Document Title	Specification of Ethernet Interface
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	417
Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R21-11

Document Change History

Date	Release	Changed by	Change Description
2021-11-25	R21-11	AUTOSAR Release Management	• Updates on 10BASE-T1S • EthernetWakeOnDataLine Specification items valid (no “Draft” tag) • Clarification on Return codes and error reporting • Updates on ReworkofPNCrelatedComM-andNMhandling COncept • Clarification on "Synchronous /Asynchronous" APIs • Removed section EthIfSwitchTimeStampIndication-Config • Removed SWS_EthIf_00248 • Updated uptraces of Security Event tables
2020-11-30	R20-11	AUTOSAR Release Management	• Introduction of Security Event reporting (DRAFT) • Introduction of EthernetWakeupOnDataline (DRAFT) • Introduction of 10BASE-T1S (DRAFT)

Document Change History

Date	Release	Changed by	Change Description
2019-11-28	R19-11	AUTOSAR Release Management	• Some empty pages removed • API Table for EthIf_MainFunctionRx_<PriorityProcessing ShortName> corrected • EthSwt_PortModeType introduced to explicitly distinguish between Port Mode and Transceiver Mode • Missing and duplicate service IDs corrected • Missing API of EthSwt and EthTrcv are added in EthIf • "BSWDistribution" (CONC_643) added as draft • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	• Explicite link control in Ethernet transceiver • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2017-12-08	4.3.1	AUTOSAR Release Management	• Improved transceiver tests (Signal quality) • Enhanced sequence charts (Ethernet switch handling)
2016-11-30	4.3.0	AUTOSAR Release Management	• Diagnostics access APIs added • gPTP Timestamp rework • Ethernet Switch enhancements (Port Groups) • Wireless Ethernet support
2015-07-31	4.2.2	AUTOSAR Release Management	• EthIf_TransceiverInit and EthIf_ControllerInit removed • Development Error Tracer renamed to Default Error Tracer
2014-10-31	4.2.1	AUTOSAR Release Management	• Change from Synchronous to Asynchronous API • gPTP Timestamp Support • Ethernet Switch Support • Ethernet Wakeup Support

Document Change History			
Date	**Release**	**Changed by**	**Change Description**
2014-03-31	4.1.3	AUTOSAR Release Management	• Extended UL_RxIndication • Editorial changes
2013-10-31	4.1.2	AUTOSAR Release Management	• Introduction of Eth_GeneralTypes.h • Support of API deviation for asynchronous implementation • Changes in API of EthIf_ProvideTxBuffer and EthIf_SetPhysAddr • Editorial changes • Removed chapter(s) on change documentation
2013-03-15	4.1.1	AUTOSAR Administration	• Remove „Commercial Off The Shelf“ use case • VLAN support • 1000MBit Ethernet support
2011-12-22	4.0.3	AUTOSAR Administration	• Description of payload data in EthIf_Cbk_RxIndication adapted
2010-09-30	3.1.5	AUTOSAR Administration	• Further post-build configurable parameters • EthIf_MainFunctionTx functional requirements improved (functionality split) • 'Instance ID' removed from Version Info (concerns EthIf_GetVersionInfo API) • Additional development error in EthIf_GetVersionInfo API
2010-02-02	3.1.4	AUTOSAR Administration	• Initial Release

Disclaimer

Table of Contents

Known Limitations

Currently, chapter 5 Dependencies to other modules does not describe the versions of dependent modules. Thus, a version check will extend the chapter.

1 Introduction and functional overview

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

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

This indicates the main task of the Ethernet Interface:
Provide to upper layers a hardware independent interface to the Ethernet Communication System comprising multiple different wired or wireless Ethernet controllers and transceivers. This interface shall be uniform for all Ethernet controllers and transceivers. Thus, the upper layers (TCP/IP, EthSM, CDD, V2x modules) may access the underlying bus system in a uniform manner.

The Ethernet Interface does not directly access the Ethernet hardware (Ethernet Communication Controller and Ethernet Transceiver) but by means of one or more hardware-specific driver modules.

[SWS_EthIf_00111]⌈
In order to access the Ethernet controller(s), the Ethernet Interface shall use one or multiple Ethernet Driver modules, which abstract the specific features and interfaces of the respective Ethernet controller(s).⌋()

[SWS_EthIf_00123]⌈
In order to access the Ethernet transceiver(s), the Ethernet Interface shall use one or multiple Ethernet Transceiver Driver modules, which abstract the specific features and interfaces of the respective Ethernet transceiver(s).⌋()

[SWS_EthIf_00228]⌈
In order to access the Ethernet switch(es), the Ethernet Interface shall use one or multiple Ethernet Switch Driver modules, which abstract the specific features and interfaces of the respective Ethernet switch(es).⌋()

[SWS_EthIf_00112]⌈
Therefore, the Ethernet Interface executable code (however, not the configuration used during runtime) shall be completely independent of the Ethernet Communication Controller(s).⌋()

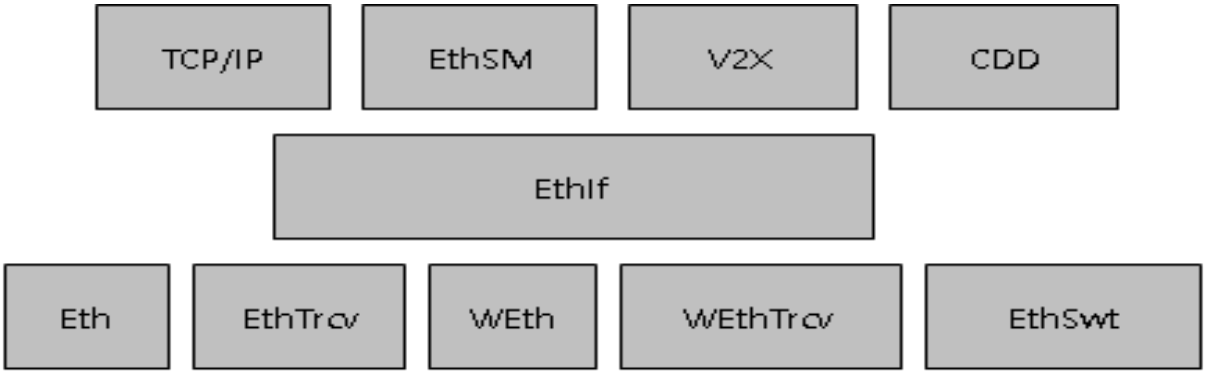

Figure 1: Ethernet stack module overview

Note: The Ethernet Intrface is specified in a way that allows for object code delivery of the code module, following the "one-fits-all" principle, i.e. the entire configuration of the Ethernet Interface can be carried out without modifying any source code. Thus, the configuration of the Ethernet Interface can be carried out largely without detailed knowledge of the underlying hardware.

2 Acronyms and abbreviations

Abbreviation / Acronym:	*Description:*
CBR	Channel Busy Ratio
CIT	Channel Idle Time
Eth	Ethernet Controller Driver (AUTOSAR BSW module)
EthIf	Ethernet Interface (AUTOSAR BSW module)
EthSM	Ethernet State Manager (AUTOSAR BSW module)
EthTrcv	Ethernet Transceiver Driver (AUTOSAR BSW module)
IP	Internet Protocol
MCG	Module Configuration Generator
MII	Media Independent Interface (standardized Interface provided by Ethernet controllers to access Ethernet transceivers)
RSSI	Received Signal Strength Indicator
TCP	Transmission Control Protocol
TCP/IP Stack	Ethernet communication stack
VLAN	Virtual Local Area Network
WEth	Wireless Ethernet Driver
WEthTrcv	Wireless Ethernet Transceiver Driver
OA TC10	Open Alliance TC10 Specification (see [25])

3 Related documentation

3.1 Input documents

[1] List of Basic Software Modules
AUTOSAR_TR_BSWModuleList.pdf

[2] Layered Software Architecture
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf

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

[4] Requirements on Ethernet Support in AUTOSAR
AUTOSAR_SRS_Ethernet.pdf

[5] Specification of Ethernet Driver
AUTOSAR_SWS_EthernetDriver.pdf

[6] Specification of Ethernet State Manager
AUTOSAR_SWS_EthernetStateManager.pdf

[7] Specification of Ethernet Transceiver Driver
AUTOSAR_SWS_EthernetTransceiver.pdf

[8] Specification of TCP/IP
AUTOSAR_SWS_TcpIp.pdf

[9] Specification of PDU Router
AUTOSAR_SWS_PDURouter.pdf

[10] BSW Scheduler Specification
AUTOSAR_SWS_Scheduler.pdf

[11] Specification of ECU Configuration
AUTOSAR_TPS_ECUConfiguration.pdf

[12] Specification of Memory Mapping
AUTOSAR_SWS_MemoryMapping.pdf

[13] Specification of Standard Types
AUTOSAR_SWS_StandardTypes.pdf

[14] Specification of Default Error Tracer
AUTOSAR_SWS_DefaulttErrorTracer.pdf

[15] Specification of Diagnostics Event Manager
AUTOSAR_SWS_DiagnosticEventManager

[16] Specification of ECU State Manager
AUTOSAR_SWS_ECUStateManager.pdf

[17] General Specification of Basic Software Modules
AUTOSAR_SWS_BSWGeneral.pdf

[18] AUTOSAR Specification of Global Time Synchronization over Ethernet
AUTOSAR_SWS_TimeSyncOverEthernet.pdf

[19] AUTOSAR Specification of Ethernet Switch Driver
AUTOSAR_SWS_EthernetSwitchDriver.pdf

[20] Wireless Ethernet Driver
AUTOSAR_SWS_WirelessEthernetDriver.pdf

[21] Wireless Ethernet Transceiver Driver
AUTOSAR_SWS_WirelessEthernetTransceiverDriver.pdf

3.2 Related standards and norms

[22] IEC 7498-1 The Basic Model, IEC Norm, 1994

[23] IEEE 802.3-2006

[24] IEEE 802.1Q-2011

[25] OPEN ALIANCE Sleep/Wake-up Specification Version 2.0 (Rel Feb 21, 2017),
http://www.opensig.org/Automotive-Ethernet-Specifications/

3.3 Related specification

AUTOSAR provides a General Specification on Basic Software modules [17] (SWS BSW General), which is also valid for Ethernet Interface.

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

4 Constraints and assumptions

4.1 Limitations

The Ethernet Interface is conceptually able to access one or more Ethernet Driver and one or more Ethernet Transceiver Driver.

It is not possible to transmit data which exceeds the available buffer size of the used Ethernet controller. Longer data has to be transmitted using the Internet Protocol (IP) or Transmission Control Protocol (TCP).

4.2 Applicability to car domains

The Ethernet BSW stack is intended to be used wherever high data rates are required but no hard real-time is required. Of course, it can also be used for less-demanding use cases, i.e. for low data rates.

5 Dependencies to other modules

This chapter lists the modules interacting with the Ethernet Interface module.

Modules that use Ethernet Interface module:

- Ethernet Communication Stack (TCP/IP Stack)
- Ethernet State Manager (EthSM)
- V2xGn

Dependencies to other Modules:

- The Ethernet Interface module doesn’t take care of configuring Ethernet Driver but requires its preceding initialization and configuration.
- The Ethernet Interface module doesn’t take care of configuring Ethernet Transceiver Driver but requires its preceding initialization and configuration.

6 Requirements traceability

Requirement	Description	Satisfied by
RS_Ids_00810	Basic SW security events	SWS_EthIf_00502, SWS_EthIf_00503
SRS_BSW_00101	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	SWS_EthIf_00304, SWS_EthIf_00306
SRS_Eth_00106	The Ethernet Transceiver Driver shall switch on/off wake up functionality at pre compile time.	SWS_EthIf_00245, SWS_EthIf_00500
SRS_Eth_00107	The Ethernet Transceiver Driver shall support access to the wake up reason.	SWS_EthIf_00486, SWS_EthIf_00490, SWS_EthIf_91004
SRS_Eth_00117	The Ethernet Transceiver Driver shall provide access to standardized hardware features	SWS_EthIf_00474, SWS_EthIf_91014, SWS_EthIf_91016, SWS_EthIf_91018, SWS_EthIf_91020, SWS_EthIf_91021, SWS_EthIf_91061
SRS_Eth_00125	The Ethernet Switch Driver shall support switch frame management	SWS_EthIf_91003, SWS_EthIf_91007
SRS_Eth_00156	The Ethernet Interface shall provide indication for a received sleep request.	SWS_EthIf_00497, SWS_EthIf_00499, SWS_EthIf_91006
SRS_Eth_00157	The Ethernet Interface shall trigger requested modes for Ethernet hardware with wake-up capability even if the requested mode has already been reached.	SWS_EthIf_00264, SWS_EthIf_00266, SWS_EthIf_00478, SWS_EthIf_00479, SWS_EthIf_00480, SWS_EthIf_00481, SWS_EthIf_00482, SWS_EthIf_00483, SWS_EthIf_00504

7 Functional specification

7.1 Ethernet BSW stack

As part of the AUTOSAR Layered Software Architecture according to [2], the Ethernet BSW modules also form a layered software stack. Figure 2 depicts the basic structure of this Ethernet BSW stack. The Ethernet Interface module accesses several Ethernet controllers using the Ethernet Driver layer, which can be made up of several Ethernet Drivers modules.

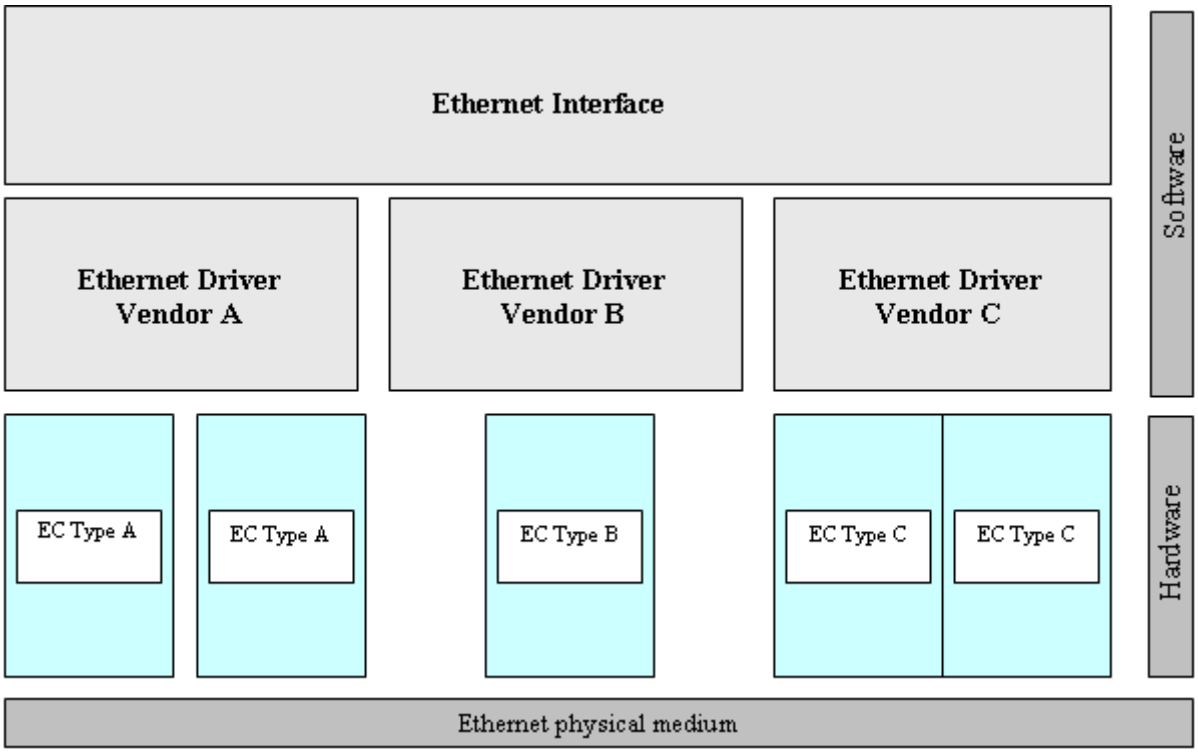

Figure 2: Basic Structure of the Ethernet BSW stack

7.1.1 Indexing scheme for Ethernet controller

Users of the Ethernet Interface identify Ethernet controller resources using an indexing scheme as depicted in Figure 3.

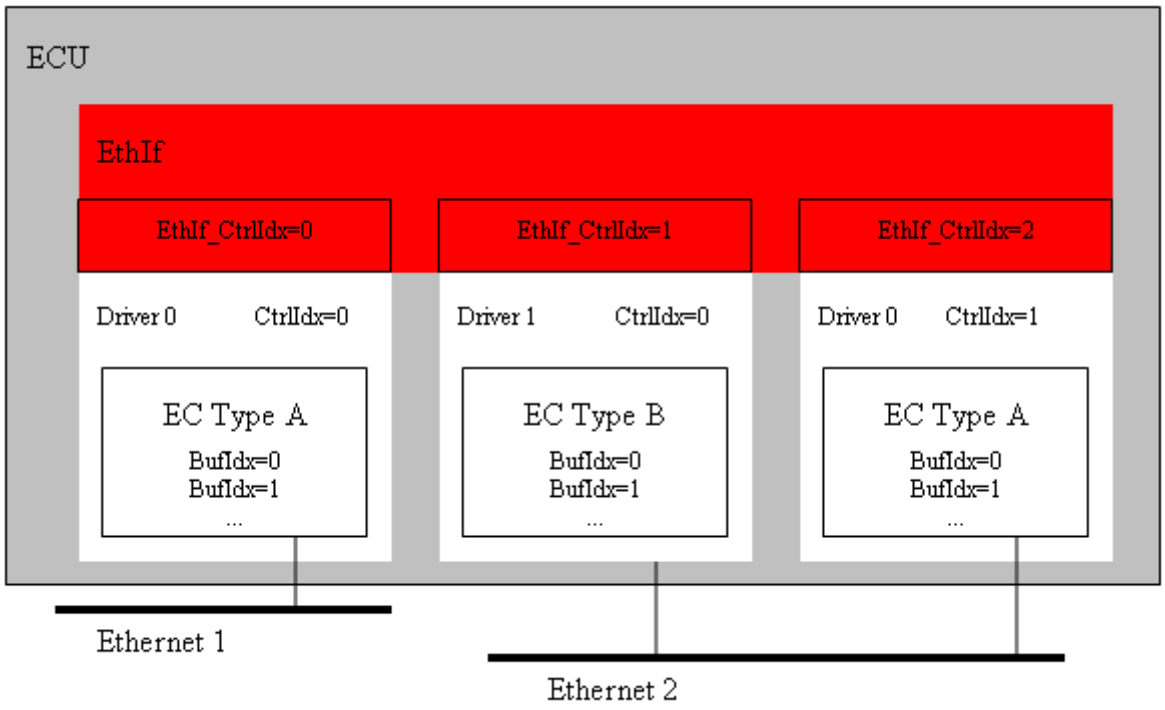

Figure 3: Ethernet Interface controller indexing scheme

[SWS_EthIf_00003] ⌈
The Ethernet Interface is using an index (EthIfCtrlIdx) to abstract the access to VLANs from the underlying communication system compromised of Ethernet Controller and Ethernet Transceiver.
Therefore the Ethernet Interface shall implement a mapping from Ethernet Interface controllers (EthIfCtrlIdx) to respective hardware ressource controllers (EthCtrlId + EthTrcvId).⌋()

7.1.2 Indexing scheme for Ethernet switches

Since the EthIf is not concerned with the individual EthSwtPorts which belong to the individual EthSwtes there is no indexing scheme for EthSwtPorts required in the EthIf. Any BSW module which interacts with EthSwtPorts can directly refer to the ECU configuration of the EthSwtPort for the indexing..

[SWS_EthIf_00224] ⌈
The EthIf shall dispatch all accesses by the EthIfSwitchIdx index to the respective EthSwt driver module with the EthSwtIdx value⌋()

7.1.3 Ethernet Interface main function

[SWS_EthIf_00004] ⌈
The Ethernet Interface shall implement main functions to be used for frame transmission confirmation and frame reception in polling mode with a calling period configurable at system configuration time.⌋()

7.1.4 Requirements

This chapter lists requirements that shall be fulfilled by Ethernet Interface module implementations.
The Ethernet Interface module environment comprises all modules which are calling interfaces of the Ethernet Interface module.

[SWS_EthIf_00005] ⌈
The Ethernet Interface module shall support pre-compile time, link time and post-build time configuration.⌋()

[SWS_EthIf_00006] ⌈
The header file *EthIf.h* shall include a software and specification version number.⌋()

[SWS_EthIf_00007] ⌈
The Ethernet Interface module shall perform a consistency check between code files and header files based on pre-process-checking the version numbers of related code files and header files.⌋()

[SWS_EthIf_00008] ⌈
In case development error detection is enabled for the Ethernet Interface module:
The Ethernet Interface module shall check API parameters for validity and report detected errors to the DET.⌋()

DET API functions are specified in [14].

[SWS_EthIf_00010] ⌈
The Ethernet Interface module shall implement the API functions specified by the Ethernet Interface SWS as real C-code functions and shall not implement the API as macros for object code deliveries.⌋()

[SWS_EthIf_00011] ⌈
None of the Ethernet Interface module header files shall define global variables.⌋()

7.1.5 Configuration description

[SWS_EthIf_00012] ⌈
The Ethernet Interface module shall provide an XML file that contains the data, which is required for the SW identification (it shall contain the vendor identification, module ID and software version information), configuration and integration process. This file should describe vendor specific configuration parameters as well as it should contain recommended configuration parameter values.⌋()

[SWS_EthIf_00117] ⌈
The MCG shall read the ECU configuration description of the Ethernet Driver and the Ethernet Interface module(s). While cluster related configuration parameters are contained in the Ethernet Interface module configuration description, Ethernet Driver related configuration data is contained in the Ethernet Driver module configuration description. The Ethernet Interface module specific configuration tool shall read both ECU module descriptions to derive the configuration data for all Ethernet Drivers mapped to the Ethernet Interface module.⌋()

[SWS_EthIf_00118] ⌈
The MCG shall ensure the consistency of the generated configuration data.⌋()

[SWS_EthIf_00013] ⌈
The configuration of the Ethernet Interface module shall be configured at ECU configuration time. None of the communication parameters shall be configured at runtime.⌋()

[SWS_EthIf_00014] ⌈
The start address of post-build time configuration data shall be passed during module initialization (see chapter 8.3.1).⌋()

An assignment of those configuration classes to configuration parameters can be found in chapter 10.

A detailed description of all Ethernet Interface related configuration parameters can be found in chapter 10 of this document. Additionally, the configuration description of the Ethernet Driver (see chapter 10 of [5]) shall be evaluated for Ethernet Interface module configuration.

7.1.6 VLAN support

[SWS_EthIf_00128] ⌈
The Ethernet Interface shall support Virtual Local Area Networks (VLAN).⌋()

[SWS_EthIf_00129] ⌈
The Ethernet Interface shall encapsulate Virtual Local Area Networks (VLAN) into virtual controllers (Ethernet Interface controller) representing a dedicated VLAN.
All BSW modules above the Ethernet Interface shall interact based on those virtual controllers.
The Ethernet Driver and Transceiver deal only with real controllers and are not aware of the existence of virtual controllers.
Caveat: the virtual controller represents the untagged VLAN if no VLAN ID is set.⌋()

[SWS_EthIf_00130] ⌈
The Ethernet Interface shall use the buffers provided by the Ethernet Driver for VLAN support.⌋()

7.1.7 Wake up support

The Ethernet Interface supports wake up depending on the parameter EthIfWakeUpSupport.

Note: Enabling wake-up support in EthIf makes only sense if the underlying EthTrcv supports also wake up.

7.1.8 Ethernet Switch Management support

Ethernet switch management enables the possibility to control an Ethernet frame regarding an Ethernet switch port specific ingress and egress handling as well as providing a Ethernet switch port specific timestamp. This functionality is essential for other BSW modules, in particular for EthTSyn, which requires Port specific information associated to a time synchronization or path-delay measurement frame.

For an introduction of the basic HW architecture and interaction, please refer to [5]. For more details regarding functional sequences, please refer to [20].

Note: Ethernet switch management API's supporting the <Upper Layer> to gather / modify Ethernet switch port specific communication attributes.

7.1.9 Handling of maintained Ethernet hardware

The Ethernet Interface handle the maintained Ethernet hardware due to its configuration:

- EthIfPhysController (representing physical Ethernet controller)
- EthIfController (representing virtual Ethernet controller to support VLANs)
- EthIfTransceiver (representing PHYs)
- EthIfSwitch (representation of an Ethernet switch)
- EthIfSwitchPortGroups (representing groups of EthSwtPorts)

At least one EthIfPhysController should be present in the configuration to interact with the Ethernet driver. EthIfController represent the connection between the physical Ethernet controller and used Ethernet hardware to communicate on and Ethernet network. This could be either an EthIfTransceiver or an EthIfSwitch or an EthIfSwitchPortGroup. If an upper layer wants to control the communication on a particular Ethernet network, it calls the corresponding EthIfController via EthIf_SetControllerMode. The Ethernet Interface handle a communication request, such that it takes care to forward the request to the corresponding Ethernet hardware:

- EthIfTransceiver
- EthIfSwitch

- EthIfSwitchPortGroup with reference of type “control”

For EthIfController with reference of type “link-information” to an EthIfSwitchPortGroup, the Ethernet Interface supervise the link state of all EthSwtPorts within a EthIfSwitchPortGroup and signal the accumulated link state to the corresponding upper layer (EthSM). Those EthIfSwitchPortGroups are controlled via a call of EthIf_SwitchPortGroupRequestMode. This is used if EthIfSwitchPortGroups are controlled according to partial network requests. Partial network requests are forwarded to BswM and a particular rule in the BswM lead to an action to control the corresponding EthIfSwitchPortGroup. Thus the upper layer of the Ethernet Interface to control the communication is EthSM and the BswM, if EthIfSwitchPortGroup switching is used. Independent if an EthIfController or an EthIfSwitchPortGroup are addressed for a communication request, the upper layer request the Ethernet Connection to be ACTIVE (ETH_MODE_ACTIVE or ETH_MODE_WITH_WAKEUP_REQUEST) or DOWN (ETH_MODE_DOWN). The Ethernet Inteface requests the corresponding lower layer to switch on the corresponding Ethernet hardware for an ACITVE-request or switch off the corresponding Ethernet Hardware for a DOWN-request.

7.1.9.1 EthIfSwitchPortGroup

The Ethernet Interface supports the grouping of Ethernet switch ports (EthIfSwitchPortGroup). The request (either ACITVE or DOWN) will be handled and rated by the Ethernet Interface. The Ethernet Interface has to decide either to put the EthifSwitchPotGroup to DOWN or ACTIVE state. ACTIVE-request for EthIfSwitchPortGroup will always overrule DOWN-request for EthIfSwitchPortGroups. If a DOWN-request for an EthIfSwitchPortGroup is ready for execution, the EthIf will check the EthSwtPorts which are referenced by the EthIfSwitchPortGroup and decide if the EthSwtPort can be set to DOWN state. If this is valid, the EthSwtPort is set to DOWN state after the configured switch off delay timer has expired.

Note: Further requirements for switching of EthIfSwitchPortGroups are available in chapter “7.1.9.2” and “8.3.21”

7.1.9.1.1 Link state accumulation of EthIfSwitchPortGroup

The Ethernet Interface need to know the actual link state of the EthIfSwitchPortGroups. The link state for an EthIfSwitchPortGroup is computed over all link states of the EthSwtPorts which are referenced by the EthIfSwitchPortGroup. The execution of the computation is called “link state accumulation” and the result is called “accumulated link state”. The accumulated link state of the EthIfSwitchPortGroup is the actual state of the EthIfSwitchPortGroup. The actual state of the EthIfSwitchPortGroup. The actual state of EthIfSwitchPortGroups referenced by an EthIfController is reported to the EthSM by calling EthSM_TrcvLinkStateChg. The actual state of EthIfSwitchPortGroups which are not referenced by any EthIfController is reported to the BswM by calling BswM_EthIf_PortGroupLinkStateChg.

[SWS_EthIf_00259] ⌈

The link state for an EthIfSwitchPortGroup is computed over all link states of the EthSwtPorts which are referenced by the EthIfSwitchPortGroup. Its status is ETHTRCV_LINK_STATE_DOWN (link down) if one of the following conditions is met:

- Referenced EthSwtPort with the role "host port" or the role "up link port" has link down state
- All referenced EthSwtPort without a role have link down state

Otherwise its accumulated link state is ETHTRCV_LINK_STATE_ACTIVE (link up).」
()

[SWS_EthIf_00260]「
If the EthIfCtrl references a EthIfSwitch but no port group is configured, the EthIf shall indicate the link state of the host port to the EthSM by calling EthSM_TrcvLinkStateChg for the EthIfController when the link state changes.」()

[SWS_EthIf_00261]「
In case a EthIfSwitchPortGroup is not connected to any EthIfController, the EthIf shall indicate the accumulated link state of the EthIfSwitchPortGroup to the BswM by calling BswM_EthIf_PortGroupLinkStateChg for the EthIfSwitchPortGroup when the link state changes (refer to SWS_EthIf_00259 for link state accumulation).」()

[SWS_EthIf_00262]「
In case a EthIfSwitchPortGroup is connected to a EthIfController, the EthIf shall indicate the accumulated link state of the EthIfSwitchPortGroup to the EthSM by calling EthSM_TrcvLinkStateChg for the EthIfController when the link state changes (refer to SWS_EthIf_00259 for link state accumulation).」()

7.1.9.2 Switching of EthIfController and the corredponding Ethernet hardware

Switching of an EthIfController is triggered via a call of EthIf_SetControllerMode. Switching of an EthIfController implicitly include the switching of the corresponding Ethernet hardware (PHY, Ethernet switch, Ethernet switch port). The Ethernet Interface interact with the lower layer via asynchronous callback notification (e.g. EthIf_TrcvModeIndication). The chapter describe the interaction of the APIs used to switch the EthIfController and the corresponding Ethernet hardware.

Note:

1. A call of the EthIf_SetControllerMode causes an asynchronous indication by calling EthIf_CtrlModeIndication, if the mode of the referenced EthIfPhysController has changed.
2. The requirements assume that Ethernet Controller (EthIfPhysControllerIdx) and the referenced Ethernet hardware (e.g. PHY, Ethernet Switch) are controlled independent from each other. For example, if ETH_MODE_ACITVE or ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST has been requested and Ethernet Controller Driver of the affected Ethernet Controller (EthIfPhysControllerIdx) has NOT indicated ETH_MODE_ACITVE yet, then those requests can be forwarded directly to the corresponding lower layers of the referenced Ethernet hardware. An implementation has to consider the following points:

- ETH_MODE_ACTIVE and ETH_MODE_DOWN are activating and de-activiating the communication capability of an Ethernet Controller, but not the control capability of connected Ethernet hardware (e.g. MDIO).
- The implementation has to ensure, that the control capabilities via an Ethernet controller are always available, if needed by the driver modules (e.g. Ethernet switch driver)

3. EthIf has to ensure that a request with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST is not overwritten by another call of EthIf_SetControllerMode with ETH_MODE_ACTIVE, if the request is deferred due to the EthIfPhysController has not already indicated ETH_MODE_ACTIVE.

[SWS_EthIf_00035] [
The function EthIf_SetControllerMode shall forward the call to function Eth_SetControllerMode of the corresponding Ethernet Controller Driver (EthIfPhysControllerIdx) with ETH_MODE_ACTIVE, if mode ETH_MODE_ACTIVE or ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST has been requested and the corresponding Ethernet Controller Driver (EthIfPhysControllerIdx) has NOT already indicated ETH_MODE_ACTIVE.]()

[SWS_EthIf_00266] [
If EthIf_SetControllerMode has been called for an EthIfController with ETH_MODE_ACTIVE and this EthIfController has a reference to an EthIfTransceiver, then EthIf shall forward the call to the following functions in the given order, if the current mode of the EthIfTransceiver is ETH_MODE_DOWN:

1. EthTrcv_SetTransceiverMode with ETH_MODE_ACTIVE
2. EthTrcv_TransceiverLinkStateRequest with ETHTRCV_LINK_STATE_ACTIVE

](SRS_Eth_00157)

[SWS_EthIf_00478] [
If EthIf_SetControllerMode has been called for an EthIfController with ETH_MODE_ACTIVE and this EthIfController has a reference to an EthIfSwitch, then EthIf shall forward the call to the following functions in the given order for all EthSwtPorts of the referenced switch if mode ETH_MODE_ACTIVE has been requested and the current EthSwtPort mode is ETH_MODE_DOWN:

1. EthSwt_SetSwitchPortMode with ETH_MODE_ACTIVE
2. EthSwt_PortLinkStateRequest with ETHTRCV_LINK_STATE_ACTIVE

](SRS_Eth_00157)

[SWS_EthIf_00264] [
If EthIf_SetControllerMode has been called for an EthIfController with ETH_MODE_ACTIVE and this EthIfController has a reference to an EthIfSwitchPortGroup of type "control", then EthIf shall forward the call to the following functions in the given order for all EthSwtPorts of the respective EthIfSwitchPortGroup if the mode ETH_MODE_ACTIVE has been requested for the first EthIfSwitchPortGroup referencing the EthSwtPort and the current EthSwtPort mode is ETH_MODE_DOWN:

1. EthSwt_SetSwitchPortMode with ETH_MODE_ACTIVE
2. EthSwt_PortLinkStateRequest with ETHTRCV_LINK_STATE_ACTIVE

](SRS_Eth_00157)

Note: EthIfController that reference EthIfSwitfhPortGroups and the reference is of type “link-information” (see ECUC_EthIf_00048, then those EthIfSwitchPortGroups could be switched according to PNC states via a dedicatd rules in the BswM. The BswM rule can be configured via the BswMEthIfSwitchPortGroupRequestMode action. The BswM call the API EthIf_SwtichPortGroupRequestMode to switch the corresponding EthIfSwitchPortGroup.

[SWS_EthIf_00272] [
If EthIf_SwitchPortGroupRequestMode has been called with ETH_MODE_ACTIVE, EthIf shall forward the call to the following functions in the given order for all EthSwtPorts of the respective EthIfSwitchPortGroup:

1.) Call EthSwt_SetSwitchPortMode with ETH_MODE_ACTIVE, if the current mode is ETH_MODE_DOWN.

2.) Call EthSwt_PortLinkStateRequest with ETHTRCV_LINK_STATE_ACTIVE, if the current link state is ETHTRCV_LINK_STATE_DOWN

]()

[SWS_EthIf_00479] [
Everytime EthIf_SetControllerMode has been called for an EthIfController with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST and this EthIfController has a reference to an EthIfTransceiver, then EthIf shall forward the call to the following functions in the given order, independent of the current mode:

1. EthTrcv_SetTransceiverMode with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST
2. EthTrcv_TransceiverLinkStateRequest with ETHTRCV_LINK_STATE_ACTIVE, only if the current state is ETHTRCV_LINK_STATE_DOWN

](SRS_Eth_00157)

[SWS_EthIf_00480] [
Everytime EthIf_SetControllerMode has been called for an EthIfController with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST and this EthIfController has a reference to an EthIfSwitch, then EthIf shall forward the call to the following functions in the given order for all EthSwtPorts of the respective EthIfSwitchPortGroup, independ of the current mode:

1. EthSwt_SetSwitchPortMode with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST
2. EthSwt_PortLinkStateRequest with ETHTRCV_LINK_STATE_ACTIVE, if the current mode is ETHTRCV_LINK_STATE_DOWN

](SRS_Eth_00157)

[SWS_EthIf_00481] [
Everytime EthIf_SetControllerMode has been called for an EthIfController with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST and this EthIfController has a reference to an EthIfSwitchPortGroup of type "control", then EthIf shall forward the call to the following functions in the given order for all EthSwtPorts of the respective EthIfSwitchPortGroup, independent of the current mode:

1. EthSwt_SetSwitchPortMode with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST
2. EthSwt_PortLinkStateRequest with ETHTRCV_LINK_STATE_ACTIVE, if the current mode is ETHTRCV_LINK_STATE_DOWN

](SRS_Eth_00157)

[SWS_EthIf_00482] [
Everytime EthIf_SwitchPortGroupRequestMode has been called with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST, EthIf shall forward the call for all EthSwtPorts of the respective EthIfSwitchPortGroup to the following functions in the given order independent of the current EthSwtPort mode:

1. EthSwt_SetSwitchPortMode with ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST
2. EthSwt_PortLinkStateRequest with ETHTRCV_LINK_STATE_ACTIVE, only if current link state is ETHTRCV_LINK_STATE_DOWN

](SRS_Eth_00157)

Rational for SWS_EthIf_00479, SWS_EthIf_00480, SWS_EthIf_00481 and SWS_EthIf_00482: A wake-up request has always to be forwarded to the lower layer independent of the current mode to ensure that a wake-up is triggered on the network. This could be used for e.g. communication channels where the Ethernet hardware is compliant to OA TC10 (see [25])

[SWS_EthIf_00483] [
If EthIf_SwitchPortGroupRequestMode is called with ETH_MODE_ACTIVE or ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST, then a running timer to delay the switch off all ports of the respective EthIfSwitchPortGroup (PortGroupIdx) shall be canceled.](SRS_Eth_00157)

[SWS_EthIf_00263] [
EthIf shall call the function Eth_SetControllerMode of the corresponding Ethernet Controller Driver (EthIfPhysControllerIdx) with ETH_MODE_DOWN, if EthIf_SetControllerMode has been called with mode ETH_MODE_DOWN for all Ethernet Interface Controller referencing the Ethernet Controller.]()

Note:

1. In case of VLAN support, EthIf has to store internally the state of each EthIfController in order to filter out the requests from upper layers and disable the callouts to upper layers when the EthIfController is disabled.

[SWS_EthIf_00484] [
If EthIf_SetControllerMode is called for an EthIfController with ETH_MODE_DOWN and this EthIfController has a reference to an EthIfTransceiver, then EthIf shall forward the call to the following functions in the given order, if the current mode of the EthIfTransceiver is ETH_MODE_ACTIVE:

1. EthTrcv_SetTransceiverMode with ETH_MODE_DOWN
2. EthTrcv_TransceiverLinkStateRequest with ETHTRCV_LINK_STATE_DOWN

]()

[SWS_EthIf_00485] [

If EthIf_SetControllerMode is called for an EthIfController with ETH_MODE_DOWN and this EthIfController has a reference to an EthIfSwitch, then EthIf shall forward the call to the following functions in the given order for all EthSwtPorts, where the current mode of the EthSwtPort is ETH_MODE_ACTIVE:

1. EthSwt_PortLinkStateRequest with ETHTRCV_LINK_STATE_DOWN
2. EthSwt_SetSwitchPortMode with ETH_MODE_DOWN

](

[SWS_EthIf_00265][
If EthIf_SetControllerMode is called for an EthIfController with ETH_MODE_DOWN and this EthIfController has a reference to an EthIfSwitchPortGroup of type "control", then EthIf shall forward the call to the following functions in the given order for all EthSwtPorts of the respective EthIf_SwitchPortGroup, but only for those EthSwtPorts where all referencing EthIfSwitchPortGroups has been requested with ETH_MODE_DOWN and the current mode of the EthSwtPort is ETH_MODE_ACTIVE:

1. EthSwt_PortLinkStateRequest with ETHTRCV_LINK_STATE_DOWN
2. EthSwt_SetSwitchPortMode with ETH_MODE_DOWN

](

Rationale: In case the respective EthIfController has no reference to an EthIf_SwitchPortGroup or the reference is of type "link information" the requested modes are not forwarded. This EthIf_SwitchPortGroups will be requested by an upper layer (e.g. BswM) with API EthIf_SwitchPortGroupRequestMode.

7.1.9.3 Additional Ethernet switch port handling

The following additional Ethernet switch port handling has been introduced to support a use case for a passive wake up of an ECU where all Ethernet switch ports of the corresponding Ethernet switches shall be switched on immediately. E.g. after a wakeup occurred. Afterwards it is checked if a PN request is received via NM frames within EthIfPortStartupActiveTime. If a PN request is received, then the corresponding EthIfSwitchPortGroups are requested with ETH_MODE_ACTIVE and corresponding Ethernet switch ports stay active. All Ethernet switch ports where the corresponding EthIfSwitchPortGroups are not requested (due to no according PN request received within EthIfPortStartupActiveTime) are switched off.

[SWS_EthIf_00275] [
If EthIf_StartAllPorts has been called, then EthIf shall forward the call to the following functions in the given order to all EthSwtPorts of the affected EthIfSwitches:

1. Call EthSwt_SetSwitchPortMode with ETH_MODE_ACTIVE, if the current mode is ETH_MODE_DOWN.
2. Call EthSwt_PortLinkStateRequest with ETHTRCV_LINK_STATE_ACTIVE, if the current link state is ETHTRCV_LINK_STATE_DOWN

and start a timer with EthIfPortStartupActiveTime for all these ports.
](

[SWS_EthIf_00276] [
After EthIf_StartAllPorts has been called, EthIf shall deactivate all those ports activated due to EthIf_StartAllPorts (see SWS_EthIf_00275) which are not requested

with ETH_MODE_ACTIVE within EthIfPortStartupActiveTime by calling the following functions in the given order:

1. EthSwt_PortLinkStateRequest with ETHTRCV_LINK_STATE_DOWN
2. EthSwt_SetSwitchPortMode with ETH_MODE_DOWN

]()

Rational: Delaying with EthIfPortStartTime is needed to ensure that NM messages with PNC information are received and the requested PNCs are activated.

Note:

1. EthIf_StartAllPorts could be called in context of BswM_EcuM_CurrentWakeup. After a wakeup occurred on the wakeup line, all EthIfSwitchPortgroups shall be activated to enable communication stack to receive NM messages (PNC information). With this it is possible to start the EthIfSwitchPortGroups without starting a PNC.
2. Further requirements for switching of EthSwtPorts, if an EthIfController referencing an EthIfSwitch are available in chapter “7.1.9.2”.

7.1.10 Communication control

The Ethernet Interface has to provide a kind of communication control to support the so-called “silent communication”. Silent communication is used for mode management to support a communication mode where the transmission path for a particular EthIfController is disabled, while the reception path is still enabled (see COMM_SILENT_COMMUNICATION). Disabling of the transmission path is exclusively introduced in the Ethernet Interface and has no impact on the used Ethernet hardware.

[SWS_EthIf_00504] DRAFT [If EthIf_SetControllerMode is called for an EthIfController with ETH_MODE_ACTIVE_TX_OFFLINE and the latest accepted controller mode for this EthIfController is ETH_MODE_ACTIVE or ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST, then ETH_MODE_TX_OFFLINE shall be stored as current controller mode. Otherwise the requested controller mode shall be rejected and function shall return with E_NOT_OK.](SRS_Eth_00157)

Note: The transmission related APIs (see SWS_EthIf_00075 and SWS_EthIf_00067) will only forward transmission requests, if the stored communication mode is ETH_MODE_ACTIVE or ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST.

7.1.11 Global Time support

For more details regarding time measurement with Switches, please refer to [19].

7.1.12 Wireless Ethernet Support

[SWS_EthIf_00340]⌈

The Ethernet Interface shall support Wireless Ethernet specific functionality, depending on the parameter EthIfEnableWEthApi.⌋ ( )

The Wireless functions are divided in controller and transceiver specific functionality. Mainly, transmission and reception parameters are being exchanged with the EthIf upper module and the controller/transceiver.
The controller is being called only for buffer specific transmission and reception parameters by the APIs:

- EthIf_GetBufWRxParams
- EthIf_GetBufWTxParams
- EthIf_SetBufWTxParams

The Transceiver is being called for general configuration of the wireless radio and the wireless radio's channel by:

- EthIf_SetRadioParams
- EthIf_SetChanRxParams
- EthIf_SetChanTxParams
- EthIf_GetChanRxParams

The parameter values are requested or transmitted by unique parameter identifiers. They are defined within the controller and transceiver specification [20] [21].

7.2 Security Events

[SWS_EthIf_00502] DRAFT⌈ If security event reporting has been enabled for the EthIf module (EthIfEnableSecurityEventReporting = true) the respective security events shall be reported to the IdsM via the interfaces defined in AUTOSAR_SWS_BSWGeneral. ⌋(RS_Ids_00810)

The following table lists the security events which are standardized for the EthIf together with their trigger conditions:
[SWS_EthIf_00503]⌈

Name	***Description***	***ID***
ETHIF_SEV_DROP_ UNKNOWN_ETHERTYPE	An ethernet datagram was dropped due the Ethertype in not known.	15
ETHIF_SEV_DROP_VLAN_ DOUBLE_TAG	An ethernet datagram was dropped due to double VLAN tag.	16
ETHIF_SEV_DROP_INV_VLAN	An ethernet datagram was dropped due to an invalid CrtlIdx/ VLAN.	17
ETHIF_SEV_DROP_ETH_ MAC_COLLISION	Ethernet datagram was dropped because local MAC was same as source MAC in an incoming frame.	18

⌋(RS_Ids_00810)

Context data is not provided by the EthIf for the security events.

7.3 Error classification

7.3.1 Development Errors

[SWS_EthIf_00017]⌈

Type of error	*Related error code*	*Error value*
API service called with invalid controller index	ETHIF_E_INV_CTRL_IDX	0x01
API service called with invalid transceiver index	ETHIF_E_INV_TRCV_IDX	0x02
API service called with invalid switch index	ETHIF_E_INV_SWT_IDX	0x03
API service called with invalid port group index	ETHIF_E_INV_PORT_GROUP_IDX	0x04
API service called when EthIf module was not initialized	ETHIF_E_UNINIT	0x05
API service called with invalid pointer in parameter list	ETHIF_E_PARAM_POINTER	0x06
API service called with invalid parameter	ETHIF_E_INV_PARAM	0x07
EthIf_Init called with an invalid configuration pointer	ETHIF_E_INIT_FAILED	0x08
Invalid port index	ETHIF_E_INV_PORT_IDX	0x09

⌋()

7.3.2 Runtime Errors

There are no runtime errors.

7.3.3 Transient Faults

There are no transient faults.

7.3.4 Production Errors

There are no production errors.

7.3.5 Extended Production Errors

There are no extended production errors.

8 API specification

8.1 Imported types

This chapter lists all types included from the following module:

[SWS_EthIf_00023][

Module	*Header File*	*Imported Type*
ComStack_Types	ComStack_Types.h	BufReq_ReturnType
EcuM	EcuM.h	EcuM_WakeupSourceType
Eth	Eth.h	Eth_SpiStatusType (draft)
	Eth_GeneralTypes.h	Eth_BufIdxType
	Eth_GeneralTypes.h	Eth_CounterType
	Eth_GeneralTypes.h	Eth_DataType
	Eth_GeneralTypes.h	Eth_FilterActionType
	Eth_GeneralTypes.h	Eth_FrameType
	Eth_GeneralTypes.h	Eth_MacVlanType
	Eth_GeneralTypes.h	Eth_ModeType (draft)
	Eth_GeneralTypes.h	Eth_RxStatsType
	Eth_GeneralTypes.h	Eth_RxStatusType
	Eth_GeneralTypes.h	Eth_TimeStampQualType
	Eth_GeneralTypes.h	Eth_TimeStampType
	Eth_GeneralTypes.h	Eth_TxErrorCounterValuesType
	Eth_GeneralTypes.h	Eth_TxStatsType
EthSwt	Eth_GeneralTypes.h	EthSwt_MacLearningType
	Eth_GeneralTypes.h	EthSwt_MgmtInfoType
	Eth_GeneralTypes.h	EthSwt_MgmtObjectType
	Eth_GeneralTypes.h	EthSwt_MgmtObjectValidType
	Eth_GeneralTypes.h	EthSwt_MgmtOwner
	Eth_GeneralTypes.h	EthSwt_PortMirrorCfgType
	Eth_GeneralTypes.h	EthSwt_PortMirrorStateType
EthTrcv	Eth_GeneralTypes.h	EthTrcv_BaudRateType

	Eth_GeneralTypes.h	EthTrcv_CableDiagResultType
	Eth_GeneralTypes.h	EthTrcv_DuplexModeType
	Eth_GeneralTypes.h	EthTrcv_LinkStateType
	Eth_GeneralTypes.h	EthTrcv_MacMethodType (draft)
	Eth_GeneralTypes.h	EthTrcv_PhyLoopbackModeType
	Eth_GeneralTypes.h	EthTrcv_PhyTestModeType
	Eth_GeneralTypes.h	EthTrcv_PhyTxModeType
	Eth_GeneralTypes.h	EthTrcv_WakeupModeType (obsolete)
	Eth_GeneralTypes.h	EthTrcv_WakeupReasonType
IdsM	IdsM_Types.h	IdsM_SecurityEventIdType
Std	Std_Types.h	Std_ReturnType
	Std_Types.h	Std_VersionInfoType
WEth	WEth_GeneralTypes.h	WEth_BufWRxParamIdType
	WEth_GeneralTypes.h	WEth_BufWTxParamIdType
WEthTrcv	WEth_GeneralTypes.h	WEthTrcv_GetChanRxParamIdType
	WEth_GeneralTypes.h	WEthTrcv_SetChanRxParamIdType
	WEth_GeneralTypes.h	WEthTrcv_SetChanTxParamIdType
	WEth_GeneralTypes.h	WEthTrcv_SetRadioParamIdType

]()

8.2 Type definitions

8.2.1 EthIf_ConfigType

[SWS_EthIf_00149][

Name	EthIf_ConfigType
Kind	Structure
Description	Implementation specific structure of the post build configuration
Available via	EthIf.h

]()

8.2.2 EthIf_SwitchPortGroupIdxType

[SWS_EthIf_91101][

Name	EthIf_SwitchPortGroupIdxType		
Kind	Type		
Derived from	uint8		
Range	0..255	--	--
Description	Data Type that represents the Ethernet interface switch port group index. The index is zero based and unique for every configured switch port group.		
Available via	EthIf.h		

]()

8.2.3 EthIf_MeasurementIdxType

[SWS_EthIf_91010][

Name	EthIf_MeasurementIdxType		
Kind	Type		
Derived from	uint8		
Range	ETHIF_MEAS_DROP_CRTLIDX	0x01	Measurement index of dropped datagrams caused by invalid CrtlIdx/VLAN
	ETHIF_MEAS_RESERVED_1	0x02-0x7F	reserved by AUTOSAR
	ETHIF_MEAS_RESERVED_2	0x80-0xEF	Vendor specific range
	ETHIF_MEAS_RESERVED_3	0xF0-0xFE	reserved by AUTOSAR (future use)
	ETHIF_MEAS_ALL	0xFF	represents all measurement indexes
Description	Index to select specific measurement data		
Available via	EthIf.h		

]()

8.2.4 EthIf_SignalQualityResultType

[SWS_EthIf_91057][

Name	EthIf_SignalQualityResultType

Kind	Structure	
Elements	HighestSignalQuality	
	Type	uint32
	Comment	the highest signal quality of a link since last clear
	LowestSignalQuality	
	Type	uint32
	Comment	the lowest link signal quality of a link since last clear
	ActualSignalQuality	
	Type	uint32
	Comment	the actual signal quality
Description	--	
Available via	EthIf.h	

]()

8.3 Function definitions

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

Note: All functions in this chapter requires previous initialization (EthIf_Init), except the following ones: EthIf_Init, EthIf_GetVersionInfo

8.3.1 EthIf_Init

[SWS_EthIf_00024][

Service Name	EthIf_Init	

Syntax	`void EthIf_Init (`
	`  const EthIf_ConfigType* CfgPtr`
	`)`	

Service ID [hex]	0x01	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CfgPtr	Points to the implementation specific structure
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	

Description	Initializes the Ethernet Interface
Available via	EthIf.h

]()

[SWS_EthIf_00025] [
The function shall store the access to the configuration structure for subsequent API calls.]()

[SWS_EthIf_00114] [
The function shall change the state of the component from uninitialized to initialized.]()

[SWS_EthIf_00116] [
If development error detection is enabled: the function shall check the parameter CfgPtr for containing a valid configuration. If the check fails, the function shall raise the development error ETHIF_E_INIT_FAILED.]()

8.3.2 EthIf_SetControllerMode

[SWS_EthIf_00034][

Service Name	EthIf_SetControllerMode	
Syntax	`Std_ReturnType EthIf_SetControllerMode ( uint8 CtrlIdx, Eth_ModeType CtrlMode )`	
Service ID [hex]	0x03	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	CtrlMode	ETH_MODE_DOWN: disable the controller ETH_MODE_ACTIVE: enable the controller ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST: enable the controller and request a wake-up on the network. ETH_MODE_TX_OFFLINE: disable transmission handling in EthIf. Please note, the according Ethernet controller is not affected
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_-ReturnType	E_OK: success E_NOT_OK: controller mode could not be changed

Description	Enables / disables the indexed controller
Available via	EthIf.h

](

Note: Further requirements regarding the call of EthIf_SetControllerMode are described in chapter “7.1.9.2” and “7.1.10”

[SWS_EthIf_00036] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00037] ⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

8.3.3 EthIf_GetControllerMode

[SWS_EthIf_00039]⌈

Service Name	EthIf_GetControllerMode	
Syntax	`Std_ReturnType EthIf_GetControllerMode ( uint8 CtrlIdx, Eth_ModeType* CtrlModePtr )`	
Service ID [hex]	0x04	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
Parameters (inout)	None	
Parameters (out)	CtrlModePtr	ETH_MODE_DOWN: the controller is disabled ETH_MODE_ ACTIVE: the controller is enabled
Return value	Std_Return-Type	E_OK: success E_NOT_OK: controller could not be initialized
Description	Obtains the state of the indexed controller	
Available via	EthIf.h	

](

[SWS_EthIf_00040] ⌈

The function EthIf_GetControllerMode shall forward the call to function Eth_GetControllerMode of the corresponding Ethernet Controller Driver (EthIfPhysControllerIdx).](SWS_EthIf_00041)

[SWS_EthIf_00041] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00042] ⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00043] ⌈
If development error detection is enabled: the function shall check the parameter CtrlModePtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.4 EthIf_CheckWakeup

[SWS_EthIf_00244]⌈

Service Name	EthIf_CheckWakeup	
Syntax	`Std_ReturnType EthIf_CheckWakeup ( EcuM_WakeupSourceType WakeupSource )`	
Service ID [hex]	0x30	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	Wakeup Source	Source device which initiated the wake up event. The source device could either be a Ethernet switch or a Ethernet transceiver
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK when the request to check for a wake-up of the affected Ethernet hardware (e.g. PHY) has been accepted. E_NOT_OK when the request to check for a wake-up of the affected Ethernet hardware is rejected.
Description	This API request the affected Ethernet hardware to check for a signaled wake-up. The used Ethernet hardware could be an Ethernet switch or Ethernet transceiver (PHY). This is used e.g. for Ethernet hardware which is compliant to the specification of Open Alliance TC10. This API is called by the integration code. The function could be called in context of the interrupt or on task level.	

Available via	EthIf.h

]()

[SWS_EthIf_00245] [
For all affected Ethernet transceiver (either referenced by EthIfTransceiver or by EthIfSwitchPortGroups) the function EthIf_CheckWakeup shall forward the call to function EthTrcv_CheckWakeup of the respective Ethernet Transceiver Driver.](SRS_Eth_00106)

[SWS_EthIf_00500] [
For all affected Ethernet switches (referenced by EthIfSwitch) the function EthIf_CheckWakeup shall forward the call to function EthSwt_SwitchCheckWakeup of the respective Ethernet Switch Driver.](SRS_Eth_00106)

[SWS_EthIf_00246] [
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.]()

[SWS_EthIf_00247] [
If development error detection is enabled: the function shall check the parameter WakeupSource for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_PARAM.]()

8.3.5 EthIf_GetPhyWakeupReason

[SWS_EthIf_91004][

Service Name	EthIf_GetPhyWakeupReason	
Syntax	`Std_ReturnType EthIf_GetPhyWakeupReason ( uint8 TrcvIdx, EthTrcv_WakeupReasonType* WakeupReasonPtr )`	
Service ID [hex]	0x69	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	TrcvIdx	Index of the transceiver within the context of the Ethernet Interface
Parameters (inout)	None	
Parameters (out)	Wakeup ReasonPtr	Pointer to structure of least recent wakeup event, which was detected by the Ethernet PHY
Return value	Std_Return-Type	E_OK: PHY wake up reason request has been accepted.

		E_NOT_OK: PHY wake up reason request has not been accepted.
Description	This function obtains the wake up reasons of the indexed Ethernet Transceiver (PHY) by calling EthTrcv_GetBusWuReason(...)	
Available via	EthIf.h	

](SRS_Eth_00107)

[SWS_EthIf_00486] ⌈
The function EthIf_GetPhyWakeupReason shall forward the call to function EthTrcv_GetBusWuReason of the corresponding Ethernet Transceiver Driver (TrcvIdx).](SRS_Eth_00107)

[SWS_EthIf_00487] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00488] ⌈
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX.⌋()

[SWS_EthIf_00489] ⌈
If development error detection is enabled: the function shall check the parameter WakeupReasonPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.6 EthIf_GetSwitchPortWakeupReason

[SWS_EthIf_91005]⌈

Service Name	EthIf_GetSwitchPortWakeupReason	
Syntax	`Std_ReturnType EthIf_GetSwitchPortWakeupReason ( uint8 SwitchIdx, uint8 SwitchPortIdx, EthTrcv_WakeupReasonType* WakeupReasonPtr )`	
Service ID [hex]	0x67	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	SwitchIdx	Index of the Ethernet switch within the context of the Ethernet Interface
	SwitchPortIdx	Index of the Ethernet switch port index in the context of the Ethernet switch driver

Parameters (inout)	None	
Parameters (out)	Wakeup ReasonPtr	Pointer to structure of least recent wakeup event, which was detected by the Ethernet switch port
Return value	Std_Return-Type	E_OK: Ethernet switch port wake up reason request has been accepted. E_NOT_OK: Ethernet switch port wake up reason request has not been accepted.
Description	This function obtains the wake up reasons of the indexed Ethernet switch port by calling EthSwt_GetSwitchPortWakeupReason().	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00490] ⌈
The function EthIf_GetSwitchPortWakeupReason shall forward the call to function EthSwt_GetSwitchPortWakeupReason of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).⌋(SRS_Eth_00107)

[SWS_EthIf_00491] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT otherwise (if DET is disabled) return E_NOT_OK.⌋()

[SWS_EthIf_00492] ⌈
If development error detection is enabled: the function shall check the parameter SwitchIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_SWT_IDX otherwise (if DET is disabled) return E_NOT_OK.⌋()

[SWS_EthIf_00493] ⌈
If development error detection is enabled: the function shall check the parameter SwitchPortIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_PORT_IDX otherwise (if DET is disabled) return E_NOT_OK.⌋()

[SWS_EthIf_00494] ⌈
If development error detection is enabled: the function shall check the parameter WakeupReasonPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.7 EthIf_GetPhysAddr

[SWS_EthIf_00061]⌈

Service Name	EthIf_GetPhysAddr	
Syntax	`void EthIf_GetPhysAddr ( uint8 CtrlIdx, uint8* PhysAddrPtr )`	
Service ID [hex]	0x08	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
Parameters (inout)	None	
Parameters (out)	PhysAddr Ptr	Physical source address (MAC address) in network byte order.
Return value	None	
Description	Obtains the physical source address used by the indexed controller	
Available via	EthIf.h	

]()

[SWS_EthIf_00062] ⌈
The function EthIf_GetPhysAddr shall forward the call to the respective Ethernet Controller Driver.⌋()

[SWS_EthIf_00063] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00064] ⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00065] ⌈
If development error detection is enabled: the function shall check the parameter PhysAddrPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.8 EthIf_SetPhysAddr

[SWS_EthIf_00132]⌈

Service Name	EthIf_SetPhysAddr

Syntax	`void EthIf_SetPhysAddr (`
	`  uint8 CtrlIdx,`
	`  const uint8* PhysAddrPtr`
	`)`	

Service ID [hex]	0x0d	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant for the same CtrlIdx, reentrant for different	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Driver.
	PhysAddr Ptr	Pointer to memory containing the physical source address (MAC address) in network byte order.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Sets the physical source address used by the indexed controller.	
Available via	EthIf.h	

]()

[SWS_EthIf_00134] ⌈
The function EthIf_SetPhysAddr shall forward the call to the respective Ethernet Controller Driver.⌋()

[SWS_EthIf_00135] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00136] ⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00137] ⌈
If development error detection is enabled: the function shall check the parameter PhysAddrPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.9 EthIf_UpdatePhysAddrFilter

[SWS_EthIf_00139]⌈

Service Name	EthIf_UpdatePhysAddrFilter

Syntax	`Std_ReturnType EthIf_UpdatePhysAddrFilter (`
	`  uint8 CtrlIdx,`
	`  const uint8* PhysAddrPtr,`
	`  Eth_FilterActionType Action`
	`)`	

Service ID [hex]	0x0c	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant for the same CtrlIdx, reentrant for different	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Driver.
	PhysAddr Ptr	Pointer to memory containing the physical destination address (MAC address) in network byte order. This is the multicast destination address of the layer 2 Ethernet packet.
	Action	Add or remove the address from the Ethernet controllers filter.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_-Return-Type	E_OK: filter was successfully changed E_NOT_OK: filter could not be changed
Description	Update the physical source address to/from the indexed controller filter. If the Ethernet Controller is not capable to do the filtering, the software has to do this.	
Available via	EthIf.h	

]()

[SWS_EthIf_00140] ⌈
The function EthIf_SetPhysAddrFilter shall forward the call to the respective Ethernet Controller Driver.⌋()

[SWS_EthIf_00141] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00142] ⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00143] ⌈

If development error detection is enabled: the function shall check the parameter PhysAddrPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.10 EthIf_GetPortMacAddr

[SWS_EthIf_00190]⌈

Service Name	EthIf_GetPortMacAddr	
Syntax	`Std_ReturnType EthIf_GetPortMacAddr ( const uint8* MacAddrPtr, uint8* SwitchIdxPtr, uint8* PortIdxPtr )`	
Service ID [hex]	0x28	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	MacAddrPtr	MAC-address for which a switch port is searched over which the node with this MAC-address can be reached.
Parameters (inout)	None	
Parameters (out)	SwitchIdxPtr	Pointer to the switch index
	PortIdxPtr	Pointer to the port index
Return value	Std_Return-Type	E_OK: success E_NOT_OK: an error occurred, e.g. multiple ports were found
Description	Obtains the port over which this MAC-address can be reached	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00191] ⌈
The function EthIf_GetPortMacAddr shall return the switch and port index over which the given MAC-address is reachable. If multiple or no ports are possible, this API call will return E_NOT_OK. EthSwt_GetPortMacAddr will be called for all Ethernet Switch drivers.⌋()

[SWS_EthIf_00192] ⌈
The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfGetPortMacAddrApi.⌋()

[SWS_EthIf_00193] ⌈

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00194] ⌈
If development error detection is enabled: the function shall check the parameter MacAddrPtr, SwitchIdxPtr and PortIdxPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.11 EthIf_GetArlTable

[SWS_EthIf_00196]⌈

Service Name	EthIf_GetArlTable	
Syntax	`Std_ReturnType EthIf_GetArlTable ( uint8 switchIdx, uint16* numberOfElements, Eth_MacVlanType* arlTableListPointer )`	
Service ID [hex]	0x29	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	switchIdx	Index of the switch within the context of the Ethernet Switch Driver
Parameters (inout)	numberOf Elements	In: Maximum number of elements which can be written into the arl Table Out: Number of elements which are currently available in the EthSwitch module.
Parameters (out)	arlTableList Pointer	Returns a pointer to the memory where the ARL table of the switch consisting of a list of structs with MAC-address, VLAN-ID and port shall be stored.
Return value	Std_Return-Type	E_OK: success E_NOT_OK: requested switchIdx is not valid or inactive
Description	Obtains the address resolution table of a switch and copies the list into a user provided buffer. The function will copy all or numberOfElements into the output list. If input value of numberOfElements is 0 the function will not copy any data but only return the number of valid entries in the cache. arlTableListPointer may be NULL_ PTR in this case.	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00197] ⌈
The function EthIf_GetArlTable shall return a list of structs with MAC-address, VLAN-ID and port for the indexed switch.⌋()

[SWS_EthIf_00254] ⌈
The function EthIf_GetArlTable shall forward the call to function EthSwt_GetArlTable of the respective Ethernet Switch Driver.⌋()

[SWS_EthIf_00198] ⌈
The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfGetArlTable.⌋()

[SWS_EthIf_00199] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00200] ⌈
If development error detection is enabled: the function shall check the parameter ArlTable for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.12 EthIf_GetCtrlIdxList

[SWS_EthIf_91053]⌈

Service Name	EthIf_GetCtrlIdxList	
Syntax	`Std_ReturnType EthIf_GetCtrlIdxList ( uint8* NumberOfCtrlIdx, uint8* CtrlIdxListPtr )`	
Service ID [hex]	0x44	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	None	
Parameters (inout)	NumberOfCtrlIdx	in: maximum number of controllers in CtrlIdxListPtr, 0 to return the number of controllers but without filling CtrlIdxListPtr. out: number of active controllers.
Parameters (out)	CtrlIdxListPtr	List of active controller indexes
Return value	Std_-Return-Type	E_OK: success E_NOT_OK: failure
Description	Returns the number and index of all active Ethernet controllers.	

Available via	EthIf.h

]()

[SWS_EthIf_00298]⌈
The optional EthIf_GetCtrlIdxList API shall return only the NumberOfCtrlIdx which are active.⌋ ()

[SWS_EthIf_00299]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋ ()

[SWS_EthIf_00300]⌈
If development error detection is enabled: the function shall check the OUT parameter CtrlIdxListPtr for being valid only if the the OUT parameter NumberOfCtrlIdx is greater 0x00. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋ ()

8.3.13 EthIf_GetVlanId

[SWS_EthIf_91052]⌈

Service Name	EthIf_GetVlanId	
Syntax	`Std_ReturnType EthIf_GetVlanId ( uint8 CtrlIdx, uint16* VlanIdPtr )`	
Service ID [hex]	0x43	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
Parameters (inout)	None	
Parameters (out)	VlanIdPtr	Pointer to store the VLAN identifier (VID) of the Ethernet controller. 0 if the the Ethernet controller represents no virtual network (VLAN).
Return value	Std_-Return-Type	E_OK: success E_NOT_OK: failure
Description	Returns the VLAN identifier of the requested Ethernet controller.	

Available via	EthIf.h

]()

[SWS_EthIf_00301]⌈

The optional EthIf_GetVlanId API shall return the VlanId of the requested CtrlIdx.⌋()

[SWS_EthIf_00302]⌈

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00303]⌈

If development error detection is enabled: the function shall check the parameter VlanId for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.14 EthIf_GetAndResetMeasurementData

[SWS_EthIf_91011]⌈

Service Name	EthIf_GetAndResetMeasurementData	

Syntax	`Std_ReturnType EthIf_GetAndResetMeasurementData (`
	`  EthIf_MeasurementIdxType MeasurementIdx,`
	`  boolean MeasurementResetNeeded,`
	`  uint32* MeasurementDataPtr`
	`)`	

Service ID [hex]	0x45	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	MeasurementIdx	Data index of measurement data
	MeasurementResetNeeded	Flag to trigger a reset of the measurement data
Parameters (inout)	None	
Parameters (out)	MeasurementDataPtr	Reference to data buffer, where to copy measurement data
Return value	Std_ReturnType	E_OK: successful E_NOT_OK: failed
Description	Allows to read and reset detailed measurement data for diagnostic purposes. Get all	

	MeasurementIdx's at once is not supported. ETHIF_MEAS_ALL shall only be used to reset all MeasurementIdx's at once. A NULL_PTR shall be provided for MeasurementDataPtr in this case.
Available via	EthIf.h

]()

[SWS_EthIf_00308]⌈
EthIf_GetAndResetMeasurementData shall return measurement data for selected measurement index.⌋()

[SWS_EthIf_00309]⌈
For measurement index ETHIF_MEAS_DROP_CRTLIDX the function shall return the number of all dropped datagrams, caused by invalid CrtlIdx/VLAN. If the VLAN is not enabled, all received VLAN tagged datagrams are invalid and shall be counted also.⌋
()

[SWS_EthIf_00310]⌈
The function shall return E_NOT_OK if the requested measurement index is not supported.⌋()

[SWS_EthIf_00312]⌈
The function shall reset all existing measurement data to 0, if MeasurementResetNeeded is true and measurement index is set to ETHIF_MEAS_ALL.⌋()

[SWS_EthIf_00313]⌈
All measurement data which counts data shall not overrun.⌋()

[SWS_EthIf_00314]⌈
The function shall accept NULL_PTR. In this case the measurement data shall not be copied.⌋()

[SWS_EthIf_00316]⌈
The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfGetAndResetMeasurementDataApi.⌋()

[SWS_EthIf_00317]⌈
If the VLAN is not active the Ethernet Interface shall increment the corresponding measurement data and filter the message.⌋()

[SWS_EthIf_00319]⌜
If development error detection is enabled: The function shall check that the service EthIf_Init () was previously called. If the check fails, the function shall raise the development error ETHIF_E_NOTINIT.⌟()

8.3.15 EthIf_StoreConfiguration

[SWS_EthIf_00214]⌈

Service Name	EthIf_StoreConfiguration	
Syntax	`Std_ReturnType EthIf_StoreConfiguration ( uint8 SwitchIdx )`	
Service ID [hex]	0x2c	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Storage/Reset request accepted E_NOT_OK: Storage/Reset request not accepted
Description	Trigger the storage/reset of the configuration of the learned MAC/Port tables of a switch in a persistent manner and will be used by e.g. CDD.	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00215] ⌈
The function EthIf_StoreConfiguration shall trigger to store the learned MAC/Port tables of a Ethernet switch.⌋()

[SWS_EthIf_00216] ⌈
The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfStoreConfigurationApi.⌋()

[SWS_EthIf_00217] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

8.3.16 EthIf_ResetConfiguration

[SWS_EthIf_00219]⌈

Service Name	EthIf_ResetConfiguration	
Syntax	`Std_ReturnType EthIf_ResetConfiguration ( uint8 SwitchIdx )`	
Service ID [hex]	0x2d	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request to persistently reset the MAC/Port table was accepted E_NOT_OK: Request to persistently reset the MAC/Port table was not accepted
Description	The function shall request to reset the configuration of the learned MAC/Port tables of a Ethernet switch in a persistent manner. This could be used by e.g. a CDD. The statically configured entries shall still remain.	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00220] ⌈
The function EthIf_ResetConfiguration shall trigger to reset the learned MAC/Port tables of a Ethernet switch.⌋()

[SWS_EthIf_00221] ⌈
The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfResetConfigurationApi.⌋()

[SWS_EthIf_00222] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

8.3.17 EthIf_GetCurrentTime

[SWS_EthIf_00154]⌈

Service Name	EthIf_GetCurrentTime	

Syntax	`Std_ReturnType EthIf_GetCurrentTime (`
	`  uint8 CtrlIdx,`
	`  Eth_TimeStampQualType* timeQualPtr,`
	`  Eth_TimeStampType* timeStampPtr`
	`)`	

Service ID [hex]	0x22	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the addresses ETH controller.
Parameters (inout)	None	
Parameters (out)	timeQualPtr	quality of HW time stamp, e.g. based on current drift
	timeStampPtr	current time stamp
Return value	Std_ReturnType	E_OK: successful E_NOT_OK: failed
Description	Returns a time value out of the HW registers according to the capability of the HW. Is the HW resolution is lower than the Eth_TimeStampType resolution resp. range, the remaining bits will be filled with 0. Important Note: EthIf_GetCurrentTime may be called within an exclusive area.	
Available via	EthIf.h	

]()

[SWS_EthIf_00155] ⌈

If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00156] ⌈

If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00157] ⌈

If development error detection is enabled: the function shall check the parameter timeQualPtr and timeStampPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

[SWS_EthIf_00158] ⌈

The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfGlobalTimeSupport.]()

[SWS_EthIf_00473] [
The EthIf module shall apply appropriate mechanisms to allow calls of EthIf_GetCurrentTime API from other partitions than its main function, e.g. by providing an EthIf satellite.]()

8.3.18 EthIf_EnableEgressTimeStamp

[SWS_EthIf_00160][

Service Name	EthIf_EnableEgressTimeStamp	
Syntax	`void EthIf_EnableEgressTimeStamp ( uint8 CtrlIdx, Eth_BufIdxType BufIdx )`	
Service ID [hex]	0x23	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the addresses ETH controller.
	BufIdx	Index of the message buffer, where Application expects egress time stamping
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Activates egress time stamping on a dedicated message object. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no "disable" functionality, due to the fact, that the message type is always "time stamped" by network design.	
Available via	EthIf.h	

]()

[SWS_EthIf_00161] [
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.]()

[SWS_EthIf_00162] ⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00164] ⌈
The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfGlobalTimeSupport.⌋()

8.3.19 EthIf_GetEgressTimeStamp

[SWS_EthIf_00166]⌈

Service Name	EthIf_GetEgressTimeStamp	
Syntax	`Std_ReturnType EthIf_GetEgressTimeStamp ( uint8 CtrlIdx, Eth_BufIdxType BufIdx, Eth_TimeStampQualType* timeQualPtr, Eth_TimeStampType* timeStampPtr )`	
Service ID [hex]	0x24	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the address ETH controller.
	BufIdx	Index of the message buffer, where the Upper Layer expects egress time stamping
Parameters (inout)	None	
Parameters (out)	timeQualPtr	quality of HW time stamp, e.g. based on current drift
	timeStampPtr	current time stamp
Return value	Std_Return-Type	E_OK: success E_NOT_OK: failed to read time stamp.
Description	Reads back the egress time stamp on a dedicated message object. It must be called within the TxConfirmation() function.	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00167] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00168] ⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00169] ⌈
If development error detection is enabled: the function shall check the parameter timeQualPtr and timeStampPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

[SWS_EthIf_00170] ⌈
The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfGlobalTimeSupport.⌋()

8.3.20 EthIf_GetIngressTimeStamp

[SWS_EthIf_00172]⌈

Service Name	EthIf_GetIngressTimeStamp	
Syntax	`Std_ReturnType EthIf_GetIngressTimeStamp ( uint8 CtrlIdx, const Eth_DataType* DataPtr, Eth_TimeStampQualType* timeQualPtr, Eth_TimeStampType* timeStampPtr )`	
Service ID [hex]	0x25	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the addresses ETH controller.
	DataPtr	Pointer to the message buffer, where Application expects ingress time stamping
Parameters (inout)	None	
Parameters (out)	timeQualPtr	quality of HW time stamp, e.g. based on current drift
	timeStampPtr	current time stamp
Return value	Std_Return-Type	E_OK: success E_NOT_OK: failed to read time stamp.
Description	Reads back the ingress time stamp on a dedicated message object. It must be called within the RxIndication() function.	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00173] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00174] ⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00175] ⌈
If development error detection is enabled: the function shall check the parameter DataPtr, timeQualPtr and timeStampPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

[SWS_EthIf_00176] ⌈
The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfGlobalTimeSupport.⌋()

8.3.21 EthIf_SwitchPortGroupRequestMode

[SWS_EthIf_91102]⌈

Service Name	EthIf_SwitchPortGroupRequestMode	
Syntax	`Std_ReturnType EthIf_SwitchPortGroupRequestMode ( EthIf_SwitchPortGroupIdxType PortGroupIdx, Eth_ModeType PortMode )`	
Service ID [hex]	0x06	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	PortGroup Idx	Index of the port group within the context of the Ethernet Interface
	PortMode	ETH_MODE_DOWN: disable the Ethernet switch port group ETH_MODE_ACTIVE: enable the Ethernet switch port group ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST: enable the port group and request for a wake-up on the network
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: success E_NOT_OK: port group mode could not be changed

Description	Request a mode for the EthIfSwtPortGroup. The call shall be forwarded to EthSwt by calling EthSwt_SetSwitchPortMode for all EthSwtPorts referenced by the port group.
Available via	EthIf.h

⌋()

[SWS_EthIf_00270]⌈
If EthIf_SwitchPortGroupRequestMode is called with ETH_MODE_DOWN EthIf shall start a timer with EthIfSwitchOffPortTimedelay for all ports of the respective EthIf_SwitchPortGroup if the mode ETH_MODE_DOWN has been requested for all EthIfSwitchPortGroups referencing the port and the current mode is ETH_MODE_ACTIVE.⌋()

[SWS_EthIf_00271]⌈
If the timer to switch off ports (see SWS_EthIf_00270) elapses for a port, EthIf shall call the following functions in the given order for the corresponding EthSwtPort:

1. EthSwt_PortLinkStateRequest with ETHTRCV_LINK_STATE_DOWN
2. EthSwt_SetSwitchPortMode with ETH_MODE_DOWN

⌋()

Note: The implementation has to ensure that EthSwtPorts within EthIfSwitchPortGroups are only disabled if all prior activation request have been withdrawn. This could be realized e.g. by a counter mechanism.

Rationale: Delaying to switch off EthSwtPorts by EthIfSwitchOffPortTimedelay is needed to ensure a simultaneous switch-off of the Ethernet switch port and the Ethernet hardware (PHY or another Ethernet switch) of the connected communication partner:

1. If the Ethernet hardware of the connected communication partner is an PHY, then the EthIfSwitchOffPortTimedelay cover the time which is needed until the PHY of the connected communication partner will be switched off, due to the NM handling.
2. If the Ethernet hardware of the connected communication partner is an Ethernet switch, then both EthSwtPorts should be switched off in the same point in time to avoid link down recognition.

[SWS_EthIf_00273]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00274]⌈
If development error detection is enabled: the function shall check that the provided parameter PortGroupIdx addresses a port group not referenced by any

EthIfController. If the check fails, the function shall raise the development error ETHIF_E_INV_PORT_GROUP_IDX.⌋()

Rationale: Avoid that a EthIfSwitchPortGroup which shall be controlled by EthIfController is incidentally called by BswM

8.3.22 EthIf_StartAllPorts

[SWS_EthIf_91103]⌈

Service Name	EthIf_StartAllPorts	
Syntax	`Std_ReturnType EthIf_StartAllPorts ( void )`	
Service ID [hex]	0x07	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Request was accepted E_NOT_OK: Request was rejected
Description	Request to set all configured and affected EthSwtPorts to ETH_MODE_ ACTIVE	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00277]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

8.3.23 EthIf_SetSwitchMgmtInfo

[SWS_EthIf_91003]⌈

Service Name	EthIf_SetSwitchMgmtInfo
Syntax	`Std_ReturnType EthIf_SetSwitchMgmtInfo ( uint8 CtrlIdx, Eth_BufIdxType BufIdx, EthSwt_MgmtInfoType* MgmtInfoPtr )`

Service ID [hex]	0x38	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of an Ethernet Interface controller
	BufIdx	Ethernet Tx Buffer index
	MgmtInfoPtr	Pointer to the management information
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: Management infos successfully set E_NOT_OK: Setting of management infos failed
Description	Provides additional management information along to an Ethernet frame that requires special treatment within the Switch. It has to be called between EthIf_ ProvideTxBuffer() and EthIf_Transmit() of the related frame.	
Available via	EthIf.h	

⌋(SRS_Eth_00125)

[SWS_EthIf_00279]⌈
The function shall be pre compile time configurable ON/OFF by the configuration parameter: EthIfSwitchManagementSupport.⌋()

[SWS_EthIf_00280]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init() was previously called.
If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00281]⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid.
If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00282]⌈
If development error detection is enabled: the function shall check the parameter BufIdx for being valid.
If the check fails, the function shall raise the development error ETHIF_E_INV_PARAM.⌋()

[SWS_EthIf_00283]⌈
If development error detection is enabled: the function shall check the parameter MgmtInfoPtr for being valid.
If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.24 EthIf_GetRxMgmtObject

[SWS_EthIf_91105]⌈

Service Name	EthIf_GetRxMgmtObject	
Syntax	`Std_ReturnType EthIf_GetRxMgmtObject ( uint8 CtrlIdx, Eth_DataType* DataPtr, EthSwt_MgmtObjectType **MgmtObjectPtr )`	
Service ID [hex]	0x47	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	CtrlIdx	Index of an Ethernet Interface controller
	DataPtr	Ethernet data pointer
Parameters (inout)	None	
Parameters (out)	**MgmtObjectPtr	MgmtObjectPtr Pointer to the management object
Return value	Std_ReturnType	E_OK: success E_NOT_OK: management object could not be obtained
Description	Request the MgmtObject of the (in this context) unique DataPtr.	
Available via	EthIf.h	

⌋()

8.3.25 EthIf_GetTxMgmtObject

[SWS_EthIf_91106]⌈

Service Name	EthIf_GetTxMgmtObject
Syntax	`Std_ReturnType EthIf_GetTxMgmtObject ( uint8 CtrlIdx, Eth_BufIdxType BufIdx, EthSwt_MgmtObjectType **MgmtObjectPtr )`
Service ID [hex]	0x48

Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	CtrlIdx	Index of an Ethernet Interface controller
	BufIdx	Ethernet Rx Buffer index
Parameters (inout)	None	
Parameters (out)	**MgmtObjectPtr	Pointer to the management object
Return value	Std_ReturnType	E_OK: success E_NOT_OK: management object could not be obtained
Description	Request the MgmtObject of the (in this context) unique BufIdx.	
Available via	EthIf.h	

]()

8.3.26 EthIf_SwitchEnableTimeStamping

[SWS_EthIf_91007][

Service Name	EthIf_SwitchEnableTimeStamping	
Syntax	`Std_ReturnType EthIf_SwitchEnableTimeStamping ( uint8 CtrlIdx, Eth_BufIdxType BufIdx, EthSwt_MgmtInfoType* MgmtInfo )`	
Service ID [hex]	0x39	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	BufIdx	Index of the message buffer, where Application expects egress time stamping
Parameters (inout)	None	
Parameters (out)	MgmtInfo	Management information
Return value	Std_Return-Type	E_OK: Time stamping on egress successfully enabled E_NOT_OK: Enabling of time stamping on egress has been failed
Description	Activates egress time stamping on a dedicated message object, addressed by CtrlIdx and BufIdx.	
Available via	EthIf.h	

](SRS_Eth_00125)

[SWS_EthIf_00387]⌈
If EthIf_SwitchEnableTimeStamping is called, the EthIf shall call EthSwt_PortEnableTimeStamp for every port in the group.⌋()

[SWS_EthIf_00285]⌈
The function shall be pre compile time configurable ON/OFF by the configuration parameter: EthIfGlobalTimeSupport.⌋()

[SWS_EthIf_00286]⌈
If development error detection is enabled: the function shall check that the service Eth_Init() was previously called.
If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00287]⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid.
If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00288]⌈
If development error detection is enabled: the function shall check the parameter BufIdx for being valid.
If the check fails, the function shall raise the development error ETHIF_E_INV_PARAM.⌋()

[SWS_EthIf_00289]⌈
If development error detection is enabled: the function shall check the parameter BufIdx for being valid.
If the check fails, the function shall raise the development error ETHIF_E_INV_PARAM.⌋()

[SWS_EthIf_00290]⌈
If development error detection is enabled: the function shall check the parameter BufIdx for being valid.
If the check fails, the function shall raise the development error ETHIF_E_INV_PARAM.⌋()

8.3.27 EthIf_VerifyConfig

[SWS_EthIf_91012]⌈

Service Name	EthIf_VerifyConfig	

Syntax	`Std_ReturnType EthIf_VerifyConfig (`
	`  uint8 SwitchIdx,`
	`  boolean* Result`
	`)`	

Service ID [hex]	0x40	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
Parameters (inout)	None	
Parameters (out)	Result	Result of verification, TRUE: configureation verified ok, FALSE: configuraton values found corrupted
Return value	Std_Return-Type	E_OK: Configuration verificaton succeeded, E_NOT_OK: Configuration verification not succeeded.
Description	Forwarded to EthSwt_VerifyConfig. EthSwt_VerifyConfig verifies the Switch Configuration depending on the HW-Architecture, HW-capability and the intended accuracy of this verification.	
Available via	EthIf.h	

]()

[SWS_EthIf_00304]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋(SRS_BSW_00101)(SRS_BSW_00369)

[SWS_EthIf_00305]⌈
The function shall be compile time configurable On/Off by the configuration parameter: EthIfVerifyConfigApi.⌋()

8.3.28 EthIf_SetForwardingMode

[SWS_EthIf_91013]⌈

Service Name	EthIf_SetForwardingMode

Syntax	`Std_ReturnType EthIf_SetForwardingMode (`
	`  uint8 SwitchIdx,`
	`  boolean mode`
	`)`

Service ID [hex]	0x41	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	mode	True Forwarding enabled, False Forwarding disabled
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: stopping of frame forwarding succeeded, E_NOT_OK: stopping of frame forwarding not succeeded.
Description	Verifies the Switch Configuration. If Configuration is not valid, Switch is reconfigured.	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00306]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋(SRS_BSW_00101)(SRS_BSW_00369)

[SWS_EthIf_00307]⌈
The function shall be compile time configurable On/Off by the configuration parameter: EthIfSetForwardingModeApi.⌋()

8.3.29 EthIf_GetTrcvSignalQuality

[SWS_EthIf_91056]⌈

Service Name	EthIf_GetTrcvSignalQuality	
Syntax	`Std_ReturnType EthIf_GetTrcvSignalQuality ( uint8 TrcvIdx, EthIf_SignalQualityResultType* ResultPtr )`	
Service ID [hex]	0x18	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.	
Parameters (in)	TrcvIdx	Index of the transceiver within the context of the Ethernet

		Interface
Parameters (inout)	None	
Parameters (out)	ResultPtr	Pointer to the memory where the signal quality in percent shall be stored.
Return value	Std_Return-Type	E_OK: The signal quality retrieved successfully E_NOT_OK: The signal quality not retrieved successfully
Description	Retrieves the signal quality of the link of the given Ethernet transceiver	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00391]⌈
The function EthIf_GetTrcvSignalQuality shall forward the call to function EthTrcv_GetPhySignalQuality of the corresponding Ethernet Transceiver Driver (TrcvIdx).⌋()

[SWS_EthIf_00392] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00393]⌊
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX.⌋()

[SWS_EthIf_00394]⌊
If development error detection is enabled: the function shall check the parameter ResultPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.30 EthIf_GetSwitchPortSignalQuality

[SWS_EthIf_91058]⌈

Service Name	EthIf_GetSwitchPortSignalQuality

Syntax	`Std_ReturnType EthIf_GetSwitchPortSignalQuality (`
	`  uint8 SwitchIdx,`
	`  uint8 SwitchPortIdx,`
	`  EthIf_SignalQualityResultType* ResultPtr`
	`)`

Service ID [hex]	0x1a	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.	
Parameters (in)	SwitchIdx	Index of the Ethernet switch within the context of the Ethernet Interface
	SwitchPortIdx	Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout)	None	
Parameters (out)	ResultPtr	Pointer to the memory where the signal quality in percent shall be stored.
Return value	Std_Return-Type	E_OK: The signal quality retrieved successfully E_NOT_OK: The signal quality not retrieved successfully
Description	Retrieves the signal quality of the link of the given Ethernet switch port	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00395]⌈
The function EthIf_GetSwitchPortSignalQuality shall forward the call to function EthSwt_GetPortSignalQuality of the corresponding Ethernet Switch Driver (SwitchIdx).⌋()

[SWS_EthIf_00396]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00397]⌈
If development error detection is enabled: the function shall check the parameter SwitchIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_SWT_IDX.⌋()

[SWS_EthIf_00495] ⌈
If development error detection is enabled: the function shall check the parameter SwitchPortIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_PORT_IDX otherwise (if DET is disabled) return E_NOT_OK.⌋()

[SWS_EthIf_00399]⌈

If development error detection is enabled: the function shall check the parameter ResultPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.31 EthIf_ClearTrcvSignalQuality

[SWS_EthIf_91059]⌈

Service Name	EthIf_ClearTrcvSignalQuality	
Syntax	`Std_ReturnType EthIf_ClearTrcvSignalQuality ( uint8 TrcvIdx )`	
Service ID [hex]	0x19	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.	
Parameters (in)	TrcvIdx	Index of the transceiver within the context of the Ethernet Interface
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: The signal quality cleared successfully E_NOT_OK: The signal quality cleared not successfully
Description	Clear the stored signal quality of the link of the given Ethernet transceiver	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00400]⌈
The function EthIf_ClearTrcvSignalQuality shall clear the stored signal quality values (see EthIf_SignalQualityResultType) of the EthIfTransceiver given by TrcvIdx.⌋()

[SWS_EthIf_00401]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00402]⌈
If development error detection is enabled: the function shall check the parameter SwitchIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX.⌋()

8.3.32 EthIf_ClearSwitchPortSignalQuality

[SWS_EthIf_91060]⌈

Service Name	EthIf_ClearSwitchPortSignalQuality	
Syntax	`Std_ReturnType EthIf_ClearSwitchPortSignalQuality ( uint8 SwitchIdx, uint8 SwitchPortIdx )`	
Service ID [hex]	0x1b	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different Ethernet switch indexes and Ethernet Switch port indexes. Non reentrant for the same SwitchPortIdx.	
Parameters (in)	SwitchIdx	Index of the Ethernet switch within the context of the Ethernet Interface
	SwitchPortIdx	Index of the Ethernet switch port within the context of the Ethernet Interface
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: The signal quality cleared successfully E_NOT_OK: The signal quality cleared not successfully
Description	Clear the stored signal quality of the link of the given Ethernet switch port	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00404]⌈
The function EthIf_ClearSwitchPortSignalQuality shall clear the stored signal quality values (see EthIf_SignalQualityResultType) of the EthSwtPort given by SwitchIdx and SwitchPortIdx.⌋()

[SWS_EthIf_00405]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00406]⌈
If development error detection is enabled: the function shall check the parameter SwitchIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_SWT_IDX.⌋()

[SWS_EthIf_00496][
If development error detection is enabled: the function shall check the parameter SwitchPortIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_PORT_IDX otherwise (if DET is disabled) return E_NOT_OK.」()

8.3.33 EthIf_SetPhyTestMode

[SWS_EthIf_91016][

Service Name	EthIf_SetPhyTestMode	
Syntax	`Std_ReturnType EthIf_SetPhyTestMode ( uint8 TrcvIdx, EthTrcv_PhyTestModeType Mode )`	
Service ID [hex]	0x17	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.	
Parameters (in)	TrcvIdx	Index of the transceiver within the context of the Ethernet Interface
	Mode	Test mode to be activated
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted.
Description	Activates a given test mode.	
Available via	EthIf.h	

](SRS_Eth_00117)

[SWS_EthIf_00324]「
The function EthIf_SetPhyTestMode shall forward the call to function EthTrcv_SetPhyTestMode of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).」()

[SWS_EthIf_00325]「
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.」()

[SWS_EthIf_00326]⌈
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX.⌋()

8.3.34 EthIf_SetPhyLoopbackMode

[SWS_EthIf_91018]⌈

Service Name	EthIf_SetPhyLoopbackMode	
Syntax	`Std_ReturnType EthIf_SetPhyLoopbackMode ( uint8 TrcvIdx, EthTrcv_PhyLoopbackModeType Mode )`	
Service ID [hex]	0x12	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.	
Parameters (in)	TrcvIdx	Index of the transceiver within the context of the Ethernet Interface
	Mode	Loopback mode to be activated
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted.
Description	Activates a given loopback mode.	
Available via	EthIf.h	

⌋(SRS_Eth_00117)

[SWS_EthIf_00327]⌈
The function EthIf_SetPhyLoopbackMode shall forward the call to function EthTrcv_SetPhyLoopbackMode of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).⌋()

[SWS_EthIf_00328]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00329]⌜
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX.⌟()

8.3.35 EthIf_SetPhyTxMode

[SWS_EthIf_91061]{DRAFT} [

Service Name	EthIf_SetPhyTxMode (draft)	
Syntax	`Std_ReturnType EthIf_SetPhyTxMode ( uint8 TrcvIdx, EthTrcv_PhyTxModeType Mode )`	
Service ID [hex]	0x13	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.	
Parameters (in)	TrcvIdx	Index of the transceiver within the context of the Ethernet Interface
	Mode	Transmission mode to be activated
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted
Description	Activates a given transmission mode. **Tags:** atp.Status=draft	
Available via	EthIf.h	

](SRS_Eth_00117)

[SWS_EthIf_00388]⌜
The function EthIf_SetPhyTxMode shall forward the call to function EthTrcv_SetPhyTxMode of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).⌟()

[SWS_EthIf_00389]⌜
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌟()

[SWS_EthIf_00390]⌈
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX.⌋()

8.3.36 EthIf_GetCableDiagnosticsResult

[SWS_EthIf_91014]⌈

Service Name	EthIf_GetCableDiagnosticsResult	
Syntax	`Std_ReturnType EthIf_GetCableDiagnosticsResult ( uint8 TrcvIdx, EthTrcv_CableDiagResultType* ResultPtr )`	
Service ID [hex]	0x14	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.	
Parameters (in)	TrcvIdx	Index of the transceiver within the context of the Ethernet Interface
Parameters (inout)	None	
Parameters (out)	ResultPtr	Pointer to the location where the cable diagnostics result shall be stored
Return value	Std_Return-Type	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted
Description	Retrieves the cable diagnostics result of a given transceiver.	
Available via	EthIf.h	

⌋(SRS_Eth_00117)

[SWS_EthIf_00330]⌈
The function EthIf_GetCableDiagnosticsResult shall forward the call to function EthTrcv_GetCableDiagnosticsResult of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).⌋()

[SWS_EthIf_00331]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00332]⌈

If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX.」()

[SWS_EthIf_00333]「
If development error detection is enabled: the function shall check the parameter ResultPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.」()

8.3.37 EthIf_GetPhyIdentifier

[SWS_EthIf_91020][

Service Name	EthIf_GetPhyIdentifier	

Syntax	`Std_ReturnType EthIf_GetPhyIdentifier (`
	`  uint8 TrcvIdx,`
	`  uint32* OrgUniqueIdPtr,`
	`  uint8* ModelNrPtr,`
	`  uint8* RevisionNrPtr`
	`)`	

Service ID [hex]	0x15	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.	
Parameters (in)	TrcvIdx	Index of the transceiver within the context of the Ethernet Interface
Parameters (inout)	None	
Parameters (out)	OrgUniqueId Ptr	Pointer to the memory where the Organizationally Unique Identifier shall be stored.
	ModelNrPtr	Pointer to the memory where the Manufacturer's Model Number shall be stored.
	RevisionNrPtr	Pointer to the memory where the Revision Number shall be stored.
Return value	Std_Return-Type	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted
Description	Obtains the PHY identifier of the Ethernet Interface according to IEEE 802.3-2015 chapter 22.2.4.3.1 PHY Identifer.	
Available via	EthIf.h	

](SRS_Eth_00117)

[SWS_EthIf_00334]「

The function EthIf_GetPhyIdentifier shall forward the call to function EthTrcv_GetPhyIdentifier of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).⌋()

[SWS_EthIf_00335]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00336]⌈
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX.⌋()

[SWS_EthIf_00337]⌈
If development error detection is enabled: the function shall check the parameter OrgUniqueIdPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

[SWS_EthIf_00338]⌈
If development error detection is enabled: the function shall check the parameter ModelNrPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

[SWS_EthIf_00339]⌈
If development error detection is enabled: the function shall check the parameter RevisionNrPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.38 EthIf_GetBufWRxParams

[SWS_EthIf_91002]⌈

Service Name	EthIf_GetBufWRxParams

Syntax	`Std_ReturnType EthIf_GetBufWRxParams (`
	`  uint8 CtrlIdx,`
	`  const WEth_BufWRxParamIdType* RxParamIds,`
	`  uint32* ParamValues,`
	`  uint8 NumParams`
	`)`

Service ID [hex]	0x32
Sync/Async	Synchronous

Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	RxParamIds	IDs of the Parameters to read
	NumParams	Number of Parameters
Parameters (inout)	None	
Parameters (out)	ParamValues	Values of the Parameters requested
Return value	Std_ReturnType	E_OK: success E_NOT_OK: failed reading parameters
Description	Read out values related to the receive direction of the transceiver for a received packet. For example, this could be RSSI or Channel belonging to one single packet.	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00341]⌈
The function EthIf_GetBufWRxParams shall forward the call to function WEth_GetBufWRxParams of the respective Wireless Ethernet Controller Driver.⌋()

[SWS_EthIf_00342]⌈
The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfEnableWEthApi.⌋()

[SWS_EthIf_00343]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00344]⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00345]⌈
If development error detection is enabled: the function shall check the parameter RxParamIds for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

[SWS_EthIf_00346]⌜
If development error detection is enabled: the function shall check the parameter ParamValues for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌟()

Note: The function requires previous reception (EthIf_RxIndication).

8.3.39 EthIf_GetBufWTxParams

[SWS_EthIf_91054]⌈

Service Name	EthIf_GetBufWTxParams	

Syntax	`Std_ReturnType EthIf_GetBufWTxParams (`
	`  uint8 CtrlIdx,`
	`  const WEth_BufWTxParamIdType* TxParamIds,`
	`  uint32* ParamValues,`
	`  uint8 NumParams`
	`)`	

Service ID [hex]	0x31	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	TxParamIds	IDs of the Parameter that are requested
	NumParams	Number of Parameters that are requested
Parameters (inout)	None	
Parameters (out)	ParamValues	Values of the Parameters requested
Return value	Std_ReturnType	E_OK: success E_NOT_OK: failed reading parameters
Description	Read out values related to the transmit direction of the transceiver for a transmitted packet. For example, this could be transaction ID belonging to one single packet.	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00347]⌜
The function EthIf_GetBufWTxParams shall forward the call to function WEth_GetBufWTxParams of the respective Wireless Ethernet Controller Driver.⌟()

[SWS_EthIf_00348]⌜
The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfEnableWEthApi.⌟()

[SWS_EthIf_00349]⌜
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌟()

[SWS_EthIf_00350]⌜
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌟()

[SWS_EthIf_00351]⌜
If development error detection is enabled: the function shall check the parameter TxParamIds for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌟()

[SWS_EthIf_00352]⌜
If development error detection is enabled: the function shall check the parameter ParamValues for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌟()

Note: The function requires previous transmission (EthIf_Transmit).

8.3.40 EthIf_SetBufWTxParams

[SWS_EthIf_91017]⌈

Service Name	EthIf_SetBufWTxParams

Syntax	`Std_ReturnType EthIf_SetBufWTxParams (`
	`  uint8 CtrlIdx,`
	`  Eth_BufIdxType BufIdx,`
	`  const WEth_BufWTxParamIdType* TxParamIds,`
	`  const uint32* ParamValues,`
	`  uint8 NumParams`
	`)`

Service ID [hex]	0x33
Sync/Async	Synchronous
Reentrancy	Non Reentrant

Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	BufIdx	Index of the buffer resource
	TxParamIds	IDs of the Parameter that are provided to the transmit radio
	ParamValues	Values of the Parameters that are provided to the transmit radio
	NumParams	Number of Parameters that are provided to the transmit radio
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: success E_NOT_OK: failed setting parameter
Description	Set values related to the transmit direction of the transceiver for a specific buffer (packet to be sent). For example, this can be the desired transmit power or the channel belonging to one single packet.	
Available via	EthIf.h	

]()

[SWS_EthIf_00353]⌈
The function EthIf_SetBufWTxParams shall forward the call to function WEth_SetBufWTxParams of the respective Wireless Ethernet Controller Driver.⌋()

[SWS_EthIf_00354]⌈
The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfEnableWEthApi.⌋()

[SWS_EthIf_00355]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00356]⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00357]⌈

If development error detection is enabled: the function shall check the parameter BufIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_PARAM.⌋()

[SWS_EthIf_00358]⌈
If development error detection is enabled: the function shall check the parameter TxParamIds for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

[SWS_EthIf_00359]⌈
If development error detection is enabled: the function shall check the parameter ParamValues for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

Note: The function requires previous buffer request (EthIf_ProvideTxBuffer).

8.3.41 EthIf_SetRadioParams

[SWS_EthIf_91026]⌈

Service Name	EthIf_SetRadioParams	
Syntax	`Std_ReturnType EthIf_SetRadioParams ( uint8 TrcvId, const WEthTrcv_SetRadioParamIdType* ParamIds, const uint32* ParamValue, uint8 NumParams )`	
Service ID [hex]	0x34	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	TrcvId	Index of the transceiver
	ParamIds	IDs of the Parameters to set
	ParamValue	Values of the Parameters to set
	NumParams	Number of Parameters to set
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: success E_NOT_OK: failed writing parameters

Description	Set values related to a transceiver's wireless radio. For example, this could be the selection of the radio settings (channel, ...).
Available via	EthIf.h

]()

[SWS_EthIf_00360]⌈
The function EthIf_SetRadioParams shall forward the call to function WEthTrcv_SetRadioParams of the respective Wireless Ethernet Transceiver Driver.⌋
()

[SWS_EthIf_00361]⌈
The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfEnableWEthApi.⌋()

[SWS_EthIf_00362]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00363]⌈
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX.⌋()

[SWS_EthIf_00364]⌈
If development error detection is enabled: the function shall check the parameter ParamIds for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

[SWS_EthIf_00365]⌈
If development error detection is enabled: the function shall check the parameter ParamValues for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.42 EthIf_SetChanRxParams

[SWS_EthIf_91034]⌈

Service Name	EthIf_SetChanRxParams

Syntax	`Std_ReturnType EthIf_SetChanRxParams (`
	`  uint8 TrcvId,`
	`  uint8 RadioId,`

	`const WEthTrcv_SetChanRxParamIdType* ParamIds,`
	`const uint32* ParamValues,`
	`uint8 NumParams`
	`)`	

Service ID [hex]	0x35	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	TrcvId	Index of the transceiver
	RadioId	Index of the Transceiver's Radio (including channel)
	ParamIds	IDs of the Parameters to set
	ParamValues	Values of the Parameters to set
	NumParams	Number of Parameters to set
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: success E_NOT_OK: failed writing parameters
Description	Set values related to the receive direction of a transceiver's wireless channel. For example, this could be a channel parameter like the frequency.	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00366]⌈
The function EthIf_SetChanRxParams shall forward the call to function WEthTrcv_SetChanRxParams of the respective Wireless Ethernet Transceiver Driver.⌋()

[SWS_EthIf_00367]⌈
The function EthIf_SetChanRxParams shall be pre compile time configurable On/Off by the configuration parameter: EthIfEnableWEthApi.⌋()

[SWS_EthIf_00368]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00369]⌈

If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX.」()

[SWS_EthIf_00370]「
If development error detection is enabled: the function shall check the parameter RadioIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_PARAM.」()

[SWS_EthIf_00371]「
If development error detection is enabled: the function shall check the parameter RxParamIds for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.」()

[SWS_EthIf_00372]「
If development error detection is enabled: the function shall check the parameter ParamValues for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.」()

8.3.43 EthIf_SetChanTxParams

[SWS_EthIf_91042][

Service Name	EthIf_SetChanTxParams	

Syntax	`Std_ReturnType EthIf_SetChanTxParams (`
	`  uint8 TrcvId,`
	`  uint8 RadioId,`
	`  const WEthTrcv_SetChanTxParamIdType* TxParamIds,`
	`  const uint32* ParamValues,`
	`  uint8 NumParams`
	`)`	

Service ID [hex]	0x36	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	TrcvId	Index of the transceiver
	RadioId	Index of the Transceiver's Radio (including channel)
	TxParamIds	IDs of the Parameters to set
	ParamValues	Values of the Parameters to set
	NumParams	Number of Parameters to set
Parameters (inout)	None	

Parameters (out)	None	
Return value	Std_ReturnType	E_OK: success E_NOT_OK: failed writing parameters
Description	Set values related to the transmit direction of a transceiver's wireless channel. For example, this could be the bitrate of a channel.	
Available via	EthIf.h	

]()

[SWS_EthIf_00373]⌜
The function EthIf_SetChanTxParams shall forward the call to function WEthTrcv_SetChanTxParams of the respective Wireless Ethernet Transceiver Driver.⌟()

[SWS_EthIf_00374]⌜
The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfEnableWEthApi.⌟()

[SWS_EthIf_00375]⌜
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌟()

[SWS_EthIf_00376]⌜
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX.⌟()

[SWS_EthIf_00377]⌜
If development error detection is enabled: the function shall check the parameter RadioIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_PARAM.⌟()

[SWS_EthIf_00378]⌜
If development error detection is enabled: the function shall check the parameter TxParamIds for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌟()

[SWS_EthIf_00379]⌜

If development error detection is enabled: the function shall check the parameter ParamValues for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.44 EthIf_GetChanRxParams

[SWS_EthIf_91050]⌈

Service Name	EthIf_GetChanRxParams	

Syntax	`Std_ReturnType EthIf_GetChanRxParams (`
	`  uint8 TrcvId,`
	`  uint8 RadioId,`
	`  const WEthTrcv_GetChanRxParamIdType* ParamIds,`
	`  uint32* ParamValues,`
	`  uint8 NumParams`
	`)`	

Service ID [hex]	0x37	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	TrcvId	Index of the transceiver
	RadioId	Index of the Transceiver's Radio (including channel)
	ParamIds	IDs of the Parameters to read
	NumParams	Number of Parameters to read
Parameters (inout)	None	
Parameters (out)	ParamValues	Values of the requested Parameters
Return value	Std_ReturnType	E_OK: success E_NOT_OK: failed reading parameters
Description	Read values related to the receive direction of the transceiver. For example, this could be a Channel Busy Ratio (CBR) or the average Channel Idle Time (CIT).	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00380]⌈
The function EthIf_GetChanRxParams shall forward the call to function WEthTrcv_GetChanRxParams of the respective Wireless Ethernet Transceiver Driver.⌋()

[SWS_EthIf_00381]⌈

The function shall be pre compile time configurable On/Off by the configuration parameter: EthIfEnableWEthApi.」()

[SWS_EthIf_00382]「
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.」()

[SWS_EthIf_00383]「
If development error detection is enabled: the function shall check the parameter TrcvIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_TRCV_IDX.」()

[SWS_EthIf_00384]「
If development error detection is enabled: the function shall check the parameter RadioIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_PARAM.」()

[SWS_EthIf_00385]「
If development error detection is enabled: the function shall check the parameter RxParamIds for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.」()

[SWS_EthIf_00386]「
If development error detection is enabled: the function shall check the parameter ParamValues for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.」()

8.3.45 EthIf_ProvideTxBuffer

[SWS_EthIf_00067][

Service Name	EthIf_ProvideTxBuffer

Syntax	`BufReq_ReturnType EthIf_ProvideTxBuffer (`
	`  uint8 CtrlIdx,`
	`  Eth_FrameType FrameType,`
	`  uint8 Priority,`
	`  Eth_BufIdxType* BufIdxPtr,`
	`  uint8** BufPtr,`
	`  uint16* LenBytePtr`
	`)`

Service ID [hex]	0x09
Sync/Async	Synchronous

Reentrancy	Reentrant	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	FrameType	Ethernet Frame Type (EtherType)
	Priority	Priority value which shall be used for the 3-bit PCP field of the VLAN tag
Parameters (inout)	LenBytePtr	in: desired length in bytes, out: granted length in bytes
Parameters (out)	BufIdxPtr	Index to the granted buffer resource. To be used for subsequent requests
	BufPtr	Pointer to the granted buffer
Return value	BufReq_Return-Type	BUFREQ_OK: success BUFREQ_E_NOT_OK: development error detected BUFREQ_E_BUSY: all buffers in use BUFREQ_E_OVFL: requested buffer too large
Description	Provides access to a transmit buffer of the specified Ethernet controller.	
Available via	EthIf.h	

]()

[SWS_EthIf_00146] ⌈
If CtrlIdx refers to an EthIfCtrl where no EthIfVlanID is configured, the parameters FrameType and Priority are not used.⌋()

[SWS_EthIf_00147] ⌈
If VLAN is used
- EthIf shall increment the input desired length by 4 bytes before calling the Ethernet Driver module
- EthIf shall store the PCP (Priority parameter), CFI (always 0), VID (configured VLAN ID) and value of the FrameType parameter at the beginning of the buffer received from Eth_ProvideTxBuffer).
- EthIf shall increment the BufPtr by 4 bytes when returning the granted buffer
- EthIf shall decrement the output granted length by 4 bytes⌋()

[SWS_EthIf_00068] ⌈
If the latest accepted controller mode is equal to ETH_MODE_ACTIVE or ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST for the given EthIfController, then the function EthIf_ProvideTxBuffer shall forward the call to the respective Ethernet Controller Driver. Otherwise the function shall reject the request for a transmission buffer and return with E_NOT_OK. ⌋()

[SWS_EthIf_00069] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00070] ⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00071] ⌈
If development error detection is enabled: the function shall check the parameter BufIdxPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

[SWS_EthIf_00072] ⌈
If development error detection is enabled: the function shall check the parameter BufPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

[SWS_EthIf_00073] ⌈
If development error detection is enabled: the function shall check the parameter LenBytePtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.46 EthIf_Transmit

[SWS_EthIf_00075][

Service Name	EthIf_Transmit	

Syntax	`Std_ReturnType EthIf_Transmit (`
	`  uint8 CtrlIdx,`
	`  Eth_BufIdxType BufIdx,`
	`  Eth_FrameType FrameType,`
	`  boolean TxConfirmation,`
	`  uint16 LenByte,`
	`  const uint8* PhysAddrPtr`
	`)`	

Service ID [hex]	0x0a	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different buffer indexes and Ctrl indexes	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	BufIdx	Index of the buffer resource
	FrameType	Ethernet frame type
	TxConfirmation	Activates transmission confirmation
	LenByte	Data length in byte
	PhysAddrPtr	Physical target address (MAC address) in network byte order
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: success E_NOT_OK: transmission failed
Description	Triggers transmission of a previously filled transmit buffer	
Available via	EthIf.h	

]()

[SWS_EthIf_00250] [If CtrlIdx refers to an EthIfCtrl where an EthIfVlanID is configured, the parameters FrameType is not used, and 0x8100 is provided to Eth_Transmit
instead.]()

[SWS_EthIf_00076] [
If the latest accepted controller mode is equal to ETH_MODE_ACTIVE or ETH_MODE_ACTIVE_WITH_WAKEUP_REQUEST for the given EthIfController,

then the function EthIf_Transmit shall forward the call to the respective Ethernet Controller Driver. Otherwise the function shall reject the request for a transmission and return with E_NOT_OK.⌋()

[SWS_EthIf_00077]⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00078]⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00079]⌈
If development error detection is enabled: the function shall check the parameter BufIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_PARAM.⌋()

[SWS_EthIf_00080]⌈
If development error detection is enabled: the function shall check the parameter PhysAddrPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.47 EthIf_GetVersionInfo

[SWS_EthIf_00082]⌈

Service Name	EthIf_GetVersionInfo	
Syntax	`void EthIf_GetVersionInfo ( Std_VersionInfoType* VersionInfoPtr )`	
Service ID [hex]	0x0b	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	VersionInfoPtr	Version information of this module
Return value	None	
Description	Returns the version information of this module	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00127] ⌈ If development error detection is enabled: the function shall check the parameter VersionInfoPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.3.48 EthIf_GetSwitchPortMode

[SWS_EthIf_91107]⌈

Service Name	EthIf_GetSwitchPortMode	
Syntax	`Std_ReturnType EthIf_GetSwitchPortMode ( uint8 SwitchIdx, uint8 SwitchPortIdx, Eth_ModeType* PortModePtr )`	
Service ID [hex]	0x49	
Sync/Async	Synchronous /Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
Parameters (inout)	None	
Parameters (out)	PortModePtr	ETH_MODE_DOWN: The Ethernet switch port of the given Ethernet switch is disabled ETH_MODE_ACTIVE: The Ethernet switch port of the given Ethernet switch is enabled
Return value	Std_-Return-Type	E_OK: success E_NOT_OK: The mode of the indexed switch port could not be obtained, or the function is called in state ETHSWT_STATE_UNINIT or ETHSWT_STATE_INIT.
Description	Obtains the mode of the indexed switch port	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00415]⌈ The function EthIf_GetSwitchPortMode shall forward the call to function EthSwt_GetSwitchPortMode of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).⌋()

8.3.49 EthIf_GetTransceiverMode

[SWS_EthIf_91108]⌈

Service Name	EthIf_GetTransceiverMode
Syntax	`Std_ReturnType EthIf_GetTransceiverMode (`

	`uint8 TrcvIdx,`
	`Eth_ModeType* TrcvModePtr`
	`)`	

Service ID [hex]	0x4a	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	TrcvIdx	Index of the transceiver within the context of the Ethernet Transceiver Driver
Parameters (inout)	None	
Parameters (out)	TrcvModePtr	ETH_MODE_DOWN: the transceiver is disabled ETH_MODE_ ACTIVE: the transceiver is enable
Return value	Std_Return-Type	E_OK: success E_NOT_OK: transceiver could not be initialized
Description	Obtains the state of the indexed transceiver	
Available via	EthIf.h	

]()

[SWS_EthIf_00417][The function EthIf_GetTransceiverMode shall forward the call to function EthTrcv_GetTransceiverMode of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).]()

8.3.50 EthIf_SwitchPortGetLinkState

[SWS_EthIf_91109][

Service Name	EthIf_SwitchPortGetLinkState	

Syntax	`Std_ReturnType EthIf_SwitchPortGetLinkState (`
	`uint8 SwitchIdx,`
	`uint8 SwitchPortIdx,`
	`EthTrcv_LinkStateType* LinkStatePtr`
	`)`	

Service ID [hex]	0x4b	
Sync/Async	Synchronous /Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPort Idx	Index of the port at the addressed switch
Parameters (inout)	None	

Parameters (out)	LinkState Ptr	ETHTRCV_LINK_STATE_DOWN: Switch port is disconnected ETHTRCV_LINK_STATE_ACTIVE: Switch port is connected
Return value	Std_-Return-Type	E_OK: success E_NOT_OK: Link state of the indexed switch port could not be obtained, or the function is called in state ETHSWT_STATE_UNINIT or ETHSWT_STATE_INIT.
Description	Obtains the link state of the indexed switch port	
Available via	EthIf.h	

]()

[SWS_EthIf_00419][The function EthIf_SwitchPortGetLinkState shall forward the call to function EthSwt_GetLinkState of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.51 EthIf_TransceiverGetLinkState

[SWS_EthIf_91110][

Service Name	EthIf_TransceiverGetLinkState	

Syntax	`Std_ReturnType EthIf_TransceiverGetLinkState (`
	`  uint8 TrcvIdx,`
	`  EthTrcv_LinkStateType* LinkStatePtr`
	`)`	

Service ID [hex]	0x4c	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	TrcvIdx	Index of the transceiver within the context of the Ethernet Transceiver Driver
Parameters (inout)	None	
Parameters (out)	LinkState Ptr	ETHTRCV_LINK_STATE_DOWN: transceiver is disconnected ETHTRCV_LINK_STATE_ACTIVE: transceiver is connected
Return value	Std_-ReturnType	E_OK: success E_NOT_OK: transceiver could not be initialized
Description	Obtains the link state of the indexed transceiver	
Available via	EthIf.h	

]()

[SWS_EthIf_00421][The function EthIf_TransceiverGetLinkState shall forward the call to function EthTrcv_GetLinkState of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).]()

8.3.52 EthIf_SwitchPortGetBaudRate

[SWS_EthIf_91111]⌈

Service Name	EthIf_SwitchPortGetBaudRate	
Syntax	`Std_ReturnType EthIf_SwitchPortGetBaudRate ( uint8 SwitchIdx, uint8 SwitchPortIdx, EthTrcv_BaudRateType* BaudRatePtr )`	
Service ID [hex]	0x4d	
Sync/Async	Synchronous /Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	Switch PortIdx	Index of the port at the addressed switch
Parameters (inout)	None	
Parameters (out)	Baud RatePtr	ETHTRCV_BAUD_RATE_10MBIT: 10MBit connection ETHTRCV_ BAUD_RATE_100MBIT: 100MBit connection ETHTRCV_BAUD_ RATE_1000MBIT: 1000MBit connection ETHTRCV_BAUD_ RATE_2500MBIT: 2500MBit connection
Return value	Std_- Return- Type	E_OK: success E_NOT_OK: Baud rate of the indexed switch port could not be obtained, or the function is called in state ETHSWT_STATE_UNINIT or ETHSWT_ STATE_INIT.
Description	Obtains the baud rate of the indexed switch port	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00423]⌈ The function EthIf_SwitchPortGetBaudRate shall forward the call to function EthSwt_GetBaudRate of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).⌋()

8.3.53 EthIf_TransceiverGetBaudRate

[SWS_EthIf_91112]⌈

Service Name	EthIf_TransceiverGetBaudRate
Syntax	`Std_ReturnType EthIf_TransceiverGetBaudRate ( uint8 TrcvIdx, EthTrcv_BaudRateType* BaudRatePtr`

	)	
Service ID [hex]	0x4e	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	TrcvIdx	Index of the transceiver within the context of the Ethernet Transceiver Driver
Parameters (inout)	None	
Parameters (out)	Baud RatePtr	ETHTRCV_BAUD_RATE_10MBIT: 10MBit connection ETHTRCV_ BAUD_RATE_100MBIT: 100MBit connection ETHTRCV_BAUD_ RATE_1000MBIT: 1000MBit connection ETHTRCV_BAUD_ RATE_2500MBIT: 2500MBit connection
Return value	Std_- Return- Type	E_OK: success E_NOT_OK: transceiver could not be initialized
Description	Obtains the baud rate of the indexed transceiver	
Available via	EthIf.h	

]()

[SWS_EthIf_00426][The function EthIf_TransceiverGetBaudRate shall forward the call to function EthTrcv_GetBaudRate of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).]()

8.3.54 EthIf_SwitchPortGetDuplexMode

[SWS_EthIf_91113][

Service Name	EthIf_SwitchPortGetDuplexMode	
Syntax	`Std_ReturnType EthIf_SwitchPortGetDuplexMode ( uint8 SwitchIdx, uint8 SwitchPortIdx, EthTrcv_DuplexModeType* DuplexModePtr )`	
Service ID [hex]	0x4f	
Sync/Async	Synchronous /Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPort Idx	Index of the port at the addressed switch

Parameters (inout)	None	
Parameters (out)	Duplex ModePtr	ETHTRCV_DUPLEX_MODE_HALF: half duplex connections ETHTRCV_DUPLEXMODE_FULL: full duplex connection
Return value	Std_-Return-Type	E_OK: success E_NOT_OK: duplex mode of the indexed switch port could not be obtained, or the function is called in state ETHSWT_STATE_UNINIT or ETHSWT_STATE_INIT.
Description	Obtains the duplex mode of the indexed switch port	
Available via	EthIf.h	

]()

[SWS_EthIf_00428][The function EthIf_SwitchPortGetDuplexMode shall forward the call to function EthSwt_GetDuplexMode of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.55 EthIf_TransceiverGetDuplexMode

[SWS_EthIf_91114][

Service Name	EthIf_TransceiverGetDuplexMode	
Syntax	`Std_ReturnType EthIf_TransceiverGetDuplexMode ( uint8 TrcvIdx, EthTrcv_DuplexModeType* DuplexModePtr )`	
Service ID [hex]	0x50	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	TrcvIdx	Index of the transceiver within the context of the Ethernet Transceiver Driver
Parameters (inout)	None	
Parameters (out)	Duplex ModePtr	ETHTRCV_DUPLEX_MODE_HALF: half duplex connections ETHTRCV_DUPLEX_MODE_FULL: full duplex connection
Return value	Std_-ReturnType	E_OK: success E_NOT_OK: transceiver could not be initialized
Description	Obtains the duplex mode of the indexed transceiver	
Available via	EthIf.h	

]()

[SWS_EthIf_00430][The function EthIf_TransceiverGetDuplexMode shall forward the call to function EthTrcv_GetDuplexMode of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).]()

8.3.56 EthIf_SwitchPortGetCounterValues

[SWS_EthIf_91115][

Service Name	EthIf_SwitchPortGetCounterValues	
Syntax	`Std_ReturnType EthIf_SwitchPortGetCounterValues ( uint8 SwitchIdx, uint8 SwitchPortIdx, Eth_CounterType* CounterPtr )`	
Service ID [hex]	0x51	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
Parameters (inout)	None	
Parameters (out)	CounterPtr	counter values according to IETF RFC 1757, RFC 1643 and RFC 2233.
Return value	Std_ReturnType	E_OK: success E_NOT_OK: counter values read failure
Description	Reads a list with drop counter values of the corresponding port of the switch. The meaning of these values is described at Eth_CounterType.	
Available via	EthIf.h	

]()

[SWS_EthIf_00432][The function EthIf_SwitchPortGetCounterValues shall forward the call to function EthSwt_GetCounterValues of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.57 EthIf_SwitchPortGetRxStats

[SWS_EthIf_91116][

Service Name	EthIf_SwitchPortGetRxStats
Syntax	`Std_ReturnType EthIf_SwitchPortGetRxStats ( uint8 SwitchIdx, uint8 SwitchPortIdx,`

	`Eth_RxStatsType* RxStatsPtr` `)`	
Service ID [hex]	0x52	
Sync/Async	Synchronous /Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPort Idx	Index of the port at the addressed switch
Parameters (inout)	None	
Parameters (out)	RxStatsPtr	List of values according to IETF RFC 2819 (Remote Network Monitoring Management Information Base)
Return value	Std_Return-Type	E_OK: success E_NOT_OK: drop counter could not be obtained
Description	Returns a list of statistic counters defined with Eth_RxTatsType. The majority of these Counters are derived from the IETF RFC2819.	
Available via	EthIf.h	

]()

[SWS_EthIf_00434][The function EthIf_SwitchPortGetRxStats shall forward the call to function EthSwt_GetRxStats of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.58 EthIf_SwitchPortGetTxStats

[SWS_EthIf_91117][

Service Name	EthIf_SwitchPortGetTxStats	

Syntax	`Std_ReturnType EthIf_SwitchPortGetTxStats (`
	`  uint8 SwitchIdx,`
	`  uint8 SwitchPortIdx,`
	`  Eth_TxStatsType* TxStatsPtr`
	`)`	

Service ID [hex]	0x53	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	--
	SwitchPortIdx	Index of the port at the addressed switch
Parameters (inout)	None	

Parameters (out)	TxStatsPtr	List of values to read statistic values for transmission.
Return value	Std_ReturnType	E_OK: success E_NOTOK: Tx-statistics could not be obtained
Description	List of values to read statistic values for transmission.	
Available via	EthIf.h	

]()

[SWS_EthIf_00436][The function EthIf_SwitchPortGetTxStats shall forward the call to function EthSwt_GetTxStats of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.59 EthIf_SwitchPortGetTxErrorCounterValues

[SWS_EthIf_91118][

Service Name	EthIf_SwitchPortGetTxErrorCounterValues	
Syntax	`Std_ReturnType EthIf_SwitchPortGetTxErrorCounterValues ( uint8 SwitchIdx, uint8 SwitchPortIdx, Eth_TxErrorCounterValuesType* TxStatsPtr )`	
Service ID [hex]	0x54	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Drive
	SwitchPortIdx	Index of the port at the addressed switch
Parameters (inout)	None	
Parameters (out)	TxStatsPtr	List of values to read statistic error counter values for transmission.
Return value	Std_Return-Type	E_OK: success, E_NOTOK: Tx-statistics could not be obtained
Description	List of values to read statistic error counter values for transmission from.	
Available via	EthIf.h	

]()

[SWS_EthIf_00438][The function EthIf_SwitchPortGetTxErrorCounterValues shall forward the call to function EthSwt_GetTxErrorCounterValues of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.60 EthIf_SwitchPortGetMacLearningMode

[SWS_EthIf_91119][

Service Name	EthIf_SwitchPortGetMacLearningMode	
Syntax	`Std_ReturnType EthIf_SwitchPortGetMacLearningMode ( uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_MacLearningType* MacLearningModePtr )`	
Service ID [hex]	0x55	
Sync/Async	Synchronous /Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
Parameters (inout)	None	
Parameters (out)	MacLearning ModePtr	Defines whether MAC addresses shall be learned and if they shall be learned in software or hardware.
Return value	Std_ReturnType	E_OK: success E_NOT_OK: configuration could be persistently reset
Description	Returns the MAC learning mode, i.e. 1.) HW learning enabled, 2.) Hardware learning disabled, 3.) Software learning enabled. Note: This feature is hardware dependent, i.e. the switch hardware needs to support the different learning modes	
Available via	EthIf.h	

]()

[SWS_EthIf_00440][The function EthIf_SwitchPortGetMacLearningMode shall forward the call to function EthSwt_GetMacLearningMode of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.61 EthIf_GetSwitchPortIdentifier

[SWS_EthIf_91120][

Service Name	EthIf_GetSwitchPortIdentifier
Syntax	`Std_ReturnType EthIf_GetSwitchPortIdentifier ( uint8 SwitchIdx, uint8 SwitchPortIdx, uint32* OrgUniqueIdPtr, uint8* ModelNrPtr, uint8* RevisionNrPtr`

	)	
Service ID [hex]	0x56	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPort Idx	Index of the port at the addressed switch
Parameters (inout)	None	
Parameters (out)	OrgUnique IdPtr	Pointer to the memory where the Organizationally Unique Identifier (OUI) shall be stored.
	ModelNrPtr	Pointer to the memory where the Manufacturer's Model Number shall be stored.
	RevisionNr Ptr	Pointer to the memory where the Revision Number shall be stored.
Return value	Std_Return-Type	E_OK: organizationally unique identifier of the Ethernet transceiver could be read. E_NOT_OK: organizationally unique identifier of the Ethernet transceiver could not be obtained (i.e. OUI is not available).
Description	This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.	
Available via	EthIf.h	

]()

[SWS_EthIf_00442][The function EthIf_GetSwitchPortIdentifier shall forward the call to function EthSwt_GetPortIdentifier of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.62 EthIf_GetSwitchIdentifier

[SWS_EthIf_91121][

Service Name	EthIf_GetSwitchIdentifier

Syntax	`Std_ReturnType EthIf_GetSwitchIdentifier (`
	`  uint8 SwitchIdx,`
	`  uint32* OrgUniqueIdPtr`
	`)`

Service ID [hex]	0x57
Sync/Async	Synchronous

Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
Parameters (inout)	None	
Parameters (out)	OrgUniqueId Ptr	Pointer to the memory where the Organizationally Unique Identifier shall be stored.
Return value	Std_Return-Type	E_OK: organizationally unique identifier of the Ethernet switch could be read. E_NOT_OK: organizationally unique identifier of the Ethernet switch could not be read (i.e. no OUI is available for this Ethernet switch)
Description	Obtain the Organizationally Unique Identifier that is given by the IEEE of the indexed Ethernet switch. This function shall provide the OUI of Ethernet switch. The OUI has a size of 24 bit. If a ethernet switch can provide the OUI the 8 most significant bits of the OUI shall be set to 0x00xxxxxx. If a Ethernet switch can not provide the OUI the 8 most significant bits of the OUI shall be set to 0xFFxxxxxx.	
Available via	EthIf.h	

]()

[SWS_EthIf_00444][The function EthIf_GetSwitchIdentifier shall forward the call to function EthSwt_GetSwitchIdentifier of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.63 EthIf_WritePortMirrorConfiguration

[SWS_EthIf_91122][

Service Name	EthIf_WritePortMirrorConfiguration	
Syntax	`Std_ReturnType EthIf_WritePortMirrorConfiguration ( uint8 MirroredSwitchIdx, const EthSwt_PortMirrorCfgType* PortMirrorConfigurationPtr )`	
Service ID [hex]	0x58	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	MirroredSwitch Idx	Index of the switch within the context of the Ethernet Switch Driver, where the Ethernet switch port is located, that has to be mirrored
	PortMirror ConfigurationPtr	--
Parameters (inout)	None	

Parameters (out)	None	
Return value	Std_ReturnType	E_OK: the port mirror configuration for the indexed Ethernet switch port was written. E_NOT_OK: the port mirror configuration for the indexed Ethernet switch port was not written. (i.e. indexed ethernet switch is not available) ETHSWT_PORT_MIRRORING_CONFIGURATION_NOT_SUPPORTED: port mirroring configuration is not supported by Ethernet switch driver or by the Ethernet switch hardware
Description	Store the given port mirror configuration in a shadow buffer in the Ethernet switch driver for the given MirroredSwitchIdx.	
Available via	EthIf.h	

]()

[SWS_EthIf_00446][The function EthIf_WritePortMirrorConfiguration shall forward the call to function EthSwt_WritePortMirrorConfiguration of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.64 EthIf_ReadPortMirrorConfiguration

[SWS_EthIf_91123][

Service Name	EthIf_ReadPortMirrorConfiguration	
Syntax	`Std_ReturnType EthIf_ReadPortMirrorConfiguration ( uint8 MirroredSwitchIdx, EthSwt_PortMirrorCfgType* PortMirrorConfigurationPtr )`	
Service ID [hex]	0x59	
Sync/Async	Asynchronous Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	MirroredSwitchIdx	Index of the Ethernet switch within the context of the Ethernet Switch Driver, where the Ethernet switch ports are located, that have to be mirrored
Parameters (inout)	None	
Parameters (out)	PortMirror ConfigurationPtr	Pointer to the memory where the port configuration shall be stored.
Return value	Std_ReturnType	E_OK: the port mirror configuration for the indexed Ethernet switch port was red successfully. E_NOT_OK: the port mirror configuration for the indexed Ethernet switch was not red successfully. (i.e. indexed Ethernet switch is not available)
Description	Obtain the port mirror configuration of the given Ethernet switch.	

Available via	EthIf.h

]()

[SWS_EthIf_00448][The function EthIf_ReadPortMirrorConfiguration shall forward the call to function EthSwt_ReadPortMirrorConfiguration of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.65 EthIf_DeletePortMirrorConfiguration

[SWS_EthIf_91124][

Service Name	EthIf_DeletePortMirrorConfiguration	
Syntax	`Std_ReturnType EthIf_DeletePortMirrorConfiguration ( uint8 MirroredSwitchIdx )`	
Service ID [hex]	0x5a	
Sync/Async	Synchronous	
Reentrancy	Reentrant Reentrant for different MirroredSwitchIdx. Non reentrant for the same SwitchIdx.	
Parameters (in)	MirroredSwitch Idx	Index of the switch within the context of the Ethernet Switch Driver.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_Return-Type	E_OK: Port mirror configuration was deleted successfully E_NOT_OK: Port mirror configuration was not deleted successfully. (e.g. the port mirroring is enabled)
Description	Delete the stored port mirror configuration of the given MirroredSwitchIdx. If no port mirror configuration was found for the given MirroredSwitchIdx, the return value shall be E_OK.	
Available via	EthIf.h	

]()

[SWS_EthIf_00450][The function EthIf_DeletePortMirrorConfiguration shall forward the call to function EthSwt_DeletePortMirrorConfiguration of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.66 EthIf_GetPortMirrorState

[SWS_EthIf_91125][

Service Name	EthIf_GetPortMirrorState	
Syntax	`Std_ReturnType EthIf_GetPortMirrorState ( uint8 SwitchIdx, uint8 PortIdx, EthSwt_PortMirrorStateType* PortMirrorStatePtr )`	
Service ID [hex]	0x5b	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdx	Index of the port at the addressed switch
Parameters (inout)	None	
Parameters (out)	PortMirror StatePtr	Pointer to the memory where the port mirroring state (either PORT_ MIRRORING_ENABLED or PORT_MIRRORING_DISABLED)of the given Ethernet switch port shall be stored.
Return value	Std_- Return- Type	E_OK: the port mirroring state for the indexed Ethernet switch port returned successfully. E_NOT_OK: the port mirror configuration for the indexed Ethernet switch returned not successfully. (i.e. indexed ethernet switch port is not available)
Description	Obtain the current status of the port mirroring for the indexed Ethernet switch port	
Available via	EthIf.h	

]()

[SWS_EthIf_00452][The function EthIf_GetPortMirrorState shall forward the call to function EthSwt_GetPortMirrorState of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.67 EthIf_SetPortMirrorState

[SWS_EthIf_91126][

Service Name	EthIf_SetPortMirrorState
Syntax	`Std_ReturnType EthIf_SetPortMirrorState ( uint8 MirroredSwitchIdx, uint8 PortIdx, EthSwt_PortMirrorStateType PortMirrorState )`
Service ID [hex]	0x5c
Sync/Async	Synchronous

Reentrancy	Non Reentrant	
Parameters (in)	Mirrored SwitchIdx	Index of the Ethernet switch within the context of the Ethernet Switch Driver, where the port mirroring configuration is located that has to be enabled and disabled, repectively.
	PortIdx	Index of the port at the addressed switch
	PortMirror State	Contain the requested port mirroring state either PORT_ MIRRORING_ENABLED or PORT_MIRRORING_DISABLED
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_- Return- Type	Std_ReturnType E_OK: the requested port mirroring state for the indexed Ethernet switch port was set successfully. E_NOT_OK: the requested port mirroring state for the indexed Ethernet switch was not set successfully. (i.e. indexed Ethernet switch is not available, no port mirrior configuration is available)
Description	Request to set the given port mirroring state of the port mirror configuration for the given Ethernet switch.	
Available via	EthIf.h	

]()

[SWS_EthIf_00454][The function EthIf_SetPortMirrorState shall forward the call to function EthSwt_SetPortMirrorState of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.68 EthIf_SetPortTestMode

[SWS_EthIf_91127][

Service Name	EthIf_SetPortTestMode	
Syntax	`Std_ReturnType EthIf_SetPortTestMode ( uint8 SwitchIdx, uint8 PortIdx, EthTrcv_PhyTestModeType Mode )`	
Service ID [hex]	0x5d	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdx	Index of the port at the addressed switch
	Mode	Test mode to be activated

Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_-Return-Type	E_OK: the port test mode for the indexed Ethernet switch port was set successfully. E_NOT_OK: the port test mode for the indexed Ethernet switch was not set successfully. (i.e. indexed Ethernet switch port is not available)
Description	Activates a given test mode of the indexed Ethernet switch port.	
Available via	EthIf.h	

]()

[SWS_EthIf_00456][The function EthIf_SetPortTestMode shall forward the call to function EthSwt_SetPortTestMode of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.69 EthIf_SetPortLoopbackMode

[SWS_EthIf_91128][

Service Name	EthIf_SetPortLoopbackMode	
Syntax	`Std_ReturnType EthIf_SetPortLoopbackMode ( uint8 SwitchIdx, uint8 PortIdx, EthTrcv_PhyLoopbackModeType Mode )`	
Service ID [hex]	0x5e	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdx	Index of the port at the addressed switch
	Mode	Loop-back mode to be activated
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_-Return-Type	E_OK: the port mirroring loop-back back mode for the indexed Ethernet switch port was activated successfully. E_NOT_OK: the port mirroring loop-back back mode for the indexed Ethernet switch port was not activated successfully. (i.e. indexed Ethernet switch port is not available)

Description	Activates a given test loop-back mode of the indexed Ethernet switch port.
Available via	EthIf.h

]()

[SWS_EthIf_00458][The function EthIf_SetPortLoopbackMode shall forward the call to function EthSwt_SetPortLoopbackMode of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.70 EthIf_SetPortTxMode

[SWS_EthIf_91129][

Service Name	EthIf_SetPortTxMode	
Syntax	`Std_ReturnType EthIf_SetPortTxMode ( uint8 SwitchIdx, uint8 PortIdx, EthTrcv_PhyTxModeType Mode )`	
Service ID [hex]	0x5f	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdx	Index of the port at the addressed switch
	Mode	Transmission mode to be activated
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_-Return-Type	E_OK: the port Tx mode for the indexed Ethernet switch port was activated successfully. E_NOT_OK: the port Tx mode for the indexed Ethernet switch port was not activated successfully. (i.e. indexed Ethernet switch port is not available)
Description	Activates a given transmission mode of the indexed Ethernet switch port.	
Available via	EthIf.h	

]()

[SWS_EthIf_00460][The function EthIf_SetPortTxMode shall forward the call to function EthSwt_SetPortTxMode of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.71 EthIf_GetPortCableDiagnosticsResult

[SWS_EthIf_91130][

Service Name	EthIf_GetPortCableDiagnosticsResult	
Syntax	`Std_ReturnType EthIf_GetPortCableDiagnosticsResult ( uint8 SwitchIdx, uint8 PortIdx, EthTrcv_CableDiagResultType* ResultPtr )`	
Service ID [hex]	0x60	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdx	Index of the port at the addressed switch
Parameters (inout)	None	
Parameters (out)	ResultPtr	Pointer to the location where the cable diagnostics result shall be stored
Return value	Std_-Return-Type	E_OK:the port cable diagnostic result for the indexed Ethernet switch port was obtained successfully. E_NOT_OK: the port cable diagnostic result for the indexed Ethernet switch port was not obtained successfully. (i.e. indexed Ethernet switch port is not available)
Description	Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.	
Available via	EthIf.h	

]()

[SWS_EthIf_00462][The function EthIf_GetPortCableDiagnosticsResult shall forward the call to function EthSwt_GetPortCableDiagnosticsResult of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.72 EthIf_RunPortCableDiagnostic

[SWS_EthIf_91131][

Service Name	EthIf_RunPortCableDiagnostic
Syntax	`Std_ReturnType EthIf_RunPortCableDiagnostic ( uint8 SwitchIdx, uint8 PortIdx )`
Service ID	0x61

[hex]		
Sync/Async	Asynchronous Asynchronous	
Reentrancy	Reentrant Reentrant for different SwitchIdx and PortIdx. Non reentrant for the same SwitchIdx and PortIdx.	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver.
	PortIdx	Index of the port at the addressed switch.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	E_OK: The trigger to run the cable diagnostic has been accepted E_NOT_OK: The trigger to run the cable diagnostic has not been accepted
Description	Trigger the cable diagnostics of the given Ethernet Switch port (PortIdx) by calling EthTrcv_RunCableDiagnostic of the referenced Ethernet transceiver.	
Available via	EthIf.h	

]()

[SWS_EthIf_00464][If the function EthIf_RunPortCableDiagnostic is called, EthIf shall ensure that the corresponding EthIfController is in mode ETH_MODE_ACTIVE and forward the call to function EthSwt_RunPortCableDiagnostic of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.73 EthIf_RunCableDiagnostic

[SWS_EthIf_91132][

Service Name	EthIf_RunCableDiagnostic	
Syntax	`Std_ReturnType EthIf_RunCableDiagnostic ( uint8 TrcvIdx )`	
Service ID [hex]	0x62	
Sync/Async	Asynchronous Asynchronous	
Reentrancy	Reentrant Reentrant for different TrcvIdx. Non reentrant for the same TrcvIdx.	
Parameters (in)	TrcvIdx	Index of the Ethernet transceiver within the context of the Ethernet Transceiver Driver.
Parameters (inout)	None	
Parameters	None	

(out)		
Return value	Std_Return-Type	E_OK: The trigger has been accepted. E_NOT_OK: The trigger has not been accepted.
Description	Trigger the cable diagnostics for the given Ethernet transceiver.	
Available via	EthIf.h	

]()

[SWS_EthIf_00466][If the function EthIf_RunCableDiagnostic is called, EthIf shall ensure that the corresponding EthIfController is in mode ETH_MODE_ACTIVE and forward the call to function EthTrcv_RunCableDiagnostic of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).]()

8.3.74 EthIf_SwitchGetCfgDataRaw

[SWS_EthIf_91133][

Service Name	EthIf_SwitchGetCfgDataRaw	

Syntax	`Std_ReturnType EthIf_SwitchGetCfgDataRaw (`
	`  uint8 SwitchIdx,`
	`  uint32 Offset,`
	`  uint16 Length,`
	`  uint8* BufferPtr`
	`)`	

Service ID [hex]	0x63	
Sync/Async	Asynchronous Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the Ethernet switch within the context of the Ethernet Switch Driver
	Offset	Offset of the Ethernet switch memory from where the reading starts
	Length	Length of data in bytes that shall be copied
Parameters (inout)	None	
Parameters (out)	BufferPtr	Pointer to the location where the data shall be copied
Return value	Std_Return-Type	E_OK: the data read was triggered successfully E_NOT_OK: the data read was not triggered successfully (i.e. indexed Ethernet switch is not available)
Description	Retrieves the data in memory of the indexed Ethernet switch in variable length	
Available via	EthIf.h	

]()

[SWS_EthIf_00468][The function EthIf_SwitchGetCfgDataRaw shall forward the call to function EthSwt_GetCfgDataRaw of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.75 EthIf_SwitchGetCfgDataInfo

[SWS_EthIf_91134][

Service Name	EthIf_SwitchGetCfgDataInfo	
Syntax	`Std_ReturnType EthIf_SwitchGetCfgDataInfo ( uint8 SwitchIdx, uint32* DataSizePtr, uint32* DataAdressPtr )`	
Service ID [hex]	0x64	
Sync/Async	Asynchronous	
Reentrancy	Reentrant	
Parameters (in)	SwitchIdx	Index of the Ethernet switch within the context of the Ethernet Switch Driver
Parameters (inout)	None	
Parameters (out)	DataSizePtr	Pointer to the location where the total size of the configuration data shall be copied
	DataAdress Ptr	Pointer to the location where the start address of the configuration registers shall be copied
Return value	Std_Return-Type	E_OK: the data was obtained successfully E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
Description	Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.	
Available via	EthIf.h	

]()

[SWS_EthIf_00470][The function EthIf_SwitchGetCfgDataInfo shall forward the call to function EthSwt_GetCfgDataInfo of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.76 EthIf_SwitchPortGetMaxFIFOBufferFillLevel

[SWS_EthIf_91135][

Service Name	EthIf_SwitchPortGetMaxFIFOBufferFillLevel

Syntax	`Std_ReturnType EthIf_SwitchPortGetMaxFIFOBufferFillLevel ( uint8 SwitchPortIdx, uint8 PortIdx, uint8 SwitchPortEgressFifoIdx, uint32* SwitchPortEgressFifoBufferLevelPtr )`	
Service ID [hex]	0x65	
Sync/Async	Asynchronous	
Reentrancy	Reentrant Reentrant for different SwitchIdx and PortIdx. Non reentrant for the same SwitchIdx and PortIdx.	
Parameters (in)	SwitchPortIdx	Index of the Ethernet switch within the context of the Ethernet Switch Driver.
	PortIdx	Index of the Ethernet switch egress port at the addressed Ethernet switch.
	SwitchPortEgress FifoIdx	Index of the egress FIFO of the addressed Ethernet switch port
Parameters (inout)	None	
Parameters (out)	SwitchPortEgress FifoBufferLevelPtr	Pointer to a memory location, where the maximum amount of allocated FIFO buffer (in bytes) since the last read out shall be stored
Return value	Std_ReturnType	E_OK: success E_NOT_OK: The maximal FIFO buffer level could not be obtained
Description	The function retrieves the maximum amount of allocated FIFO buffer of the indexed Ethernet switch egress port. If the Ethernet switch hardware does not support Ethernet switch port based maximal FIFO buffer level, the content of SwitchPort EgressFifoBufferLevelPtr shall be set to 0xFFFFFFFF. This API may be called by e.g. a CDD.	
Available via	EthIf.h	

]()

[SWS_EthIf_00472][The function EthIf_SwitchPortGetMaxFIFOBufferFillLevel shall forward the call to function EthSwt_GetMaxFIFOBufferFillLevel of the corresponding Ethernet Switch Driver (EthIfSwitchIdx).]()

8.3.77 EthIf_TransceiverGetMacMethod

[SWS_EthIf_91021][

Service Name	EthIf_TransceiverGetMacMethod
Syntax	`Std_ReturnType EthIf_TransceiverGetMacMethod ( uint8* TrcvIdx, EthTrcv_MacMethodType* MacModePtr`

	)	
Service ID [hex]	0x66	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	TrcvIdx	Index of the transceiver within the context of the Ethernet Transceiver Driver
Parameters (inout)	None	
Parameters (out)	MacMode Ptr	ETHTRCV_MAC_TYPE_CSMA_CD: Carrier-sense multiple access with collicion detection. ETHTRCV_MAC_TYPE_PLCA: Physical layer collision avoidance.
Return value	Std_-Return-Type	E_OK: success. E_NOT_OK: transceiver request has not been accepted.
Description	Obtains the media access mode of the transceiver.	
Available via	EthIf.h	

](SRS_Eth_00117)

[SWS_EthIf_00474][The function EthIf_TransceiverGetMacMethod shall forward the call to function EthTrcv_GetMacMethod of the corresponding Ethernet Transceiver Driver (EthIfTransceiverIdx).](SRS_Eth_00117)

[SWS_EthIf_00475] [If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.]()

[SWS_EthIf_00476] [If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.]()

[SWS_EthIf_00477] [If development error detection is enabled: the function shall check the parameter MacModePtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.]()

8.3.78 EthIf_EthGetSpiStatus

[SWS_EthIf_91022]{DRAFT} [

Service Name	EthIf_EthGetSpiStatus (draft)
Syntax	`Std_ReturnType EthIf_EthGetSpiStatus (` `  uint8* CtrlIdx,`

	Eth_SpiStatusType* SpiStatusPtr)	
Service ID [hex]	0x6a	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the controller within the context of the Ethernet controller Driver
Parameters (inout)	None	
Parameters (out)	SpiStatusPtr	Status of the SPI interface
Return value	Std_Return-Type	E_OK: success. E_NOT_OK: Controller request has not been accepted.
Description	When MACPHY controller are used, obtains the SPI interface status. **Tags:** atp.Status=draft	
Available via	EthIf.h	

]()

[SWS_EthIf_00505] **DRAFT** ⌈ The function EthIf_EthGetSpiStatus shall forward the call to function Eth_GetSpiStatus of the corresponding Ethernet Driver (CtrlIdx).⌋()

[SWS_EthIf_00506] **DRAFT** ⌈ If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00507] **DRAFT** ⌈ If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00508] **DRAFT** ⌈ If development error detection is enabled: the function shall check the parameter SpiStatusPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

8.4 Callback notifications

This is a list of functions provided for other modules.

8.4.1 EthIf_RxIndication

[SWS_EthIf_00085]⌈

Service Name	EthIf_RxIndication
Syntax	void EthIf_RxIndication (uint8 CtrlIdx,

	`Eth_FrameType FrameType,`
	`boolean IsBroadcast,`
	`const uint8* PhysAddrPtr,`
	`const Eth_DataType* DataPtr,`
	`uint16 LenByte`
	`)`	

Service ID [hex]	0x10	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	CtrlIdx	Index of the physical Ethernet controller within the context of the Ethernet Interface
	FrameType	Frame type of received Ethernet frame
	Is Broadcast	parameter to indicate a broadcast frame
	PhysAddr Ptr	Pointer to Physical source address (MAC address in network byte order) of received Ethernet frame
	DataPtr	Pointer to payload of received Ethernet frame.
	LenByte	Length (bytes) of the payload in received frame.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Handles a received frame received by the indexed controller	
Available via	EthIf.h	

]()

[SWS_EthIf_00086] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00087] ⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00088] ⌈
If development error detection is enabled: the function shall check the parameter DataPtr for being valid. If the check fails, the function shall raise the development error ETHIF_E_PARAM_POINTER.⌋()

[SWS_EthIf_00151] ⌈
The Ethernet Driver shall indicate broadcast message with the parameter 'IsBroadcast' to the Ethernet Interface.⌋()

[SWS_EthIf_00145] ⌈
If the VLAN is not active the Ethernet Interface shall increment the corresponding measurement data and filter the message⌋()

8.4.2 EthIf_TxConfirmation

[SWS_EthIf_00091]⌈

Service Name	EthIf_TxConfirmation	
Syntax	`void EthIf_TxConfirmation ( uint8 CtrlIdx, Eth_BufIdxType BufIdx, Std_ReturnType Result )`	
Service ID [hex]	0x11	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	Ctrl Idx	Index of the physical Ethernet controller within the context of the Ethernet Interface
	BufIdx	Index of the transmitted buffer
	Result	E_OK: The transmission was successful, E_NOT_OK: The transmission failed.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Confirms frame transmission by the indexed controller	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00255]⌈
EthIf_TxConfirmation shall pass the Result received within EthIf_TxConfirmation to the configured upper layer via <UL>_TxConfirmation.⌋()

[SWS_EthIf_00092] ⌈
If development error detection is enabled: the function shall check that the service EthIf_Init was previously called. If the check fails, the function shall raise the development error ETHIF_E_UNINIT.⌋()

[SWS_EthIf_00093] ⌈
If development error detection is enabled: the function shall check the parameter CtrlIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_CTRL_IDX.⌋()

[SWS_EthIf_00094] ⌈
If development error detection is enabled: the function shall check the parameter BufIdx for being valid. If the check fails, the function shall raise the development error ETHIF_E_INV_PARAM.⌋()

8.4.3 EthIf_CtrlModeIndication

[SWS_EthIf_00231]⌈

Service Name	EthIf_CtrlModeIndication	
Syntax	`void EthIf_CtrlModeIndication ( uint8 CtrlIdx, Eth_ModeType CtrlMode )`	
Service ID [hex]	0x0e	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant for the same CtrlIdx, reentrant for different	
Parameters (in)	CtrlIdx	Index of the physical Ethernet controller within the context of the Ethernet Interface
	CtrlMode	Notified Ethernet controller mode
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Called asynchronously when mode has been read out. Triggered by previous Eth_SetControllerMode call. Can directly be called within the trigger functions.	
Available via	EthIf.h	

⌋()

[SWS_EthIf_00252] ⌈
The function shall call EthSM_CtrlModeIndication.⌋()

8.4.4 EthIf_TrcvModeIndication

[SWS_EthIf_00232]⌈

Service Name	EthIf_TrcvModeIndication	

Syntax	`void EthIf_TrcvModeIndication (`
	`  uint8 TrcvIdx,`
	`  Eth_ModeType TrcvMode`
	`)`	

Service ID [hex]	0x0f	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant for the same CtrlIdx, reentrant for different	
Parameters (in)	TrcvIdx	Index of the Ethernet transceiver within the context of the Ethernet Interface
	TrcvMode	Notified Ethernet transceiver mode
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Called asynchronously when a mode change has been read out. If the function is triggered by previous call of EthTrcv_SetTransceiverMode it can directly be called within the trigger function.	
Available via	EthIf.h	

]()

8.4.5 EthIf_SwitchPortModeIndication

[SWS_EthIf_91055][

Service Name	EthIf_SwitchPortModeIndication	

Syntax	`void EthIf_SwitchPortModeIndication (`
	`  uint8 SwitchIdx,`
	`  uint8 SwitchPortIdx,`
	`  Eth_ModeType PortMode`
	`)`	

Service ID [hex]	0x46	
Sync/Async	Asynchronous	
Reentrancy	Non Reentrant	
Parameters (in)	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch.

	PortMode	Notified Ethernet Switch port mode.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	The EthIf shall determine the expected notifications based on the EthSwtPort configuration. In case the EthSwtPort references an EthTrcv the EthIf expects a notification from the EthTrcv via API EthIf_TrcvModeIndication(). Otherwise the EthIf expects a notification from the EthSwt via API EthIf_SwitchPortModeIndication()	
Available via	EthIf.h	

]()

8.4.6 EthIf_SleepIndication

[SWS_EthIf_91006]{DRAFT} [

Service Name	EthIf_SleepIndication (draft)	

Syntax	`void EthIf_SleepIndication (`
	`  uint8 TrcvIdx`
	`)`	

Service ID [hex]	0x68	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	TrcvIdx	Index of the Ethernet transceiver within the context of the Ethernet Interface
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	This API is called by the corresponding EthTrcv, if a sleep indication was detected on the network. This could be used e.g. for Ethernet hardware which is compliant to the OA TC10. In this case the Ethernet hardware (PHY) detect an Sleep.Indication which was triggered by a Sleep.Request of the connected link partner. **Tags:** atp.Status=draft	
Available via	EthIf.h	

](SRS_Eth_00156)

[SWS_EthIf_00497] DRAFT ⌈
The function shall call EthSM_SleepIndication with the corresponding EthIfCtrl.⌋(SRS_Eth_00156)

8.5 Scheduled functions

8.5.1 EthIf_MainFunctionRx

[SWS_EthIf_00097]⌈

Service Name	EthIf_MainFunctionRx
Syntax	`void EthIf_MainFunctionRx ( void )`
Service ID [hex]	0x20
Description	The function checks for new received frames and issues reception indications in polling mode.
Available via	SchM_EthIf.h

⌋()

[SWS_EthIf_00099] ⌈
The receive frame check shall be pre compile time configurable On/Off by the configuration parameter: EthIfEnableRxInterrupt.⌋()

8.5.2 EthIf_MainFunctionRx_<PriorityProcessing ShortName>

[SWS_EthIf_91051]⌈

Service Name	EthIf_MainFunctionRx_<PriorityProcessing ShortName>
Syntax	`void EthIf_MainFunctionRx_<PriorityProcessing ShortName> ( void )`
Service ID [hex]	0x42
Description	The function checks for new received frames at the related Ethernet controller and reception queue by calling Eth_Receive() with the respective FifoIdx. EthIf_Main FunctionRx shall receive frames from all FIFOs that are not assigned for processing via EthIfPhysCtrlRxMainFunctionPriorityProcessing.
Available via	EthIf_SchM.h

](

8.5.3 EthIf_MainFunctionTx

[SWS_EthIf_00113][

Service Name	EthIf_MainFunctionTx

Syntax	`void EthIf_MainFunctionTx (`
	`  void`
	`)`

Service ID [hex]	0x21
Description	The function issues transmission confirmations in polling mode. It checks also for transceiver state changes.
Available via	SchM_EthIf.h

](

[SWS_EthIf_00100] [
The transmission confirmation check shall be pre compile time configurable On/Off by the configuration parameter: EthIfEnableTxInterrupt.]()

[SWS_EthIf_00101] [
The frequency of polling the transceiver state change shall be configurable by the configuration parameter: EthIfTrcvLinkStateChgMainReload.]()

8.5.4 EthIf_MainFunctionState

[SWS_EthIf_91104][

Service Name	EthIf_MainFunctionState

Syntax	`void EthIf_MainFunctionState (`
	`  void`
	`)`

Service ID [hex]	0x05
Sync/Async	Asynchronous
Reentrancy	Non Reentrant
Parameters (in)	None
Parameters (inout)	None
Parameters (out)	None

Return value	None
Description	The function is polling different communication hardware (Ethernet transceiver, Ethernet switch ports) related information, e.g. link state, signal quality.
Available via	EthIf_SchM.h

⌋()

[SWS_EthIf_00407]⌈
The function EthIf_MainFunctionState shall poll Ethernet communication hardware related information with the period of EthIfMainFunctionStatePeriod.⌋()

[SWS_EthIf_00408]⌈
For each Ethernet switch port where a link state ETHTRCV_LINK_STATE_ACTIVE is yielded and references an Ethernet Transceiver the function shall poll the signal quality by calling EthSwt_GetPortSignalQuality().⌋()

[SWS_EthIf_00409]⌈
For each Ethernet transceiver where a link state of ETHTRCV_LINK_STATE_ACTIVE is yielded the function shall poll the signal quality by calling EthTrcv_GetPhySignalQuality().⌋()

[SWS_EthIf_00410]⌈
The obtained signal quality value shall be stored as type of EthIf_SignalQualityResultType. The value shall always be stored as ActualSignalQuality. If the obtained signal quality is higher than the stored highest signal quality (HighestSignalQuality), then HighestSignalQuality shall be updated with the obtained signal quality. If the obtained signal quality is lower than the lowest signal quality (LowestSignalQuality), then LowestSignalQuality shall be updated with the obtained signal quality.⌋()

[SWS_EthIf_00498] DRAFT ⌈
EthIf shall check its maintained Ethernet hardware (Ethernet switch port, Ethernet transceiver), if the Ethernet hardware has reached the requested mode and requested link state under the following conditions:

- the timer to switch off the EthSwtPort (see EthIfSwitchOffPortTimeDelay) is not running AND
- the timer to keep the EthSwtPort in ETH_MODE_ACTIVE (see EthIfPortStartupActiveTime) is not running and the EthSwtPort has not been requested with ETH_MODE_ACTIVE

If EthIf detects that the requested mode and / or requested link state has not reached, EthIf shall re-trigger the requested mode and link state, respectively.⌋()

Note:

1. This shall ensure to re-trigger a wake-up on the network, if e.g. OA TC10 compliant hardware is used (see [25]).
2. Additionally, the check shall not try to re-establish a requested mode if the timer to switch off the EthSwtPort (requested via EthIfSwitchOffPortTimeDelay) or the timer to keep the EthSwtPort active (requested via EthIfPortStartupActiveTime) is running. Switching-off of the

Ethernet hardware in an Ethernet switched network after EthIfSwitchOffPortTimeDelay expires, lead to a situation that an Ethernet switch port and the connected Ethernet hardware (PHY) of the link partner are not synchronized. Thus, first the connected PHY will be switched off and after EthIfSwitchOffPortTimeDelay the Ethernet switch port. This is acceptable since the network management has already confirmed to go to sleep. For example, if using OA TC10 compliant Ethernet hardware, the ECU which is connected to the Ethernet switch trigger a Sleep.Request on the network and bring the connected Ethernet switch ports and its own Ethernet hardware to sleep mode, due to the specified OA TC10 synchronized shutdown of the Ethernet hardware. Thus, the ECU that maintain the Ethernet switch may detect a link down on the affected Ethernet switch port, which should be ignored by the EthIf, if the switch-off of the Ethernet switch port was already triggered but not forwarded to the Ethernet switch.

[SWS_EthIf_00499] DRAFT ⌈
For EthIfTransceiver where the referenced EthTrcv is acting as a passive communication slave (EthTrcvActAsSlavePassiveEnabled set to TRUE), EthIf shall check for unexpected link down. If an unexpected link down (link state is requested with ETHTRCV_LINK_STATE_ACTIVE, but current link state is ETHTRCV_LINK_STATE_DOWN) lasts as long as specified in EthIfQualifiedUnexptecedLinkDownTime, EthIf shall trigger to release the affected communication channel by calling EthSM_SleepIndication. If an unexpected link down was detected, the EthSM shall immediatedly be indicated via EthSM_TrcvLinkStateChg without considering EthIfQualifiedUnexpectedLinkDownTime. ⌋(SRS_Eth_00156)

Note: [SWS_EthIf_00499] should grant that a communication channel that act as an passive communication channel will shutdown even though the communiation master could not transmit a sleep over the network (e.g. hardware failure, unexpected shutdown of the ECU that act as communication master, a.s.o).

8.6 Expected Interfaces

This chapter lists all interfaces required from other modules.

8.6.1 Mandatory Interfaces

This chapter defines all interfaces required to fulfill the core functionality of the module.

8.6.2 Optional Interfaces

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

[SWS_EthIf_00103]⌈

API Function	*Header*	*Description*

	File	
BswM_EthIf_-PortGroupLink-StateChg	BswM_ EthIf.h	Function called by EthIf to indicate the link state change of a certain Ethernet switch port group.
Eth_Get-ControllerMode	Eth.h	Obtains the communication state of the indexed controller
Eth_GetPhys-Addr	Eth.h	Obtains the physical source address used by the indexed controller
Eth_ProvideTx-Buffer	Eth.h	Provides access to a transmit buffer of the FIFO related to the specified priority
Eth_ReadMii	Eth.h	Reads a transceiver register
Eth_Receive	Eth.h	Receive a frame from the related fifo.
Eth_Set-ControllerMode	Eth.h	Enables / Disables Rx/Tx communication of the indexed controller
Eth_Transmit	Eth.h	Triggers transmission of a previously filled transmit buffer
Eth_Tx-Confirmation	Eth.h	Triggers frame transmission confirmation
Eth_WriteMii	Eth.h	Configures a transceiver register or triggers a function offered by the receiver
EthSM_Ctrl-ModeIndication	EthSM.h	Called when mode has been read out. Either triggered by previous Eth If_GetControllerMode or by EthIf_SetControllerMode call. Can directly be called within the trigger functions.
EthSM_Sleep-Indication (draft)	EthSM.h	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
EthSM_Trcv-LinkStateChg	EthSM.h	This service is called by the Ethernet Interface to report a transceiver link state change.
EthSwt_Port-EnableTime-Stamp	Eth Swt.h	Activates egress time stamping on a dedicated message object on a dedicated port of a Switch if EthSwtPortTimeStampSupport is set to TRUE for this port. The selective activation of dedicated message objects for time stamping reduces the number of notification calls only to the required calls. Some HW does store once the egress time stamp marker and some HW needs it always before transmission. There will be no disabled functionality, due to the fact, that the message type is always "time stamped" by network design.
EthSwt_Set-MgmtInfo	Eth Swt.h	Extends the Ethernet frame prepared previously by EthSwt_EthTx PrepareFrame() with the management information to achieve transmission only on specific ports.

EthTrcv_Get-BaudRate	Eth Trcv.h	Obtains the baud rate of the indexed transceiver
EthTrcv_Get-DuplexMode	Eth Trcv.h	Obtains the duplex mode of the indexed transceiver
EthTrcv_Get-LinkState	Eth Trcv.h	Obtains the link state of the indexed transceiver
EthTrcv_Get-Transceiver-Mode	Eth Trcv.h	Obtains the state of the indexed transceiver
EthTrcv_Set-Transceiver-Mode	Eth Trcv.h	Enables / disables the indexed transceiver
EthTrcv_Start-AutoNegotiation	Eth Trcv.h	Restarts the negotiation of the transmission parameters used by the indexed transceiver
IdsM_Set-SecurityEvent	IdsM.h	This API is the application interface to report security events to the Ids M.
IdsM_Set-SecurityEvent-WithContextData	IdsM.h	This API is the application interface to report security events with context data to the IdsM.
WEth_GetBufW-RxParams	WEth.h	Read out values related to the receive direction for a received packet. For example, this could be RSSI or Channel belonging to one single packet.This API is valid only within the context of WEth_Receive
WEth_GetBufW-TxParams	WEth.h	Read out values related to the transmit direction for a transmitted packet. For example, this could be transaction ID belonging to one single packet.This API is valid only within the context of WEth_Tx Confirmation.
WEth_SetBufW-TxParams	WEth.h	Set values related to the transmit direction for a specific buffer (packet to be sent). For example, this can be the desired transmit power or the channel belonging to one single packet.
WEthTrcv_Get-ChanRxParams	WEth Trcv.h	Read values related to the receive direction of the transceiver. For example, this could be a Channel Busy Ratio (CBR) or the average Channel Idle Time (CIT).
WEthTrcv_Set-ChanRxParams	WEth Trcv.h	Set values related to the receive direction of a transceiver's wireless channel.For example, this could be a channel parameter like the frequency.
WEthTrcv_Set-ChanTxParams	WEth Trcv.h	Set values related to the transmit direction of a transceiver's wireless channel. For example, this could be the bitrate of a channel.
WEthTrcv_Set-RadioParams	WEth Trcv.h	Set values related to a transceiver's wireless radio. For example, this could be the selection of the radio settings (channel, ...).

](

8.6.3 Configurable interfaces

This chapter lists all interfaces with configurable target functions. The target function is usually a callback function. The function names are configurable.

[SWS_EthIf_00104]⌈

Service Name	<User>_RxIndication	

Syntax	`void <User>_RxIndication (`
	`  uint8 CtrlIdx,`
	`  Eth_FrameType FrameType,`
	`  boolean IsBroadcast,`
	`  const uint8* PhysAddrPtr,`
	`  const uint8* DataPtr,`
	`  uint16 LenByte`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Dont care	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	FrameType	frame type of received Ethernet frame
	Is Broadcast	parameter to indicate a broadcast frame
	PhysAddr Ptr	pointer to Physical source address (MAC address in network byte order) of received Ethernet frame
	DataPtr	Pointer to payload of the received Ethernet frame (i.e. Ethernet header is not provided).
	LenByte	Length of received data.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Indicates the reception of an Ethernet frame	
Available via	configurable	

⌋()

[SWS_EthIf_00105] ⌈
The callback function shall be configurable by the configuration parameter: EthIfRxIndicationFunction.⌋()

[SWS_EthIf_00106]⌈

Service Name	<UL>_TxConfirmation	

Syntax	`void <UL>_TxConfirmation (`
	`  uint8 CtrlIdx,`
	`  Eth_BufIdxType BufIdx,`
	`  Std_ReturnType Result`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Dont care	
Parameters (in)	Ctrl Idx	Index of the Ethernet controller within the context of the Ethernet Interface
	BufIdx	Index of the buffer resource
	Result	--
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Confirms the transmission of an Ethernet frame	
Available via	configurable	

]()

[SWS_EthIf_00107] [
The callback function shall be configurable by the configuration parameter: EthIfTxConfirmationFunction.]()

[SWS_EthIf_00108][

Service Name	<User>_TrcvLinkStateChg	

Syntax	`void <User>_TrcvLinkStateChg (`
	`  uint8 CtrlIdx,`
	`  EthTrcv_LinkStateType TrcvLinkState`
	`)`	

Sync/Async	Synchronous	
Reentrancy	Don't care	
Parameters (in)	CtrlIdx	Index of the Ethernet controller within the context of the Ethernet Interface
	TrcvLink State	ETHTRCV_LINK_STATE_DOWN transceiver link is down ETHTRCV_LINK_STATE_ACTIVE transceiver link is up
Parameters (inout)	None	
Parameters	None	

(out)	
Return value	None
Description	Indicates the change of a transceiver state
Available via	configurable

]()

[SWS_EthIf_00109] [
The callback function shall be configurable by the configuration parameter: EthIfTrcvLinkStateChgFunction.]()

[SWS_EthIf_00229] [
EthIfControllers not referring to an Ethernet Transceiver, i.e. no valid EthIfEthTrcvRef is configured, shall act as if the transceiver was present and the transceiver status was ETHTRCV_LINK_STATE_ACTIVE.]()

[SWS_EthIf_00230] [
Upon change of link state <User>_TrcvLinkStateChg shall be invoked for every affected EthIfController.]()

Terms and definitions:
Reentrant: interface is reentrant
Don't care: reentrancy of interface not relevant for this module (in general it is in this case not reentrant).

9 Sequence diagrams

The sequence diagrams show the basic operations carried out during operation. They show the interaction of the Ethernet Interface with upper layer BSW module and the underlying Ethernet Controller Driver.

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

9.1 Initialization

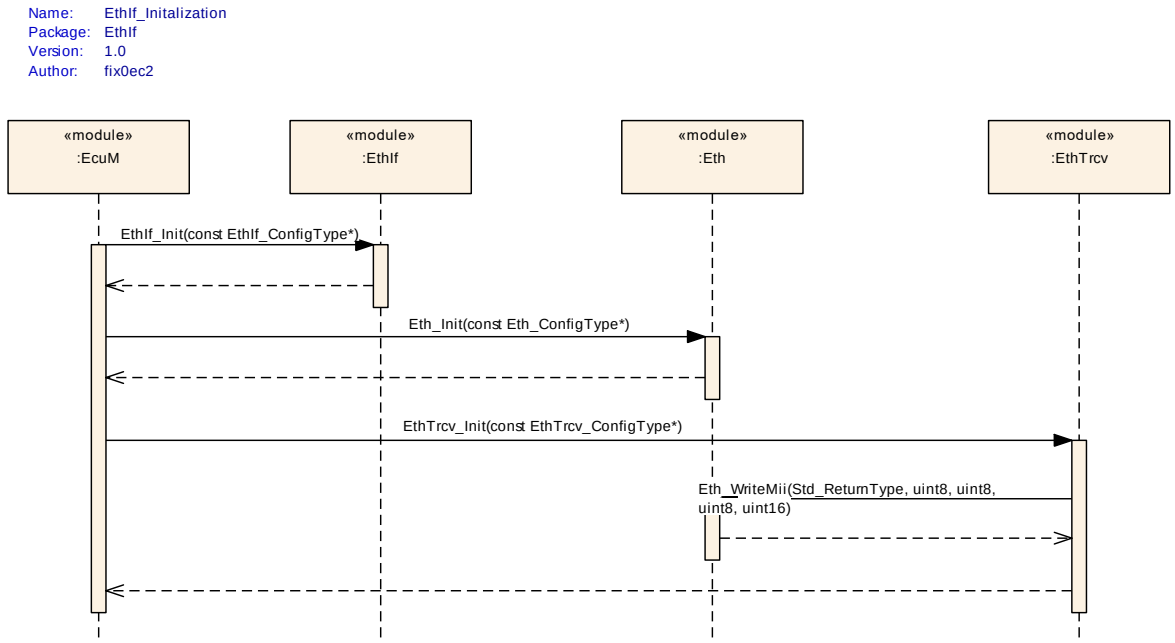

Figure 4: Initialization

9.2 Communication Initialization

Name: EthIf_CommunicationInitialization
Package: EthIf
Version: 1.0
Author: fix0ec2

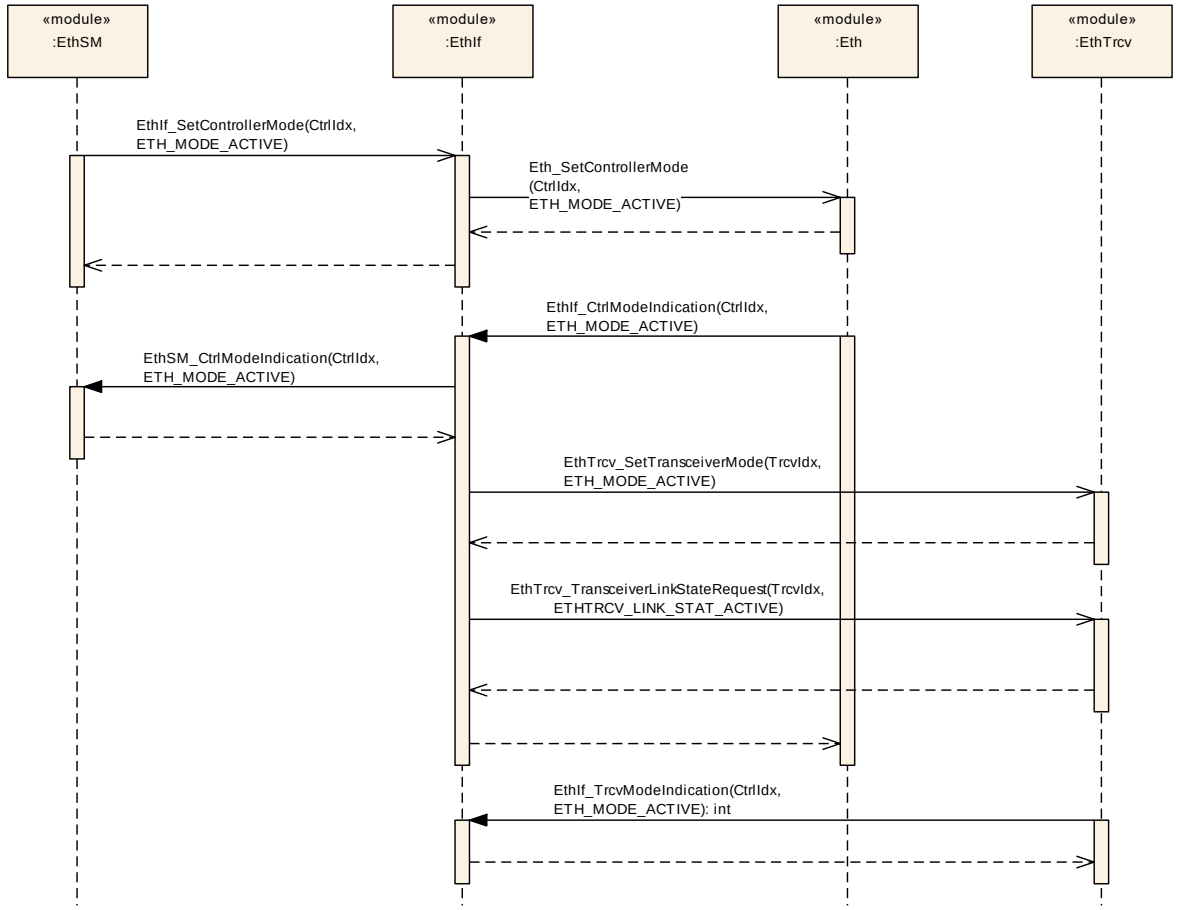

Figure 5: Communication Initialization

9.3 Switch Initialization

Name: EthIf_SwitchInitalization
Package: EthIf
Version: 1.0
Author: fix0ec2

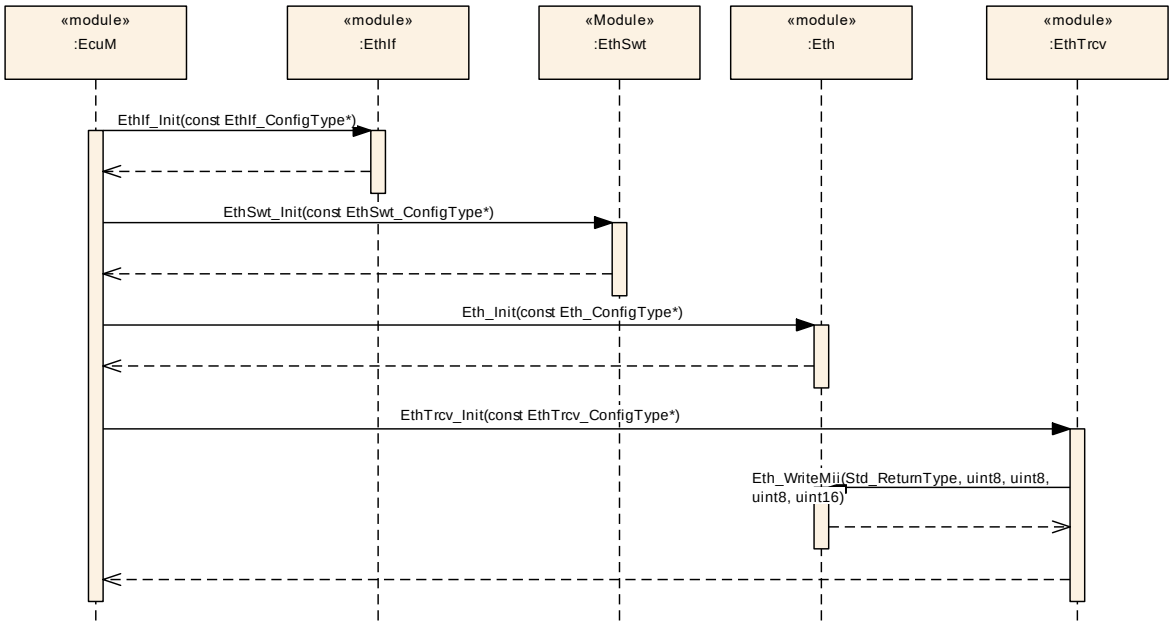

Figure 6: Switch Initialization

9.4 Data Transmission

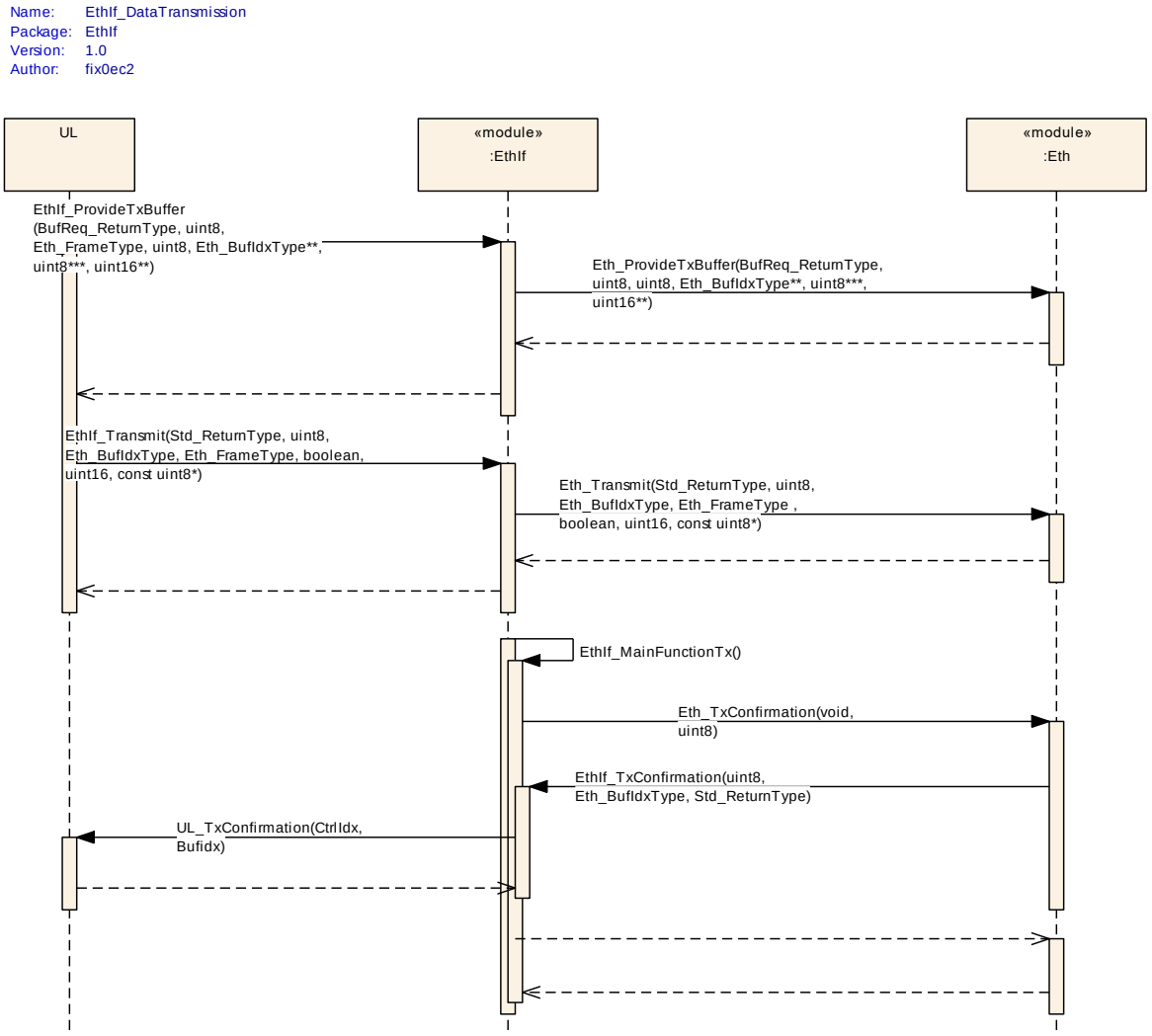

Figure 7: Frame Transmission in Polling Mode

[SWS_EthIf_00115]
In each call of EthIf_MainFunctionTx the component shall call Eth_TxConfirmation for all Ethernet Controller Drivers.
Note: The Ethernet Interface expects that each Ethernet Controller Driver issues confirmations for all transmitted frames using the call-back function EthIf_TxConfirmation.

[SWS_EthIf_00125]
EthIf_ TxConfirmation shall forward the confirmation to the registered call-back functions <User>_TxConfirmation.

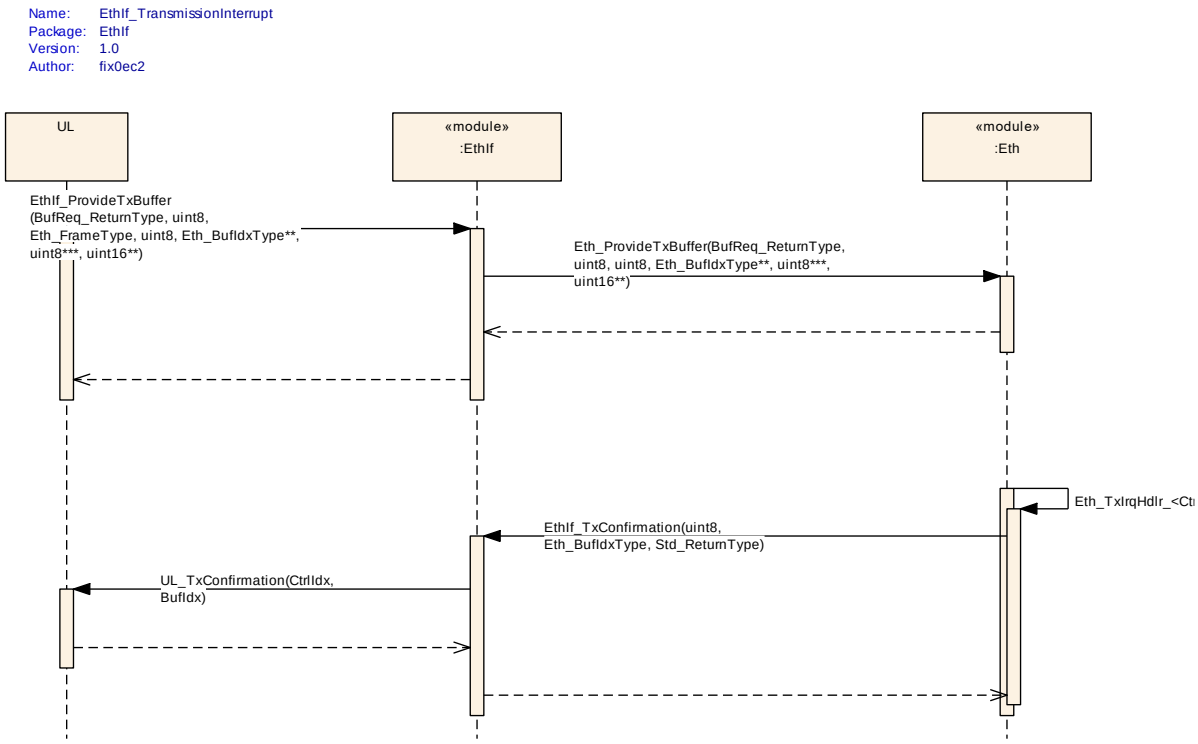

Figure 8: Frame Transmission in Interrupt Mode

9.5 Data Reception

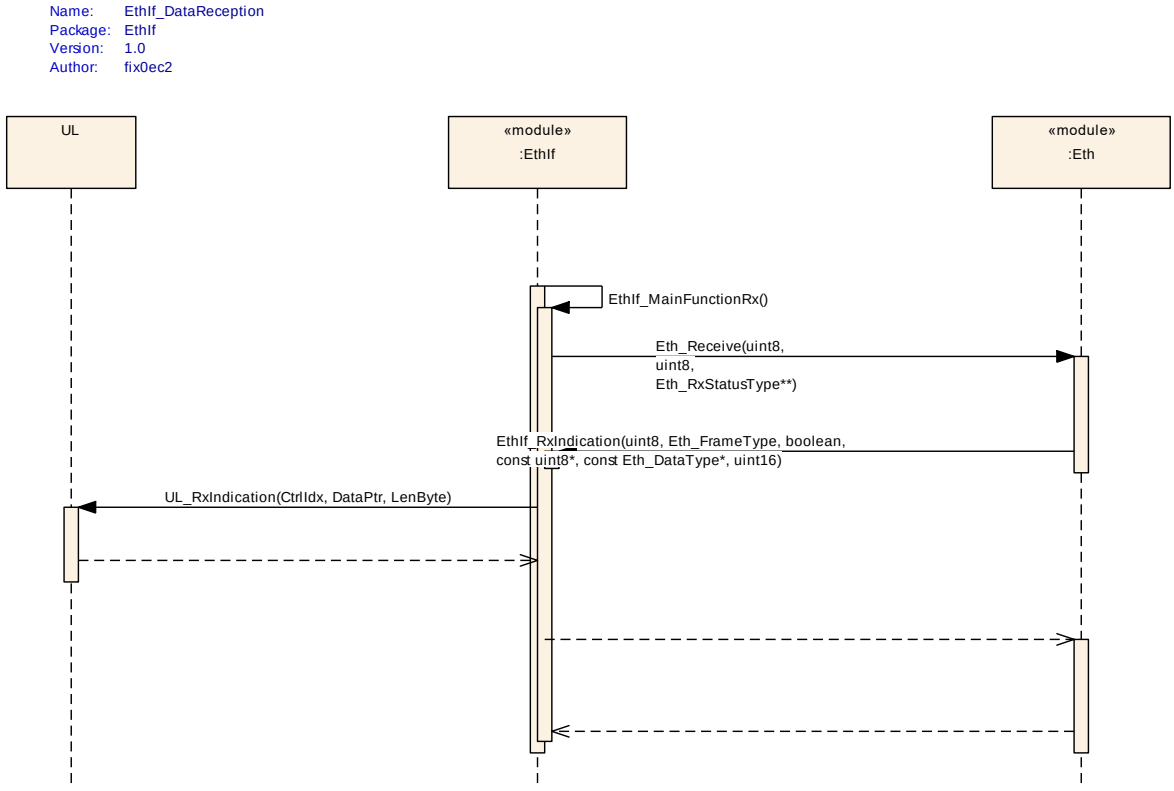

Figure 9: Frame Reception in Polling Mode

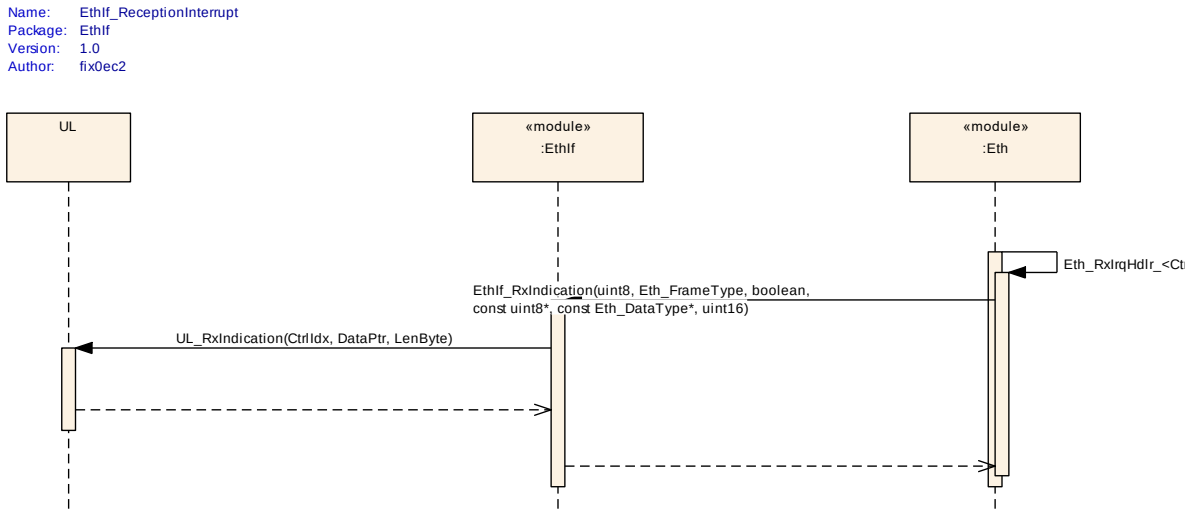

Figure 10: Frame Reception in Interrupt Mode

9.6 Link State Change

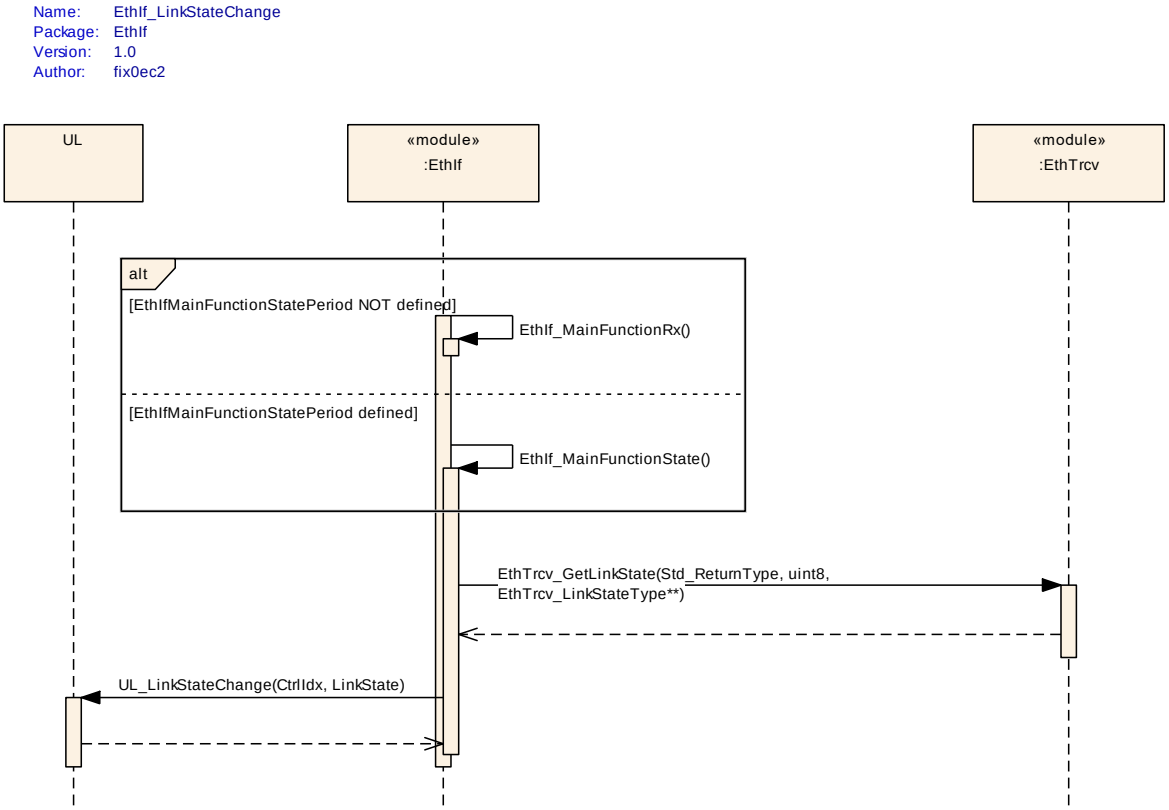

Figure 11: Link State Change

9.7 Link State Change without Port Groups

Name: EthIf_EthSwt_LinkStateChange_NoPortGroup
Package: EthIf
Version: 1.0
Author: fix0ec2

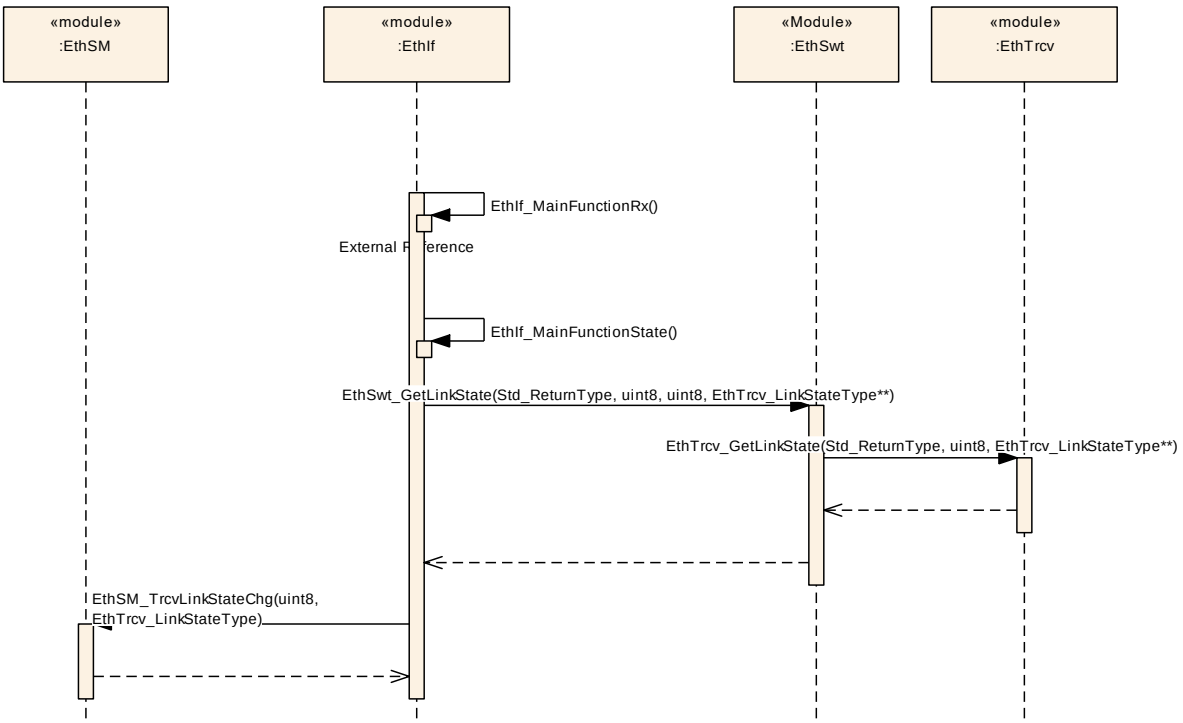

Figure 12: Link State Change without Port Groups

9.8 Link State Change with Port Groups

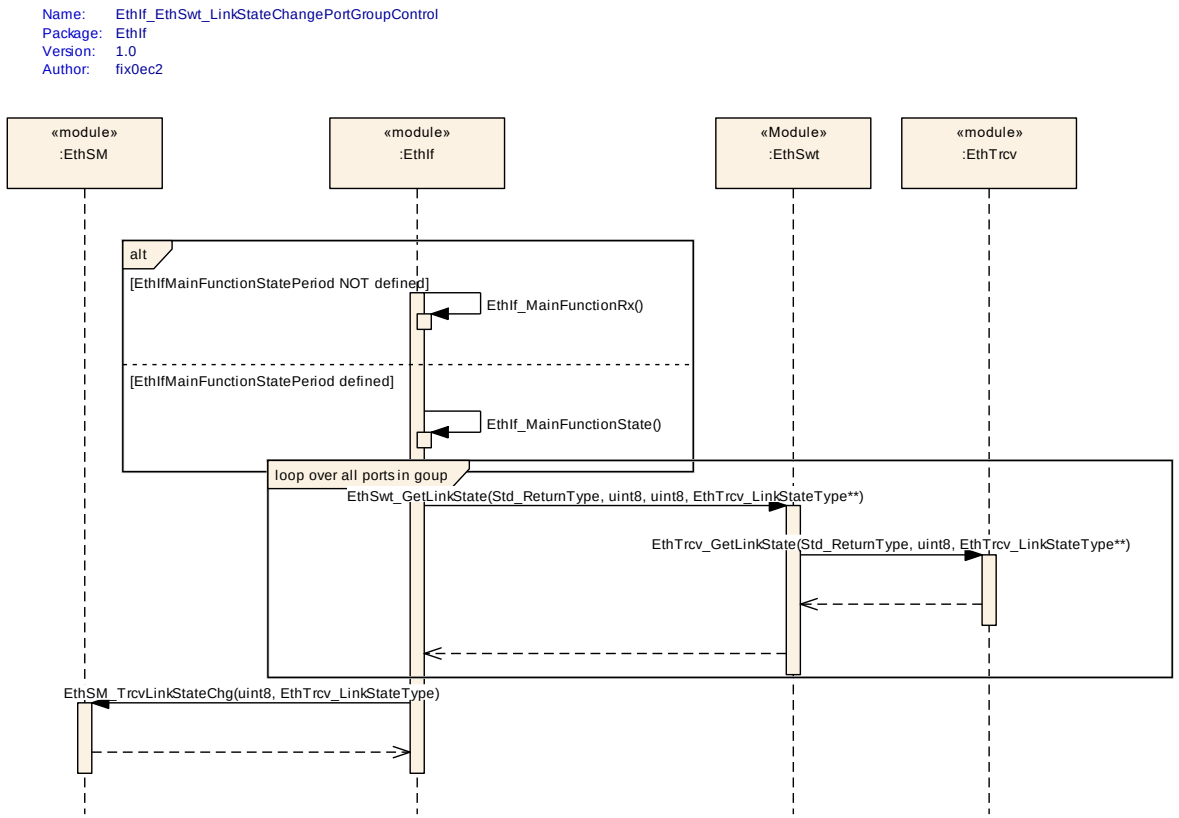

Figure 13: Link State Change with Port Groups

9.9 Link State Change with Port Groups and Partial Network Cluster

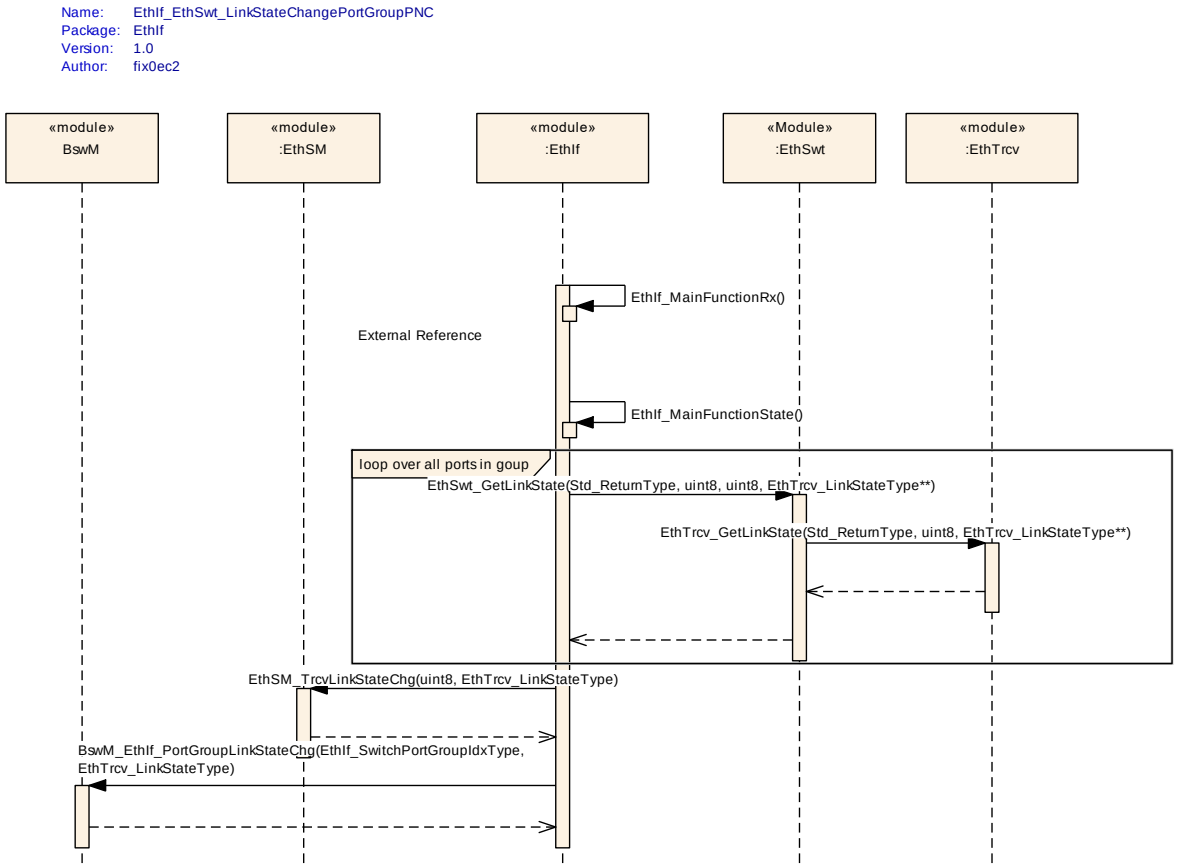

Figure 14: Link State Change with Port Groups and Partial Network Cluster

9.10 Switch Management support

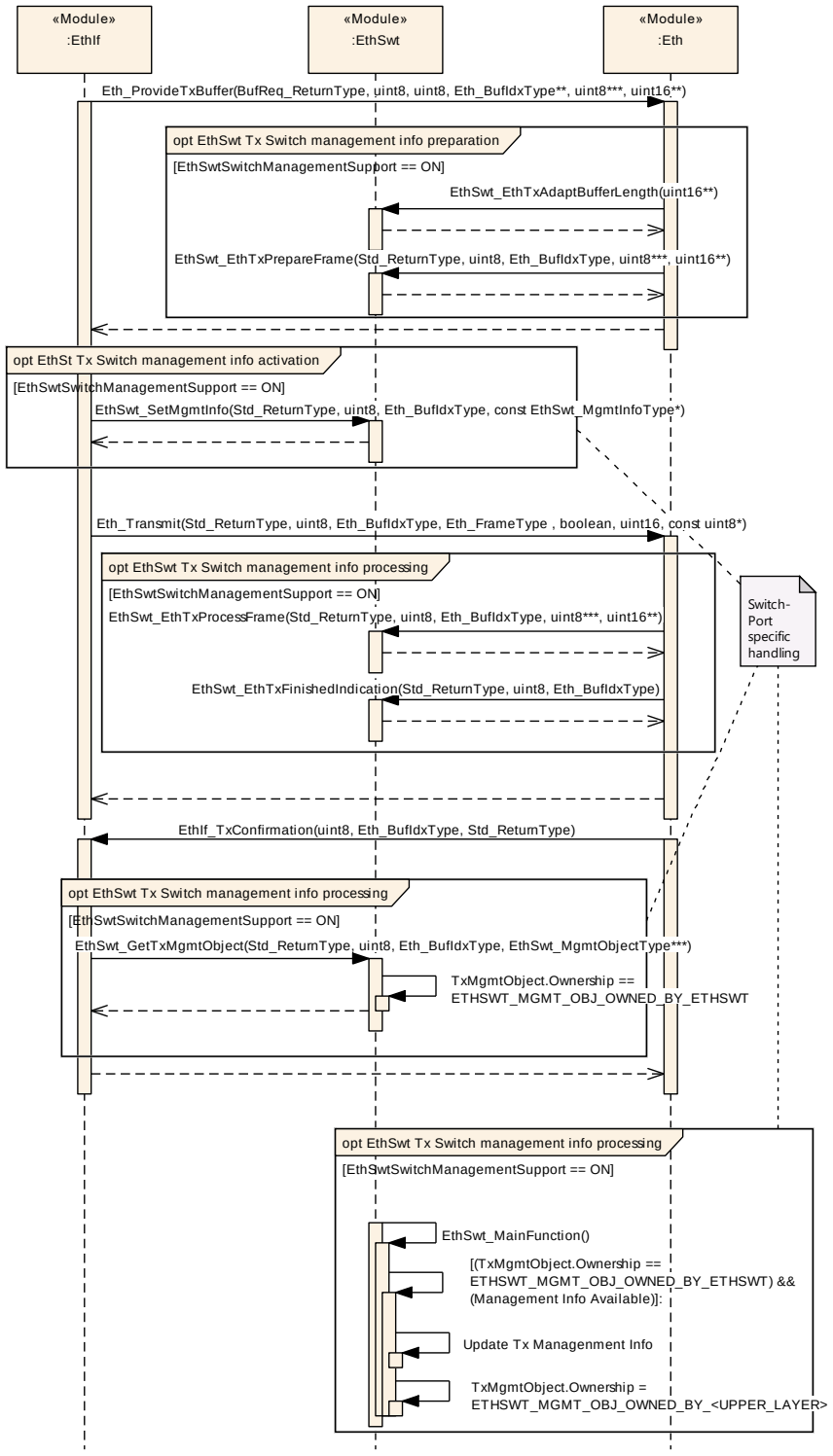

Figure 15: Switch Management support for transmission

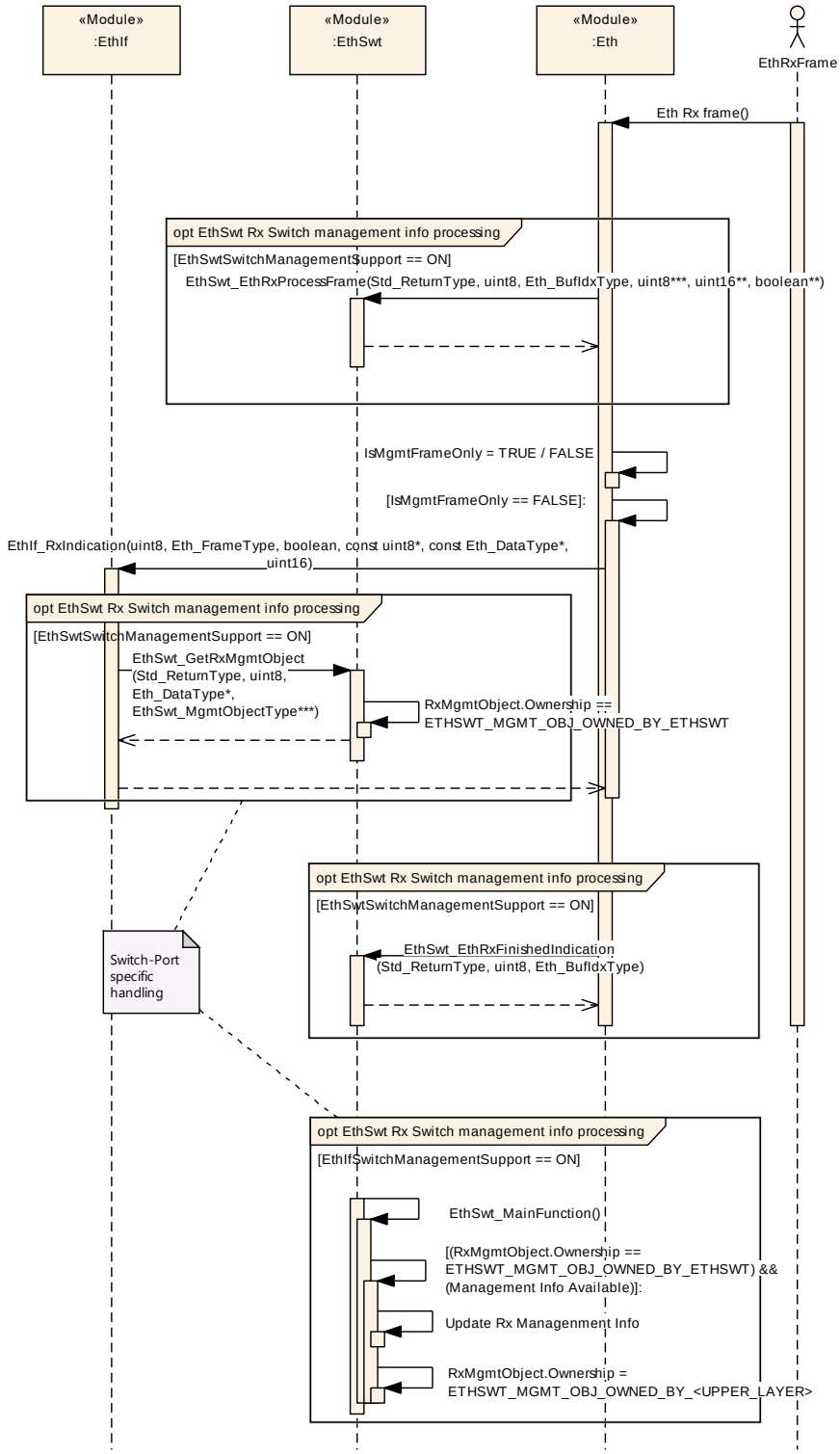

Figure 16: Switch Management support for reception

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 Ethernet Interface.

Chapter 10.3 specifies published information of the module Ethernet Interface.

10.1 Containers and configuration parameters

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

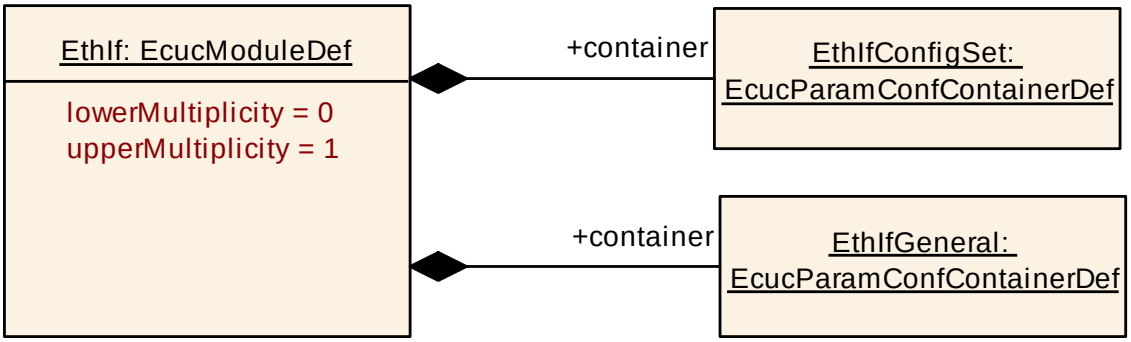

Figure 10.1: Ethernet Interface

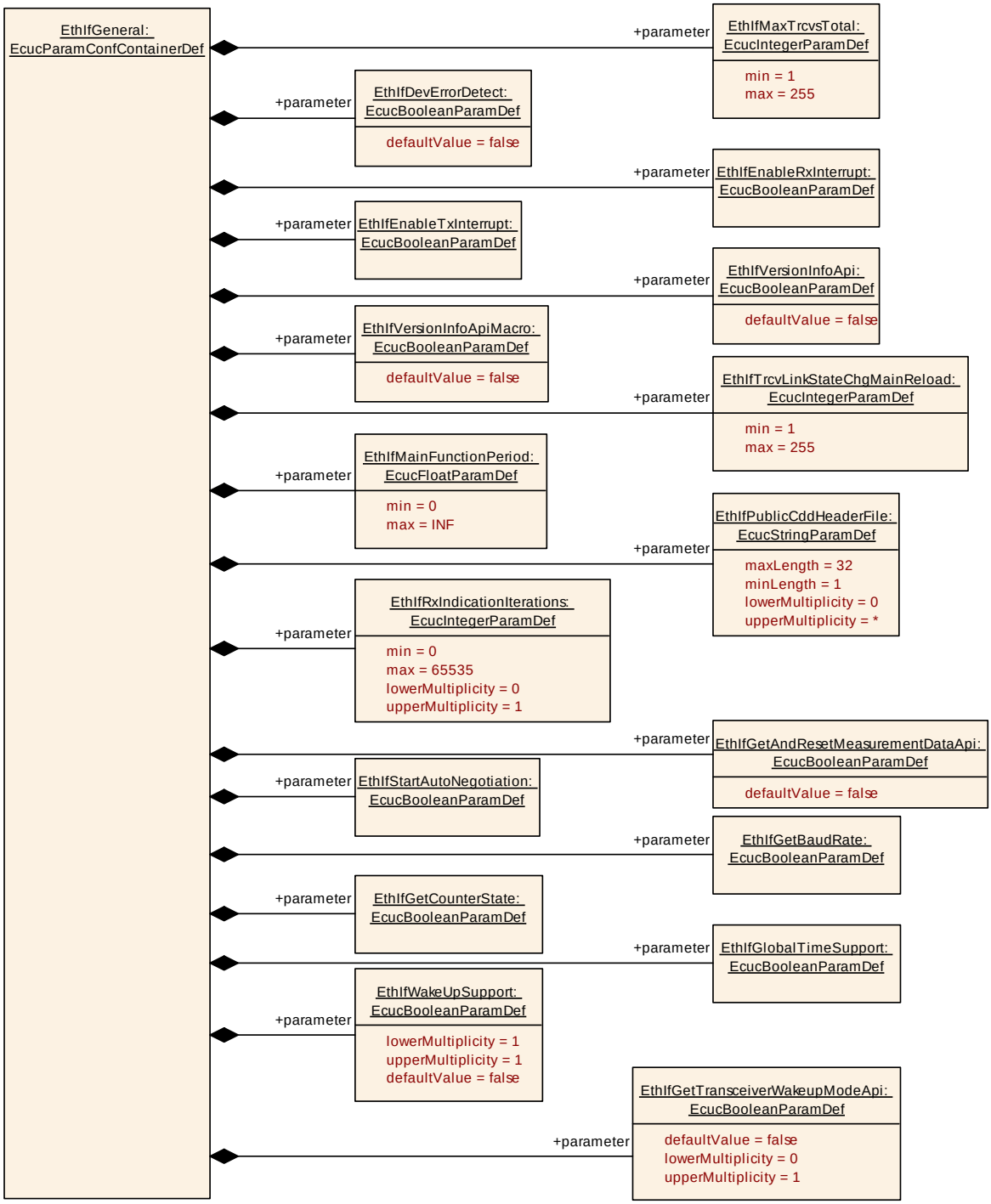

Figure 10.2a: Ethernet Interface general configuration structure

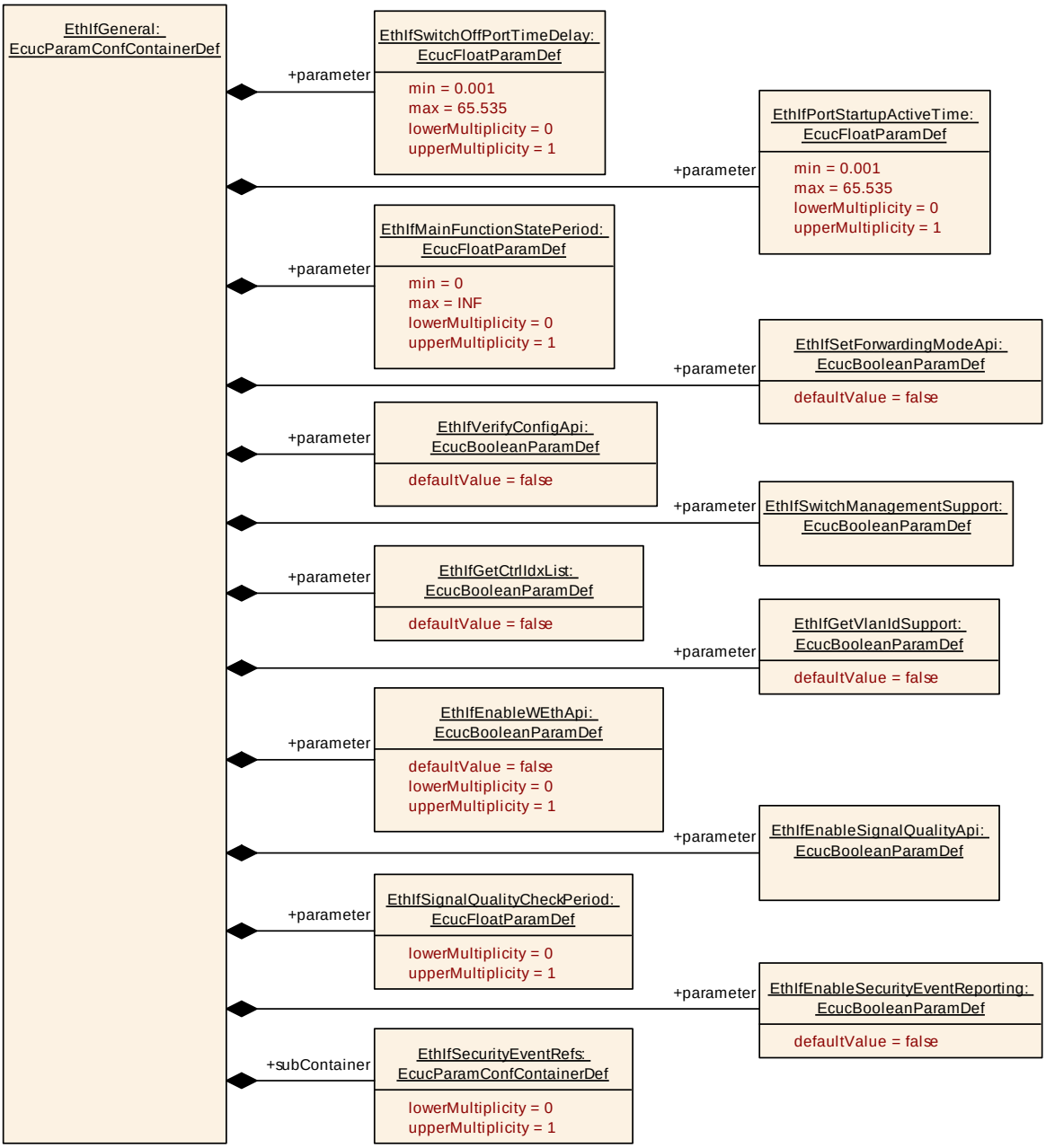

Figure 10.2b: Ethernet Interface general configuration structure (continued)

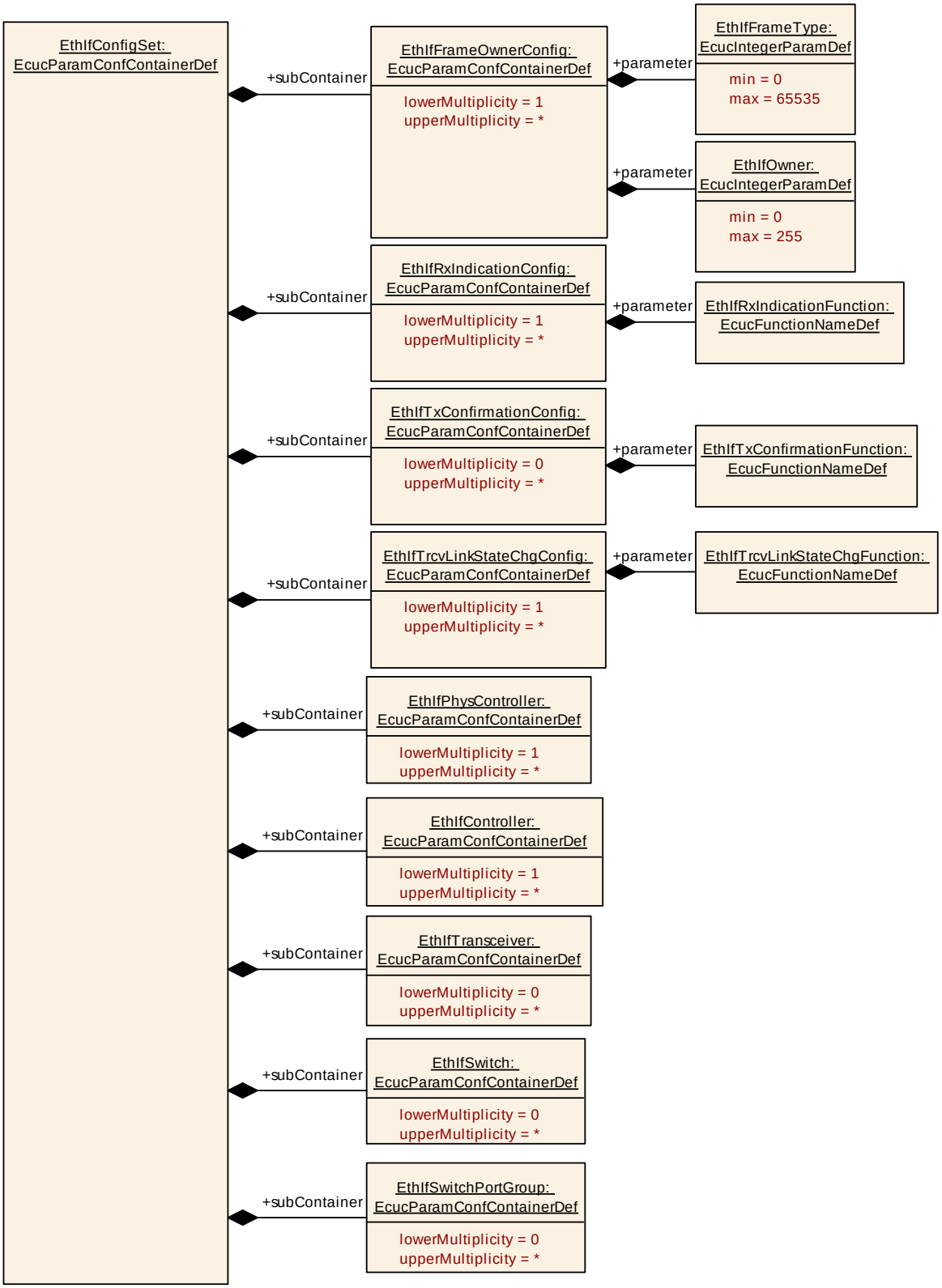

Figure 10.3: Ethernet Interface interface configuration structure

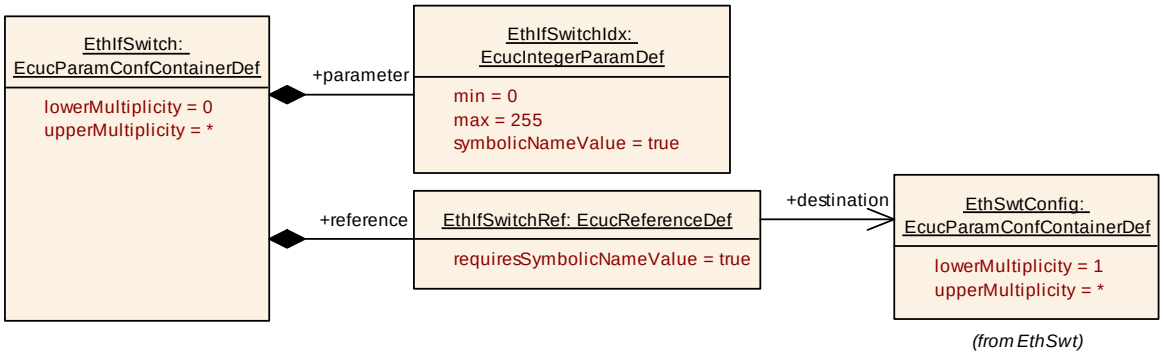

Figure 10.4: Ethernet Interface Switch configuration structure

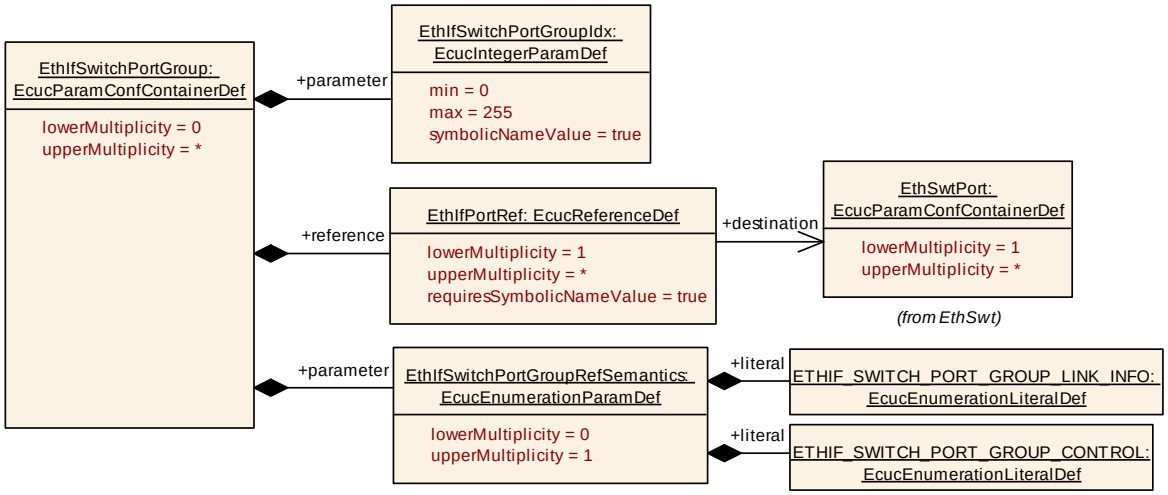

Figure 10.5: Ethernet Interface SwitchPortGroup configuration structure

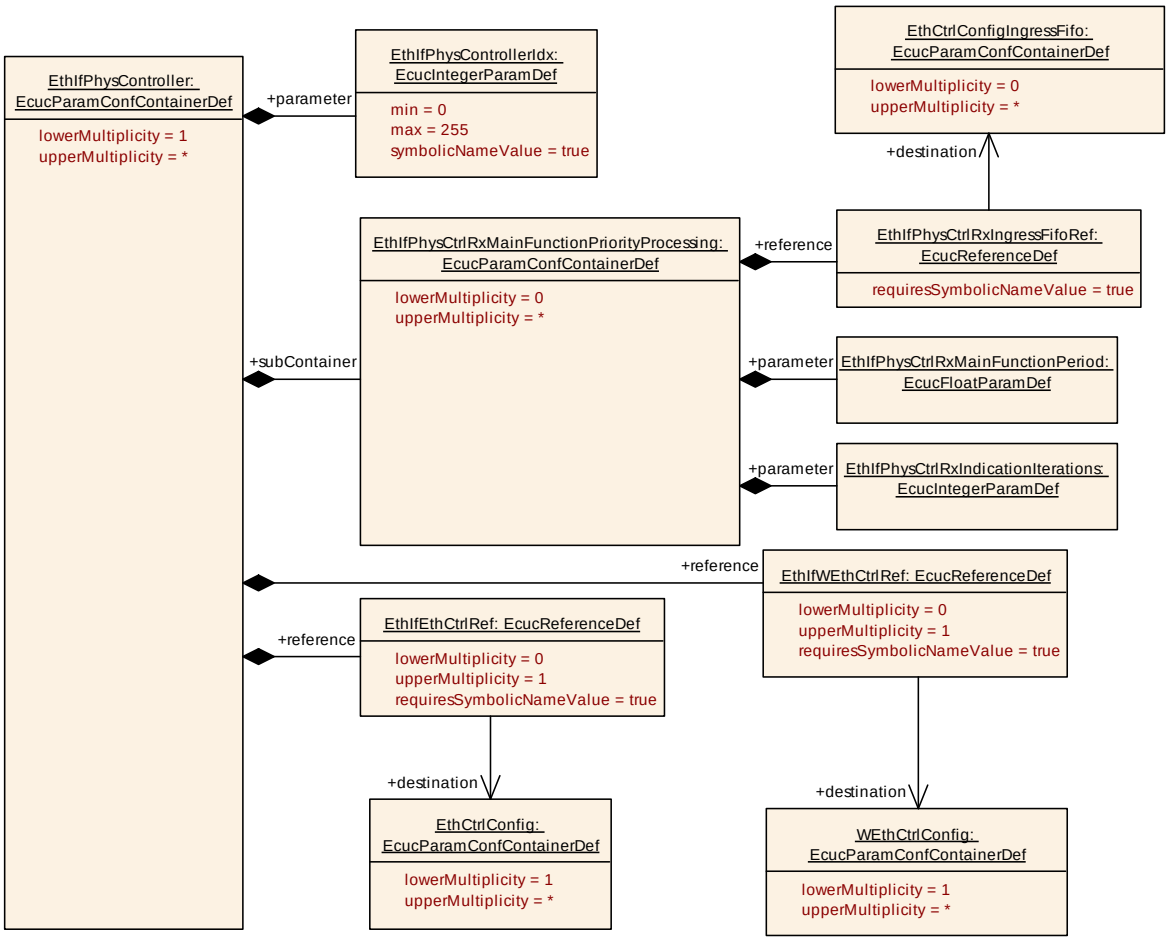

Figure 10.6: Ethernet Interface physical controller configuration structure

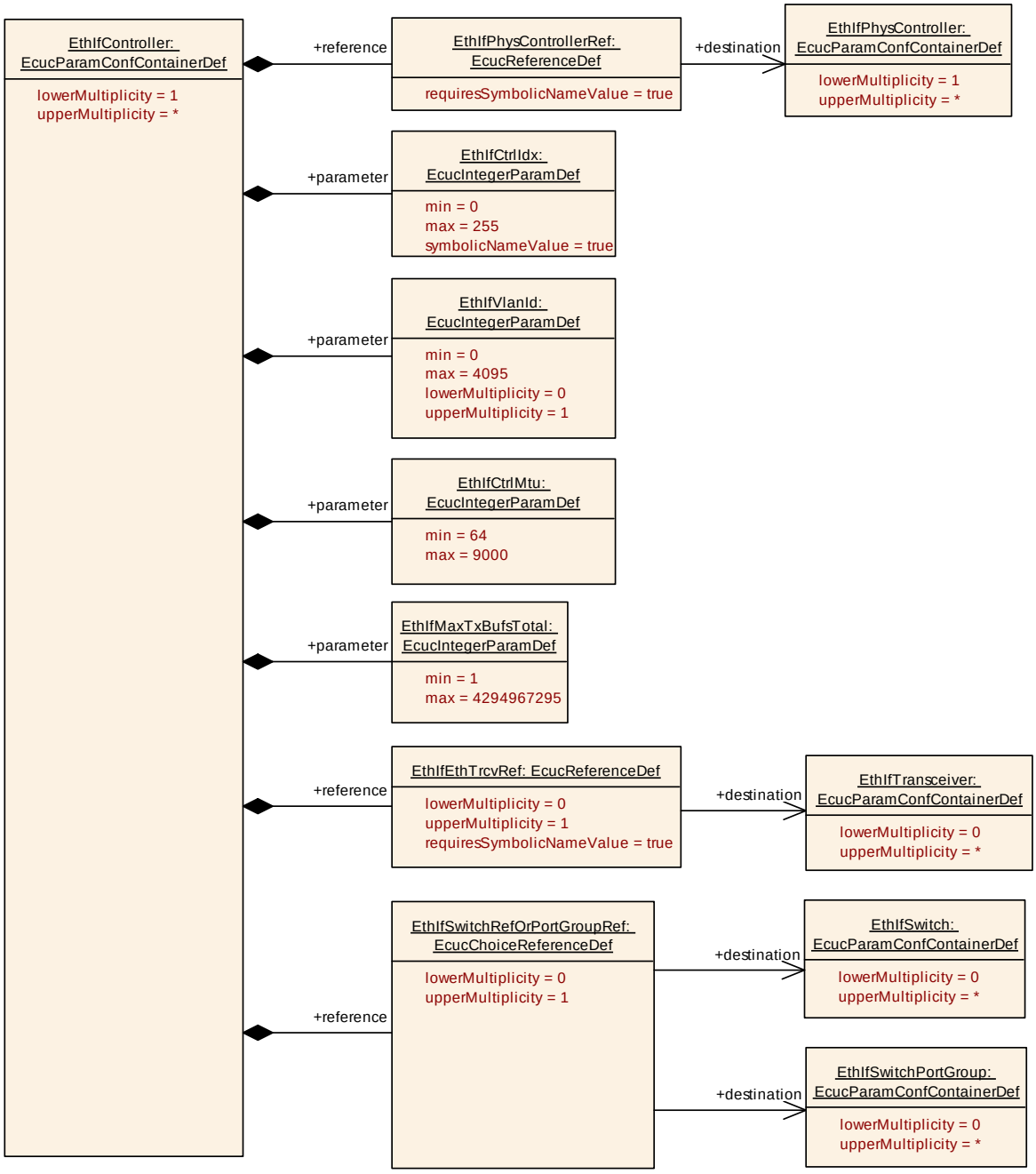

Figure 10.7: Ethernet Interface controller configuration structure

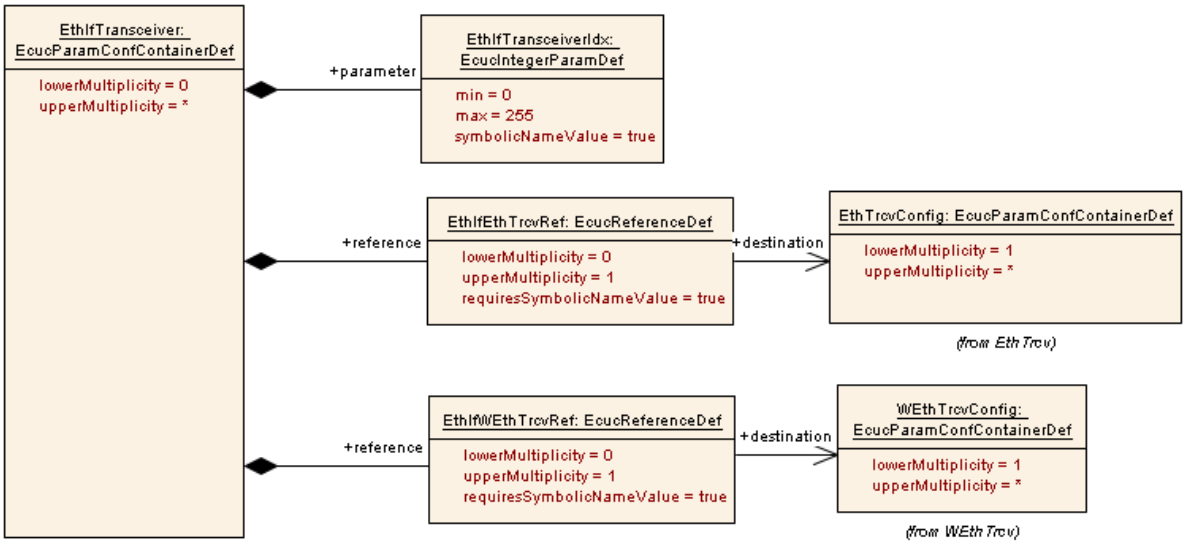

Figure 10.8: Ethernet Interface transceiver configuration structure

11 Not applicable requirements

[SWS_EthIf_00999]
These requirements are not applicable to this specification (BSW00170).