<table>
<thead>
<tr>
<th>Date</th>
<th>Release</th>
<th>Changed by</th>
<th>Change Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2017-12-08</td>
<td>4.3.1</td>
<td>AUTOSAR Release Management</td>
<td>• Minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation</td>
</tr>
<tr>
<td>2016-11-30</td>
<td>4.3.0</td>
<td>AUTOSAR Release Management</td>
<td>• Offset message formats changed</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Extended Offset message formats added</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Immediate Time Synchronization message transmission</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Various enhancements and corrections</td>
</tr>
<tr>
<td>2015-07-31</td>
<td>4.2.2</td>
<td>AUTOSAR Release Management</td>
<td>• CanTSyn_SetTransmissionMode changed to return &quot;void&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Minor corrections / clarifications / editorial changes</td>
</tr>
<tr>
<td>2014-10-31</td>
<td>4.2.1</td>
<td>AUTOSAR Release Management</td>
<td>• Initial Release</td>
</tr>
</tbody>
</table>
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.
Table of Contents

1 Introduction and functional overview ............................................................... 5
2 Acronyms, Abbreviations and Definitions .......................................................... 6
3 Related documentation ......................................................................................... 7
   3.1 Input documents ............................................................................................ 7
   3.2 Related specification ..................................................................................... 7
4 Constraints and assumptions ................................................................................ 8
   4.1 Limitations ..................................................................................................... 8
   4.2 Applicability to car domains ........................................................................ 8
5 Dependencies to other modules ............................................................................ 9
   5.1 File structure ................................................................................................ 10
      5.1.1 Code file structure ............................................................................... 10
      5.1.2 Header file structure .......................................................................... 10
6 Requirements traceability ..................................................................................... 12
7 Functional specification ......................................................................................... 15
   7.1 Overview ....................................................................................................... 15
   7.2 Module Handling ........................................................................................... 15
      7.2.1 Initialization ......................................................................................... 15
   7.3 Message Format ............................................................................................ 16
      7.3.1 SYNC and FUP Message ..................................................................... 17
      7.3.2 Offset Messages ............................................................................... 18
   7.4 Acting as Time Master .................................................................................... 21
      7.4.1 SYNC and FUP message processing .................................................... 22
      7.4.2 OFS message processing .................................................................... 23
      7.4.3 Transmission mode ............................................................................ 24
      7.4.4 Debounce Time .................................................................................. 25
      7.4.5 Immediate Time Synchronization .......................................................... 25
      7.4.6 Calculation and Assembling of Time Synchronization Messages ........... 27
   7.5 Acting as Time Slave ...................................................................................... 30
      7.5.1 SYNC and FUP message processing .................................................... 30
      7.5.2 OFS and OFNS message processing .................................................... 31
      7.5.3 Validation and Disassembling of Time Synchronization Messages ........ 33
   7.6 Error Classification ....................................................................................... 37
      7.6.1 Development Errors ............................................................................ 37
      7.6.2 Runtime Errors .................................................................................. 37
      7.6.3 Transient Faults .................................................................................. 37
      7.6.4 Production Errors ............................................................................... 37
      7.6.5 Extended Production Errors ................................................................. 38
8 API specification .................................................................................................... 39
   8.1 API ............................................................................................................... 39
      8.1.1 Imported types .................................................................................... 39
8.1.2 Type definitions ................................................................. 39
8.1.3 Function definitions .......................................................... 40
8.1.4 Call-back notifications ........................................................ 41
8.1.5 Scheduled functions .......................................................... 43
8.1.6 Expected Interfaces ........................................................... 44

9 Sequence diagrams .................................................................. 45
  9.1 StbM_GetCurrentTime <Master CAN SYNC/FUP> ....................... 45
  9.2 StbM_BusSetGlobalTime <Slave CAN SYNC/FUP> ...................... 46

10 Configuration specification ......................................................... 47
  10.1 How to read this chapter ....................................................... 47
  10.2 Containers and configuration parameters ................................. 48
    10.2.1 Variants ........................................................................ 48
    10.2.2 CanTSyn ....................................................................... 48
    10.2.3 CanTSynGeneral ............................................................. 49
    10.2.4 CanTSynGlobalTimeDomain ........................................... 50
    10.2.5 CanTSynGlobalTimeSyncDataIDList ................................ 53
    10.2.6 CanTSynGlobalTimeSyncDataIDListElement ...................... 54
    10.2.7 CanTSynGlobalTimeFupDataIDList .................................... 55
    10.2.8 CanTSynGlobalTimeFupDataIDListElement ....................... 56
    10.2.9 CanTSynGlobalTimeOfsDataIDList ................................... 57
    10.2.10 CanTSynGlobalTimeOfsDataIDListElement ....................... 58
    10.2.11 CanTSynGlobalTimeOfnsDataIDList ................................ 59
    10.2.12 CanTSynGlobalTimeOfnsDataIDListElement ...................... 60
    10.2.13 CanTSynGlobalTimeMaster ............................................ 61
    10.2.14 CanTSynGlobalTimeMasterPdu ....................................... 64
    10.2.15 CanTSynGlobalTimeSlave ............................................. 66
    10.2.16 CanTSynGlobalTimeSlavePdu ........................................ 69
  10.3 Published Information ......................................................... 70
1 Introduction and functional overview

The CanTSyn module handles the distribution of time information over CAN buses.

Just transmitting the time information from the master to the slaves in a broadcast CAN message has the disadvantage that the time value becomes inaccurate due to CAN specific effects like arbitration and BSW specific delays.

The concept proposes a two-step mechanism:

- In a first broadcast message (the so-called SYNC message), the second portion of the time information (t0r) is transmitted. The transmitting ECU, i.e. the Time Master, uses CAN low-level mechanisms like the “CAN transmit confirmation” to detect the point in time (t1r) when the message was actually transmitted, i.e. it takes a timestamp. A receiving ECU, i.e. the Time Slave, receives the message and uses CAN low-level mechanisms like the “CAN receive indication” to detect the point in time (t2r) when the message was actually received.

- In a second broadcast message (the so-called Follow-Up (FUP) message), the Time Master transmits the offset between the time information transmitted in the previous SYNC message and the actual detected transmission time. No timestamp is taken for the FUP message, neither on the transmitting nor on the receiving side.

- The Time Slave can now combine the information within the SYNC and within the FUP message and with its previously taken timestamp for the received SYNC message and determine the transmitted time information in a more precise way by just receiving one message and omitting timestamps.

The following Figure shows the CAN Time Synchronization mechanism.

Figure 1: CAN 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 [4].

<table>
<thead>
<tr>
<th>Abbreviation / Acronym:</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(G)TD</td>
<td>(Global) Time Domain</td>
</tr>
<tr>
<td>(G)TM</td>
<td>(Global) Time Master</td>
</tr>
<tr>
<td>&lt;Bus&gt;TSyn</td>
<td>A bus specific Time Synchronization module</td>
</tr>
<tr>
<td>CAN</td>
<td>Controller Area Network</td>
</tr>
<tr>
<td>CanTSyn</td>
<td>Time Synchronization module for CAN</td>
</tr>
<tr>
<td>CRC</td>
<td>Cyclic Redundancy Checksum</td>
</tr>
<tr>
<td>Debounce Time</td>
<td>Minimum gap between two Tx messages with the same PDU</td>
</tr>
<tr>
<td>DEM</td>
<td>Diagnostic Event Manager</td>
</tr>
<tr>
<td>DET</td>
<td>Default Error Tracer</td>
</tr>
<tr>
<td>DLC</td>
<td>Data Length Code</td>
</tr>
<tr>
<td>FUP message</td>
<td>Follow-Up message</td>
</tr>
<tr>
<td>OFNS message</td>
<td>Offset adjustment message</td>
</tr>
<tr>
<td>OFS message</td>
<td>Offset Synchronization message</td>
</tr>
<tr>
<td>StbM</td>
<td>Synchronized Time-Base Manager</td>
</tr>
<tr>
<td>SYNC message</td>
<td>Time Synchronization message</td>
</tr>
<tr>
<td>TG</td>
<td>Time Gateway</td>
</tr>
<tr>
<td>Timesync</td>
<td>Time Synchronization</td>
</tr>
<tr>
<td>TS</td>
<td>Time Slave</td>
</tr>
<tr>
<td>TSD</td>
<td>Time Sub-domain</td>
</tr>
</tbody>
</table>
3 Related documentation

3.1 Input documents

[1] Requirements on Synchronized Time-Base Manager
AUTOSAR_SRS_SynchronizedTimeBaseManager.pdf

[2] Layered Software Architecture
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf

AUTOSAR_SWS_BSWGeneral.pdf

AUTOSAR_SWS_SynchronizedTimeBaseManager.pdf

AUTOSAR_SWS_CRCLibrary.pdf

AUTOSAR_SWS_CANInterface.pdf

AUTOSAR_SWS_DefaultErrorTracer.pdf

[8] Specification of Basic Software Mode Manager
AUTOSAR_SWS_BSWModeManager.pdf

3.2 Related specification

AUTOSAR provides a General Specification on Basic Software (SWS BSW General [3]) which is also valid for CanTSyn.

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

4.1 Limitations

The current version of CanTSyn does not support hardware timestamp capabilities. The first consequence is that the Time Synchronization is less accurate due to Rx-/Tx-ISR latencies and execution time until the current time is retrieved. The second consequence is the need of interrupts in the CAN driver for the Global Time PDUs.

The Time Base in the SYNC and OFS messages is limited to 32 bit, wherefore the maximum supported time value is 4294967295 seconds ($2^{32}-1$).

Time Masters, Time Gateways 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 CAN (CanTSyn) has interfaces towards the Synchronized Time-Base Manager (StbM), the CAN Interface (CanIf), the Basic Software Mode Manager (BswM) and the Default Error Tracer (DET).

- StbM – Get and set the current time value
- CanIf – Receiving and transmitting messages

Figure 2: Module dependencies of the CanTSyn module
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

Besides the files defined in section 5.1.7 “Header file structure” of the SWS BSW General [3], the Time Synchronization over CAN needs to include the files defined below.

[SWS_CanTSyn_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 CAN.
Figure 3: File structure of CanTSyn
### Requirements traceability

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
<th>Satisfied by</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRS_BSW_00301</td>
<td>All AUTOSAR Basic Software Modules shall only import the necessary information</td>
<td>SWS_CanTSyn_00002</td>
</tr>
<tr>
<td>SRS_BSW_00323</td>
<td>All AUTOSAR Basic Software Modules shall check passed API parameters for validity</td>
<td>SWS_CanTSyn_00088, SWS_CanTSyn_00097, SWS_CanTSyn_00100, SWS_CanTSyn_00134</td>
</tr>
<tr>
<td>SRS_BSW_00337</td>
<td>Classification of development errors</td>
<td>SWS_CanTSyn_00005, SWS_CanTSyn_00097, SWS_CanTSyn_00100, SWS_CanTSyn_00134</td>
</tr>
<tr>
<td>SRS_BSW_00385</td>
<td>List possible error notifications</td>
<td>SWS_CanTSyn_00089</td>
</tr>
<tr>
<td>SRS_BSW_00456</td>
<td>A Header file shall be defined in order to harmonize BSW Modules</td>
<td>SWS_CanTSyn_00002</td>
</tr>
<tr>
<td>SRS_StbM_20018</td>
<td>The StbM shall initialize the Local Time Base with 0 at startup if configured as Time Slave</td>
<td>SWS_CanTSyn_00003, SWS_CanTSyn_00006</td>
</tr>
<tr>
<td>SRS_StbM_20019</td>
<td>The StbM shall initialize the Global Time Base with a configurable startup value if configured as Time Master</td>
<td>SWS_CanTSyn_00003, SWS_CanTSyn_00006</td>
</tr>
<tr>
<td>SRS_StbM_20032</td>
<td>The CAN Timesync Module shall provide the Time Base after reception of a valid Timesync messages</td>
<td>SWS_CanTSyn_00064, SWS_CanTSyn_00072, SWS_CanTSyn_00133, SWS_CanTSyn_00135</td>
</tr>
<tr>
<td>SRS_StbM_20033</td>
<td>The CAN Timesync module shall support means to protect the Time synchronization protocol</td>
<td>SWS_CanTSyn_00007, SWS_CanTSyn_00015, SWS_CanTSyn_00016, SWS_CanTSyn_00017, SWS_CanTSyn_00018, SWS_CanTSyn_00031, SWS_CanTSyn_00041, SWS_CanTSyn_00048, SWS_CanTSyn_00049, SWS_CanTSyn_00050, SWS_CanTSyn_00054, SWS_CanTSyn_00055, SWS_CanTSyn_00056, SWS_CanTSyn_00111, SWS_CanTSyn_00112, SWS_CanTSyn_00126, SWS_CanTSyn_00127, SWS_CanTSyn_00128, SWS_CanTSyn_00129</td>
</tr>
<tr>
<td>SRS_StbM_20036</td>
<td>The CAN Timesync module shall use the time measurement and synchronization protocol to transmit and receive an offset value</td>
<td>SWS_CanTSyn_00030, SWS_CanTSyn_00035, SWS_CanTSyn_00036, SWS_CanTSyn_00037, SWS_CanTSyn_00038, SWS_CanTSyn_00039, SWS_CanTSyn_00040, SWS_CanTSyn_00041, SWS_CanTSyn_00042, SWS_CanTSyn_00043, SWS_CanTSyn_00044, SWS_CanTSyn_00046, SWS_CanTSyn_00048, SWS_CanTSyn_00049, SWS_CanTSyn_00050, SWS_CanTSyn_00054, SWS_CanTSyn_00055, SWS_CanTSyn_00056, SWS_CanTSyn_00065, SWS_CanTSyn_00066, SWS_CanTSyn_00067, SWS_CanTSyn_00068, SWS_CanTSyn_00069, SWS_CanTSyn_00070, SWS_CanTSyn_00071, SWS_CanTSyn_00074, SWS_CanTSyn_00077, SWS_CanTSyn_00078, SWS_CanTSyn_00079, SWS_CanTSyn_00079, SWS_CanTSyn_00080</td>
</tr>
<tr>
<td>SRS_StbM_20037</td>
<td>The CAN Timesync module shall support user specific data within the time measurement and synchronization protocol</td>
<td>SWS_CanTSyn_00011, SWS_CanTSyn_00012, SWS_CanTSyn_00013, SWS_CanTSyn_00014</td>
</tr>
<tr>
<td>SRS_StbM_20038</td>
<td>The CAN Timesync configuration shall allow the CanTSyn to support different roles for a Time Base</td>
<td>SWS_CanTSyn_00108, SWS_CanTSyn_00135</td>
</tr>
<tr>
<td>SRS_StbM_20057</td>
<td>The StbM shall provide measurement data to the application</td>
<td>SWS_CanTSyn_00115, SWS_CanTSyn_00116</td>
</tr>
<tr>
<td>SRS_StbM_20068</td>
<td>The CAN Timesync module shall support classic CAN and CAN FD</td>
<td>SWS_CanTSyn_00010, SWS_CanTSyn_00015, SWS_CanTSyn_00016, SWS_CanTSyn_00017, SWS_CanTSyn_00018, SWS_CanTSyn_00036, SWS_CanTSyn_00041, SWS_CanTSyn_00055, SWS_CanTSyn_00071, SWS_CanTSyn_00072, SWS_CanTSyn_00077, SWS_CanTSyn_00085, SWS_CanTSyn_00111, SWS_CanTSyn_00112, SWS_CanTSyn_00130, SWS_CanTSyn_00131, SWS_CanTSyn_00132</td>
</tr>
</tbody>
</table>
7 Functional specification

This chapter defines the behavior of the Time Synchronization over CAN. 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 CAN is responsible to realize the CAN specific Time Synchronization protocol. Time Synchronization principles and common wording is described in [4].

7.2 Module Handling

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

[SWS_CanTSyn_00135]
If CanTSyn calls an API of the StbM, it shall use the Time Base ID of the Time Base referenced via the parameter CanTSynSynchronizedTimeBaseRef of the corresponding Time Domain.
J(SRS_StbM_20032, SRS_StbM_20038)

7.2.1 Initialization

The Time Synchronization over CAN is initialized via CanTSyn_Init(). Except for CanTSyn_GetVersionInfo() and CanTSyn_Init(), the API functions of the Time Synchronization over CAN may only be called when the module has been properly initialized.

[SWS_CanTSyn_00003]
A call to CanTSyn_Init() initializes all internal variables and sets the Time Synchronization over CAN to the initialized state.
J(SRS_StbM_20018, SRS_StbM_20019)

[SWS_CanTSyn_00005]
When DET reporting is enabled (see CanTSynDevErrorDetect), the Time Synchronization over CAN shall call Det_ReportError() with the error code CANTSYN_E_UNINIT when any API other than CanTSyn_GetVersionInfo() or CanTSyn_Init() is called in uninitialized state.
J(SRS_BSW_00337)

[SWS_CanTSyn_00006]
When CanTSyn_Init() is called in initialized state, the Time Synchronization over CAN shall re-initialize its internal variables.
J(SRS_StbM_20018, SRS_StbM_20019)
The Sequence Counter (SC) shall be initialized with 0.

7.3 Message Format

SYNC, FUP, OFS and OFNS messages are assigned to a dedicated message type “TimeSync”.

SYNC, FUP, OFS and OFNS messages of the same Time Domain share the same CAN ID by using a multiplexed signal group. For different Time Domains the same CAN ID may be used if Timesync messages are sent by the same Time Master or Time Gateway. For different Time Domains different CAN IDs shall be used if Timesync messages are sent by different Time Masters or Time Gateways. The multiplexer is located at Byte 0, named as “Type”.

The usage of a CRC is optional. To ensure a great variability between several time observing units, the configuration decides of how to handle CRC secured Timesync 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.

The byte order for time value signals in Time Synchronization messages is “Big Endian”.

The DLC of SYNC, FUP, OFS and OFNS messages is 8 for classic CAN.

The DLC of SYNC, FUP, OFS and OFNS messages is 16 for CAN FD if CanTSynUseExtendedMsgFormat is TRUE.

Depending on its type Time Synchronization messages may contain User Data according to the given message format.

User Data shall be read consistently from incoming Time Synchronization messages that contain User Data Fields.

User Data shall be written consistently to outgoing Time Synchronization messages that contain User Data Fields.
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.

7.3.1 SYNC and FUP Message

SYNC not CRC secured message format:
- Byte 0: Type = 0x10
- Byte 1: User Byte 1, 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: User Byte 0, default: 0
- Byte 4-7: SyncTimeSec = 32 bit LSB of the 48 bits seconds part of the time

If CanTSynUseExtendedMsgFormat = TRUE:
- Byte 8-15: reserved, always 0

FUP not CRC secured message format:
- Byte 0: Type = 0x18
- 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: reserved (Bit 7 to Bit 3), default: 0
  SGW (Bit 2)
  SyncToGTM = 0
  SyncToSubDomain = 1
  OVS = Overflow of seconds (Bit 1 to Bit 0)
- Byte 4-7: SyncTimeNSec = 32 Bit time value in nanoseconds

If CanTSynUseExtendedMsgFormat = TRUE:
- Byte 8-15: reserved, always 0
7.3.2 Offset Messages

Offset messages can be multiplexed with the Time Synchronization messages (using the same PDU, etc.).
For Classic CAN (CAN 2.0) two different Offset messages are used, OFS and OFNS. For both of them there are variants with and without a CRC field.
For CAN FD, if CanTSynUseExtendedMsgFormat is TRUE, the content of OFS and OFNS is merged into a single Extended OFS message (variants with and without a CRC field exist as well).

[SWS_CanTSyn_00132]
CanTSynUseExtendedMsgFormat shall always be FALSE for CAN 2.0 buses.
J(SRS_StbM_20068)

[SWS_CanTSyn_00130]
If CanTSynUseExtendedMsgFormat is FALSE, then the Normal Offset Message Format shall be used as specified in section 7.3.2.1.
J(SRS_StbM_20068)

[SWS_CanTSyn_00131]
If CanTSynUseExtendedMsgFormat is TRUE, then the Extended Offset Message Format shall be used as specified in section 7.3.2.2.
J(SRS_StbM_20068)

7.3.2.1 Normal Offset Messages

[SFS_CanTSyn_00126]
OFS not CRC secured message format:
Byte 0: Type = 0x34
Byte 1: User Byte 1, default: 0
Byte 2: \( D = \text{Time Domain 16 to 31 (Bit 7 to Bit 4)} \)
\( SC = \text{Sequence Counter (Bit 3 to Bit 0)} \)
Byte 3: User Byte 0, default: 0
Byte 4-7: \( OffSec = 32 \text{ Bit offset time value in seconds} \)
(SRS_StbM_20033, SRS_StbM_20036)

[SWS_CanTSyn_00127]]
OFNS not CRC secured message format:
Byte 0: \( \text{Type} = 0x3C \)
Byte 1: User Byte 2, default: 0
Byte 2: \( D = \text{Time Domain 16 to 31 (Bit 7 to Bit 4)} \)
\( SC = \text{Sequence Counter (Bit 3 to Bit 0)} \)
Byte 3: reserved (Bit 7 to Bit 1), default: 0
\( SGW (\text{Bit 0}) \)
\( SyncToGTM = 0 \)
\( SyncToSubDomain = 1 \)
Byte 4-7: \( OffNSec = 32 \text{ Bit offset time value in nanoseconds} \)
(SRS_StbM_20033, SRS_StbM_20036)

[SWS_CanTSyn_00128]]
OFNS CRC secured message format:
Byte 0: \( \text{Type} = 0x44 \)
Byte 1: \( CRC \)
Byte 2: \( D = \text{Time Domain 16 to 31 (Bit 7 to Bit 4)} \)
\( SC = \text{Sequence Counter (Bit 3 to Bit 0)} \)
Byte 3: User Byte 0, default: 0
Byte 4-7: \( OffSec = 32 \text{ Bit offset time value in seconds} \)
(SRS_StbM_20033, SRS_StbM_20036)

[SWS_CanTSyn_00129]]
OFNS CRC secured message format:
Byte 0: \( \text{Type} = 0x4C \)
Byte 1: \( CRC \)
Byte 2: \( D = \text{Time Domain 16 to 31 (Bit 7 to Bit 4)} \)
\( SC = \text{Sequence Counter (Bit 3 to Bit 0)} \)
Byte 3: reserved (Bit 7 to Bit 1), default: 0
\( SGW (\text{Bit 0}) \)
\( SyncToGTM = 0 \)
\( SyncToSubDomain = 1 \)
Byte 4-7: \( OffNSec = 32 \text{ Bit offset time value in nanoseconds} \)
(SRS_StbM_20033, SRS_StbM_20036)

7.3.2.2 Extended Offset messages

If \text{CanTSynUseExtendedMsgFormat} \text{ is TRUE}, the message layout of the Extended OFS message is as follows. A separate OFNS message is not required.

[SWS_CanTSyn_00111]]
OFNS not CRC secured message format for CAN FD PDUs:
Byte 0: \( Type = 0x54 \)
Byte 1: User Byte 2, 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: reserved (Bit 7 to Bit 1), default: 0
\( SGW (Bit 0) \)
\( SyncToGTM = 0 \)
\( SyncToSubDomain = 1 \)
Byte 4: User Byte 0, default: 0
Byte 5: User Byte 1, default: 0
Byte 6: reserved, default: 0
Byte 7: reserved, default: 0
Byte 8-11: \( OfsTimeSec = 32 \) Bit offset time value in seconds
Byte 12-15: \( OfsTimeNSec = 32 \) Bit offset time value in nanoseconds
\( \lfloor (SRS_StbM_20033, SRS_StbM_20036, SRS_StbM_20068) \rfloor \)

[SWS_CanTSyn_00112]
OFS CRC secured message format for CAN FD PDUs:
Byte 0: \( Type = 0x64 \)
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: reserved (Bit 7 to Bit 1), default: 0
\( SGW (Bit 0) \)
\( SyncToGTM = 0 \)
\( SyncToSubDomain = 1 \)
Byte 4: User Byte 0, default: 0
Byte 5: User Byte 1, default: 0
Byte 6: reserved, default: 0
Byte 7: reserved, default: 0
Byte 8-11: \( OfsTimeSec = 32 \) Bit offset time value in seconds
Byte 12-15: \( OfsTimeNSec = 32 \) Bit offset time value in nanoseconds
\( \lfloor (SRS_StbM_20033, SRS_StbM_20036, SRS_StbM_20068) \rfloor \)
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 4: Terminology Example
A master shall transmit SYNC, FUP, OFS and OFNS messages by calling
`CanIf_Transmit` with the PdulId derived via `CanTSynGlobalTimePduRef` of the corresponding Time Domain
(SRS_StbM_20031)

### 7.4.1 SYNC and FUP message processing

A Time Master shall start each Time Synchronization sequence for a Synchronized Time Base with a SYNC message.
(SRS_StbM_20031, SRS_StbM_20035)

A Time Master shall finish each Time Synchronization sequence for a Synchronized Time Base with a FUP message.
(SRS_StbM_20031, SRS_StbM_20035)

Any timeout while waiting for `CanTSyn_TxConfirmation()` function resets the state machine to start with a new SYNC transmission again.
(SRS_StbM_20034, SRS_StbM_20035)

For a Synchronized Time Base a Time Master is using a cyclic transmission of SYNC messages (according Figure 5: Master CAN SYNC/FUP) with
`CanTSynGlobalTimeTxPeriod` if the GLOBAL_TIME_BASE bit within the `timeBaseStatus` is set and
`CanTSynGlobalTimeTxPeriod` is unequal to 0 and if the associated cyclicMsgResumeCounter is not running (see 7.4.5).
(SRS_StbM_20031, SRS_StbM_20035)

The SYNC and FUP sequence shall not be interrupted, neither by Time Synchronization messages of the same Time Domain nor by Time Synchronization messages of other Time Domains if the same CAN ID is used for the Time Synchronization messages.
(SRS_StbM_20035)

Depending on `CanTSynGlobalTimeTxCrcSecured` the SYNC / FUP message shall be of type:

<table>
<thead>
<tr>
<th>CanTSynGlobalTimeTxCrcSecured</th>
<th>SYNC</th>
<th>FUP</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC_NOT_SUPPORTED</td>
<td>0x10</td>
<td>0x18</td>
</tr>
<tr>
<td>SYNC not CRC secured message</td>
<td></td>
<td>FUP not CRC secured message</td>
</tr>
<tr>
<td>CRC_SUPPORTED</td>
<td>0x20</td>
<td>0x28</td>
</tr>
</tbody>
</table>

---

- AUTOSAR confidential -
Specification of Time Synchronization over CAN
AUTOSAR CP Release 4.3.1

<table>
<thead>
<tr>
<th></th>
<th>SYNC CRC secured message</th>
<th>FUP CRC secured message</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(SRS_StbM_20033, SRS_StbM_20035)</td>
<td></td>
</tr>
</tbody>
</table>

[SWS_CanTSyn_00032]
A transmitter of FUP messages (Time Master) is using as trigger condition for SYNC to FUP that the debounceCounter value reaches 0 as described in 7.4.4.
(SRS_StbM_20031, SRS_StbM_20035)

[SWS_CanTSyn_00033]
Each transmission request of a SYNC message shall be monitored for a transmit confirmation timeout CanTSynMasterConfirmationTimeout (ECUC_CanTSyn_00020 : ). If the timeout occurs, the transmission request shall be revoked and no FUP message shall be sent.
(SRS_StbM_20034, SRS_StbM_20035)

7.4.2 OFS message processing

[SWS_CanTSyn_00035]
A Time Master shall start each Time Synchronization sequence for an Offset Time Base with an OFS message.
(SRS_StbM_20031, SRS_StbM_20036)

[SWS_CanTSyn_00036]
If CanTSynUseExtendedMsgFormat is FALSE, a Time Master shall finish each Time Synchronization sequence for an Offset Time Base with an OFNS message.
(SRS_StbM_20031, SRS_StbM_20036, SRS_StbM_20068)

Note: If CanTSynUseExtendedMsgFormat is TRUE, OFNS messages are not required.

[SWS_CanTSyn_00037]
Any Timeout while waiting for CanTSyn_TxConfirmation() function resets the state machine to start with a new OFS transmission again.
(SRS_StbM_20034, SRS_StbM_20036)

[SWS_CanTSyn_00038]
For an Offset Time Base the Time Master is using a cyclic transmission of OFS messages (CanTSynGlobalTimeTxPeriod (refer ECUC_CanTSyn_00017 : ) if the GLOBAL_TIME_BASE bit within the timeBaseStatus is set and CanTSynGlobalTimeTxPeriod is unequal to 0 and if the associated cyclicMsgResumeCounter is not running (see 7.4.5).
(SRS_StbM_20031, SRS_StbM_20036)

[SWS_CanTSyn_00039]
The OFS and OFNS sequence shall not be interrupted, neither by Time Synchronization messages of the same Time Domain nor by Time Synchronization

(SRS_StbM_20033, SRS_StbM_20035)
messages of other Time Domains if the same CAN ID is used for the Time Synchronization messages.

[SWS_CanTSyn_00040]
A transmitter of OFNS messages (Time Master) is using as trigger condition for OFS to OFNS that the debounceCounter value reaches 0 as described in 7.4.4.

[SWS_CanTSyn_00041]
Depending on CanTSynGlobalTimeTxCrcSupported (ECUC_CanTSyn_00015 : ) the OFS / OFNS message shall be of type:

<table>
<thead>
<tr>
<th>CanTSynGlobalTimeTxCrcSupported</th>
<th>OFS</th>
<th>OFNS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAN</td>
<td>CRC_NOT_SUPPORTED</td>
<td>0x34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OFS not CRC secured message</td>
</tr>
<tr>
<td></td>
<td>CRC_SUPPORTED</td>
<td>0x44</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OFS CRC secured message</td>
</tr>
<tr>
<td>CAN FD (CanTSynUseExtendedMsgFormat = TRUE)</td>
<td>CRC_NOT_SUPPORTED</td>
<td>0x54</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OFS not CRC secured message</td>
</tr>
<tr>
<td></td>
<td>CRC_SUPPORTED</td>
<td>0x64</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OFS CRC secured message</td>
</tr>
</tbody>
</table>

[SWS_CanTSyn_00042]
Each OFS transmission request shall be monitored for a transmit confirmation timeout CanTSynMasterConfirmationTimeout (ECUC_CanTSyn_00020 : ). If the timeout occurs, the transmission request shall be revoked and no OFNS message shall be sent.

[SWS_CanTSyn_00043]
If CanTSyn_SetTransmissionMode(Controller, Mode) is called and parameter Mode equals CANTSYN_TX_OFF, all transmit request from CanTSyn shall be omitted on this CAN channel.

[SWS_CanTSyn_00044]
If CanTSyn_SetTransmissionMode(Controller, Mode) is called and parameter Mode equals CANTSYN_TX_ON, all transmit request from CanTSyn on this CAN channel shall be able to be transmitted.
7.4.4 Debounce Time

The debounce time shall inhibit transmission bursts of a specific CAN PDU. Inhibiting transmission bursts of Timesync messages on a specific CAN bus is not possible if multiple PDUs are used for multiple Time Domains since there is no inter-PDU debounce time configurable within the CanTSyn module.

[SWS_CanTSyn_00123]
If CanTSynGlobalTimeDebounceTime (ECUC_CanTSyn_00045 : ) is greater than 0 for a Time Base, CanTSyn shall always do debouncing for the corresponding Timesync PDUs as described below, otherwise CanTSyn shall not do any debouncing.

J(SRS_StbM_20031)

[SWS_CanTSyn_00124]
CanTSynGlobalTimeDebounceTime (ECUC_CanTSyn_00045 : ) represents the debounce value of a PDU specific debounceCounter that shall be started after the Timesync PDU has been sent. CanTSyn shall decrement the debounceCounter value on each invocation of CanTSyn_MainFunction(), if no Timesync PDU is transmitted.

J(SRS_StbM_20031)

[SWS_CanTSyn_00125]
A new Timesync PDU shall only be sent if the corresponding debounceCounter has a value equal or less than 0.

J(SRS_StbM_20031)

Note: Since the decrement of the debounceCounter takes place in the CanTSyn_MainFunction() call but the start of the counter takes place when the Timesync PDU has been sent (either in the subsequent CanTSyn_MainFunction() call or in the transmit confirmation callback function) the effective debounce time will be equal or larger than CanTSynGlobalTimeDebounceTime. The extension of the debounce time shall be limited to the value of CanTSynMainFunctionPeriod.

7.4.5 Immediate Time Synchronization

In addition to the cyclic Timesync message transmission, an immediate message transmission might be required.
Depending on configuration, the CanTSyn module checks on each CanTSyn_MainFunction() call the necessity for a Timesync message transmission for each Time Base, where a Master Port belongs to.

[SWS_CanTSyn_00117]
If `CanTSynImmediateTimeSync (ECUC_CanTSyn_00043 : )` is set to TRUE for a Time Base, CanTSyn shall check on each `CanTSyn_MainFunction()` call by calling `StbM_GetTimeBaseUpdateCounter()`, if the `timeBaseUpdateCounter` of the corresponding Time Base has changed.

\[(SRS_StbM_20031)\]

[SWS_CanTSyn_00118]
If `CanTSynImmediateTimeSync (ECUC_CanTSyn_00043 : )` is set to TRUE and the `timeBaseUpdateCounter` of a Time Base has changed and the `GLOBAL_TIME_BASE` bit of the `timeBaseStatus` is set, CanTSyn shall trigger an immediate transmission of Time Synchronization messages for the corresponding Time Base.

\[(SRS_StbM_20031)\]

Note: `timeBaseStatus` can be obtained by `StbM_GetTimeBaseStatus()` or `StbM_GetCurrentTime()`.

Note: The debounceTimer as described in 7.4.4 shall always be considered.

[SWS_CanTSyn_00119]
If `CanTSynImmediateTimeSync (ECUC_CanTSyn_00043 : )` is set to TRUE, `cyclicMsgResumeCounter` and `CanTSynCyclicMsgResumeTime (ECUC_CanTSyn_00044 : )` shall be considered.

\[(SRS_StbM_20031)\]

[SWS_CanTSyn_00120]
`CanTSynCyclicMsgResumeTime (ECUC_CanTSyn_00044 : )` represents the timeout value of a `cyclicMsgResumeCounter` that shall be started when either a SYNC or OFS message has been sent immediately, asynchronous to the cyclic Timesync message transmission. `cyclicMsgResumeCounter` shall be decremented on each invocation of `CanTSyn_MainFunction()`, if no Timesync PDU is transmitted asynchronously.

\[(SRS_StbM_20031)\]

[SWS_CanTSyn_00121]
If the `cyclicMsgResumeCounter` has reached a value equal or less than zero, CanTSyn shall resume cyclic Timesync message transmission by sending either a SYNC or OFS message.

\[(SRS_StbM_20031)\]

[SWS_CanTSyn_00122]
If the `cyclicMsgResumeCounter` is started CanTSyn shall stop cyclic Timesync message transmission.

\[(SRS_StbM_20031)\]
7.4.6 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).

7.4.6.1 Global Time Calculation

[SWS_CanTSyn_00045]
The transmitter of a Synchronized Time Base (Time Master) shall perform the following steps to distribute the Synchronized Time Base exactly:

1. On transmission of SYNC message
   a. Get Synchronized Time Base value T0 via StbM_GetCurrentTime() and write second portion of T0 to SyncTimeSec
   b. Get raw time T0_raw for time measurement of transmission delay via StbM_GetCurrentTimeRaw()

2. On SYNC message TX confirmation (or inside the subsequent MainFunction call)
   a. Retrieve time difference T0_diff (calculated with T0_raw) of the transmission delay via StbM_GetCurrentTimeDiff()
   b. Calculate T4 for FUP message as T4 = (T0ns + T0_diff) with T0ns as nanosecond portion of T0

3. On transmission of FUP message
   a. Write second portion of T4 (T4 >= 1s) to OVS
   b. Write nanosecond portion of T4 to SyncTimeNSec

[SWS_StbM_20035]

With these steps, the Synchronized Time Base value at the transmitter side has been calculated (T0 + T4).

[SWS_CanTSyn_00046]
The transmitter of an Offset Time Base (Time Master) shall perform the following steps to distribute the Offset Time Base exactly:

1. Retrieve current Offset Time via StbM_GetOffset()
2. Write second portion of the Offset Time to OfsTimeSec
3. Write nanosecond portion of the Offset Time to OfsTimeNSec

[SRS_StbM_20036]

Note: OFS and OFNS messages shall not be time stamped.

7.4.6.2 OVS Calculation

[SWS_CanTSyn_00047]
OVS shall be set within FUP messages if the transmitter detects a nanosecond overflow greater than the defined range of StbM_TimeStampType.nanoseconds
whereas the left over part of seconds which does not fit into StbM_TimeStampType.nanoseconds shall be written to OVS.

7.4.6.3 SGW Calculation

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

7.4.6.4 Sequence Counter Calculation

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 and OFS messages and shall be incremented by 1 continuously on every transmission request of a SYNC or OFS message. It shall wrap around at 15 to 0 again.

The Sequence Counter (SC) value for a FUP message shall be set to the SC value of the corresponding SYNC message. The SC value for an OFNS message shall be set to the SC value of the corresponding OFS message.

7.4.6.5 CRC Calculation

The function Crc_CalculateCRC8H2F() as defined in [5] shall be used to calculate the CRC if configured.

The DataID shall be calculated as DataID = DataIDList[SC], where DataIDList (ECUC_CanTSyn_00024 : ECUC_CanTSyn_00025 : ECUC_CanTSyn_00026 : ECUC_CanTSyn_00041 : ) 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.

If CanTSynUseExtendedMsgFormat is FALSE, the CRC shall be calculated over Time Synchronization message Byte 2 to Byte 7 and DataID, where Byte 2 is applied first, followed by the other bytes in ascending order, and DataID last.
If `CanTSynUseExtendedMsgFormat` is `TRUE`, the `CRC` shall be calculated over Time Synchronization message `Byte 2 to Byte 15` and `DataID` for Extended Timesync message formats, where `Byte 2` is applied first, followed by the other bytes in ascending order, and `DataID` last.

\[(SRS_StbM_20033, SRS_StbM_20035, SRS_StbM_20036, SRS_StbM_20068)\]

### 7.4.6.6 Message Assembling

**[SWS_CanTSyn_00056]**

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

1. Calculate `OVS` (FUP only)
2. Calculate `SGW` (FUP, OFNS and Extended OFS)
3. Calculate `SC`
4. Copy all data to the appropriate position within the related message
5. Calculate `CRC` (configuration dependent)

\[(SRS_StbM_20033, SRS_StbM_20035, SRS_StbM_20036)\]
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 and FUP message processing

[SWS_CanTSyn_00057]
The CanTSyn shall only accept a SYNC message with Type equal to 0x20 and a correct CRC value if CanTSynRxCrcValidated is configured to CRC_VALIDATED.
](SRS_StbM_20034, SRS_StbM_20035)

[SWS_CanTSyn_00058]
The CanTSyn shall only accept a SYNC message with Type equal to 0x10 if CanTSynRxCrcValidated is configured to CRC_NOT_VALIDATED.
](SRS_StbM_20035)

[SWS_CanTSyn_00059]
The CanTSyn shall only accept a SYNC message with Type equal to 0x10 or 0x20 if CanTSynRxCrcValidated is configured to CRC_IGNORED.
](SRS_StbM_20035)

[SWS_CanTSyn_00109]
The CanTSyn shall only accept a FUP message with an identical Sequence Counter to the value of the corresponding SYNC message and Type equal to 0x28 and a correct CRC value if CanTSynRxCrcValidated is configured to CRC_VALIDATED.
](SRS_StbM_20034, SRS_StbM_20035)

[SWS_CanTSyn_00060]
The CanTSyn shall only accept a FUP message with an identical Sequence Counter to the value of the corresponding SYNC message and Type equal to 0x18 if CanTSynRxCrcValidated is configured to CRC_NOT_VALIDATED.
](SRS_StbM_20034, SRS_StbM_20035)

[SWS_CanTSyn_00061]
The CanTSyn shall only accept a FUP message with an identical Sequence Counter to the value of the corresponding SYNC message and Type equal to 0x18 or 0x28 if CanTSynRxCrcValidated is configured to CRC_IGNORED.
](SRS_StbM_20034, SRS_StbM_20035)
[SWS_CanTSyn_00110]
The CanTSyn shall only accept a FUP message with an identical Sequence Counter to the value of the corresponding SYNC message and Type equal to 0x18 or a FUP message with an identical sequence counter to the value of the corresponding SYNC message and Type equal to 0x28 and a correct CRC value if CanTSynRxCrValidated is configured to CRC_OPTIONAL.

J(SRS_StbM_20034, SRS_StbM_20035)

[SWS_CanTSyn_00063]
For each configured Time Slave (CanTSynGlobalTimeSlave) the CanTSyn module shall observe the reception timeout CanTSynGlobalTimeFollowUpTimeout (ECUC_CanTSyn_00006 :) between the SYNC and its FUP message. If the reception timeout occurs the sequence shall be reset (i.e. waiting for a new SYNC message).

J(SRS_StbM_20034, SRS_StbM_20035)

Note: The general timeout monitoring for the Time Base update is located in the StbM and not in the Timesync modules.

[SWS_CanTSyn_00064]
For valid FUP messages a new Global Time value shall be calculated and forwarded to the StbM module via StbM_BusSetGlobalTime() (according to Figure 6: Slave CAN SYNC/FUP).

J(SRS_StbM_20032, SRS_StbM_20034)

[SWS_CanTSyn_00115]
On an invocation of StbM_BusSetGlobalTime() the parameter PathDelay of the measureDataPtr structure shall be set to 0.

J(SRS_StbM_20057)

7.5.2 OFS and OFNS message processing

[SWS_CanTSyn_00065]
The CanTSyn shall only accept an OFS message with Type equal to 0x44 or 0x64 and a correct CRC value if CanTSynRxCrValidated is configured to CRC_VALIDATED.

J(SRS_StbM_20034, SRS_StbM_20036)

[SWS_CanTSyn_00066]
The CanTSyn shall only accept an OFS message with Type equal to 0x34 or 0x54 if CanTSynRxCrValidated is configured to CRC_NOT_VALIDATED.

J(SRS_StbM_20036)

[SWS_CanTSyn_00067]
The CanTSyn shall only accept an OFS message with Type equal to 0x34, 0x44, 0x54 or 0x64 if CanTSynRxCrValidated is configured to CRC_IGNORED.

J(SRS_StbM_20036)
The CanTSyn shall only accept an OFS message with Type equal to 0x34 or 0x54 or an OFS message with Type equal to 0x44 or 0x64 and a correct CRC value if CanTSynRxCrcValidated is configured to CRC_OPTIONAL.

([SWS_CanTSyn_00068])
The CanTSyn shall only accept an OFNS message with an identical Sequence Counter to the value of the corresponding OFS message and Type equal to 0x4C and a correct CRC value if CanTSynRxCrcValidated is configured to CRC_VALIDATED.

([SWS_CanTSyn_00069])
The CanTSyn shall only accept an OFNS message with an identical Sequence Counter to the value of the corresponding OFS message and Type equal to 0x3C if CanTSynRxCrcValidated is configured to CRC_NOT_VALIDATED.

([SWS_CanTSyn_00070])
The CanTSyn shall only accept an OFNS message with an identical Sequence Counter to the value of the corresponding OFS message and Type equal to 0x3C or 0x4C if CanTSynRxCrcValidated is configured to CRC_IGNORED.

([SWS_CanTSyn_00114])
The CanTSyn shall observe for each configured Time Slave (CanTSynGlobalTimeSlave) the reception timeout CanTSynGlobalTimeFollowUpTimeout (ECUC_CanTSyn_00006 :) between the OFS and its OFNS message. If the reception timeout occurs the sequence shall be reset (i.e. waiting for a new OFS message).

([SWS_CanTSyn_00071])
If CanSynUseExtendedMsgFormat is FALSE, the CanTSyn shall calculate a new Offset Time value (according to

Note: The general timeout monitoring for the Time Base update is located in the StbM and not in the Timesync modules.

([SWS_CanTSyn_00072])
For valid OFNS messages and if CanSynUseExtendedMsgFormat is FALSE, the CanTSyn shall calculate a new Offset Time value (according to
[SWS_CanTSyn_00074]) and forward it to the StbM module via StbM_BusSetGlobalTime().

If CanTSynUseExtendedMsgFormat is TRUE, the CanTSyn shall calculate a new Offset Time value (according to [SWS_CanTSyn_00074]) after receiving a valid OFS message and forward the new Offset Time value to the StbM module via StbM_BusSetGlobalTime().

J(SRS_StbM_20032, SRS_StbM_20034, SRS_StbM_20068)

[SWS_CanTSyn_00116]
On an invocation of StbM_BusSetGlobalTime() the parameter PathDelay of the measureDataPtr structure shall be set to 0.
J(SRS_StbM_20057)

7.5.3 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).

7.5.3.1 Global Time Calculation

[SWS_CanTSyn_00073]
The receiver of a Synchronized Time Base shall perform the following steps to retrieve the Synchronized Time Base exactly:

1. On SYNC message RX indication, which delivers Synchronized Time Base part T0, retrieve Local Time stamp T2_raw via StbM_GetCurrentTimeRaw()

2. On FUP message reception (either in RX indication or in the subsequent MainFunction invocation), which delivers Synchronized Time Base part T4 = (OVS + SyncTimeNSec), retrieve the time difference between current Local Time stamp T3_raw and time stamp of the previously received Synchronized Time Base T2_raw via StbM_GetCurrentTimeDiff(), which delivers T3_diff = (T3_raw - T2_raw)

3. Calculate Global Time Base to update the Time Slave's Local Time Base as:
   GlobalTimeBase = T3_diff + (T0 + T4).

J(SRS_StbM_20035)

Note: The calculation in step 3 shall happen as close as possible to taking the time stamp T3_raw

[SWS_CanTSyn_00074]
The receiver of an Offset Time Base shall perform the following steps to assemble the Offset Time:
1. Get second portion of the Offset Time out of $OfsTimeSec$
2. Get nanosecond portion of the Offset Time out of $OfsTimeNSec$

$(SRS_{StbM\_20036})$

**Note:** OFS and OFNS messages are not time stamped.

### 7.5.3.2 OVS Consideration

**[SWS_CanTSyn_00075]**

$OVS$ (FUP only) shall be considered on the receiver side to retrieve the second portion of the received Synchronized Time Base.

$(SRS_{StbM\_20035})$

### 7.5.3.3 SGW Calculation

**[SWS_CanTSyn_00133]**

If the $SGW$ value (FUP, OFNS and Extended OFS) is set to $SyncToSubDomain$, the $SYNC\_TO\_GATEWAY$ bit within $timeBaseStatus$ shall be set to $TRUE$. Otherwise, it shall be set to $FALSE$.

$(SRS_{StbM\_20032, SRS_{StbM\_20034})$

### 7.5.3.4 Sequence Counter Validation

**[SWS_CanTSyn_00076]**

The Sequence Counter of each $SYNC$ message must match to the Sequence Counter of the next incoming FUP message of the same Time Domain. Otherwise, the contents of the already received $SYNC$ message shall be discarded and the received FUP message shall be ignored.

$(SRS_{StbM\_20034, SRS_{StbM\_20035})$

**[SWS_CanTSyn_00077]**

If $CanTSynUseExtendedMsgFormat$ is $FALSE$, the Sequence Counter of each $OFS$ message must match to the Sequence Counter of the next incoming $OFNS$ message of the same Time Domain. If the $SC$s do not match, the received $OFNS$ message shall be ignored and the contents of the already received $OFS$ message shall be discarded.

$(SRS_{StbM\_20034, SRS_{StbM\_20036, SRS_{StbM\_20068})$

**[SWS_CanTSyn_00078]**

The Sequence Counter Jump Width between two consecutive $SYNC$ or two consecutive $OFS$ messages of the same Time Domain shall be greater than 0 and smaller than or equal to $CanTSynGlobalTimeSequenceCounterJumpWidth$. Otherwise, a $TimeSlave$ shall ignore the respective $SYNC$ / $OFS$ message.

The $CanTSynGlobalTimeSequenceCounterJumpWidth$ value 0 is not allowed.

$(SRS_{StbM\_20034, SRS_{StbM\_20035, SRS_{StbM\_20036})$

**[SWS_CanTSyn_00079]**
At Startup or if a Time Base update timeout has been detected (TIMEOUT bit set in Time Base synchronization status timeBaseStatus), a Time Slave shall not check the Sequence Counter of the 1st received SYNC (or OFS) message per Time Domain against the defined Sequence Counter Jump Width.

\[ (SRS\_StbM\_20034, SRS\_StbM\_20035, SRS\_StbM\_20036) \]

**Note:** There are scenarios when it makes sense to skip the check of the Sequence Counter Jump Width, e.g. at startup (Time Slaves start asynchronously to the Time Master) or after a message timeout to allow for Sequence Counter (re-)synchronization. In case of a timeout the error has been detected already by the timeout monitoring, there is no benefit in generating a subsequent error by the jump width check.

### 7.5.3.5 CRC Validation

**[SWS\_CanTSyn\_00080]**

The function `Crc\_CalculateCRC8H2F()` as defined in [5] shall be used to validate the CRC if configured.

\[ (SRS\_StbM\_20034, SRS\_StbM\_20035, SRS\_StbM\_20036) \]

**[SWS\_CanTSyn\_00084]**

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

\[ (SRS\_StbM\_20034, SRS\_StbM\_20035) \]

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

**[SWS\_CanTSyn\_00085]**

If `CanTSynUseExtendedMsgFormat` is FALSE, the CRC shall be calculated over Time Synchronization message Byte 2 to Byte 7 and DataID, where Byte 2 is applied first, followed by the other Bytes in ascending order, and DataID last.

If `CanTSynUseExtendedMsgFormat` is TRUE, the CRC shall be calculated over Time Synchronization message Byte 2 to Byte 15 and DataID for Extended Timesync message formats, where Byte 2 is applied first, followed by the other bytes in ascending order, and DataID last.

\[ (SRS\_StbM\_20034, SRS\_StbM\_20035, SRS\_StbM\_20036, SRS\_StbM\_20068) \]

### 7.5.3.6 Message Disassembling

**[SWS\_CanTSyn\_00086]**

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

1. `Type` matches depending on the `CanTSynRxCrcValidated` parameter
2. `SC` value is within the accepted range (refer to [SWS\_CanTSyn\_00078] and [SWS\_CanTSyn\_00079])
3. `D` matches to the defined Time Domain range for each `Type`
4. \( D \) matches to one of the configured Time Domains (given by parameter CanTSynGlobalTimeDomainId)

5. \( \text{SyncTimeNSec} \) (FUP / OFNS / Extended OFS only) matches the defined range of StbM_TimeStampType.nanoseconds.

6. \( \text{CRC} \) (including DataID) matches depending on the CanTSynRxCrValidated parameter

\[ \text{[SWS_CanTSyn_00087]} \]
For each received Time Synchronization message the CanTSyn shall disassemble the message after successful validation (refer to \[\text{SWS_CanTSyn_00086}\]).

\[ \text{[SWS_CanTSyn_00087]} \]
### 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_CanTSyn_00088]

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

\[(SRS_StbM_20034, SRS_BSW_00323)\]

#### 7.6.1 Development Errors

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

[SWS_CanTSyn_00089]

CanTSyn shall use the following errors:

<table>
<thead>
<tr>
<th>Type or error</th>
<th>Related error code</th>
<th>Value [hex]</th>
</tr>
</thead>
<tbody>
<tr>
<td>API service called with wrong PDU or SDU</td>
<td>CANTSYN_E_INVALID_PDUID</td>
<td>0x01</td>
</tr>
<tr>
<td>API service used in un-initialized state</td>
<td>CANTSYN_E_UNINIT</td>
<td>0x02</td>
</tr>
<tr>
<td>A pointer is NULL</td>
<td>CANTSYN_E_NULL_POINTED</td>
<td>0x03</td>
</tr>
<tr>
<td>CanTSyn initialization failed</td>
<td>CANTSYN_E_INIT_FAILED</td>
<td>0x04</td>
</tr>
<tr>
<td>API called with invalid parameter</td>
<td>CANTSYN_E_PARAM</td>
<td>0x05</td>
</tr>
<tr>
<td>Invalid Controller index</td>
<td>CANTSYN_E_INV_CTRL_IDX</td>
<td>0x06</td>
</tr>
</tbody>
</table>

\[(SRS_BSW_00385)\]

#### 7.6.2 Runtime Errors

No Runtime Errors defined.

#### 7.6.3 Transient Faults

No Transient Faults defined.

#### 7.6.4 Production Errors

No Production Errors defined.
7.6.5 Extended Production Errors

No Extended Production Errors defined.
8  API specification

8.1  API

8.1.1 Imported types

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

8.1.2 Type definitions

8.1.2.1 CanTSyn_ConfigType

8.1.2.2 CanTSyn_TransmissionModeType
8.1.3 Function definitions

8.1.3.1 CanTSyn_Init

**Description:** Handles the enabling and disabling of the transmission mode

(SRS_StbM_20035)

CANTSYN_E_INIT_FAILED is reported as specified in [reference to SWS BSW General] by SWS_BSW_00050.

See section 7.2.1 for details.

8.1.3.2 CanTSyn_GetVersionInfo

**Description:** Returns the version information of this module.

(SRS_StbM_20035)

8.1.3.3 CanTSyn_SetTransmissionMode

**Description:**

(SRS_StbM_20035)
### CanTSyn_SetTransmissionMode

<table>
<thead>
<tr>
<th>Service ID[hex]</th>
<th>0x03</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sync/Async</td>
<td>Synchronous</td>
</tr>
<tr>
<td>Reentrancy</td>
<td>Non Reentrant</td>
</tr>
<tr>
<td>Parameters (in)</td>
<td>CtrlIdx, Mode</td>
</tr>
<tr>
<td>Parameters (inout)</td>
<td>None</td>
</tr>
<tr>
<td>Parameters (out)</td>
<td>None</td>
</tr>
<tr>
<td>Return value</td>
<td>None</td>
</tr>
</tbody>
</table>

**Description:**
This API is used to turn on and off the TX capabilities of the CanTSyn.

[SWS_CanTSyn_00134]

The function `CanTSyn_SetTransmissionMode()` shall inform the DET, if development error detection is enabled (`CanTSynDevErrorDetect` is set to TRUE) and if function call has failed because of the following reasons:
- Invalid CtrlIdx (`CANTSYN_E_INV_CTRL_IDX`)
- Invalid Mode (`CANTSYN_E_PARAM`)

[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 `CanTSyn_Cbk.h`.

#### 8.1.4.1 CanTSyn_RxIndication

```
void CanTSyn_RxIndication(
    PduIdType RxPduId,
    const PduInfoType* PduInfoPtr
);
```

<table>
<thead>
<tr>
<th>Service ID[hex]</th>
<th>0x42</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sync/Async</td>
<td>Synchronous</td>
</tr>
<tr>
<td>Reentrancy</td>
<td>Reentrant for different PduIds. Non reentrant for the same PduId.</td>
</tr>
<tr>
<td>Parameters (in)</td>
<td>RxPduId, PduInfoPtr</td>
</tr>
<tr>
<td>Parameters (inout)</td>
<td>None</td>
</tr>
<tr>
<td>Parameters (out)</td>
<td>None</td>
</tr>
<tr>
<td>Return value</td>
<td>None</td>
</tr>
</tbody>
</table>

**Description:**
Indication of a received PDU from a lower layer communication interface module.

[SWS_CanTSyn_00096]
Note: The callback function `CanTSyn_RxIndication()` called by the CAN Interface and implemented by the CanTSyn module. It is called in case of a receive indication event of the CAN Driver.

[SWS_CanTSyn_00097]
The callback function `CanTSyn_RxIndication()` shall inform the DET, if development error detection is enabled (CanTSynDevErrorDetect is set to TRUE) and if function call has failed because of the following reasons:

- Invalid PDU ID (`CNTSYN_E_INVALID_PDUID`)
- `PduInfoPtr` or `SduDataPtr` equals `NULL_PTR` (`CNTSYN_E_NULL_POINTER`)

(SRS_BSW_00323, SRS_BSW_00337)

**Caveats** of `CanTSyn_RxIndication()`:

- Until this service returns, the CAN Interface will not access `canSduPtr`. The `canSduPtr` is only valid and can be used by upper layers until the indication returns. The CAN Interface guarantees that the number of configured bytes for this `CanTSynRxPduId` is valid. The call context is either on interrupt level (interrupt mode) or on task level (polling mode). This callback service is re-entrant for multiple CAN controller usage.

  **Note:** Using polling mode as call context significantly increases the latency and thus reduces the precision. It is therefore highly recommended to only use interrupt mode.

- The CanTSyn module is initialized correctly.

8.1.4.2 CanTSyn_TxConfirmation

[SWS_CanTSyn_00099] |

<table>
<thead>
<tr>
<th>Service name:</th>
<th>CanTSyn_TxConfirmation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax:</td>
<td><code>void CanTSyn_TxConfirmation(</code></td>
</tr>
<tr>
<td></td>
<td><code>PduIdType TxPduId,</code></td>
</tr>
<tr>
<td></td>
<td><code>Std_ReturnType result</code></td>
</tr>
<tr>
<td>Service ID[hex]:</td>
<td>0x40</td>
</tr>
<tr>
<td>Sync/Async:</td>
<td>Synchronous</td>
</tr>
<tr>
<td>Reentrancy:</td>
<td>Reentrant for different Pduls. Non reentrant for the same Pdul.</td>
</tr>
<tr>
<td>Parameters (in):</td>
<td><code>TxPduId</code> ID of the PDU that has been transmitted.</td>
</tr>
<tr>
<td></td>
<td><code>result</code> <code>E_OK</code>: The PDU was transmitted. <code>E_NOT_OK</code>: Transmission of the PDU failed.</td>
</tr>
<tr>
<td>Parameters (inout):</td>
<td>None</td>
</tr>
<tr>
<td>Parameters (out):</td>
<td>None</td>
</tr>
<tr>
<td>Return value:</td>
<td>None</td>
</tr>
<tr>
<td>Description:</td>
<td>The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.</td>
</tr>
</tbody>
</table>

(SRS_StbM_20035)

**Note:** The callback function `CanTSyn_TxConfirmation()` is called by the CAN Interface and implemented by the CanTSyn module.
The callback function `CanTSyn_TxConfirmation()` shall inform the DET, if development error detection is enabled (`CanTSynDevErrorDetect` is set to `TRUE`) and if the function call has failed because of the following reason:

- Invalid PDU ID (`CANTSYN_E_INVALID_PDUID`), i.e., a PDU ID not configured by parameter `CanTSynGlobalTimeMasterConfirmationHandleId`

*Caveats of `CanTSyn_TxConfirmation()`:

- The call context is either on interrupt level (interrupt mode) or on task level (polling mode). This callback service is re-entrant for multiple CAN controller usage.
  
  **Note:** Using polling mode as call context significantly increases the latency and thus reduces the precision. It is therefore highly recommended to only use interrupt mode.

- The CanTSyn module is initialized correctly.

### 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 CanTSyn>MainFunction

*Service name:* CanTSyn_MainFunction

*Syntax:* `void CanTSyn_MainFunction(

    void )`

*Service ID[hex]:* 0x06

*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. Timesync message transmission

*(SRS_StbM_20035)*

The frequency of invocations of `CanTSyn_MainFunction()` is determined by the configuration parameter `CanTSynMainFunctionPeriod` (refer to `ECUC_CanTSyn_00019 :`).

*(SRS_StbM_20035)*
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_CanTSyn_00105] [ ][ ]

<table>
<thead>
<tr>
<th>API function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StbM_GetCurrentTimeDiff</td>
<td>Returns time difference of the nanoseconds part of the Virtual Local Time of the referenced Time Base minus the time given by the parameter givenTimeStamp.</td>
</tr>
<tr>
<td>StbM_GetCurrentTimeRaw</td>
<td>Returns nanosecond part of the Virtual Local Time of the referenced Time Base.</td>
</tr>
</tbody>
</table>

] (SRS_StbM_20035)

8.1.6.2 Optional Interfaces

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

[SWS_CanTSyn_00106] [ ][ ]

<table>
<thead>
<tr>
<th>API function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CanIf_Transmit</td>
<td>Requests transmission of a PDU.</td>
</tr>
<tr>
<td>Crc_CalculateCRC8H2F</td>
<td>This service makes a CRC8 calculation with the Polynomial 0x2F on Crc_Length</td>
</tr>
<tr>
<td>Det_ReportError</td>
<td>Service to report development errors.</td>
</tr>
<tr>
<td>StbM_BusSetGlobalTime</td>
<td>Allows the Time Base Provider Modules to forward a new Global Time value to the StbM, which has been received from a bus.</td>
</tr>
<tr>
<td>StbM_GetCurrentTime</td>
<td>Returns a time value (Local Time Base derived from Global Time Base) in standard format.</td>
</tr>
<tr>
<td>StbM_GetOffset</td>
<td>Allows the Timesync Modules to get the current Offset Time and User Data.</td>
</tr>
<tr>
<td>StbM_GetTimeBaseStatus</td>
<td>Returns the detailed status of the Time Base. For Offset Time Bases the status of the Offset Time Base itself and the status of the underlying Synchronized Time Base is returned.</td>
</tr>
<tr>
<td>StbM_GetTimeBaseUpdateCounter</td>
<td>Allows the Timesync Modules to detect, whether a Time Base should be transmitted immediately in the subsequent &lt;Bus&gt;TSyn_MainFunction() cycle.</td>
</tr>
</tbody>
</table>

] (SRS_StbM_20035)
9 Sequence diagrams

9.1 StbM_GetCurrentTime <Master CAN SYNC/FUP>

![Sequence diagram for StbM_GetCurrentTime](image)

**Figure 5: Master CAN SYNC/FUP**
9.2 StbM_BusSetGlobalTime <Slave CAN SYNC/FUP>

![Diagram of StbM_BusSetGlobalTime](image)

_**Figure 6: Slave CAN SYNC/FUP**_
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 CAN.

Section 10.2.16 specifies published information of the Time Synchronization over CAN.

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 CAN. The detailed meaning of the parameters is described in chapters 7 and 8.

10.2.1 Variants

[SWS_CanTSyn_00108]
The Time Synchronization over CAN shall support the configuration for Time Master, Time Slave and Time Gateway.
(SRS_StbM_20038)

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

10.2.2 CanTSyn

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC_CanTSyn_00001 :</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Module Name</strong></td>
<td>CanTSyn</td>
</tr>
<tr>
<td><strong>Module Description</strong></td>
<td>Configuration of the Synchronized Time-base Manager (StbM) module with respect to global time handling on CAN.</td>
</tr>
<tr>
<td><strong>Post-Build Variant Support</strong></td>
<td>true</td>
</tr>
<tr>
<td><strong>Supported Config Variants</strong></td>
<td>VARIANT-POST-BUILD, VARIANT-PRE-COMPILE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Included Containers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Container Name</strong></td>
</tr>
<tr>
<td>CanTSynGeneral</td>
</tr>
<tr>
<td>CanTSynGlobalTimeDomain</td>
</tr>
</tbody>
</table>
10.2.3 CanTSynGeneral

**SWS Item** ECUC_CanTSyn_00003 :
**Container Name** CanTSynGeneral
**Description** This container holds the general parameters of the CAN-specific Synchronized Time-base Manager

**SWS Item** ECUC_CanTSyn_00002 :
**Name** CanTSynDevErrorDetect
**Parent Container** CanTSynGeneral
**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
**SWS Item**  
**ECUC_CanTSyn_00019 :**

**Name**  
CanTSynMainFunctionPeriod

**Parent Container**  
CanTSynGeneral

**Description**  
Schedule period of the main function CanTSyn_MainFunction. Unit: [s].

**Multiplicity**  
1

**Type**  
EcucFloatParamDef

**Range**  
]0 .. INF[

**Default value**  
--

**Post-Build Variant Value**  
false

**Value Configuration Class**

<table>
<thead>
<tr>
<th>Pre-compile time</th>
<th>Link time</th>
<th>Post-build time</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

**Value Configuration Class**

<table>
<thead>
<tr>
<th>Pre-compile time</th>
<th>Link time</th>
<th>Post-build time</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

**Scope / Dependency**  
scope: local

**SWS Item**  
**ECUC_CanTSyn_00023 :**

**Name**  
CanTSynVersionInfoApi

**Parent Container**  
CanTSynGeneral

**Description**  

**Multiplicity**  
1

**Type**  
EcucBooleanParamDef

**Default value**  
false

**Post-Build Variant Value**  
false

**Value Configuration Class**

<table>
<thead>
<tr>
<th>Pre-compile time</th>
<th>Link time</th>
<th>Post-build time</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

**Value Configuration Class**

<table>
<thead>
<tr>
<th>Pre-compile time</th>
<th>Link time</th>
<th>Post-build time</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

**Scope / Dependency**  
scope: local

**No Included Containers**

### 10.2.4 CanTSynGlobalTimeDomain

**SWS Item**  
**ECUC_CanTSyn_00004 :**

**Container Name**  
CanTSynGlobalTimeDomain

**Description**  
This represents the existence of a global time domain on CAN. The CanTSyn module can administrate several global time domains at the same time that in itself form a hierarchy of domains and sub-domains. If the CanTSyn exists it is assumed that at least one global time domain exists.

**Configuration Parameters**

**SWS Item**  
**ECUC_CanTSyn_00005 :**

**Name**  
CanTSynGlobalTimeDomainId

**Parent Container**  
CanTSynGlobalTimeDomain

**Description**  
The global time domain ID.

**Multiplicity**  
1

**Type**  
EcucIntegerParamDef

**Range**  
0 .. 31

**Default value**  
--
### Specification of Time Synchronization over CAN

**AUTOSAR CP Release 4.3.1**

**SWS Item**

<table>
<thead>
<tr>
<th>ECUC_CanTSyn_00042</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td><strong>Parent Container</strong></td>
</tr>
</tbody>
</table>

**Description**

Switches support for 16 Byte Timesync messages on or off (for CAN FD only)

- true: use 16 byte Timesync message formats (for CAN FD only).
- false: use 8 byte Timesync message formats.

**Multiplicity**

1

**Type**

EcucBooleanParamDef

**Default value**

--

**Post-Build Variant Value**

true

---

**SWS Item**

<table>
<thead>
<tr>
<th>ECUC_CanTSyn_00022</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
</tr>
<tr>
<td><strong>Parent Container</strong></td>
</tr>
</tbody>
</table>

**Description**

Mandatory reference to the required synchronized time-base.

**Multiplicity**

1

**Type**

Symbolic name reference to [StbMSynchronizedTimeBase]

**Post-Build Variant Value**

false

---

**Included Containers**

<table>
<thead>
<tr>
<th>Container Name</th>
<th>Multiplicity</th>
<th>Scope / Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>CanTSynGlobalTimeFupDataIDLList</td>
<td>0..1</td>
<td>The DataIDLList for FUP messages ensures the identification of data elements due to CRC calculation process.</td>
</tr>
<tr>
<td>CanTSynGlobalTimeMaster</td>
<td>0..1</td>
<td>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.</td>
</tr>
<tr>
<td>CanTSynGlobalTimeOfnsDataIDLList</td>
<td>0..1</td>
<td>The DataIDLList for OFNS messages ensures the identification of data elements due to CRC calculation process.</td>
</tr>
<tr>
<td>CanTSynGlobalTimeOfsDataIDLList</td>
<td>0..1</td>
<td>The DataIDLList for OFS messages ensures the identification of data elements due to CRC calculation process.</td>
</tr>
<tr>
<td>CanTSynGlobalTimeSlave</td>
<td>0..1</td>
<td>Configuration of a global time slave. Each global time domain is required to have at least one time slave. The configured ECU may or may not represent a time slave.</td>
</tr>
<tr>
<td>CanTSynGlobalTimeSyncDataIDLList</td>
<td>0..1</td>
<td>The DataIDLList for SYNC messages ensures the identification of data elements due to CRC calculation process.</td>
</tr>
</tbody>
</table>
10.2.5 CanTSynGlobalTimeSyncDataIDList

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC CanTSyn 00024 :</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Container Name</strong></td>
<td>CanTSynGlobalTimeSyncDataIDList</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>The DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation process.</td>
</tr>
<tr>
<td><strong>Post-Build Variant Multiplicity</strong></td>
<td>true</td>
</tr>
<tr>
<td><strong>Multiplicity Configuration Class</strong></td>
<td>Pre-compile time</td>
</tr>
<tr>
<td></td>
<td>Link time</td>
</tr>
<tr>
<td></td>
<td>Post-build time</td>
</tr>
</tbody>
</table>

**Configuration Parameters**

<table>
<thead>
<tr>
<th>Included Containers</th>
<th>Multiplicity</th>
<th>Scope / Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>CanTSynGlobalTimeSyncDataIDListElemen t</td>
<td>16</td>
<td>Element of the DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation process.</td>
</tr>
</tbody>
</table>
10.2.6 CanTSTsGlobalTimeSyncDataIDListElement

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC_CanTSyn_00028 :</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container Name</td>
<td>CanTSTsGlobalTimeSyncDataIDListElement</td>
</tr>
<tr>
<td>Description</td>
<td>Element of the DataIDList for SYNC messages ensures the identification of data elements due to CRC calculation process.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWS Item</td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Parent Container</td>
</tr>
<tr>
<td>Description</td>
</tr>
</tbody>
</table>
data elements due to CRC calculation process.

<table>
<thead>
<tr>
<th>Multiplicity</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>EcucIntegerParamDef</td>
</tr>
<tr>
<td>Range</td>
<td>0 .. 15</td>
</tr>
<tr>
<td>Default value</td>
<td>-</td>
</tr>
<tr>
<td>Post-Build Variant Value</td>
<td>true</td>
</tr>
</tbody>
</table>

**Value Configuration Class**

- **Pre-compile time**: X All Variants
- **Link time**: --
- **Post-build time**: --

**Scope / Dependency**

- scope: local

---

**SWS Item** ECUC_CanTSyn_00030 :

**Name** CanTSynGlobalTimeSyncDataIDListValue

**Parent Container** CanTSynGlobalTimeSyncDataIDListElement

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

<table>
<thead>
<tr>
<th>Multiplicity</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>EcucIntegerParamDef</td>
</tr>
<tr>
<td>Range</td>
<td>0 .. 255</td>
</tr>
<tr>
<td>Default value</td>
<td>-</td>
</tr>
<tr>
<td>Post-Build Variant Value</td>
<td>true</td>
</tr>
</tbody>
</table>

**Value Configuration Class**

- **Pre-compile time**: X All Variants
- **Link time**: --
- **Post-build time**: --

**Scope / Dependency**

- scope: local

---

**SWS Item** ECUC_CanTSyn_00025 :

**Container Name** CanTSynGlobalTimeFupDataIDList

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

<table>
<thead>
<tr>
<th>Multiplicity</th>
<th>true</th>
</tr>
</thead>
</table>

**Configuration Parameters**

**Included Containers**

<table>
<thead>
<tr>
<th>Container Name</th>
<th>Multiplicity</th>
<th>Scope / Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>CanTSynGlobalTimeFupDataIDListElement</td>
<td>16</td>
<td>Element of the DataIDList for FUP messages ensures the identification of data elements due to CRC calculation process.</td>
</tr>
</tbody>
</table>
10.2.8 CanTSynGlobalTimeFupDataIDLListElement

**SWS Item**

- **ECUC_CanTSyn_00031**
  - **Container Name**: CanTSynGlobalTimeFupDataIDLListElement
  - **Description**: Element of the DataIDLList for FUP messages ensures the identification of data elements due to CRC calculation process.

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC_CanTSyn_00032</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td>CanTSynGlobalTimeFupDataIDLListIndex</td>
</tr>
<tr>
<td><strong>Parent Container</strong></td>
<td>CanTSynGlobalTimeFupDataIDLListElement</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Index of the DataIDLList for FUP messages ensures the identification of</td>
</tr>
</tbody>
</table>
### CanTSynGlobalTimeFupDataIDListValue

| Multiplicity | 1 |
| Type         | EcuIntegerParamDef |
| Range        | 0 .. 15 |
| Default value| -- |
| Post-Build Variant Value | true |
| Value Configuration Class | Pre-compile time: X All Variants |
| Scope / Dependency | scope: local |

Description:
Value of the DataIDList for FUP messages ensures the identification of data elements due to CRC calculation process.

### CanTSynGlobalTimeOfsDataIDList

| Multiplicity | 1 |
| Type         | EcuIntegerParamDef |
| Range        | 0 .. 255 |
| Default value| -- |
| Post-Build Variant Value | true |
| Value Configuration Class | Pre-compile time: X All Variants |
| Scope / Dependency | scope: local |

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

### CanTSynGlobalTimeOfsDataIDListElement

<table>
<thead>
<tr>
<th>Container Name</th>
<th>Multiplicity</th>
<th>Scope / Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>CanTSynGlobalTimeOfsDataIDListElement</td>
<td>16</td>
<td>Element of the DataIDList for OFS messages ensures the identification of data elements due to CRC calculation process.</td>
</tr>
</tbody>
</table>
10.2.10  CanTSynGlobalTimeOfsDataIDListElement

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC_CanTSyn_00034</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container Name</td>
<td>CanTSynGlobalTimeOfsDataIDListElement</td>
</tr>
<tr>
<td>Description</td>
<td>Element of the DataIDList for OFS messages ensures the identification of data elements due to CRC calculation process.</td>
</tr>
</tbody>
</table>

**Configuration Parameters**

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC_CanTSyn_00035</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>CanTSynGlobalTimeOfsDataIDListIndex</td>
</tr>
<tr>
<td>Parent Container</td>
<td>CanTSynGlobalTimeOfsDataIDListElement</td>
</tr>
</tbody>
</table>
### 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 | true |
| Value Configuration Class | Pre-compile time | X | All Variants |
| Link time | -- |
| Post-build time | -- |
| Scope / Dependency | scope: local |

#### SWS Item
ECUC_CanTSyn_00036 :

**Name** CanTSynGlobalTimeOfsDataIDListValue

**Parent Container** CanTSynGlobalTimeOfsDataIDListElement

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

#### No Included Containers

### 10.2.11 CanTSynGlobalTimeOfnsDataIDList

**SWS Item** ECUC_CanTSyn_00041 :

**Container Name** CanTSynGlobalTimeOfnsDataIDList

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

| Multiplicity Configuration Class | Pre-compile time | X | All Variants |
| Link time | -- |
| Post-build time | -- |

#### Configuration Parameters

<table>
<thead>
<tr>
<th>Included Containers</th>
</tr>
</thead>
<tbody>
<tr>
<td>CanTSynGlobalTimeOfnsDataIDListElement</td>
</tr>
</tbody>
</table>

Element of the DataIDList for OFNS messages ensures the identification of data elements due to CRC calculation process.
10.2.12 CanTSynGlobalTimeOfnsDataIDLlistElement

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC_CanTSyn_00037</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container Name</td>
<td>CanTSynGlobalTimeOfnsDataIDLlistElement</td>
</tr>
<tr>
<td>Description</td>
<td>Element of the DataIDLlist for OFNS messages ensures the identification of data elements due to CRC calculation process.</td>
</tr>
<tr>
<td>Configuration Parameters</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC_CanTSyn_00038</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>CanTSynGlobalTimeOfnsDataIDLlistIndex</td>
</tr>
</tbody>
</table>
### Specification of Time Synchronization over CAN

AUTOSAR CP Release 4.3.1

**Parent Container**: CanTSynGlobalTimeOfnsDataIDListElement

**Description**: Index of the DataIDList for OFNS messages ensures the identification of data elements due to CRC calculation process.

<table>
<thead>
<tr>
<th>Multiplicity</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td>EcucIntegerParamDef</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>0..15</td>
</tr>
<tr>
<td><strong>Default value</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>Post-Build Variant Value</strong></td>
<td>true</td>
</tr>
</tbody>
</table>

**Value Configuration Class**

<table>
<thead>
<tr>
<th>Pre-compile time</th>
<th>X</th>
<th>All Variants</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link time</td>
<td>--</td>
<td></td>
</tr>
<tr>
<td>Post-build time</td>
<td>--</td>
<td></td>
</tr>
</tbody>
</table>

**Scope / Dependency**: scope: local

---

**SWS Item**: ECUC_CanTSyn_00039

**Name**: CanTSynGlobalTimeOfnsDataIDListValue

**Parent Container**: CanTSynGlobalTimeOfnsDataIDListElement

**Description**: Value of the DataIDList for OFNS messages ensures the identification of data elements due to CRC calculation process.

<table>
<thead>
<tr>
<th>Multiplicity</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td>EcucIntegerParamDef</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>0..255</td>
</tr>
<tr>
<td><strong>Default value</strong></td>
<td>--</td>
</tr>
<tr>
<td><strong>Post-Build Variant Value</strong></td>
<td>true</td>
</tr>
</tbody>
</table>

**Value Configuration Class**

<table>
<thead>
<tr>
<th>Pre-compile time</th>
<th>X</th>
<th>All Variants</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link time</td>
<td>--</td>
<td></td>
</tr>
<tr>
<td>Post-build time</td>
<td>--</td>
<td></td>
</tr>
</tbody>
</table>

**Scope / Dependency**: scope: local

---

**No Included Containers**

---

### 10.2.13 CanTSynGlobalTimeMaster

**SWS Item**: ECUC_CanTSyn_00007

**Container Name**: CanTSynGlobalTimeMaster

**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.

<table>
<thead>
<tr>
<th>Multiplicity</th>
<th>true</th>
</tr>
</thead>
</table>

**Multiplicity Configuration Class**

<table>
<thead>
<tr>
<th>Pre-compile time</th>
<th>X</th>
<th>All Variants</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link time</td>
<td>--</td>
<td></td>
</tr>
<tr>
<td>Post-build time</td>
<td>--</td>
<td></td>
</tr>
</tbody>
</table>

**Configuration Parameters**

---

**SWS Item**: ECUC_CanTSyn_00044

**Name**: CanTSynCyclicMsgResumeTime

**Parent Container**: CanTSynGlobalTimeMaster

**Description**: Defines the time where the 1st regular cycle time based message transmission takes place, after an immediate transmission before. Unit: seconds

<table>
<thead>
<tr>
<th>Multiplicity</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td>EcucFloatParamDef</td>
</tr>
</tbody>
</table>
### Specification of Time Synchronization over CAN

#### AUTOSAR CP Release 4.3.1

**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

---

**SWS Item**: ECUC_CanTSyn_00045 :

**Name**
- CanTSynGlobalTimeDebounceTime

**Parent Container**
- CanTSynGlobalTimeMaster

**Description**
- This represents the configuration of a TX debounce time for SYNC, FUP, OFS and OFNS messages compared to a message before with the same PDU. 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

---

**SWS Item**: ECUC_CanTSyn_00015 :

**Name**
- CanTSynGlobalTimeTxCrcSecured

**Parent Container**
- CanTSynGlobalTimeMaster

**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

---

**SWS Item**: ECUC_CanTSyn_00017 :

**Name**
- CanTSynGlobalTimeTxPeriod

**Parent Container**
- CanTSynGlobalTimeMaster

**Description**
- This represents configuration of 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
### SWS Item ECUC_CanTSyn_00043:

<table>
<thead>
<tr>
<th>Name</th>
<th>CanTSynImmediateTimeSync</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent Container</td>
<td>CanTSynGlobalTimeMaster</td>
</tr>
<tr>
<td>Description</td>
<td>Enables/Disables the cyclic polling of StbM_GetTimeBaseUpdateCounter() within CanTSyn_MainFunction().</td>
</tr>
<tr>
<td>Multiplicity</td>
<td>1</td>
</tr>
<tr>
<td>Type</td>
<td>EcucBooleanParamDef</td>
</tr>
<tr>
<td>Default value</td>
<td>--</td>
</tr>
<tr>
<td>Post-Build Variant Value</td>
<td>true</td>
</tr>
</tbody>
</table>

**Value Configuration Class**

- **Pre-compile time**: X All Variants
- **Link time**: --
- **Post-build time**: --

**Scope / Dependency**

- scope: local

---

### SWS Item ECUC_CanTSyn_00020:

<table>
<thead>
<tr>
<th>Name</th>
<th>CanTSynMasterConfirmationTimeout</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent Container</td>
<td>CanTSynGlobalTimeMaster</td>
</tr>
<tr>
<td>Description</td>
<td>This represents the confirmation timeout after transmission of a SYNC message resp. OFS message. Unit: seconds.</td>
</tr>
<tr>
<td>Multiplicity</td>
<td>1</td>
</tr>
<tr>
<td>Type</td>
<td>EcucFloatParamDef</td>
</tr>
<tr>
<td>Range</td>
<td>[0 .. INF]</td>
</tr>
<tr>
<td>Default value</td>
<td>--</td>
</tr>
<tr>
<td>Post-Build Variant Value</td>
<td>true</td>
</tr>
</tbody>
</table>

**Value Configuration Class**

- **Pre-compile time**: X All Variants
- **Link time**: --
- **Post-build time**: --

**Scope / Dependency**

- scope: local

---

#### Included Containers

<table>
<thead>
<tr>
<th>Container Name</th>
<th>Multiplicity</th>
<th>Scope / Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>CanTSynGlobalTimeMasterPd u</td>
<td>1</td>
<td>This container encloses the configuration of the PDU that is supposed to contain the global time information.</td>
</tr>
</tbody>
</table>
10.2.14 CanTSynGlobalTimeMasterPdu

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC_CanTSyn_00009 :</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container Name</td>
<td>CanTSynGlobalTimeMasterPdu</td>
</tr>
</tbody>
</table>

**CanTSyn : EcucModuleDef**
- lowerMultiplicity = 0
- upperMultiplicity = 1

**CanTSynGlobalTimeDomain : EcucParamConfContainerDef**
- lowerMultiplicity = 1
- upperMultiplicity = *

**CanTSynGlobalTimeMaster : EcucParamConfContainerDef**
- lowerMultiplicity = 0
- upperMultiplicity = 1

**CanTSynMasterConfirmationTimeout : EcucFloatParamDef**
- min = 0.0
- lowerMultiplicity = 1
- upperMultiplicity = 1

**CanTSynGlobalTimeTxPeriod : EcucFloatParamDef**
- min = 0.0
- lowerMultiplicity = 1
- upperMultiplicity = 1

**CanTSynGlobalTimeTxRcvSecured : EcucEnumerationParamDef**
- lowerMultiplicity = 1
- upperMultiplicity = 1

**CanTSynImmediateTimeSync : EcucBooleanParamDef**

**CanTSynCyclicMsgResumeTime : EcucFloatParamDef**
- min = 0.0

**CanTSynGlobalTimeDebounceTime : EcucFloatParamDef**
- min = 0.0

**CanTSynGlobalTimeMasterPdu : EcucParamConfContainerDef**
- lowerMultiplicity = 1
- upperMultiplicity = 1

**CRC_SUPPORTED : EcucEnumerationLiteralDef**

**CRC_NOT_SUPPORTED : EcucEnumerationLiteralDef**

(from StbM)
**Description**
This container encloses the configuration of the PDU that is supposed to contain the global time information.

<table>
<thead>
<tr>
<th>Configuration Parameters</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC_CanTSyn_00008 :</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>CanTSynGlobalTimeMasterConfirmationHandleId</td>
</tr>
<tr>
<td>Parent Container</td>
<td>CanTSynGlobalTimeMasterPdu</td>
</tr>
<tr>
<td>Description</td>
<td>This represents the handle ID of the PDU that contains the global time information.</td>
</tr>
<tr>
<td>Multiplicity</td>
<td>1</td>
</tr>
<tr>
<td>Type</td>
<td>EcucIntegerParamDef (Symbolic Name generated for this parameter)</td>
</tr>
<tr>
<td>Range</td>
<td>0 .. 65535</td>
</tr>
<tr>
<td>Default value</td>
<td>--</td>
</tr>
<tr>
<td>Post-Build Variant Value</td>
<td>true</td>
</tr>
</tbody>
</table>

**Value Configuration Class**

| Pre-compile time | X | All Variants |
| Link time        | -- |
| Post-build time  | -- |

**Scope / Dependency**

scope: local

---

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC_CanTSyn_00027 :</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>CanTSynGlobalTimePduRef</td>
</tr>
<tr>
<td>Parent Container</td>
<td>CanTSynGlobalTimeMasterPdu</td>
</tr>
<tr>
<td>Description</td>
<td>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.</td>
</tr>
<tr>
<td>Multiplicity</td>
<td>1</td>
</tr>
<tr>
<td>Type</td>
<td>Reference to [Pdu ]</td>
</tr>
<tr>
<td>Post-Build Variant Value</td>
<td>true</td>
</tr>
</tbody>
</table>

**Value Configuration Class**

| Pre-compile time | X | All Variants |
| Link time        | -- |
| Post-build time  | -- |

**Scope / Dependency**

scope: local

---

**No Included Containers**
10.2.15 CanTSynGlobalTimeSlave

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC_CanTSyn_00012 :</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container Name</td>
<td>CanTSynGlobalTimeSlave</td>
</tr>
<tr>
<td>Description</td>
<td>Configuration of a global time slave. Each global time domain is required to have at least one time slave. The configured ECU may or may not represent a time slave.</td>
</tr>
<tr>
<td>Post-Build Variant</td>
<td>true</td>
</tr>
<tr>
<td>Multiplicity</td>
<td>true</td>
</tr>
<tr>
<td>Multiplicity Class</td>
<td>Pre-compile time X All Variants</td>
</tr>
<tr>
<td>Link time</td>
<td>--</td>
</tr>
<tr>
<td>Post-build time</td>
<td>--</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Configuration Parameters</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC_CanTSyn_00006 :</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>CanTSynGlobalTimeFollowUpTimeout</td>
</tr>
</tbody>
</table>
### Specification of Time Synchronization over CAN

**AUTOSAR CP Release 4.3.1**

**Parent Container**: CanTSynGlobalTimeSlave

**Description**: Rx timeout for the follow-up message. This is only relevant for selected bus systems Unit: seconds

**Multiplicity**: 1

**Type**: EcucFloatParamDef

**Range**: [0 .. INF]

**Default value**: --

**Value Configuration Class**: Pre-compile time: X, All Variants

**Post-Build Variant Value**: true

**Scope / Dependency**: scope: local

---

**SWS Item**: ECUC_CanTSyn_00011

**Name**: CanTSynGlobalTimeSequenceCounterJumpWidth

**Description**: The SequenceCounterJumpWidth specifies the maximum allowed gap of the Sequence Counter between twoSYNC resp. two OFS messages.

**Multiplicity**: 1

**Type**: EcucIntegerParamDef

**Range**: 1 .. 15

**Default value**: --

**Value Configuration Class**: Pre-compile time: X, All Variants

**Post-Build Variant Value**: true

**Scope / Dependency**: scope: local

---

**SWS Item**: ECUC_CanTSyn_00021

**Name**: CanTSynRxCrcValidated

**Description**: Definition of whether or not validation of the CRC is supported.

**Multiplicity**: 1

**Type**: EcucEnumerationParamDef

**Range**: CRC_IGNORED, CRC_NOT_VALIDATED, CRC_OPTIONAL, CRC_VALIDATED

<table>
<thead>
<tr>
<th>Value Configuration Class</th>
<th>Pre-compile time</th>
<th>Link time</th>
<th>Post-build time</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRC_IGNORED</td>
<td>X</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>CRC_NOT_VALIDATED</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>CRC_OPTIONAL</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>CRC_VALIDATED</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

**Description**:
- **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.
Synchronization messages are ignored.

<table>
<thead>
<tr>
<th>Post-Build Variant Value</th>
<th>Value</th>
<th>Pre-compile time</th>
<th>X</th>
<th>All Variants</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Class</td>
<td>Link time</td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Post-build time</td>
<td>--</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scope / Dependency</td>
<td>scope: local</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Included Containers**

<table>
<thead>
<tr>
<th>Container Name</th>
<th>Multiplicity</th>
<th>Scope / Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>CanTSynGlobalTimeSlavePd u</td>
<td>1</td>
<td>This container encloses the configuration of the PDU that is supposed to contain the global time information.</td>
</tr>
</tbody>
</table>
### 10.2.16  CanTSynGlobalTimeSlavePdu

<table>
<thead>
<tr>
<th>SWS Item</th>
<th>ECUC_CanTSyn_00014 :</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container Name</td>
<td>CanTSynGlobalTimeSlavePdu</td>
</tr>
<tr>
<td>Description</td>
<td>This container encloses the configuration of the PDU that is supposed to contain the global time information.</td>
</tr>
<tr>
<td><strong>Configuration Parameters</strong></td>
<td></td>
</tr>
<tr>
<td>SWS Item</td>
<td>ECUC_CanTSyn_00013 :</td>
</tr>
<tr>
<td>Container Name</td>
<td>CanTSynGlobalTimeSlavePdu</td>
</tr>
</tbody>
</table>
### Name
CanTSynGlobalTimeSlaveHandleId

### Parent Container
CanTSynGlobalTimeSlavePdu

### Description
This represents the handle ID of the PDU that contains the global time information.

### Multiplicity
1

### Type
EcuIntegerParamDef (Symbolic Name generated for this parameter)

### Range
0 .. 65535

### Default value
--

### Post-Build Variant Value
true

### Value Configuration Class

<table>
<thead>
<tr>
<th>Pre-compile time</th>
<th>Link time</th>
<th>Post-build time</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

### Scope / Dependency
scope: local

---

### SWS Item
ECUC_CanTSyn_00040 :

### Name
CanTSynGlobalTimePduRef

### Parent Container
CanTSynGlobalTimeSlavePdu

### 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

<table>
<thead>
<tr>
<th>Pre-compile time</th>
<th>Link time</th>
<th>Post-build time</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

### Scope / Dependency
scope: local

---

### No Included Containers

---

### 10.3 Published Information

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