MICROSAR StbM

Technical Reference

Version 5.8.1

Authors	Stephanie Schaaf, Buesra Bayrak, Anant Gupta, Stephanie Baumgartner, Thilo Rachlitz, Bernd Sigle
Status	Released

Document Information

History

Author	Date	Version	Remarks
Stephanie Schaaf	2012-12-06	1.0.0	Initial version
Stephanie Schaaf	2013-04-10	1.1.0	Set status to released Added StbM_InitMemory Added description of critical sections
Stephanie Schaaf	2013-10-30	2.0.0	Fixed review findings
Stephanie Schaaf	2014-11-05	3.0.0	Support new global time synchronization concept
Stephanie Schaaf	2015-01-29	3.1.0	Renamed document
Stephanie Schaaf	2015-08-12	3.2.0	Changed StbM_TimeStampRawType to structure
Stephanie Schaaf	2016-01-08	4.0.0	Support of High Resolution Time Base Reference Clock based on GPT Added EthTSyn_SetGlobalTime to used services
Stephanie Schaaf	2016-03-22	4.1.0	Support of time gateways
Stephanie Schaaf	2017-03-29	4.2.0	Support of time correction Removed EthTSyn from used services Added EthIf to used services
Stephanie Schaaf	2017-06-29	5.1.0	Support time precision measurement
Buesra Bayrak, Anant Gupta, Stephanie Baumgartner Thilo Rachlitz	2017-08-01	5.2.0	New Notifications for time expiration and status changed events Support of immediate time synchronization Added handling of time leaps
Thilo Rachlitz, Stephanie Baumgartner	2017-09-27	5.3.0	Minor AR4.3 extensions Added StbM_GetTimeBaseStatus Changed availability of service ports and operations according to AR 4.3
Bernd Sigle	2018-02-28	5.4.0	Updated AUTOSAR architecture Updated referenced documents
Stephanie Baumgartner	2018-05-22	5.5.0	Support schedule table synchronization for different counter resolutions Support time base specific time difference calculation according to AR 4.3.1 Improved exclusive area description
Stephanie Baumgartner	2018-07-16	5.6.0	First ASIL D version Minor corrections

Thilo Rachlitz	2018-08-14	5.7.0	StbM avoids accessing an uncertain Ethernet clock source Added description of parameter 'ConfigPtr' of StbM_Init() Updated chapter 4.1 to new template
Stephanie Baumgartner	2018-10-22	5.8.0	MISRA-C:2012 compliance Updated used services
Bernd Sigle	2019-01-23	5.8.1	Minor improvements

Reference Documents

No.	Source	Title	Version
[1]	AUTOSAR	AUTOSAR_SWS_SynchronizedTimeBaseManager.pdf	4.3.0
[2]	AUTOSAR	AUTOSAR_TR_BSWModuleList.pdf	4.2.2
[3]	AUTOSAR	AUTOSAR_SWS_DefaultErrorTracer.pdf	4.2.2
[4]	AUTOSAR	AUTOSAR_SWS_RTE.pdf	4.2.2
[5]	AUTOSAR	AUTOSAR_SWS_OS.pdf	4.2.2
[6]	AUTOSAR	AUTOSAR_SWS_EthernetInterface.pdf	4.3.0
[7]	AUTOSAR	AUTOSAR_SWS_GPTDriver.pdf	4.2.2

Scope of the Document

This technical reference describes the general use of the Synchronized Time-Base Manager.

Caution
We have configured the programs in accordance with your specifications in the questionnaire. Whereas the programs do support other configurations than the one specified in your questionnaire, Vector´s release of the programs delivered to your company is expressly restricted to the configuration you have specified in the questionnaire.

Contents

Illustrations

Tables

1 Component History

The component history gives an overview over the important milestones that are supported in the different versions of the component.

Component Version	New Features
1.0.0	Initial creation
1.1.0	Added StbM_InitMemory
3.0.0	Support new global time synchronization concept
4.0.0	Support of High Resolution Time Base Reference Clock based on GPT
4.1.0	Support of time gateways
4.2.0	Support of time correction
5.1.0	Support of time precision measurement
5.2.0	Support of immediate time synchronization
5.3.0	Minor improvements
5.4.0	Module refactored
5.5.0	Support schedule table synchronization for different counter resolutions Support time base specific time difference calculation acc. to AR 4.3.1
5.6.0	First ASIL D version
5.7.0	Avoid accessing an uncertain Ethernet clock source
5.8.0	MISRA-C:2012 compliance
5.8.1	Minor improvements and corrections

Table 1-1 Component history

2 Introduction

This document describes the functionality, API and configuration of the AUTOSAR BSW module StbM as specified in [1].

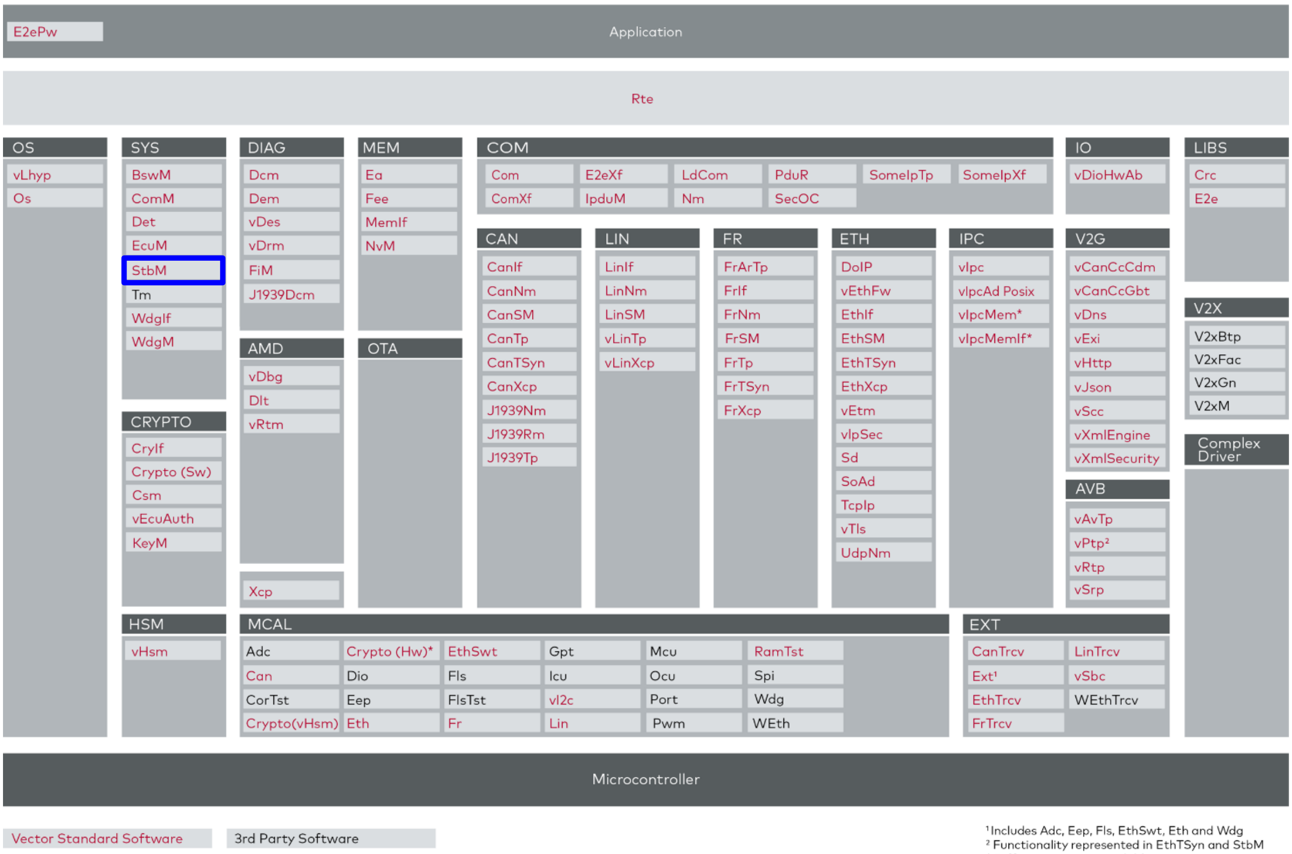
Supported AUTOSAR Release*:	4	
Supported Configuration Variants:	pre-compile	
Vendor ID:	STBM_VENDOR_ID	30 decimal (= Vector-Informatik, according to HIS)
Module ID:	STBM_MODULE_ID	160 decimal (according to ref. [2])

* For the precise AUTOSAR Release 4.x please see the release specific documentation.

The purpose of the Synchronized Time-Base Manager is to provide synchronized time bases to its customers, i.e., time bases, which are synchronized with time bases on other nodes of a distributed system.

2.1 Architecture Overview

The following figure shows where the StbM is located in the AUTOSAR architecture.



[1] Includes Adc, Eep, Fls, EthSwt, Eth and Wdg
[2] Functionality represented in EthTSyn and StbM
* Different variants available

Figure 2-1 AUTOSAR 4.3 Architecture Overview

The next figure shows the interfaces to adjacent modules of the StbM. These interfaces are described in chapter 5.

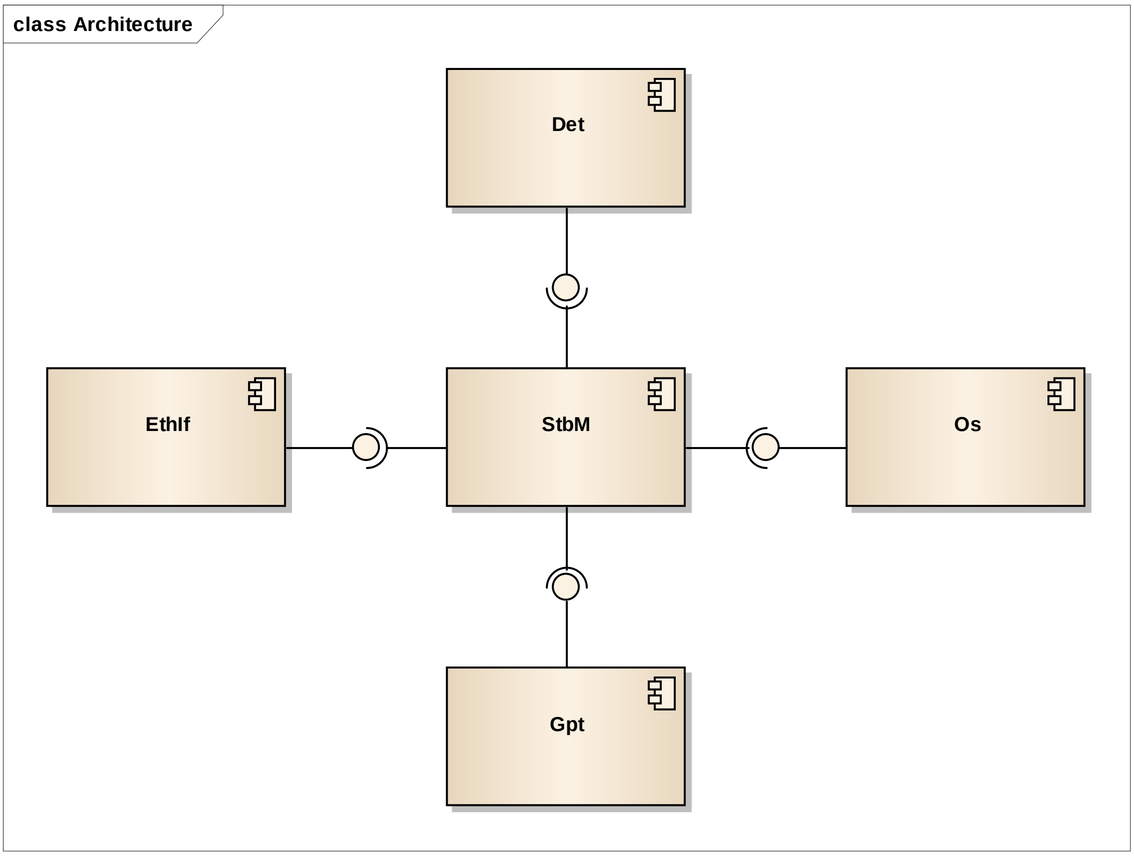


Figure 2-2 Interfaces to adjacent modules of the StbM

Applications do not access the services of the BSW modules directly. They use the service ports provided by the BSW modules via the RTE. The service ports provided by the StbM are listed in chapter 5.5 and are defined in [1].

3 Functional Description

3.1 Features

The features listed in the following tables cover the complete functionality specified for the StbM.

The AUTOSAR standard functionality is specified in [1], the corresponding features are listed in the tables

- Table 3-1 Supported AUTOSAR standard conform features
- Table 3-2 Not supported AUTOSAR standard conform features

Vector Informatik provides further StbM functionality beyond the AUTOSAR standard. The corresponding features are listed in the table

- Table 3-3 Features provided beyond the AUTOSAR standard

The following features specified in [1] are supported:

Supported AUTOSAR Standard Conform Features
Synchronization of runnable entities and OS schedule tables
Provision of absolute time value
Autonomous maintenance of the time-base
Support of time gateways
Support of High Resolution Time Base Reference Clock based on GPT
Support of time correction
Support of time precision measurement
Support of status notifications
Support of Time Notifications
Support of immediate time synchronization

Table 3-1 Supported AUTOSAR standard conform features

3.1.1 Deviations

The following features specified in [1] are not supported:

Not Supported AUTOSAR Standard Conform Features
Storage of the time base at shutdown
Loading of the time base during initialization
Variant Post-Build
Complex Device Driver Interface

Table 3-2 Not supported AUTOSAR standard conform features

3.1.2 Additions/ Extensions

The following features are provided beyond the AUTOSAR standard:

Features Provided Beyond The AUTOSAR Standard
Memory Initialization

Table 3-3 Features provided beyond the AUTOSAR standard

3.1.2.1 Memory Initialization

AUTOSAR expects the startup code to automatically initialize RAM. Not every startup code of embedded targets reinitializes all variables correctly. It is possible that the state of a variable may not be initialized as expected. To avoid this problem the Vector AUTOSAR StbM provides an additional function to initialize the relevant variables of the StbM. See also chapters 3.2 and 5.2.2 for details.

3.1.3 Limitations

There are no known limitations.

3.1.4 Notifications for time expiration and status changed events

Time notifications or status changed events can be registered by notification customers. A notification customer can either be a SW-C or a BSW-module. The StbM allows customers to be notified when an alarm expires or any of the registered status changed events occurs.

3.1.4.1 Status Notifications

The StbM allows notification customers to register for one or more status change events. Whenever a status change is triggered the customer is informed by a notification callback. This callback function is called in the context of the `StbM_MainFunction()`. However, this may result in a delay from the point of time when the event occurred and when the customer is actually notified. To decrease this delay, a smaller MainFunction cycle time needs to be configured.

Since the notification of the customers is realized via service ports the delay is also influenced by the task mapping in the RTE and the OS task configuration. If a customer has to be notified as early as possible, the priority of the task the notification runnable is mapped to has to be increased. Otherwise this task will wait until other higher priority tasks are finished.

Table 5-12 lists all events which can be detected.

3.1.4.2 Time Notifications

The StbM allows notification customers to be notified whenever a defined timer expires. This expire time is set through function `StbM_StartTimer` by the notification customer.

A GPT timer is used to monitor the timeout of the expiration timer within a configurable interval. After the GPT timer expires, `StbM_TimerCallback` is called to notify StbM about the expiration. The GPT timer needs to be referenced in

> /MICROSAR/StbM/StbMGeneral/StbMGptTimerRef

Further, the referenced GPT Timer has to configure `StbM_TimerCallback` as callback function in

> /Gpt/GptChannelConfigSet/GptChannelConfiguration/GptNotification

`StbM_TimerCallback` might be called in interrupt context. However, customer notifications are called in task context to decouple the application from the interrupt context. Therefore, the user has to map the function `StbM_NotificationFunction` to an existent task. Depending on the priority of the task, the time deviation between the calculated expire time and actual time may be more or less. This time deviation is provided along with the callback function. If a notification customer has to be notified as early as possible, the priority of this task has to be increased. Otherwise this task will wait until other higher priority tasks are finished.

It should be noted that this time notification feature is only compatible with the Microsar RTE.

Caution
StbM_NotificationFunction only works with MICROSAR RTE.

The provided expire time in `StbM_StartTimer` must, at least, exceed the cycle time of the Main Function to ensure that the timer can be monitored properly. During runtime, the start of a timer may also be denied, if the GPT Timer is currently running and the newly provided expire time would expire before expiration of the GPT Timer.

3.2 Initialization

The Synchronized Time-Base Manager is initialized by calling `StbM_Init()`. This is done by the ECU State Manager (EcuM).

On platforms in which the Random Access Memory (RAM) is not initialized to zero by the startup code the function `StbM_InitMemory` has to be called first and then a call to `StbM_Init` can be realized.

3.3 States

The StbM has no internal state machine, it is operational after initialization.

3.4 Main Functions

The `StbM_MainFunction()` updates the local time bases, monitors timeouts for the detection of lost synchronization and triggers customers, which includes the synchronization of OS ScheduleTables and notification of customers about status related events. Besides the `StbM_MainFunction()` notifies the application about new available measurement data blocks and starts a GPT timer, if a time notification customer wants to be notified when an alarm expires.

3.5 Error Handling

3.5.1 Development Error Reporting

By default, development errors are reported to the DET using the service `Det_ReportError()` as specified in [3], if development error reporting is enabled (i.e. pre-compile parameter `STBM_DEV_ERROR_DETECT==STD_ON`).

If another module is used for development error reporting, the function prototype for reporting the error can be configured by the integrator, but must have the same signature as the service `Det_ReportError()`.

The reported StbM ID is 160.

The reported service IDs identify the services which are described in 5.2. The following table presents the service IDs and the related services:

Service ID	Service
0x00	StbM_Init
0x04	StbM_MainFunction
0x05	StbM_GetVersionInfo
0x07	StbM_GetCurrentTime
0x08	StbM_GetCurrentTimeExtended
0x09	StbM_GetCurrentTimeRaw
0x0A	StbM_GetCurrentTimeDiff
0x0B	StbM_SetGlobalTime
0x0C	StbM_SetUserData
0x0D	StbM_SetOffset
0x0E	StbM_GetOffset
0x0F	StbM_BusSetGlobalTime
0x10	StbM_UpdateGlobalTime
0x11	StbM_GetRateDeviation
0x12	StbM_SetRateCorrection
0x13	StbM_GetTimeLeap
0x14	StbM_GetTimeBaseStatus
0x15	StbM_StartTimer
0x16	StbM_GetSyncTimeRecordHead
0x17	StbM_GetOffsetTimeRecordHead
0x1B	StbM_GetTimeBaseUpdateCounter
0x1C	StbM_TriggerTimeTransmission

Table 3-4 Service IDs

The errors reported to DET are described in the following table:

Error Code		Description
0x0A	STBM_E_PARAM	API requests called with wrong parameter
0x0B	STBM_E_NOT_INITIALIZED	Synchronized Time-Base Manager is not initialized
0x10	STBM_E_PARAM_POINTER	Invalid pointer in parameter list
0x12	STBM_E_SERVICE_DISABLED	API disabled by configuration

Table 3-5 Errors reported to DET

3.5.2 Production Code Error Reporting

No production error codes are currently used by StbM.

4 Integration

This chapter gives necessary information for the integration of the MICROSAR StbM into an application environment of an ECU.

4.1 Embedded Implementation

The delivery of the StbM consists out of these files:

File Name	Description	Integration Tasks
StbM.c	Main implementation file of the StbM.	-
StbM.h	Main header file of the StbM.	-
StbM_Types.h	Header file that contains the type definitions of the StbM.	-
StbM_Cfg.c	Generated file that contains definitions of structures in pre-compile-time variant.	-
StbM_Cfg.h	Generated file that contains declarations of structures in pre-compile-time variant.	-

Table 4-1 Implementation files

4.2 Critical Sections

The StbM has code sections which need protection against interrupts and OS tasks which can interrupt each other. Therefore, the StbM uses one exclusive area which requires a global interrupt lock:

STBM_EXCLUSIVE_AREA_0

Depending on the StbM configuration, the StbM calls OS APIs like GetCounterValue() and GetElapsedValue(). According the AUTOSAR OS specification it is not allowed to call these OS APIs with disabled interrupts. Nevertheless, the StbM module requires the interrupt lock to be able to guarantee high accuracy and data consistency.

Caution

If the StbM is configured to use OS APIs and the implementation method of the exclusive area is configured to OS_INTERRUPT_BLOCKING or ALL_INTERRUPT_BLOCKING, the OS may report the error E_OS_DISABLEDINT notified by the OS ErrorHook(). In that case the implementation method of the exclusive area STBM_EXCLUSIVE_AREA_0 inside the RTE / SchM configuration needs to be set to OS_RESOURCE or CUSTOM.

If OS_RESOURCE is selected the ISR(s) of the bus specific TSyn modules e.g. the CAN ISR(s) need to reference the OS Resource created by the RTE.

If CUSTOM is selected the SchM APIs for entering and exiting the exclusive area need to be implemented manually by using an interrupt lock mechanism but without calling OS APIs like SuspendOSInterrupts() or DisableAllInterrupts().

Note:

The exclusive area implementation method CUSTOM is a MICROSAR RTE extension and might not be available in other RTEs.

For details about exclusive areas refer to [4].

4.3 OSScheduleTable Synchronization

For the synchronization of OS schedule tables by the StbM an OS is needed that supports the synchronization of schedule tables. Furthermore it is required, that the ticks of an OS counter, which drives a schedule table, have a duration of at most 1 microsecond.

5 API Description

For an interfaces overview please see Figure 2-2.

5.1 Type Definitions

The types defined by the StbM are described in this chapter.

Type Name	C-Type	Description	Value Range
StbM_SynchronizedTimeBaseType	uint16	Variables of this type are used to represent the kind of synchronized time-base.	`0..2^16-1`
StbM_RateDeviationType	sint16	Variables of this type are used to express a rate deviation in ppm.	`-32000..32000`
StbM_TimeDiffType	sint32	Variables of this type are used to express time differences as signed values in nanoseconds	`-2147483647 .. 2147483647`
StbM_CustomerIdType	uint16	Unique identifier of a notification customer	`0..255`

Table 5-1 Type definitions

[StbM_TimeBaseStatusType]

This structure is used to express if and how a local time base is synchronized to the global time master. The type is a bit field of individual status bits, although not every combination is possible, i.e. any of the bits STBM_TIMEOUT, STBM_TIMELEAP and STBM_SYNC_TO_GATEWAY can only be set if the STBM_GLOBAL_TIME_BASE bit is set.

Struct Element Name	Kind	Mask	Description	Value Range
TIMEOUT	bit	0x01	Bit 0	`0x00` No timeout on receiving Synchronization Messages
				`0x01` Timeout on receiving Synchronization Messages
SYNC_TO_GATEWAY	bit	0x04	Bit 2	`0x00` Local Time Base is synchronous to Global Time Master
				`0x01` Local Time Base updates are based on a Time Gateway below the Global Time Master
GLOBAL_TIME_BASE	bit	0x08	Bit 3	`0x00` Local Time Base is based on Local Time Base reference clock only (never synchronized with Global Time Base)

Struct Element Name	Kind	Mask	Description	Value Range
				0x01 Local Time Base was at least one time synchronized with Global Time Base
TIMELEAP_FUTURE	bit	0x10	Bit 4	0x00 No leap into the future within the received time for Time Base
				0x01 Leap into the future within the received time for Time Base exceeds a configured threshold
TIMELEAP_PAST	bit	0x20	Bit 5	0x00 No leap into the past within the received time for Time Base
				0x01 Leap into the past within the received time for Time Base exceeds a configured threshold

Table 5-2 StbM_TimeBaseStatusType

[StbM_TimeStampType]

This structure is used for expressing time stamps including relative time and absolute calendar time.

Struct Element Name	C-Type	Description	Value Range
timeBaseStatus	StbM_TimeBase StatusType	Status of the Time Base	0..2^4-1
nanoseconds	uint32	Nanoseconds part of the time	0..999999999
seconds	uint32	32 bit LSB of the 48 bits Seconds part of the time	0..2^32-1
secondsHi	uint16	16 bit MSB of the 48 bits Seconds part of the time	0..2^16-1

Table 5-3 StbM_TimeStampType

[StbM_TimeStampRawType]

This structure is used for expressing time stamps in raw format in nanoseconds only.

Struct Element Name	C-Type	Description	Value Range
nanoseconds	uint32	Time nanoseconds	0..2^32-1

Table 5-4 StbM_TimeStampRawType

[StbM_TimeStampExtendedType]

This structure is used for expressing time stamps including relative time and absolute calendar time.

Struct Element Name	C-Type	Description	Value Range
timeBaseStatus	StbM_TimeBase StatusType	Status of the Time Base	`0..2^4-1`
nanoseconds	uint32	Nanoseconds part of the time	`0..999999999`
seconds	uint64	48 bit Seconds part of the time	`0..2^48-1`

Table 5-5 StbM_TimeStampExtendedType

[StbM_UserDataType]

This structure is used for expressing the user data of the time base.

Struct Element Name	C-Type	Description	Value Range
userDataLength	uint8	User Data Length in bytes	`0..3`
userByte0	uint8	User Byte 0	`0..2^8-1`
userByte1	uint8	User Byte 1	`0..2^8-1`
userByte2	uint8	User Byte 2	`0..2^8-1`

Table 5-6 StbM_UserDataType

[StbM_MeasurementType]

This structure contains additional measurement data.

Struct Element Name	C-Type	Description	Value Range
pathDelay	uint32	Propagation delay in nanoseconds	`0..999999999`

Table 5-7 StbM_MeasurementType

[StbM_SyncRecordTableHeadType]

This structure contains the information of the record table header of the Synchronized Time Base.

Struct Element Name	C-Type	Description	Value Range
SynchronizedTimeDo main	uint8	Time Domain	`0..15`
HWfrequency	uint32	HW Frequency in Hz	`0..2^32-1`
HWprescaler	uint32	Prescaler value	`0..2^32-1`

Table 5-8 StbM_SyncRecordTableHeadType

[StbM_SyncRecordTableBlockType]

This structure contains the information of the record table block of the Synchronized Time Base.

Struct Element Name	C-Type	Description	Value Range
GlbSeconds	uint32	Seconds of the Local Time Base directly after synchronization with the Global Time Base	`0..2^32-1`
GlbNanoSeconds	uint32	Nanoseconds of the Local Time Base directly after synchronization with the Global Time Base	`0..999999999`
TimeBaseStatus	StbM_TimeBaseStatusType	Time Base Status of the Local Time Base directly after synchronization with the Global Time Base	`0..2^4-1`
HWcounter	uint32	HW counter reference value directly after synchronization with the Global Time Base	`0..2^32-1`
RateDeviation	sint16	Calculated Rate Deviation directly after rate deviation measurement	`-32000..32000`
LocSeconds	uint32	Seconds of the Local Time Base directly before synchronization with the Global Time Base	`0..2^32-1`
LocNanoSeconds	uint32	Nanoseconds of the Local Time Base directly before synchronization with the Global Time Base	`0..999999999`
PathDelay	uint32	Current propagation delay in nanoseconds	`0..999999999`

Table 5-9 StbM_SyncRecordTableBlockType

[StbM_OffsetRecordTableHeadType]

This structure contains the information of the record table header of the Offset Time Base.

Struct Element Name	C-Type	Description	Value Range
OffsetTimeDomain	uint8	Time Domain	`16..31`

Table 5-10 StbM_OffsetRecordTableHeadType

[StbM_OffsetRecordTableBlockType]

This structure contains the information of the record table block of the Offset Time Base.

Struct Element Name	C-Type	Description	Value Range
GlbSeconds	uint32	Seconds of the Offset Time Base	`0..2^32-1`
GlbNanoSeconds	uint32	Nanoseconds of the Offset Time Base	`0..999999999`
TimeBaseStatus	StbM_TimeBaseStatusType	Time Base Status of the Local Time Base directly after synchronization	`0..2^4-1`

Struct Element Name	C-Type	Description	Value Range
		with the Global Time Base	

Table 5-11 StbM_OffsetRecordTableBlockType

[StbM_TimeBaseNotificationType]

This 32 Bit bitfield defines a number of global time related events. This Type is used for storing the events in the status variable NotificationEvents and for setting the mask variable NotificationMask which defines a subset of events for which an interrupt request shall be raised.

Status Event Name	Kind	Mask	Status Event Set Condition
EV_GLOBAL_TIME_BASE	bit	0x01U	1: GLOBAL_TIME_BASE bit has changed from 0 to 1 0: otherwise
EV_TIMEOUT_OCCURED	bit	0x02U	1: TIMEOUT bit has changed from 0 to 1 0: otherwise
EV_TIMEOUT_REMOVED	bit	0x04U	1: TIMEOUT bit has changed from 1 to 0 0: otherwise
EV_TIMELEAP_FUTURE	bit	0x08U	1: TIMELEAP_FUTURE bit has changed from 0 to 1 0: otherwise
EV_TIMELEAP_FUTURE_REMOVED	bit	0x10U	1: TIMELEAP_FUTURE bit has changed from 1 to 0 0: otherwise
EV_TIMELEAP_PAST	bit	0x20U	1: TIMELEAP_PAST bit has changed from 0 to 1 0: otherwise
EV_TIMELEAP_PAST_REMOVED	bit	0x40U	1: TIMELEAP_PAST bit has changed from 1 to 0 0: otherwise
EV_SYNC_TO_SUBDOMAIN	bit	0x80U	1: SYNC_TO_GATEWAY bit has changed from 0 to 1 0: otherwise
EV_SYNC_TO_GLOBAL_MASTER	bit	0x100U	1: SYNC_TO_GATEWAY bit has changed from 1 to 0 0: otherwise
EV_RESYNC	bit	0x200U	1: resynchronization has occurred and a new time value has been applied 0: otherwise
EV_RATECORRECTION	bit	0x400U	1: a valid rate correction has been calculated (not beyond limits)

			0: otherwise

Table 5-12 StbM_TimeBaseNotificationType

5.2 Services provided by StbM

5.2.1 StbM_GetVersionInfo

Prototype	
`void StbM_GetVersionInfo ( Std_VersionInfoType *versioninfo )`	
Parameter	
`versioninfo`	Pointer to the memory location holding the version information of the StbM.
Return code	
–	-
Functional Description	
This API can be used to get the version information of the StbM.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is reentrant. > This API is only available if enabled by the configuration parameter `StbMVersionInfoApi`.	
Expected Caller Context	
> No restriction	

Table 5-13 StbM_GetVersionInfo

5.2.2 StbM_InitMemory

Prototype	
`void` **`StbM_InitMemory`** `( void )`	
Parameter	
–	-
Return code	
–	-
Functional Description	
Initializes the global variables in case an initializing startup code is not used. This function sets the StbM into an uninitialized state.	
Particularities and Limitations	
> This function is synchronous. > This function is non-reentrant. > If this function is used it shall be called before any other StbM function after startup.	
Expected Caller Context	
> Task context	

Table 5-14 StbM_InitMemory

5.2.3 StbM_Init

Prototype	
`void` **`StbM_Init`** `( const StbM_ConfigType *ConfigPtr )`	
Parameter	
`ConfigPtr`	Pointer to the selected configuration set.
Return code	
–	-
Functional Description	
This API initializes the StbM.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant. > This API should be called by the ECU State Manger during the startup phase. > This function has to be called before any other StbM service function is called (except StbM_InitMemory()).	
Expected Caller Context	
> Task context	

Table 5-15 StbM_Init

5.2.4 StbM_GetCurrentTime

Prototype	
`Std_ReturnType` **`StbM_GetCurrentTime`** `( StbM_SynchronizedTimeBaseType timeBaseId, StbM_TimeStampType *timeStampPtr, StbM_UserDataType *userDataPtr )`	
Parameter	
`timeBaseId`	The synchronized time-base, whose time is of interest.
`timeStampPtr`	Current time stamp that is valid at this time.
`userDataPtr`	User data of the time base.
Return code	
`Std_ReturnType`	E_OK: The time stamp of the time-base has been updated. E_NOT_OK: A DET error occurred or the EthIf is not available and the time stamp has not been updated.
Functional Description	
This API can be used to get the current time value of the submitted time-base in standard format.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant.	
Expected Caller Context	
> No restriction	

Table 5-16 StbM_GetCurrentTime

Caution
It is the responsibility of the user to make sure that the StbM_GetCurrentTime() API is called inside an exclusive area together with the code which processes the new time value. Otherwise an interrupt or task can postpone further processing and the read value might be outdated depending on the length of the interruption.

If the time base uses an OS counter as local time clock an implementation mechanism that does not disable interrupts has to be used for the exclusive area.

5.2.5 StbM_GetCurrentTimeExtended

Prototype	
`Std_ReturnType` **`StbM_GetCurrentTimeExtended`** `( StbM_SynchronizedTimeBaseType timeBaseId, StbM_TimeStampExtendedType *timeStampPtr, StbM_UserDataType *userDataPtr )`	
Parameter	
`timeBaseId`	The synchronized time-base, whose time is of interest.
`timeStampPtr`	Current time stamp that is valid at this time.
`userDataPtr`	User data of the time base.
Return code	
`Std_ReturnType`	E_OK: The time stamp of the time-base has been updated. E_NOT_OK: A DET error occurred or the EthIf is not available and the time stamp has not been updated.
Functional Description	
This API can be used to get the current time value of the submitted time-base in extended format.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant. > This API is only available if enabled by the configuration parameter `StbMGetCurrentTimeExtendedAvailable.`	
Expected Caller Context	
> No restriction	

Table 5-17 StbM_GetCurrentTimeExtended

Caution

It is the responsibility of the user to make sure that the StbM_GetCurrentTimeExtended() API is called inside an exclusive area together with the code which processes the new time value. Otherwise an interrupt or task can postpone further processing and the read value might be outdated depending on the length of the interruption.

If the time base uses an OS counter as local time clock an implementation mechanism that does not disable interrupts has to be used for the exclusive area.

5.2.6 StbM_GetCurrentTimeRaw

Prototype	
`Std_ReturnType` **`StbM_GetCurrentTimeRaw`** `( StbM_SynchronizedTimeBaseType timeBaseId, StbM_TimeStampRawType *timeStampRawPtr )`	
Parameter	
`timeBaseId`	The synchronized time-base, whose raw time is of interest.
`timeStampRawPtr`	Current time stamp that is valid at this time.
Return code	
`Std_ReturnType`	E_OK: The time stamp has been updated. E_NOT_OK: A DET error occurred or the EthIf is not available and the time stamp has not been updated.
Functional Description	
This API can be used to get a time value in raw format from the submitted time-base.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant.	
Expected Caller Context	
> No restriction	

Table 5-18 StbM_GetCurrentTimeRaw

5.2.7 StbM_GetCurrentTimeDiff

Prototype	
Std_ReturnType **StbM_GetCurrentTimeDiff** (StbM_SynchronizedTimeBaseType timeBaseId, StbM_TimeStampRawType givenTimeStamp, StbM_TimeStampRawType *timeStampDiffPtr)	
Parameter	
timeBaseId	The synchronized time-base, whose time difference is of interest.
givenTimeStamp	Given time stamp as difference calculation basis.
timeStampDiffPtr	Time difference of current time stamp that is valid at this time minus given time stamp.
Return code	
Std_ReturnType	E_OK: The time difference value has been updated. E_NOT_OK: A DET error occurred or the EthIf is not available and the time difference value has not been updated.
Functional Description	
This API can be used to get the time difference of current time raw of the submitted time-base minus given time raw.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant.	
Expected Caller Context	
> No restriction	

Table 5-19 StbM_GetCurrentTimeDiff

5.2.8 StbM_SetGlobalTime

Prototype	
`Std_ReturnType` **`StbM_SetGlobalTime`** `( StbM_SynchronizedTimeBaseType timeBaseId, StbM_TimeStampType *timeStampPtr, StbM_UserDataType *userDataPtr )`	
Parameter	
`timeBaseId`	The synchronized time-base, whose time is set.
`timeStampPtr`	New time stamp.
`userDataPtr`	New user data.
Return code	
`Std_ReturnType`	E_OK: The time stamp and user data of the time-base have been updated. E_NOT_OK: A DET error occurred or the EthIf is not available and the time stamp and user data have not been updated.
Functional Description	
This API allows the customers to set the new global time that has to be valid for the system.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant.	
Expected Caller Context	
> No restriction	

Table 5-20 StbM_SetGlobalTime

5.2.9 StbM_SetUserData

Prototype	
`Std_ReturnType` **`StbM_SetUserData`** `( StbM_SynchronizedTimeBaseType timeBaseId, StbM_UserDataType *userDataPtr )`	
Parameter	
`timeBaseId`	The synchronized time-base, whose user data is set.
`userDataPtr`	New user data.
Return code	
`Std_ReturnType`	E_OK: The user data of the time-base has been updated. E_NOT_OK: A DET error occurred and the user data has not been updated.
Functional Description	
This API allows the customers to set the new user data that has to be valid for the system.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant.	
Expected Caller Context	
> No restriction	

Table 5-21 StbM_SetUserData

5.2.10 StbM_SetOffset

Prototype	
`Std_ReturnType` **`StbM_SetOffset`** `( StbM_SynchronizedTimeBaseType timeBaseId, StbM_TimeStampType *timeStampPtr )`	
Parameter	
`timeBaseId`	The offset time-base, whose offset time is set.
`timeStampPtr`	New offset time stamp.
Return code	
`Std_ReturnType`	E_OK: The offset time stamp of the time-base has been updated. E_NOT_OK: A DET error occurred or the EthIf is not available and the offset time stamp has not been updated.
Functional Description	
This API allows the customers and timebase provider modules to set the offset time that has to be valid for the system.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant.	
Expected Caller Context	
> No restriction	

Table 5-22 StbM_SetOffset

5.2.11 StbM_GetOffset

Prototype	
Std_ReturnType **StbM_GetOffset** (StbM_SynchronizedTimeBaseType timeBaseId, StbM_TimeStampType *timeStampPtr)	
Parameter	
timeBaseId	The offset time-base, whose offset time is of interest.
timeStampPtr	Current offset time stamp.
Return code	
Std_ReturnType	E_OK: The offset time stamp has been updated. E_NOT_OK: A DET error occurred and the offset time stamp has not been updated.
Functional Description	
This API allows the timebase provider modules to get the current offset time.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant.	
Expected Caller Context	
> No restriction	

Table 5-23 StbM_GetOffset

5.2.12 StbM_BusSetGlobalTime

Prototype	
Std_ReturnType **StbM_BusSetGlobalTime** (StbM_SynchronizedTimeBaseType timeBaseId, StbM_TimeStampType *timeStampPtr, StbM_UserDataType *userDataPtr, StbM_MeasurementType *measureDataPtr)	
Parameter	
timeBaseId	The synchronized time-base, whose time is set.
timeStampPtr	New time stamp.
userDataPtr	New user data.
measureDataPtr	New measurement data.
Return code	
Std_ReturnType	E_OK: The time stamp and user data of the time-base have been updated. E_NOT_OK: A DET error occurred or the EthIf is not available and the time stamp and user data have not been updated.
Functional Description	
This API allows the timebase provider modules to forward a new Global Time to the StbM, which has been received from different busses.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant.	
Expected Caller Context	
> No restriction	

Table 5-24 StbM_BusSetGlobalTime

5.2.13 StbM_GetRateDeviation

<table>
<tr><th colspan="2">Prototype</th></tr>
<tr><td colspan="2"><code>Std_ReturnType <b>StbM_GetRateDeviation</b> (StbM_SynchronizedTimeBaseType timeBaseId, StbM_RateDevationType *rateDeviation)</code></td></tr>
<tr><th colspan="2">Parameter</th></tr>
<tr><td><code>timeBaseId</code></td><td>The time-base, whose rate deviation is of interest.</td></tr>
<tr><td><code>rateDevation</code></td><td>Value of the current rate deviation of a time base.</td></tr>
<tr><th colspan="2">Return code</th></tr>
<tr><td><code>Std_ReturnType</code></td><td>E_OK: The rate deviation has been updated.
E_NOT_OK: A DET error occurred and the rate deviation has not been updated.</td></tr>
<tr><th colspan="2">Functional Description</th></tr>
<tr><td colspan="2">This API returns the value of the current rate deviation of a time base.</td></tr>
<tr><th colspan="2">Particularities and Limitations</th></tr>
<tr><td colspan="2">&gt; Service ID: see table 'Service IDs'
> This function is synchronous.
> This function is reentrant.</td></tr>
<tr><th colspan="2">Expected Caller Context</th></tr>
<tr><td colspan="2">&gt; No restriction</td></tr>
</table>

Table 5-25 StbM_GetRateDeviation

5.2.14 StbM_SetRateCorrection

Prototype	
`Std_ReturnType` **`StbM_SetRateCorrection`** `( StbM_SynchronizedTimeBaseType timeBaseId, StbM_RateDevationType rateDeviation )`	
Parameter	
`timeBaseId`	The time-base, whose rate deviation is set.
`rateDevation`	Value of the applied rate deviation.
Return code	
`Std_ReturnType`	E_OK: The rate correction of the time-base has been updated. E_NOT_OK: A DET error occurred or the EthIf is not available and the rate correction has not been updated.
Functional Description	
This API allows to set the rate of a synchronized time base (being either a pure local time base or not).	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is reentrant.	
Expected Caller Context	
> No restriction	

Table 5-26 StbM_SetRateCorrection

5.2.15 StbM_GetSyncTimeRecordHead

Prototype	
`Std_ReturnType` **`StbM_GetSyncTimeRecordHead`** `( StbM_SynchronizedTimeBaseType timeBaseId, StbM_SyncRecordTableHeadType *syncRecordTableHead )`	
Parameter	
`timeBaseId`	The time-base, whose header is of interest.
`syncRecordTableHead`	Header of the recorded snapshot data.
Return code	
`Std_ReturnType`	E_OK: The record table header has been updated. E_NOT_OK: A DET error occurred and the record table header has not been updated.
Functional Description	
This API allows the customers to access the recorded snapshot data header of the table belonging to the Synchronized Time Base.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is reentrant.	
Expected Caller Context	
> No restriction	

Table 5-27 StbM_GetSyncTimeRecordHead

5.2.16 StbM_GetOffsetTimeRecordHead

Prototype	
`Std_ReturnType` **`StbM_GetOffsetTimeRecordHead`** `( StbM_SynchronizedTimeBaseType timeBaseId, StbM_OffsetRecordTableHeadType *offsetRecordTableHead )`	
Parameter	
`timeBaseId`	The time-base, whose header is of interest.
`offsetRecordTableHead`	Header of the recorded snapshot data.
Return code	
`Std_ReturnType`	E_OK: The record table header has been updated. E_NOT_OK: A DET error occurred and the record table header has not been updated.
Functional Description	
This API allows the customers to access the recorded snapshot data header of the table belonging to the Offset Time Base.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is reentrant.	
Expected Caller Context	
> No restriction	

Table 5-28 StbM_GetOffsetTimeRecordHead

5.2.17 StbM_StartTimer

Prototype	
`Std_ReturnType` **`StbM_StartTimer`** `( StbM_SynchronizedTimeBaseType timeBaseId, StbM_CustomerIdType customerId, StbM_TimeStampType expireTime )`	
Parameter	
`timeBaseId`	ID of the Time Base, relative to which the timer shall be started
`customerId`	ID of the notification customer
`expireTime`	Time value relative to current Time Base value of the Notification Customer, when the Timer shall expire
Return code	
`Std_ReturnType`	E_OK: Starting the timer was successful. E_NOT_OK: Starting timer was not successful.
Functional Description	
This API sets a time value which the Time Base value is compared against.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reenttrant.	
Expected Caller Context	
> No restriction	

Table 5-29 StbM_StartTimer

5.2.18 StbM_NotificationFunction

Prototype	
`void` **`StbM_NotificationFunction`** `( void )`	
Parameter	
–	-
Return code	
–	-
Functional Description	
This API calls the callback functions for time notification customers. This function only works with MICROSAR RTE.	
Particularities and Limitations	
> This function is synchronous. > This function is non-reentrant.	
Expected Caller Context	
> Task context	

Table 5-30 StbM_NotificationFunction

5.2.19 StbM_UpdateGlobalTime

Prototype	
`Std_ReturnType` **`StbM_UpdateGlobalTime`** `( StbM_SynchronizedTimeBaseType timeBaseId, const StbM_TimeStampType *timeStamp, const StbM_UserDataType *userData )`	
Parameter	
`timeBaseId`	The synchronized time-base, whose time is updated.
`timeStamp`	New time stamp.
`userData`	New user data.
Return code	
`Std_ReturnType`	E_OK: The time stamp and user data of the time-base have been updated. E_NOT_OK: A DET error occurred or the EthIf is not available and the time stamp and user data have not been updated.
Functional Description	
This API allows the customers to set the new global time that has to be valid for the system. Using UpdateGlobalTime will not lead to an immediate transmission of the global time.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant.	
Expected Caller Context	
> No restriction	

Table 5-31 StbM_UpdateGlobalTime

5.2.20 StbM_TriggerTimeTransmission

Prototype	
`Std_ReturnType` **`StbM_TriggerTimeTransmission`** `( StbM_SynchronizedTimeBaseType timeBaseId )`	
Parameter	
`timeBaseId`	The synchronized time-base, whose immediate transmission shall be triggered.
Return code	
`Std_ReturnType`	E_OK: The immediate transmission of the time-base has been triggered. E_NOT_OK: A DET error occurred and the immediate time transmission has not been triggered.
Functional Description	
This API allows the customers to force the Timesync modules to transmit the current time base again.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant.	
Expected Caller Context	
> No restriction	

Table 5-32 StbM_TriggerTimeTransmission

5.2.21 StbM_GetTimeBaseUpdateCounter

Prototype	
`uint8` **`StbM_GetTimeBaseUpdateCounter`** `( StbM_SynchronizedTimeBaseType timeBaseId )`	
Parameter	
`timeBaseId`	The synchronized time-base, whose update counter is of interest.
Return code	
`uint8`	Current counter value of the time base.
Functional Description	
This API allows the Timesync modules to detect, whether a time base should be transmitted immediately in the subsequent <Bus>TSyn_MainFunction() cycle.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant.	
Expected Caller Context	
> No restriction	

Table 5-33 StbM_GetTimeBaseUpdateCounter

5.2.22 StbM_GetTimeLeap

Prototype	
Std_ReturnType **StbM_GetTimeLeap** (StbM_SynchronizedTimeBaseType timeBaseId, StbM_TimeDiffType *timeJump)	
Parameter	
timeBaseId	The time-base, whose time leap is of interest.
timeJump	Value of the last time leap of a time base.
Return code	
Std_ReturnType	E_OK: Time leap is valid. E_NOT_OK: A DET error occurred or no time leap occured or time leap is out of range.
Functional Description	
This API returns the value of the last time leap, if StbMTimeLeapFuture/PastThreshold is exceeded.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is reentrant.	
Expected Caller Context	
> No restriction	

Table 5-34 StbM_ GetTimeLeap

5.2.23 StbM_GetTimeBaseStatus

Prototype	
Std_ReturnType **StbM_GetTimeBaseStatus**(StbM_SynchronizedTimeBaseType timeBaseId, StbM_TimeBaseStatusType *syncTimeBaseStatus, StbM_TimeBaseStatusType *offsetTimeBaseStatus)	
Parameter	
timeBaseId	The time-base, whose status is of interest.
syncTimeBaseStatus	Status of the Synchronized Time Base.
offsetTimeBaseStatus	Status of the Offset Time Base.
Return code	
Std_ReturnType	E_OK: Status is valid. E_NOT_OK: A DET error occurred or no status could be retrieved.
Functional Description	
This API returns the status of a Time Base. For Offset Time Bases the status of the underlying Synchronized Time Base is also returned.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is reentrant.	
Expected Caller Context	
> No restriction	

Table 5-35 StbM_ GetTimeBaseStatus

5.2.24 StbM_MainFunction

Prototype	
`void` **`StbM_MainFunction`** `( void )`	
Parameter	
–	-
Return code	
–	-
Functional Description	
This function will be called cyclically by a task body provided by the BSW Scheduler. It will invoke the triggered customers and synchronize the referenced OS ScheduleTables.	
Particularities and Limitations	
> Service ID: see table 'Service IDs' > This function is synchronous. > This function is non-reentrant.	
Expected Caller Context	
> Task context	

Table 5-36 StbM_MainFunction

5.3 Services used by StbM

In the following table services provided by other components, which are used by the StbM are listed. For details about prototype and functionality refer to the documentation of the providing component.

Component	API
Det	Det_ReportError
Os	GetCounterValue GetElapsedValue GetScheduleTableStatus SyncScheduleTable SetEvent ActivateTask
EthIf	EthIf_GetCurrentTime EthIf_GetControllerMode
RTE / SchM	SchM_Enter_StbM_STBM_EXCLUSIVE_AREA_0 SchM_Exit_StbM_STBM_EXCLUSIVE_AREA_0
Gpt	Gpt_StartTimer Gpt_GetTimeElapsed Gpt_EnableNotification

Table 5-37 Services used by the StbM

5.4 Configurable Interfaces

5.4.1 Notifications

At its configurable interfaces the StbM defines notifications that can be mapped to callback functions provided by other modules. The mapping is not statically defined by the StbM but can be performed at configuration time. The function prototypes that can be used for the configuration have to match the appropriate function prototype signatures, which are described in the following sub-chapters.

5.4.1.1 SyncTimeRecordBlockCallback

Prototype	

`Std_ReturnType` **`SyncTimeRecordBlockCallback<TimeBase>`** `(`
`StbM_SyncRecordTableBlockType *syncRecordTableBlock )`	

Parameter	
`syncRecordTableBlock`	Block of the table.
Return code	
`Std_ReturnType`	E_OK: Table access done. E_NOT_OK: Table contains no data or access invalid.
Functional Description	
This function provides a recorded snapshot data block of the measurement data table belonging to the Synchronized Time Base.	
Particularities and Limitations	
> This function is synchronous. > This function is non-reentrant.	
Call context	
> Task context	

Table 5-38 SyncTimeRecordBlockCallback

5.4.1.2 OffsetTimeRecordBlockCallback

Prototype	

`Std_ReturnType` **`OffsetTimeRecordBlockCallback<TimeBase>`** `(`
`StbM_OffsetRecordTableBlockType *offsetRecordTableBlock )`	

Parameter	
`offsetRecordTableBlock`	Block of the table.
Return code	
`Std_ReturnType`	E_OK: Table access done. E_NOT_OK: Table contains no data or access invalid.
Functional Description	
This function provides a recorded snapshot data block of the measurement data table belonging to the Offset Time Base.	
Particularities and Limitations	
> This function is synchronous. > This function is non-reentrant.	
Call context	
> Task context	

Table 5-39 OffsetTimeRecordBlockCallback

5.4.1.3 StatusNotificationCallback

Prototype	

`Std_ReturnType` **`StatusNotificationCallback<TimeBase>`**
`(StbM_TimeBaseNotificationType eventNotification )`	

Parameter	
`eventNotification`	Holds the notification bits for the different Time Base related events
Return code	
`Std_ReturnType`	E_OK: successful. E_NOT_OK: failed.
Functional Description	
This callback notifies the <Customer>, when a Time Base reaches the time value set by StbM_SetTimer for the <TimeBase>	
Particularities and Limitations	
> This function is synchronous. > This function is non-reentrant.	
Call context	
> Task context	

Table 5-40 StatusNotificationCallback

5.4.1.4 <Customer>_TimeNotificationCallback

Prototype	
Std_ReturnType **<Customer>_TimeNotificationCallback<TimeBase>** (StbM_TimeDiffType deviationtime)	
Parameter	
deviationTime	Deviation between actual time value captured when callback is called and expiration time.
Return code	
Std_ReturnType	E_OK: successful. E_NOT_OK: failed.
Functional Description	
The callback notifies the customers, when a <TimeBase> related event occurs, which is enabled by the notification mask	
Particularities and Limitations	
> This function is synchronous. > This function is non-reentrant.	
Call context	
> Task context	

Table 5-41 <Customer>_TimeNotificationCallback

5.5 Service Ports

The service port names use placeholders. Their meaning is described in the following table.

Placeholder	Meaning
TB	Name of the time base
C	Name of the time notification customer

Table 5-42 Meaning of placeholders in service port names

5.5.1 Client Server Interface

A client server interface is related to a Provide Port at the server side and a Require Port at client side.

5.5.1.1 Provide Ports on StbM Side

At the Provide Ports of the StbM the API functions described in 5.2 that are invoked via Operations are available as Runnable Entities. The mapping from a SWC client call to an Operation is performed by the RTE. In this mapping the RTE adds Port Defined Argument Values to the client call of the SWC, if configured.

The following sub-chapters present the Provide Ports defined for the StbM and the Operations defined for the Provide Ports, the API functions related to the Operations and the Port Defined Argument Values to be added by the RTE.

5.5.1.1.1 GlobalTime_Master_<TB>

This provide port is only available, if the configuration parameter `StbMIsSystemWideGlobalTimeMaster` or `StbMAllowSystemWideGlobalTimeMaster` is enabled for the appropriate time base and the `StbMSynchronizedTimeBaseIdentifier` of the time base is less than 128.

The operation SetOffset is only available, if the time base is an offset time base, i.e. it has an `StbMSynchronizedTimeBaseIdentifier` in the range 16 till 31.

The operation TriggerTimeTransmission is only available, if the `StbMSynchronizedTimeBaseIdentifier` of the time base is less than 32.

Operation	API Function	Port Defined Argument Values
SetGlobalTime	StbM_SetGlobalTime	StbM_SynchronizedTimeBaseType 0..n
SetOffset	StbM_SetOffset	StbM_SynchronizedTimeBaseType 0..n
SetUserData	StbM_SetUserData	StbM_SynchronizedTimeBaseType 0..n
SetRateCorrection	StbM_SetRateCorrection	StbM_SynchronizedTimeBaseType 0..n
TriggerTimeTransmission	StbM_TriggerTimeTransmission	StbM_SynchronizedTimeBaseType 0..n
UpdateGlobalTime	StbM_UpdateGlobalTime	StbM_SynchronizedTimeBaseType 0..n

Table 5-43 GlobalTime_Master_<TB>

5.5.1.1.2 GlobalTime_Slave_<TB>

This provide port is only available, if the time base has an `StbMSynchronizedTimeBaseIdentifier` less than 128.

The operation GetCurrentTimeExtended is only available, if the parameter `StbMGetCurrentTimeExtendedAvailable` is enabled.

The operation GetSyncTimeRecordHead is only available, if the parameter `StbMTimeRecordingSupport` is enabled and the time base is a synchronized time base, i.e. it has an `StbMSynchronizedTimeBaseIdentifier` less than 16.

The operation GetOffsetTimeRecordHead is only available, if the parameter `StbMTimeRecordingSupport` is enabled and the time base is an offset time base, i.e. it has an `StbMSynchronizedTimeBaseIdentifier` in the range 16 till 31.

The operation GetTimeLeap is only available, if the time base has an `StbMSynchronizedTimeBaseIdentifier` less than 32.

Operation	API Function	Port Defined Argument Values
GetCurrentTime	StbM_GetCurrentTime	StbM_SynchronizedTimeBaseType 0..n
GetCurrentTimeExtended	StbM_GetCurrentTimeExtended	StbM_SynchronizedTimeBaseType 0..n
GetRateDeviation	StbM_GetRateDeviation	StbM_SynchronizedTimeBaseType 0..n

Operation	API Function	Port Defined Argument Values
GetSyncTimeRecordHead	StbM_GetSyncTimeRecordHead	StbM_SynchronizedTimeBaseType 0..n
GetOffsetTimeRecordHead	StbM_GetOffsetTimeRecordHead	StbM_SynchronizedTimeBaseType 0..n
GetTimeLeap	StbM_GetTimeLeap	StbM_SynchronizedTimeBaseType 0..n
GetTimeBaseStatus	StbM_GetTimeBaseStatus	StbM_SynchronizedTimeBaseType 0..n

Table 5-44 GlobalTime_Slave_<TB>

5.5.1.1.3 StartTimer_<TB>_<C>

This provide port is only available for each notification customer of the time base, if the `StbMSynchronizedTimeBaseIdentifier` of the time base is less than 128.

Operation	API Function	Port Defined Argument Values
StartTimer	StbM_StartTimer	▶ StbM_SynchronizedTimeBaseType 0..n ▶ StbM_CustomerIdType 0..n

Table 5-45 StartTimer_<TB>_<C>

5.5.1.2 Require Ports on StbM Side

At its Require Ports the StbM calls Operations. These Operations have to be provided by the SWCs by means of Runnable Entities. These Runnable Entities implement the callback functions expected by the StbM.

The following sub-chapters present the Require Ports defined for the StbM, the Operations that are called from the StbM and the related Notifications, which are described in chapter 5.4.

5.5.1.2.1 StbM_MeasurementNotification_<TB>

This required port is only available, if the parameter `StbMTimeRecordingSupport` is enabled and the `StbMSynchronizedTimeBaseIdentifier` of the time base is less than 32.

The operation SetSyncTimeRecordTable is only available, if the time base is a synchronized time base, i.e. it has an `StbMSynchronizedTimeBaseIdentifier` less than 16.

The operation SetOffsetTimeRecordTable is only available, if the time base is an offset time base, i.e. it has an `StbMSynchronizedTimeBaseIdentifier` in the range 16 till 31.

Operation	Notification
SetSyncTimeRecordTable	SyncTimeRecordBlockCallback
SetOffsetTimeRecordTable	OffsetTimeRecordBlockCallback

Table 5-46 StbM_MeasurementNotification_<TB>

5.5.1.2.2 GlobalTime_TimeEvent_<TB>_<C>

This required port is only available for each notification customer of the time base, if the `StbMSynchronizedTimeBaseIdentifier` of the time base is less than 128.

Operation	Notification
NotifyTime	TimeNotificationCallback

Table 5-47 GlobalTime_TimeEvent_<TB>_<C>

5.5.2 Sender-Receiver Interface

The Sender-Receiver interfaces and ports described here are used to generate the RTE between application software components and the StbM.

5.5.2.1 Provided Ports on StbM side

5.5.2.1.1 StatusNotification

The StbM is able to send status change events via Provided Sender-Receiver Ports. This provide port is only available, if the time base has an `StbMSynchronizedTimeBaseIdentifier` less than 128 and a status notification mask configured.

The Sender-Receiver-Interface is named as

- StatusNotification

The related Provided Ports are named as

- GlobalTime_StatusEvent_<TB>

Further, the belonging data element is specified as

- eventNotification

6 Configuration

In the StbM the attributes can be configured with the following tools:

- Configuration in DaVinci Configurator

6.1 Configuration Variants

The StbM supports the configuration variants

- `VARIANT-PRE-COMPILE`

The configuration classes of the StbM parameters depend on the supported configuration variants. For their definitions please see the StbM_bswmd.arxml file.

7 Glossary and Abbreviations

7.1 Glossary

Term	Description
DaVinci Configurator	Configuration and generation tool for MICROSAR components

Table 7-1 Glossary

7.2 Abbreviations

Abbreviation	Description
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
BSW	Basis Software
DET	Development Error Tracer
ECU	Electronic Control Unit
EthIf	Ethernet Interface
EthTsyn	Time Synchronization over Ethernet
Gpt	General Purpose Timer
HIS	Hersteller Initiative Software
MICROSAR	Microcontroller Open System Architecture (the Vector AUTOSAR solution)
OS	Operating System
RTE	Runtime Environment
SchM	Schedule Manager
SRS	Software Requirement Specification
StbM	Synchronized Time-Base Manager
SWC	Software Component
SWS	Software Specification

Table 7-2 Abbreviations

8 Contact

Visit our website for more information on

- News
- Products
- Demo software
- Support
- Training data
- Addresses

www.vector.com