| Document Title | Specification of Synchronized Time-Base Manager |
|---|---|
| **Document Owner** | AUTOSAR |
| **Document Responsibility** | AUTOSAR |
| **Document Identification No** | 421 |
| **Document Classification** | Standard |

| | |
|---|---|
| **Document Version** | 2.0.0 |
| **Document Status** | Final |
| **Part of Release** | 4.0 |
| **Revision** | 3 |

**Document Change History**

| Date | Version | Changed by | Change Description |
|---|---|---|---|
| 03.11.2011 | 2.0.0 | AUTOSAR Administration | • Added functionality for absolute time provision |
| 03.11.2010 | 1.1.0 | AUTOSAR Administration | • SRS_General: BSW004<br>• Binding character of the Standardized AUTOSAR Interfaces mentioned in the SWS Documents.<br>• Missing Port Driver DET Error Codes |
| 30.11.2009 | 1.0.0 | AUTOSAR Administration | Initial Release |

## Disclaimer



## Advice for users

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

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

# Table of Contents

# 1 Introduction and functional overview

This specification specifies the functionality, API and the configuration of the Synchronized Time-Base Manager (StbM).

The basic purpose of the Synchronized Time-Base Manager is to provide a “global time” to other BSW modules or to the application. Global time means, that different entities within the system have the same definition of time and passage of time. In the system, multiple “global times” can exist simultaneously (e.g. the FlexRay time definition, the TTCAN time definition …).
A second timebase called “AbsoluteTime” is provided for long term synchronizations realized by counting timer overflows of the global time.

In addition, the Synchronized Time-Base Manager shall provide means for state/error notification (e.g. loss and (re-) establishment of global time).

Main use cases for the application of the Synchronized Time-Base Manager are the following:

- Synchronized execution of several RunnableEntities within one system cluster
- Provision of an accurate definition of the absolute value of time
- Provision of an accurate definition of the passage of time

In section 1.1, a detailed definition of potential use cases are specified.

**Important note**: The proposed solution for the Synchronized Time-Base Manager mainly focuses on the “Synchronization of RunnableEntities”. However, further refinements of the specification shall satisfy as well the other use cases.

## 1.1 Use Cases

### 1.1.1 Synchronization of RunnableEntities

An arbitrary number of RunnableEntities must be executed synchronously. Synchronous means that they shall start with a well-defined and guaranteed relative offset (e.g. relative offset “0”, means the execution shall occur at the same point in time).

Such a requirement can be specified by the AUTOSAR Timing Extensions [10] and must be fulfilled independently of the actual deployment of the software components. However, as the limitations in section 4.1.1 indicate, only ECUs in the same network cluster are considered for the establishment of a synchronized time-base. Thus, the fulfillment of the requirement described above can only be guaranteed by deploying the related software components within the same cluster (e.g. FlexRay or TTCAN).

A classical application of this use cases is the sensor data read out or synchronous actuator triggering by different RunnableEntities.

### 1.1.2 Time provision

The application (and other BSW modules) shall have a central module that is responsible for the provision of information about the absolute time and passage of time.

A classical application of this use case is the access to synchronized calendar time by the application, e.g. for diagnostic events storage.

Other possible scenarios:

- Measuring the passage of time between two tagged system states (e.g. start and end of a RunnableEntity, deadline monitoring for timing-relevant functions).
- Guaranteeing the accurate triggering of OS alarms every (well-defined) interval

### 1.1.3 Notification mechanism

The application (and other BSW modules) shall have the possibility of getting informed about the current status of the definition of time within the cluster. Thus, some kind of notification mechanism is required, which informs the application (and other BSW modules) upon state changes or error occurrences.

### 1.1.4 Absolute time provision

In some cases an absolute time value (relative to the start of the ECU) is needed. E.g. in safety related systems it might be necessary to test the system every 48 hours.

## 1.2 Synchronized Time-Base Manager as broker

Basically, the Synchronized Time-Base Manager is interacting with two different roles, as Figure 1 shows:

1) **Provider**
   The StbM requires information from other modules (e.g. receiving the global time defined by the FlexRay Interface). So far, the *provider* has the task to deliver a “synchronized time-base”.

2) **Customer**
   The StbM collects information about time. This functionality can be used by several *customers*: either other BSW modules or application SW-Cs.

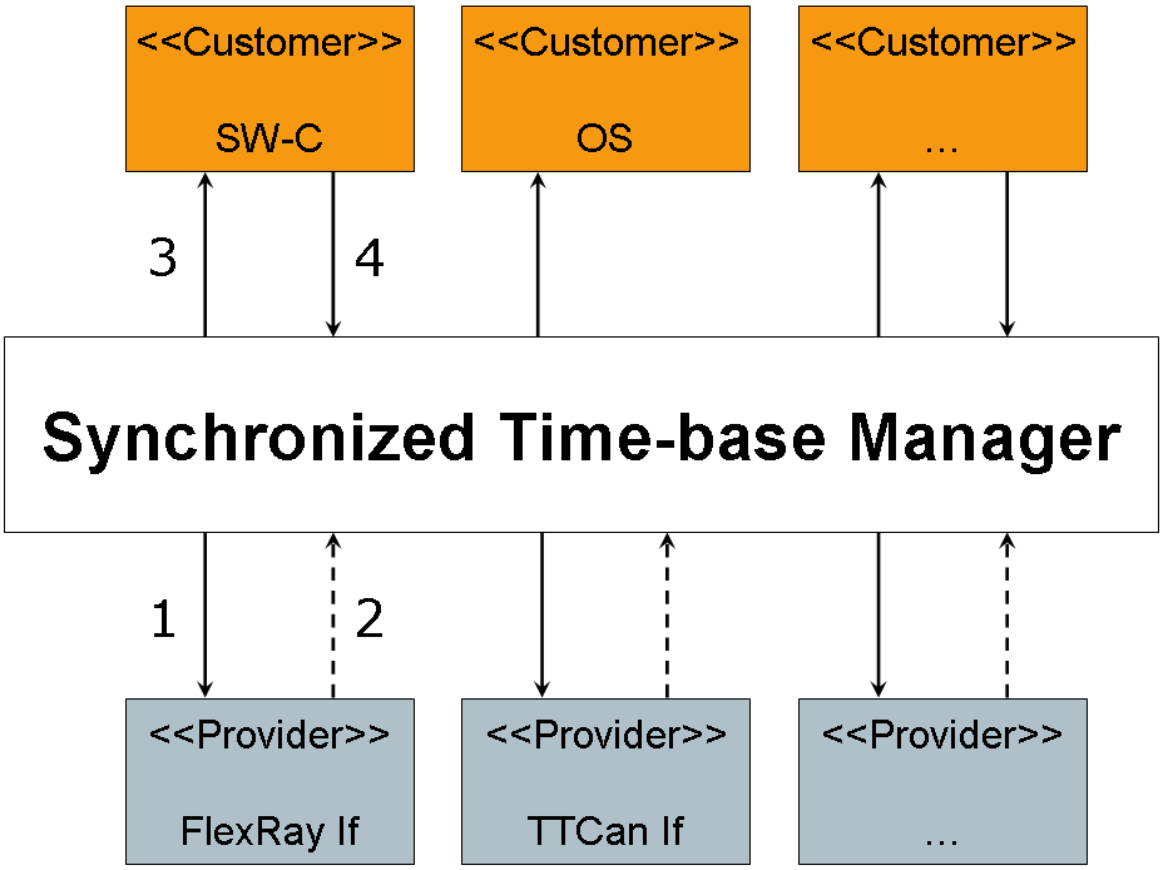



**Figure 1: Synchronized Time-Base Manager as consumer and provider**

So to say, the Synchronized Time-Base Manager acts as **time-base broker** by offering the customers access to synchronized time-bases. Doing this, the Synchronized Time-Base Manager abstracts from the “real” time-base provider.

In the following, the different providers and customers and the meaning of the several arrows in the figure above are described in a more detailed way.

## 1.3 Provider

The Synchronized Time-Base Manager itself does not provide any facility (e.g. protocols) for establishing a synchronized time across multiple nodes. Thus, the service requires other modules within the AUTOSAR BSW stack, which can provide this functionality. The FlexRay Interface and the TTCAN Interface are examples of modules which provide the definition of a synchronized time-base.

Calling the provider (arrow “1” in Figure 1) or getting called by the provider (arrow “2” in Figure 1) are the possible ways how to communicate with the provider.

## 1.4 Customer

The Synchronized Time-Base Manager has the requirement of satisfying the customers need in regard to time and passage of time. This section describes the different classes of customers and how they access the functionality.

**Note**: The classes are not completely disjoint. Thus, one specific customer can potentially be mapped to different classes of customers.

a) **Triggered customer**
This kind of customer is triggered by the Synchronized Time-Base Manager. Thus, the module itself is aware of the required functionality of the customer, and uses the defined interface of the customer to access it (e.g. accessing the Os SyncScheduleTable() API for synchronizing the respective OS ScheduleTable with the global time).

b) **Active customer**
This kind of customer autonomously calls the Synchronized Time-Base Manager, getting knowledge about the global time value (e.g. asking for the actual date and time).

c) **Notification customer**
This kind of customer is interested in the current state of the Synchronized Time-Base Manager and wants to get informed about state changes and/or error occurrences (e.g. loss of global time).

Customers a) and c) are triggered by the Synchronized Time-Base Manager (arrow “3” in Figure 1). Customer b) accesses itself the Synchronized Time-Base Manager (arrow “4” in Figure 1).

# 2 Acronyms and abbreviations

Acronyms and abbreviations, which have a local scope and therefore are not contained in the AUTOSAR glossary, must appear in a local glossary.

| ***Abbreviation / Acronym:*** | ***Description:*** |
|---|---|
| StbM | Synchronized Time-Base Manager |
| | |
| | |
| | |

# 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

[3] Specification of ECU Configuration
AUTOSAR_TPS_ECUConfiguration.pdf

[4] Specification of Operating System
AUTOSAR_SWS_OS.pdf

[5] Specification of FlexRay Interface
AUTOSAR_SWS_FlexRayInterface.pdf

[6] Specification of TTCAN Interface
AUTOSAR_SWS_TTCANInterface.pdf

[7] Virtual Functional Bus
AUTOSAR_EXP_VFB.pdf

[8] Software Component Template
AUTOSAR_TPS_SoftwareComponentTemplate.pdf

[9] Basic Software Module Description Template
AUTOSAR_TPS_BSWModuleDescriptionTemplate.pdf

[10] Specification of TimingExtensions
AUTOSAR_TPS_TimingExtensions.pdf

[11] Specification of Memory Mapping
AUTOSAR_SWS_MemoryMapping.pdf

[12] List of Basic Software Modules
AUTOSAR_TR_BSWModuleList.pdf

[13] General Requirements on Basic Software Modules
AUTOSAR_SRS_BSWGeneral.pdf

## 3.2 Company Reports, Academic Work, etc.

[14] Real-Time Systems and Software
Publisher: John Wiley & Sons Inc Publication
Date: 2001; Author: Alan C. Shaw

# 4 Constraints and assumptions

## 4.1 Limitations

The current module proposal has a number of limitations for the application of the Synchronized Time-Base Manager within an AUTOSAR system.

### 4.1.1 Cluster

For simplicity, only ECUs in the same network cluster are considered. A vehicle-wide definition of time can be achieved by introducing time gateways between clusters but this is left to the implementer.

### 4.1.2 Time agreement protocol

The current solution of the Synchronized Time-Base Manager does not provide its own time agreement protocol. Thus, the Synchronized Time-Base Manager shall use the functionality of time-base *provider* as defined in 1.3[1].

### 4.1.3 OS ScheduleTable

The Synchronized Time-Base Manager shall perform the functionality of synchronizing OS ScheduleTables with a respective synchronized time-base. However, the StbM considers only the case when the targeted OS ScheduleTable is **explicitly** synchronized. The **implicit** synchronization does not affect the StbM, because the synchronization mechanism bypasses the module (for more information about the difference between explicit and implicit synchronization, please refer to [4]). Thus, when talking in the following about synchronization of OS ScheduleTables, always the explicit one is meant.

### 4.1.4 Mode switches

The Synchronized Time-Base Manager does not deal with mode switches during runtime.

### 4.1.5 Out of scope

- Responsibility for those occurred errors during global time establishment, which are not caused by the module itself (e.g. loss of FlexRay global time is a FlexRay issue, not an issue of the Synchronized Time-Base Manager).
- Errors occurred during interaction with *customers*. Let's take the OS as example. Calling the explicit OS ScheduleTable synchronization may cause

[1] This limitation is only required for distributed global time needs. Obviously, when only a single ECU is considered (e.g. when the whole functionality is deployed to only one ECU), a global time is not explicitly required.

an exception, because the delta between the submitted parameter “counterValue” and the Os internal counter is higher than the tolerance range of affected expiry points. Dealing with this exception is an OS issue, not an issue of the Synchronized Time-Base Manager.

## 4.2 Terminology

So far, referencing a common definition of time has been declared throughout as “global time” in this document. However, with the help of this section a more precise definition of the term “global time” is given.

**Important note**: We avoid the use of the term “global time” for the rest of this specification, as there is no single “global time”. Instead, we use the term “synchronized time-base”.

For the purposes of this specification, we use the terms “physical time-base”, “software time-base” and “synchronized time-base” as follows:

### 4.2.1 Physical Time-Base

**Definition:** A physical time-base is a physical device which allows obtaining knowledge about the passage of time based on its physical properties (e.g. oscillation of a quartz crystal with a known frequency, atomic clock, earth rotation).

### 4.2.2 Software Time-Base

**Definition:** A software time-base is a software-provided measure of time derived from one or more physical time-bases. A time-base can be realized as

- an event source (e.g. a regular interrupt or alarm)
- a clock value data entity (e.g. a hardware or software counter representing time)

A software time-base is a means for *customers* to be triggered based on the passage of time and/or to obtain knowledge about the passage of time.

### 4.2.3 Synchronized Time-Base

**Definition:** A synchronized time-base is a software time-base existing at a processing entity (actor / processor / node of a distributed system) that is synchronized with software time-bases at different processing entities. A synchronized time-base can be achieved by time protocols or time agreement protocols that derive the synchronized time-base in a defined way from one or more physical time-bases. Examples are the network time protocol (NTP) and FlexRay time agreement protocol.

The synchronization will apply to the clock rate and optionally apply also to the clock absolute value.

A synchronized time-base allows synchronized action of the processing units. Synchronized time-bases are often called “global time”, like the so called “FlexRay global time”. We do not use the term “global time” here because it is an important feature of the module specified in this document that it can cope with different synchronized time-bases which may vary in terms of rate and absolute value.

For long term time determination a timer overflow counting mechanism provides an AbsoluteTime.

More than one synchronized time-base can exist at one processing unit, e.g. a FlexRay node will have the synchronized time-base retrieved from the FlexRay time agreement protocol in the network cluster but might also have a synchronized time-base derived from the time provided by a UTC time server (which is based on a set of atomic clocks). Both synchronized time-bases will probably have slightly different rate, and there is no relationship defined between their absolute values.

Note: A synchronized time-base is derived in a defined way (time protocol or time agreement protocol) from a defined set of physical time-bases.

## 4.3 Applicability to car domains

The concept is targeted at supporting time-critical and safety-related automotive applications such as airbag systems and braking systems. This doesn’t mean that the concept has all that is required by such systems though, but crucial timing-related features that cannot be deferred to implementation are considered.

In addition, the cluster limitation as described in 4.1.1 must be considered as restriction for the applicability.

## 4.4 Conflicts

None.

# 5 Dependencies to other modules

## 5.1 Code file structure

**[StbM001]** ⌈
The code file structure shall include the following files:

StbM.c general source code file of the Synchronized Time-Base Manager
StbM_Cfg.c contains pre-compile time configurable parameters⌋ (BSW00419, BSW00383)

## 5.2 Header file structure

**[StbM002]** ⌈
The module shall include the StbM.h file, providing the function prototypes to access the underlying functions of the module. ⌋( )

**[StbM003]** ⌈
The module shall include the file Rte_StbM.h as AUTOSAR Service Application Header, providing the types of the Synchronized Time-Base Manager and the function prototypes for the interaction with the RTE. ⌋( )

**[StbM132]** ⌈
The module header file StbM.h shall include the Rte_StbM_Type.h file, providing the types, which are commonly used by BSW Modules and Software Components.
Only types that are not already defined in Rte_StbM_Type.h shall remain in StbM.h. ⌋( )

**[StbM069]** ⌈
The module shall include the SchM_StbM.h file, providing the declaration of the API services the SchM offers to the Synchronized Time-Base Manager. ⌋(BSW00435)

**[StbM005]** ⌈
The module shall include the StbM_Cfg.h file, providing the configuration parameters for the Synchronized Time-Base Manager. ⌋(BSW00381)

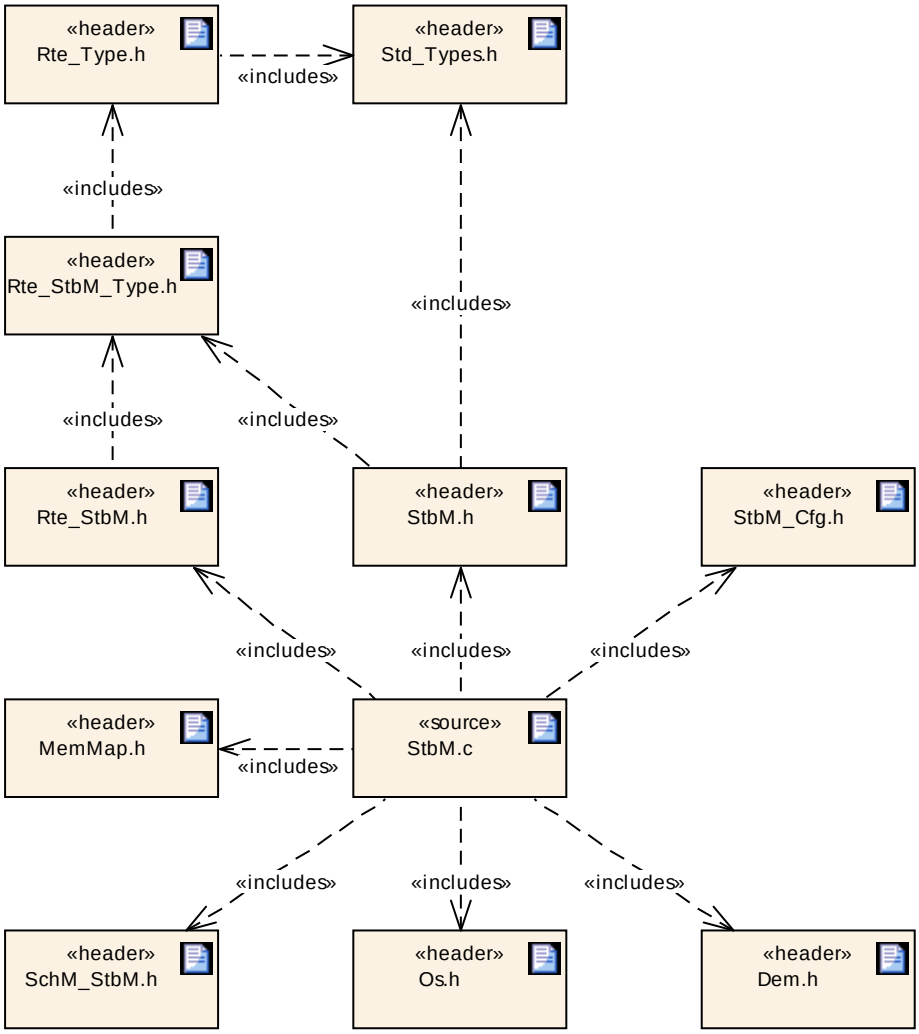
**[StbM065]** ⌈
In addition, the header files structure shall contain the following header files:

Os.h: general header file of the Operating System
Dem.h: general header file of the Diagnostic Event Manager
MemMap.h general header file including the compiler and linker specific keywords for memory allocation

StbM.c shall include Os.h, Dem.h and MemMap.h.

」(BSW00384, BSW00339, BSW00436)

**[StbM115]**「
Only StbM.h shall be included by the users of the Synchronized Time-Base Manager」( )


«header»
Rte_Type.h
«includes»
«header»
Std_Types.h
«includes»
«includes»
«header»
Rte_StbM_Type.h
«includes»
«includes»
«includes»
«header»
Rte_StbM.h
«header»
StbM.h
«header»
StbM_Cfg.h
«includes»
«includes»
«includes»
«header»
MemMap.h
«includes»
«source»
StbM.c
«includes»
«includes»
«includes»
«header»
SchM_StbM.h
«header»
Os.h
«header»
Dem.h

# 6 Requirements traceability

| Requirement | Satisfied by |
|---|---|
| - | StbM125 |
| - | StbM136 |
| - | StbM124 |
| - | StbM113 |
| - | StbM116 |
| - | StbM111 |
| - | StbM013 |
| - | StbM112 |
| - | StbM088 |
| - | StbM012 |
| - | StbM104 |
| - | StbM086 |
| - | StbM079 |
| - | StbM094 |
| - | StbM110 |
| - | StbM105 |
| - | StbM128 |
| - | StbM131 |
| - | StbM133 |
| - | StbM099 |
| - | StbM118 |
| - | StbM134 |
| - | StbM100 |
| - | StbM109 |
| - | StbM003 |
| - | StbM132 |
| - | StbM117 |
| - | StbM002 |
| - | StbM129 |
| - | StbM096 |
| - | StbM091 |
| - | StbM127 |
| - | StbM126 |
| - | StbM087 |
| - | StbM137 |
| - | StbM108 |
| - | StbM106 |
| - | StbM115 |

| | |
|---|---|
| - | StbM090 |
| - | StbM098 |
| - | StbM006 |
| - | StbM097 |
| - | StbM093 |
| - | StbM103 |
| - | StbM059 |
| - | StbM130 |
| - | StbM095 |
| - | StbM135 |
| - | StbM101 |
| - | StbM102 |
| - | StbM051 |
| - | StbM107 |
| - | StbM078 |
| - | StbM138 |
| - | StbM019 |
| - | StbM089 |
| - | StbM121 |
| - | StbM114 |
| BSW00304 | StbM140 |
| BSW00306 | StbM140 |
| BSW00307 | StbM140 |
| BSW00308 | StbM140 |
| BSW00309 | StbM140 |
| BSW00312 | StbM140 |
| BSW00314 | StbM140 |
| BSW00323 | StbM044 |
| BSW00325 | StbM140 |
| BSW00326 | StbM140 |
| BSW00327 | StbM041 |
| BSW00328 | StbM140 |
| BSW00330 | StbM140 |
| BSW00333 | StbM140 |
| BSW00334 | StbM140 |
| BSW00336 | StbM140 |
| BSW00337 | StbM039, StbM042, StbM040, StbM041 |
| BSW00338 | StbM046, StbM047, StbM044, StbM045, StbM043 |
| BSW00339 | StbM065, StbM058 |
| BSW00341 | StbM140 |
| BSW00342 | StbM140 |
| BSW00344 | StbM140 |

| | |
|---|---|
| BSW00345 | StbM061 |
| BSW00347 | StbM140 |
| BSW00350 | StbM043 |
| BSW00353 | StbM140 |
| BSW00355 | StbM140 |
| BSW00358 | StbM052 |
| BSW00359 | StbM140 |
| BSW00360 | StbM140 |
| BSW00361 | StbM140 |
| BSW00370 | StbM140 |
| BSW00371 | StbM140 |
| BSW00375 | StbM140 |
| BSW00376 | StbM057 |
| BSW00378 | StbM140 |
| BSW00380 | StbM140 |
| BSW00381 | StbM005 |
| BSW00383 | StbM001 |
| BSW00384 | StbM065 |
| BSW00385 | StbM041 |
| BSW00386 | StbM041 |
| BSW00387 | StbM140 |
| BSW00398 | StbM140 |
| BSW00399 | StbM140 |
| BSW004 | StbM068, StbM123 |
| BSW00400 | StbM140 |
| BSW00404 | StbM140 |
| BSW00405 | StbM140 |
| BSW00406 | StbM046 |
| BSW00407 | StbM066 |
| BSW00409 | StbM039 |
| BSW00412 | StbM140 |
| BSW00413 | StbM140 |
| BSW00414 | StbM052 |
| BSW00415 | StbM140 |
| BSW00416 | StbM140 |
| BSW00417 | StbM140 |
| BSW00419 | StbM001 |
| BSW00422 | StbM140 |
| BSW00424 | StbM067 |
| BSW00425 | StbM057 |
| BSW00426 | StbM140 |
| BSW00427 | StbM140 |

| | |
|---|---|
| BSW00428 | StbM140 |
| BSW00429 | StbM092, StbM020 |
| BSW00432 | StbM140 |
| BSW00433 | StbM140 |
| BSW00435 | StbM069, StbM070 |
| BSW00436 | StbM065, StbM071 |
| BSW00437 | StbM140 |
| BSW00438 | StbM140 |
| BSW00439 | StbM140 |
| BSW00440 | StbM140 |
| BSW00441 | StbM140 |
| BSW00442 | StbM072, StbM076, StbM075, StbM074, StbM073 |
| BSW00449 | StbM055, StbM053, StbM054 |
| BSW00450 | StbM120 |
| BSW00453 | StbM140 |
| BSW00455 | StbM140 |
| BSW005 | StbM140 |
| BSW006 | StbM140 |
| BSW007 | StbM140 |
| BSW009 | StbM140 |
| BSW010 | StbM140 |
| BSW101 | StbM052 |
| BSW159 | StbM061 |
| BSW160 | StbM140 |
| BSW161 | StbM140 |
| BSW162 | StbM140 |
| BSW164 | StbM140 |
| BSW168 | StbM140 |
| BSW170 | StbM140 |
| BSW420001 | StbM082, StbM020, StbM025, StbM028, StbM029, StbM026, StbM037, StbM038 |
| BSW420002 | StbM077, StbM083, StbM020, StbM022 |
| BSW420003 | StbM082, StbM025, StbM028, StbM029, StbM026 |
| BSW420005 | StbM080, StbM081, StbM015, StbM050 |
| BSW420006 | StbM050 |
| BSW420007 | StbM030, StbM031, StbM032, StbM033, StbM034, StbM035, StbM036 |
| BSW420008 | StbM037, StbM038 |
| BSW420009 | StbM085, StbM084 |

# 7 Functional specification

## 7.1 Background & Rationale

The overall objective is to provide synchronized time-bases in safety-related systems built on AUTOSAR technology.

In the AUTOSAR context, the need for a synchronized time-base comes from the AUTOSAR Time Determinism concept. If AUTOSAR compliant application software must execute with a predictable timing (as the time determinism concept requires), a suitable common time definition must be used. The Synchronized Time-Base Manager is responsible for the propagation of such a time definition.

## 7.2 Overview

The purpose of the Synchronized Time-Base Manager is to provide synchronized time-bases for AUTOSAR software. Based on the use cases discussed in section 1.1, the Synchronized Time-Base Manager shall provide the following functionality:

- **Access to synchronized time-base**:

  As it is not in the scope of the Synchronized Time-Base Manager to introduce new time agreement protocols, the Synchronized Time-Base Manager uses the existing facilities in the AUTOSAR layered software architecture (see [2] for more details). Currently, the Synchronized Time-Base Manager synchronized time-bases propagated by FlexRay and TTCAN clusters are supported.

- **Provision of synchronized time-bases:**

  As described in section 1.4, the Synchronized Time-Base Manager has well-defined *customers* (e.g. the OS), which require the functionality of the module. For the provision of the synchronized time-bases, two general approaches are eligible. On the one hand, the Synchronized Time-Base Manager triggers the customer (e.g. by synchronizing the OS ScheduleTable). On the other hand, the Synchronized Time-Base Manager must provide an appropriate API that allows *customers* to have direct access to the module (e.g. when asking for the current calendar time).

- **Notification mechanism**:

  Whenever state changes occur or the Synchronized Time-Base Manager detects incorrect behavior, registered notification *customers* (e.g. application, see 1.4) must be informed about those circumstances.

In Figure 1, the arrows 3 and 4 show the communication caused by the provision of time-bases and the notification mechanism. In addition, arrows 1 and 2 depict the access of the Synchronized Time-Base Manager to synchronized time-bases.

The rest of this chapter provides a deeper look into the previously mentioned functionality of the Synchronized Time-Base Manager.

## 7.3 Implementation requirements

**[StbM061]** ⌈
The Synchronized Time-Base Manager shall support pre-compile time configuration. ⌋ (BSW00345, BSW159)

**[StbM091]** ⌈
The interfaces of the Synchronized Time-Base Manager shall not be implemented as macros, unless all interfacing modules are defined to be pre-compile. ⌋ ( )

**[StbM068]** ⌈
The header file *StbM.h* shall contain a software and specification version number. ⌋ (BSW004)

**[StbM070]** ⌈
The StbM module source code file(s) shall include *SchM_StbM.h* if data consistency mechanisms of the BSW scheduler are required. ⌋ (BSW00435)

**[StbM071]** ⌈
The StbM module header file shall include *MemMap.h* and apply the memory mapping abstraction mechanisms as specified by [11]. ⌋ (BSW00436)

**[StbM072]** ⌈
Each variable that shall be accessible by AUTOSAR Debugging, shall be defined as global variable. ⌋ (BSW00442)

**[StbM073]** ⌈
All type definitions of variables which shall be debugged shall be accessible by the header file StbM.h. ⌋ (BSW00442)

**[StbM074]** ⌈
The declaration of variables in the header file shall be such, that it is possible to calculate the size of the variables by C-"sizeof".⌋ (BSW00442)

**[StbM075]** ⌈
Variables available for debugging shall be described in the respective Basic Software Module Description. ⌋ (BSW00442)

## 7.4 Clock time formats

**[StbM006]** ⌈
The Synchronized Time-Base Manager shall provide an abstract time format to its *customers*. ⌋ ( )

**[StbM078]** ⌈
The abstract time format provided by the Synchronized Time-Base Manager shall have the notion of ticks. ⌋ ( )

**[StbM079]** ⌈
In addition to the abstract time format, the Synchronized Time-Base Manager shall allow access to the respective tick duration, which defines the duration of one tick in microseconds. ⌋ ( )

The tick value in addition to the tick resolution defines the value of the respective time-base.

**[StbM135]** ⌈
The SynchronizedTimeBaseManager should provide an absolute time format to its customers. The absolute time format shall consist of a tick value and the number of overflows of the tick value counter. ⌋ ( )

Taking the overflows into account an absolute time can be determined.

## 7.5 Startup behavior

This chapter describes the actions, which shall be performed during StbM_Init().

### 7.5.1 Preconditions

Note:
The C initialization code (also known as start-up code) which initializes global and static variables with the initial values shall be executed before any call of a module function.

**[StbM012]** ⌈
Required basic software modules for the Synchronized Time-Base Manager must be available (running) before the Synchronized Time-Base Manager accesses them. ⌋ ( )

### 7.5.2 Initialization

**[StbM013]** ⌈
The Synchronized Time-Base Manager shall establish the initial state of the module (initial state: No synchronization time-base available), preparing the module for the actual functionality of providing synchronized time-base to the *customers*. ⌋ ( )

## 7.6 Shutdown behavior

None.

## 7.7 Normal operation

### 7.7.1 Access to synchronized time-bases

As stated in section 4.1.2, the Synchronized Time-Base Manager does not provide any time agreement protocol. Such a protocol has the functionality of establishing a synchronized time-base among arbitrary members (e.g. ECUs).

**[StbM015]** ⌈
The Synchronized Time-Base Manager shall provide an adapter mechanism, which allows making the access to the time-base (e.g. FlexRay time-base) configurable. ⌋ (BSW420005)

The Synchronized Time-Base Manager requires the following information of each time-base:

1. The *time-base value* is represented by the notion of ticks + tick duration.
2. The *sync state* indicates if the time-base is synchronized or not

Note:
The adapter will be triggered by the StbM via callouts (see section 8.5.3 for more details about the callout signature).

**[StbM080]** ⌈
The Synchronized Time-Base manager shall be independent of underlying *providers*. ⌋ (BSW420005)

For each *provider*, a respective adapter shall be available which translates the provider specific information (time-base value + time-base state) into the abstract representation required by the Synchronized Time-Base Manager.

**[StbM081]** ⌈
Time-base *providers* shall be called periodically by the Synchronized Time-Base Manager. ⌋ (BSW420005)

**[StbM050]** ⌈
In case the Synchronized Time-Base Manager cannot acquire the synchronized time-base from the *providers* (e.g. because a temporary FlexRay synchronization loss has been occurred), the module must be able to maintain the time-base autonomously. In this case, the interested *customers* must be informed about the

state change (see 7.9 for more information about the notification mechanism). ⌋ (BSW420005, BSW420006)

For the mentioned autonomous maintenance of the time-base, the Synchronized Time-Base Manager shall have access to the local time. In this case, the local time replaces the (currently not available) time-base from the original *provider*.

The access to the local time is realized via the definition of the configuration parameter STBM_LOCAL_TIME_REF (see section 10.2). In addition, each *triggered customer* can specify via the configuration parameter STBM_TRIGGER_IN_SYNCSTATE (see section 10.2) if he wants to be triggered when only the local time is available.

This is realized via the definition of the local time as synchronized time-base and setting the configuration parameter StbMMcuClockRef (see section 10.2).

### 7.7.2 Provision of synchronized time-bases

**[StbM019]** ⌈
The Synchronized Time-Base Manager shall use the concept of “Sync by absolute time value”.

Thus, the module itself does inform the *customers* about the absolute **definition of time** (e.g. 454 ticks). This is in contrast to “Sync by relative time value”, where the *customer* will be informed about the elapsed time **since** the last update. ⌋ ( )

**[StbM082]** ⌈
The Synchronized Time-Base Manager shall provide the following information to its *customers*:

1. The *time-base value* is represented by the notion of ticks + tick duration.
2. The *sync state* indicates if the time-base is synchronized or not⌋ (BSW420001, BSW420003)

**[StbM136]** ⌈
The Synchronized Time-Base Manager shall provide the following information to its *customers*:
The time-base absolute value is represented by the notion of ticks + tick duration + number of overflows⌋ ( )

**[StbM137]** ⌈
On every overflow of the tick value counter, the counter systemTicks shall be incremented by one. ⌋ ( )

**[StbM084]** ⌈
Customers of type *triggered customer* shall be invoked periodically by the Synchronized Time-Base Manager. ⌋ (BSW420009)

**[StbM093]** ⌈
The triggering period shall be configurable for each *triggered customer*. ⌋ ( )

**[StbM083]** ⌈
Customers of type *triggered customer* will be invoked by the Synchronized Time-Base Manager by providing the tick value of the respective time-base. ⌋ (BSW420002)

**[StbM085]** ⌈
Each customer of type *triggered customer* shall have the possibility to specify whether he wants to be invoked by the Synchronized Time-Base Manager in case the required time-base is not synchronized. ⌋ (BSW420009)

**[StbM020]** ⌈
The Synchronized Time-Base Manager must interact with the OS as *triggered customer*. The module calls the OS API for synchronizing OS ScheduleTables. ⌋ (BSW00429, BSW420001, BSW420002)

It is configurable at pre-compile time if an OS ScheduleTable shall be synchronized with a synchronization time-base.

**[StbM022]** ⌈
The Synchronized Time-Base Manager shall provide a means to configure the time-base to which the OS ScheduleTable should be synchronized. ⌋ (BSW420002)

Based on the configuration, the Synchronized Time-Base Manager synchronizes the OS counter value of the associated OS ScheduleTable.

The Synchronized Time-Base Manager is not responsible for starting and stopping the execution of OS ScheduleTables.

**[StbM077]** ⌈
The Synchronized Time-Base Manager shall synchronize OS ScheduleTables only when the associated synchronized time-base is synchronized. ⌋ (BSW420002)

**[StbM092]** ⌈
The Synchronized Time-Base Manager shall synchronize only OS ScheduleTables that are in one of the states WAITING, RUNNING or RUNNING_SYNCHRONOUS.

This implies that the Synchronized Time-Base Manager shall ask the OS for the status of the OS ScheduleTable before performing the synchronization. ⌋(BSW00429)

Note:
The Synchronized Time-Base Manager should ignore possible errors caused by the sequential execution of a) getting OS ScheduleTable status and b) performing the synchronization (e.g. someone else might have called a service to stop the OS ScheduleTable in the meantime).

**[StbM025]** ⌈
The Synchronized Time-Base Manager shall provide an interface for accessing the synchronized time-base. ⌋(BSW420001, BSW420003)

In this case, the information exchange is triggered by the *active customer*, but not by the Synchronized Time-Base Manager itself. The calling customer specifies the required synchronized time-base.

**[StbM026]** ⌈
The API for accessing the synchronized time-bases shall be used by application software components (see [7]) as well as other BSW modules (see [1]). ⌋(BSW420001, BSW420003)

**[StbM028]** ⌈
For the interaction with application software components, standardized AUTOSAR interfaces shall be specified. ⌋(BSW420001, BSW420003)

**[StbM029]** ⌈
For the interaction with other BSW modules, respective interfaces (see 8 for information about the interface definition) shall be available. ⌋(BSW420001, BSW420003)

### 7.7.3 Plausibility check

**[StbM030]** ⌈
The Synchronized Time-Base Manager shall provide an internal plausibility check mechanism for validating the synchronized time-bases. ⌋(BSW420007)

**[StbM031]** ⌈
The Synchronized Time-Base Manager shall monitor the cyclic execution of the StbM_Mainfunction (see section 8.4) for updating the status and values of the synchronized time-bases. ⌋(BSW420007)

## 7.8 Fault detection

**[StbM032]** ⌈
The Synchronized Time-Base Manager monitors its own triggering rate to guarantee a continuous time-base recalculation. ⌋(BSW420007)

**[StbM033]** ⌈
The Synchronized Time-Base Manager shall detect the loss of synchronized time-bases (e.g. because the FlexRay time agreement protocol can not retrieve a synchronized time-base in the network cluster). ⌋(BSW420007)

**[StbM034]** ⌈
The Synchronized Time-Base Manager shall detect the (re-) establishment of synchronized time-bases (e.g. because the FlexRay time agreement protocol has re-established the synchronized time-base in the network cluster). ⌋(BSW420007)

**[StbM035]** ⌈
The Synchronized Time-Base Manager shall detect errors during the invocation of synchronized time-base *providers* (e.g. FlexRay interface). ⌋(BSW420007)

**[StbM036]** ⌈
The Synchronized Time-Base Manager shall detect errors during the invocation of *customers* (e.g. OS). ⌋(BSW420007)

## 7.9 Notification mechanism

**[StbM037]** ⌈
The Synchronized Time-Base Manager shall provide notifications for *customers* that want to be informed about status changes detected by the Synchronized Time-Base Manager. The data type defined in section 8.2.1 classifies the different states. ⌋(BSW420001, BSW420008)

**[StbM038]** ⌈
The Synchronized Time-Base Manager shall support application software components (see [7]) as well as other BSW modules (see [1]) as potential *customers* for the notification mechanism. ⌋(BSW420001, BSW420008)

**[StbM076]** ⌈

The status detected by the Synchronized Time-Base Manager and defined in section 8.2.1 shall be available for debugging. ⌋(BSW00442)

## 7.10 Error classification

**[StbM039]** ⌈
Values for production code event IDs are assigned externally by the configuration of the DEM. They are published in the file Dem_IntErrId.h and included via Dem.h. ⌋(BSW00337, BSW00409)

**[StbM040]** ⌈
Development error values are of type uint8. ⌋(BSW00337)

**[StbM041]** ⌈
The following errors and exceptions shall be detectable by the Synchronized Time-Base Manager depending on its build version (development/production mode).

| *Type or error* | *Relevance* | *Related error code* | *Value [hex]* |
|---|---|---|---|
| API requests called with wrong parameter | Development | STBM_E_PARAM | 0x0A |
| Synchronized Time-Base Manager is not initialized | Development | STBM_E_NOT_INITIALIZED | 0x0B |
| API request integrity failed | Production | STBM_E_INTEGRITY_FAILED | Assigned by DEM |
| API request failed | Production | STBM_E_REQ_FAILED | Assigned by DEM |
| Invalid pointer in parameter list | Development | STBM_E_PARAM_POINTER | 0x10 |
| StbM initialization failed | Production | STBM_E_INIT_FAILED | 0x11 |

⌋(BSW00337, BSW00385, BSW00386, BSW00327)

**[StbM042]** ⌈
Additional errors that are detected because of specific implementation or specific hardware properties shall be added in the Synchronized Time-Base Manager implementation specification. The classification and enumeration shall be compatible to the errors listed above. ⌋(BSW00337)

Note:
There exist errors, which are of interest for the user of the Synchronized Time-Base Manager, but the source of failure is somewhere else (e.g. the FlexRay time-base is not synchronized). Thus, they do not appear in the above-mentioned error list and the Synchronized Time-Base Manger does not perform an error handling for those kinds of errors.

## 7.11 Error detection

**[StbM043]** ⌈
The detection of development errors is configurable (ON / OFF) at pre-compile time.The switch STBM_*DEV_ERROR_DETECT* (see chapter 10.2) shall activate or deactivate the detection of all development errors. ⌋(BSW00338, BSW00350)

**[StbM044]** ⌈
If the STBM_*DEV_ERROR_DETECT* switch is enabled, API parameter checking is enabled. The detailed description of the detected errors can be found in chapter 7.10 and chapter 8. ⌋(BSW00338, BSW00323)

**[StbM045]** ⌈
The detection of production code errors cannot be switched off. ⌋(BSW00338)

**[StbM101]** ⌈
If development error detection is enabled for this module (see 10.2), the following table specifies which DET error values shall be reported for each API call:

| ***API call*** | ***Error condition*** | ***DET related error value*** |
|---|---|---|
| StbM_Init | | -- |
| StbM_GetVersionInfo | Invalid pointer in parameter list | STBM_E_PARAM_POINTER |
| Stbm_GetSyncState | API requests called with wrong parameter | STBM_E_PARAM |
| | StbM is not yet initialized | STBM_E_NOT_INITIALIZED |
| | Invalid pointer in parameter list | STBM_E_PARAM_POINTER |
| StbM_GetGlobalTime | API requests called with wrong parameter | STBM_E_PARAM |
| | StbM is not yet initialized | STBM_E_NOT_INITIALIZED |
| | Invalid pointer in parameter list | STBM_E_PARAM_POINTER |
| StbM_GetTickDuration | API requests called with wrong parameter | STBM_E_PARAM |
| | StbM is not yet initialized | STBM_E_NOT_INITIALIZED |
| StbM_GetAbsoluteTime | API requests called with wrong parameter | STBM_E_PARAM |
| | StbM is not yet initialized | STBM_E_NOT_INITIALIZED |
| | Invalid pointer in parameter list | STBM_E_PARAM_POINTER |

⌋( )

## 7.12 Error notification

**[StbM046]** ⌈
Detected development errors shall be reported to the *Det_ReportError* service of the Development Error Tracer (DET) if the pre-processor switch *STBM_DEV_ERROR_DETECT* is set (see chapter 10.2). ⌋(BSW00406, BSW00338)

**[StbM047]** ⌈
Production errors shall be reported to Diagnostic Event Manager. ⌋(BSW00338)

## 7.13 Version checking

**[StbM123]** ⌈
The Synchronized Time-Base Manager shall perform Inter Module Checks to avoid integration of incompatible files.

The imported included files shall be checked by preprocessing directives. ⌋(BSW004)

The following version numbers shall be verified:

- <MODULENAME>_AR_RELEASE_MAJOR_VERSION
- <MODULENAME>_AR_RELEASE_MINOR_VERSION

Where <MODULENAME> is the module abbreviation of the other (external) modules which provide header files included by the Synchronized Time-Base Manager.

If the values are not identical to the values expected, an error shall be reported.

# 8 API specification

## 8.1 Imported types

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

**[StbM051]** ⌈

| *Module* | *Imported Type* |
|---|---|
| Dem | Dem_EventIdType |
| | Dem_EventStatusType |
| Os | AccessType |
| | ApplicationStateRefType |
| | ApplicationType |
| | CounterType |
| | ISRType |
| | MemorySizeType |
| | MemoryStartAddressType |
| | ObjectAccessType |
| | ObjectTypeType |
| | ScheduleTableStatusRefType |
| | ScheduleTableType |
| | StatusType |
| | TaskType |
| | TickRefType |
| | TickType |
| | TrustedFunctionIndexType |
| | TrustedFunctionParameterRefType |
| Std_Types | Std_ReturnType |
| | Std_VersionInfoType |

⌋ ( )

## 8.2 Type definitions

### 8.2.1 StbM_SyncStatusType

| | | |
|---|---|---|
| ***Name:*** | `StbM_SyncStatusType` | |
| ***Type:*** | `Enumeration` | |
| ***Range:*** | `STBM_STATE_NOT_SYNC` | The time-base is not synchronized |
| | `STBM_STATE_SYNC` | The time-base is synchronized |
| ***Description:*** | Variables of this type are used to represent the status of the synchronized time-base. | |

### 8.2.2 StbM_SynchronizedTimeBaseType

| | | | |
|---|---|---|---|
| ***Name:*** | `StbM_SynchronizedTimeBaseType` | | |
| ***Type:*** | `uint16` | | |
| ***Range:*** | `0..2^16-1` | `--` | -- |
| ***Description:*** | Variables of this type are used to represent the kind of synchronized time-base. | | |

### 8.2.3 StbM_TickType

| | | | |
|---|---|---|---|
| ***Name:*** | `StbM_TickType` | | |
| ***Type:*** | `uint16` | | |
| ***Range:*** | `0..2^16-1` | `--` | -- |
| ***Description:*** | Variables of this type are used to represent the notion of ticks. The range of the type shall be configurable via the respective ECUC parameter "StbM_TickTypeRange". | | |

### 8.2.4 StbM_SystemTimeType

| | | | |
|---|---|---|---|
| ***Name:*** | `StbM_SystemTimeType` | | |
| ***Type:*** | `Structure` | | |
| ***Element:*** | `StbM_TickType` | `ticks` | number of ticks |
| | `uint16` | `tickDuration` | duration of one tick in microseconds |
| | `uint32` | `systemTicks` | number of overflows |
| ***Description:*** | Variables of this type are used for expressing long time intervals, e.g. time stamps, where the usage of StbM_TickTime would cause overflows. | | |

## 8.3 Function definitions

This is a list of functions provided for upper layer modules.

### 8.3.1 StbM_GetVersionInfo

**[StbM066]** ⌈

| | | |
|---|---|---|
| ***Service name:*** | StbM_GetVersionInfo | |
| ***Syntax:*** | `void StbM_GetVersionInfo( Std_VersionInfoType* versioninfo )` | |
| ***Service ID[hex]:*** | 0x05 | |
| ***Sync/Async:*** | Synchronous | |
| ***Reentrancy:*** | Reentrant | |
| ***Parameters (in):*** | None | |
| ***Parameters (inout):*** | None | |
| ***Parameters (out):*** | versioninfo | Pointer to the memory location holding the version information of the module. |
| ***Return value:*** | None | |
| ***Description:*** | Returns the version information of this module. | |

⌋ (BSW00407)

**[StbM094]** ⌈
If development error detection for the StbM module is enabled the function StbM_GetVersionInfo shall raise the development error STBM_E_PARAM_POINTER and return if versioninfo is a NULL pointer (NULL_PTR). ⌋ ( )

**[StbM095]** ⌈
The function StbM_GetVersionInfo shall return the version information of this module. The version information includes:

- Module Id
- Vendor Id
- Vendor specific version numbers. ⌋ ( )

**[StbM096]** ⌈
The function StbM_GetVersionInfo shall be pre compile time configurable On/Off by the configuration parameter StbMVersionInfoApi. ⌋ ( )

### 8.3.2 StbM_Init

**[StbM052]** ⌈

| | |
|---|---|
| ***Service name:*** | StbM_Init |
| ***Syntax:*** | `void StbM_Init(` |

| | |
|---|---|
| | void<br>) |
| ***Service ID[hex]:*** | 0x00 |
| ***Sync/Async:*** | Synchronous |
| ***Reentrancy:*** | Non Reentrant |
| ***Parameters (in):*** | None |
| ***Parameters (inout):*** | None |
| ***Parameters (out):*** | None |
| ***Return value:*** | None |
| ***Description:*** | Initializes the Synchronized Time-base Manager |

⌋(BSW101, BSW00358, BSW00414)

**[StbM097]** ⌈
The ECU State Manager shall call the function StbM_Init during the startup phase of the ECU in order to initialize the module. ⌋( )

The StbM is not functional until this function has been called.

**[StbM098]** ⌈
The StbM module's environment shall make sure that the DEM has been initialized before the StbM is initialized. ⌋( )

**[StbM099]** ⌈
The StbM module shall report the error STBM_E_INIT_FAILED if the initialization of the StbM module failed. ⌋( )

**[StbM100]** ⌈
A static status variable denoting if the StbM is initialized shall be initialized with value 0 before any APIs of the StbM are called. ⌋( )

**[StbM121]** ⌈
StbM_Init shall set the static status variable to a value not equal to 0. ⌋( )

### 8.3.3 StbM_GetSyncState

**[StbM053]** ⌈

| | | |
|---|---|---|
| ***Service name:*** | StbM_GetSyncState | |
| ***Syntax:*** | Std_ReturnType StbM_GetSyncState(<br>StbM_SynchronizedTimeBaseType timeBaseID,<br>StbM_SyncStatusType* syncState<br>) | |
| ***Service ID[hex]:*** | 0x01 | |
| ***Sync/Async:*** | Synchronous | |
| ***Reentrancy:*** | Non Reentrant | |
| ***Parameters (in):*** | timeBaseID | The synchronized time-base, whose state is of interest. |
| ***Parameters (inout):*** | None | |
| ***Parameters (out):*** | syncState | The synchronization state of the time-base. |

| | | |
|---|---|---|
| ***Return value:*** | Std_ReturnType | -- |
| ***Description:*** | Returns the current synchronization state of the submitted time-base. | |

⌋ (BSW00449)

**[StbM102]** ⌈
The function StbM_GetSyncState shall return the current synchronization state of the submitted time-base. ⌋ ( )

Regarding error detection, the requirement StbM101 is applicable to the function StbM_GetSyncState.

**[StbM116]** ⌈
If an error occurs during the execution of the function StbM_GetSyncState, all out parameters shall stay untouched. ⌋ ( )

### 8.3.4 StbM_GetGlobalTime

**[StbM054]** ⌈

| | | |
|---|---|---|
| ***Service name:*** | StbM_GetGlobalTime | |
| ***Syntax:*** | `Std_ReturnType StbM_GetGlobalTime(`<br>`  StbM_SynchronizedTimeBaseType timeBaseID,`<br>`  StbM_TickType* ticks`<br>`)` | |
| ***Service ID[hex]:*** | 0x02 | |
| ***Sync/Async:*** | Synchronous | |
| ***Reentrancy:*** | Non Reentrant | |
| ***Parameters (in):*** | timeBaseID | The synchronized time-base, whose time definition is of interest. |
| ***Parameters (inout):*** | None | |
| ***Parameters (out):*** | ticks | The current definition of time represented by the notion of ticks. |
| ***Return value:*** | Std_ReturnType | -- |
| ***Description:*** | Returns the current time of the submitted synchronized time-base. The time is represented by the notion of ticks. | |

⌋ (BSW00449)

**[StbM103]** ⌈
The function StbM_GetGlobalTime shall return the current time of the submitted time-base. ⌋ ( )

Regarding error detection, the requirement StbM101 is applicable to the function StbM_GetGlobalTime.

**[StbM117]** ⌈
If an error occurs during the execution of the function StbM_GetGlobalTime, all out parameters shall stay untouched. ⌋ ( )

### 8.3.5 StbM_GetTickDuration

**[StbM055]** ⌈

| | | |
|---|---|---|
| ***Service name:*** | StbM_GetTickDuration | |
| ***Syntax:*** | `Std_ReturnType StbM_GetTickDuration(`<br>`  StbM_SynchronizedTimeBaseType timeBaseID,`<br>`  uint16* tickDuration`<br>`)` | |
| ***Service ID[hex]:*** | 0x03 | |
| ***Sync/Async:*** | Synchronous | |
| ***Reentrancy:*** | Non Reentrant | |
| ***Parameters (in):*** | timeBaseID | The synchronized time-base, whose tick duration is of interest. |
| ***Parameters (inout):*** | None | |
| ***Parameters (out):*** | tickDuration | Number of microseconds of one tick. |
| ***Return value:*** | Std_ReturnType | -- |
| ***Description:*** | Returns the duration of one time-base tick in microseconds. | |

⌋ (BSW00449)

**[StbM104]** ⌈
The function StbM_GetTickDuration shall return the current time of the submitted time-base. ⌋ ( )

Regarding error detection, the requirement StbM101 is applicable to the function StbM_GetTickDuration.

**[StbM118]** ⌈
If an error occurs during the execution of the function StbM_GetTickDuration, all out parameters shall stay untouched. ⌋ ( )

### 8.3.6 StbM_GetAbsoluteTime

| | | |
|---|---|---|
| ***Service name:*** | StbM_GetAbsoluteTime | |
| ***Syntax:*** | `void StbM_GetAbsoluteTime(`<br>`  StbM_SynchronizedTimeBaseType timeBaseID,`<br>`  StbM_SystemTimeType* systemTime`<br>`)` | |
| ***Service ID[hex]:*** | 0x06 | |
| ***Sync/Async:*** | Synchronous | |
| ***Reentrancy:*** | Non Reentrant | |
| ***Parameters (in):*** | timeBaseID | The synchronized time-base, whose state is of interest. |
| ***Parameters (inout):*** | None | |
| ***Parameters (out):*** | systemTime | absolute time stamp |
| ***Return value:*** | None | |
| ***Description:*** | Returns an absolute time value relative to the start of the ECU. | |

**[StbM133]** ⌈
The function StbM_GetAbsoluteTime shall return the current time of the submitted time-base plus the number of timer overflows. ⌋ ( )

Regarding error detection, the requirement StbM101 is applicable to the function StbM_GetAbsoluteTime.

**[StbM134]** ⌈
If an error occurs during the execution of the function StbM_GetAbsoluteTime, all out parameters shall stay untouched. ⌋ ( )

## 8.4 Scheduled functions

### 8.4.1 StbM_MainFunction

**[StbM067]** ⌈
BSW module main processing functions are only allowed to be allocated to basic tasks by the function StbM_MainFunction. ⌋(BSW00424)

**[StbM057]** ⌈

| | |
|---|---|
| ***Service name:*** | StbM_MainFunction |
| ***Syntax:*** | `void StbM_MainFunction(`<br>`  void`<br>`)` |
| ***Service ID[hex]:*** | 0x04 |
| ***Timing:*** | FIXED_CYCLIC |
| ***Description:*** | This function will be called cyclically by a task body provided by the BSW Scheduler.<br><br>The StbM_MainFunction invokes each time-base provider, gathering the required information (sync status, tick value, tick duration).<br>In addition, each triggered customer will be invoked, providing the status and tick value of the associated time-base. |

⌋(BSW00425, BSW00376)

**[StbM105]** ⌈
The function StbM_MainFunction shall perform the time-base *provider* callout. ⌋( )

**[StbM106]** ⌈
In case of state changes of associated time-bases, the function StbM_MainFunction shall perform the triggered customer sync state callback. ⌋( )

**[StbM107]** ⌈
The function StbM_MainFunction shall perform the triggered customer time provision callback. ⌋( )

**[StbM120]** ⌈
In case the module has not yet been initialized, the function StbM_MainFunction shall return immediately without performing any functionality and without raising any errors. ⌋(BSW00450)

## 8.5 Expected Interfaces

In this chapter all interfaces required from other modules are listed.

### 8.5.1 Mandatory Interfaces

This chapter defines all interfaces which are required to fulfill the core functionality of the Synchronized Time-Base Manager.

**[StbM058]** ⌈

| *API function* | *Description* |
|---|---|
| Dem_ReportErrorStatus | Queues the reported events from the BSW modules (API is only used by BSW modules). The interface has an asynchronous behavior, because the processing of the event is done within the Dem main function. |
| Det_ReportError | Service to report development errors. |

⌋ (BSW00339)

### 8.5.2 Optional Interfaces

This chapter defines all interfaces which are required to fulfill an optional functionality of the Synchronized Time-Base Manager.

**[StbM059]** ⌈

| *API function* | *Description* |
|---|---|
| AllowAccess | This service sets the own state of an OS-Application from APPLICATION_RESTARTING to APPLICATION_ACCESSIBLE. |
| CallTrustedFunction | A (trusted or non-trusted) OS-Application uses this service to call a trusted function |
| CheckISRMemoryAccess | This service checks if a memory region is write/read/execute accessible and also returns information if the memory region is part of the stack space. |
| CheckObjectAccess | This service determines if the OS-Applications, given by ApplID, is allowed to<br>use the IDs of a Task, ISR, Resource,<br>Counter, Alarm or Schedule Table in API calls. |
| CheckObjectOwnership | This service determines to which OS-Application a given Task, ISR, Counter, Alarm or Schedule Table belongs |
| CheckTaskMemoryAccess | This service checks if a memory region is write/read/execute accessible and also returns information if the memory region is part of the stack space. |
| GetApplicationID | This service determines the currently running OS-Application (a unique identifier has to be allocated to each application). |
| GetApplicationState | This service returns the current state of an OS-Application. |
| GetCounterValue | This service reads the current count value of a counter (returning either the hardware timer ticks if counter is driven by hardware or the software ticks when user drives counter). |
| GetElapsedValue | This service gets the number of ticks between the current tick value and a previously read tick value. |
| GetISRID | This service returns the identifier of the currently executing ISR. |
| GetScheduleTableStatus | This service queries the state of a schedule table (also with respect to synchronization). |
| IncrementCounter | This service increments a software counter. |
| NextScheduleTable | This service switches the processing from one schedule table to another schedule table. |
| SetScheduletableAsync | This service stops synchronization of a schedule table. |
| StartScheduleTableAbs | This service starts the processing of a schedule table at an absolute value "Start" on the underlying counter. |

| | |
|---|---|
| StartScheduleTableRel | This service starts the processing of a schedule table at "Offset" relative to the "Now" value on the underlying counter. |
| StartScheduleTableSynchron | This service starts an explicitly synchronized schedule table synchronously. |
| StopScheduleTable | This service cancels the processing of a schedule table immediately at any point while the schedule table is running. |
| SyncScheduleTable | This service provides the schedule table with a synchronization count and start synchronization. |

⌋ ( )

### 8.5.3 Configurable Interfaces

In this chapter all interfaces are listed where the target function could be configured. The target function is usually a callback function. The names of these kinds of interfaces are not fixed because they are configurable.

**[StbM108]** ⌈
For each synchronized time-base, respective call-out functions shall be configurable which provide the synchronization state StbM086, the tick value StbM087 and the tick duration StbM088 of the time-base. ⌋ ( )

**[StbM109]** ⌈
For each *triggered customer*, respective callback functions shall be configurable for the provision of the synchronization state StbM089 and the tick value StbM090. The Synchronized Time-Base Manager invokes the callback functions within the StbM_Mainfunction (see section 8.4). ⌋ ( )

#### 8.5.3.1 SyncStateProviderCallout

**[StbM086]** ⌈

| | | |
|---|---|---|
| ***Service name:*** | SyncStateProviderCallout | |
| ***Syntax:*** | `void SyncStateProviderCallout(`<br>`  StbM_SynchronizedTimeBaseType timeBaseID,`<br>`  StbM_SyncStatusType* syncState`<br>`)` | |
| ***Sync/Async:*** | Synchronous | |
| ***Reentrancy:*** | Non Reentrant | |
| ***Parameters (in):*** | timeBaseID | The synchronized time-base, whose state is of interest. |
| ***Parameters (inout):*** | None | |
| ***Parameters (out):*** | syncState | The synchronization state of the time-base. |
| ***Return value:*** | None | |
| ***Description:*** | The provider returns the current synchronization state of the time-base via callout. | |

⌋ ( )

**[StbM110]** ⌈
The call-out function SyncStateProviderCallout shall be called by the Synchronized Time-Base Manager for gathering the current synchronization state of the time-base.
⌋ ( )

### 8.5.3.2 GlobalTimeProviderCallout

**[StbM087]** ⌈

| | | |
|---|---|---|
| ***Service name:*** | GlobalTimeProviderCallout | |
| ***Syntax:*** | `void GlobalTimeProviderCallout(`<br>`  StbM_SynchronizedTimeBaseType timeBaseID,`<br>`  StbM_TickType* ticks`<br>`)` | |
| ***Sync/Async:*** | Synchronous | |
| ***Reentrancy:*** | Non Reentrant | |
| ***Parameters (in):*** | timeBaseID | The synchronized time-base, whose time definition is of interest. |
| ***Parameters (inout):*** | None | |
| ***Parameters (out):*** | ticks | The current definition of time represented by the notion of ticks. |
| ***Return value:*** | None | |
| ***Description:*** | The provider returns the current time of the synchronized time-base via callout. The time is represented by the notion of ticks. | |

⌋ ( )

**[StbM111]** ⌈
The call-out function GlobalTimeProviderCallout shall be called by the Synchronized Time-Base Manager for gathering the current tick value of the time-base. ⌋ ( )

### 8.5.3.3 TickDurationProviderCallout

**[StbM088]** ⌈

| | | |
|---|---|---|
| ***Service name:*** | TickDurationProviderCallout | |
| ***Syntax:*** | `void TickDurationProviderCallout(`<br>`  StbM_SynchronizedTimeBaseType timeBaseID,`<br>`  uint16* tickDuration`<br>`)` | |
| ***Sync/Async:*** | Synchronous | |
| ***Reentrancy:*** | Non Reentrant | |
| ***Parameters (in):*** | timeBaseID | The synchronized time-base, whose state is of interest. |
| ***Parameters (inout):*** | None | |
| ***Parameters (out):*** | tickDuration | The duration of one time-base tick in microseconds. |
| ***Return value:*** | None | |
| ***Description:*** | The provider returns the duration of one time-base tick in microseconds via callout. | |

⌋ ( )

**[StbM112]** ⌈
The call-out function TickDurationProviderCallout shall be called by the Synchronized Time-Base Manager for gathering the duration of one time-base tick. ⌋ ( )

#### 8.5.3.4 SyncStateCustomerCallbackFunction

**[StbM089]** ⌈

| | | |
|---|---|---|
| ***Service name:*** | SyncStateCustomerCallbackFunction | |
| ***Syntax:*** | `void SyncStateCustomerCallbackFunction( StbM_SynchronizedTimeBaseType timeBaseID, StbM_SyncStatusType syncState )` | |
| ***Sync/Async:*** | Synchronous | |
| ***Reentrancy:*** | Non Reentrant | |
| ***Parameters (in):*** | timeBaseID | The synchronized time-base, whose time definition is of interest. |
| | syncState | The synchronization state of the time-base. |
| ***Parameters (inout):*** | None | |
| ***Parameters (out):*** | None | |
| ***Return value:*** | None | |
| ***Description:*** | The triggered customer will be triggered with the current synchronization state of the time-base via callback.<br>The callback will be executed whenever a state change occurs. | |

⌋ ( )

**[StbM113]** ⌈
The callback function SyncStateCustomerCallbackFunction shall be invoked by the Synchronized Time-Base Manager for providing the current synchronization state of the time-base to the *triggered customer*. ⌋ ( )

#### 8.5.3.5 GlobalTimeCustomerCallbackFunction

**[StbM090]** ⌈

| | | |
|---|---|---|
| ***Service name:*** | GlobalTimeCustomerCallbackFunction | |
| ***Syntax:*** | `void GlobalTimeCustomerCallbackFunction( StbM_SynchronizedTimeBaseType timeBaseID, StbM_TickType ticks )` | |
| ***Sync/Async:*** | Synchronous | |
| ***Reentrancy:*** | Non Reentrant | |
| ***Parameters (in):*** | timeBaseID | The synchronized time-base, whose time definition is of interest. |
| | ticks | The current definition of time represented by the notion of ticks. |
| ***Parameters (inout):*** | None | |
| ***Parameters (out):*** | None | |
| ***Return value:*** | None | |
| ***Description:*** | The triggered customer will be triggered with the current time-base value via callback.<br>The callback will be executed periodically by the StbM. | |

⌋ ( )

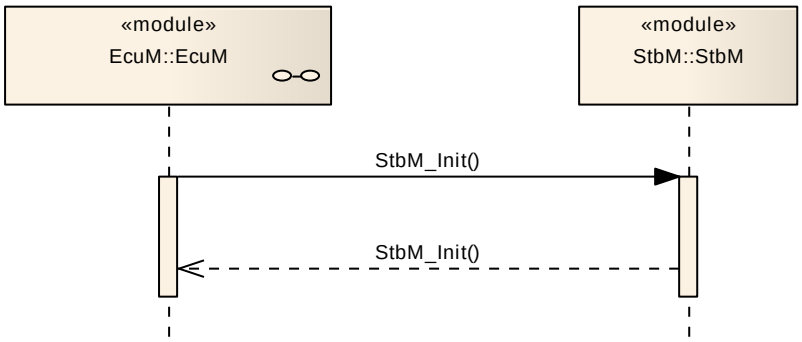
**[StbM114]** ⌈
The callback function GlobalTimeCustomerCallbackFunction shall be invoked by the Synchronized Time-Base Manager for providing the current tick value of the time-base to the *triggered customer*. ⌋ ( )

# 9 Sequence diagrams

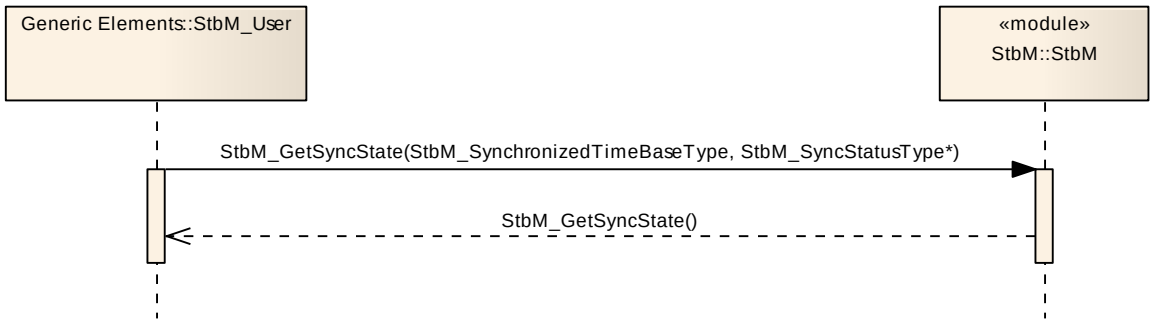
The sequence diagrams in this chapter show the basic operations of the Synchronized Time-Base Manager.

Please note that the sequence diagrams are an extension for illustrational purposes to ease understanding of the specification.

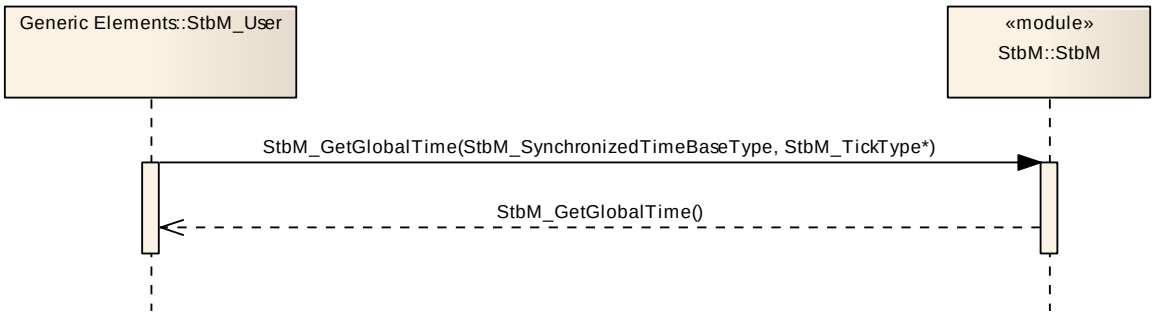
## 9.1 StbM_Init



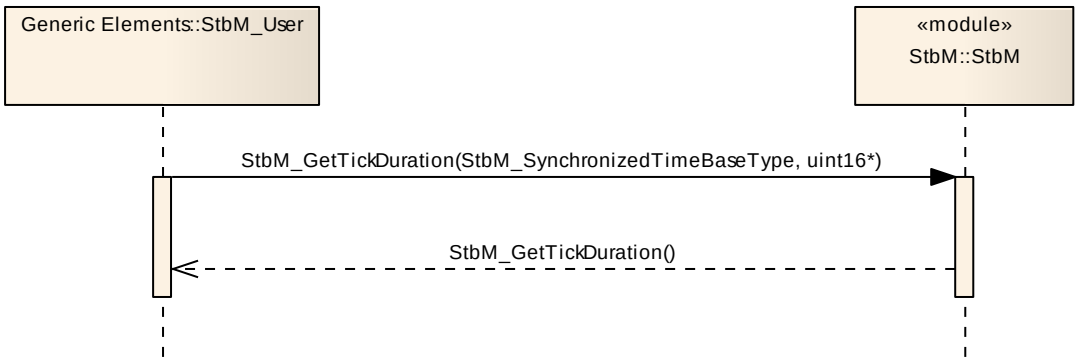

## 9.2 StbM_GetSyncState



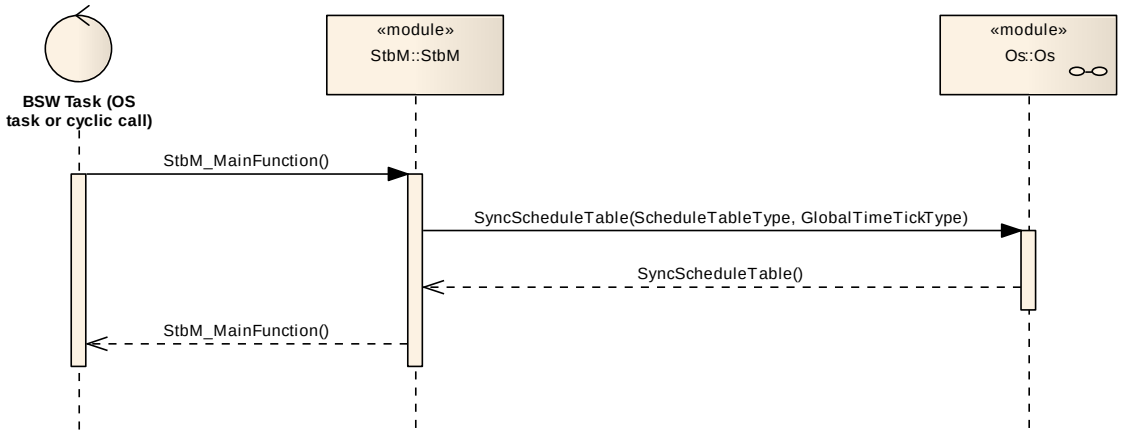
## 9.3 StbM_GetGlobalTime



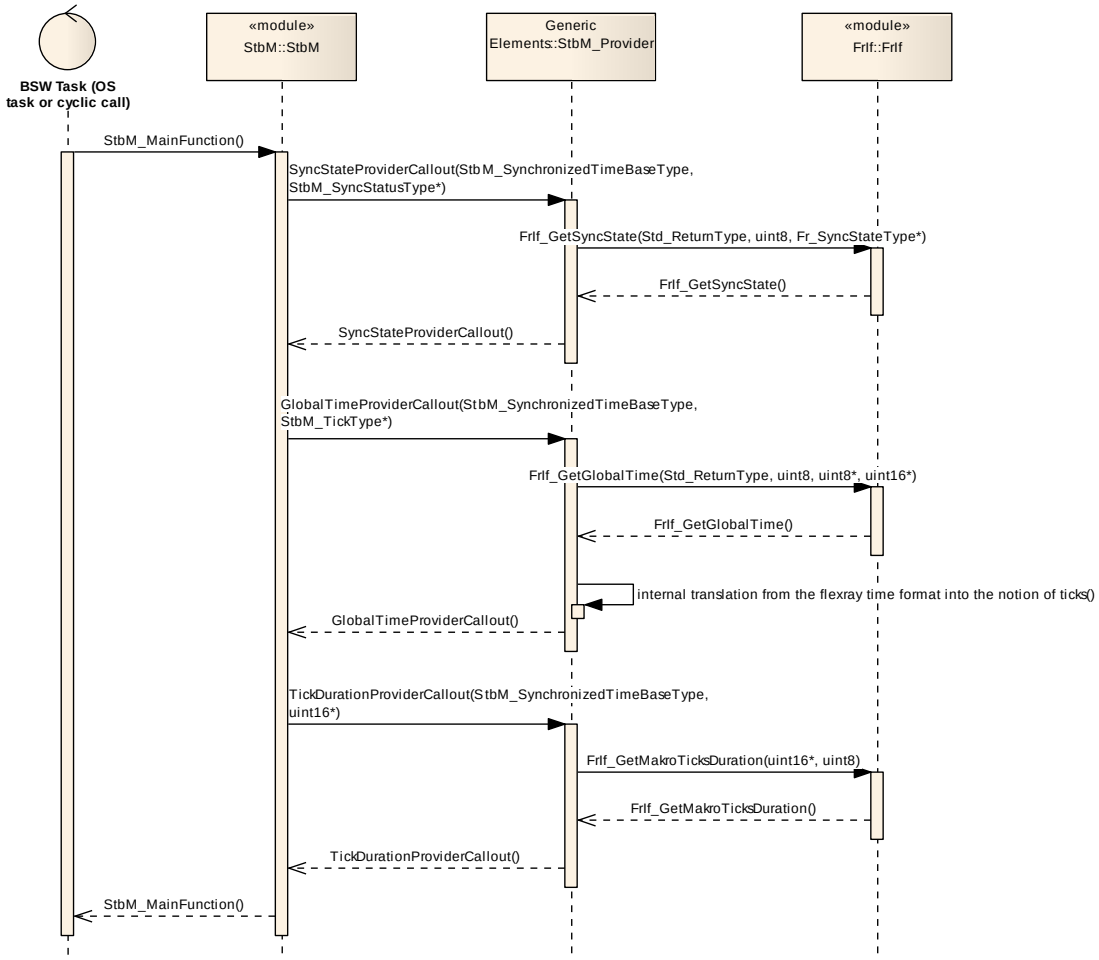

## 9.4 StbM_GetTTickDuration



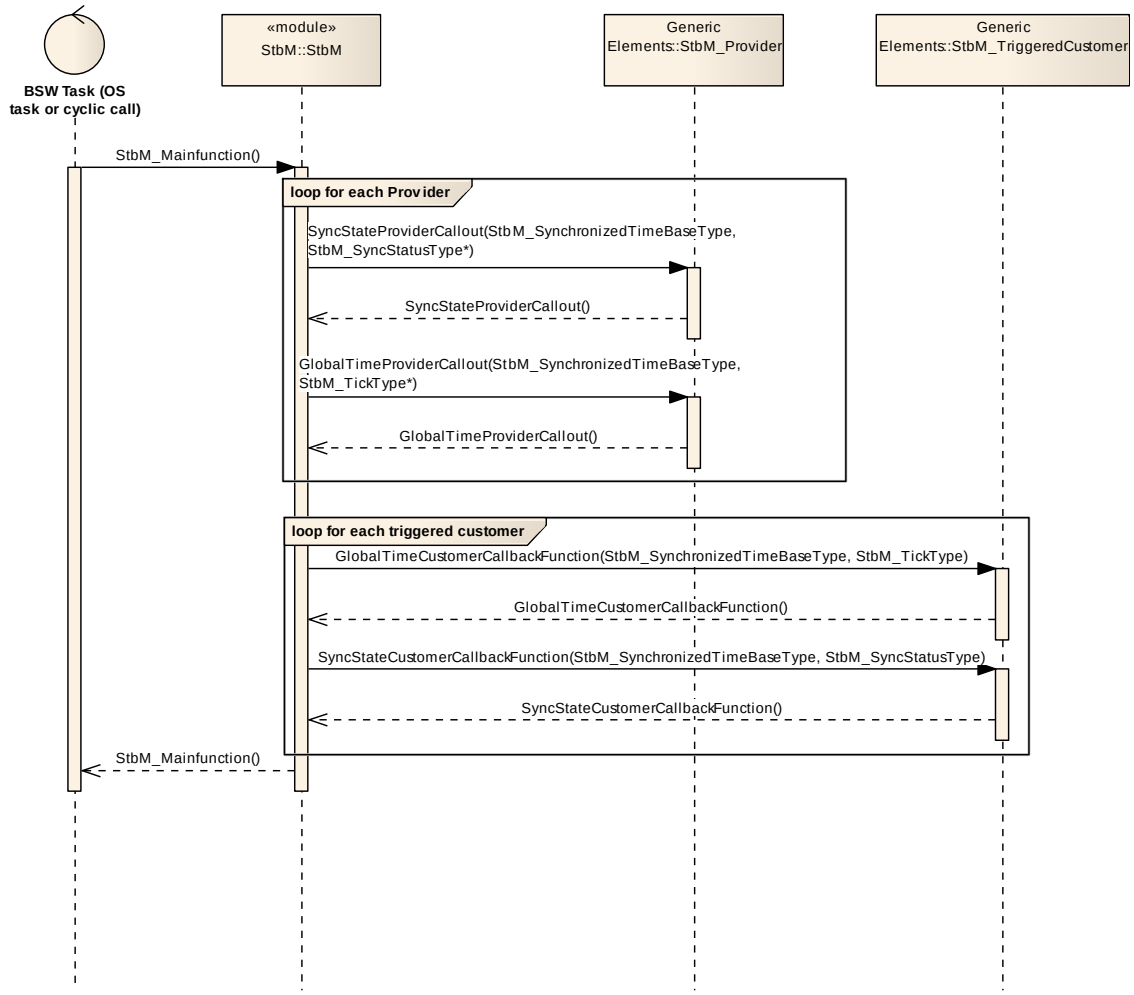

## 9.5 Explicit synchronization of OS ScheduleTable



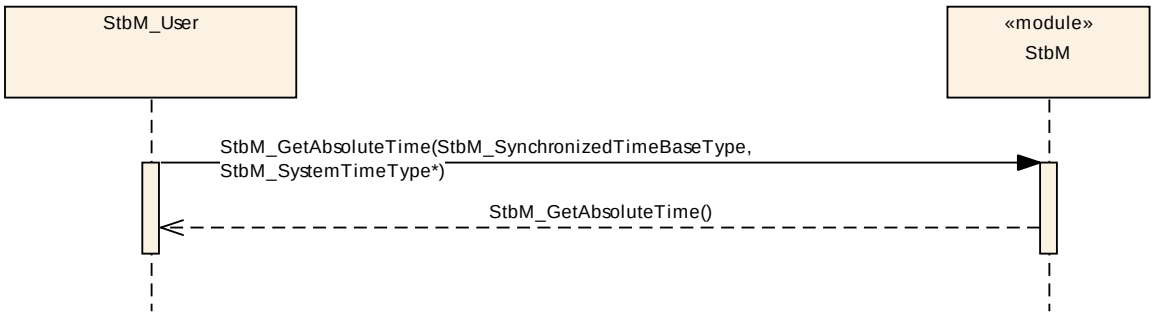
## 9.6 Provider callout with FlexRay



## 9.7 Cyclic interaction with provider and triggered customer



## 9.8 StbM_GetAbsoluteTime



# 10 Configuration specification

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

Chapter 10.2 specifies the structure (containers) and the parameters of the Synchronized Time-Base Manager
.
Chapter 10.3 specifies published information of the module Synchronized Time-Base Manager.

## 10.1 How to read this chapter

In addition to this section, it is highly recommended to read the documents:

- AUTOSAR Layered Software Architecture [2]
- AUTOSAR ECU Configuration Specification [3]
  This document describes the AUTOSAR configuration methodology and the AUTOSAR configuration metamodel in detail.

The following is only a short survey of the topic and it will not replace the ECU Configuration Specification document.

### 10.1.1 Configuration and configuration parameters

Configuration parameters define the variability of the generic part(s) of an implementation of a module. This means that only generic or configurable module implementation can be adapted to the environment (software/hardware) in use during system and/or ECU configuration.

The configuration of parameters can be achieved at different times during the software process: before compile time, before link time or after build time. In the following, the term “configuration class” (of a parameter) shall be used in order to refer to a specific configuration point in time.

### 10.1.2 Variants

Variants describe sets of configuration parameters. E.g., variant 1: only pre-compile time configuration parameters; variant 2: mix of pre-compile- and post build time-configuration parameters. In one variant a parameter can only be of one configuration class.

### 10.1.3 Containers

Containers structure the set of configuration parameters. This means:

- *all* configuration parameters are kept in containers.
- (sub-) containers can reference (sub-) containers. It is possible to assign a multiplicity to these references. The multiplicity then defines the possible number of instances of the contained parameters.

## 10.2 Containers and configuration parameters

**[StbM062]** ⌈
The following chapters summarize all configuration parameters. The detailed meanings of the parameters describe Chapters 7 and Chapter 8. ⌋(BSW159, BSW167)

**[StbM063]** ⌈
The configuration tool must check the consistency of the configuration at configuration time. ⌋(BSW167)

**[StbM064]** ⌈
Configuration rules and constraints for plausibility checks shall be performed during configuration time, wherever possible. ⌋(BSW167)

### 10.2.1 Variants

Variant1: This variant allows only pre-compile time configuration parameters.

### 10.2.2 StbM

| ***Module Name*** | *StbM* |
|---|---|
| ***Module Description*** | Configuration of the Synchronized Time-base Manager (StbM) module. |

| ***Included Containers*** | | |
|---|---|---|
| ***Container Name*** | ***Multiplicity*** | ***Scope / Dependency*** |
| StbMDemEventParameterRefs | 0..1 | Container for the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references. |
| StbMGeneral | 1 | This container holds the general parameters of the Synchronized Time-base Manager |
| StbMSynchronizedTimeBase | 1..* | Synchronized time.base collects the information about a specific time-base provider within the system. |
| StbMTriggeredCustomer | 1..* | The triggered customer is directliy triggered by the Synchronized Time-base Manager by getting synchronized with the current (global) definition of time and passage of time. |

### 10.2.3 StbMGeneral

| | |
|---|---|
| ***SWS Item*** | **STBM002_Conf :** |
| ***Container Name*** | StbMGeneral{STBM_GENERAL} |
| ***Description*** | This container holds the general parameters of the Synchronized Time-base Manager |
| ***Configuration Parameters*** | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM026_Conf :** | | |
| ***Name*** | StbMAbsoluteTimeApi {STBM_ABSOLUTE_TIME_API} | | |
| ***Description*** | Enables/Disables the StbM_GetAbsoluteTime API. | | |
| ***Multiplicity*** | 1 | | |
| ***Type*** | EcucBooleanParamDef | | |
| ***Default value*** | -- | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM012_Conf :** | | |
| ***Name*** | StbMDevErrorDetect {STBM_DEV_ERROR_DETECT} | | |
| ***Description*** | Switch for enabling the development error detection. | | |
| ***Multiplicity*** | 1 | | |
| ***Type*** | EcucBooleanParamDef | | |
| ***Default value*** | -- | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM001_Conf :** | | |
| ***Name*** | StbMTickTypeRange {STBM_TICKTYPE_RANGE} | | |
| ***Description*** | Defines the upper range of the type "StbM_TickType". | | |
| ***Multiplicity*** | 1 | | |
| ***Type*** | EcucIntegerParamDef | | |
| ***Range*** | 0 .. 65535 | | |
| ***Default value*** | -- | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | |
|---|---|
| ***SWS Item*** | **STBM013_Conf :** |
| ***Name*** | StbMVersionInfo {STBM_VERSION_INFO_API} |
| ***Description*** | Activate/Deactivate the version information API (StbM_GetVersionInfo). True: version information API activated False: version information API deactivated. |
| ***Multiplicity*** | 1 |

| | | | |
|---|---|---|---|
| ***Type*** | EcucBooleanParamDef | | |
| ***Default value*** | -- | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| |
|---|
| ***No Included Containers*** |

### 10.2.4 StbMDemEventParameterRefs

| | |
|---|---|
| ***SWS Item*** | **STBM022_Conf :** |
| ***Container Name*** | StbMDemEventParameterRefs |
| ***Description*** | Container for the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references. |
| ***Configuration Parameters*** | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM023_Conf :** | | |
| ***Name*** | STBM_E_INIT_FAILED | | |
| ***Description*** | Reference to the DemEventParameter which shall be issued when the error "StbM initialization failed" has occured. | | |
| ***Multiplicity*** | 0..1 | | |
| ***Type*** | Reference to [ DemEventParameter ] | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM024_Conf :** | | |
| ***Name*** | STBM_E_INTEGRITY_FAILED | | |
| ***Description*** | Reference to the DemEventParameter which shall be issued when the error "API request integrity failed" has occured. | | |
| ***Multiplicity*** | 0..1 | | |
| ***Type*** | Reference to [ DemEventParameter ] | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | |
|---|---|
| ***SWS Item*** | **STBM025_Conf :** |
| ***Name*** | STBM_E_REQ_FAILED |
| ***Description*** | Reference to the DemEventParameter which shall be issued when the error "API request failed" has occured. |
| ***Multiplicity*** | 0..1 |
| ***Type*** | Reference to [ DemEventParameter ] |

| | | | |
|---|---|---|---|
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| |
|---|
| ***No Included Containers*** |

### 10.2.5 StbMSynchronizedTimeBase

| | |
|---|---|
| ***SWS Item*** | **STBM003_Conf :** |
| ***Container Name*** | StbMSynchronizedTimeBase{STBM_SYNCHRONIZED_TIMEBASE} |
| ***Description*** | Synchronized time.base collects the information about a specific time-base provider within the system. |
| ***Configuration Parameters*** | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM014_Conf :** | | |
| ***Name*** | StbMGlobalTimeProviderCallout {STBM_GLOBALTIME_PROVIDER_CALLOUT} | | |
| ***Description*** | Entry address of the time-base specific callout routine which shall be invoked by the StbM for gathering the current time-base value. In case the synchronized time-base is derived from the local time, this container can be omitted. For this case, the StbM achieves the current time-base value by calling the OS interface "GetCounterValue" with the respective OSCounter configured via the ECUC param "StbMLocalTimeRef". | | |
| ***Multiplicity*** | 0..1 | | |
| ***Type*** | EcucFunctionNameDef | | |
| ***Default value*** | -- | | |
| ***maxLength*** | -- | | |
| ***minLength*** | -- | | |
| ***regularExpression*** | -- | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | |
|---|---|
| ***SWS Item*** | **STBM015_Conf :** |
| ***Name*** | StbMSyncStateProviderCallout {STBM_SYNCSTATE_PROVIDER_CALLOUT} |
| ***Description*** | Entry address of the time-base specific callout routine which shall be invoked by the StbM for gathering the current time-base status. In case the synchronized time-base is derived from the local time, this container can be omitted. For this case, the state is always "STBM_STATE_SYNC". |
| ***Multiplicity*** | 0..1 |
| ***Type*** | EcucFunctionNameDef |
| ***Default value*** | -- |
| ***maxLength*** | -- |
| ***minLength*** | -- |
| ***regularExpression*** | -- |

| | | | |
|---|---|---|---|
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM021_Conf :** | | |
| ***Name*** | StbMSynchronizedTimeBaseIdentifier {STBM_SYNCHRONIZED_TIMEBASE_IDENTIFIER} | | |
| ***Description*** | Identification of a synchronized time-base via a unique identifier. | | |
| ***Multiplicity*** | 1 | | |
| ***Type*** | EcucIntegerParamDef (Symbolic Name generated for this parameter) | | |
| ***Range*** | 0 .. 65535 | | |
| ***Default value*** | -- | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM016_Conf :** | | |
| ***Name*** | StbMTickDurationProviderCallout {STBM_TICKDURATION_PROVIDER_CALLOUT} | | |
| ***Description*** | Entry address of the time-base specific callout routine which shall be invoked by the StbM for gathering the current time-base tick duration. In case the synchronized time-base is derived from the local time, this container can be omitted. For this case, the StbM achieves the tick duration by configuring an alarm which uses the HW counter configured via the ECUC param "StbMLocalTimeRef". Note: The tick duration of the local time will not change during runtime. | | |
| ***Multiplicity*** | 0..1 | | |
| ***Type*** | EcucFunctionNameDef | | |
| ***Default value*** | -- | | |
| ***maxLength*** | -- | | |
| ***minLength*** | -- | | |
| ***regularExpression*** | -- | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM005_Conf :** | | |
| ***Name*** | StbMFlexRayClusterRef {STBM_FLEXRAY_CLUSTER_REF} | | |
| ***Description*** | Optional reference to the FlexRay cluster. | | |
| ***Multiplicity*** | 0..1 | | |
| ***Type*** | Foreign reference to [ FLEXRAY-CLUSTER ] | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | |
|---|---|
| ***SWS Item*** | **STBM006_Conf :** |
| ***Name*** | StbMLocalTimeRef {STBM_LOCAL_TIME_REF} |
| ***Description*** | Optional sub container in case a local time shall be accessed. In case this subcontainer is used, the destinated OS counter has to be configured properly, meaning: - the counter is directly driven by a HW timer - the counter's OsCounterTicksPerBase set to one tick in ms. |

| | | | |
|---|---|---|---|
| ***Multiplicity*** | 0..1 | | |
| ***Type*** | Reference to [ OsCounter ] | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM011_Conf :** | | |
| ***Name*** | StbMTtcanClusterRef {STBM_TTCAN_CLUSTER_REF} | | |
| ***Description*** | Optional reference to the Ttcan cluster. | | |
| ***Multiplicity*** | 0..1 | | |
| ***Type*** | Foreign reference to [ TTCAN-CLUSTER ] | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| |
|---|
| ***No Included Containers*** |

### 10.2.6 StbMTriggeredCustomer

| | |
|---|---|
| ***SWS Item*** | **STBM004_Conf :** |
| ***Container Name*** | StbMTriggeredCustomer{STBM_TRIGGERED_CUSTOMER} |
| ***Description*** | The triggered customer is directliy triggered by the Synchronized Time-base Manager by getting synchronized with the current (global) definition of time and passage of time. |
| ***Configuration Parameters*** | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM017_Conf :** | | |
| ***Name*** | StbMGlobalTimeCustomerCallback {STBM_GLOBALTIME_CUSTOMER_CALLBACK} | | |
| ***Description*** | Entry address of the customer specific call-back routine which shall be invoked by the StbM periodically for time value propagation. This configuration is only valid if the explicit OS ScheduleTable is NOT defined as triggered customer (via the reference "StbMOSScheduleTableRef"). | | |
| ***Multiplicity*** | 0..1 | | |
| ***Type*** | EcucFunctionNameDef | | |
| ***Default value*** | -- | | |
| ***maxLength*** | -- | | |
| ***minLength*** | -- | | |
| ***regularExpression*** | -- | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | |
|---|---|
| ***SWS Item*** | **STBM018_Conf :** |
| ***Name*** | StbMSyncStateCustomerCallback {STBM_SYNCSTATE_CUSTOMER_CALLBACK} |
| ***Description*** | Entry address of the customer specific call-back routine which shall be |

| | | | |
|---|---|---|---|
| | invoked by the StbM when state changes occur. This configuration is only valid if the explicit OS ScheduleTable is NOT defined as triggered customer (via the reference "StbMOSScheduleTableRef"). | | |
| ***Multiplicity*** | 0..1 | | |
| ***Type*** | EcucFunctionNameDef | | |
| ***Default value*** | -- | | |
| ***maxLength*** | -- | | |
| ***minLength*** | -- | | |
| ***regularExpression*** | -- | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM019_Conf :** | | |
| ***Name*** | StbMTriggerInSyncState {STBM_TRIGGER_IN_SYNCSTATE} | | |
| ***Description*** | Activate/Deactivate the triggering of the customer in case the related time-base (StbmSynchronizedTimeBaseRef) is not synchronized. True: the customer will only be triggered when the related time-base is synchronized. False: the customer will be triggered with the local time-base when no synchronization for the related time-base is established. | | |
| ***Multiplicity*** | 1 | | |
| ***Type*** | EcucBooleanParamDef | | |
| ***Default value*** | -- | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM020_Conf :** | | |
| ***Name*** | StbMTriggeredCustomerPeriod {STBM_TRIGGERED_CUSTOMER_PERIOD} | | |
| ***Description*** | The triggering period of the triggered customer, called by the StbM_MainFunction. The period is documented in microseconds. | | |
| ***Multiplicity*** | 1 | | |
| ***Type*** | EcucIntegerParamDef | | |
| ***Range*** | 0 .. 4294967295 | | |
| ***Default value*** | -- | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM007_Conf :** | | |
| ***Name*** | StbMOSScheduleTableRef {STBM_OS_SCHEDULETABLE_REF} | | |
| ***Description*** | Optional reference to synchronized OS ScheduleTables. This configuration is only valid if the triggered customer shall be an OS ScheduleTable. In this case, the OS ScheduleTable will be explicitely synchronized by the StbM. | | |
| ***Multiplicity*** | 0..1 | | |
| ***Type*** | Reference to [ OsScheduleTable ] | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| | | | |
|---|---|---|---|
| ***SWS Item*** | **STBM010_Conf :** | | |
| ***Name*** | StbMSynchronizedTimeBaseRef {SYNCHRONIZEDTIMEBASE_REF} | | |
| ***Description*** | Mandatory reference to the required synchronized time-base. | | |
| ***Multiplicity*** | 1 | | |
| ***Type*** | Reference to [ StbMSynchronizedTimeBase ] | | |
| ***ConfigurationClass*** | ***Pre-compile time*** | X | All Variants |
| | ***Link time*** | -- | |
| | ***Post-build time*** | -- | |
| ***Scope / Dependency*** | | | |

| |
|---|
| ***No Included Containers*** |

## 10.3 Published Information

**[StbM139]** ⌈The standardized common published parameters as required by BSW00402 in the SRS General on Basic Software Modules [12] shall be published within the header file of this module and need to be provided in the BSW Module Description. The according module abbreviation can be found in the List of Basic Software Modules [13]. ⌋(BSW00402)

# 11 AUTOSAR Service implemented by the Synchronized Time-Base Manager

## 11.1 Scope of this chapter

This chapter is an addition to the specification of the Synchronized Time-Base Manager. Whereas the other parts of the specification define the behaviour and the C-interfaces of the corresponding basic software module, this chapter formally specifies the corresponding AUTOSAR Service in terms of the SWC Template. The interfaces described here will be visible on the VFB and are used to generate the RTE between application software components and the Synchronized Time-Base Manager.

## 11.2 Overview

### 11.2.1 Architecture

In the AUTOSAR ECU Architecture (see [2]) the StbM module implements an AUTOSAR Service as indicated in Figure 2.

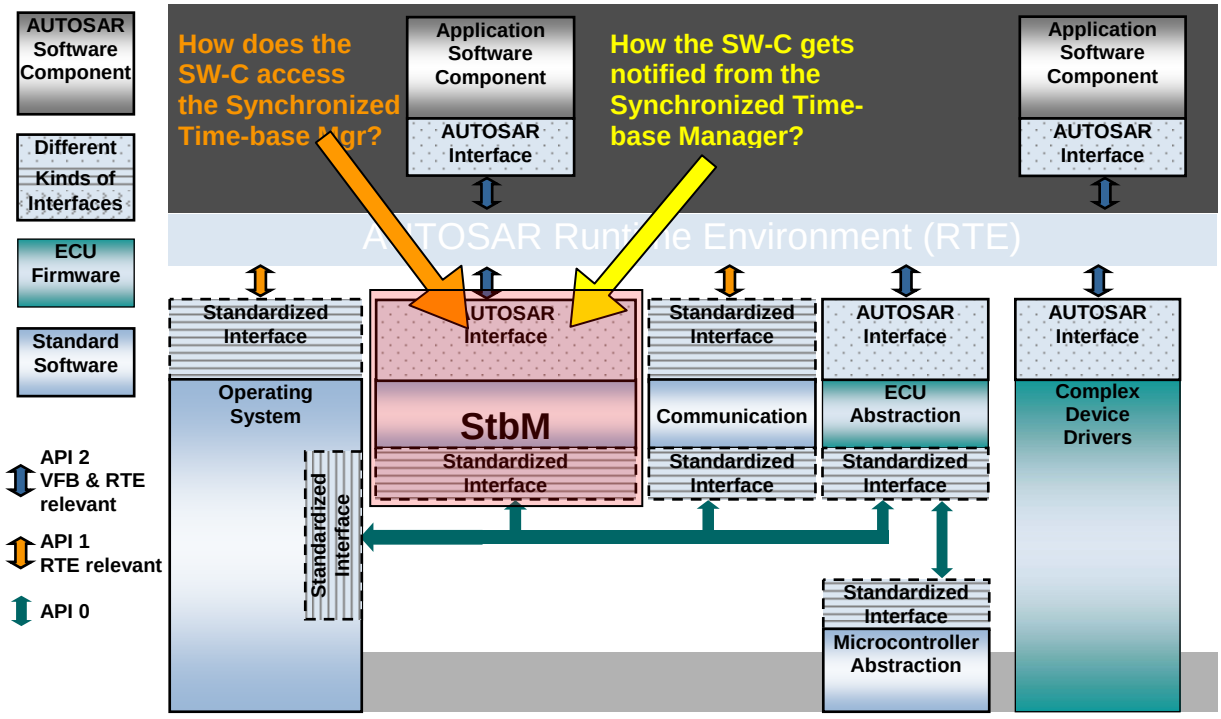



**Figure 2: The Synchronized Time-Base Manager in the ECU architecture**

From the viewpoint of the BSW module “Synchronized Time-Base Manager”, there are two kinds of dependencies between the service and application software components:

- The application accesses the Synchronized Time-Base Manager

- The application is optionally triggered with the value and/or notified upon state changes of the associated time-base. This invocation is initiated by the Synchronized Time-Base Manager.

These dependencies must be described in terms of the AUTOSAR meta-model which will contribute to the SW-C Description of the application component as well as to the SW-C Description of the StbM Service.

### 11.2.2 Requirements

There are three sources of requirements for this specification:

- The requirements for the functionality of the Synchronized Time-Base Manager are specified in [1].
- In order to model the VFB view of the service, the chapter on AUTOSAR Services of the VFB specification [7] has to be considered as an additional requirement.
- For the formal description of the SW-C attributes [8] gives the requirements.

### 11.2.3 Use Cases

As shown in Figure 1, the Synchronized Time-Base Manager is either interacting with the role *customer* or with the role *provider*. Application software components can only act as customers, thus the interaction with the Synchronized Time-Base Manager via the RTE shall be done in the following use cases:

- **Application software component accesses the Synchronized Time-Base Manager:** The customer autonomously calls the Synchronized Time-Base Manager, getting knowledge about the definition of time and the state of the module.
- **Synchronized Time-Base Manager notifies application software component:** The Synchronized Time-Base Manager informs the customer about state changes and/or error occurrences (e.g. the synchronisation state of a time-base has changed).

## 11.3 Specification of the Ports and Port Interfaces

This chapter specifies the ports and port interfaces which are needed in order to operate the Synchronized Time-Base Manager functionality over the VFB. Note that there are ports on both sides of the RTE: The SW-C description of the Synchronized Time-Base Manager defines the ports below the RTE. Each software component which uses the service must contain “service ports” in its own SW-C description which are connected to the ports of the Synchronized Time-Base Manager, so that the RTE can be generated.

### 11.3.1 Ports and Port Interfaces for accessing the service

#### 11.3.1.1 Data Types

This chapter describes the data types which will be used in the port interfaces for accessing the Synchronized Time-Base Manager.

**[StbM124]** ⌈
The data types `uint8`, `uint16` and `sint32` used in the interfaces refer to the basic AUTOSAR data types.

The data type `StbM_SyncStatusType` indicates the status of the synchronized time-base as follows:

```
IntegerType StbM_SyncStatusType {
      lowerLimit = 0
      upperLimit = 1
};
```

The following constants are defined within this type:

```
0 -> STBM_STATE_SYNC
1 -> STBM_STATE_NOT_SYNC⌋()
```

#### 11.3.1.2 Port Interfaces

**[StbM125]** ⌈
In order to allow the application software components to access the **status** and the **value** of the synchronized time-base, the Client-Server Interface must have the following properties:

```
ClientServerInterface StbM_TimeBaseValue {
      /* access the current status of the synchronized time-base */
      GetSyncState(OUT StbM_SyncStatusType syncState);

      /* access the current definition of time represented by the notion of
      ticks*/
      GetGlobalTime(OUT StbM_TickType ticks);

      /* access the current definition of tickDuration */
      GetTickDuration(OUT uint16 tickDuration);
}
```

⌋()

**[StbM126]** ⌈
Using operation “GetSyncState()”, the application has the possibility to access the current status of the synchronized time-base. The other two operations of the interface are required for the access to the clock time: a) the tick value of the associated time-base (operation “GetTime()”), b) the duration of one tick in ms (operation “GetTickDuration()”).⌋()

Note: The interface itself does not specify which synchronized time-base should be referenced. It does only specify how the information from the time-base can be

accessed. The following sections show, how the required synchronized time-base can be documented by augmenting the definition of Ports.

**[StbM138]** ⌈
In order to allow the application software components to access the **status,** the **value** and the number of **timer overflows** of the synchronized time-base, the Client-Server Interface must have the following properties:

```
ClientServerInterface StbM_AbsoluteTimeBaseValue {
      /* access the current status of the synchronized time-base */
      GetSyncState(OUT StbM_SyncStatusType syncState);

      /* access the current definition of time represented by the notion of
      ticks*/
      GetGlobalTime(OUT StbM_TickType ticks);

      /* access the current definition of tickDuration */
      GetTickDuration(OUT uint16 tickDuration);

      /* access the absolute time since ECU start */
      GetAbsoluteTime(OUT StbM_SystemTimeType systemTime);
}
```

⌋ ( )

#### 11.3.1.3 Ports

**[StbM127]** ⌈
Each software component must provide exactly one Port for each synchronized time-base it wants to access. ⌋ ( )

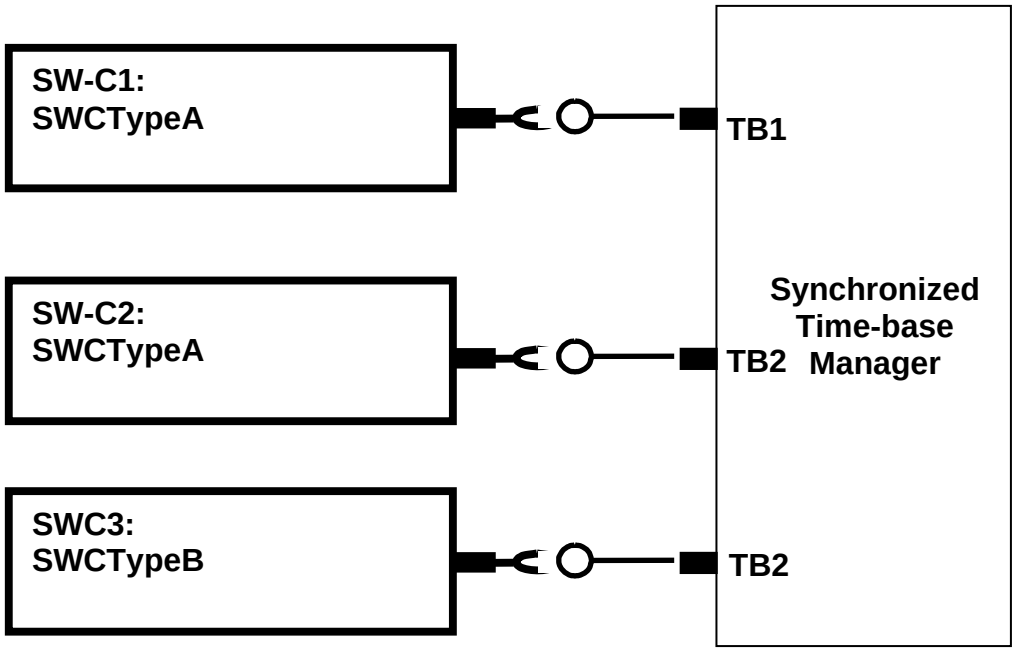



**Figure 3: Example of application software components connected to the Synchronized Time-Base Manager via service ports. On the left side, there are two instances of component SWCTypeA and one instance of component SWCTypeB. The Port names on the right side define the requested synchronized time-base. No notification ports are configured.**

On the software component side, there is one Port for each synchronized time-base. For each Port component side, a respective Port on the service side exists. The name of the Ports on service side defines which synchronized time-base is provided by the service. These names (TB1 for FlexRay time-base and TB2 for TTCan time-base) are examples and they are not standardized[2].
The RTE generator takes the port name and uses it as parameter of the module API calls.

### 11.3.2 Ports and Port Interfaces for application triggering and notification

As depicted in section 1.4, some application components may want to get informed by the Synchronized Time-Base Manager when state changes occur. Thus, some kind of notification mechanism is required. The Synchronized Time-Base Manager uses this mechanism whenever such information should be forwarded to interested application software components.

#### 11.3.2.1 Port Interfaces

**[StbM128]** ⌈In order to notify the application about changes/errors in the Synchronized Time-Base Manager, the following AUTOSAR standardized interface must be implemented:

```
SenderReceiverInterface StbM_TimeBase_TriggerCustomer {
      /* DataElement which provides the current value of the synchronized
         time-base*/
      StbM_TickType ticks;
}

SenderReceiverInterface StbM_TimeBase_StateNotification {
      /* DataElement which provides the current state of the synchronized
         time-base*/
      StbM_SyncStatusType syncState;
}
```

⌋( )

**[StbM129]** ⌈The Synchronized Time-Base Manager will trigger the first interface (StbM_TimeBase_TriggerCustomer) periodically when the value of the respective time-base has been updated. The second interface will be triggered (StbM_TimeBase_StateNotification) whenever a state change is detected. ⌋( )

Note:
There is no need to submit the respective time-base (e.g. as parameter by using a C/S interface instead of an S/R interface). Ports, which are communicating with the Synchronized Time-Base Manager are grouped together via the ServiceNeeds (see the Software Component Template specification for more information about ServiceNeeds [8]), when they are asking for the same time-base.

[2] Obviously, it does not make sense to ask for the TTCan global time as synchronized time-base and invoking the operation “GetFrTime()”, getting the FlexRay time representation as clock time format. However, in order to define the Interface between application software component and Synchronized Time-base Manager as simple as possible, the Interface would allow such a invocation.

#### 11.3.2.2 Ports

**[StbM130]** ⌈In analogy to the Port definition for above, each application software component must provide:

- One Port for each synchronized time-base, about whose time value he wants to get informed periodically.
- One Port for each synchronized time-base, about whose state changes he wants to get informed. ⌋ ( )

## 11.4 InternalBehavior

**[StbM131]** ⌈
The InternalBehavior of the Synchronized Time-Base Manager is only seen by the local RTE. Besides the definition of the time-base identifiers as port defined arguments, it must specify the operation invoked runnables:

```
InternalBehavior TimeService {

      // definition of associated operation-invoked RTE-events not shown
      // (it is done in the same way as for any SWC type)

      // section “runnable entities”:
      RunnableEntity GetSyncState
            symbol “StbM_GetSyncState”
            canbeInvokedConcurrently = TRUE

      RunnableEntity GetGlobalTime
            symbol “StbM_GetGlobalTime”
            canbeInvokedConcurrently = TRUE

      RunnableEntity GetTickDuration
            symbol “StbM_GetTickDurartion”
            canbeInvokedConcurrently = TRUE

      RunnableEntity GetAbsoluteTime
            symbol “StbM_GetAbsoluteTime”
            canbeInvokedConcurrently = TRUE
      // end of section “runnable entities”

};
```

⌋ ( )

# 12 Not applicable requirements

**[StbM140]** ⌈These requirements are not applicable to this specification.⌋ (BSW00344, BSW00404, BSW00405, BSW170, BSW00380, BSW00412, BSW00387, BSW00398, BSW00399, BSW00400, BSW00438, BSW00375, BSW00416, BSW00437, BSW168, BSW00426, BSW00427, BSW00428, BSW00432, BSW00433, BSW00336, BSW00422, BSW00417, BSW00455, BSW161, BSW162, BSW005, BSW00415, BSW164, BSW00325, BSW00326, BSW00342, BSW160, BSW00453, BSW007, BSW00413, BSW00347, BSW00441, BSW00307, BSW00314, BSW00370, BSW00353, BSW00361, BSW00328, BSW00312, BSW006, BSW00439, BSW00304, BSW00355, BSW00378, BSW00306, BSW00308, BSW00309, BSW00371, BSW00359, BSW00360, BSW00440, BSW00330, BSW009, BSW010, BSW00333, BSW00341, BSW00334)