Document Title	Specification of Ethernet State Manager
Document Category	Software specification (SWS)
Document UID	415
Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R26-11

Document Change History			
Date	**Release**	**Changed by**	**Description**
2026-11-26	R26-11	AUTOSAR Release Management	• COMM_SILENT_COMMUNICATION removed • GetVersionInfo() removed
2025-11-27	R25-11	AUTOSAR Release Management	• Added multiplicity for certain containers
2024-11-27	R24-11	AUTOSAR Release Management	• EthSM state diagram changes in regard to Silent Communication
2023-11-23	R23-11	AUTOSAR Release Management	• Add new Paramter EthSMTcpIpUsed to enable/disable interaction with a TcpIp module for the EthSm Network • Removed unnecessary requirements: [SWS_EthSM_00008], [SWS_EthSM_00010], [SWS_EthSM_00013]
2022-11-24	R22-11	AUTOSAR Release Management	• Update state machine behaviour • Removal of obsolete items • Set items to valid: [SWS_EthSM_00216], [SWS_EthSM_00217], [SWS_EthSM_00220], [ECUC_EthSM_00110]
2021-11-25	R21-11	AUTOSAR Release Management	• Update state machine behaviour in 'ETHSM_STATE_ONLINE' and 'ETHSM_STATE_WAIT_OFFLINE' • Editorial changes

▽

△

2020-11-30	R20-11	AUTOSAR Release Management	• Reworked the statemachine specification of EthSM • Added handling for request of 'COMM_FULL_COMMUNICATION _WITH_WAKEUP_REQUEST' • Cleaned up the document structure
2019-11-28	R19-11	AUTOSAR Release Management	• No content changes • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	• Error classification has been fixed • Editorial changes
2017-12-08	4.3.1	AUTOSAR Release Management	• Default error is removed • Editorial changes
2016-11-30	4.3.0	AUTOSAR Release Management	• Remove Set and Get Transceiver mode functionality • Correct EthSM_TcpIpModeIndication callback return value • Harmonize main function period with the other modules • Remove Get current internal mode
2015-07-31	4.2.2	AUTOSAR Release Management	• Harmonize Sequence diagrams, Network State Machine and Functional Description • Debugging support marked as obsolete • Report to DET if TcpIp state is not accepted • Adaptations related to renaming of DET, • Error Handling: tables for Runtime Errors and Transient Faults added
2014-10-31	4.2.1	AUTOSAR Release Management	• Change from Synchronous to Asynchronous API • Additional callback functions added • Existing behavior of functions changes • Editorial changes

▽

△

2014-03-31	4.1.3	AUTOSAR Release Management	• Corrective action after timeout • Non mutually exclusive transitions from ETHSM_STATE_ONLINE • Editorial changes
2013-10-31	4.1.2	AUTOSAR Release Management	• Optimization of full com request • Standardization of internal state names • Asynchronous behavior of several interfaces • Several clarifications and corrections • Editorial changes • Removed chapter(s) on change documentation
2013-03-15	4.1.1	AUTOSAR Release Management	• New State Machine (new sub states and new state conditions, new APIs) • Update chapter 10 • Added Production Error if Transceiver Link is down • General Update (corrections and formulations)
2011-12-22	4.0.3	AUTOSAR Release Management	• Update Chapter 10 (Parameter adjustment)
2010-09-30	3.1.5	AUTOSAR Release Management	• Functional changes: • Correction of the naming convention of SW modul version information • Correction of chapter 10 - configuration parameter 'EthSMNetworkIndex' • Remove InstanceID from GetVersionId structure • Additional callback function: Call of SoAd_<Bus>SM_ModeIndication realized after the successful initialization of the EthTrcv and the EthController. • Non functional changes: • Adding a self loop with 'No initialization' in the state diagramm

▽

△

2010-02-02	3.1.4	AUTOSAR Release Management	• Initial Release

Disclaimer

This work (specification and/or software implementation) and the material contained in it, as released by AUTOSAR, is for the purpose of information only. AUTOSAR and the companies that have contributed to it shall not be liable for any use of the work.

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

This work may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only. For any other purpose, no part of the work may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

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

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Table of Contents

1 Introduction and functional overview 9
2 Acronyms and Abbreviations 10
3 Related documentation 11
3.1 Input documents . . . 11
3.2 Related standards and norms . . . 11
3.3 Related specification . . . 11
4 Constraints and assumptions 12
4.1 Limitations . . . 12
4.2 Applicability to car domains . . . 12
5 Dependencies to other modules 13
5.1 File structure . . . 13
5.1.1 Code file structure . . . 13
6 Requirements Tracing 14
7 Functional specification 15
7.1 Translation of network communication mode requests . . . 15
7.2 Output of current network communication modes . . . 15
7.3 Control of peripherals . . . 16
7.3.1 Ethernet Interface Controllers . . . 16
7.4 Multiple networks . . . 16
7.5 Background and Rationale . . . 16
7.6 Network mode state machine . . . 17
7.6.1 Initial transition . . . 21
7.6.2 Behaviour in sub-state ETHSM_STATE_OFFLINE . . . 21
7.6.2.1 Transition from sub-state OFFLINE to WAIT_TRCVLINK . . . 21
7.6.3 Behavior in sub-state ETHSM_STATE_WAIT_TRCVLINK . . . 22
7.6.3.1 Transition from sub-state WAIT_TRCVLINK to OFFLINE . . . 22
7.6.3.2 Transition from sub-state WAIT_TRCVLINK to WAIT_ONLINE . 22
7.6.4 Behaviour in sub-state ETHSM_STATE_WAIT_ONLINE . . . 23
7.6.4.1 Transition from sub-state WAIT_ONLINE to WAIT_TRCVLINK . 23
7.6.4.2 Transition from sub-state WAIT_ONLINE to OFFLINE . . . 23
7.6.4.3 Transition from sub-state WAIT_ONLINE to ONLINE . . . 23
7.6.5 Behaviour in sub-state ETHSM_STATE_ONLINE . . . 24
7.6.5.1 Transition from sub-state ONLINE to WAIT_ONLINE . . . 24
7.6.5.2 Transition from sub-state ONLINE to WAIT_OFFLINE . . . 24
7.6.5.3 Transition from sub-state ONLINE to ONHOLD . . . 25
7.6.6 Behaviour in sub-state ETHSM_STATE_WAIT_OFFLINE . . . 25
7.6.6.1 Transition from sub-state WAIT_OFFLINE to OFFLINE . . . 26
7.6.6.2 Transistion from sub-state WAIT_OFFLINE to ONLINE . . . 26

7.6.7 Behaviour in sub-state ETHSM_STATE_ONHOLD . . . 26
7.6.7.1 Transistion from ONHOLD to ONLINE . . . 27
7.6.7.2 Transition from sub-state ONHOLD to WAIT_TRCVLINK . . . 27
7.6.7.3 Transition from sub-state ONHOLD to OFFLINE . . . 27
7.6.8 Information about state transitions . . . 28
7.7 Commercial Off The Shelf stack usage . . . 28
7.8 Error classification . . . 29
7.8.1 Development Errors . . . 29
7.8.2 Runtime Errors . . . 29
7.8.3 Production Errors . . . 30
8 API specification 31
8.1 Imported types . . . 31
8.2 Type definitions . . . 31
8.2.1 EthSM_NetworkModeStateType . . . 31
8.3 Function definitions . . . 32
8.3.1 Generic functions . . . 32
8.3.1.1 EthSM_Init . . . 32
8.3.1.2 EthSM_RequestComMode . . . 32
8.3.1.3 EthSM_GetCurrentComMode . . . 34
8.3.2 Mode indication functions . . . 35
8.3.2.1 EthSM_CtrlModeIndication . . . 35
8.3.2.2 EthSM_TrcvLinkStateChg . . . 35
8.3.2.3 EthSM_TcpIpModeIndication . . . 36
8.3.2.4 EthSM_SleepIndication . . . 37
8.4 Scheduled functions . . . 38
8.4.1 EthSM_MainFunction . . . 39
8.5 Expected interfaces . . . 39
8.5.1 Mandatory Interfaces . . . 39
8.5.2 Optional Interfaces . . . 40
9 Sequence diagrams 41
10 Configuration specification 44
10.1 How to read this chapter . . . 44
10.2 Containers and configuration parameters . . . 44
10.2.1 Configuration Tool . . . 44
10.2.2 EthSM . . . 44
10.2.3 EthSMGeneral . . . 46
10.2.4 EthSMNetwork . . . 47
10.2.5 EthSMDemEventParameterRefs . . . 49
10.3 Published Information . . . 50
A Not applicable requirements 51

B Change history of AUTOSAR traceable items 52
B.1 Traceable item history of this document according to AUTOSAR Release R24-11 . . . 52
B.1.1 Added Specification Items in R24-11 . . . 52
B.1.2 Changed Specification Items in R24-11 . . . 52
B.1.3 Deleted Specification Items in R24-11 . . . 53
B.2 Traceable item history of this document according to AUTOSAR Release R25-11 . . . 53
B.2.1 Added Specification Items in R25-11 . . . 53
B.2.2 Changed Specification Items in R25-11 . . . 53
B.2.3 Deleted Specification Items in R25-11 . . . 53
B.3 Traceable item history of this document according to AUTOSAR Release R26-11 . . . 53
B.3.1 Added Specification Items in R26-11 . . . 53
B.3.2 Changed Specification Items in R26-11 . . . 54
B.3.3 Deleted Specification Items in R26-11 . . . 54

1 Introduction and functional overview

This specification specifies the functionality, API and the configuration of the AUTOSAR Basic Software module Ethernet State Manager.

In the AUTOSAR Layered Software Architecture, the Ethernet State Manager belongs to the ECU Abstraction Layer, or more precisely, to the Communication Hardware Abstraction.

The main task of the Ethernet State Manager can be summarized as follows:

[SWS_EthSM_00001] ⌈The Ethernet State Manager shall provide an abstract interface to the AUTOSAR Communication Manager to startup or shutdown the communication on an Ethernet cluster.⌋

[SWS_EthSM_00002] ⌈The Ethernet State Manager does not directly access the Ethernet hardware (Ethernet Communication Controller and Ethernet Transceiver), but by means of the Ethernet Interface. The Ethernet Interface redirects the request to the appropriate driver module.⌋

Figure 1.1 is an example of an Autosar architecture including an Ethernet network.

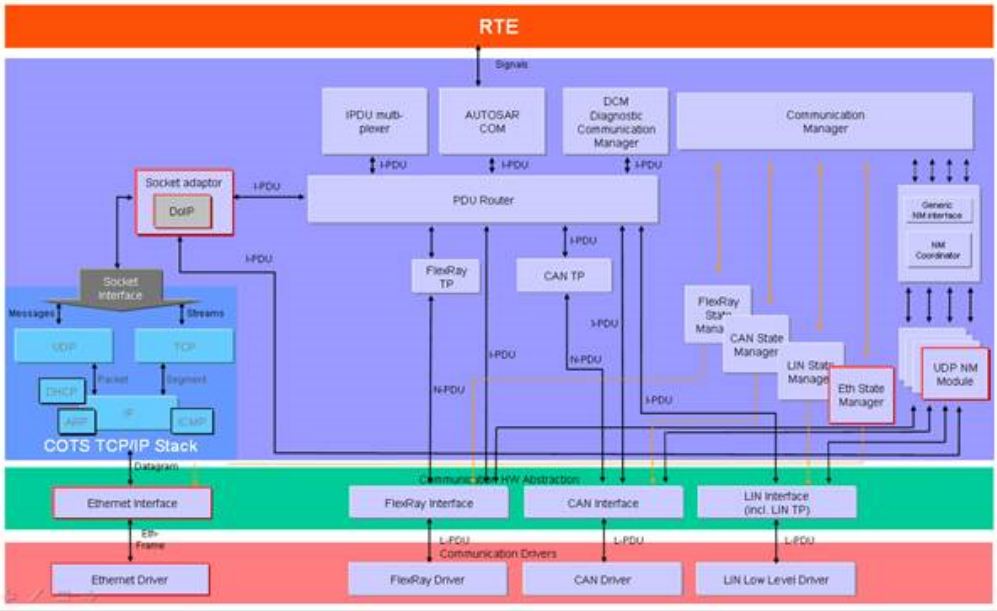

Figure 1.1: Example of an Autosar architecture including an Ethernet network

2 Acronyms and Abbreviations

Abbreviation / Acronym:	Description:
API	Application Program Interface
BSW	Basic Software
BswM	Basic Software Mode Manager
ComM	Communication Manager
DEM	Diagnostic Event Manager
DET	Default Error Tracer
EcuM	ECU State Manager
Eth	Ethernet Controller
EthTrcv	Ethernet Transceiver
EthSM	Ethernet State Manager
EthIf	Ethernet Interface
SchM	BSW Scheduler
SoAd	Socket Adapter
OA TC10	Open Alliance TC10 specification (see [1])

3 Related documentation

3.1 Input documents

[1] OPEN Sleep/Wake-up Specification for Automotive Ethernet
http://www.opensig.org/Automotive-Ethernet-Specifications/

[2] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral

[3] System Template
AUTOSAR_CP_TPS_SystemTemplate

3.2 Related standards and norms

No related standards.

3.3 Related specification

AUTOSAR provides a General Specification on Basic Software modules *SWS BSW General*, [2], which is also valid for Ethernet State Manager.

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

4 Constraints and assumptions

4.1 Limitations

The EthSM can be used for Ethernet communication only. Its dedication is to operate with the EthIf to control one or multiple underlying Ethernet Controllers and Ethernet Transceiver Drivers. Other protocols than Ethernet (i.e. CAN, LIN or FlexRay) are not supported.

The following items are not supported by the current version of this specification:

- Wake on LAN. (Please note, wake-up and sleep on dataline according to the OA TC10 specification (see [1]) is supported)

The actual EthSM requires an IP-based communication stack. To get FULL_COMMUNICATION it is necessary to get an active IP communication. In further specifications, an alternative "low level" state machine will be introduced. This state machine only works on driver/transceiver level (without IP communication). This is necessary to realize other communication protocols (e.g. IEEE 1722).

4.2 Applicability to car domains

The Ethernet State Manager can be used for all domain applications always when the Ethernet protocol is used. The Ethernet BSW Stack can be used wherever high data rates are required.

5 Dependencies to other modules

AUTOSAR BSW Scheduler The BSW Scheduler calls the main functions of the EthSM, which are necessary for the cyclic processes of the EthSM.

AUTOSAR Communication Manager The ComM requests network communication modes and is notified by the EthSM when a communication mode is reached.

AUTOSAR Ethernet Interface The EthSM uses the API of the EthIf to initialize the Ethernet Communication Hardware and to control the operating modes of the Ethernet Controllers and Ethernet Transceivers assigned to the Ethernet Networks.

The Ethernet Interface uses the API of the EthSM to provide the transceiver link state.

AUTOSAR Default Error Tracer In order to be able to report development errors, the Ethernet State Manager has to have access to the error hook of the Default Error Tracer.

AUTOSAR Diagnostic Event Manager In order to be able to report production errors the Ethernet State Manager has to have access to the Diagnostic Event Manager.

ECU State Manager The EcuM initializes the EthSM.

AUTOSAR Bsw Manager The BswM is notified by the EthSM when an internal state is reached.

AUTOSAR TcpIp TcpIp is called to request the TCPIP state (e.g. Online, Offline, On Hold, ...). TcpIP uses the API of the EthSM to provide the TCPIP state.

5.1 File structure

5.1.1 Code file structure

For details refer to [2] Chapter 5.1.6 *“Code file structure”*.

Remark:

Actually the module EthSM doesn’t provide link time configuration and post-build time configuration.

6 Requirements Tracing

The following tables reference the requirements specified in [2] and links to the fulfillment of these. Please note that if column “Satisfied by” is empty for a specific requirement this means that this requirement is not fulfilled by this document.

Requirement	Description	Satisfied by
[SRS_BSW_00003]	All software modules shall provide version and identification information	[SWS_EthSM_00060]
[SRS_BSW_00101]	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	[SWS_EthSM_00043]
[SRS_BSW_00159]	All modules of the AUTOSAR Basic Software shall support a tool based configuration	[SWS_EthSM_00081]
[SRS_BSW_00318]	Each AUTOSAR Basic Software Module file shall provide version numbers in the header file	[SWS_EthSM_00060]
[SRS_BSW_00358]	The return type of init() functions implemented by AUTOSAR Basic Software Modules shall be void	[SWS_EthSM_00043]
[SRS_BSW_00374]	All Basic Software Modules shall provide a readable module vendor identification	[SWS_EthSM_00060]
[SRS_BSW_00405]	BSW Modules shall support multiple configuration sets	[SWS_EthSM_00043]
[SRS_BSW_00406]	API handling in uninitialized state	[SWS_EthSM_00054] [SWS_EthSM_00060] [SWS_EthSM_00115] [SWS_EthSM_00120]
[SRS_BSW_00414]	Init functions shall have a pointer to a configuration structure as single parameter	[SWS_EthSM_00043]
[SRS_BSW_00424]	BSW module main processing functions shall not be allowed to enter a wait state	[SWS_EthSM_00081]
[SRS_BSW_00425]	The BSW module description template shall provide means to model the defined trigger conditions of schedulable objects	[SWS_EthSM_00081]
[SRS_BSW_00458]	Classification of production errors	[SWS_EthSM_00202]
[SRS_Eth_00158]	The Ethernet state manager shall trigger requested modes for Ethernet hardware with wake-up capability even if the requested mode has already been reached.	[SWS_EthSM_00206] [SWS_EthSM_00207] [SWS_EthSM_00212]
[SRS_Eth_00159]	The Ethernet state manager shall forward sleep requests indicated by the maintained Ethernet hardware that act as communication slaves to the responsible upper layer.	[SWS_EthSM_00213] [SWS_EthSM_00214] [SWS_EthSM_00215] [SWS_EthSM_91004]

Table 6.1: Requirements Tracing

7 Functional specification

An ECU can have different communication networks. Each network has to be identified with a unique network handle. The ComM requests communication modes from the networks. It knows by its configuration, which handle is assigned to what kind of network. In case of Ethernet, it uses the Ethernet state manager, which is responsible for the control flow abstraction of Ethernet networks. The following sections describe this in detail.

7.1 Translation of network communication mode requests

[SWS_EthSM_00014] ⌈The EthSM shall provide to the ComM an API, which can be used by the ComM to request communication modes of Ethernet networks.⌋

[SWS_EthSM_00015] ⌈Depending on the parameters handed over by this API, the EthSM shall execute a state transition of the related network mode state machine (refer to section 7.6).⌋

[SWS_EthSM_00016] ⌈This transition shall translate the request into a respective API call to control the assigned Ethernet peripherals.⌋

[SWS_EthSM_00224]

Status: DRAFT

⌈When EthSM transceiver link state gets called `EthSM_TrcvLinkStateChg` and the config parameter `EthSMDataLinkInfoApi` is set to TRUE, then EthSM shall call `ISO15118Chrg_DataLinkIndication`.⌋

7.2 Output of current network communication modes

The current communication mode of a network can be different from the requested mode. The EthSM has to provide the information on the current communication mode to the ComM by the two following kind of interfaces:

[SWS_EthSM_00017] ⌈The EthSM shall provide an API, which can be polled by the ComM to get the current communication mode of an Ethernet network.⌋

[SWS_EthSM_00018] ⌈The EthSM shall use a call-back notification of ComM to notify ComM of a change in communication modes.⌋

7.3 Control of peripherals

7.3.1 Ethernet Interface Controllers

One Ethernet Interface Controllers and one ComM channel belong to one certain Ethernet network (handle).

[SWS_EthSM_00022] ⌈Depending on the network mode state machine, the EthSM shall control the Ethernet Interface Controller modes of each Ethernet network.⌋

[SWS_EthSM_00023] ⌈The EthSM shall use the API of the EthIf to control the operating modes of the assigned Ethernet Interface Controllers.⌋

7.4 Multiple networks

The Ethernet State Manager shall be able to handle separate networks. This concerns separate physical networks (see also chapter 7.3) and also separate VLAN's on the same physical network.

In both cases, the separation is done by separate handles per physical or virtual network. VLANs appear on higher layers (ComM) as separate networks. E.g.: If there is one physical Ethernet Interface Controller and two VLANs assigned to it, two ComM channels exists.

7.5 Background and Rationale

Explanation:

The application is responsible to recognize if the Ethernet network is needed or not.

One possible use case could be the usage of the Ethernet network in a tester connection (see description below).

Use Case: Use Ethernet in a tester connection

For example, the detection could takes place over a separate hardware pin of the ECU. In this case, the activation of the hardware pin and therefore the activation of the Ethernet network can only realized through the offboard-diagnostic tester. Reasons for the deactivation of the Ethernet network could be:

- The tester deactivate via the separate hardware pin the network
- The application deactivate the network
- The application recognize a timeout
- The link status of the network failed

Note:

The Ethernet network is requested by the application to wake-up or shutdown. The applicaction either need to communicate or do not need to communicate on the Ethernet network. Therefore, the EthSM statemashine has to provide the corresponding main states ETHSM_FULL_COMMUNICATION and ETHSM_NO_COMMUNICATION, but no other states, e.g. ETHSM_SILENT_COMMUNICATION.

7.6 Network mode state machine

[SWS_EthSM_00024] ⌈The EthSM shall implement for each configured network handle one network mode state machine (EthSM statemachine).⌋

[SWS_EthSM_00203] ⌈The EthSM statemachine shall consist of 2 main states: ETHSM_FULL_COMMUNICATION and ETHSM_NO_COMMUNICATION.⌋

[SWS_EthSM_00204] ⌈The ETHSM_FULL_COMMUNICATION shall have 3 sub-states: ETHSM_STATE_OFFLINE, ETHSM_STATE_WAIT_TRCVLINK and ETHSM_STATE_WAIT_ONLINE.⌋

[SWS_EthSM_00205] ⌈The ETHSM_NO_COMMUNICATION shall have 3 sub-states: ETHSM_STATE_ONLINE, ETHSM_STATE_ONHOLD and ETHSM_STATE_WAIT_OFFLINE.⌋

The following points give an overview, on how the EthSM statemachine generally reacts on requested communication modes:

1. If communication mode COMM_FULL_COMMUNICATION is requested for a certain Ethernet network and the corresponding EthSM statemachine is in state ETHSM_NO_COMMUNICATION, then at least the following preconditions have to be fulfilled to transit to ETHSM_FULL_COMMUNICATION:
   - Request ETH_MODE_ACTIVE for the corresponding Ethernet Interface Controller.
   - The corresponding Ethernet hardware has indicated ETHTRCV_LINK_STATE_ACTIVE (active link state, thus, Ethernet cable is connected) via EthSM_TrcvLinkStateChg
   - The corresponding TcpIpCtrl has indicated TCPIP_STATE_ONLINE (IP communication is available) via EthSM_TcpIpModeIndication
2. If communication mode COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST is requested for a certain Ethernet network, EthSMWakeupSleepOnDatalineEnabled is set to TRUE and the corresponding EthSM statemachine is in state ETHSM_NO_COMMUNICATION, then at least the following preconditions have to be fulfilled to transit to ETHSM_FULL_COMMUNICATION:
   - Request ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST for the corresponding Ethernet Interface Controller.

- The corresponding Ethernet hardware has indicated ETHTRCV_LINK_STATE_ACTIVE (active link state, thus, Ethernet cable is connected) via EthSM_TrcvLinkStateChg
- The corresponding TcpIpCtrl has indicated TCPIP_STATE_ONLINE (IP communication is available) via EthSM_TcpIpModeIndication

3. If the ComM requests COMM_NO_COMMUNICATION and the corresponding EthSM state machine is in state ETHSM_STATE_WAIT_OFFLINE, then the following preconditions have to be fulfilled to transist to ETHSM_NO_COMMUNICATION:
   - EthSM request ETH_MODE_DOWN for the corresponding Ethernet Interface Controller
   - The corresponding TcpIpCtrl has indicated TCPIP_STATE_OFFLINE
4. If communication mode COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST is requested for a certain Ethernet network, EthSMWakeupSleepOnDatalineEnabled is set to TRUE and the corresponding EthSM statemachine is in state ETHSM_FULL_COMMUNICATION, then the EthSM request ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST for the corresponding Ethernet Interface Controller, even though the current state is ETHSM_FULL_COMMUNICATION

The re-trigger of the Ethernet Interface Controller described in point 5 in the previous section is used to request the lower layer to trigger a wake-up on the network. This could be used for example for Ethernet hardware which is compliant with OA TC10 ([1])

The internal states are shown in Figure 7.1. Details are described in the following sub chapters. An overview of the internal states are described in [SWS_EthSM_00041].

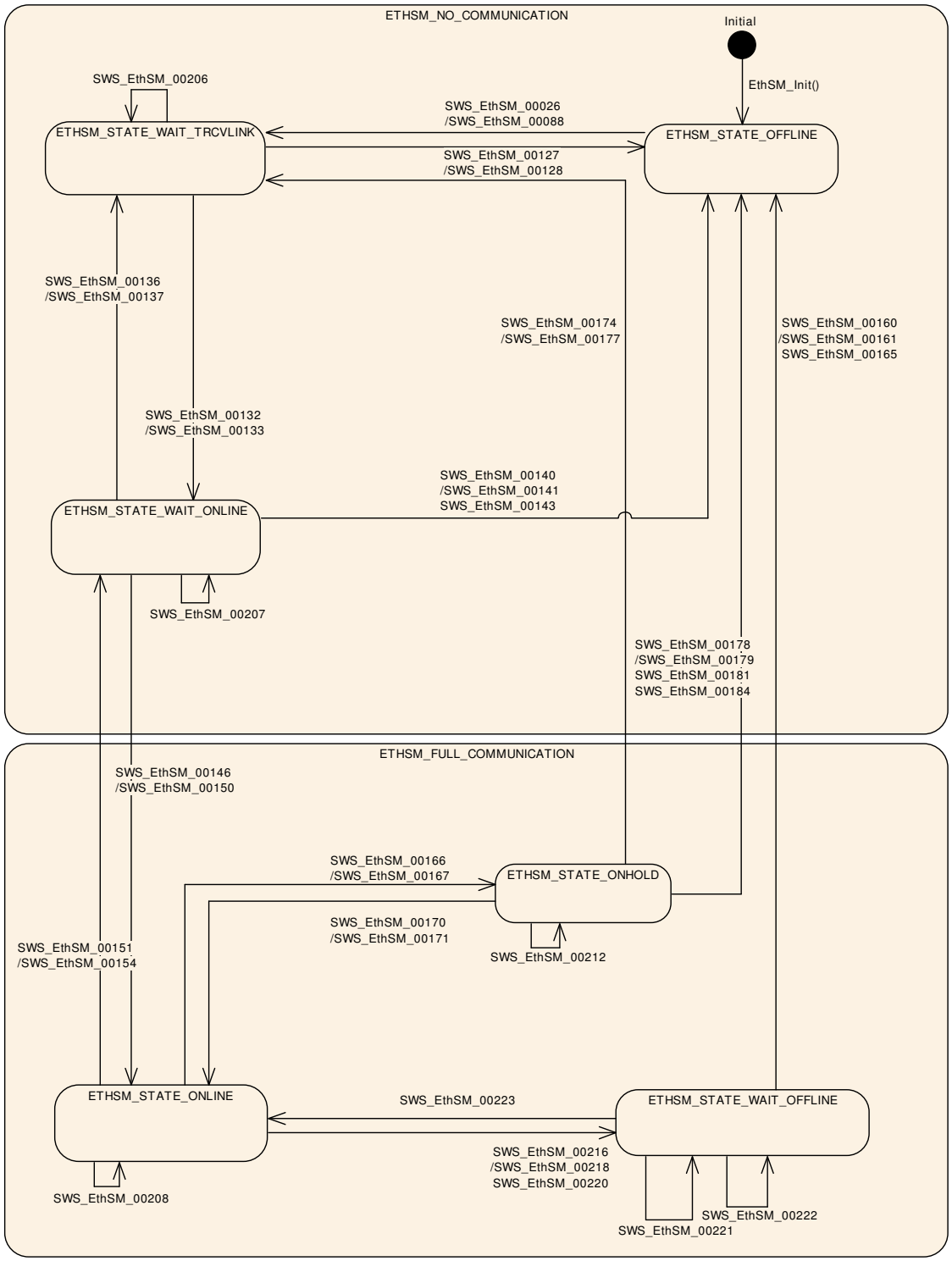

Figure 7.1: Network mode state machine of the EthSM

Number	Heading
[SWS_EthSM_00026]	Transition from OFFLINE to WAIT_TRCVLINK
[SWS_EthSM_00206]	Self transition WAIT_TRCVLINK
[SWS_EthSM_00127]	Transition from WAIT_TRCVLINK to OFFLINE
[SWS_EthSM_00132]	Transition from WAIT_TRCVLINK to WAIT_ONLINE
[SWS_EthSM_00207]	Self transition WAIT_ONLINE
[SWS_EthSM_00136]	Transition from WAIT_ONLINE to WAIT_TRCVLINK
[SWS_EthSM_00140]	Transition from WAIT_ONLINE to OFFLINE
[SWS_EthSM_00146]	Transition from WAIT_ONLINE to ONLINE
[SWS_EthSM_00208]	Self transition STATE_ONLINE
[SWS_EthSM_00151]	Transition from ONLINE to WAIT_ONLINE
[SWS_EthSM_00216]	Transition from ONLINE to WAIT_OFFLINE
[SWS_EthSM_00166]	Transition from ONLINE to ONHOLD
[SWS_EthSM_00221]	Self transition WAIT_OFFLINE
[SWS_EthSM_00222]	Self transition WAIT_OFFLINE
[SWS_EthSM_00160]	Transition from WAIT_OFFLINE to OFFLINE
[SWS_EthSM_00223]	Transition from WAIT_OFFLINE to ONLINE
[SWS_EthSM_00212]	Self transition ONHOLD
[SWS_EthSM_00170]	Transition from ONHOLD to ONLINE
[SWS_EthSM_00174]	Transition from ONHOLD to WAIT_TRCVLINK
[SWS_EthSM_00178]	Transition from ONHOLD to OFFLINE

Table 7.1: EthSM SWS items referenced in the state machine diagram

Table 7.2 shows the expected state of the EthSM sub state, the state of the affected Ethernet hardware and the IP assignment, according to the requested communication mode of the ComM.

Requested ComM Mode	EthSM sub state	Controller state	Transceiver state	Transceiver link state	TcpIpCtrl state
No Com	ETHSM_STATE_ OFFLINE	DOWN	DOWN	DOWN	OFFLINE
No Com	ETHSM_STATE_ WAIT_TRCVLINK	ACTIVE	ACTIVE	DOWN	OFFLINE
No Com	ETHSM_STATE_ WAIT_ONLINE	ACTIVE	ACTIVE	ACTIVE	OFFLINE
Full Com	ETHSM_STATE_ ONLINE	ACTIVE	ACTIVE	ACTIVE	ONLINE
Full Com	ETHSM_STATE_ ONHOLD	ACTIVE	ACTIVE	DOWN	ONLINE
No Com	ETHSM_STATE_ WAIT_OFFLINE	ACTIVE	ACTIVE	ACTIVE	ONLINE

Table 7.2: EthSM sub states and the excepted states of the according Ethernet hardware and the IP assignment.

The following preconditions have to be fulfilled to transit to ETHSM_FULL_COMMUNICATION:

- Ethernet Interface Controller and transceiver are active

- The transceiver link state is active
- An active IP communication is available

The first step is set the Ethernet controller to ETH_MODE_ACTIVE. After this is done, the Ethernet State Manager is in the sub state ETHSM_STATE_WAIT_TRCVLINK. In this sub state the state manager has to wait for the monitored link state information of the Ethernet transceiver. After the link state is set to ETHTRCV_MODE_ACTIVE, the Ethernet State Manager transit to sub state ETHSM_STATE_WAIT_ONLINE. In this sub state the Ethernet state manager has to wait for the monitored TcpIp stateinformation of the TcpIp module. After the TcpIp module indicate TCPIP_STATE_ONLINE (= IP communication is available), the Ethernet State Manager transit to main state ETHSM_FULL_COMMUNICATION and sub state ETHSM_STATE_ONLINE.

7.6.1 Initial transition

[SWS_EthSM_00025] ⌈After the initialization of the EthSM the state machine shall have a transition to ETHSM_STATE_OFFLINE.⌋

Note: The initialization of the EthSM causes no further transactions in other modules. So no separate sequence diagram is needed.

7.6.2 Behaviour in sub-state ETHSM_STATE_OFFLINE

7.6.2.1 Transition from sub-state OFFLINE to WAIT_TRCVLINK

[SWS_EthSM_00026] **Transition from OFFLINE to WAIT_TRCVLINK** ⌈In the state ETHSM_STATE_OFFLINE the state machine shall have a transition to ETHSM_STATE_WAIT_TRCVLINK, if ComM requests for the corresponding network handle: COMM_FULL_COMMUNICATION or COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST and EthSMWakeupSleepOnDatalineEnabled is set to TRUE⌋

Note: The transition is shown as sequence diagram in Figure 9.2.

[SWS_EthSM_00088] ⌈The transition from ETHSM_STATE_OFFLINE to ETHSM_STATE_WAIT_TRCVLINK shall set the controller mode according to the requested ComM communication mode:

- If ComM requested COMM_FULL_COMMUNICATION, then the controller mode of the corresponding EthIfCtrl shall be requested with ETH_MODE_ACTIVE
- If ComM requested COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST and EthSMWakeupSleepOnDatalineEnabled is set to TRUE, then the controller mode of corresponding EthIfCtrl shall be requested with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST

⌋

7.6.3 Behavior in sub-state ETHSM_STATE_WAIT_TRCVLINK

[SWS_EthSM_00206] Self transition WAIT_TRCVLINK

Upstream requirements: SRS_Eth_00158

⌈If EthSMWakeupSleepOnDatalineEnabled is set to TRUE and the EthSM statmachine is in state ETHSM_STATE_WAIT_TRCVLINK, and ComM request now COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST, then EthSM shall request the corresponding EthIfCtrl with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST.⌋

7.6.3.1 Transition from sub-state WAIT_TRCVLINK to OFFLINE

[SWS_EthSM_00127] Transition from WAIT_TRCVLINK to OFFLINE ⌈In the state ETHSM_STATE_WAIT_TRCVLINK the state machine shall have a transition to ETHSM_STATE_OFFLINE, if the ComM requests COMM_NO_COMMUNICATION for the corresponding network handle.⌋

[SWS_EthSM_00128] ⌈The transition from ETHSM_STATE_WAIT_TRCVLINK to ETHSM_STATE_OFFLINE shall set the controller mode to ETH_MODE_DOWN.⌋

7.6.3.2 Transition from sub-state WAIT_TRCVLINK to WAIT_ONLINE

[SWS_EthSM_00132] Transition from WAIT_TRCVLINK to WAIT_ONLINE ⌈In the state ETHSM_STATE_WAIT_TRCVLINK the state machine shall have a transition to ETHSM_STATE_WAIT_ONLINE, if the latest indicated link state reported by Ethernet Interface is ETHTRCV_LINK_STATE_ACTIVE for the corresponding network handle.⌋

Note: The transition is shown as sequence diagram in Figure 9.2.

[SWS_EthSM_00133] ⌈The transition from ETHSM_STATE_WAIT_TRCVLINK to ETHSM_STATE_WAIT_ONLINE shall request the TcpIp state TCPIP_STATE_ONLINE from the TcpIp module.⌋

[SWS_EthSM_00255] PASS condition for Link down detection ⌈If the optional configuration parameter ETHSM_E_LINK_DOWN exists, ETHSM_E_LINK_DOWN with EventStatus DEM_EVENT_STATUS_PASSED shall be reported to the DEM module when switching from ETHSM_STATE_WAIT_TRCVLINK to ETHSM_STATE_WAIT_ONLINE.⌋

7.6.4 Behaviour in sub-state ETHSM_STATE_WAIT_ONLINE

[SWS_EthSM_00207] Self transition WAIT_ONLINE

Upstream requirements: SRS_Eth_00158

⌈If EthSMWakeupSleepOnDatalineEnabled is set to TRUE and EthSM is in state ETHSM_STATE_WAIT_ONLINE and ComM requests now COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST, then EthSM shall request the corresponding EthIfCtrl with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST.⌋

7.6.4.1 Transition from sub-state WAIT_ONLINE to WAIT_TRCVLINK

[SWS_EthSM_00136] Transition from WAIT_ONLINE to WAIT_TRCVLINK ⌈In the state ETHSM_STATE_WAIT_ONLINE the state machine shall have a transition to ETHSM_STATE_WAIT_TRCVLINK, if the Ethernet interface reports ETHTRCV_LINK_STATE_DOWN for the corresponding network handle.⌋

[SWS_EthSM_00137] ⌈The transition from ETHSM_STATE_WAIT_ONLINE to ETHSM_STATE_WAIT_TRCVLINK shall request the TcpIp state TCPIP_STATE_OFFLINE from the TcpIp module.⌋

7.6.4.2 Transition from sub-state WAIT_ONLINE to OFFLINE

[SWS_EthSM_00140] Transition from WAIT_ONLINE to OFFLINE ⌈In the state ETHSM_STATE_WAIT_ONLINE the state machine shall have a transition to ETHSM_STATE_OFFLINE, if the ComM requests COMM_NO_COMMUNICATION for the corresponding network handle.⌋

[SWS_EthSM_00141] ⌈The transition from ETHSM_STATE_WAIT_ONLINE to ETHSM_STATE_OFFLINE sets the controller mode to ETH_MODE_DOWN.⌋

[SWS_EthSM_00143] ⌈The transition from ETHSM_STATE_WAIT_ONLINE to ETHSM_STATE_OFFLINE shall request the TcpIp state TCPIP_STATE_OFFLINE from the TcpIp module.⌋

7.6.4.3 Transition from sub-state WAIT_ONLINE to ONLINE

[SWS_EthSM_00146] Transition from WAIT_ONLINE to ONLINE ⌈If EthSMTcpIpUsed is TRUE, in the state ETHSM_STATE_WAIT_ONLINE the state machine shall have a transition to ETHSM_STATE_ONLINE, if the TcpIp modul reports TCPIP_STATE_ONLINE for the corresponding network handle. If EthSMTcpIpUsed

is FALSE, in the state ETHSM_STATE_WAIT_ONLINE the state machine shall have a transition to ETHSM_STATE_ONLINE unconditionally.⌋

Note: The transition is shown as sequence diagram in Figure 9.2.

[SWS_EthSM_00150] ⌈After the successful transition from ETHSM_STATE_WAIT_ONLINE to ETHSM_STATE_ONLINE the Ethernet State Manager shall set the main state to ETHSM_FULL_COMMUNICATION and call the callback function ComM_<Bus>SM_ModeIndication of the ComM with communication mode COMM_FULL_COMMUNICATION.⌋

7.6.5 Behaviour in sub-state ETHSM_STATE_ONLINE

[SWS_EthSM_00208] **Self transition STATE_ONLINE** ⌈If the EthSM statemachine is in state ETHSM_STATE_ONLINE, EthSMWakeupSleepOnDatalineEnabled is set to TRUE and ComM request COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST, then the controller mode of the corresponding EthIfCtrl shall be requested with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST.⌋

7.6.5.1 Transition from sub-state ONLINE to WAIT_ONLINE

[SWS_EthSM_00151] **Transition from ONLINE to WAIT_ONLINE** ⌈If EthSMTcpIpUsed is TRUE, in the state ETHSM_STATE_ONLINE the state machine shall have a transition to ETHSM_STATE_WAIT_ONLINE, if the TcpIp modul reports TCPIP_STATE_OFFLINE for the corresponding network handle.⌋

[SWS_EthSM_00154] ⌈After the successful transition from ETHSM_STATE_ONLINE to ETHSM_STATE_WAIT_ONLINE the Ethernet State Manager shall set the main state to ETHSM_NO_COMMUNICATION and call the callback function ComM_<Bus>SM_ModeIndication of the ComM with communication mode COMM_NO_COMMUNICATION.⌋

7.6.5.2 Transition from sub-state ONLINE to WAIT_OFFLINE

[SWS_EthSM_00216] **Transition from ONLINE to WAIT_OFFLINE** ⌈In the state ETHSM_STATE_ONLINE the state machine shall do a transition to ETHSM_STATE_WAIT_OFFLINE, if the ComM requests COMM_NO_COMMUNICATION for the corresponding network handle.⌋

Note: The transition is shown as sequence diagram in Figure 9.2.

[SWS_EthSM_00218] TCPIP to OFFLINE in transition from WAIT_ONLINE to WAIT_OFFLINE ⌈During the transition between ETHSM_STATE_ONLINE and ETHSM_STATE_WAIT_OFFLINE triggered by EthSM_RequestComMode with COMM_NO_COMMUNICATION, EthSM shall call the API TcpIp_RequestComMode with TCPIP_STATE_OFFLINE.⌋

[SWS_EthSM_00220] ⌈During the transition between ETHSM_STATE_ONLINE and ETHSM_STATE_WAIT_OFFLINE triggered by EthSM_RequestComMode with COMM_NO_COMMUNICATION, the API TcpIp_RequestComMode shall be called with TCPIP_STATE_OFFLINE.⌋

7.6.5.3 Transition from sub-state ONLINE to ONHOLD

[SWS_EthSM_00166] Transition from ONLINE to ONHOLD ⌈In the state ETHSM_STATE_ONLINE the state machine shall have a transition to ETHSM_STATE_ONHOLD, if the Ethernet Interface reports ETHTRCV_LINK_STATE_DOWN for the corresponding network handle.⌋

Note: The transition is shown as sequence diagram in Figure 9.3.

[SWS_EthSM_00167] ⌈The transition from ETHSM_STATE_ONLINE to ETHSM_STATE_ONHOLD shall request the TcpIp state TCPIP_STATE_ONHOLD from the TcpIp module.⌋

[SWS_EthSM_00188] ⌈If the optional configuration parameter ETHSM_E_LINK_DOWN exists, ETHSM_E_LINK_DOWN with EventStatus DEM_EVENT_STATUS_FAILED shall be reported to the DEM module when switching from ETHSM_STATE_ONLINE to

ETHSM_STATE_ONHOLD.⌋

7.6.6 Behaviour in sub-state ETHSM_STATE_WAIT_OFFLINE

[SWS_EthSM_00221] Self transition WAIT_OFFLINE ⌈In the state ETHSM_STATE_WAIT_OFFLINE, if ComM requests COMM_FULL_COMMUNICATION for the corresponding network handle, EthSM shall set the controller mode to ETH_MODE_ACTIVE and call the API TcpIp_RequestComMode with TCPIP_STATE_ONLINE⌋

[SWS_EthSM_00222] Self transition WAIT_OFFLINE ⌈In the state ETHSM_STATE_WAIT_OFFLINE, if ComM requests COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST for the corresponding network handle and EthSMWakeupSleepOnDatalineEnabled is set to TRUE, EthSM shall set the controller mode to

ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST and call the API `TcpIp_RequestComMode` with TCPIP_STATE_ONLINE⌋

7.6.6.1 Transition from sub-state WAIT_OFFLINE to OFFLINE

[SWS_EthSM_00160] Transition from WAIT_OFFLINE to OFFLINE ⌈If EthSMTcpIpUsed is TRUE, in the state ETHSM_STATE_WAIT_OFFLINE the state machine shall have a transition to `ETHSM_STATE_OFFLINE`, if the TcpIp modul reports TCPIP_STATE_OFFLINE for the corresponding network handle. If EthSMTcpIpUsed is FALSE, in the state ETHSM_STATE_WAIT_OFFLINE the state machine shall have a transition to `ETHSM_STATE_OFFLINE` unconditionally.⌋

[SWS_EthSM_00161] ⌈The transition from ETHSM_STATE_WAIT_OFFLINE to `ETHSM_STATE_OFFLINE` shall set the controller mode to ETH_MODE_DOWN.⌋

[SWS_EthSM_00165] ⌈After the successful transition from ETHSM_STATE_WAIT_OFFLINE to `ETHSM_STATE_OFFLINE` the Ethernet State Manager shall set the main state to ETHSM_NO_COMMUNICATION and call the callback function ComM_<Bus>SM_ModeIndication of the ComM with communication mode COMM_NO_COMMUNICATION.⌋

7.6.6.2 Transistion from sub-state WAIT_OFFLINE to ONLINE

[SWS_EthSM_00223] Transition from WAIT_OFFLINE to ONLINE ⌈In the state ETHSM_STATE_WAIT_OFFLINE the state machine shall have a transition to `ETHSM_STATE_ONLINE`, if the TcpIp module reports TCPIP_STATE_ONLINE for the corresponding network handle, ComM requested COMM_FULL_COMMUNICATION or COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST⌋

7.6.7 Behaviour in sub-state `ETHSM_STATE_ONHOLD`

[SWS_EthSM_00212] Self transition ONHOLD

Upstream requirements: SRS_Eth_00158

⌈If the EthSM statmachine is in state `ETHSM_STATE_ONHOLD`, EthSMWakeupSleepOnDatalineEnabled is set to TRUE and ComM request COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST, then the controller mode of the corresponding EthIfCtrl shall be requested with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST.⌋

7.6.7.1 Transistion from ONHOLD to ONLINE

[SWS_EthSM_00170] Transition from ONHOLD to ONLINE ⌈In the state ETHSM_STATE_ONHOLD the state machine shall have a transition to ETHSM_STATE_ONLINE, if the Ethernet interface reports ETHTRCV_LINK_STATE_ACTIVE for the corresponding network handle.⌋

[SWS_EthSM_00171] ⌈The transition from ETHSM_STATE_ONHOLD to ETHSM_STATE_ONLINE shall request the TcpIp state TCPIP_STATE_ONLINE from the TcpIp module.⌋

[SWS_EthSM_00196] ⌈If the optional configuration parameter ETHSM_E_LINK_DOWN exists, ETHSM_E_LINK_DOWN with EventStatus DEM_EVENT_STATUS_PASSED shall be reported to the DEM module when switching from ETHSM_STATE_ONHOLD to

ETHSM_STATE_ONLINE.⌋

7.6.7.2 Transition from sub-state ONHOLD to WAIT_TRCVLINK

[SWS_EthSM_00174] Transition from ONHOLD to WAIT_TRCVLINK ⌈If EthSMTcpIpUsed is TRUE, in the state ETHSM_STATE_ONHOLD the state machine shall have a transition to ETHSM_STATE_WAIT_TRCVLINK, if the TcpIp modul reports TCPIP_STATE_OFFLINE for the corresponding network handle. If EthSMTcpIpUsed is FALSE, in the state ETHSM_STATE_ONHOLD the state machine shall have a transition to ETHSM_STATE_WAIT_TRCVLINK unconditionally.⌋

Note: The transition is shown as sequence diagram in Figure 9.3.

[SWS_EthSM_00177] ⌈After the successful transition from ETHSM_STATE_ONHOLD to E ETHSM_STATE_WAIT_TRCVLINK the Ethernet State Manager shall set the main state to ETHSM_NO_COMMUNICATION and call the callback function ComM_<Bus>SM_ModeIndication of the ComM with communication mode COMM_NO_COMMUNICATION.⌋

7.6.7.3 Transition from sub-state ONHOLD to OFFLINE

[SWS_EthSM_00178] Transition from ONHOLD to OFFLINE ⌈In the state ETHSM_STATE_ONHOLD the state machine shall have a transition to ETHSM_STATE_OFFLINE, if the ComM requests COMM_NO_COMMUNICATION for the corresponding network handle.⌋

[SWS_EthSM_00179] ⌈The transition from ETHSM_STATE_ONHOLD to ETHSM_STATE_OFFLINE shall set the controller mode to ETH_MODE_DOWN.⌋

[SWS_EthSM_00181] ⌈The transition from ETHSM_STATE_ONHOLD to ETHSM_STATE_OFFLINE shall request the TcpIp state TCPIP_STATE_OFFLINE from the TcpIp module.⌋

[SWS_EthSM_00184] ⌈After the successful transition from ETHSM_STATE_ONHOLD to ETHSM_STATE_OFFLINE the Ethernet State Manager shall set the main state to ETHSM_NO_COMMUNICATION and call the callback function ComM_<Bus>SM_ModeIndication of the ComM with communication mode COMM_NO_COMMUNICATION.⌋

7.6.8 Information about state transitions

[SWS_EthSM_00083] ⌈Evertime the EthSM statemachine enters a sub-state, the EthSM shall report the entered sub-state towards the BswM by calling BswM_EthSM_CurrentState(<entered sub-state>) (e.g. when entering sub-state ETHSM_STATE_ONLINE, EthSM shall call BswM_EthSM_CurrentState(ETHSM_STATE_ONLINE). The transistion to ETHSM_STATE_OFFLINE after initialization (see [SWS_EthSM_00025]) shall not be reported towards the BswM.⌋

Note: The BswM need the information about the EthSM internal states (see [SWS_EthSM_00041])

Note: The ComM needs the information about the communication states, e.g. COMM_FULL_COMMUNICATION or COMM_NO_COMMUNICATION. The calls towards the ComM are specified explicitely within the sub-states.

7.7 Commercial Off The Shelf stack usage

A commercial off the shelf stack (COTS) shall be useable. The commercial stack is useable without adaptation (Variant 1 in Figure 7.2). However, the Ethernet State Manager is not able to control the Ethernet Interface Controller and Ethernet transceiver in this case. The commercial stack may be adapted for usage with the Ethernet Interface. In this case, the Ethernet State Manager is able to control both Ethernet Interface Controller and Ethernet transceiver (Variant 2 in Figure 7.2).

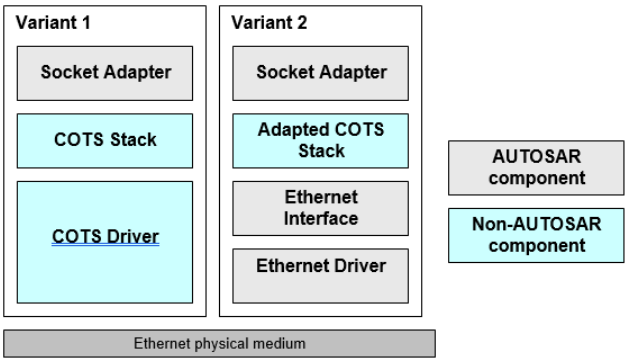

Figure 7.2: BSW stack architecture variants

[SWS_EthSM_00078] ⌈It is possible to set the Ethernet State Manager in a dummy mode. In this mode, the Ethernet State Manager doesn't support the API to the Ethernet interface. The API to the ComM is available but the functionality is deactivated. The function calls from the ComM will be answered with the return value E_OK.⌋

Note: see chapter 10 configuration specification

7.8 Error classification

Chapter [2, General Specification of Basic Software Modules] 7.2 *"Error Handling"* describes the error handling of the Basic Software in detail. Above all, it constitutes a classification scheme consisting of five error types which may occur in BSW modules.

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

7.8.1 Development Errors

[SWS_EthSM_00200] Definition of development errors in module EthSM ⌈

Type of error	*Related error code*	*Error value*
Invalid communication mode requested	ETHSM_E_INVALID_NETWORK_MODE	0x01
EthSM module was not initialized	ETHSM_E_UNINIT	0x02
Invalid pointer in parameter list	ETHSM_E_PARAM_POINTER	0x03
Invalid parameter in parameter list	ETHSM_E_INVALID_NETWORK_HANDLE	0x04
Invalid Tcplp state	ETHSM_E_INVALID_TCP_IP_MODE	0x05
Invalid parameter in parameter list	ETHSM_E_PARAM_CONTROLLER	0x07

⌋

7.8.2 Runtime Errors

There are no runtime errors

7.8.3 Production Errors

[SWS_EthSM_00202] Link down detection

Upstream requirements: SRS_BSW_00458

⌈

Diagnostic Event (Error Name)	ETHSM_E_LINK_DOWN
Description	It shall be reported when the transceiver switches to "down" while communication has already been established and is requested because of communication request.
Failed condition	During transition from ETHSM_STATE_ONLINE to ETHSM_STATE_ONHOLD, which is triggered by EthSM_TrcvLinkStateChg(ETHTRCV_LINK_STATE_DOWN).
Passed condition	During transition from ETHSM_STATE_ONHOLD to ETHSM_STATE_ONLINE or transition from ETHSM_STATE_WAIT_TRCVLINK to ETHSM_STATE_WAIT_ ONLINE, which are triggered by EthSM_TrcvLinkStateChg(ETHTRCV_LINK_STATE_ ACTIVE).

⌋

8 API specification

8.1 Imported types

[SWS_EthSM_91001] Definition of imported datatypes of module EthSM ⌈

Module	*Header File*	*Imported Type*
ComM	Rte_ComM_Type.h	ComM_ModeType
Comtype	ComStack_Types.h	NetworkHandleType
Dem	Rte_Dem_Type.h	Dem_EventIdType
	Rte_Dem_Type.h	Dem_EventStatusType
Eth	Eth_GeneralTypes.h	Eth_ModeType
EthTrcv	Eth_GeneralTypes.h	EthTrcv_LinkStateType
Std	Std_Types.h	Std_ReturnType
TcpIp	TcpIp.h	TcpIp_StateType

⌋

8.2 Type definitions

8.2.1 EthSM_NetworkModeStateType

[SWS_EthSM_00041] Definition of datatype EthSM_NetworkModeStateType ⌈

Name	EthSM_NetworkModeStateType		
Kind	Enumeration		
Range	ETHSM_STATE_OFFLINE	–	EthSM is initialized in this state. (TcpIp state is OFFLINE)
	ETHSM_STATE_WAIT_TRCVLINK	–	ComM requests COMM_FULL_COMMUNICATION in this state. Controller will be set to ACTIVE. EthSM waits for transceiver link state (ACTIVE).
	ETHSM_STATE_WAIT_ONLINE	–	Transceiver link state is ACTIVE EthSM waits for IP communication (TcpIP state is ONLINE)
	ETHSM_STATE_ONLINE	–	IP communication is available ComM state COMM_FULL_COMMUNICATION is reached
	ETHSM_STATE_ONHOLD	–	EthSM lost active transceiver link state, TcpIP state is still ONLINE)
	ETHSM_STATE_WAIT_OFFLINE	–	ComM requested COMM_NO_COMMUNICATION in this state (TcpIp state is requested with OFFLINE)
Description	This type shall define the states of the network mode state machine.		
Available via	EthSM.h		

⌋

8.3 Function definitions

This chapter describes all API functions provided by the module

Note:

Depending on the Ethernet hardware, it may become necessary that implementations deviate from API specifications in respect to the asynchronous or synchronous behaviour.

8.3.1 Generic functions

8.3.1.1 EthSM_Init

[SWS_EthSM_00043] Definition of API function EthSM_Init

Upstream requirements: SRS_BSW_00405, SRS_BSW_00101, SRS_BSW_00358, SRS_BSW_00414

⌈

Service Name	EthSM_Init
Syntax	`void EthSM_Init ( void )`
Service ID [hex]	0x07
Sync/Async	Synchronous
Reentrancy	Non Reentrant
Parameters (in)	None
Parameters (inout)	None
Parameters (out)	None
Return value	None
Description	This function initialize the EthSM.
Available via	EthSM.h

⌋

8.3.1.2 EthSM_RequestComMode

[SWS_EthSM_00050] Definition of API function EthSM_RequestComMode ⌈

Service Name	EthSM_RequestComMode
Syntax	`Std_ReturnType EthSM_RequestComMode ( NetworkHandleType network, ComM_ModeType mode )`
Service ID [hex]	0x40
Sync/Async	Asynchronous
Reentrancy	Reentrant for different networks. Non reentrant for the same network.

▽

△

Parameters (in)	network	Handle of the communication network
	mode	Requested communication mode
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request accepted E_NOT_OK: Request not accepted
Description	Request of a new communication mode.	
Available via	EthSM.h	

⌋

Remark: The function reentrancy is limited to different network handles. Reentrancy for the same network is not to be regarded here.

[SWS_EthSM_00051] ⌈The function EthSM_RequestComMode checks the network handle of the request. It only accepts the request, if the network handle of the request is a handle contained in the EthSM configuration (configuration parameter EthSMNetworkHandle). In this case the return value is set to E_OK.

If it is not contained in the configuration, the function denies the request. In this case the return value is set to E_NOT_OK.⌋

[SWS_EthSM_00052] ⌈The function EthSM_RequestComMode shall report ETHSM_E_INVALID_NETWORK_HANDLE to the DET, if it does not accept the network handle of the request.⌋

[SWS_EthSM_00095] ⌈The function EthSM_RequestComMode shall report ETHSM_E_INVALID_NETWORK_MODE to the DET, if it does not accept the mode of the request.⌋

[SWS_EthSM_00053] ⌈If the function EthSM_RequestComMode accepts the function call, it shall store the communication mode for the network handle and the corresponding network mode switch of the state machine shall be initiated in the next main function cycle latest.⌋

Note: COMM_SILENT_COMMUNICATION is not processed by the Ethernet stack and will be ignored by EthSM.

[SWS_EthSM_00054]

Upstream requirements: SRS_BSW_00406

⌈The function EthSM_RequestComMode shall report ETHSM_E_UNINIT to the DET, if the EthSM is not initialized yet.⌋

8.3.1.3 EthSM_GetCurrentComMode

[SWS_EthSM_00055] Definition of API function EthSM_GetCurrentComMode ⌈

Service Name	EthSM_GetCurrentComMode	
Syntax	`Std_ReturnType EthSM_GetCurrentComMode ( NetworkHandleType network, ComM_ModeType* mode )`	
Service ID [hex]	0x41	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different networks. Non reentrant for the same network.	
Parameters (in)	network	Handle of the communication channel
Parameters (inout)	None	
Parameters (out)	mode	Pointer to the memory location where the current communication mode shall be stored
Return value	Std_ReturnType	`E_OK`: Request accepted `E_NOT_OK`: Request not accepted
Description	Query the current communication mode.	
Available via	EthSM.h	

⌋

[SWS_EthSM_00057] ⌈The function `EthSM_GetCurrentComMode` checks the network handle of the service request. It only accepts the service, if the network handle of the request is a handle contained in the EthSM configuration (configuration parameter EthSMNetworkHandle). In this case the return value is set to `E_OK`.

If it is not contained in the configuration, the function denies the request. In this case the return value is set to `E_NOT_OK`.⌋

[SWS_EthSM_00058] ⌈The function `EthSM_GetCurrentComMode` shall report `ETHSM_E_INVALID_NETWORK_HANDLE` to the DET, if it does not accept the network handle of the request.⌋

[SWS_EthSM_00059] ⌈The function `EthSM_GetCurrentComMode` puts out the current communication mode for the network handle to the designated pointer of type ComM_ModeType, if it accepts the request.⌋

Remark: Because the Ethernet hardware needs a certain time to proceed with the request and there is currently no notification mechanism specified, the real hardware mode and the mode notified by the EthSM might be different until the hardware is ready.

[SWS_EthSM_00060]

Upstream requirements: SRS_BSW_00406, SRS_BSW_00374, SRS_BSW_00003, SRS_BSW_00318

⌈The function `EthSM_GetCurrentComMode` shall report `ETHSM_E_UNINIT` to the DET, if the EthSM is not initialized yet.⌋

8.3.2 Mode indication functions

8.3.2.1 EthSM_CtrlModeIndication

[SWS_EthSM_00190] Definition of callback function EthSM_CtrlModeIndication ⌈

Service Name	EthSM_CtrlModeIndication	
Syntax	`void EthSM_CtrlModeIndication ( uint8 CtrlIdx, Eth_ModeType CtrlMode )`	
Service ID [hex]	0x09	
Sync/Async	Synchronous	
Reentrancy	Reentrant (only for different Ethernet controllers)	
Parameters (in)	CtrlIdx	Ethernet Interface Controller whose mode has changed
	CtrlMode	Notified Ethernet Interface Controller mode
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Called when mode has been read out. Either triggered by previous EthIf_GetControllerMode or by EthIf_SetControllerMode call. Can directly be called within the trigger functions.	
Available via	EthSM.h	

⌋

[SWS_EthSM_00191] ⌈If the function `EthSM_CtrlModeIndication` gets a CtrlIdx, which is not configured in the configuration of the EthSM module, it shall call the function `Det_ReportError` with ErrorId parameter ETHSM_E_PARAM_CONTROLLER.⌋

[SWS_EthSM_00192] ⌈If the EthSM module is not initialized, when the function `EthSM_CtrlModeIndication` is called, then the function `EthSM_CtrlModeIndication` shall call the function `Det_ReportError` with ErrorId parameter ETHSM_E_UNINIT.⌋

8.3.2.2 EthSM_TrcvLinkStateChg

[SWS_EthSM_00109] Definition of callback function EthSM_TrcvLinkStateChg ⌈

Service Name	EthSM_TrcvLinkStateChg
Syntax	`void EthSM_TrcvLinkStateChg ( uint8 CtrlIdx, EthTrcv_LinkStateType TransceiverLinkState )`
Service ID [hex]	0x06
Sync/Async	Synchronous
Reentrancy	Non Reentrant

▽

△

Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	TransceiverLinkState	Actual transceiver link state of the specific network handle
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This service is called by the Ethernet Interface to report a transceiver link state change.	
Available via	EthSM.h	

⌋

[SWS_EthSM_00112] ⌈The function `EthSM_TrcvLinkStateChg` shall report ETHSM_E_PARAM_CONTROLLER to the DET, if it does not accept the CtrlIdx of the function call.⌋

[SWS_EthSM_00114] ⌈If the function `EthSM_TrcvLinkStateChg` does not report a DET error, it shall store the transceiver link state for the affected network handle. The corresponding network mode switch of the state machine shall be initiated latest in the next main function cycle.⌋

[SWS_EthSM_00115]

Upstream requirements: SRS_BSW_00406

⌈The function `EthSM_TrcvLinkStateChg` shall report ETHSM_E_UNINIT to the DET, if the EthSM is not initialized yet.⌋

8.3.2.3 EthSM_TcpIpModeIndication

[SWS_EthSM_00110] Definition of callback function EthSM_TcpIpModeIndication ⌈

Service Name	EthSM_TcpIpModeIndication	
Syntax	`void EthSM_TcpIpModeIndication ( uint8 CtrlIdx, TcpIp_StateType TcpIpState )`	
Service ID [hex]	0x08	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	EthIf controller index to identify the communication network where the TcpIp state is changed
	TcpIpState	Actual TcpIp state of the specific network handle
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This service is called by the TcpIp to report the actual TcpIp state (e.g. online, offline).	

▽

△

Available via	EthSM_TcpIp.h

⌋

[SWS_EthSM_00116] ⌈If the function `EthSM_TcpIpModeIndication` gets a CtrlIdx, which is not configured in the configuration of the EthSM module, it shall call the function `Det_ReportError` with ErrorId parameter ETHSM_E_PARAM_CONTROLLER.⌋

[SWS_EthSM_00118] ⌈If development error detection is enabled, the parameter TcpIpState shall be checked for being in the allowed range.

In case it is outside of the allowed range, the function `EthSM_TcpIpModeIndication` shall ignore the state indication and report development error `ETHSM_E_INVALID_TCP_IP_MODE` to the DET.⌋

[SWS_EthSM_00119] ⌈If the function `EthSM_TcpIpModeIndication` accepts the function call, it shall store the TcpIp state for the affected network handle and the corresponding network mode switch of the state machine shall be initiated in the next main function cycle latest.⌋

[SWS_EthSM_00120]

Upstream requirements: SRS_BSW_00406

⌈The function `EthSM_TcpIpModeIndication` shall report `ETHSM_E_UNINIT` to the DET, if the EthSM is not initialized yet.⌋

8.3.2.4 EthSM_SleepIndication

[SWS_EthSM_91004] Definition of API function EthSM_SleepIndication

Status: DRAFT
Upstream requirements: SRS_Eth_00159

⌈

Service Name	EthSM_SleepIndication (draft)	
Syntax	`void EthSM_SleepIndication ( uint8 Ctrl_Idx )`	
Service ID [hex]	0x0a	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Ctrl_Idx	Ethernet Interface Controller where the corresponding EthTrcv detect a sleep indication on the network.Sleep.
Parameters (inout)	None	
Parameters (out)	None	

▽

△

Return value	None
Description	This API is called by the EthIf and indicate that a sleep indication was detected on the network. This API is only called if the ECU is acting as a passive communication slave on the corresponding communication channel (the referenced EthTrcv of the affected EthIfTransceiver has set EthTrcvActAsSlavePassiveEnabled to TRUE). This could be used e.g. for Ethernet hardware which is compliant to the OA TC10. In this case the Ethernet hardware detect an Sleep.Indication which was triggered by a Sleep.Request of the connected link partner. **Tags:** atp.Status=draft
Available via	EthSM.h

⌋

[SWS_EthSM_00213] DRAFT

Upstream requirements: SRS_Eth_00159

⌈If the function `EthSM_SleepIndication` gets a CtrlIdx, which is not configured in the configuration of the EthSM module, it shall call the function `Det_ReportError` with ErrorId parameter ETHSM_E_PARAM_CONTROLLER, if development error reporting is enabled (see EthSMDevErrorDetect).⌋

[SWS_EthSM_00214] DRAFT

Upstream requirements: SRS_Eth_00159

⌈If the EthSM module is not initialized, when the function `EthSM_SleepIndication` is called, then the function `EthSM_SleepIndication` shall call the function `Det_ReportError` with ErrorId parameter ETHSM_E_UNINIT, if development error reporting is enabled (see EthSMDevErrorDetect).⌋

[SWS_EthSM_00215] DRAFT

Upstream requirements: SRS_Eth_00159

⌈If `EthSM_SleepIndication` is called, EthSM shall forward this call to ComM by calling `ComM_<Bus>SM_BusSleepMode` with the corresponding communication channel.⌋

8.4 Scheduled functions

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

8.4.1 EthSM_MainFunction

[SWS_EthSM_00035] Definition of scheduled function EthSM_MainFunction ⌈

Service Name	EthSM_MainFunction
Syntax	`void EthSM_MainFunction ( void )`
Service ID [hex]	0x01
Description	Cyclic Main Function which is called from the Scheduler.
Available via	SchM_EthSM.h

⌋

[SWS_EthSM_00093] ⌈The function `EthSM_MainFunction` shall be called cyclically with a fixed cycle time. The cycle time could be defined via the configuration parameter ETHSM_MAIN_FUNCTION_PERIOD.⌋

[SWS_EthSM_00197] ⌈The main function of the EthSM module shall operate the effects of the EthSM state machine, which the EthSM module shall implement for each configured network.⌋

[SWS_EthSM_00198] ⌈The EthSM shall monitor the requested and current state of the Ethernet Interface Controller. If the EthSM detects a mismatch, it shall bring the hardware back to the corresponding state. (i.e. FullCOM requires the state ETH_MODE_ACTIVE; NoCom requires ETH_MODE_DOWN)⌋

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 module.

[SWS_EthSM_91002] Definition of mandatory interfaces required by module Eth SM ⌈

API Function	*Header File*	*Description*
BswM_EthSM_CurrentState	BswM_EthSM.h	Function called by EthSM to indicate its current state.
ComM_<Bus>SM_ModeIndication	ComM.h	Indication of the actual bus mode by the corresponding Bus State Manager. ComM shall propagate the indicated state to the users with means of the RTE and BswM.

▽

△

API Function	Header File	Description
Dem_SetEventStatus	Dem.h	Called by SW-Cs or BSW modules to report monitor status information to the Dem. BSW modules calling Dem_SetEventStatus can safely ignore the return value. This API will be available only if ({Dem/Dem ConfigSet/DemEventParameter/DemEvent ReportingType} == STANDARD_REPORTING)
EthIf_SetControllerMode	EthIf.h	Enables / disables the indexed controller
TcpIp_RequestComMode	TcpIp.h	By this API service the TCP/IP stack is requested to change the TcpIp state of the communication network identified by EthIf controller index.

⌋

8.5.2 Optional Interfaces

This chapter defines all interfaces which are required to fulfill an optional functionality of the module.

[SWS_EthSM_91003] Definition of optional interfaces requested by module Eth SM ⌈

API Function	Header File	Description
ComM_<Bus>SM_BusSleepMode	ComM.h	Notification of the corresponding Bus State Manager that the actual bus mode is Bus-Sleep. Only applicable for ComM channels with ComMNm Variant set to SLAVE_ACTIVE or SLAVE_PASSIVE. E.g. LIN slaves (ComMNMVariant = SLAVE_ ACTIVE) or Ethernet channels with OA TC10 compliant Ethernet hardware which act as passive communication slave (ComMNMVariant = SLAVE_ PASSIVE and EthTrcvActAsSlavePassiveEnabled set to TRUE)
Det_ReportError	Det.h	Service to report development errors.
EthIf_GetControllerMode	EthIf.h	Obtains the state of the indexed controller
ISO15118Chrg_DataLinkIndication (draft)	ISO15118Chrg.h	This API is called by the EthSM to inform the ISO15118Chrg module about the state of the data link connection. **Tags:** atp.Status=draft

⌋

9 Sequence diagrams

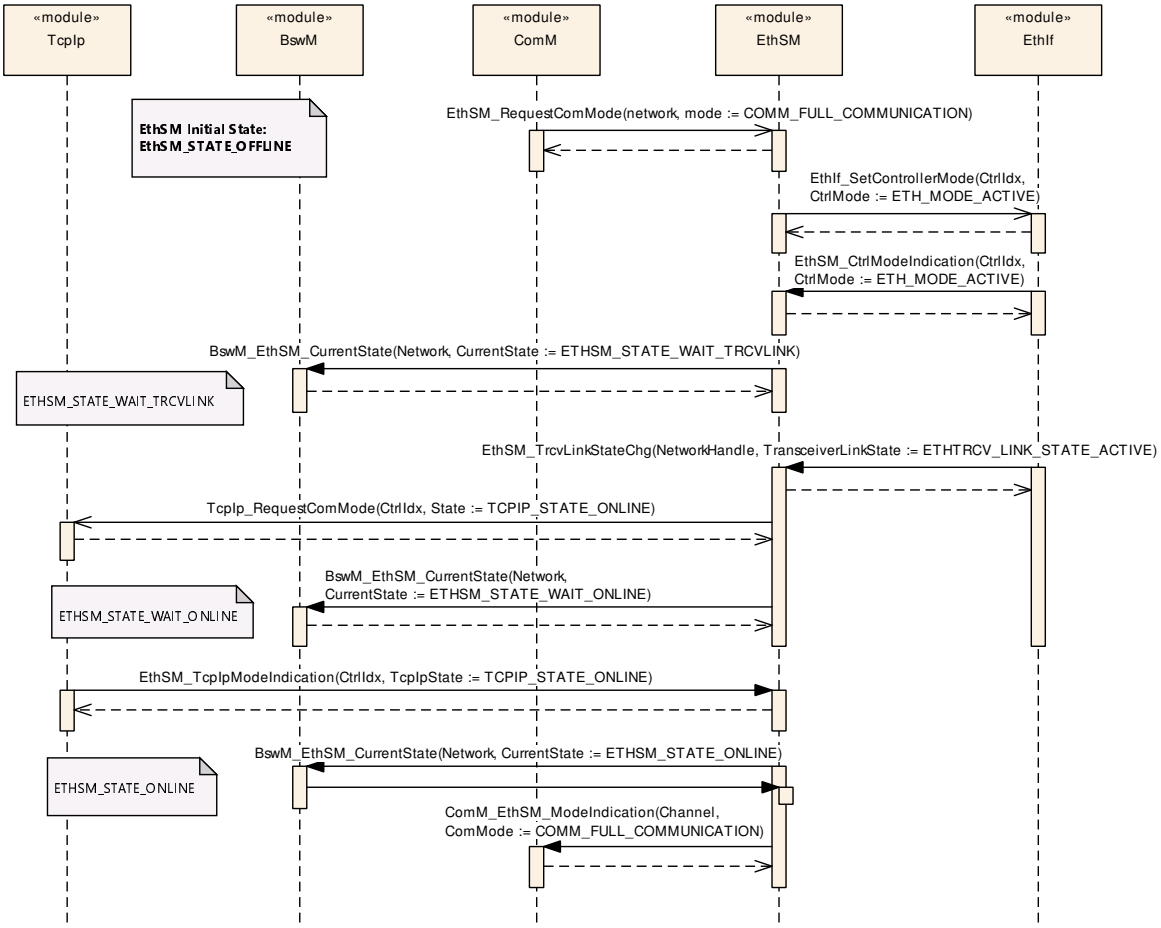

Figure 9.1: Network mode state machine - transition from no to full communication

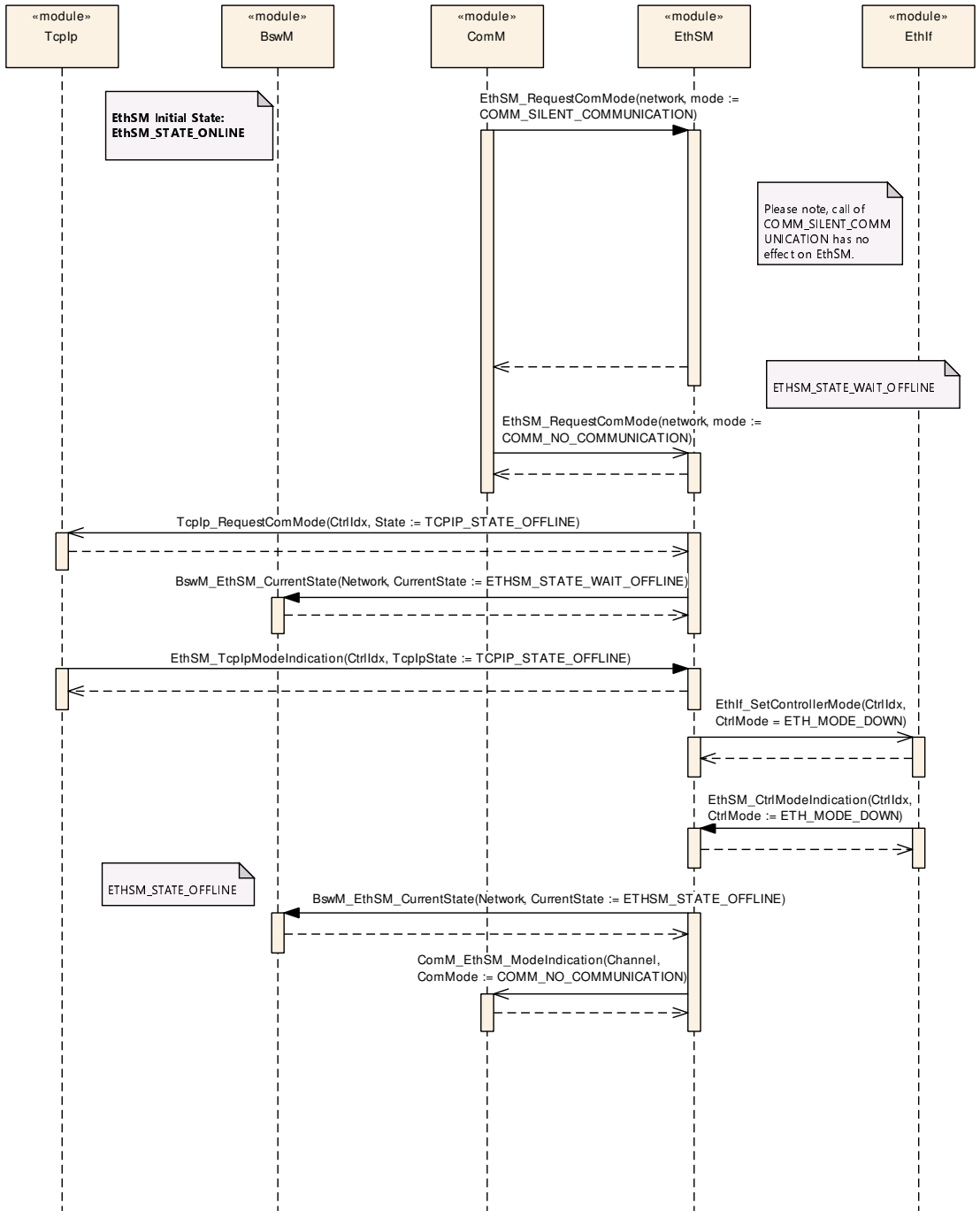

Figure 9.2: Network mode state machine - transition from full to no communication

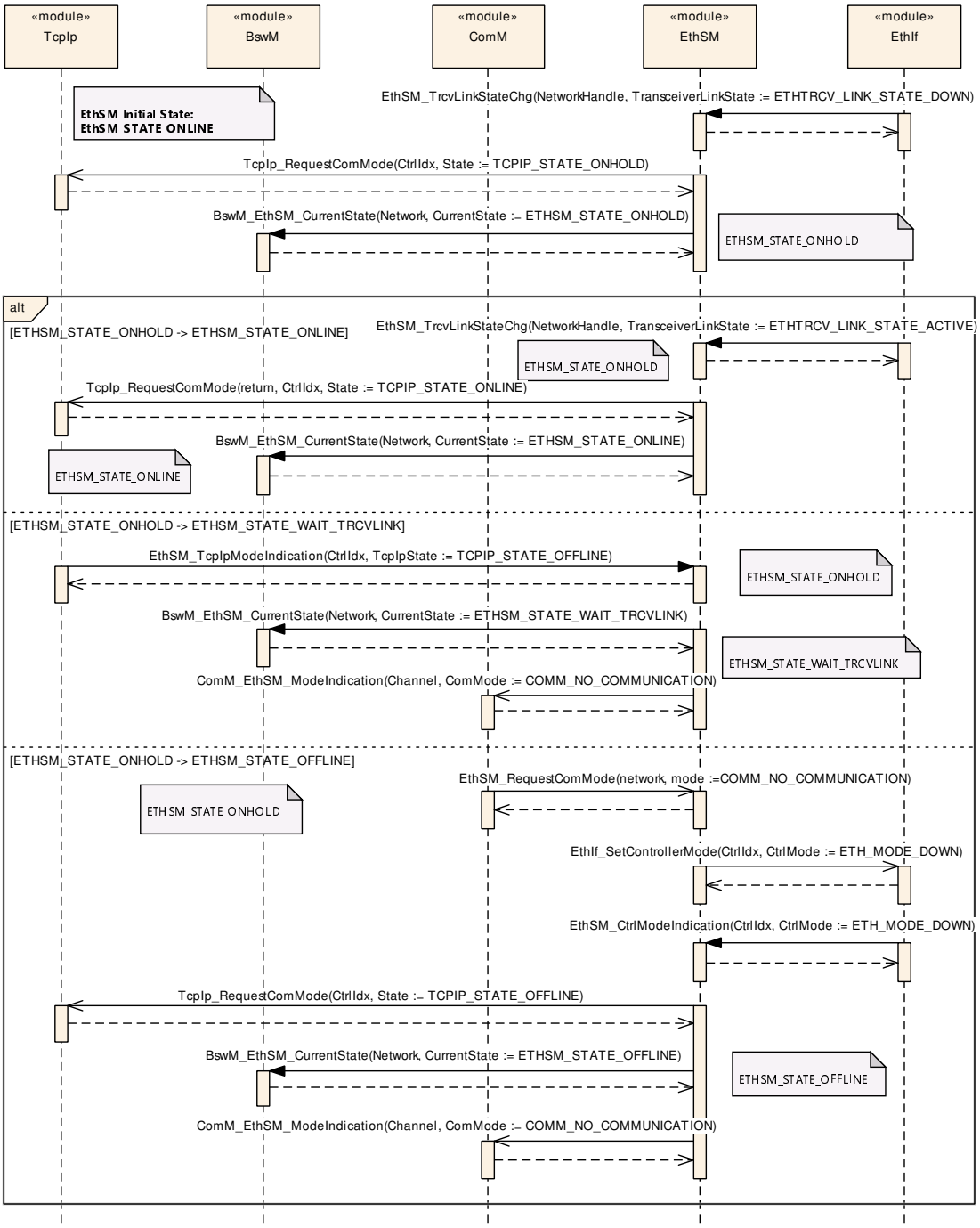

Figure 9.3: Network mode state machine - sub state ONHOLD

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 module SWS_EthernetStateManager.

Chapter 10.3 specifies published information of the module SWS_EthernetStateManager.

10.1 How to read this chapter

For details refer to [2] Chapter 10.1 *"Introduction to configuration specification"*.

10.2 Containers and configuration parameters

The following chapters summarize all configuration parameters. The detailed meanings of the parameters describe Chapter 7 and Chapter 8.

10.2.1 Configuration Tool

[SWS_EthSM_00081]

Upstream requirements: SRS_BSW_00159, SRS_BSW_00424, SRS_BSW_00425

⌈A configuration tool will create a configuration structure that is understood by the EthSM.⌋

10.2.2 EthSM

[ECUC_EthSM_00108] Definition of EcucModuleDef EthSM ⌈

Module Name	EthSM
Description	Configuration of the Ethernet State Manager
Post-Build Selectable Support	false
Post-Build Loadable Support	false

Included Containers		
Container Name	**Multiplicity**	**Description**
EthSMGeneral	1	This container contains the global parameter of the Ethernet State Manager.
EthSMNetwork	1..*	This container contains the Ethernet network-specific parameters of each Ethernet network. It also contains the reference to combination of controller and transceiver assigned to an Ethernet network.

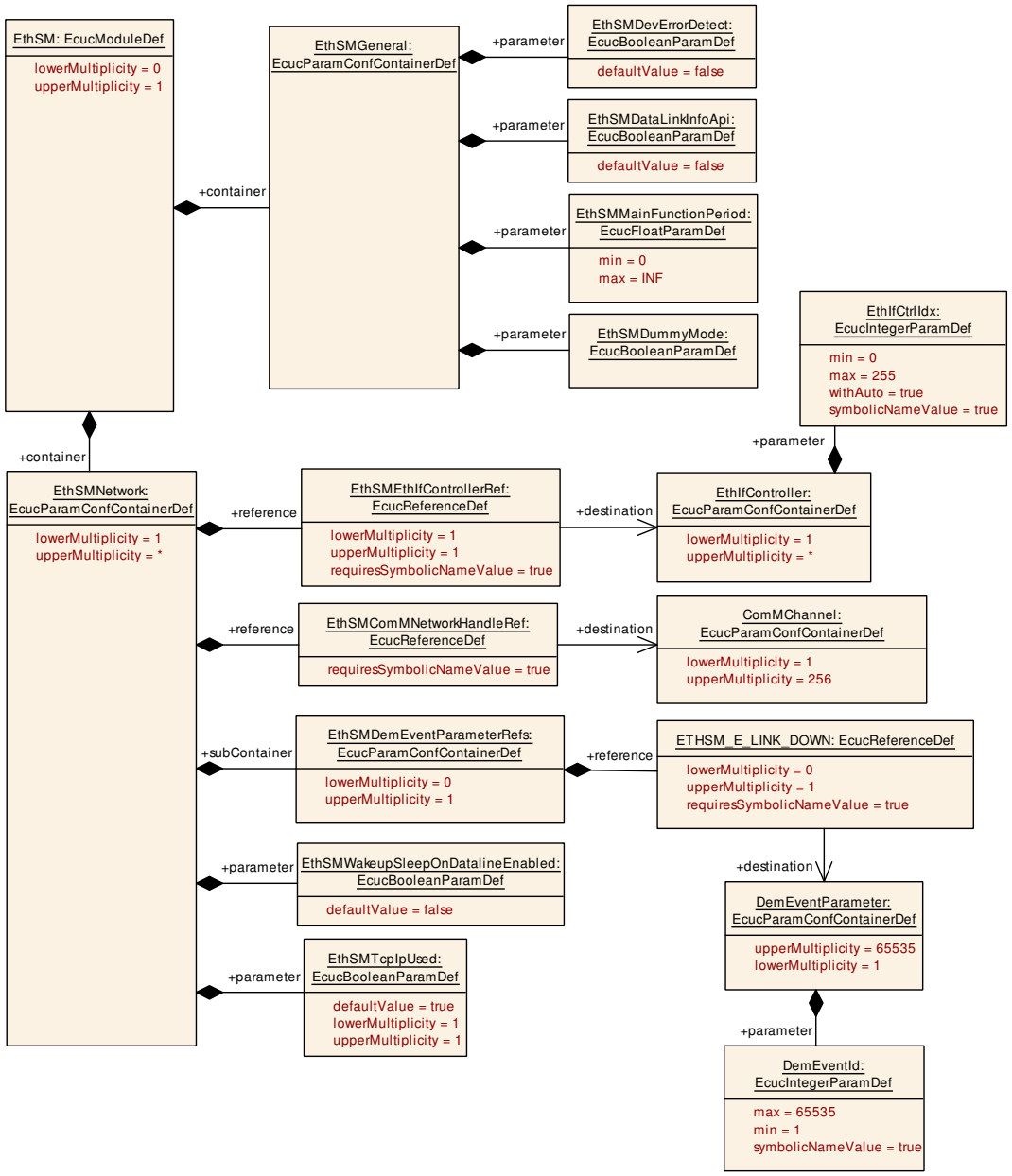

Figure 10.1

10.2.3 EthSMGeneral

[ECUC_EthSM_00063] Definition of EcucParamConfContainerDef EthSMGeneral ⌈

Container Name	EthSMGeneral
Parent Container	EthSM
Upstream mapping defined	–
Description	This container contains the global parameter of the Ethernet State Manager.
Multiplicity	1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
EthSMDataLinkInfoApi	1	[ECUC_EthSM_00113]
EthSMDevErrorDetect	1	[ECUC_EthSM_00065]
EthSMDummyMode	1	[ECUC_EthSM_00079]
EthSMMainFunctionPeriod	1	[ECUC_EthSM_00066]

No Included Containers

⌋

[ECUC_EthSM_00113] Definition of EcucBooleanParamDef EthSMDataLinkInfoApi

Status: DRAFT

⌈

Parameter Name	EthSMDataLinkInfoApi
Parent Container	EthSMGeneral
Upstream mapping defined	–
Description	Enables and disables the API: ISO15118Chrg_DataLinkIndication **Tags:** atp.Status=draft
Multiplicity	1
Type	EcucBooleanParamDef
Default value	false
Pre-Compile only	true

⌋

[ECUC_EthSM_00065] Definition of EcucBooleanParamDef EthSMDevErrorDetect ⌈

Parameter Name	EthSMDevErrorDetect
Parent Container	EthSMGeneral
Upstream mapping defined	–

▽

△

Description	Switches the development error detection and notification on or off. • true: detection and notification is enabled. • false: detection and notification is disabled.
Multiplicity	1
Type	EcucBooleanParamDef
Default value	false
Pre-Compile only	true

⌋

[ECUC_EthSM_00079] Definition of EcucBooleanParamDef EthSMDummyMode ⌈

Parameter Name	EthSMDummyMode
Parent Container	EthSMGeneral
Upstream mapping defined	–
Description	Disables the API to the EthIf. The API to the ComM is available but the functionality is deactivated. The function calls from the ComM will be answered with the return value E_OK.
Multiplicity	1
Type	EcucBooleanParamDef
Default value	–
Pre-Compile only	true

⌋

[ECUC_EthSM_00066] Definition of EcucFloatParamDef EthSMMainFunctionPeriod ⌈

Parameter Name	EthSMMainFunctionPeriod
Parent Container	EthSMGeneral
Upstream mapping defined	–
Description	Specifies the period in seconds that the MainFunction has to be triggered with.
Multiplicity	1
Type	EcucFloatParamDef
Range	]0 .. INF[
Default value	–
Pre-Compile only	true

⌋

10.2.4 EthSMNetwork

[ECUC_EthSM_00067] Definition of EcucParamConfContainerDef EthSMNetwork ⌈

Container Name	EthSMNetwork
Parent Container	EthSM
Upstream mapping defined	[3, CP_TPS_SystemTemplate]
Description	This container contains the Ethernet network-specific parameters of each Ethernet network. It also contains the reference to combination of controller and transceiver assigned to an Ethernet network.
Multiplicity	1..*
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	**Multiplicity**	**ECUC ID**
EthSMTcpIpUsed	1	[ECUC_EthSM_00112]
EthSMWakeupSleepOnDatalineEnabled	1	[ECUC_EthSM_00109]
EthSMComMNetworkHandleRef	1	[ECUC_EthSM_00068]
EthSMEthIfControllerRef	1	[ECUC_EthSM_00105]

Included Containers		
Container Name	**Multiplicity**	**Description**
EthSMDemEventParameterRefs	0..1	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId symbolic value. The standardized errors are provided in this container and can be extended by vendor-specific error references.

⌋

[ECUC_EthSM_00112] Definition of EcucBooleanParamDef EthSMTcpIpUsed ⌈

Parameter Name	EthSMTcpIpUsed
Parent Container	EthSMNetwork
Upstream mapping defined	–
Description	Specifies if the interaction with a TcpIp module is enabled or disabled for the Eth SMNetwork
Multiplicity	1
Type	EcucBooleanParamDef
Default value	true
Pre-Compile only	true

⌋

[ECUC_EthSM_00109] Definition of EcucBooleanParamDef EthSMWakeupSleepOnDatalineEnabled ⌈

Parameter Name	EthSMWakeupSleepOnDatalineEnabled
Parent Container	EthSMNetwork
Upstream mapping defined	[3, CP_TPS_SystemTemplate]

▽

△

Description	Specifies if an active communication request shall be forwarded to the lower layer to trigger a wake-up on the Ethernet network, e.g. trigger a wake-up on dataline if OA TC10 compliant Ethernet hardware is used. If the parameter is set to TRUE and EthSM in called with COMM_FULL_COMMUNICATION_WITH_WAKEUP_REQUEST, then EthSM call the corresponding EthIfController everytime with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST to trigger an wake-up request in the lower layer.
Multiplicity	1
Type	EcucBooleanParamDef
Default value	false
Pre-Compile only	true

⌋

[ECUC_EthSM_00068] Definition of EcucReferenceDef EthSMComMNetwork HandleRef ⌈

Parameter Name	EthSMComMNetworkHandleRef
Parent Container	EthSMNetwork
Upstream mapping defined	–
Description	Unique handle to identify one certain Ethernet network. Reference to one of the network handles configured for the ComM.
Multiplicity	1
Type	Symbolic name reference to ComMChannel
Pre-Compile only	true

⌋

[ECUC_EthSM_00105] Definition of EcucReferenceDef EthSMEthIfControllerRef ⌈

Parameter Name	EthSMEthIfControllerRef
Parent Container	EthSMNetwork
Upstream mapping defined	–
Description	Reference to EthIfCtrl container where a ETH controller and transceiver (optional) combination is configured.
Multiplicity	1
Type	Symbolic name reference to EthIfController
Pre-Compile only	true

⌋

10.2.5 EthSMDemEventParameterRefs

[ECUC_EthSM_00106] Definition of EcucParamConfContainerDef EthSMDem EventParameterRefs ⌈

Container Name	EthSMDemEventParameterRefs
Parent Container	EthSMNetwork
Upstream mapping defined	–
Description	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The Event Id is taken from the referenced DemEventParameter's DemEventId symbolic value. The standardized errors are provided in this container and can be extended by vendor-specific error references.
Multiplicity	0..1
Pre-Compile only	true
Configuration Parameters	

Included Parameters		
Parameter Name	Multiplicity	ECUC ID
ETHSM_E_LINK_DOWN	0..1	[ECUC_EthSM_00107]

No Included Containers

⌋

[ECUC_EthSM_00107] Definition of EcucReferenceDef ETHSM_E_LINK_DOWN ⌈

Parameter Name	ETHSM_E_LINK_DOWN
Parent Container	EthSMDemEventParameterRefs
Upstream mapping defined	–
Description	Reference to configured DEM event to report bus off errors for this Eth network.
Multiplicity	0..1
Type	Symbolic name reference to DemEventParameter
Pre-Compile only	true

⌋

10.3 Published Information

For details refer to [2] Chapter 10.3 *"Published Information"*.

A Not applicable requirements

[SWS_EthSM_NA_00999]

Upstream requirements: SRS_BSW_00344, SRS_BSW_00404, SRS_BSW_00170, SRS_BSW_00395, SRS_BSW_00398, SRS_BSW_00399, SRS_BSW_00400, SRS_BSW_00438, SRS_BSW_00375, SRS_BSW_00416, SRS_BSW_00437, SRS_BSW_00168, SRS_BSW_00423, SRS_BSW_00426, SRS_BSW_00427, SRS_BSW_00428, SRS_BSW_00429, SRS_BSW_00432, SRS_BSW_00433, SRS_BSW_00336, SRS_BSW_00369, SRS_BSW_00417

⌈These requirements are not applicable to this specification.⌋

B Change history of AUTOSAR traceable items

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

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

B.1.1 Added Specification Items in R24-11

Number	Heading
[SWS_EthSM_00226]	Transition from WAIT_ONLINE to WAIT_OFFLINE
[SWS_EthSM_00227]	During transition from WAIT_ONLINE to WAIT_OFFLINE
[SWS_EthSM_00228]	After transition from WAIT_ONLINE to WAIT_OFFLINE - state change and ComM
[SWS_EthSM_00229]	Transition from ONHOLD to WAIT_OFFLINE
[SWS_EthSM_00230]	During transition from ONHOLD to WAIT_OFFLINE
[SWS_EthSM_00255]	PASS condition for Link down detection

Table B.1: Added Specification Items in R24-11

B.1.2 Changed Specification Items in R24-11

Number	Heading
[ECUC_EthSM_00113]	Definition of EcucBooleanParamDef EthSMDataLinkInfoApi
[SWS_EthSM_00026]	Transition from OFFLINE to WAIT_TRCVLINK
[SWS_EthSM_00035]	Definition of scheduled function EthSM_MainFunction
[SWS_EthSM_00088]	
[SWS_EthSM_00114]	
[SWS_EthSM_00132]	Transition from WAIT_TRCVLINK to WAIT_ONLINE
[SWS_EthSM_00221]	Self transition WAIT_OFFLINE
[SWS_EthSM_00222]	Self transition WAIT_OFFLINE
[SWS_EthSM_00223]	Transition from WAIT_OFFLINE to ONLINE
[SWS_EthSM_00224]	
[SWS_EthSM_91001]	Definition of imported datatypes of module EthSM
[SWS_EthSM_91003]	Definition of optional interfaces requested by module EthSM

Table B.2: Changed Specification Items in R24-11

B.1.3 Deleted Specification Items in R24-11

none

B.2 Traceable item history of this document according to AUTOSAR Release R25-11

B.2.1 Added Specification Items in R25-11

none

B.2.2 Changed Specification Items in R25-11

Number	Heading
[ECUC_EthSM_00063]	Definition of EcucParamConfContainerDef EthSMGeneral
[ECUC_EthSM_00067]	Definition of EcucParamConfContainerDef EthSMNetwork
[ECUC_EthSM_00106]	Definition of EcucParamConfContainerDef EthSMDemEventParameterRefs

Table B.3: Changed Specification Items in R25-11

B.2.3 Deleted Specification Items in R25-11

none

B.3 Traceable item history of this document according to AUTOSAR Release R26-11

B.3.1 Added Specification Items in R26-11

none

B.3.2 Changed Specification Items in R26-11

Number	Heading
[SWS_EthSM_00035]	Definition of scheduled function EthSM_MainFunction
[SWS_EthSM_00216]	Transition from ONLINE to WAIT_OFFLINE
[SWS_EthSM_00218]	TCPIP to OFFLINE in transition from WAIT_ONLINE to WAIT_OFFLINE

Table B.4: Changed Specification Items in R26-11

B.3.3 Deleted Specification Items in R26-11

Number	Heading
[ECUC_EthSM_00092]	Definition of EcucBooleanParamDef EthSMVersionInfoApi
[ECUC_EthSM_00110]	Definition of EcucBooleanParamDef EthSMForwardSilentCommunication Enabled
[SWS_EthSM_00046]	Definition of API function EthSM_GetVersionInfo
[SWS_EthSM_00217]	
[SWS_EthSM_00226]	Transition from WAIT_ONLINE to WAIT_OFFLINE
[SWS_EthSM_00227]	During transition from WAIT_ONLINE to WAIT_OFFLINE
[SWS_EthSM_00228]	After transition from WAIT_ONLINE to WAIT_OFFLINE - state change and ComM
[SWS_EthSM_00229]	Transition from ONHOLD to WAIT_OFFLINE
[SWS_EthSM_00230]	During transition from ONHOLD to WAIT_OFFLINE

Table B.5: Deleted Specification Items in R26-11