn_00018]
⌈A synchronization message sequence consists of a SYNC message per Time Domain transmitted once per FlexRay cycle.
⌋(SRS_StbM_20039, SRS_StbM_20043)

[SWS_FrTSyn_00019]
⌈For each configured Time Master (`FrTSynGlobalTimeMaster`) the FrTSyn module shall periodically transmit a SYNC messages with the cycle `FrTSynGlobalTimeTxPeriod` (**ECUC_FrTSyn_00014 :**) including the time stamp that will be valid at the next FlexRay cycle 0 (see Figure 5: Master FR SYNC).
⌋(SRS_StbM_20039, SRS_StbM_20043)

[SWS_FrTSyn_00021]
⌈Depending on `FrTSynGlobalTimeTxCrcSecured` the SYNC message shall be of type:

FrTSynGlobalTimeTxCrcSecured	SYNC
CRC_NOT_SUPPORTED	0x10 SYNC not CRC secured message
CRC_SUPPORTED	0x20 SYNC CRC secured message

⌋(SRS_StbM_20041, SRS_StbM_20043)

7.4.2 OFS message processing

[SWS_FrTSyn_00022]
⌈An offset message sequence consists of an OFS message per Time Domain transmitted once per FlexRay cycle.
⌋(SRS_StbM_20043, SRS_StbM_20044)

[SWS_FrTSyn_00023]
⌈For each configured Time Master (`FrTSynGlobalTimeMaster`) the FrTSyn module shall periodically transmit a OFS messages with the cycle `FrTSynGlobalTimeTxPeriod` (**ECUC_FrTSyn_00014 :**) including the offset time stamp.
⌋(SRS_StbM_20039, SRS_StbM_20043, SRS_StbM_20044)

[SWS_FrTSyn_00025]
⌈Depending on `FrTSynGlobalTimeTxCrcSecured` the OFS message shall be of type:

FrTSynGlobalTimeTxCrcSecured	OFS
CRC_NOT_SUPPORTED	0x30 OFS not CRC secured message
CRC_SUPPORTED	0x40 OFS CRC secured message

⌋(SRS_StbM_20041, SRS_StbM_20043, SRS_StbM_20044)

7.4.3 Transmission mode

[SWS_FrTSyn_00026]
⌈If `FrTSyn_SetTransmissionMode(Controller, Mode)` is called and parameter `Mode` equals `FRTSYN_TX_OFF`, all transmit request from FrTSyn shall be omitted on this FlexRay channel.
⌋(SRS_StbM_20039, SRS_StbM_20043)

[SWS_FrTSyn_00027]
⌈If `FrTSyn_SetTransmissionMode(Controller, Mode)` is called and parameter `Mode` equals `FRTSYN_TX_ON`, all transmit request from FrTSyn on this FlexRay channel shall be able to be transmitted.
⌋(SRS_StbM_20039, SRS_StbM_20043)

7.4.4 Calculation and Assembling of Time Synchronization Messages on FlexRay

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).

7.4.4.1 Global Time Calculation

[SWS_FrTSyn_00028]
⌈The transmitter of a Synchronized Time Base (Time Master) shall perform the following steps to distribute the Synchronized Time Base:

1. Get `currentCycle` and `currentMacroticks` from FlexRay driver

2. Retrieve current Synchronized Time Base as T_{SYNC} via `StbM_GetCurrentTime()`

3. Calculate the time stamp (in future) of the start of the next FlexRay cycle, where the subsequent SYNC message will sent with
   T0 = T_{SYNC} + (`MacroticksPerCycle` * (64 - `currentCycle`) - `currentMacroticks`) * `MacrotickDuration`

4. Calculate *SyncTimeSec* (second portion of T0) and *SyncTimeNSec* (nanosecond portion of T0)

⌋(SRS_StbM_20043)

Note: `MacroticksPerCycle` and `MacrotickDuration` are given statically by configuration.

[SWS_FrTSyn_00029]
⌈The transmitter of an Offset Time Base (Time Master) shall perform the following steps to distribute the Offset Time Base:

1. Get second portion of the Offset Time Base and write to *OfsTimeSec*
2. Get nanosecond portion of the Offset Time and write to *OfsTimeNSec*

⌋(SRS_StbM_20043)

Note: The Offset Time Base has not been time stamped.

7.4.4.2 SGW Calculation

[SWS_FrTSyn_00020]
⌈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*.
⌋(SRS_StbM_20043)

7.4.4.3 Sequence Counter Calculation

[SWS_FrTSyn_00030]
⌈A Sequence Counter (*SC*) of 4bit is representing numbers from 0 to 15 per Time Domain. The Sequence Counter is independent between SYNC and OFS messages and shall be incremented by 1 continuously on every transmission request of a SYNC or OFS message. It wraps around at 15 to 0 again.
⌋(SRS_StbM_20041, SRS_StbM_20043)

7.4.4.4 CRC Calculation

[SWS_FrTSyn_00031]
⌈The function `Crc_CalculateCRC8H2F()` as defined in [4] shall be used to calculate the *CRC* if configured.
⌋(SRS_StbM_20041, SRS_StbM_20043)

[SWS_FrTSyn_00032]
⌈The *CRC* start value shall be 0xFF.
⌋(SRS_StbM_20041, SRS_StbM_20043)

[SWS_FrTSyn_00033]
⌈The *CRC* final XOR-value shall be 0xFF.
⌋(SRS_StbM_20041, SRS_StbM_20043)

[SWS_FrTSyn_00034]
⌈The *CRC* polynomial shall be 0x2F.
⌋(SRS_StbM_20041, SRS_StbM_20043)

[SWS_FrTSyn_00035]
⌈The `DataID` will be calculated as `DataID = DataIDList[`*SC*`]`, whereat `DataIDList` is given by configuration for each message *Type*.
⌋(SRS_StbM_20041, SRS_StbM_20043)

A specific `DataID` out of a predefined `DataIDList` ensures the identification of data elements of time synchronization messages.

[SWS_FrTSyn_00036]
⌈The *CRC* will be calculated over time synchronization message *Byte 2* to *Byte 15* and `DataID`.
⌋(SRS_StbM_20041, SRS_StbM_20043)

7.4.4.5 Message Assembling

[SWS_FrTSyn_00037]
⌈For each transmission of a time synchronization message the FrTSyn module shall assemble the message like follows:

1. Calculate *SC*
2. Calculate *FCNT*
3. Calculate *SGW*
4. Copy all data to the appropriate position within the related message
5. Calculate *CRC* (configuration dependent)

⌋(SRS_StbM_20043)

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]
⌈The FrTSyn shall only accept SYNC message with *Type* equal to `0x20` and a correct *CRC* value if `FrTSynRxCrcValidated` is configured to `CRC_VALIDATED`. All other messages shall be ignored.
⌋(SRS_StbM_20042, SRS_StbM_20043)

[SWS_FrTSyn_00039]
⌈The FrTSyn shall only accept SYNC message with *Type* equal to `0x10` if `FrTSynRxCrcValidated` is configured to `CRC_NOT_VALIDATED`. All other messages shall be ignored.
⌋(SRS_StbM_20043)

[SWS_FrTSyn_00040]
⌈The FrTSyn shall only accept SYNC message with *Type* equal to `0x10` or `0x20` if `FrTSynRxCrcValidated` is configured to `CRC_IGNORED`. All other messages shall be ignored.⌋(SRS_StbM_20043)

[SWS_FrTSyn_00041]
⌈For valid SYNC messages a new Global Time value shall be calculated and forwarded to the StbM module via `StbM_BusSetGlobalTime()` (see Figure 6: Slave FR SYNC).
⌋(SRS_StbM_20040, SRS_StbM_20042, SRS_StbM_20043)

7.5.2 OFS message processing

[SWS_FrTSyn_00042]
⌈The FrTSyn shall only accept OFS message with *Type* equal to `0x40` and a correct *CRC* value if `FrTSynRxCrcValidated` is configured to `CRC_VALIDATED`. All other messages shall be ignored.
⌋(SRS_StbM_20042, SRS_StbM_20043, SRS_StbM_20044)

[SWS_FrTSyn_00043]
⌈The FrTSyn shall only accept OFS message with *Type* equal to `0x30` if `FrTSynRxCrcValidated` is configured to `CRC_NOT_VALIDATED`. All other messages shall be ignored.
⌋(SRS_StbM_20043, SRS_StbM_20044)

[SWS_FrTSyn_00044]
⌈The FrTSyn shall only accept OFS message with *Type* equal to `0x30` or `0x40` if `FrTSynRxCrcValidated` is configured to `CRC_IGNORED`. All other messages shall be ignored.
⌋(SRS_StbM_20043, SRS_StbM_20044)

[SWS_FrTSyn_00045]
⌈For valid OFS messages a new offset time value shall be calculated (according **[SWS_FrTSyn_00047]**) and forwarded to the StbM module via `StbM_SetOffset()`.
⌋(SRS_StbM_20040, SRS_StbM_20042, SRS_StbM_20043)

7.5.3 Validation and Disassembling of Time Synchronization Messages on FlexRay

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).

7.5.3.1 Global Time Calculation

[SWS_FrTSyn_00046]
⌈The receiver of a Synchronized Time Base (Time Slave) shall perform the following steps to assemble the Synchronized Time Base:

1. On SYNC message RX indication store received time stamp T0 (*SyncTimeSec, SyncTimeNSec*)

2. Get `currentCycle` and `currentMacroticks` from FlexRay driver

3. Calculate T1 to update the Time Slaves Local Time Base as:
   T1 = T0 + (((`MacroticksPerCycle` * `currentCycle`) + `currentMacroticks`) * `MacrotickDuration`)

4. If the `currentCycle` has passed the retrieved *FCNT* from the transmitter side then the previously calculated T1 must be adjusted by the maximum of the cycle counter to:
   T1 = T1 – (`MacroticksPerCycle` * 64 * `MacrotickDuration`)

⌋(SRS_StbM_20043)

Note: `MacroticksPerCycle` and `MacrotickDuration` are given statically by configuration.

[SWS_FrTSyn_00047]
⌈The receiver of an Offset Time Base (Time Slave) shall perform the following steps to assemble the Offset Time Base:

1. Get second portion of the Offset Time Base out of *OfsTimeSec*
2. Get nanosecond portion of the Offset Time out of *OfsTimeNSec*

⌋(SRS_StbM_20043, SRS_StbM_20044)

Note: The Offset Time Base has not been time stamped.

7.5.3.2 Sequence Counter Validation

[SWS_FrTSyn_00048]
⌈The Sequence Counter Jump Width between two SYNC resp. two OFS messages must be always smaller or equal to `FrTSynGlobalTimeSequenceCounterJumpWidth`. The value 0 is not allowed.
⌋(SRS_StbM_20042, SRS_StbM_20043)

[SWS_FrTSyn_00049]
⌈Due to an asynchronous startup of Time Slaves compared to the Time Master the 1st Sequence Counter of the 1st received SYNC and OFS message per Time Domain shall not be checked against the defined Sequence Counter Jump Width.
⌋(SRS_StbM_20042, SRS_StbM_20043, SRS_StbM_20044)

7.5.3.3 CRC Validation

[SWS_FrTSyn_00050]
⌈The function `Crc_CalculateCRC8H2F()` as defined in [4] shall be used to validate the *CRC* if configured.
⌋(SRS_StbM_20042, SRS_StbM_20043)

[SWS_FrTSyn_00051]
⌈The *CRC* start value shall be 0xFF.
⌋(SRS_StbM_20042, SRS_StbM_20043)

[SWS_FrTSyn_00052]
⌈The *CRC* final XOR-value shall be 0xFF.
⌋(SRS_StbM_20042, SRS_StbM_20043)

[SWS_FrTSyn_00053]
⌈The *CRC* polynomial shall be 0x2F.
⌋(SRS_StbM_20042, SRS_StbM_20043)

[SWS_FrTSyn_00054]
⌈The `DataID` will be calculated as `DataID = DataIDList[`*SC*`]`, whereat `DataIDList` is given by configuration for each message *Type*.
⌋(SRS_StbM_20042, SRS_StbM_20043)

A specific `DataID` out of a predefined `DataIDList` ensures the identification of data elements of time synchronization messages.

[SWS_FrTSyn_00055]
⌈The *CRC* will be calculated over time synchronization message *Byte 2* to *Byte 15* and `DataID`.
⌋(SRS_StbM_20042, SRS_StbM_20043)

7.5.3.4 Message Disassembling

[SWS_FrTSyn_00056]
⌈For each received time synchronization message the FrTSyn module shall validate the message like follows (all conditions must match):

1. *Type* matches depending on the `FrTSynRxCrcValidated` parameter.
2. *SC* matches to the expected value.
3. *D* matches to the defined Time Domain range for each *Type*
4. *D* matches to one of the configured Time Domains
5. *Time stamp* matches the defined range of `StbM_TimeStampType.nanoseconds`.
6. *CRC* (including `DataID`) matches depending on the `FrTSynRxCrcValidated` parameter.

⌋(SRS_StbM_20043)

[SWS_FrTSyn_00057]
⌈For each received time synchronization message the FrTSyn module shall disassemble the message after successful validation **[SWS_FrTSyn_00056]**.
⌋(SRS_StbM_20042, SRS_StbM_20043)

7.6 Error Classification

This chapter lists and classifies all errors that can be detected by this software module. Each error is classified to relevance (development / production) and the related error code (unique label for the error). For development errors this table also specifies the unique values, which correspond to the error codes.

[SWS_FrTSyn_00058]
⌈On errors and exceptions, the FrTSyn module shall not modify its current module state but shall simply report the error event.
⌋(SRS_StbM_20042, SRS_BSW_00323)

[SWS_FrTSyn_00059]
⌈FrTSyn shall use following errors:

Type or error	***Relevance***	***Related error code***	***Value [hex]***
API service called with wrong PDU or SDU.	Development	`FRTSYN_E_INVALID_PDUID`	`0x01`
API service used in un-initialized state	Development	`FRTSYN_E_NOT_INITIALIZED`	`0x20`
A pointer is invalid	Development	`FRTSYN_E_NULL_POINTER`	`0x21`

⌋(SRS_BSW_00385)

7.7 Error Detection

The detection of development errors is configurable (see section 10.2, `FrTSynDevErrorDetect`).

7.8 Error Notification

The module ID 163 of FrTSyn, which is used as a parameter in the `Det_ReportError()` call, is exported via the macro definition `FRTSYN_MODULE_ID` in *`FrTSyn.h`*.

8 API specification

8.1 API

8.1.1 Imported types

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

[SWS_FrTSyn_00060]
⌈

Module	*Imported Type*
ComStack_Types	PduIdType
	PduInfoType
FrIf	FrIf_StateType
StbM	StbM_SynchronizedTimeBaseType
	StbM_TimeStampType
	StbM_UserDataType
Std_Types	Std_ReturnType
	Std_VersionInfoType

⌋(SRS_StbM_20043)

8.1.2 Type definitions

8.1.2.1 FrTSyn_ConfigType

[SWS_FrTSyn_00061]
⌈

Name:	`FrTSyn_ConfigType`		
Type:	`Structure`		
Element:	`void`	`implementation specific`	--
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.		

⌋(SRS_StbM_20043)

8.1.2.2 FrTSyn_TransmissionModeType

[SWS_FrTSyn_00062]
⌈

Name:	FrTSyn_TransmissionModeType	
Type:	Enumeration	
Range:	FRTSYN_TX_OFF	Transmission Disabled
	FRTSYN_TX_ON	Transmission Enabled
Description:	Handles the enabling and disabling of the transmission mode	

⌋(SRS_StbM_20043)

8.1.3 Function definitions

8.1.3.1 FrTSyn_Init

[SWS_FrTSyn_00063]
⌈

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.	

⌋(SRS_StbM_20043)
See section 7.2.1 for details.

8.1.3.2 FrTSyn_GetVersionInfo

[SWS_FrTSyn_00064]
⌈

Service name:	FrTSyn_GetVersionInfo	

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

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

⌋(SRS_StbM_20043)

8.1.3.3 FrTSyn_SetTransmissionMode

[SWS_FrTSyn_00065]
⌈

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.	

⌋(SRS_StbM_20043)

8.1.4 Call-back notifications

This is a list of functions provided for other modules. The function prototypes of the callback functions shall be provided in the file *`FrTSyn_Cbk.h`*.

8.1.4.1 FrTSyn_RxIndication

[SWS_FrTSyn_00066]

⌈

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 I-PDU.
	PduInfoPtr	Contains the length (SduLength) of the received I-PDU and a pointer to a buffer (SduDataPtr) containing the I-PDU.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Indication of a received I-PDU from a lower layer communication interface module.	

⌋(SRS_StbM_20043)

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

[SWS_FrTSyn_00067]

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

- Invalid PDU ID (`FRTSYN_E_INVALID_PDUID`)
- FrTSyn was not initialized (`FRTSYN_E_NOT_INITIALIZED`)
- `PduInfoPtr` or `SduDataPtr` equals `NULL_PTR` (`FRTSYN_E_NULL_POINTER`)

⌋(SRS_BSW_00323, SRS_BSW_00337)

[SWS_FrTSyn_00068]

⌈Caveats of `FrTSyn_RxIndication()`:

- The FrTSyn module is initialized correctly.

⌋(SRS_StbM_20043)

8.1.4.2 FrTSyn_TriggerTransmit

[SWS_FrTSyn_00069]
⌈

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.	

⌋(SRS_StbM_20043)

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

[SWS_FrTSyn_00070]
⌈The callback function `FrTSyn_TriggerTransmit()` shall inform the DET, if development error detection is enabled (`FRTSYN_DEV_ERROR_DETECT` is set to TRUE) and if function call has failed because of the following reasons:

- Invalid PDU ID (`FRTSYN_E_INVALID_PDUID`)
- FrTSyn was not initialized (`FRTSYN_E_NOT_INITIALIZED`)
- `PduInfoPtr` or `SduDataPtr` equals `NULL_PTR` (`FRTSYN_E_NULL_POINTER`)

⌋(SRS_BSW_00323, SRS_BSW_00337)

8.1.5 Scheduled functions

These functions are directly called by 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]
⌈

Service name:	FrTSyn_MainFunction

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

Service ID[hex]:	0x04
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	Main function for cyclic call / resp. SYNC transmission

⌋(SRS_StbM_20043)

[SWS_FrTSyn_00072]
⌈The frequency of invocations of `FrTSyn_MainFunction()` is determined by the configuration parameter `FrTSynMainFunctionPeriod`.
⌋(SRS_StbM_20043)

[SWS_FrTSyn_00073]
⌈The scheduled function `FrTSyn_MainFunction()` shall inform the DET, if development error detection is enabled (`FRTSYN_DEV_ERROR_DETECT` is set to TRUE) and if function call has failed because of the following reasons:

- The FrTSyn module was not initialized (`FRTSYN_E_NOT_INITIALIZED`).

⌋(SRS_BSW_00337)

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]

⌈

API function	*Description*
FrIf_GetGlobalTime	Wraps the FlexRay Driver API function Fr_GetGlobalTime().
FrIf_GetMacrotickDuration	Retrieves the Duration of a Macrotick in ns
FrIf_GetMacroticksPerCycle	Retrieves the amount of Macroticks per Cycle
FrIf_GetState	Get current FrIf state.

⌋(SRS_StbM_20035)

⌋(SRS_StbM_20043)

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]

⌈

API function	*Description*
Crc_CalculateCRC8H2F	This service makes a CRC8 calculation with the Polynomial 0x2F on Crc_Length
Det_ReportError	Service to report development errors.
FrIf_Transmit	Requests the sending of a PDU.
StbM_BusSetGlobalTime	Allows the Timebase Provider Modules to forward a new Global Time to the StbM, which has been received from different busses.
StbM_GetCurrentTime	Returns a time value (Local Time Base derived from Global Time Base) in standard format.
StbM_GetOffset	Allows the Timebase Provider Modules to get the currentoffset time.
StbM_SetOffset	Allows the Customers and the Timebase Provider Modules to set the offset time that has to be valid for the system.

⌋(SRS_StbM_20043)

9 Sequence diagrams

9.1 StbM_GetCurrentTime <Master FR SYNC>

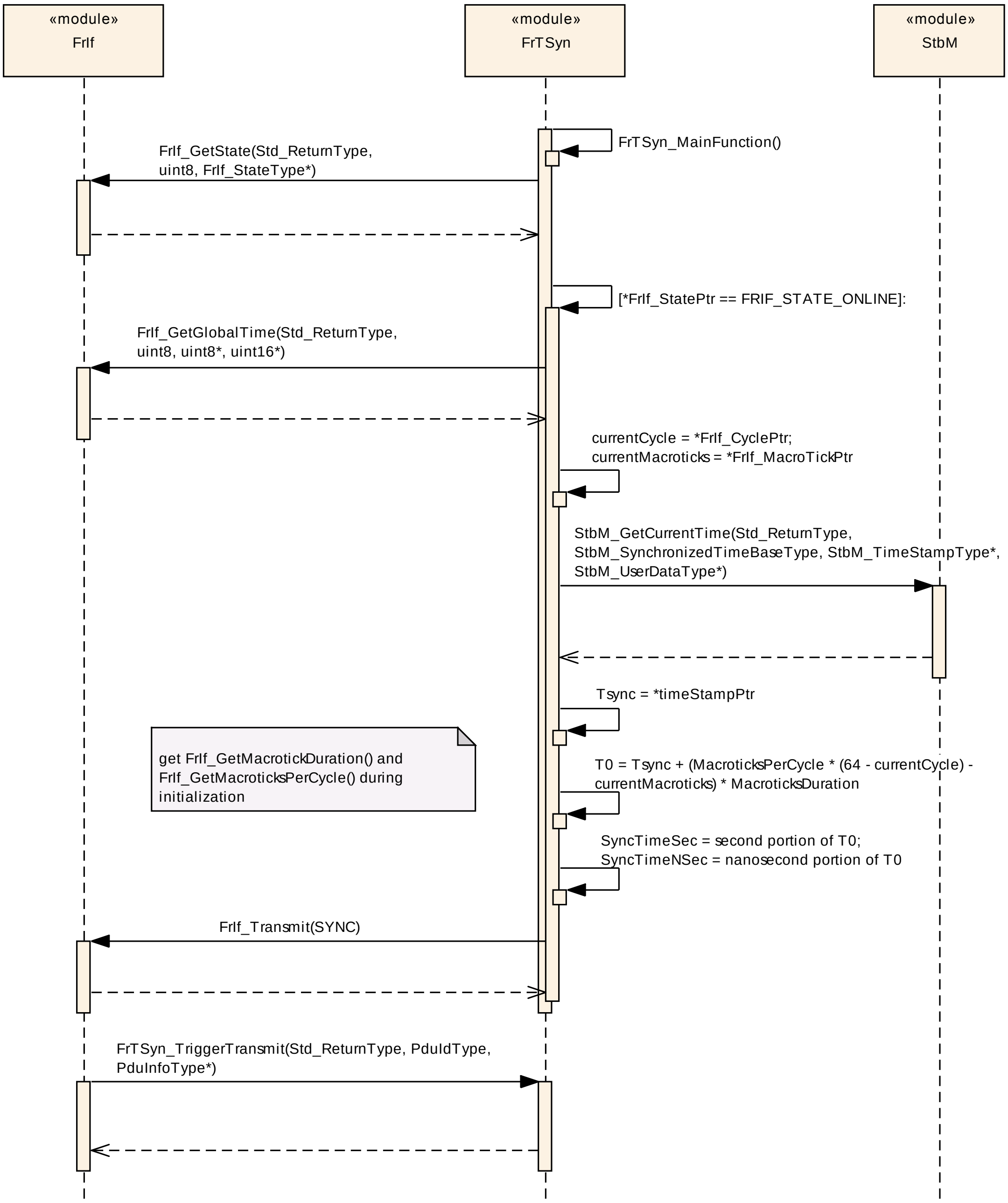

«module»
FrIf
«module»
FrTSyn
«module»
StbM
FrTSyn_MainFunction()
FrIf_GetState(Std_ReturnType, uint8, FrIf_StateType*)
[*FrIf_StatePtr == FRIF_STATE_ONLINE]:
FrIf_GetGlobalTime(Std_ReturnType, uint8, uint8*, uint16*)
currentCycle = *FrIf_CyclePtr;
currentMacroticks = *FrIf_MacroTickPtr
StbM_GetCurrentTime(Std_ReturnType, StbM_SynchronizedTimeBaseType, StbM_TimeStampType*, StbM_UserDataType*)
Tsync = *timeStampPtr
get FrIf_GetMacrotickDuration() and FrIf_GetMacroticksPerCycle() during initialization
T0 = Tsync + (MacroticksPerCycle * (64 - currentCycle) - currentMacroticks) * MacroticksDuration
SyncTimeSec = second portion of T0;
SyncTimeNSec = nanosecond portion of T0
FrIf_Transmit(SYNC)
FrTSyn_TriggerTransmit(Std_ReturnType, PduIdType, PduInfoType*)

Figure 5: Master FR SYNC

9.2 StbM_BusSetGlobalTime <Slave FR SYNC>

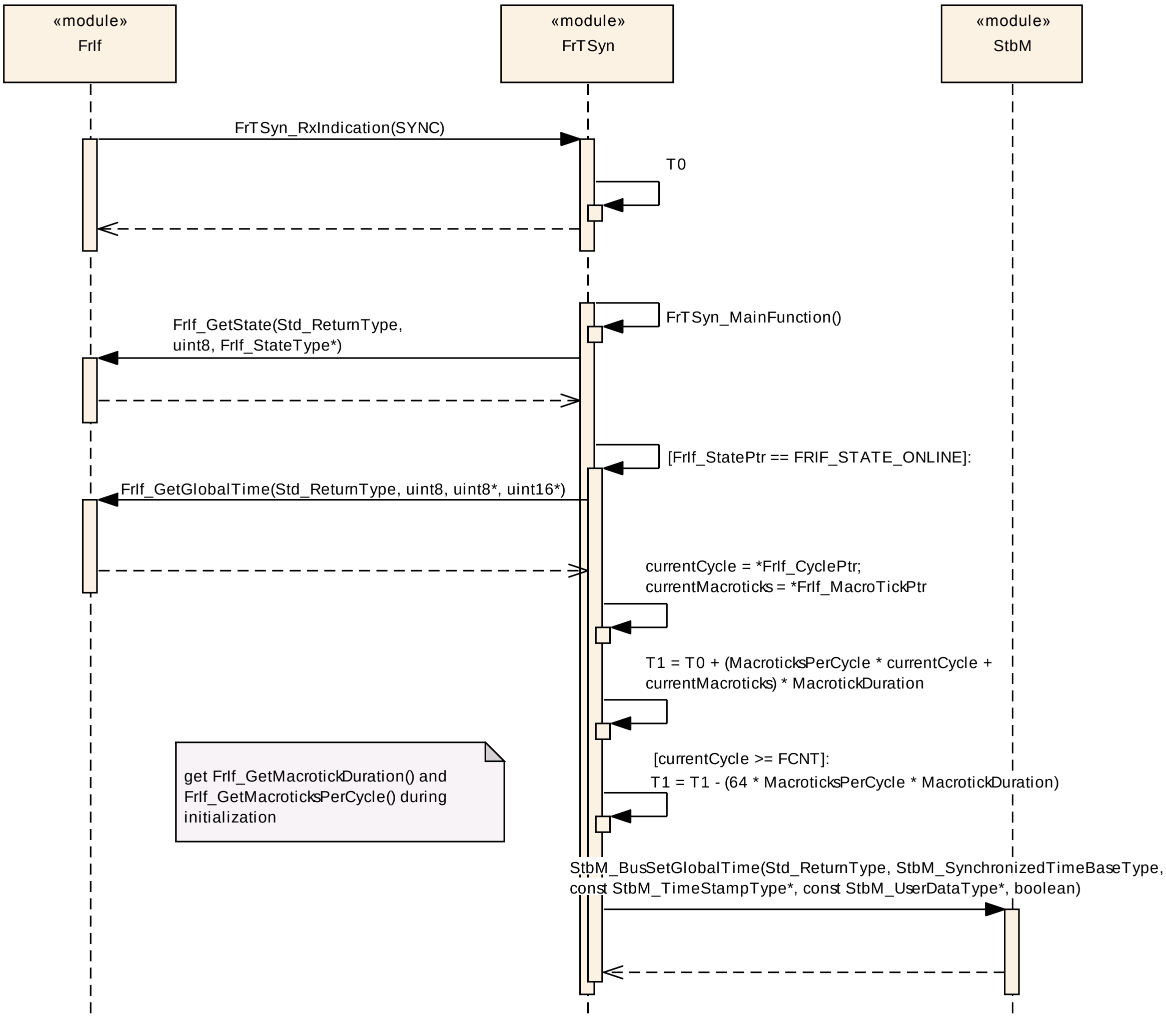

«module»
FrIf
«module»
FrTSyn
«module»
StbM
FrTSyn_RxIndication(SYNC)
T0
FrTSyn_MainFunction()
FrIf_GetState(Std_ReturnType, uint8, FrIf_StateType*)
[FrIf_StatePtr == FRIF_STATE_ONLINE]:
FrIf_GetGlobalTime(Std_ReturnType, uint8, uint8*, uint16*)
currentCycle = *FrIf_CyclePtr;
currentMacroticks = *FrIf_MacroTickPtr
T1 = T0 + (MacroticksPerCycle * currentCycle + currentMacroticks) * MacrotickDuration
[currentCycle >= FCNT]:
T1 = T1 - (64 * MacroticksPerCycle * MacrotickDuration)
get FrIf_GetMacrotickDuration() and FrIf_GetMacroticksPerCycle() during initialization
StbM_BusSetGlobalTime(Std_ReturnType, StbM_SynchronizedTimeBaseType, const StbM_TimeStampType*, const StbM_UserDataType*, boolean)

Figure 6: Slave FR SYNC

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 10.3 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_BSWGeneral*.

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

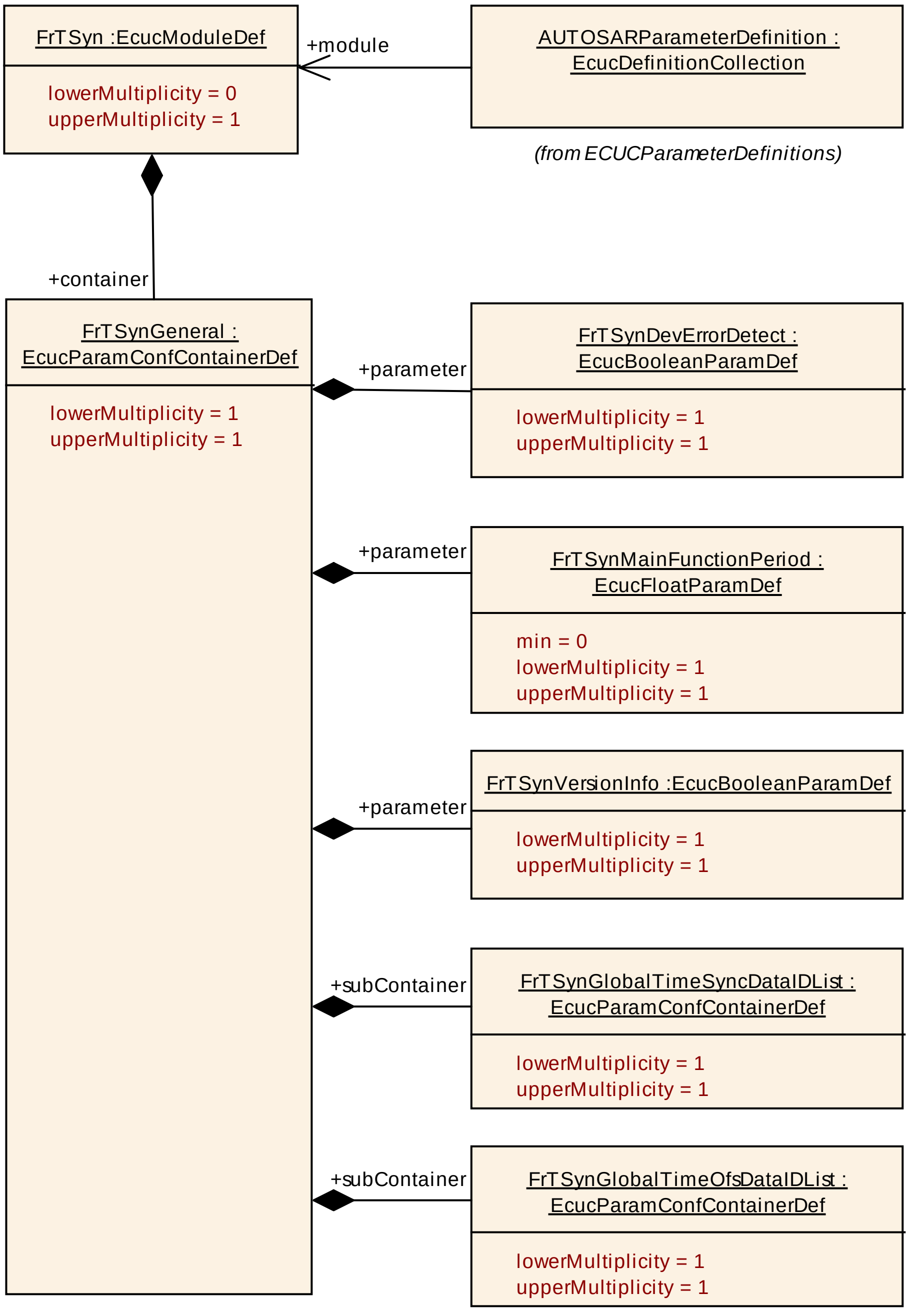
[SWS_FrTSyn_00076]
⌈The Time Synchronization over FlexRay shall support the configuration variants VARIANT-PRE-COMPILE, VARIANT-LINK-TIME, and VARIANT-POST-BUILD.
⌋(SRS_BSW_00396)

[SWS_FrTSyn_00077]
⌈The Time Synchronization over FlexRay shall support the configuration for Time Master, Time Slave and Time Gateway.
⌋(SRS_StbM_20038)

10.2.2 FrTSyn

SWS Item	**ECUC_FrTSyn_00001 :**
Module Name	*FrTSyn*
Module Description	This represents the specific configuration variant for the TSyn on Flexray.
Post-Build Variant Support	true

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

SWS Item	**ECUC_FrTSyn_00003 :**
Container Name	FrTSynGeneral
Description	This container holds the general parameters of the Flexray-specific Synchronized Time-base Manager
Configuration Parameters	

SWS Item	**ECUC_FrTSyn_00002 :**
Name	FrTSynDevErrorDetect
Description	Switches the Default Error Tracer (Det) detection and notification ON or OFF.

	• true: enabled (ON). • false: disabled (OFF).		
Multiplicity	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		

SWS Item	**ECUC_FrTSyn_00016 :**		
Name	FrTSynMainFunctionPeriod		
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		

SWS Item	**ECUC_FrTSyn_00019 :**		
Name	FrTSynVersionInfo		
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	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

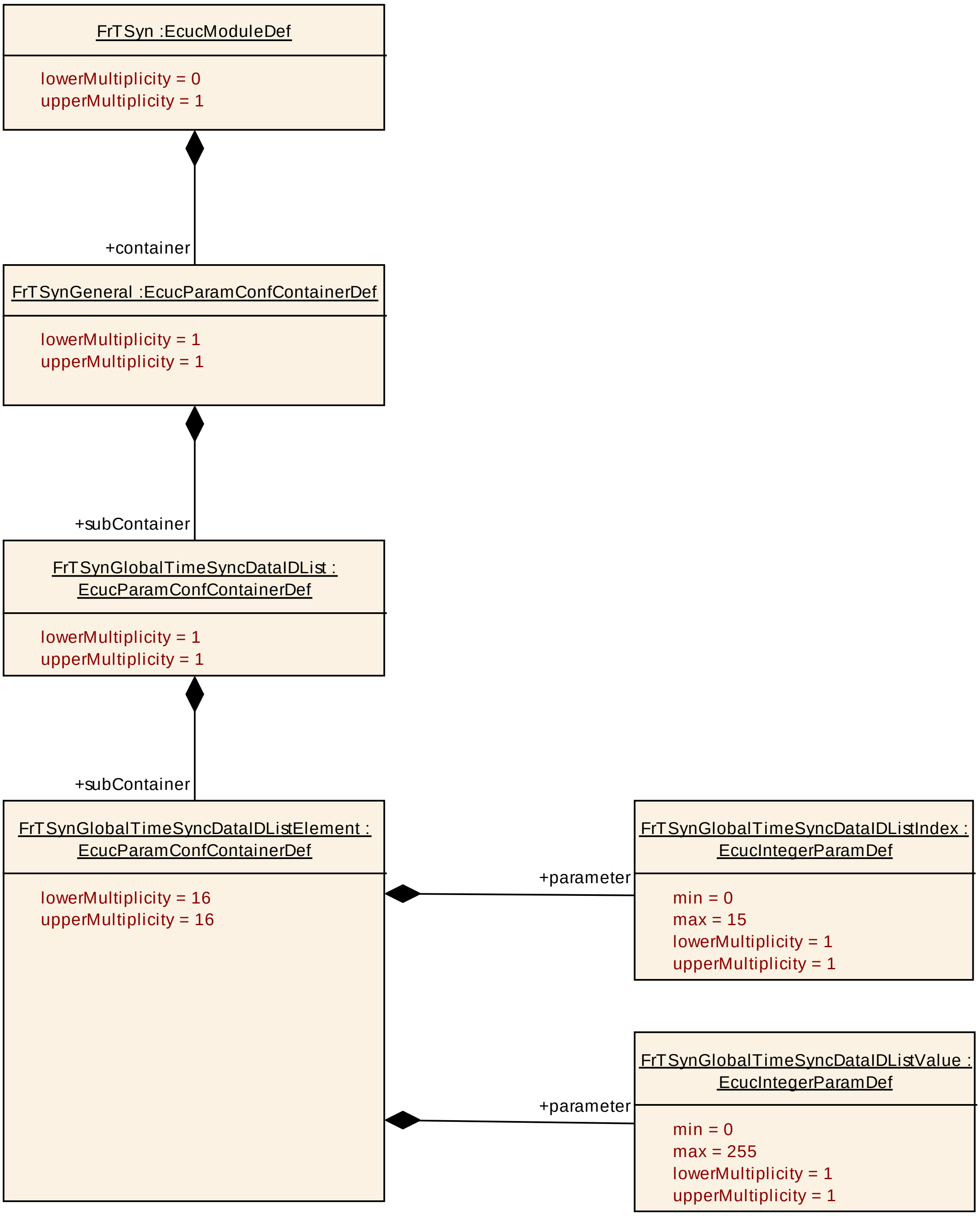
Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
FrTSynGlobalTimeOfsDataIDList	1	The DataIDList for OFS messages ensures the identification of data elements due to CRC calculation process.
FrTSynGlobalTimeSyncDataIDList	1	The DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation process.

10.2.4 FrTSynGlobalTimeSyncDataIDList

SWS Item	**ECUC_FrTSyn_00023 :**
Container Name	FrTSynGlobalTimeSyncDataIDList

Description	The DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation process.
Configuration 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 process.



10.2.5 FrTSynGlobalTimeSyncDataIDListElement

SWS Item	**ECUC_FrTSyn_00025 :**
Container Name	FrTSynGlobalTimeSyncDataIDListElement
Description	Element of the DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation process.
Configuration Parameters	

SWS Item	**ECUC_FrTSyn_00026 :**		
Name	FrTSynGlobalTimeSyncDataIDListIndex		
Description	Index of the DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation process.		
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		

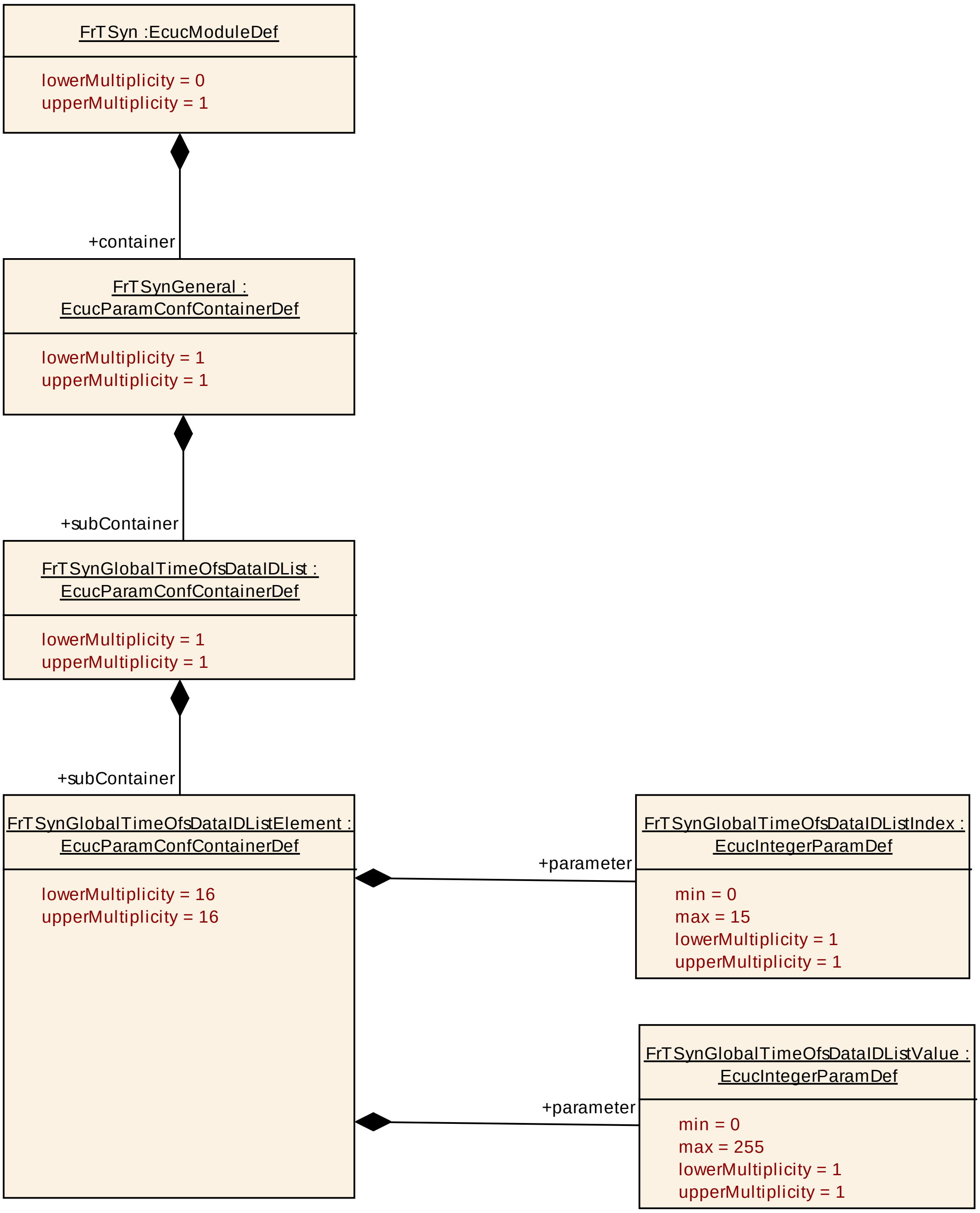
SWS Item	**ECUC_FrTSyn_00027 :**		
Name	FrTSynGlobalTimeSyncDataIDListValue		
Description	Value of the DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation process.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	*Pre-compile time*	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

10.2.6 FrTSynGlobalTimeOfsDataIDList

SWS Item	**ECUC_FrTSyn_00024 :**
Container Name	FrTSynGlobalTimeOfsDataIDList
Description	The DataIDList for OFS messages ensures the identification of data elements due to CRC calculation process.
Configuration Parameters	

Included Containers		
Container Name	*Multiplicity*	*Scope / Dependency*
FrTSynGlobalTimeOfsDataIDListElement	16	Element of the DataIDList for OFS messages ensures the identification of data elements due to CRC calculation process.



10.2.7 FrTSynGlobalTimeOfsDataIDListElement

SWS Item	**ECUC_FrTSyn_00028 :**
Container Name	FrTSynGlobalTimeOfsDataIDListElement
Description	Element of the DataIDList for OFS messages ensures the identification of data elements due to CRC calculation process.
Configuration Parameters	

SWS Item	**ECUC_FrTSyn_00029 :**		
Name	FrTSynGlobalTimeOfsDataIDListIndex		
Description	Index of the DataIDList for OFS messages ensures the identification of data elements due to CRC calculation process.		
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		

SWS Item	**ECUC_FrTSyn_00030 :**		
Name	FrTSynGlobalTimeOfsDataIDListValue		
Description	Value of the DataIDList for OFS messages ensures the identification of data elements due to CRC calculation process.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
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		

No Included Containers

10.2.8 FrTSynGlobalTimeDomain

SWS Item	**ECUC_FrTSyn_00004 :**
Container Name	FrTSynGlobalTimeDomain
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	

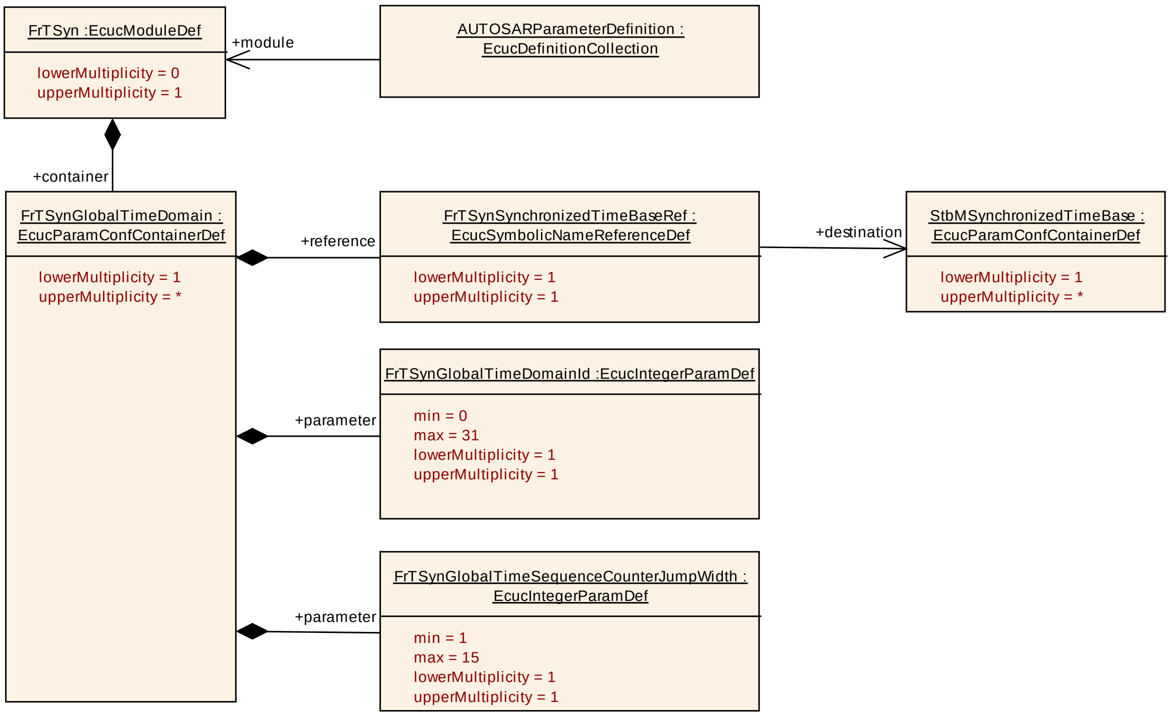
SWS Item	**ECUC_FrTSyn_00005 :**		
Name	FrTSynGlobalTimeDomainId		
Description	The global time domain ID.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 31		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: local

SWS Item	**ECUC_FrTSyn_00022 :**	
Name	FrTSynGlobalTimeSequenceCounterJumpWidth	
Description	The SequenceCounterJumpWidth specifies the maximum allowed gap of the Sequence Counter between two SYNC resp. two OFS messages.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	1 .. 15	
Default value	--	
Post-Build Variant Value	false	
Scope / Dependency	scope: local	

SWS Item	**ECUC_FrTSyn_00018 :**		
Name	FrTSynSynchronizedTimeBaseRef		
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		

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.



10.2.9 FrTSynGlobalTimeMaster

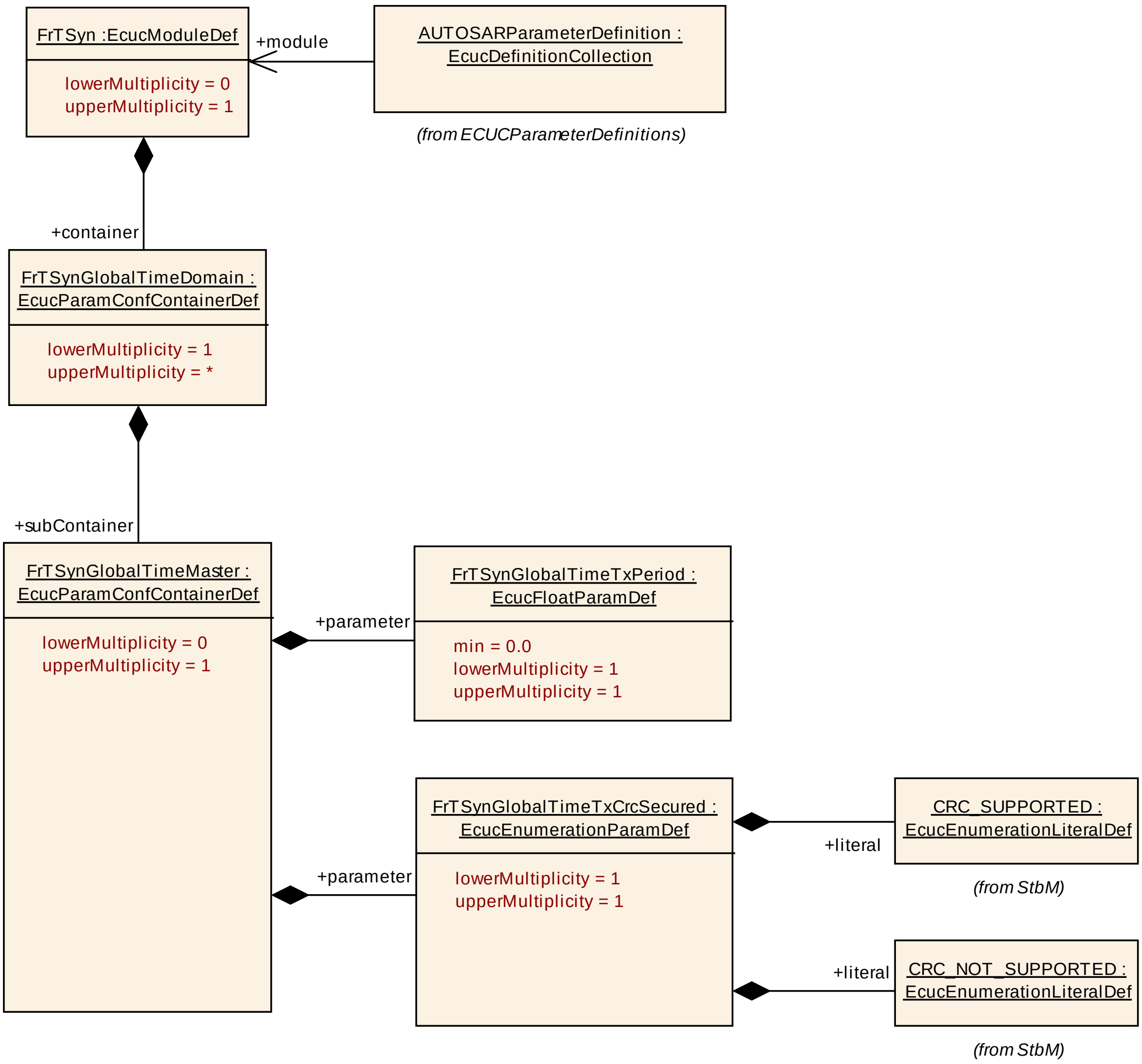
SWS Item	**ECUC_FrTSyn_00006 :**
Container Name	FrTSynGlobalTimeMaster
Description	Configuration of the global time master. Each global time domain is required to have exactly one global time master. This master may or may not exist on the configured ECU.
Configuration Parameters	

SWS Item	**ECUC_FrTSyn_00013 :**		
Name	FrTSynGlobalTimeTxCrcSecured		
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	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_FrTSyn_00014 :**
Name	FrTSynGlobalTimeTxPeriod
Description	This represents the TX period. 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		

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.



10.2.10 FrTSynGlobalTimeMasterPdu

SWS Item	**ECUC_FrTSyn_00008 :**

Container Name	FrTSynGlobalTimeMasterPdu
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	

SWS Item	**ECUC_FrTSyn_00007 :**		
Name	FrTSynGlobalTimeMasterConfirmationHandleId		
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		

SWS Item	**ECUC_FrTSyn_00020 :**		
Name	FrTSynGlobalTimePduRef		
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	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

No Included Containers

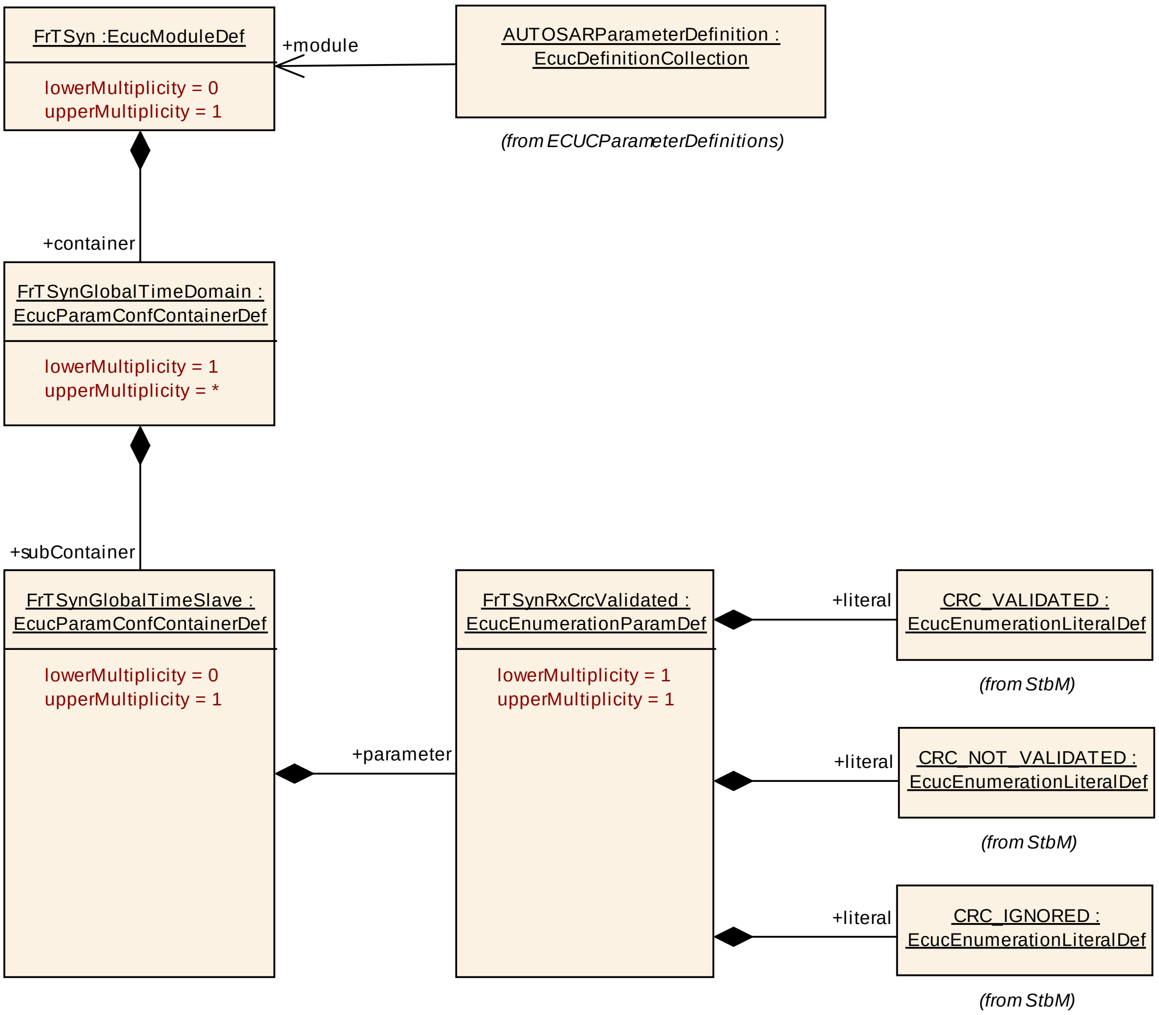
10.2.11 FrTSynGlobalTimeSlave

SWS Item	**ECUC_FrTSyn_00010 :**
Container Name	FrTSynGlobalTimeSlave
Description	This represents the time slave for the enclosing global time domain.
Configuration Parameters	

SWS Item	**ECUC_FrTSyn_00017 :**	
Name	FrTSynRxCrcValidated	
Description	This parameter controls whether or not CRC validation shall be supported.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	CRC_IGNORED	The FrTSyn accepts time synchronization messages with all Types. The CRC will be ignored.
	CRC_NOT_VALIDATED	The FrTSyn accepts time

		synchronization messages with Type equal to 0x10, 0x30 without validating the CRC. All other time synchronization messages are ignored.	
	CRC_VALIDATED	The FrTSyn accepts time synchronization messages with Type equal to 0x20, 0x40 with a correct CRC value. All other time synchronization messages are ignored.	
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
FrTSynGlobalTimeSlavePdu	1	This container carries all properties required to configure the PDU received by the time slave for the given global time domain.


FrTSyn :EcucModuleDef
lowerMultiplicity = 0
upperMultiplicity = 1
+module
AUTOSARParameterDefinition : EcucDefinitionCollection
(from ECUCParameterDefinitions)
+container
FrTSynGlobalTimeDomain : EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
+subContainer
FrTSynGlobalTimeSlave : EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+parameter
FrTSynRxCrcValidated : EcucEnumerationParamDef
lowerMultiplicity = 1
upperMultiplicity = 1
+literal
CRC_VALIDATED : EcucEnumerationLiteralDef
(from StbM)
+literal
CRC_NOT_VALIDATED : EcucEnumerationLiteralDef
(from StbM)
+literal
CRC_IGNORED : EcucEnumerationLiteralDef
(from StbM)

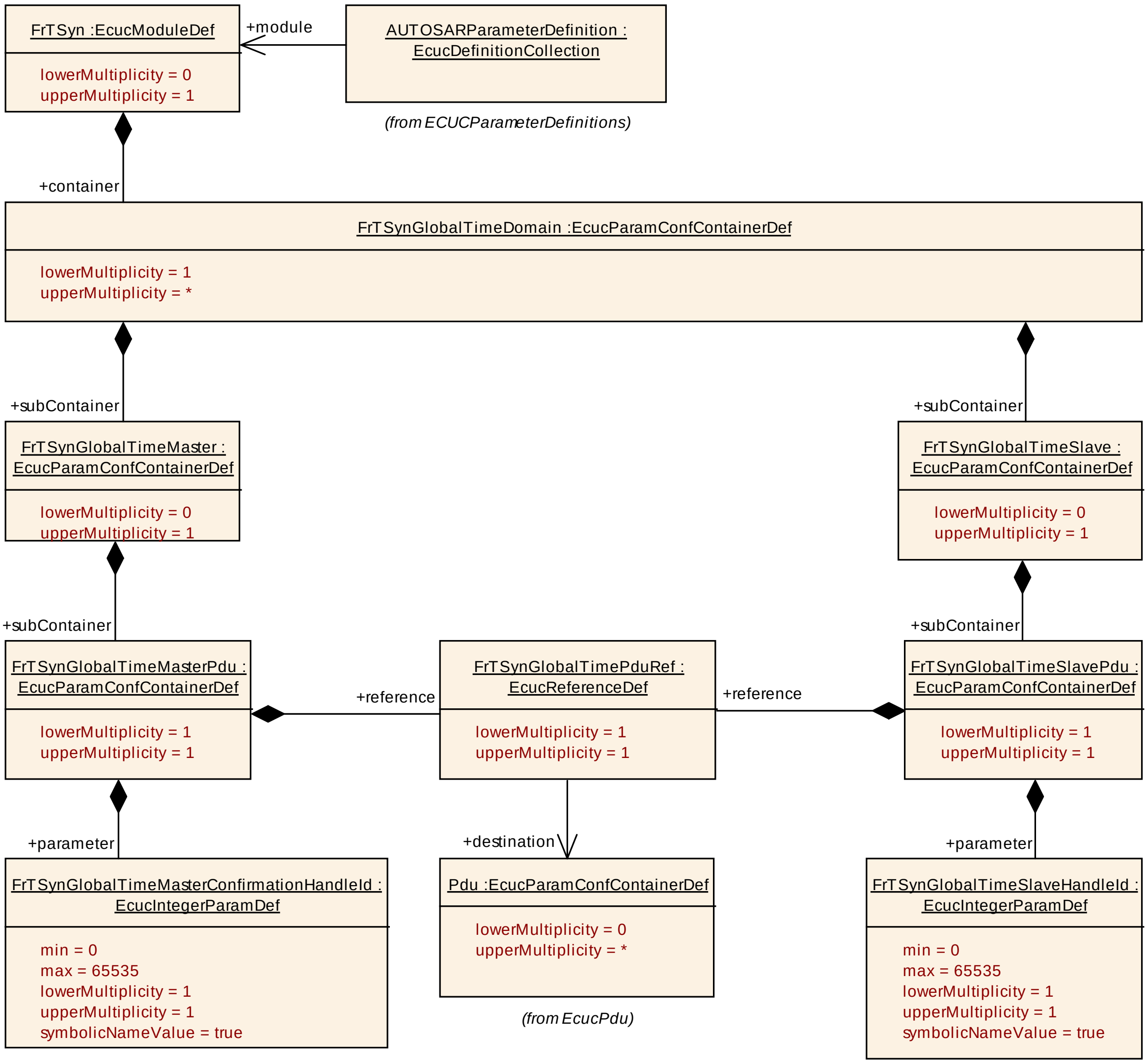
10.2.12 FrTSynGlobalTimeSlavePdu

SWS Item	**ECUC_FrTSyn_00012 :**
Container Name	FrTSynGlobalTimeSlavePdu
Description	This container carries all properties required to configure the PDU received by the time slave for the given global time domain.
Configuration Parameters	

SWS Item	**ECUC_FrTSyn_00011 :**		
Name	FrTSynGlobalTimeSlaveHandleId		
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		

SWS Item	**ECUC_FrTSyn_00021 :**		
Name	FrTSynGlobalTimePduRef		
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	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

No Included Containers



10.3 Published Information

For details, refer to the chapter 10.3 “Published Information” in *SWS_BSWGeneral*.