

<b>Document Title</b>	Specification of Ethernet Switch Driver
<b>Document Owner</b>	AUTOSAR
<b>Document Responsibility</b>	AUTOSAR
<b>Document Identification No</b>	656

<b>Document Status</b>	Final
<b>Part of AUTOSAR Standard</b>	Classic Platform
<b>Part of Standard Release</b>	4.4.0

<b>Document Change History</b>			
<b>Date</b>	<b>Release</b>	<b>Changed by</b>	<b>Change Description</b>
2018-10-31	4.4.0	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>• Clarified Port Mirroring concepts</li> <li>• Introduced timeout for ARL table entries</li> <li>• Added counter synchronization for cascaded switches</li> </ul>
2017-12-08	4.3.1	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>• minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation</li> </ul>

<b>Document Change History</b>			
<b>Date</b>	<b>Release</b>	<b>Changed by</b>	<b>Change Description</b>
2016-11-30	4.3.0	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>• Restructured VLAN-membership as a port-related configuration parameter</li> <li>• Introduced configuration of rate policers on ingress side</li> <li>• Introduced filter configuration for double tagged frames</li> <li>• Introduced configuration of minimum buffer size for FIFOS</li> <li>• Introduced Types to read HW-statistic by List pointer; reorganized interfaces to read HW-statistics.</li> <li>• Introduced Compensation of Ethernet switch delays for Global Time Synchronization</li> <li>• Add / update elements to describe MAC interface and physical interface</li> <li>• Added testing functionality for diagnostic use cases</li> <li>• Added Possibility to switch off ports and switch instances according to VLAN or PNC.</li> <li>• Introduced interfaces for verification of switch configuration</li> </ul>
2015-07-31	4.2.2	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>• minor corrections / clarifications / editorial changes; For details please refer to the Change Documentation</li> </ul>
2014-10-31	4.2.1	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>• Initial Release</li> </ul>

## Disclaimer

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

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

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

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

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

## Table of Contents

1	Introduction and functional overview .....	7
2	Acronyms and abbreviations .....	8
3	Related documentation.....	9
3.1	Related standards and norms .....	9
3.2	Related specification .....	9
4	Constraints and assumptions .....	10
4.1	Limitations .....	10
4.2	Applicability to car domains .....	10
5	Dependencies to other modules.....	11
6	Requirements traceability .....	12
7	Functional specification .....	16
7.1	Ethernet BSW stack .....	16
7.1.1	Indexing scheme .....	17
7.1.2	Ethernet Switch Port Mirroring.....	17
7.1.3	Handling of cable diagnostic.....	18
7.1.4	Functional Description .....	19
7.2	Error Classifications.....	29
7.2.1	Development Errors.....	30
7.2.2	Runtime Errors .....	32
7.2.3	Transient Faults.....	32
7.2.4	Production Errors.....	32
7.2.5	Extended Production Errors .....	32
8	API specification.....	34
8.1	Imported types.....	34
8.2	Type definitions .....	34
8.2.1	EthSwt_StateType.....	34
8.2.2	EthSwt_ConfigType.....	35
8.2.3	EthSwt_MacLearningType .....	35
8.2.4	EthSwt_MgmtInfoType .....	35
8.2.5	EthSwt_PortMirrorCfgType.....	35
8.2.6	EthSwt_PortMirrorStateType .....	37
8.2.7	EthSwt_ReturnType .....	37
8.2.8	EthSwt_MgmtOwner.....	38
8.2.9	EthSwt_Mgmt_ObjectType.....	38
8.2.10	EthSwt_MgmtObjectValidType .....	38
8.3	Function definitions.....	39
8.3.1	EthSwt_Init .....	39
8.3.2	EthSwt_SwitchInit.....	40
8.3.3	EthSwt_SetSwitchPortMode.....	41
8.3.4	EthSwt_GetSwitchPortMode .....	42
8.3.5	EthSwt_StartSwitchPortAutoNegotiation .....	43
8.3.6	EthSwt_GetLinkState .....	44
8.3.7	EthSwt_GetBaudRate .....	45

8.3.8	EthSwT_GetDuplexMode .....	45
8.3.9	EthSwT_GetPortMacAddr .....	46
8.3.10	EthSwT_GetArItable .....	47
8.3.11	EthSwT_GetCounterValues.....	48
8.3.12	EthSwT_GetRxStats.....	49
8.3.13	EthSwT_GetTxStats .....	49
8.3.14	EthSwT_GetTxErrorCounterValues.....	50
8.3.15	EthSwT_GetSwitchReg .....	51
8.3.16	EthSwT_SetSwitchReg.....	51
8.3.17	EthSwT_ReadTrcvRegister .....	52
8.3.18	EthSwT_WriteTrcvRegister .....	53
8.3.19	EthSwT_EnableVlan.....	53
8.3.20	EthSwT_StoreConfiguration .....	54
8.3.21	EthSwT_ResetConfiguration .....	55
8.3.22	EthSwT_SetMacLearningMode .....	55
8.3.23	EthSwT_GetMacLearningMode.....	56
8.3.24	EthSwT_NvmSingleBlockCallback .....	57
8.3.25	EthSwT_GetVersionInfo .....	58
8.3.26	EthSwT_MgmtInit .....	59
8.3.27	EthSwT_EthRxProcessFrame .....	59
8.3.28	EthSwT_EthRxFinishedIndication.....	60
8.3.29	EthSwT_EthTxPrepareFrame.....	61
8.3.30	EthSwT_EthTxAdaptBufferLength .....	61
8.3.31	EthSwT_SetMgmtInfo.....	62
8.3.32	EthSwT_EthTxProcessFrame.....	62
8.3.33	EthSwT_EthTxFinishedIndication .....	63
8.3.34	EthSwT_PortEnableTimeStamp .....	64
8.3.35	EthSwT_VerifyConfig.....	64
8.3.36	EthSwT_SetForwardingMode .....	65
8.3.37	EthSwT_GetPortSignalQuality.....	65
8.3.38	EthSwT_GetPortIdentifier .....	66
8.3.39	EthSwT_GetSwitchIdentifier .....	67
8.3.40	EthSwT_WritePortMirrorConfiguration.....	68
8.3.41	EthSwT_ReadPortMirrorConfiguration .....	69
8.3.42	EthSwT_DeletePortMirrorConfiguration.....	69
8.3.43	EthSwT_GetPortMirrorState .....	70
8.3.44	EthSwT_SetPortMirrorState .....	71
8.3.45	EthSwT_SetPortTestMode .....	72
8.3.46	EthSwT_SetPortLoopbackMode.....	72
8.3.47	EthSwT_SetPortTxMode .....	73
8.3.48	EthSwT_RunPortCableDiagnostic.....	74
8.3.49	EthSwT_GetPortCableDiagnosticsResult.....	75
8.3.50	EthSwT_GetCfgDataRow .....	75
8.3.51	EthSwT_GetCfgDataInfo .....	76
8.3.52	EthSwT_PortLinkStateRequest .....	77
8.3.53	EthSwT_GetMaxFIFOBufferFillLevel.....	77
8.3.54	EthSwT_GetRxMgmtObject.....	78
8.3.55	EthSwT_GetTxMgmtObject.....	79
8.4	Call-back notifications.....	79
8.4.1	<user>_PersistentConfigurationResult.....	79

8.5	Scheduled functions .....	80
8.6	Expected Interfaces.....	81
8.6.1	Mandatory Interfaces.....	81
8.6.2	Optional Interfaces .....	81
8.6.3	Configurable interfaces.....	82
8.7	Service Interfaces.....	83
9	Sequence diagrams .....	84
9.1	Switch Management support.....	85
10	Configuration specification .....	88
10.1	Containers and configuration parameters .....	88
10.1.1	EthSwT .....	88
10.1.2	EthSwTConfig.....	88
10.1.3	EthSwTDemEventParameterRefs .....	91
10.1.4	EthSwTGeneral .....	94
10.1.5	EthSwTPort .....	109
10.1.6	EthSwTPortIngress.....	115
10.1.7	EthSwTPortPolicer.....	119
10.1.8	EthSwTPriorityRegeneration .....	120
10.1.9	EthSwTPriorityTrafficClassAssignment .....	121
10.1.10	EthSwTPortEgress .....	122
10.1.11	EthSwTPortScheduler .....	123
10.1.12	EthSwTPortSchedulerPredecessor .....	124
10.1.13	EthSwTPortShaper.....	125
10.1.14	EthSwTPortFifo.....	126
10.1.15	EthSwTPortVlanMembership.....	127
10.1.16	EthSwTSpi.....	128
10.1.17	EthSwTSpiSequence.....	129
10.1.18	EthSwTNvm.....	130
10.2	Constraints.....	132

# 1 Introduction and functional overview

In the AUTOSAR Layered Software Architecture, the Ethernet Switch Driver belongs to the Communication Hardware Abstraction.

This indicates the main task of the Ethernet Switch Driver:

Provide to the upper layers (e.g. Ethernet Interface) a hardware independent interface comprising a switch with several ports. This interface shall be uniform for all Ethernet switches. Thus, the upper layers may access the underlying communication technology in a uniform manner.

A single Ethernet Switch Driver module supports only one type of switch hardware. The Ethernet physical layer ports are configured by the Ethernet Transceiver Driver. The Ethernet Switch Driver's prefix generates a unique namespace. The Ethernet Interface can access different Ethernet controller types using different Ethernet Switch Drivers using this prefix. The decision which driver to use to access a particular transceiver is a configuration parameter of the Ethernet Interface.

Figure 1 depicts the lower part of the Ethernet stack. Accesses via an SPI- and MII/MDIO-Hardware-Interface for switch specific configuration or functions are directly done via the Ethernet Driver or the SPI driver.

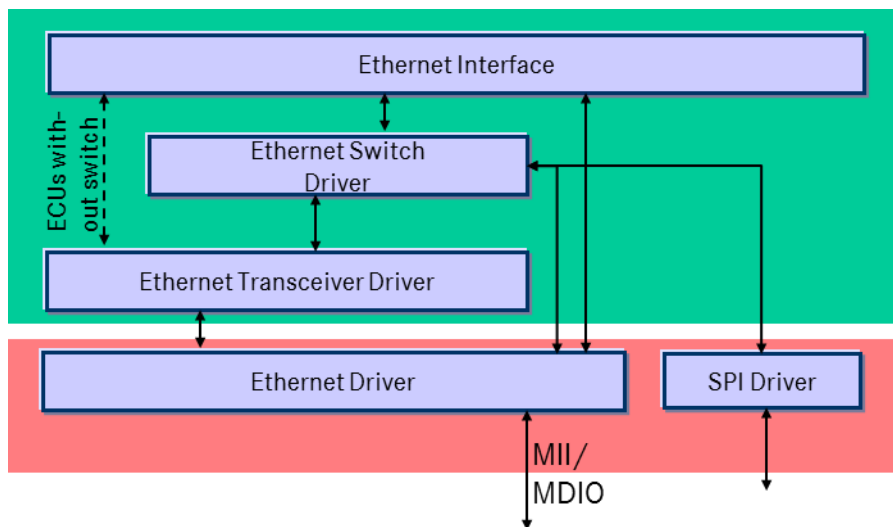


Figure 1 Ethernet Switch Driver in layer architecture

## 2 Acronyms and abbreviations

<b>Abbreviation / Acronym:</b>	<b>Description:</b>
Eth	Ethernet Controller Driver (AUTOSAR BSW module)
EthIf	Ethernet Interface (AUTOSAR BSW module)
EthTrcv	Ethernet Transceiver Driver (AUTOSAR BSW module)
MII	Media Independent Interface (standardized interface provided by Ethernet controllers to access Ethernet transceivers)
MDIO	Management Data Input/Output



### 3 Related documentation

- [1] AUTOSAR Layered Software Architecture  
AUTOSAR\_EXP\_LayeredSoftwareArchitecture.pdf
- [2] AUTOSAR General Requirements on Basic Software Modules  
AUTOSAR\_SRS\_BSWGeneral.pdf
- [3] AUTOSAR General Specification for Basic Software Modules  
AUTOSAR\_SWS\_BSWGeneral.pdf
- [4] AUTOSAR Requirements on Ethernet Support in AUTOSAR  
AUTOSAR\_SRS\_Ethernet.pdf
- [5] AUTOSAR Specification of Ethernet Interface  
AUTOSAR\_SWS\_EthernetInterface.pdf
- [6] AUTOSAR Specification of Transceiver Driver  
AUTOSAR\_SWS\_TransceiverDriver.pdf
- [7] AUTOSAR Specification of Ethernet Driver  
AUTOSAR\_SWS\_EthernetDriver.pdf
- [8] AUTOSAR Specification of Global Time Synchronization over Ethernet  
AUTOSAR\_SWS\_TimeSyncOverEthernet.pdf
- [9] AUTOSAR Specification of ECU State Manager  
AUTOSAR\_SWS\_ECUStateManager.pdf
- [10] AUTOSAR Specification of NVRAM Manager  
AUTOSAR\_SWS\_NVRAMManager.pdf
- [11] AUTOSAR Specification of SPI Handler / Driver  
AUTOSAR\_SWS\_SPIHandlerDriver.pdf
- [12] AUTOSAR Specification of Diagnostic Event Manager  
AUTOSAR\_SWS\_DiagnosticEventManager.pdf
- [13] AUTOSAR Specification of Default Error Tracer  
AUTOSAR\_SWS\_DefaultErrorTracer.pdf
- [14] AUTOSAR Glossary  
AUTOSAR\_TR\_Glossary.pdf
- [15] AUTOSAR\_SWS\_StandardTypes.pdf

#### 3.1 Related standards and norms

- [15] IEEE 802.1Q, <http://standards.ieee.org/getieee802/download/802.1Q-2011.pdf>
- [16] IEEE 802.3, <http://standards.ieee.org/about/get/802/802.3.html>
- [17] IEEE 802.1, <http://standards.ieee.org/about/get/802/802.1.html>

#### 3.2 Related specification

AUTOSAR provides a General Specification on Basic Software (SWS\_BSWGeneral) [3] which is also valid for Ethernet Switch Driver.

Thus, the specifications SWS\_BSWGeneral [3], SRS\_Ethernet [4] shall be considered as additional and required specification for Ethernet Switch Driver.

## 4 Constraints and assumptions

### 4.1 Limitations

The Ethernet Switch Driver module is only able to handle a single thread of execution. The execution must not be pre-empted by itself.

The implementation is limited to 10Mbit/s, 100MBit/s and 1000Mbit/s Ethernet and transceivers connected via (gigabit) Media Independent Interface (xMII).

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

The switch driver does not support the following features:

- MAC-based Ingress Filtering: No filtering options for Ethernet frames based on MAC-addresses is supported.

### 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 Switch Driver module.

Modules that use the Ethernet Switch Driver module:

- Ethernet Interface (EthIf) calls the Ethernet Switch driver for initializing and accessing the switch device.

Modules used by the Ethernet Switch Driver module:

- Ethernet Controller Driver (Eth) for transceiver access via Media Independent Interface (MII).
- Ethernet Transceiver Driver (EthTrcv) for configuring the PHY ports and controlling/checking the ports.
- The configuration of the Ethernet Switch device can be either via MDIO or SPI. In case of an SPI interface access to SPI module is necessary.

Dependencies to other Modules:

- On certain systems the Ethernet switch might share resources with other components, and may depend on their configuration. If those resources are within the scope of other modules (e.g. PLL configuration, memory mapping, etc.) the Ethernet Switch Driver module does not take care of configuring those components but requires their preceding initialization.

## 6 Requirements traceability

Requirement	Description	Satisfied by
SRS_BSW_00003	All software modules shall provide version and identification information	SWS_EthSwT_00131
SRS_BSW_00101	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	SWS_EthSwT_00006, SWS_EthSwT_00007, SWS_EthSwT_00008, SWS_EthSwT_00011, SWS_EthSwT_00012
SRS_BSW_00161	The AUTOSAR Basic Software shall provide a microcontroller abstraction layer which provides a standardized interface to higher software layers	SWS_EthSwT_00099, SWS_EthSwT_00130
SRS_BSW_00162	The AUTOSAR Basic Software shall provide a hardware abstraction layer	SWS_EthSwT_00099, SWS_EthSwT_00130
SRS_BSW_00171	Optional functionality of a Basic-SW component that is not required in the ECU shall be configurable at pre-compile-time	SWS_EthSwT_00022, SWS_EthSwT_00029, SWS_EthSwT_00035, SWS_EthSwT_00042, SWS_EthSwT_00049, SWS_EthSwT_00056, SWS_EthSwT_00058, SWS_EthSwT_00090, SWS_EthSwT_00095, SWS_EthSwT_00109, SWS_EthSwT_00124, SWS_EthSwT_00129, SWS_EthSwT_00177, SWS_EthSwT_00186, SWS_EthSwT_00191, SWS_EthSwT_00195, SWS_EthSwT_00202, SWS_EthSwT_00210, SWS_EthSwT_00215, SWS_EthSwT_00220, SWS_EthSwT_00225, SWS_EthSwT_00229, SWS_EthSwT_00230, SWS_EthSwT_00240, SWS_EthSwT_00243, SWS_EthSwT_00246, SWS_EthSwT_00249, SWS_EthSwT_00253, SWS_EthSwT_00257, SWS_EthSwT_00261, SWS_EthSwT_00264, SWS_EthSwT_00268, SWS_EthSwT_00273, SWS_EthSwT_00287, SWS_EthSwT_00291, SWS_EthSwT_00297, SWS_EthSwT_00303, SWS_EthSwT_00308, SWS_EthSwT_00312, SWS_EthSwT_00317, SWS_EthSwT_00322, SWS_EthSwT_00327, SWS_EthSwT_00332, SWS_EthSwT_00338, SWS_EthSwT_00344, SWS_EthSwT_00350, SWS_EthSwT_00362, SWS_EthSwT_00370, SWS_EthSwT_00379, SWS_EthSwT_00403, SWS_EthSwT_00405, SWS_EthSwT_00427, SWS_EthSwT_00432
SRS_BSW_00347	A Naming separation of different instances of BSW drivers shall be in place	SWS_EthSwT_00131
SRS_BSW_00350	All AUTOSAR Basic Software Modules shall allow the enabling/disabling of detection and reporting of	SWS_EthSwT_00386, SWS_EthSwT_00387, SWS_EthSwT_00389, SWS_EthSwT_00390, SWS_EthSwT_00391, SWS_EthSwT_00392, SWS_EthSwT_00393

	development errors.	
SRS_BSW_00369	All AUTOSAR Basic Software Modules shall not return specific development error codes via the API	SWS_EthSwT_00128, SWS_EthSwT_00164
SRS_BSW_00375	Basic Software Modules shall report wake-up reasons	SWS_EthSwT_00098
SRS_BSW_00385	List possible error notifications	SWS_EthSwT_00001, SWS_EthSwT_00113, SWS_EthSwT_00395
SRS_BSW_00386	The BSW shall specify the configuration for detecting an error	SWS_EthSwT_00016, SWS_EthSwT_00164
SRS_BSW_00395	The Basic Software Module specifications shall list all configuration parameter dependencies	SWS_EthSwT_00165
SRS_BSW_00406	A static status variable denoting if a BSW module is initialized shall be initialized with value 0 before any APIs of the BSW module is called	SWS_EthSwT_00123
SRS_BSW_00413	An index-based accessing of the instances of BSW modules shall be done	SWS_EthSwT_00120, SWS_EthSwT_00154, SWS_EthSwT_00156, SWS_EthSwT_00157, SWS_EthSwT_00180
SRS_BSW_00433	Main processing functions are only allowed to be called from task bodies provided by the BSW Scheduler	SWS_EthSwT_00114, SWS_EthSwT_00115
SRS_BSW_323	-	SWS_EthSwT_00009, SWS_EthSwT_00154, SWS_EthSwT_00156, SWS_EthSwT_00157, SWS_EthSwT_00180
SRS_BSW_369	-	SWS_EthSwT_00009, SWS_EthSwT_00154, SWS_EthSwT_00156, SWS_EthSwT_00157, SWS_EthSwT_00180
SRS_ETH_00087	-	SWS_EthSwT_00031, SWS_EthSwT_00032, SWS_EthSwT_00060, SWS_EthSwT_00061, SWS_EthSwT_00086, SWS_EthSwT_00087, SWS_EthSwT_00091, SWS_EthSwT_00092, SWS_EthSwT_00098, SWS_EthSwT_00111, SWS_EthSwT_00117, SWS_EthSwT_00118, SWS_EthSwT_00125, SWS_EthSwT_00126, SWS_EthSwT_00127, SWS_EthSwT_00136, SWS_EthSwT_00162, SWS_EthSwT_00181, SWS_EthSwT_00182, SWS_EthSwT_00183, SWS_EthSwT_00187, SWS_EthSwT_00188, SWS_EthSwT_00193, SWS_EthSwT_00194, SWS_EthSwT_00196, SWS_EthSwT_00197, SWS_EthSwT_00203, SWS_EthSwT_00204, SWS_EthSwT_00226, SWS_EthSwT_00227, SWS_EthSwT_00228, SWS_EthSwT_00235
SRS_ETH_00114	-	SWS_EthSwT_00134, SWS_EthSwT_00172, SWS_EthSwT_00173
SRS_Eth_00114	Ethernet Switch Filtering and	SWS_EthSwT_00233

	Policing	
SRS_ETH_00118	-	SWS_EthSwT_00018, SWS_EthSwT_00019, SWS_EthSwT_00023, SWS_EthSwT_00025, SWS_EthSwT_00026, SWS_EthSwT_00038, SWS_EthSwT_00044, SWS_EthSwT_00045, SWS_EthSwT_00051, SWS_EthSwT_00052, SWS_EthSwT_00098, SWS_EthSwT_00154, SWS_EthSwT_00156, SWS_EthSwT_00157, SWS_EthSwT_00164, SWS_EthSwT_00217, SWS_EthSwT_00222
SRS_ETH_00119	-	SWS_EthSwT_00037, SWS_EthSwT_00038, SWS_EthSwT_00098, SWS_EthSwT_00117, SWS_EthSwT_00118, SWS_EthSwT_00154, SWS_EthSwT_00203, SWS_EthSwT_00204, SWS_EthSwT_00430, SWS_EthSwT_00431
SRS_ETH_00120	-	SWS_EthSwT_00098, SWS_EthSwT_00217, SWS_EthSwT_00222
SRS_Eth_00120	Hardware access via MII and/or SPI	SWS_EthSwT_00206, SWS_EthSwT_00207, SWS_EthSwT_00211, SWS_EthSwT_00212, SWS_EthSwT_00216, SWS_EthSwT_00221
SRS_ETH_00121	-	SWS_EthSwT_00132, SWS_EthSwT_00133, SWS_EthSwT_00134, SWS_EthSwT_00135, SWS_EthSwT_00172, SWS_EthSwT_00173, SWS_EthSwT_00178, SWS_EthSwT_00179, SWS_EthSwT_00234
SRS_ETH_00122	-	SWS_EthSwT_00086, SWS_EthSwT_00087, SWS_EthSwT_00091, SWS_EthSwT_00092, SWS_EthSwT_00125, SWS_EthSwT_00126, SWS_EthSwT_00127, SWS_EthSwT_00136, SWS_EthSwT_00182, SWS_EthSwT_00183, SWS_EthSwT_00193, SWS_EthSwT_00194, SWS_EthSwT_00196
SRS_Eth_00122	Persistent storage of configurations	SWS_EthSwT_00098, SWS_EthSwT_00192
SRS_ETH_00123	-	SWS_EthSwT_00416, SWS_EthSwT_00417, SWS_EthSwT_00418, SWS_EthSwT_00419, SWS_EthSwT_00420, SWS_EthSwT_91017, SWS_EthSwT_91020
SRS_Eth_00123	Testing and diagnostic of switch ports	SWS_EthSwT_00293, SWS_EthSwT_00299, SWS_EthSwT_00305, SWS_EthSwT_00309, SWS_EthSwT_00313, SWS_EthSwT_00318, SWS_EthSwT_00323, SWS_EthSwT_00328, SWS_EthSwT_00334, SWS_EthSwT_00340, SWS_EthSwT_00346, SWS_EthSwT_00421, SWS_EthSwT_00422, SWS_EthSwT_00424, SWS_EthSwT_00425, SWS_EthSwT_00426, SWS_EthSwT_91014, SWS_EthSwT_91015, SWS_EthSwT_91016, SWS_EthSwT_91018, SWS_EthSwT_91019, SWS_EthSwT_91021, SWS_EthSwT_91022, SWS_EthSwT_91023, SWS_EthSwT_91024, SWS_EthSwT_91025, SWS_EthSwT_91029, SWS_EthSwT_91030, SWS_EthSwT_91031, SWS_EthSwT_91032
SRS_ETH_00125	-	SWS_EthSwT_00098, SWS_EthSwT_00240, SWS_EthSwT_00241, SWS_EthSwT_00242,

		SWS_EthSwt_00243, SWS_EthSwt_00244, SWS_EthSwt_00245, SWS_EthSwt_00282, SWS_EthSwt_00378
SRS_Eth_00125	The Ethernet Switch Driver shall support switch frame management	SWS_EthSwt_91002, SWS_EthSwt_91003, SWS_EthSwt_91004, SWS_EthSwt_91005, SWS_EthSwt_91006, SWS_EthSwt_91007, SWS_EthSwt_91008, SWS_EthSwt_91009, SWS_EthSwt_91010, SWS_EthSwt_91028
SRS_ETH_00126	-	SWS_EthSwt_00181, SWS_EthSwt_91012, SWS_EthSwt_91013
SRS_Eth_00126	Independent reset of host ECU and switch hardware	SWS_EthSwt_00292
SRS_ETH_00128	-	SWS_EthSwt_00106, SWS_EthSwt_00199, SWS_EthSwt_00372, SWS_EthSwt_00373
SRS_Eth_00128	The Ethernet Switch Driver shall provide statistic counter values per port	SWS_EthSwt_00198, SWS_EthSwt_00231, SWS_EthSwt_91000, SWS_EthSwt_91001
SRS_ETH_00458	-	SWS_EthSwt_00128

## 7 Functional specification

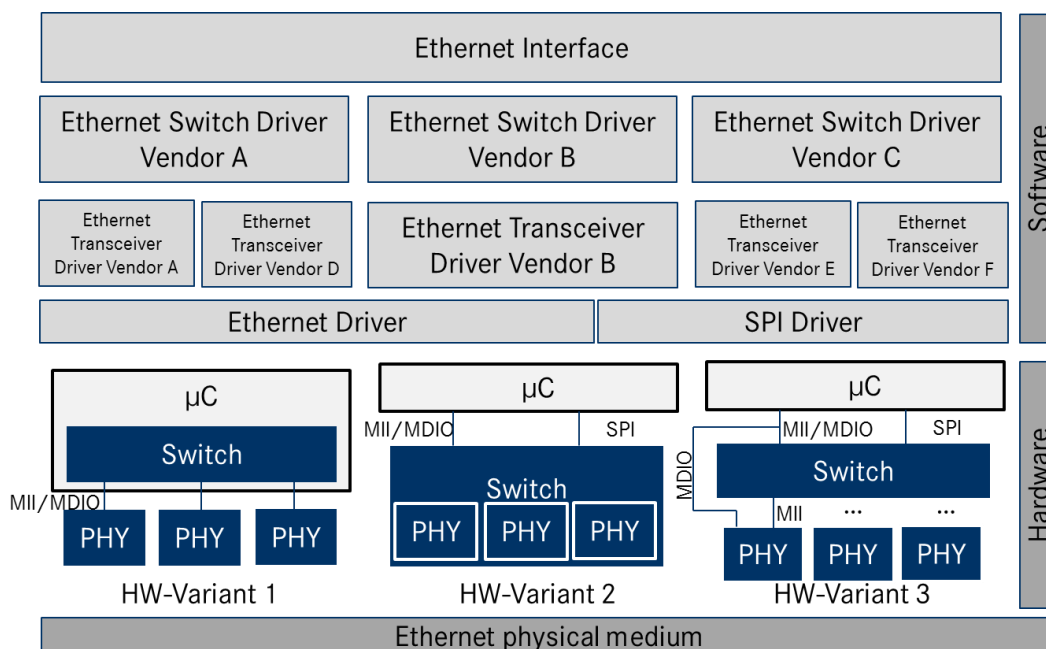
### 7.1 Ethernet BSW stack

As part of the AUTOSAR Layered Software Architecture according to Figure 2, the Ethernet BSW modules also form a layered software stack.

Figure 2 depicts the basic Ethernet BSW stack. The EthIf module accesses several switches using one or more Ethernet Switch Driver modules. The role of the Ethernet transceiver driver is to configure and control the physical layer ports (PHY) integrated into or connected to a switch. Whereas, the role of the Ethernet switch driver is the configuration and control of the switch. In case the Ethernet interface wants to access a PHY, it has to use the APIs of the switch driver which forward the API call to the addressed transceiver driver.

By separating the transceiver driver from the switch driver, different hardware architectures will be supported. In HW-Variant 1, the PHYs are separate devices from different vendors. They are connected via MII and MDIO to a switch which is integrated in to a  $\mu$ C. In HW-Variant 2, the switch has integrated PHYs. In HW-Variant 3, the  $\mu$ C can control the switch via MDIO or SPI and the switch has three external PHYs which can be controlled via MDIO. In this case, different Ethernet transceiver drivers might occur.

Please note that the functional behavior of the ingress and egress port of a switch is implemented in hardware in the switch devices (see 0). Thus, the configuration from chapter 10 in some parts has to be written to the switch device.

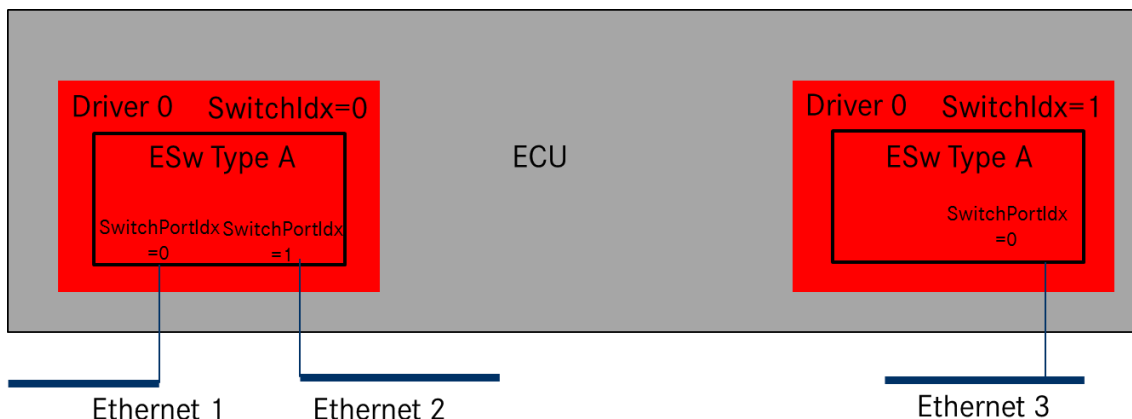


**Figure 2 Basic Structure of the Ethernet BSW stack. (Note: The different hardware variants are alternative setups)**



### 7.1.1 Indexing scheme

Users of the Ethernet Switch Driver identify switch resources using an indexing scheme as depicted in Figure 7.2.



**Figure 3 Ethernet Switch Driver indexing scheme**

**[SWS\_EthSwT\_00099]** [The Ethernet Switch Driver shall use a zero-based index to abstract the access for upper software layers.] (SRS\_BSW\_00161, SRS\_BSW\_00162)

**[SWS\_EthSwT\_00130]** [The SwitchPortIdx is an index for a port at the switch.] (SRS\_BSW\_00161, SRS\_BSW\_00162)

**[SWS\_EthSwT\_00120]** [The parameter EthSwTIdx within the configuration shall correspond to the argument used in the API.] (SRS\_BSW\_00413)

**[SWS\_EthSwT\_00180]** [The parameter EthSwTIndex shall be used to distinguish different instances of a switch driver module in case the API Det\_ReportError(uint16 ModuleId, uint8 InstanceId, uint8 ApId, uint8 ErrorId) is called.] ( SRS\_BSW\_00413, SRS\_BSW\_323, SRS\_BSW\_369)

**[SWS\_EthSwT\_00131]** [In case different Switch devices are used in one ECU, the function names of the different Ethernet Switch drivers must be modified such that no two functions with the same names are generated. It is the responsibility of the user to take care that no two functions with the same names are configured. The names may be extended with a vendor ID or a type ID.] (SRS\_BSW\_00003, SRS\_BSW\_00347)

### 7.1.2 Ethernet Switch Port Mirroring

Ethernet switch port mirroring use the common established functionality of the Ethernet switch hardware to mirror traffic of one or more Ethernet switch ports (mirrored port) to a another Ethernet switch port (capture port). The mirroring configuration is given by the port mirror configuration (see **[SWS\_EthSwT\_91017]** ). The port mirror configuration is set up per Ethernet switch. The configuration is

stored persistently by the Ethernet switch driver. Therefore a shadow buffer is used to store the port mirror configuration during runtime and stored persistently according to the NvM storing strategy (e.g. store the shadow buffer persistently upon ECU shutdown). The port mirror configuration could be activated and de-activated, respectively, explicitly via dedicated APIs. The port mirroring is controlled by a dedicated diagnostic CDD with receive diagnostic request and forward them to the Ethernet switch driver.

**[SWS\_EthSwT\_00416]**

The port mirror configuration (see **[SWS\_EthSwT\_91017]** ) shall be written to a shadow buffer of the Ethernet switch driver per Ethernet Switch by calling `EthSwT_WritePortMirrorConfiguration`.

] (SRS\_ETH\_00123)

**Note:** One port mirror configuration is maintained per Ethernet switch.

**[SWS\_EthSwT\_00417]**

The port mirror configuration shall be enabled and disabled, respectively, per Ethernet Switch by calling `EthSwT_SetPortMirrorState`. The current state of the stored port mirror configuration shall be stored persistently, to outlast an ECU reset and to restore the port mirroring activities after an ECU reset.] ( SRS\_ETH\_00123)

**[SWS\_EthSwT\_00418]**

The stored port mirror configuration shall be marked as "to be deleted" by calling `EthSwT_DeletePortMirrorConfiguration`, if the port mirroring of the given Ethernet switch index is disabled (see **[SWS\_EthSwT\_91022]** ). Otherwise the request to delete the port mirror configuration shall be rejected.

] (SRS\_ETH\_00123)

**Note:** The shadow buffer is stored persistently according to the NvM storing strategy, e.g. store the shadow buffer persistently upon ECU shutdown.

**[SWS\_EthSwT\_00419]**

The current port mirroring state shall be returned by calling `EthSwT_GetPortMirrorState`.] ( SRS\_ETH\_00123)

**[SWS\_EthSwT\_00420]** The port mirror configuration per Ethernet switch shall be returned by calling `EthSwT_ReadPortMirrorConfiguration`.] ( SRS\_ETH\_00123)

### 7.1.3 Handling of cable diagnostic

Cable diagnostic measurement is triggered by calling `EthSwT_RunPortCableDiagnostic`. The current state of the cable diagnostic measurement is polled by calling `EthSwT_GetPortCableDiagnosticsResult`. If `EthSwT_GetPortCableDiagnosticsResult` return with other value than `ETHTRCV_CABLEDIAG_PENDING`, then the cable diagnostic has finished.

Its up to the caller to re-trigger cable diagnostic again, if the measurement failed by returning `ETHTRCV_CABLEDIAG_ERROR`.

**[SWS\_EthSwT\_00428]**

The cable diagnostic APIs (EthSwT\_RunPortCableDiagnostic, EthSwT\_GetPortCableDiagnosticsResult) shall only be called for Ethernet switch ports of a Ethernet switch, where the Ethernet switch ports reference an Ethernet transceiver.] ()

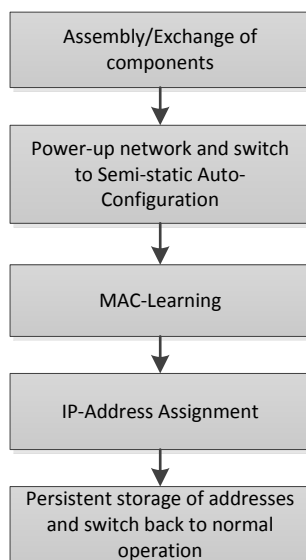
**Note:** The upper layer is a CDD that triggers the cable diagnostic measurement and maintains the cable diagnostic result. The EthSwT forwards the API calls to the EthTrcv (see [SWS\_EthSwT\_00429] and [SWS\_EthSwT\_00346]).

**7.1.4 Functional Description**

**7.1.4.1 Learning Phase at Start-up**

**[SWS\_EthSwT\_00226]** [ The switch driver shall support a learning phase which can be divided into several sequential steps. ](SRS\_ETH\_00087)

**Note:** After assembly and initial power-up of the network, three learning phases follow which include MAC-Learning and IP-Address Assignment. Afterwards the learned parameters are stored to one or several non-volatile memories to make them available for subsequent start-ups. This process is shown in Figure 4. As an example for triggering this process, the DCM receives a diagnostic request via a bus system or a broadcast message in the Ethernet network. This diagnostic request can be forwarded to an SWC or CCD which triggers the auto-configuration process. However, the trigger is not part of this specification.

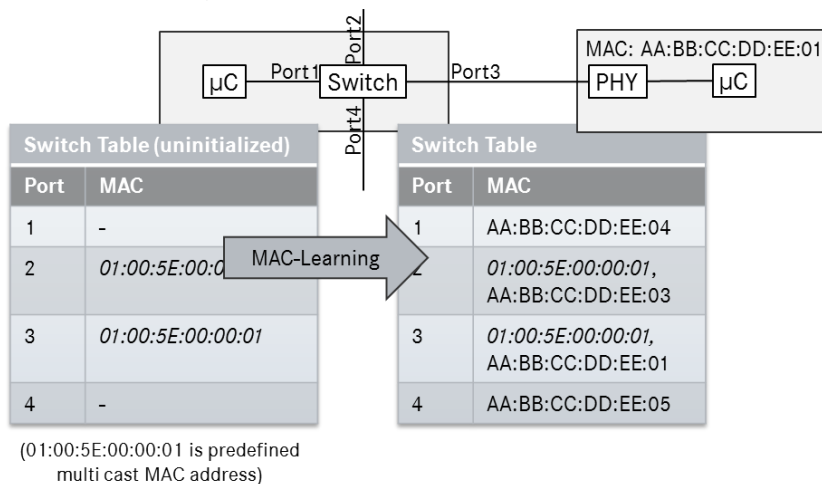


**Figure 4 Learning Process**

**MAC-Learning (Optional Step):** In this phase, messages need to be sent through the network and the switch will learn new MAC addresses (cf. Figure 5). These MAC-addresses will be stored in addition to predefined addresses, e.g. multicast MAC addresses which are configured during the vehicle network design. If static learning

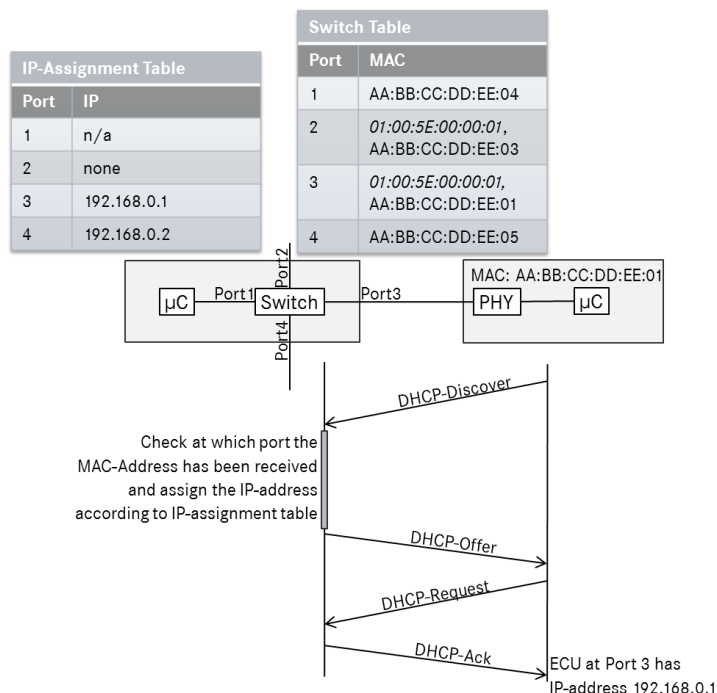
is executed, i.e. MAC address will be persistently stored, it might be possible to add dynamically learned entries in the tables.

If software MAC learning is supported by switch hardware and the switch hardware expects an external  $\mu\text{C}$  (see Variant 2 and 3 in Figure 2), packets with unknown MAC Source Address will be routed to this  $\mu\text{C}$ . The MAC learning is done by integration code. It is intentionally not defined where this algorithm is located within the AUTOSAR stack as this might need a very time-optimized solution.



**Figure 5 MAC-learning within the switch**

**IP-Address Assignment:** In this phase, ECUs without a predefined IP-address will start to acquire an IP-address via DHCP (cf. Figure 6). Thus, these ECUs will run a DHCP-client while the ECU with the switch will run a DHCP server. In order to be able to assign always the same IP-address to a certain node, the DHCP server needs the information at which port the MAC address has been received. This port information can be interpreted as a “domain name” in the internet which is resolved to an IP address using a domain name server (DNS). With this port information the DHCP-server will assign the IP-address according to the IP-Assignment Table to the node. As mentioned above, this allows the assignment of MAC addresses by the Tier 1 and assignment of IP addresses by the OEM. With this mechanism it is also possible to assign different IP addresses to several VLANs at the same port. For this purpose, the IP-Assignment Table needs to be extended with a VLAN-column. Please note that the MAC-Learning-Phase can be combined with this phase.



**Figure 6 IP-address assignment via DHCP**

**[SWS\_EthSwT\_00136]** [The Ethernet Switch driver shall support an API which allows to store learned parameters like address resolution tables in a persistent manner by using the API EthSwT\_StoreConfiguration. This persistent storage can be done in an NVRAM of the host CPU which runs the Ethernet Switch driver. Alternatively, this can be done in a memory of the switch itself. The trigger for storing the learned configuration or resetting the stored configuration can be done e.g. by a DCM.](SRS\_ETH\_00122, SRS\_ETH\_00087)

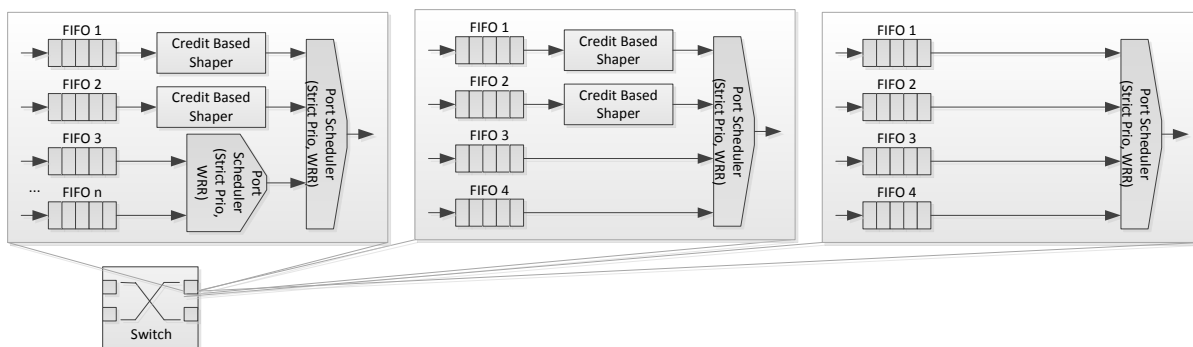
**[SWS\_EthSwT\_00181]** [The Ethernet Switch driver shall support an API which allows to reset learned parameters like address resolution tables by using the API EthSwT\_ResetConfiguration.](SRS\_ETH\_00126, SRS\_ETH\_00087)

**[SWS\_EthSwT\_00162]** [The switch driver shall provide APIs to read the MAC-address to switch port mapping from the switch device to support the IP-address assignment by using the API GetPortMacAddr.](SRS\_ETH\_00087)

**[SWS\_EthSwT\_00407]** [ Unused ARL table entries shall be removed from the ARL table after the timeout configured via EthSwTArITableEntryTimeout (ECUC\_EthSwT\_00127), if this parameters is present in the configuration.] ()

**7.1.4.2 Configuration of Egress Port Structure**

As shown in Figure 7, the switch consists of a certain number of ports. Each port has its own set of egress FIFOs in which the incoming packets will be buffered. How the messages in the FIFOs will be forwarded depends mainly on the shaping and port scheduling mechanisms. Thus, the parameterization of the egress port influences the latency of messages within the network. Please note that the egress port structures in Figure 7 are meant as an example. Other structures with different FIFO numbers are possible.



**Figure 7 Ethernet Egress Port Structure**

Considering the limitations of the hardware, such port structures shall be configurable within e.g. an initialization phase of the Ethernet Switch (see Section 10.1.6ff.)

**[SWS\_EthSwt\_00132]** | The configuration of the Ethernet switch driver shall support different Ethernet egress port structures by the configuration EthSwtPortEgress.  
 | (SRS\_ETH\_00121)

Implementation note:

As Switch HW has very vendor specific structure a more specific description of the applying algorithms is not feasible here. The Routing takes place inside the Switch HW.

The configuration of the Schedulers is done with the container EthSwtPortEgress and its sub-container EthSwtPortScheduler with multiplicity 1 to \*. As depicted in Figure 7-6 multiple schedulers on one port are possible. The Scheduler link which points from Predecessor to Predecessor is done with the references EthSwtPortEgressPredecessorRef. Shaper Algorithms are configured with the container EthSwtPortShaper and are linked to their FIFO through the EthSwtPortEgressPredecessorFifoRef. One FIFO can have multiple schedulers additionally to a shaper algorithm and the port-scheduler.

EthSwtPortSchedulerPredecessorOrder could be used to define the service weight of a FIFO if EthSwtPortSchedulerAlgorithm is configured to ETHSWT\_SCHEDULER\_DEFICIT\_ROUND\_ROBIN or ETHSWT\_SCHEDULER\_STRICT\_PRIORITY. In case of EthSwtPortSchedulerAlgorithm is ETHSWT\_SCHEDULER\_STRICT\_PRIORITY the EthSwtPortSchedulerPredecessorOrder defines the order of the schedulers, The FIFO which is referenced (via Shaper or directly) by the Scheduler with the highest value of EthSwtPortSchedulerPredecessorOrder is the FIFO with the highest priority.

Besides the modeling of egress ports, it is necessary to specify how incoming packets are forwarded to the egress ports. For this purpose, different assignment policies of packets to egress port FIFOs are implemented in switches. As an example, the Ethernet priority field can be evaluated and mapped to a so-called traffic class. Such a traffic class is again mapped to an egress FIFO. Other header information of the Ethernet frame can be also used for the assignment of Ethernet frames to egress FIFOs. For the mapping to a certain traffic class, the following

tables are necessary. While the first table shows the mapping of ingress-ports to traffic classes, the second table shows the priority-based mapping which can be defined per ingress port. Both tables are in conflict with each other, i.e. it has to be decided which mapping is applied.

### 1. Ingress-Port to Traffic Class Mapping

Port-based Mapping	Traffic Class
e.g. Port2, Port3, Port4	7
e.g. Port1	6
-	5
-	4
-	3
-	2
-	1
-	0

### 2. PCP-field (Priority Code Point) to Traffic Class Mapping

PCP-based Mapping	Traffic Class
Prio 0	7
Prio 1	6
Prio 2-7	5
-	4
-	3
-	2
-	1
-	0

After mapping the packets to a traffic class, they will be mapped to a certain FIFO at the egress side of the switch. This mapping can vary from egress port to egress port.

### 3. Traffic Class to FIFO Mapping

Traffic Class	FIFO (if 4 FIFOs available)
7	3
6	2
5-0	1
-	0

While the frame forwarding is a hardware mechanism of the switch, the tables how the frames will be forwarded shall be configurable (see Section 10.1.12ff.). Please note that the traffic class assignment is done after the priority regeneration.

**[SWS\_EthSwt\_00133]** | The switch configuration shall support to configure the Ethernet frame forwarding mechanisms of a switch by the configuration parameters EthSwtPortTrafficClassAssignment, EthSwtPriorityTrafficClassAssignment, EthSwtPortFifoTrafficClassAssignment.  
| (SRS\_ETH\_00121)



**[SWS\_EthSwt\_00234]** The Parameter EthSwtPortFifoMinimumLength shall define the minimum length for one dedicated FIFO on one Switch.] ( SRS\_ETH\_00121)

**Note:** The actual length can be longer. The decision on the length is very likely to be taken by the Switch HW or fixed by the Switch design. To define the minimum in ECUC is supposed to guarantee that some priorities have enough egress buffer.

### 7.1.4.3 Vlan-Membership on Switch-ports

Every Port holds the list of Vlan-Memberships. This Membership describe ingress and egress behavior in terms of filtering, and rate policing and tagging or untagging. For each VLAN identifier a table is necessary which stores at which egress port the corresponding VLAN is tagged or untagged. For an 8-port switch, this table could look like the following example where T stands for tagging and U for untagging:

VLAN Forwarding Table								
VLAN-ID	Port Number							
	1	2	3	4	5	6	7	8
1	T	T	-	U	-	-	-	T
2	T	U	-	T	-	-	-	T
...								
4094								

Incoming packets which contain a VLAN-ID of e.g. 1 can be forwarded to the ports 1, 2, 4, and 8. At ports 1, 2, and 8 these packets will be transmitted with the VLAN tag and at port 4 the tag will be removed. If a broadcast message with e.g. VLAN-ID 2 will be received at port 2. It will be forwarded to port 1, 4, and 8. The other ports 3, 5, 6, and 7 are not in the same VLAN. Thus, the packet will not be forwarded to these egress ports. The table considers only messages, which contain a VLAN-ID within the switch. (see also 10.1.12).

**[SWS\_EthSwt\_00134]** [

The switch configuration shall support the configuration how packets will be forwarded with respect to configured VLANs by using the configuration parameters of the subcontainer EthSwtPortVlanMembership.

] (SRS\_ETH\_00121, SRS\_ETH\_00114)

**Note:** VLAN-Memberships of a port are modeled with the container EthSwtPort:EthSwtPortVlanMembership where the EthSwtPortVlanDefaultPriority and EthSwtPortVlanForwardingType are configured.

### 7.1.4.4 Rate Policing on Ingress Side

If HW supports Rate Policing a policer can be configured using the parameters EthSwtPortRatePolicedTimeInterval, EthSwtPortRatePolicedByteCount, EthSwtPortRatePolicedPriority and EthSwtPortRateVlanMembershipRef.

It is possible to rate only on Priority configuring no membership reference. If the policing shall only check the Vlan, the Priority can be omitted.

An example of a rate limit definition might be a maximum number of data Bytes inside the payload allowed to pass over a 5 ms period.



The policer can either drop the violating frame or block the violating Source based on the MAC-Address depending on the configuration parameter EthSwtPortRateViolationAction.

#### 7.1.4.5 VLAN-modification at ingress side

Another table specifies a port-based modification of the VLAN-ID or an insertion of the VLAN-ID into the Ethernet message:

Ingress VLAN Modification/Insertion Table								
Port Number	1	2	3	4	5	6	7	8
VLAN-ID	2	-	-	6	-	-	-	-

In this example, all incoming messages at port one will get the VLAN-ID 2 no matter they already had one before. At port 4, all incoming messages will get a 6 as their VLAN-ID. At the remaining ports, no VLAN-IDs will be inserted and an existing VLAN-ID in the Ethernet-message will remain without modification.

[SWS\_EthSwt\_00135] [The switch configuration shall support the configuration how VLANs will be inserted into packets or existing VLANs will be modified by the configuration EthSwtPortIngressVlanModification.

] (SRS\_ETH\_00121)

#### 7.1.4.6 Priority-Code-Point-Regeneration

Within the VLAN-tag, the PCP-field (priority code point) is another parameter which can be modified at an ingress port of an Ethernet switch. For this purpose a so-called priority regeneration table has to be defined:

Priority Regeneration Table								
Ingress PCP	0	1	2	3	4	5	6	7
Regenerated PCP	0	1	2	3	4	5	6	7

Table 1 In this table, the “Ingress PCP” is mapped to the “Regenerated PCP”.

[SWS\_EthSwt\_00178] [ The switch configuration shall support the configuration how the PCP field of incoming packets will be modified before they are forwarded to the egress port, i.e. a priority regeneration table can be configured (Please also refer to ECUC\_EthSwt\_00057 to ECUC\_EthSwt\_00059.

] (SRS\_ETH\_00121)

#### 7.1.4.7 Direct Traffic Class Assignment

[SWS\_EthSwt\_00179] [ The switch configuration shall support the configuration of a default traffic class for incoming frames (Please also refer to ECUC\_EthSwt\_00023).

] (SRS\_ETH\_00121)

#### 7.1.4.8 Behavior in case of untagged frames in a VLAN network

There are three ways to handle untagged frames:

- Drop all untagged frames at ingress side of the port
- forward untagged
- tag all untagged frames with a default Vlan and default priority.

**Implementation Hint:** If there is a Vlan-Tag there is also a priority.

To drop all untagged frames at the ingress side of the port the parameter `EthSwtPortIngressDropUntagged` need be set to `TRUE` and the parameters `EthSwtPortIngressDefaultVlan` and `EthSwtPortIngressDefaultPriority` need not be set by setting the multiplicity of both parameters to 0. To add a Default Tag to the untagged frame the parameter `EthSwtPortIngressDropUntagged` need be set to `FALSE` and the parameters `EthSwtPortIngressDefaultVlan` and `EthSwtPortIngressDefaultPriority` need be set to the intended tag.

To forward untagged frames from ingress to egress side the parameter `EthSwtPortIngressDropUntagged` need be set to `FALSE` and the parameters `EthSwtPortIngressDefaultVlan` and `EthSwtPortIngressDefaultPriority` can be set according to internal forwarding rules. This Default Vlan than needs to be configured to be untagged on the egress port by `EthSwtPortVlanForwardingType` set to `ETHSWT_SENT_UNTAGGED`.

**Note:** The handling of untagged frames by the HW is expected to be located before all other modifications of the Vlan and the Priority and before the Traffic Class assignment.

#### 7.1.4.9 Behavior in case of double tagged frames in a VLAN network

**[SWS\_EthSwt\_00233]** The Switch Driver shall support the configuration of dropping double tagged frames via the configuration parameter `EthSwtDropDoubleTagged` if the Switch hardware supports dropping of double tagged frames. ] (SRS\_Eth\_00114)

#### 7.1.4.10 Switch Management support

Switch Management enables the possibility to control an Ethernet frame regarding a Switch-Port specific ingress and egress handling as well as providing a 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 [7] .

**[SWS\_EthSwt\_00240]** The Switch driver shall offer Switch management APIs

- `EthSwt_EthRxProcessFrame()`,
- `EthSwt_EthRxFinishedIndication()`,
- `EthSwt_EthTxAdaptBufferLength()`,
- `EthSwt_EthTxPrepareFrame()`,
- `EthSwt_SetMgmtInfo()`,
- `EthSwt_EthTxProcessFrame()` and
- `EthSwt_EthTxFinishedIndication()`

if `EthSwtManagementSupportApi` is set to `TRUE` .] ( SRS\_BSW\_00171, SRS\_ETH\_00125 )

**Note:** Switch management APIs support the EthIf to gather / modify Switch-Port specific communication attributes.

**[SWS\_EthSwT\_00241]** The Switch Driver management APIs

- EthSwT\_EthRxProcessFrame(),
- EthSwT\_EthRxFinishedIndication(),
- EthSwT\_EthTxAdaptBufferLength(),
- EthSwT\_EthTxPrepareFrame(),
- EthSwT\_SetMgmtInfo(),
- EthSwT\_EthTxProcessFrame() and
- EthSwT\_EthTxFinishedIndication()

shall support the Ethernet Driver to gather the Switch specific management information out of an Ethernet frame for reception or to prepare an Ethernet frame for management mode conformant frame transmission, e.g. the egress route of a frame.  
] (SRS\_ETH\_00125)

**[SWS\_EthSwT\_00242]** The Switch Driver management APIs

EthSwT\_EthTxProcessFrame() and EthSwT\_EthTxFinishedIndication() shall return immediately, if EthSwT\_SetMgmtInfo() has not been called before a call of EthSwT\_EthTxProcessFrame().] (SRS\_ETH\_00125)

**[SWS\_EthSwT\_00282]** If the management information for reception will not be available within the time specified in the configuration parameter EthSwT\_MgmtInfoIndicationTimeout, the pointer MgmtInfoPtr shall be set to NULL before calling the function EthIf\_SwitchMgmtInfoIndication().] (SRS\_ETH\_00125)

#### 7.1.4.11 Global Time support

For more details regarding time measurement with Switches, please refer to [8].

**[SWS\_EthSwT\_00243]** The Switch driver shall access the port specific hardware time stamps if EthSwTPortTimeStampSupport of the port is set to TRUE.] (SRS\_BSW\_00171, SRS\_ETH\_00125)

**[SWS\_EthSwT\_00244]** If EthSwT\_EnableTimeStamp is called for a SwitchIdx, the switch driver shall enable the time-stamping for all his ports where EthSwTPortTimeStampSupport is set to TRUE.]( SRS\_ETH\_00125)

**[SWS\_EthSwT\_00378]** If EthSwT\_PortEnableTimeStamp is called for a PortIdx, the switch driver shall enable the time-stamping for this port if EthSwTPortTimeStampSupport is set to TRUE for this port.] ( SRS\_ETH\_00125)

**[SWS\_EthSwT\_00245]** The Switch driver shall inform the EthIf about the availability of port specific ingress and egress timestamps using the APIs EthIf\_SwitchIngressTimeStampIndication and EthIf\_SwitchEgressTimeStampIndication, if EthSwTGlobalTimeSupportApi is set to TRUE.]( SRS\_ETH\_00125)

**Note:** Global Time support typically requires the activation of the Switch management support functionality within the Switch device.

#### 7.1.4.12 Counter synchronization of Ethernet switches which are connected via uplink ports

Some Ethernet Switches provide the possibility to synchronize their internal clock. For Ethernet switches which are connected via uplink ports it is not necessary to measure the delay between the connected uplink ports, if the clock synchronization clock is activated. (`EthSwcClockSynchronizationSupport` set to `TRUE`).

**[SWS\_EthSwc\_00408]** The Switch driver shall enable clock synchronization with another Ethernet switch to which it is connected via uplink port, if `EthSwcClockSynchronizationSupport` is set to `TRUE`. ] ( )

#### **[SWS\_EthSwc\_CONSTR\_00409]**

The port specific timestamping (`EthSwcPortTimeStampSupport`) can be set to `TRUE`, if clock synchronization for connected Ethernet switches is deactivated (`EthSwcClockSynchronizationSupport` set to `FALSE`).

] ( )

#### **[SWS\_EthSwc\_CONSTR\_00410]**

The port specific timestamping (`EthSwcPortTimeStampSupport`) can be set to `True`, if `EthSwcClockSynchronizationSupport` is activated and `EthSwcPortRole` is not `ETHSWC_UP_LINK_PORT`. `EthSwcPorts` with `EthSwcPortRole` `ETHSWC_UP_LINK_PORT` are connected to another Ethernet switch and not considered for the time delay compensation, if `EthSwcClockSynchronizationSupport` is activated.

] ( )

#### 7.1.4.13 Verification of Configuration

There are some situations where the Host controller needs to verify the Switch configuration.

**[SWS\_EthSwc\_00292]** If the parameter `EthSwcVerifyConfigApi` is set to `TRUE` the function `EthSwc_VerifyConfig` shall be used to verify switch configuration. ] (SRS\_Eth\_00126)

**Implementation hint:** As Switch configuration is highly HW-Architecture dependent the steps inside the function are implementation specific.

In some use cases, it is necessary to stop frame forwarding during the verification using the optional function `EthSwc_SetForwardingMode`.

The function `EthSwc_VerifyConfig` could for example do the following steps:

1. Stop frame forwarding by calling `EthSwc_SetForwardingMode(FALSE)`.
2. Verify the switch configuration
3. In case the switch configuration is valid then frame forwarding shall be enabled by calling `EthSwc_SetForwardingMode(TRUE)`.(if disabled in step 1)

4. In case the switch configuration is not valid then the switch shall be reset and reconfigured.

**Note:** Please note that a reset of the Host Controller does not necessarily need a reset of the connected Switch HW. This needs to be evaluated individually very carefully as a reset raises the risk of uncontrolled communication during reset phase of the host controller.

**Note:** The Verification of the Switch Configuration as described above is just an example how and when this Verification may be done. It is very dependent on the used switch HW as well as the individual HW-Architecture and even Power supply and Reset strategy of the Switch of the ECU how the Configuration is verified or even how it can be verified. The only thing what this Module specifies is the interface to the upper layer to apply some verification on the switch configuration.

#### 7.1.4.14 Testing and Diagnostic of Switch Ports

If configured, the Ethernet Switch Driver provides following interfaces to apply Testing and diagnostic functionalities

- EthSwt\_GetPortSignalQuality
- EthSwt\_GetPortIdentifier
- EthSwt\_GetSwitchIdentifier
- EthSwt\_WritePortMirrorConfiguration
- EthSwt\_ReadPortMirrorConfiguration
- EthSwt\_GetPortMirrorState
- EthSwt\_SetPortMirrorState
- EthSwt\_SetPortTestMode
- EthSwt\_SetPortLoopbackMode
- EthSwt\_SetPortTxMode
- EthSwt\_GetPortCableDiagnosticsResult
- EthSwt\_GetCfgDataRaw
- EthSwt\_GetCfgDataInfo

The Availability of these functions is strongly depending on the possibilities of the used Transceiver-(Phy)-HW.

#### 7.1.4.15 Low Power Mode Support

**[SWS\_EthSwt\_00376]** If EthSwtLowPowerModeSupport is set to TRUE and at least one EthSwtPort of a Ethernet switch is enabled and the corresponding Ethernet switch HW is in an inactive or low power mode the Ethernet switch HW shall be set to an active mode in which forwarding of Ethernet frames is possible.] ()

**[SWS\_EthSwt\_00377]** If EthSwtLowPowerModeSupport is set to TRUE and no EthSwtPort for a certain Ethernet switch is enabled, the corresponding Ethernet switch HW shall be set to an inactive or low power mode.] ()

## 7.2 Error Classifications

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

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

### 7.2.1 Development Errors

#### [SWS\_EthSwt\_00001] Development Error Types[

Type or error	Related error code	Value [hex]
Invalid switch index	ETHSWT_E_INV_SWITCH_IDX	0x01
EthSwt module was not initialized	ETHSWT_E_UNINIT	0x02
Invalid pointer in parameter list	ETHSWT_E_PARAM_POINTER	0x03
Invalid API which is not available by another module	ETHSWT_E_INV_API	0x05
Invalid switch port index	ETHSWT_E_INV_SWITCHPORT_IDX	0x06
Invalid Controller Index	ETHSWT_E_INV_CTRL_IDX	0x07
Invalid input parameter	ETHSWT_E_INV_PARAM	0x08
Invalid configuration	ETHSWT_E_INIT_FAILED	0x09

] (SRS\_BSW\_00385 )

#### [SWS\_EthSwt\_00009] [

If development error detection is enabled, the function EthSwt\_Init shall check the parameter CfgPtr for being valid. If the check fails, EthSwt\_Init shall raise the development error ETHSWT\_E\_INIT\_FAILED. Note: Please note that in case of variant pre-compile NULL\_PTR is allowed.]( SRS\_BSW\_323, SRS\_BSW\_369)

[SWS\_EthSwt\_00164] [The switch driver shall check whether the lower layer driver, i.e. the EthTrcv provides the APIs which can be called by an upper layer module (EthIf) of the switch driver and will be forwarded to the lower layer. In case of missing APIs, the switch driver shall raise the development error ETHSWT\_E\_INV\_API if APIs are missing in the lower layer module.](SRS\_BSW\_00369, SRS\_BSW\_00386, SRS\_ETH\_00118)

**Note:** This check will be performed upon calling a certain API. For this check the input parameter SwitchPortIdx and a configuration table which needs to be derived from the configuration of the Ethernet transceiver drivers which are attached to the Ethernet switch driver are necessary. This functionality is necessary if development error tracing is activated. This check is necessary because an Ethernet switch driver API can be called by an upper layer module with the argument SwitchPortIdx. This value of this SwitchPortIdx can be in a valid range, but some Ethernet transceiver driver which are used by the switch driver support the API and some do not support this API. In order to resolve this conflict, this check has been implemented.

#### [SWS\_EthSwt\_00156] [

The function EthSwt\_SetSwitchPortMode shall check whether the EthTrcv\_SetTransceiverMode() API of the indexed transceiver driver is available by checking whether for this SwitchPortIdx the corresponding EthTrcv API is available. If this is not the case, the function shall return E\_NOT\_OK and if development error



tracing is activated by EthSwtDevErrorDetect the ETHSWT\_E\_INV\_API shall be raised.] (SRS\_BSW\_00413, SRS\_BSW\_323, SRS\_BSW\_369, SRS\_ETH\_00118)

**[SWS\_EthSwt\_00157]** [

The function EthSwt\_GetSwitchPortMode shall check whether the EthTrcv\_GetTransceiverMode() API of the indexed transceiver driver is available by checking whether for this SwitchPortIdx the corresponding EthTrcv API is available. If this is not the case, the function shall return E\_NOT\_OK and if development error tracing is activated by EthSwtDevErrorDetect the ETHSWT\_E\_INV\_API shall be raised.] (SRS\_BSW\_00413, SRS\_BSW\_323, SRS\_BSW\_369, SRS\_ETH\_00118)

**[SWS\_EthSwt\_00386]** [ If development error detection is enabled, all functions except EthSwt\_Init shall check that the service EthSwt\_Init was previously called. If the check fails, the function shall raise the development error ETHSWT\_E\_UNINIT. ] ( SRS\_BSW\_00350)

**[SWS\_EthSwt\_00387]** [ If development error detection is enabled, all functions with input parameter SwitchIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT\_E\_INV\_SWITCH\_IDX.] ( SRS\_BSW\_00350)

**[SWS\_EthSwt\_00389]**

If development error detection is enabled, all functions with input parameter SwitchPortIdx or PortIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT\_E\_INV\_SWITCHPORT\_IDX] (SRS\_BSW\_00350)

**[SWS\_EthSwt\_00390]** [ If development error detection is enabled, all functions with input parameter CtrlIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT\_E\_INV\_CTRL\_IDX. ] (SRS\_BSW\_00350)

**[SWS\_EthSwt\_00391]** [ If development error detection is enabled, all functions with input parameter BufIdx shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT\_E\_INV\_PARAM. ] (SRS\_BSW\_00350)

**[SWS\_EthSwt\_00392]** [ If development error detection is enabled, all functions with inout or output pointer parameter shall check the parameter for being valid. If the check fails, the functions shall raise the development error ETHSWT\_E\_PARAM\_POINTER.] ( SRS\_BSW\_00350)

**[SWS\_EthSwt\_00393]** [ If development error tracing is activated by EthSwtDevErrorDetect, the functions which call an Ethernet Transceiver API and do not obtain the functionality directly from the switch port interface shall check whether the API of the indexed transceiver driver is available. If this is not the case, the

functions shall raise the development error ETHSWT\_E\_INV\_API. ] (SRS\_BSW\_00350)

**[SWS\_EthSwt\_00154]**

The function EthSwt\_GetLinkState shall check whether the EthTrcv\_GetLinkState() API of the indexed transceiver driver is available by checking whether for this SwitchPortIdx the corresponding EthTrcv API is available. If this is not the case, the function shall return E\_NOT\_OK and if development error tracing is activated by EthSwtDevErrorDetect the ETHSWT\_E\_INV\_API shall be raised.] (SRS\_ETH\_00118, SRS\_ETH\_00119, SRS\_BSW\_00413, SRS\_BSW\_323, SRS\_BSW\_369)

**7.2.2 Runtime Errors**

There are no runtime errors.

**7.2.3 Transient Faults**

There are no transient errors.

**7.2.4 Production Errors**

There are no production errors.

**7.2.5 Extended Production Errors**

**[SWS\_EthSwt\_00113]**

<b>Error Name:</b>	ETHSWT_E_ACCESS	
<b>Short Description:</b>	Ethernet Switch Access Failure	
<b>Long Description:</b>	This production error shall be issued when the switch is not accessible.	
<b>Recommended DTC:</b>	N/A	
<b>Detection Criteria:</b>	Fail	When access to the Ethernet Switch fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
	Pass	When access to the Ethernet Switch succeeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
<b>Secondary Parameters:</b>	N/A	
<b>Time Required:</b>	N/A	
<b>Monitor Frequency</b>	N/A	
<b>MIL illumination:</b>	N/A	

] (SRS\_BSW\_00385)

**[SWS\_EthSwt\_00395]**



<b>Error Name:</b>	ETHSWT_E_SYNCPORT2PHY	
<b>Short Description:</b>	Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes.	
<b>Long Description:</b>	While getting the Ethernet switch port mode, the Ethernet switch driver detected an inconsistent state between Ethernet switch port and the referenced Ethernet transceiver Mode.	
<b>Recommended DTC:</b>	N/A	
<b>Detection Criteria:</b>	Fail	When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found inconsistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
	Pass	When getting the Ethernet switch port mode together with the Ethernet transceiver mode and the mode of the two referenced modules was found consistent the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
<b>Secondary Parameters:</b>	N/A	
<b>Time Required:</b>	N/A	
<b>Monitor Frequency</b>	N/A	
<b>MIL illumination:</b>	N/A	

] (SRS\_BSW\_00385)

## 8 API specification

### 8.1 Imported types

This chapter lists all types included from the following files:

[SWS\_EthSwt\_00002] [

Module	Header File	Imported Type
Dem	Rte_Dem_Type.h	Dem_EventIdType
	Rte_Dem_Type.h	Dem_EventStatusType
Eth_GeneralTypes	Eth_GeneralTypes.h	EthTrcv_BaudRateType
	Eth_GeneralTypes.h	EthTrcv_CableDiagResultType
	Eth_GeneralTypes.h	EthTrcv_DuplexModeType
	Eth_GeneralTypes.h	EthTrcv_LinkStateType
	Eth_GeneralTypes.h	EthTrcv_ModeType
	Eth_GeneralTypes.h	EthTrcv_PhyLoopbackModeType
	Eth_GeneralTypes.h	EthTrcv_PhyTestModeType
	Eth_GeneralTypes.h	EthTrcv_PhyTxModeType
	Eth_GeneralTypes.h	Eth_BufIdxType
	Eth_GeneralTypes.h	Eth_CounterType
	Eth_GeneralTypes.h	Eth_DataType
	Eth_GeneralTypes.h	Eth_MacVlanType
	Eth_GeneralTypes.h	Eth_RxStatsType
	Eth_GeneralTypes.h	Eth_TimeStampType
	Eth_GeneralTypes.h	Eth_TxErrorCounterValuesType
Eth_GeneralTypes.h	Eth_TxStatsType	
NvM	Rte_NvM_Type.h	NvM_BlockIdType
	Rte_NvM_Type.h	NvM_RequestResultType
Spi	Spi.h	Spi_AsyncModeType
	Spi.h	Spi_ChannelType
	Spi.h	Spi_DataBufferType
	Spi.h	Spi_NumberOfDataType
	Spi.h	Spi_SequenceType
Std_Types	StandardTypes.h	Std_ReturnType
	StandardTypes.h	Std_VersionInfoType

] ()

### 8.2 Type definitions

#### 8.2.1 EthSwt\_StateType

[SWS\_EthSwt\_00123] [

<b>Name:</b>	EthSwt_StateType		
<b>Type:</b>	Enumeration		
<b>Range:</b>	ETHSWT_STATE_UNINIT	0x00	Switch is not yet configured
	ETHSWT_STATE_INIT	0x01	Switch is configured
	ETHSWT_STATE_ACTIVE	0x02	Switch is active
<b>Description:</b>	Status supervision used for Development Error Detection. The state shall be available for debugging.		
<b>Available via:</b>	EthSwt.h		

|(SRS\_BSW\_00406)

### 8.2.2 EthSwT\_ConfigType

[SWS\_EthSwT\_00165] |

<b>Name:</b>	EthSwT_ConfigType		
<b>Type:</b>	Structure		
<b>Element:</b>	void	implementation specific	--
<b>Description:</b>	Implementation specific structure of the post build configuration.		
<b>Available via:</b>	EthSwT.h		

|(SRS\_BSW\_00395)

### 8.2.3 EthSwT\_MacLearningType

[SWS\_EthSwT\_00227] |

<b>Name:</b>	EthSwT_MacLearningType		
<b>Type:</b>	Enumeration		
<b>Range:</b>	ETHSWT_MACLEARNING_HWDISABLED	--	If hardware learning disabled, the switch must not learn new MAC addresses
	ETHSWT_MACLEARNING_HWENABLED	--	If hardware learning enabled, the switch learns new MAC addresses
	ETHSWT_MACLEARNING_SWENABLED	--	If software learning enabled, the hardware learning is disabled and the switch forwards packets with an unknown source address to a host CPU
<b>Description:</b>	The interpretation of this value		
<b>Available via:</b>	EthSwT.h		

|(SRS\_ETH\_00087)

### 8.2.4 EthSwT\_MgmtInfoType

[SWS\_EthSwT\_91002] |

<b>Name:</b>	EthSwT_MgmtInfoType		
<b>Type:</b>	Structure		
<b>Element:</b>	uint8	SwitchIdx	Switch index
	uint8	SwitchPortIdx	Port index of the switch
<b>Description:</b>	Type for holding the management information received/transmitted on Switches (ports).		
<b>Available via:</b>	EthSwT.h		

|(SRS\_Eth\_00125)

### 8.2.5 EthSwT\_PortMirrorCfgType

[SWS\_EthSwT\_91017] |

<b>Name:</b>	EthSwT_PortMirrorCfgType		
<b>Type:</b>	Structure		
<b>Element:</b>	uint8 [6]	srcMacAddrFilter	Specifies the source MAC address [0..255,0..255,0..255,0..255,0..255,0..255] that should be mirrored. If set to

			0,0,0,0,0,0, no source MAC address filtering shall take place.
uint8[6]	dstMacAddrFilter		Specifies the destination MAC address [0..255,0..255,0..255,0..255,0..255,0..255] that should be mirrored. If set to 0,0,0,0,0,0, no destination MAC address filtering shall take place.
uint16	VlanIdFilter		Specifies the VLAN address 0..4094 that should be mirrored. If set to 65535, no VLAN filtering shall take place.
uint8	MirroringPacketDivider		Divider if only a subset of received frames should be mirrored. E.g. MirroringPacketDivider = 2 means every second frames is mirrored
uint8	MirroringMode		specifies the mode how the mirrored traffic should be tagged : 0x00 == No VLAN retagging; 0x01 == VLAN retagging; 0x03 == VLAN Double tagging
uint32	TrafficDirectionIngressBitMask		Specifies the bit mask of Ethernet switch ingress port traffic direction to be mirrored. The bit mask is calculated depending of the values of EthSwtPortIdx. (e.g. set EthSwtPortIdx == 2 => TrafficDirectionIngressBitMask = 0b0000 0000 0000 0000 0000 0000 0100). 0b0 == enable ingress port mirroring 0b1 == disable ingress port mirroring  Example: TrafficDirectionIngressBitMask = 0b0000 0000 0000 0000 0000 0000 0100 => Ingress traffic mirroring is enabled of Ethernet switch port with EthSwtPortIdx=2
uint32	TrafficDirectionEgressBitMask		Specifies the bit mask of Ethernet switch egress port traffic direction to be mirrored. The bit mask is calculated depending of the values of EthSwtPortIdx. (e.g. set EthSwtPortIdx == 2 => TrafficDirectionEgressBitMask = 0b0000 0000 0000 0000 0000 0000 0100). 0b0 == enable egress port mirroring 0b1 == disable egress port mirroring  Example: TrafficDirectionEgressBitMask = 0b0000 0000 0000 0000 0000 0000 0001 => Egress traffic mirroring is enabled of Ethernet switch port with EthSwtPortIdx=0
uint8	CapturePortIdx		Specifies the Ethernet switch port which capture the mirrored traffic
uint8	ReTaggingVlanId		Specifies the VLAN address 0..4094 which shall be used for re-tagging if MirroringMode is set to 0x01 (VLAN re-tagging). If the value is set to 65535, the value shall be ignored, because the VLAN address for re-tagging is provided by the Ethernet switch configuration

	uint8	DoubleTaggingVlanId	Specifies the VLAN address 0..4094 which shall be used for double-tagging if MirroringMode is set to 0x02 (VLAN double tagging). If the value is set to 65535, the value shall be ignored, because the VLAN address for double tagging is provided by the Ethernet switch configuration
<b>Description:</b>	The EthSwt_PortMirrorCfgType specify the port mirror configuration which is set up per Ethernet switch. The configuration is written to the Ethernet switch driver by calling EthSwt_WritePortMirrorConfiguration. One port mirror configuration is maintained per Ethernet Switch.		
<b>Available via:</b>	EthSwt.h		

] (SRS\_ETH\_00123)

### 8.2.6 EthSwt\_PortMirrorStateType

[SWS\_EthSwt\_91020] [

<b>Name:</b>	EthSwt_PortMirrorStateType		
<b>Type:</b>	Enumeration		
<b>Range:</b>	PORT_MIRRORING_DISABLED	0x00	port mirroring disabled
	PORT_MIRRORING_ENABLED	0x01	port mirroring enabled
<b>Description:</b>	Type to request or obtain the port mirroring state (enable/disable) for a particular port mirror configuration per Ethernet switch.		
<b>Available via:</b>	EthSwt.h		

] (SRS\_ETH\_00123)

### 8.2.7 EthSwt\_ReturnType

[SWS\_EthSwt\_91033] [

<b>Range:</b>	ETHSWT_PORT_MIRRORING_CONFIGURATION_NOT_SUPPORTED	0x02	port mirroring configuration is not supported by Ethernet switch driver or by the Ethernet switch hardware
<b>Description:</b>	Overlaid return value of Std_ReturnType for Ethernet switch driver API EthSwt_WritePortMirrorConfiguration, if the port mirroring configuration is not supported by Ethernet switch driver or by the Ethernet switch hardware (e.g. the configured mirrored traffic direction (see SWS_EthSwt_91017 "TrafficDirectionIngressBitMask" and "TrafficDirectionEgressBitMask") for ingress and egress traffic of the same port is not supported, or the addressed Ethernet switch ports within the port mirror configuration are not accessible by the Ethernet switch driver)		
<b>Available via:</b>	--		

] ()

### 8.2.8 EthSwt\_MgmtOwner

#### [SWS\_EthSwt\_91035]

<b>Name:</b>	EthSwt_MgmtOwner		
<b>Type:</b>	Enumeration		
<b>Range:</b>	ETHSWT_MGMT_OBJ_UNUSED	0x00	Object unused
	ETHSWT_MGMT_OBJ_OWNED_BY_ETHSWT	0x01	Object used and EthSwt collects needed data
	ETHSWT_MGMT_OBJ_OWNED_BY_UPPER_LAYER	0x02	Object used and the upper layer does calculations
<b>Description:</b>	Holds information if upper layer or EthSwt is owner of mgmt_obj.		
<b>Available via:</b>	EthSwt.h		

] ()

### 8.2.9 EthSwt\_Mgmt\_ObjectType

#### [SWS\_EthSwt\_91037]

<b>Name:</b>	EthSwt_MgmtObjectType		
<b>Type:</b>	Structure		
<b>Element:</b>	EthSwt_MgmtObjectValidType	Validation	The validation information for the mgmt_obj.
	Eth_TimeStampType	IngressTimestamp	The ingress timestamp value out of the switch.
	Eth_TimeStampType	EgressTimestamp	The egress timestamp value out of the switch.
	EthSwt_MgmtInfoType	MgmtInfo	Received/Transmitted Management information of the switches.
	EthSwt_MgmtOwner	Ownership	The ownership of MgmtObj.
<b>Description:</b>	Provides information about all struct member elements. The ownership gives information whether EthSwt has finished its activities in providing all struct member elements.		
<b>Available via:</b>	EthSwt.h		

] ()

#### [SWS\_EthSwt\_00433]

A MgmtObject is just allowed to be owned between EthSwt and only one <UPPER\_LAYER>. The structure element can be identified unambiguously using the DataPtr in Rx- and BufIdx in Tx-context, because both elements are definitively unique within the RxIndication() / TxConfirmation() context.

] ()

### 8.2.10 EthSwt\_MgmtObjectValidType

#### [SWS\_EthSwt\_91036]

<b>Name:</b>	EthSwt_MgmtObjectValidType		
<b>Type:</b>	Structure		
<b>Element:</b>	Std_ReturnType	IngressTimestampValid	IngressTimestampValid shall be set to E_NOT_OK if ingress timestamp is not available

	Std_ReturnType	EgressTimestampValid	EgressTimestampValid shall be set to E_NOT_OK if ingress timestamp is not available.
	Std_ReturnType	MgmtInfoValid	MgmtInfoValid shall be set to E_NOT_OK if ingress timestamp is not available(e.g. timeout).
<b>Description:</b>	Will be set from EthSwt and marks EthSwt_MgmtObject as valid or not. So the upper layer will be able to detect inconsistencies.		
<b>Available via:</b>	EthSwt.h		

] ()

## 8.3 Function definitions

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

### 8.3.1 EthSwt\_Init

#### [SWS\_EthSwt\_00006] [

<b>Service name:</b>	EthSwt_Init		
<b>Syntax:</b>	<pre>void EthSwt_Init(     const EthSwt_ConfigType* CfgPtr )</pre>		
<b>Service ID[hex]:</b>	0x01		
<b>Sync/Async:</b>	Synchronous		
<b>Reentrancy:</b>	Non Reentrant		
<b>Parameters (in):</b>	CfgPtr	Points to the implementation specific structure	
<b>Parameters (inout):</b>	None		
<b>Parameters (out):</b>	None		
<b>Return value:</b>	None		
<b>Description:</b>	Initializes the Ethernet Switch Driver		
<b>Available via:</b>	EthSwt.h		

] (SRS\_BSW\_00101)

#### [SWS\_EthSwt\_00007] [

The function EthSwt\_Init shall store the access to the configuration structure for subsequent API calls. ] (SRS\_BSW\_00101)

#### [SWS\_EthSwt\_00008] [

The function EthSwt\_Init shall change the state of all switches controlled by this Switch Driver from ETHSWT\_STATE\_UNINIT to ETHSWT\_STATE\_INIT. ] (SRS\_BSW\_00101)

[SWS\_EthSwt\_00421] [ The EthSwt shall check for enabled port mirror configuration. The enabled port mirror configuration shall be activated by reconfiguring the Ethernet switch hardware according to the port mirror configuration, before frame forwarding is being enabled. ] (SRS\_Eth\_00123)

#### [SWS\_EthSwt\_00422] [

If the PortMirrorState is set to 0x01 (port mirroring enabled), then the stored port mirror configuration for the given Ethernet switch shall be written to hardware registers of the given Ethernet switch and enable port mirroring.] (SRS\_Eth\_00123)

**[SWS\_EthSwt\_00423]**

If the PortMirrorState is set to 0x00 (port mirroring disabled) the corresponding hardware registers of the given Ethernet switch shall be reset (to the HW's default values) and the port mirroring shall be disabled.] ()

**8.3.2 EthSwt\_SwitchInit**

**[SWS\_EthSwt\_00010]** [

<b>Service name:</b>	EthSwt_SwitchInit	
<b>Syntax:</b>	Std_ReturnType EthSwt_SwitchInit( uint8 SwitchIdx )	
<b>Service ID[hex]:</b>	0x02	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: switch could not be initialized
<b>Description:</b>	Initializes the indexed switch with a given configuration for the switch index	
<b>Available via:</b>	EthSwt.h	

] ()

**[SWS\_EthSwt\_00011]**

EthSwt\_SwitchInit shall:

- if necessary configure all configuration parameters (e.g. port structure, VLAN configuration, ...) at all ports of the switch and the switch itself.
- if necessary perform a soft reset, i.e. resetting the switch via register setting not via a reset pin. This is hardware dependent and might not be supported by all switch devices.
- After resetting the switch and EthSwtLowPowerModeSupport set to TRUE, the Ethernet switch shall enter an inactive or low power mode. If EthSwtLowPowerModeSupport is not defined or set to FALSE the Ethernet switch shall enter an active state

] (SRS\_BSW\_00101)

Note: Please Note that this function can take a very long time to complete and in this case can not be called by EcuM or BswM. Instead it e.g. could be called from a background task.

**[SWS\_EthSwt\_00374]** All Ethernet switch HW ports which are not configured as a EthSwtPort shall be switched off during initialization. This Ethernet switch HW ports shall never be switched on during runtime] ()



[SWS\_EthSwt\_00375] [ All EthSwtPorts shall be set to "ETHTRCV\_MODE\_DOWN" during initialization.] ()

[SWS\_EthSwt\_00012]

EthSwt\_SwitchInit shall change the state of the indexed switch from ETHSWT\_STATE\_INIT to ETHSWT\_STATE\_ACTIVE. ] (SRS\_BSW\_00101)

[SWS\_EthSwt\_00016]

The function EthSwt\_SwitchInit shall check the access to the Ethernet Switch hardware, i.e. by trying to read or write registers during the configuration of the switch. If the access to the registers fails, the function shall raise the extended production error ETHSWT\_E\_ACCESS and return E\_NOT\_OK.]( SRS\_BSW\_00386)

Note: Access to the Ethernet Switch hardware is device dependent, e.g. access through the Ethernet Controller Mii, access through SPI, ... etc.

### 8.3.3 EthSwt\_SetSwitchPortMode

[SWS\_EthSwt\_00018] [

<b>Service name:</b>	EthSwt_SetSwitchPortMode	
<b>Syntax:</b>	Std_ReturnType EthSwt_SetSwitchPortMode( uint8 SwitchIdx, uint8 SwitchPortIdx, EthTrcv_ModeType PortMode )	
<b>Service ID[hex]:</b>	0x03	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
	PortMode	ETHTRCV_MODE_DOWN: disable the addressed port at the switch ETHTRCV_MODE_ACTIVE: enable the addressed port at the switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: The indexed switch port could not be set to PortMode
<b>Description:</b>	Enables/disables the indexed switch port	
<b>Available via:</b>	EthSwt.h	

] (SRS\_ETH\_00118)

[SWS\_EthSwt\_00019] [

The function EthSwt\_SetSwitchPortMode() shall put the indexed port of the switch into the specified mode. If EthSwtPort references an EthTrcv then the function EthTrcv\_SetTransceiverMode() of the Ethernet Transceiver Driver shall additionally be called with the corresponding transceiver mode. ] ( SRS\_ETH\_00118)

**[SWS\_EthSwt\_00396]** [ When calling the function EthSwt\_SetSwitchPortMode() with mode ETHTRCV\_MODE\_DOWN, the EthSwt shall disable the Ethernet switch port directly for reduction of power consumption, if it is possible.] ()

**[SWS\_EthSwt\_00397]** [ When calling the function EthSwt\_SetSwitchPortMode(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT\_E\_ACCESS and return E\_NOT\_OK, otherwise pass the extended production error ETHSWT\_E\_ACCESS and return E\_OK.] ()

**[SWS\_EthSwt\_00398]** [ If EthSwtPort does not references an EthTrcv, EthSwt shall indicate a mode of the port by the API EthIf\_SwitchPortModeIndication latest during the next EthSwt\_MainFunction.(SRS\_ETH\_00118)] ()

**[SWS\_EthSwt\_00022]** [ The function EthSwt\_SetSwitchPortMode shall be pre compile time configurable On/Off by the configuration parameter: EthSwtSetSwitchPortModeApi. ] (SRS\_BSW\_00171)

**[SWS\_EthSwt\_00023]** [ If the switch is already in the requested mode E\_OK shall be returned and no development error shall be raised. ] (SRS\_ETH\_00118)

### 8.3.4 EthSwt\_GetSwitchPortMode

**[SWS\_EthSwt\_00025]** [

<b>Service name:</b>	EthSwt_GetSwitchPortMode	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_GetSwitchPortMode(     uint8 SwitchIdx,     uint8 SwitchPortIdx,     EthTrcv_ModeType* SwitchModePtr )</pre>	
<b>Service ID[hex]:</b>	0x04	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	SwitchModePtr	ETHTRCV_MODE_DOWN: the port of the switch is disabled ETHTRCV_MODE_ACTIVE: the port of the switch is enabled
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: The mode of the indexed switch port could not be obtained.
<b>Description:</b>	Obtains the mode of the indexed switch port	
<b>Available via:</b>	EthSwt.h	

] (SRS\_ETH\_00118)

**[SWS\_EthSwt\_00026]** [

The function `EthSwt_GetSwitchPortMode` shall read the mode of the indexed port of the switch. If `EthSwtPort` references an `EthTrcv` then the function shall additionally call the corresponding function `EthTrcv_GetTransceiverMode` of the Ethernet Transceiver Driver. ] (SRS\_ETH\_00118)

**[SWS\_EthSwt\_00399]**

If the obtained modes of the `EthSwtPort` and the `EthTrcv` are not aligned, the function `EthSwt_GetSwitchPortMode` shall raise the extended production error `ETHSWT_E_SYNCPORT2PHY` and return `E_NOT_OK`.

If `EthTrcv_GetTransceiverMode` returns `E_NOT_OK`, the `EthSwt_GetSwitchPortMode` shall also return `E_NOT_OK` without raising an error. ]()

**[SWS\_EthSwt\_00400]** If the function `EthSwt_GetSwitchPortMode()` is called, the function shall check the access to the Ethernet Switch Driver. If the check fails, the function shall raise the extended production error `ETHSWT_E_ACCESS` and return `E_NOT_OK`, otherwise pass the production error `ETHSWT_E_ACCESS` and return `E_OK`. ]()

**[SWS\_EthSwt\_00029]** [

The function `EthSwt_GetSwitchPortMode` shall be pre compile time configurable On/Off by the configuration parameter: `EthSwtGetSwitchPortModeApi`. ] (SRS\_BSW\_00171)

### 8.3.5 EthSwt\_StartSwitchPortAutoNegotiation

**[SWS\_EthSwt\_00031]** [

<b>Service name:</b>	EthSwt_StartSwitchPortAutoNegotiation	
<b>Syntax:</b>	Std_ReturnType EthSwt_StartSwitchPortAutoNegotiation( uint8 SwitchIdx, uint8 SwitchPortIdx )	
<b>Service ID[hex]:</b>	0x05	
<b>Sync/Async:</b>	Asynchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: Automatic negotiation could not be started for the indexed switch port.
<b>Description:</b>	Starts the auto-negotiation of the indexed switch port	
<b>Available via:</b>	EthSwt.h	

] (SRS\_ETH\_00087)

**[SWS\_EthSwt\_00032]** [

The function `EthSwt_StartSwitchPortAutoNegotiation` shall restart the automatic negotiation of the used transmission parameters of the referenced Ethernet transceiver driver by calling the function `EthTrcv_StartAutoNegotiation()`.] (SRS\_ETH\_00087)

**[SWS\_EthSwt\_00035]** [

The function `EthSwt_StartSwitchPortAutoNegotiation` shall be pre compile time configurable On/Off by the configuration parameter: `EthSwtStartSwitchPortAutoNegotiationApi`. ] (SRS\_BSW\_00171)

**8.3.6 EthSwt\_GetLinkState**

**[SWS\_EthSwt\_00037]** [

<b>Service name:</b>	EthSwt_GetLinkState	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_GetLinkState (     uint8 SwitchIdx,     uint8 SwitchPortIdx,     EthTrcv_LinkStateType* LinkStatePtr )</pre>	
<b>Service ID[hex]:</b>	0x06	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	LinkStatePtr	ETHTRCV_LINK_STATE_DOWN: Switch port is disconnected ETHTRCV_LINK_STATE_ACTIVE: Switch port is connected
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: Link state of the indexed switch port could not be obtained
<b>Description:</b>	Obtains the link state of the indexed switch port	
<b>Available via:</b>	EthSwt.h	

] (SRS\_ETH\_00119)

**[SWS\_EthSwt\_00038]** [

The function `EthSwt_GetLinkState` shall read the current (link) state of the indexed switch port. If the indexed Ethernet port references an Ethernet transceiver, the link state shall be obtained by calling the function `EthTrcv_GetLinkState()` of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the state shall be obtained from the MAC interface of the Switch port. ] (SRS\_ETH\_00118, SRS\_ETH\_00119)

**[SWS\_EthSwt\_00042]** [

The function `EthSwt_GetLinkState` shall be pre compile time configurable On/Off by the configuration parameter: `EthSwtGetLinkStateApi`. ] (SRS\_BSW\_00171)

### 8.3.7 EthSwt\_GetBaudRate

#### [SWS\_EthSwt\_00044] [

<b>Service name:</b>	EthSwt_GetBaudRate	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_GetBaudRate(     uint8 SwitchIdx,     uint8 SwitchPortIdx,     EthTrcv_BaudRateType* BaudRatePtr )</pre>	
<b>Service ID[hex]:</b>	0x07	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	BaudRatePtr	ETHTRCV_BAUD_RATE_10MBIT: 10MBit connection ETHTRCV_BAUD_RATE_100MBIT: 100MBit connection ETHTRCV_BAUD_RATE_1000MBIT: 1000MBit connection
	Std_ReturnType	E_OK: success
		E_NOT_OK: Baud rate of the indexed switch port could not be obtained
<b>Return value:</b>		
<b>Description:</b>	Obtains the baud rate of the indexed switch port	
<b>Available via:</b>	EthSwt.h	

] (SRS\_ETH\_00118)

#### [SWS\_EthSwt\_00045] [

The function EthSwt\_GetBaudRate() shall read the current baud rate of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the baud rate shall be obtained by the function EthTrcv\_GetBaudRate() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the baud rate shall be obtained from the MAC interface of the Switch port.] (SRS\_ETH\_00118)

#### [SWS\_EthSwt\_00049] [

The function EthSwt\_GetBaudRate shall be pre compile time configurable On/Off by the configuration parameter: EthSwtGetBaudRateApi. ] (SRS\_BSW\_00171)

### 8.3.8 EthSwt\_GetDuplexMode

#### [SWS\_EthSwt\_00051] [

<b>Service name:</b>	EthSwt_GetDuplexMode	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_GetDuplexMode(     uint8 SwitchIdx,     uint8 SwitchPortIdx,     EthTrcv_DuplexModeType* DuplexModePtr )</pre>	
<b>Service ID[hex]:</b>	0x08	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver

	SwitchPortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	DuplexModePtr	ETHTRCV_DUPLEX_MODE_HALF: half duplex connections ETHTRCV_DUPLEXMODE_FULL: full duplex connection
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: duplex mode of the indexed switch port could not be obtained
<b>Description:</b>	Obtains the duplex mode of the indexed switch port	
<b>Available via:</b>	EthSwt.h	

] (SRS\_ETH\_00118)

**[SWS\_EthSwt\_00052] [**

The function EthSwt\_GetDuplexMode() shall read the current duplex mode of the indexed switch port. If the indexed Ethernet port reference an Ethernet transceiver, the duplex mode shall be obtained by calling the function EthTrcv\_GetDuplexMode() of the Ethernet Transceiver Driver. If the indexed Ethernet Switch port does not reference an Ethernet transceiver, the duplex mode shall be obtained from the MAC interface of the Switch port.] (SRS\_ETH\_00118)

**[SWS\_EthSwt\_00056] [**

The function EthSwt\_GetDuplexMode shall be pre compile time configurable On/Off by the configuration parameter: EthSwtGetDuplexModeApi. ] (SRS\_BSW\_00171)

**8.3.9 EthSwt\_GetPortMacAddr**

**[SWS\_EthSwt\_00060] [**

<b>Service name:</b>	EthSwt_GetPortMacAddr	
<b>Syntax:</b>	Std_ReturnType EthSwt_GetPortMacAddr ( uint8 SwitchIdx, const uint8* MacAddrPtr, uint8* PortIdxPtr )	
<b>Service ID[hex]:</b>	0x09	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	MacAddrPtr	MAC-address for which a switch port is searched over which the node with this MAC-address can be reached.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	PortIdxPtr	Pointer to the port index
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: multiple ports were found
<b>Description:</b>	Obtains the port over which this MAC-address at the indexed switch can be reached. The result might be used for a DHCP-server which will need the port/MAC-resolution. If for the PortIdxPtr the maximal possible value (255) is returned the given MAC address cannot be reached via a port of this switch. If multiple ports were found the API returns E_NOT_OK.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_ETH\_00087)

**[SWS\_EthSwt\_00061]** |

The function EthSwt\_GetPortMacAddr shall return the port index over which the given MAC-address is reachable within the indexed switch. If for the PortIdxPtr the maximal possible value (255) is returned the given MAC address cannot be reached via a port of this switch. If multiple ports were found the API returns E\_NOT\_OK.  
| (SRS\_ETH\_00087)

**[SWS\_EthSwt\_00230]** |

The function EthSwt\_GetPortMacAddr shall be pre compile time configurable On/Off by the configuration parameter: EthSwtGetPortMacAddrApi. | (SRS\_BSW\_00171)

**8.3.10 EthSwt\_GetArITable**

**[SWS\_EthSwt\_00111]** |

<b>Service name:</b>	EthSwt_GetArITable	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_GetArITable(     uint8 switchIdx,     uint16* numberOfElements,     Eth_MacVlanType* arItableListPointer )</pre>	
<b>Service ID[hex]:</b>	0x0a	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	switchIdx	Index of the switch within the context of the Ethernet Switch Driver
<b>Parameters (inout):</b>	numberOfElements	In: Maximum number of elements which can be written into the arItable Out: Number of elements which are currently available in the EthSwitch module.
<b>Parameters (out):</b>	arItableListPointer	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.
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: requested switchIdx is not valid or inactive
<b>Description:</b>	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. arItableListPointer may be NULL_PTR in this case.	
<b>Available via:</b>	EthSwt.h	

| (SRS\_ETH\_00087)

**[SWS\_EthSwt\_00228]** |

The function EthSwt\_GetArITable shall provide a list of structs with MAC-address, VLAN-ID and port for the indexed switch. | (SRS\_ETH\_00087)

**[SWS\_EthSwt\_00197]** |

If the numberOfElements is greater 0x00, the arItableListPointer shall be filled with up to numberOfElements elements. numberOfElements shall return the number of copied elements. | (SRS\_ETH\_00087)



**[SWS\_EthSwt\_00235]** [ The EthSwt\_GetArITable API shall return only the numberOfElements if the numberOfElements is set to 0x00. In this case no data will be copied and a NULLPTR can be used for the arItableListPointer. ] ( SRS\_ETH\_00087)

**[SWS\_EthSwt\_00229]** [ The function EthSwt\_GetArITable shall be pre compile time configurable On/Off by the configuration parameter: EthSwtGetArItableApi. ] (SRS\_BSW\_00171)

### 8.3.11 EthSwt\_GetCounterValues

**[SWS\_EthSwt\_00231]** [

<b>Service name:</b>	EthSwt_GetCounterValues	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_GetCounterValues (     uint8 SwitchIdx,     uint8 SwitchPortIdx,     Eth_CounterType* CounterPtr )</pre>	
<b>Service ID[hex]:</b>	0x0c	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	CounterPtr	counter values according to IETF RFC 1757, RFC 1643 and RFC 2233.
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: counter values read failure
<b>Description:</b>	Reads a list with drop counter values of the corresponding port of the switch. The meaning of these values is described at Eth_CounterType.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00128)

**[SWS\_EthSwt\_00106]** [ EthSwt\_GetCounterValues shall read a list with drop counter values of the corresponding port of the switch. The meaning of these values is described at Eth\_CounterType. ] (SRS\_ETH\_00128)

**[SWS\_EthSwt\_00109]** [ The function EthSwt\_GetCounterValues shall be pre compile time configurable On/Off by the configuration parameter: EthSwtGetCounterValuesApi. ] (SRS\_BSW\_00171)

### 8.3.12 EthSwt\_GetRxStats

#### [SWS\_EthSwt\_00198] [

<b>Service name:</b>	EthSwt_GetRxStats	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_GetRxStats(     uint8 SwitchIdx,     uint8 SwitchPortIdx,     Eth_RxStatsType* RxStats )</pre>	
<b>Service ID[hex]:</b>	0x0d	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	RxStats	List of values according to IETF RFC 2819 (Remote Network Monitoring Management Information Base)
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: drop counter could not be obtained
<b>Description:</b>	Returns a list of statistic counters defined with Eth_RxTatsType. The majority of these Counters are derived from the IETF RFC2819.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00128)

**[SWS\_EthSwt\_00199] [** EthSwt\_GetRxStats shall return a list of statistic counters defined with Eth\_RxStatsType. The majority of these Counters are derived from the IETF RFC2819. **](** (SRS\_ETH\_00128)

#### [SWS\_EthSwt\_00202] [

The function EthSwt\_GetRxStats shall be pre compile time configurable On/Off by the configuration parameter: EthSwtGetRxStatsApi. **]** (SRS\_BSW\_00171)

### 8.3.13 EthSwt\_GetTxStats

#### [SWS\_EthSwt\_91001] [

<b>Service name:</b>	EthSwt_GetTxStats	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_GetTxStats(     uint8 SwitchIdx,     uint8 SwitchPortIdx,     Eth_TxStatsType* TxStats )</pre>	
<b>Service ID[hex]:</b>	0x20	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	TxStats	List of values to read statistic values for transmission.
<b>Return value:</b>	Std_ReturnType	E_OK: success

	E_NOTOK: Tx-statistics could not be obtained
<b>Description:</b>	Returns the list of Transmission Statistics out of IETF RFC1213 defined with Eth_TxStatsType, where the maximal possible value shall denote an invalid value, e.g. this counter is not available.
<b>Available via:</b>	EthSwt.h

] (SRS\_Eth\_00128)

**[SWS\_EthSwt\_00372]** EthSwt\_GetTxStats shall return the list of Transmission Statistics out of IETF RFC1213 defined with Eth\_TxStatsType, where the maximal possible value shall denote an invalid value, e.g. this counter is not available. ] (SRS\_ETH\_00128)

**[SWS\_EthSwt\_00362]** The function EthSwt\_GetTxStats shall be pre compile time configurable On/Off by the configuration parameter: EthSwtGetTxStatsApi. ] (SRS\_BSW\_00171)

### 8.3.14 EthSwt\_GetTxErrorCounterValues

**[SWS\_EthSwt\_91000]** [

<b>Service name:</b>	EthSwt_GetTxErrorCounterValues	
<b>Syntax:</b>	Std_ReturnType EthSwt_GetTxErrorCounterValues ( uint8 SwitchIdx, uint8 SwitchPortIdx, Eth_TxErrorCounterValuesType* TxStats )	
<b>Service ID[hex]:</b>	0x21	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Drive
	SwitchPortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	TxStats	List of values to read statistic error counter values for transmission.
<b>Return value:</b>	Std_ReturnType	E_OK: success, E_NOTOK: Tx-statistics could not be obtained
<b>Description:</b>	Returns the list of Transmission Error Counters out of IETF RFC1213 and RFC1643 defined with Eth_TxErrorCounterValuesType, where the maximal possible value shall denote an invalid value, e.g. this counter is not available.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00128)

**[SWS\_EthSwt\_00373]** EthSwt\_GetTxErrorCounterValues returns the list of Transmission Error Counters out of IETF RFC1213 and RFC1643 defined with Eth\_TxErrorCounterValuesType, where the maximal possible value shall denote an invalid value, e.g. this counter is not available. ] ( SRS\_ETH\_00128)

**[SWS\_EthSwt\_00370]** The function EthSwt\_GetTxErrorCounterValues shall be pre compile time configurable On/Off by the configuration parameter: EthSwtGetTxErrorCounterValuesApi. ] (SRS\_BSW\_00171)

### 8.3.15 EthSwt\_GetSwitchReg

#### [SWS\_EthSwt\_00206] [

<b>Service name:</b>	EthSwt_GetSwitchReg	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_GetSwitchReg(     uint8 SwitchIdx,     uint32 page,     uint32 register,     uint32* registerContent )</pre>	
<b>Service ID[hex]:</b>	0x0e	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	page	Address of a register page
	register	Address of a register
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	registerContent	Content of the addresses register
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: drop counter could not be obtained
<b>Description:</b>	Generic API for reading the content of a switch register	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00120)

#### [SWS\_EthSwt\_00207] [

The function EthSwt\_GetSwitchReg shall read the content of a switch register. ](SRS\_Eth\_00120)

#### [SWS\_EthSwt\_00210] [

The function EthSwt\_GetSwitchReg shall be pre compile time configurable On/Off by the configuration parameter: EthSwtGetSwitchRegApi. ] (SRS\_BSW\_00171)

### 8.3.16 EthSwt\_SetSwitchReg

#### [SWS\_EthSwt\_00211] [

<b>Service name:</b>	EthSwt_SetSwitchReg	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_SetSwitchReg(     uint8 SwitchIdx,     uint32 page,     uint32 register,     uint32 registerContent )</pre>	
<b>Service ID[hex]:</b>	0x0f	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	page	Address of a register page
	register	Address of a register
	registerContent	Content of the addresses register
<b>Parameters</b>	None	

<b>(inout):</b>	
<b>Parameters (out):</b>	None
<b>Return value:</b>	Std_ReturnType E_OK: success E_NOT_OK: drop counter could not be obtained
<b>Description:</b>	Generic API for writing the content of a switch register
<b>Available via:</b>	EthSwt.h

] (SRS\_Eth\_00120)

**[SWS\_EthSwt\_00212] [**

The function EthSwt\_SetSwitchReg shall write the content of a switch register. ](SRS\_Eth\_00120)

**[SWS\_EthSwt\_00215] [**

The function EthSwt\_SetSwitchReg shall be pre compile time configurable On/Off by the configuration parameter: EthSwtSetSwitchRegApi. ] (SRS\_BSW\_00171)

### 8.3.17 EthSwt\_ReadTrcvRegister

**[SWS\_EthSwt\_00216] [**

<b>Service name:</b>	EthSwt_ReadTrcvRegister	
<b>Syntax:</b>	Std_ReturnType EthSwt_ReadTrcvRegister( uint8 SwitchIdx, uint8 SwitchPortIdx, uint8 RegIdx, uint16* RegValPtr )	
<b>Service ID[hex]:</b>	0x10	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
	RegIdx	Index of the register
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	RegValPtr	Pointer to the register content
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: drop counter could not be obtained
<b>Description:</b>	Generic API for reading the content of a transceiver register	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00120)

**[SWS\_EthSwt\_00217] [**

The function EthSwt\_ReadTrcvRegister shall read the specified transceiver register through the MII or SPI of the indexed switch port. ](SRS\_ETH\_00118, SRS\_ETH\_00120)

**[SWS\_EthSwt\_00220] [**

The function EthSwt\_ReadTrcvRegister shall be pre compile time configurable On/Off by the configuration parameter: EthSwtReadTrcvRegisterApi. ] (SRS\_BSW\_00171)

### 8.3.18 EthSwt\_WriteTrcvRegister

#### [SWS\_EthSwt\_00221] [

<b>Service name:</b>	EthSwt_WriteTrcvRegister	
<b>Syntax:</b>	Std_ReturnType EthSwt_WriteTrcvRegister( uint8 SwitchIdx, uint8 SwitchPortIdx, uint8 RegIdx, uint16 RegVal )	
<b>Service ID[hex]:</b>	0x11	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
	RegIdx	Index of the register
	RegVal	Content for the indexed register
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: drop counter could not be obtained
<b>Description:</b>	Generic API for writing the content of a transceiver register	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00120)

#### [SWS\_EthSwt\_00222] [

The function EthSwt\_WriteTrcvRegister shall write the specified transceiver register through the MII or SPI of the indexed switch port. ](SRS\_ETH\_00118, SRS\_ETH\_00120)

#### [SWS\_EthSwt\_00225] [

The function EthSwt\_WriteTrcvRegister shall be pre compile time configurable On/Off by the configuration parameter: EthSwtWriteTrcvRegisterApi. ] (SRS\_BSW\_00171)

### 8.3.19 EthSwt\_EnableVlan

#### [SWS\_EthSwt\_00172] [

<b>Service name:</b>	EthSwt_EnableVlan	
<b>Syntax:</b>	Std_ReturnType EthSwt_EnableVlan( uint8 SwitchIdx, uint8 SwitchPortIdx, uint16 VlanId, boolean Enable	

	)
<b>Service ID[hex]:</b>	0x12
<b>Sync/Async:</b>	Synchronous /Asynchronous
<b>Reentrancy:</b>	Non Reentrant
<b>Parameters (in):</b>	SwitchIdx   Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx   Index of the port at the addressed switch
	VlanId   VLAN-ID to a preconfigured configuration on the given ingress port
	Enable   1 = VLAN-configuration enabled 0 = VLAN-configuration disabled (frames with given VLAN-ID will be dropped)
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	Std_ReturnType   E_OK: success E_NOT_OK: buffer level could not be obtained
<b>Description:</b>	Enables or disables a pre-configured VLAN at a certain port of a switch.
<b>Available via:</b>	EthSwt.h

] (SRS\_ETH\_00121, SRS\_ETH\_00114)

#### [SWS\_EthSwt\_00173] [

The function EthSwt\_EnableVlan shall enable or disable a pre-configured VLAN at a certain port of a switch. ](SRS\_ETH\_00121, SRS\_ETH\_00114)

#### [SWS\_EthSwt\_00177] [

The function EthSwt\_EnableVlan shall be pre compile time configurable On/Off by the configuration parameter: EthSwtEnableVlanApi. ] (SRS\_BSW\_00171)

### 8.3.20 EthSwt\_StoreConfiguration

#### [SWS\_EthSwt\_00086] [

<b>Service name:</b>	EthSwt_StoreConfiguration
<b>Syntax:</b>	Std_ReturnType EthSwt_StoreConfiguration( uint8 SwitchIdx )
<b>Service ID[hex]:</b>	0x13
<b>Sync/Async:</b>	Synchronous /Asynchronous
<b>Reentrancy:</b>	Non Reentrant
<b>Parameters (in):</b>	SwitchIdx   Index of the switch within the context of the Ethernet Switch Driver
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	Std_ReturnType   E_OK: success E_NOT_OK: Configuration could not be persistently stored
<b>Description:</b>	Stores the configuration of the learned MAC/Port tables of a switch in a persistent manner and will be used by e.g. CDD.
<b>Available via:</b>	EthSwt.h

] (SRS\_ETH\_00087, SRS\_ETH\_00122)

#### [SWS\_EthSwt\_00087] [



The function EthSwt\_StoreConfiguration shall store the configuration of the learned MAC/Port tables of a switch in a persistent manner. This can be done in two ways: 1.) Reading out the parameters and storing them in the NV-RAM of the host CPU using the NV-RAM manager. 2.) Advising the switch to store the configuration data in its local NV-RAM.] (SRS\_ETH\_00087, SRS\_ETH\_00122)

**[SWS\_EthSwt\_00090] [**

The function EthSwt\_StoreConfiguration shall be pre compile time configurable On/Off by the configuration parameter: EthSwtStoreConfigurationApi. ] (SRS\_BSW\_00171)

**8.3.21 EthSwt\_ResetConfiguration**

**[SWS\_EthSwt\_00091] [**

<b>Service name:</b>	EthSwt_ResetConfiguration	
<b>Syntax:</b>	Std_ReturnType EthSwt_ResetConfiguration( uint8 SwitchIdx )	
<b>Service ID[hex]:</b>	0x14	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: configuration could not be persistently resetted
<b>Description:</b>	Resets the configuration of the learned MAC/Port tables of a switch in a persistent manner and will be used by e.g. CDD. The statically configured entries shall still remain.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_ETH\_00087, SRS\_ETH\_00122)

**[SWS\_EthSwt\_00092] [**

The function EthSwt\_ResetConfiguration shall reset the configuration of the learned MAC/Port tables of a switch in a persistent manner. This can be done in two ways: 1.) Overwriting the learned parameters in the NV-RAM of the host CPU with preconfigured default values. 2.) Advising the switch to reset the learned configuration data in its local NV-RAM. ] (SRS\_ETH\_00122, SRS\_ETH\_00087)

**[SWS\_EthSwt\_00095] [**

The function EthSwt\_ResetConfiguration shall be pre compile time configurable On/Off by the configuration parameter: EthSwtResetConfigurationApi. ] (SRS\_BSW\_00171)

**8.3.22 EthSwt\_SetMacLearningMode**

**[SWS\_EthSwt\_00182] [**

<b>Service name:</b>	EthSwt_SetMacLearningMode
----------------------	---------------------------

<b>Syntax:</b>	Std_ReturnType EthSwt_SetMacLearningMode ( uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_MacLearningType MacLearningMode )	
<b>Service ID[hex]:</b>	0x15	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch
	MacLearningMode	Defines whether MAC addresses shall be learned and if they shall be learned in software or hardware.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: configuration could be persistently reset
	<b>Description:</b> Sets the MAC learning mode in one of the tree modes: 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.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_ETH\_00087, SRS\_ETH\_00122)

**[SWS\_EthSwt\_00183] [**

The function EthSwt\_SetMacLearningMode shall set the MAC learning mode according to EthSwt\_MacLearningType.] (SRS\_ETH\_00122, SRS\_ETH\_00087)

**Note:** This feature is hardware dependent, i.e. the switch hardware needs to support the different modes.

**[SWS\_EthSwt\_00186] [**

The function EthSwt\_SetMacLearningMode shall be pre compile time configurable On/Off by the configuration parameter: EthSwtSetMacLearningModeAPI. ] (SRS\_BSW\_00171)

### 8.3.23 EthSwt\_GetMacLearningMode

**[SWS\_EthSwt\_00187] [**

<b>Service name:</b>	EthSwt_GetMacLearningMode	
<b>Syntax:</b>	Std_ReturnType EthSwt_GetMacLearningMode ( uint8 SwitchIdx, uint8 SwitchPortIdx, EthSwt_MacLearningType* MacLearningMode )	
<b>Service ID[hex]:</b>	0x16	
<b>Sync/Async:</b>	Synchronous /Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	SwitchPortIdx	Index of the port at the addressed switch

<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	MacLearningMode	Defines whether MAC addresses shall be learned and if they shall be learned in software or hardware.
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: configuration could be persistently reset
<b>Description:</b>	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	
<b>Available via:</b>	EthSwt.h	

] (SRS\_ETH\_00087)

**[SWS\_EthSwt\_00188] [**

The function EthSwt\_GetMacLearningMode shall return the MAC learning mode according to EthSwt\_MacLearningType. ] (SRS\_ETH\_00087)

**Note:** This feature is hardware dependent, i.e. the switch hardware needs to support the different learning modes.

**[SWS\_EthSwt\_00191] [**

The function EthSwt\_GetMacLearningMode shall be pre compile time configurable On/Off by the configuration parameter: EthSwtGetMacLearningModeApi. ] (SRS\_BSW\_00171)

### 8.3.24 EthSwt\_NvmSingleBlockCallback

**[SWS\_EthSwt\_00125] [**

<b>Service name:</b>	EthSwt_NvmSingleBlockCallback	
<b>Syntax:</b>	Std_ReturnType EthSwt_NvmSingleBlockCallback( uint8 ServiceId, NvM_RequestResultType JobResult )	
<b>Service ID[hex]:</b>	0x17	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	ServiceId	Unique Service ID of NVRAM manager service
	JobResult	Covers the job result of the previous processed single block job.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: Callback function has not been processed successfully
<b>Description:</b>	Function will be called by the NVRAMManager after the switch configuration has been stored or resetted.	
<b>Available via:</b>	EthSwt_NvM.h	

] (SRS\_ETH\_00087, SRS\_ETH\_00122)

**[SWS\_EthSwt\_00126] [**

The function `EthSwt_NvmSingleBlockCallback` shall be called by the NVRAMManager after the switch configuration has been stored or reset in the the NV RAM. ] (SRS\_ETH\_00122, SRS\_ETH\_00087)

**[SWS\_EthSwt\_00196] [**

The function `EthSwt_NvmSingleBlockCallback` shall call the function `<user>_PersistentConfigurationResult` to provide the `JobResult` to the caller of `EthSwt_StoreConfiguration` or `EthSwt_ResetConfiguration`. ] (SRS\_ETH\_00122, SRS\_ETH\_00087)

**[SWS\_EthSwt\_00127] [**

The function `EthSwt_NvmSingleBlockCallback` shall always return `E_OK` according to `SWS_NvM_00368`. ] (SRS\_ETH\_00122, SRS\_ETH\_00087)

**[SWS\_EthSwt\_00128] [**

The function `EthSwt_NvmSingleBlockCallback` shall raise a development error if the `JobResult` equals `NVM_REQ_NOT_OK`, i.e. the write request has been finished unsuccessfully. Please note that a production error at this point is not necessary because the `NvM` will raise also a production error if the write to NV RAM was not successful. ] (SRS\_BSW\_00369, SRS\_ETH\_00458)

**[SWS\_EthSwt\_00129] [**

The function `EthSwt_NvmSingleBlockCallback` shall be pre compile time configurable On/Off by the existence of the container `EthSwtNvm`. ] (SRS\_BSW\_00171)

### 8.3.25 EthSwt\_GetVersionInfo

**[SWS\_EthSwt\_00058] [**

<b>Service name:</b>	<code>EthSwt_GetVersionInfo</code>
<b>Syntax:</b>	<pre>void EthSwt_GetVersionInfo(     Std_VersionInfoType* VersionInfoPtr )</pre>
<b>Service ID[hex]:</b>	<code>0x18</code>
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Reentrant
<b>Parameters (in):</b>	None
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	<code>VersionInfoPtr</code>   Pointer to where to store the version information of this module.
<b>Return value:</b>	None
<b>Description:</b>	Returns the version information of this module.
<b>Available via:</b>	<code>EthSwt.h</code>

] (SRS\_BSW\_00171)

**[SWS\_EthSwt\_00124]** [

The function EthSwt\_GetVersionInfo shall be pre compile time configurable On/Off by the configuration parameter: EthSwtVersionInfoApi. ] (SRS\_BSW\_00171)

**8.3.26 EthSwt\_MgmtInit**

**[SWS\_EthSwt\_91003]** [

<b>Service name:</b>	EthSwt_MgmtInit
<b>Syntax:</b>	void EthSwt_MgmtInit( void )
<b>Service ID[hex]:</b>	0x22
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Non Reentrant
<b>Parameters (in):</b>	None
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	None
<b>Description:</b>	Function initializes additional Switch Management handling.
<b>Available via:</b>	EthSwt.h

] (SRS\_Eth\_00125)

**[SWS\_EthSwt\_00246]** The function EthSwt\_MgmtInit() shall be pre compile time configurable ON/OFF by the configuration parameter: EthSwtManagementSupportApi.] (SRS\_BSW\_00171)

**8.3.27 EthSwt\_EthRxProcessFrame**

**[SWS\_EthSwt\_91004]** [

<b>Service name:</b>	EthSwt_EthRxProcessFrame	
<b>Syntax:</b>	Std_ReturnType EthSwt_EthRxProcessFrame( uint8 CtrlIdx, Eth_BufIdxType BufIdx, uint8** DataPtr, uint16* LengthPtr, boolean* IsMgmtFrameOnlyPtr )	
<b>Service ID[hex]:</b>	0x23	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	CtrlIdx	Ethernet Controller index
	BufIdx	Ethernet Rx Buffer index
<b>Parameters (inout):</b>	DataPtr	IN: Pointer to the position of the EtherType of a common Ethernet frame  OUT: Pointer to the position of the EtherType in the management frame
	LengthPtr	IN: Pointer to the length of the frame received

		OUT: Pointer to the length decreased by the management information length.
<b>Parameters (out):</b>	IsMgmtFrameOnlyPtr	Information about the kind of frame  FALSE: Frame is not only for management purpose, but also for normal communication.  TRUE: Frame is only for management purpose and must not be processed in common receive process
<b>Return value:</b>	Std_ReturnType	E_OK: Frame successfully processed E_NOT_OK: Frame processing failed
<b>Description:</b>	Function inspects the Ethernet frame passed by the data pointer for management information and stores it for later use in EthSwt_EthRxFinishedIndication().	
<b>Available via:</b>	EthSwt_Eth.h	

] (SRS\_Eth\_00125)

**[SWS\_EthSwt\_00249]** The function EthSwt\_EthRxProcessFrame() shall be pre compile time configurable ON/OFF by the configuration parameter: EthSwtManagementSupportApi .] (SRS\_BSW\_00171)

### 8.3.28 EthSwt\_EthRxFinishedIndication

**[SWS\_EthSwt\_91005]** [

<b>Service name:</b>	EthSwt_EthRxFinishedIndication	
<b>Syntax:</b>	Std_ReturnType EthSwt_EthRxFinishedIndication( uint8 CtrlIdx, Eth_BufIdxType BufIdx )	
<b>Service ID[hex]:</b>	0x24	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	CtrlIdx	Ethernet Controller index
	BufIdx	Ethernet Rx Buffer index
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: Frame successfully processed E_NOT_OK: Frame processing failed
<b>Description:</b>	Indication for a finished receive process for a specific Ethernet frame, which results in providing the management information retrieved during EthSwt_EthRxProcessFrame().	
<b>Available via:</b>	EthSwt_Eth.h	

] (SRS\_Eth\_00125)

**[SWS\_EthSwt\_00253]** The function EthSwt\_EthRxFinishedIndication() shall be pre compile time configurable ON/OFF by the configuration parameter: EthSwtManagementSupportApi .] (SRS\_BSW\_00171)

### 8.3.29 EthSwt\_EthTxPrepareFrame

[SWS\_EthSwt\_91006] [

<b>Service name:</b>	EthSwt_EthTxPrepareFrame	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_EthTxPrepareFrame (     uint8 CtrlIdx,     Eth_BufIdxType BufIdx,     uint8** DataPtr,     uint16* LengthPtr )</pre>	
<b>Service ID[hex]:</b>	0x25	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	CtrlIdx	Ethernet Controller index
	BufIdx	Ethernet Rx Buffer index
<b>Parameters (inout):</b>	DataPtr	IN: Pointer to the position of the EtherType of a common Ethernet frame OUT: Pointer to the position of the EtherType in the management frame
	LengthPtr	IN: Pointer to the length of the buffer without management information OUT: Pointer to the modified length needed for buffer and management information
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: Frame successfully prepared E_NOT_OK: Frame preparation failed
<b>Description:</b>	Prepares the Ethernet frame for common Ethernet communication (frame shall be handled by switch according to the common address resolution behavior) and stores the information for processing of EthSwt_EthTxFinishedIndication().	
<b>Available via:</b>	EthSwt_Eth.h	

] (SRS\_Eth\_00125)

[SWS\_EthSwt\_00257] The function EthSwt\_EthTxPrepareFrame() shall be pre compile time configurable ON/OFF by the configuration parameter: EthSwtManagementSupportApi.] (SRS\_BSW\_00171)

### 8.3.30 EthSwt\_EthTxAdaptBufferLength

[SWS\_EthSwt\_91007] [

<b>Service name:</b>	EthSwt_EthTxAdaptBufferLength	
<b>Syntax:</b>	<pre>void EthSwt_EthTxAdaptBufferLength (     uint16* LengthPtr )</pre>	
<b>Service ID[hex]:</b>	0x26	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	None	
<b>Parameters (inout):</b>	LengthPtr	IN: Pointer to the length of the buffer without management information.



	OUT: Pointer to the modified length needed for buffer and management information.
<b>Parameters (out):</b>	None
<b>Return value:</b>	None
<b>Description:</b>	Modifies the buffer length to be able to insert management information.
<b>Available via:</b>	EthSwt_Eth.h

] (SRS\_Eth\_00125)

**[SWS\_EthSwt\_00261]** [ The function `EthSwt_EthTxAdaptBufferLength()` shall be pre compile time configurable ON/OFF by the configuration parameter: `EthSwtManagementSupportApi` .] ( SRS\_BSW\_00171)

### 8.3.31 EthSwt\_SetMgmtInfo

**[SWS\_EthSwt\_91008]** [

<b>Service name:</b>	EthSwt_SetMgmtInfo	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_SetMgmtInfo(     uint8 CtrlIdx,     Eth_BufIdxType BufIdx,     const EthSwt_MgmtInfoType* MgmtInfoPtr )</pre>	
<b>Service ID[hex]:</b>	0x27	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	CtrlIdx	Ethernet Controller index
	BufIdx	Ethernet Rx Buffer index
	MgmtInfoPtr	Pointer to the management information
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: Management infos successfully set E_NOT_OK: Setting of management infos failed
<b>Description:</b>	Extends the Ethernet frame prepared previously by <code>EthSwt_EthTxPrepareFrame()</code> with the management information to achieve transmission only on specific ports.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00125)

**[SWS\_EthSwt\_00264]** [ The function `EthSwt_SetMgmtInfo()` shall be pre compile time configurable ON/OFF by the configuration parameter: `EthSwtManagementSupportApi` .] ( SRS\_BSW\_00171)

### 8.3.32 EthSwt\_EthTxProcessFrame

**[SWS\_EthSwt\_91009]** [

<b>Service name:</b>	EthSwt_EthTxProcessFrame	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_EthTxProcessFrame(     uint8 CtrlIdx,</pre>	

	Eth_BufIdxType BufIdx, uint8** DataPtr, uint16* LengthPtr )	
<b>Service ID[hex]:</b>	0x28	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	CtrlIdx	Ethernet Controller index
	BufIdx	Ethernet Rx Buffer index
<b>Parameters (inout):</b>	DataPtr	IN: Pointer to the position of the EtherType of a common Ethernet frame  OUT: Pointer to the position of the EtherType in the management frame
	LengthPtr	IN: Pointer to the length of the received frame  OUT: Pointer to the length decreased by the management information length
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: Frame successfully processed E_NOT_OK: Frame processing failed
<b>Description:</b>	Function inserts management information into the Ethernet frame.	
<b>Available via:</b>	EthSwt Eth.h	

] (SRS\_Eth\_00125)

**[SWS\_EthSwt\_00268]** The function `EthSwt_EthTxProcessFrame()` shall be pre compile time configurable ON/OFF by the configuration parameter: `EthSwtManagementSupportApi` .

] (SRS\_BSW\_00171)

### 8.3.33 EthSwt\_EthTxFinishedIndication

**[SWS\_EthSwt\_91010]** [

<b>Service name:</b>	EthSwt_EthTxFinishedIndication	
<b>Syntax:</b>	Std_ReturnType EthSwt_EthTxFinishedIndication( uint8 CtrlIdx, Eth_BufIdxType BufIdx )	
<b>Service ID[hex]:</b>	0x29	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	CtrlIdx	Ethernet Controller index
	BufIdx	Ethernet Rx Buffer index
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: Frame successfully processed E_NOT_OK: Frame processing failed
<b>Description:</b>	Indication for a finished transmit process for a specific Ethernet frame.	
<b>Available via:</b>	EthSwt Eth.h	

] (SRS\_Eth\_00125)

**[SWS\_EthSwt\_00273]** The function `EthSwt_EthTxFinishedIndication()` shall be pre compile time configurable ON/OFF by the configuration parameter: `EthSwtManagementSupportApi .]` ( SRS\_BSW\_00171)

### 8.3.34 EthSwt\_PortEnableTimeStamp

**[SWS\_EthSwt\_91028]** [

<b>Service name:</b>	EthSwt_PortEnableTimeStamp	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_PortEnableTimeStamp(     uint8 CtrlIdx,     Eth_BufIdxType BufIdx,     EthSwt_MgmtInfoType* MgmtInfoPtr )</pre>	
<b>Service ID[hex]:</b>	0x40	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	CtrlIdx	Ethernet Controller index
	BufIdx	Ethernet Rx Buffer index
	MgmtInfoPtr	Management information including SwitchIdx and SwitchPortIdx
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: Time stamping on egress successfully enabled E_NOT_OK: Enabling of time stamping on egress has been failed
<b>Description:</b>	<p>Activates egress time stamping on a dedicated message object on a dedicated port of a Switch if <code>EthSwtPortTimeStampSupport</code> 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.</p>	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00125)

**[SWS\_EthSwt\_00379]** The function `EthSwt_PortEnableTimeStamp()` shall be pre compile time configurable ON/OFF by the configuration parameter: `EthSwtGlobalTimeSupportApi .]` ( SRS\_BSW\_00171)

### 8.3.35 EthSwt\_VerifyConfig

**[SWS\_EthSwt\_91012]** [

<b>Service name:</b>	EthSwt_VerifyConfig	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_VerifyConfig(     uint8 SwitchIdx,     boolean* Result )</pre>	
<b>Service ID[hex]:</b>	0x31	
<b>Sync/Async:</b>	Synchronous	

<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	Result	Result of verification, TRUE: configuration verified ok, FALSE: configuration values found corrupted
<b>Return value:</b>	Std_ReturnType	E_OK: Configuration verification succeeded, E_NOT_OK: Configuration verification not succeeded.
<b>Description:</b>	Verifies the Switch Configuration depending on the HW-Architecture, HW-capability and the intended accuracy of this verification.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_ETH\_00126)

**[SWS\_EthSwt\_00287]** The function EthSwt\_VerifyConfig shall be compile time configurable On/Off by the configuration parameter: EthSwtVerifyConfigApi. ] (SRS\_BSW\_00171)

### 8.3.36 EthSwt\_SetForwardingMode

**[SWS\_EthSwt\_91013]** [

<b>Service name:</b>	EthSwt_SetForwardingMode	
<b>Syntax:</b>	Std_ReturnType EthSwt_SetForwardingMode( uint8 SwitchIdx, boolean mode )	
<b>Service ID[hex]:</b>	0x32	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	mode	True Forwarding enabled, False Forwarding disabled
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: stopping of frame forwarding succeeded, E_NOT_OK: stopping of frame forwarding not succeeded.
<b>Description:</b>	Configures switch to start or stop forwarding for all ports. This API call may be used during switch configuration verification.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_ETH\_00126)

**[SWS\_EthSwt\_00291]** The function EthSwt\_SetForwardingMode shall be compile time configurable On/Off by the configuration parameter: EthSwtSetForwardingModeApi. ] (SRS\_BSW\_00171)

### 8.3.37 EthSwt\_GetPortSignalQuality

**[SWS\_EthSwt\_91014]** [

<b>Service name:</b>	EthSwt_GetPortSignalQuality	
<b>Syntax:</b>	Std_ReturnType EthSwt_GetPortSignalQuality( uint8 SwitchIdx, uint8 PortIdx,	

	uint32* SignalQualityPtr )	
<b>Service ID[hex]:</b>	0x33	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	SignalQualityPtr	Pointer to the memory where the signal quality shall be stored.
<b>Return value:</b>	Std_ReturnType	E_OK: signal quality could be read. E_NOT_OK: signal quality could not be read (i.e. no Ethernet transceiver is available for this Ethernet switch port)
<b>Description:</b>	The function retrieves the signal quality of the link of the indexed Ethernet switch port. If no transceiver is referenced the signal quality shall be set to 0xFFFFFFFF.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00123)

**[SWS\_EthSwt\_00293]** The function EthSwt\_GetPortSignalQuality() shall obtain the signal quality by calling the function EthTrcv\_GetPhySignalQuality() of the referenced Ethernet Transceiver Driver. If the current signal quality is not available, the signal quality shall be set to 0xFFFFFFFF. ] ( SRS\_Eth\_00123)

**[SWS\_EthSwt\_00297]** The function EthSwt\_GetPortSignalQuality shall be pre compile time configurable On/Off by the configuration parameter: EthSwt\_GetPortSignalQualityApi.] ( SRS\_BSW\_00171)

### 8.3.38 EthSwt\_GetPortIdentifier

**[SWS\_EthSwt\_91015]** [

<b>Service name:</b>	EthSwt_GetPortIdentifier	
<b>Syntax:</b>	Std_ReturnType EthSwt_GetPortIdentifier( uint8 SwitchIdx, uint8 PortIdx, uint32* OrgUniqueIdPtr, uint8* ModelNrPtr, uint8* RevisionNrPtr )	
<b>Service ID[hex]:</b>	0x34	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	OrgUniqueIdPtr	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.
	RevisionNrPtr	Pointer to the memory where the Revision Number shall be stored.
<b>Return value:</b>	Std_ReturnType	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).
<b>Description:</b>	This function retrieves the OUI (24 bit) of the indexed Ethernet switch port.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00123)

**[SWS\_EthSwt\_00299]** The function EthSwt\_GetPortIdentifier() shall return the value of the organizationally unique identifier (OUI 24 bit) of the indexed Ethernet switch port that is connected to the indexed Ethernet switch. It shall set the 8 most significant bits of the OUI to 0xFFxxxxxx. If the Ethernet switch port references an Ethernet transceiver, the function shall obtain the OUI by calling the function EthTrcv\_GetPhyIdentifier() and set the 8 most significant bits of the OUI to 0x00xxxxxx. ] ( SRS\_Eth\_00123)

**[SWS\_EthSwt\_00394]** If neither the Ethernet switch port nor the Ethernet Transceiver Driver can provide an OUI the function EthSwt\_GetPortIdentifier() shall return E\_NOT\_OK.] ( )

**[SWS\_EthSwt\_00303]** The function EthSwt\_GetPortIdentifier shall be pre compile time configurable On/Off by the configuration parameter: EthSwt\_GetPortIdentifierApi.] ( SRS\_BSW\_00171)

### 8.3.39 EthSwt\_GetSwitchIdentifier

**[SWS\_EthSwt\_91016]** [

<b>Service name:</b>	EthSwt_GetSwitchIdentifier	
<b>Syntax:</b>	Std_ReturnType EthSwt_GetSwitchIdentifier( uint8 SwitchIdx, uint32* OrgUniqueIdPtr )	
<b>Service ID[hex]:</b>	0x35	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	OrgUniqueIdPtr	Pointer to the memory where the Organizationally Unique Identifier shall be stored.
<b>Return value:</b>	Std_ReturnType	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)
<b>Description:</b>	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.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00123)

**[SWS\_EthSwt\_00305]** The function EthSwt\_GetSwitchIdentifier shall return the value of the organizationally unique identifier of the indexed Ethernet switch.] ( SRS\_Eth\_00123)

**[SWS\_EthSwt\_00308]** The function EthSwt\_GetSwitchIdentifier shall be pre compile time configurable On/Off by the configuration parameter: EthSwt\_GetSwitchIdentifierApi.] ( SRS\_BSW\_00171)

### 8.3.40 EthSwt\_WritePortMirrorConfiguration

**[SWS\_EthSwt\_91018]** [

<b>Service name:</b>	EthSwt_WritePortMirrorConfiguration	
<b>Syntax:</b>	Std_ReturnType EthSwt_WritePortMirrorConfiguration( uint8 MirroredSwitchIdx, const EthSwt_PortMirrorCfgType* PortMirrorConfigurationPtr )	
<b>Service ID[hex]:</b>	0x36	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	MirroredSwitchIdx	Index of the switch within the context of the Ethernet Switch Driver, where the Ethernet switch port is located, that has to be mirrored
	PortMirrorConfigurationPtr	Pointer of the port configuration, which shall be stored in a shadow buffer in the Ethernet switch driver
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	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
<b>Description:</b>	Store the given port mirror configuration in a shadow buffer in the Ethernet switch driver for the given MirroredSwitchIdx.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00123)

**[SWS\_EthSwt\_00309]**

The function EthSwt\_WritePortMirrorConfiguration shall store the port mirror configuration of the given MirroredSwitchIdx in a shadow buffer. The MirroredSwitchIdx shall be used to identify the port mirror configuration within the Ethernet switch driver.] ( SRS\_Eth\_00123)

**[SWS\_EthSwt\_00312]** The function EthSwt\_WritePortMirrorConfiguration shall be pre compile time configurable On/Off by the configuration parameter: EthSwt\_WritePortMirrorConfigurationApi.] ( SRS\_BSW\_00171)

**[SWS\_EthSwt\_00424]**



The function shall return with ETHSWT\_PORT\_MIRRORING\_CONFIGURATION\_NOT\_SUPPORTED, if the port mirroring configuration is not supported by the Ethernet switch driver or by the Ethernet switch hardware, e.g.:

- the configured mirrored traffic direction (see [SWS\_EthSwt\_91017] "TrafficDirectionIngressBitMask" and "TrafficDirectionEgressBitMask") for ingress and egress traffic of the same port is not supported
- mirrored ports and capture ports, respectively, are not available within the Ethernet switch driver

] (SRS\_Eth\_00123)

### 8.3.41 EthSwt\_ReadPortMirrorConfiguration

[SWS\_EthSwt\_91019] [

<b>Service name:</b>	EthSwt_ReadPortMirrorConfiguration	
<b>Syntax:</b>	Std_ReturnType EthSwt_ReadPortMirrorConfiguration( uint8 MirroredSwitchIdx, EthSwt_PortMirrorCfgType* PortMirrorConfigurationPtr )	
<b>Service ID[hex]:</b>	0x37	
<b>Sync/Async:</b>	Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	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
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	PortMirrorConfigurationPtr	Pointer to the memory where the port configuration shall be stored.
<b>Return value:</b>	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)
<b>Description:</b>	Obtain the port mirror configuration of the given Ethernet switch.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00123)

[SWS\_EthSwt\_00313]

The function EthSwt\_ReadPortMirrorConfiguration shall return the port mirror configuration identified by the given MirroredSwitchIdx. If no port mirror configuration is found for the MirroredSwitchIdx, the function shall return E\_NOT\_OK.

] (SRS\_Eth\_00123)

[SWS\_EthSwt\_00317] [ The function EthSwt\_ReadPortMirrorConfiguration shall be pre compile time configurable On/Off by the configuration parameter: EthSwt\_ReadPortMirrorConfigurationApi.] (SRS\_BSW\_00171)

### 8.3.42 EthSwt\_DeletePortMirrorConfiguration

**[SWS\_EthSwt\_91034]** [

<b>Service name:</b>	EthSwt_DeletePortMirrorConfiguration	
<b>Syntax:</b>	Std_ReturnType EthSwt_DeletePortMirrorConfiguration( uint8 MirroredSwitchIdx )	
<b>Service ID[hex]:</b>	0	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant for different MirroredSwitchIdx. Non reentrant for the same SwitchIdx.	
<b>Parameters (in):</b>	MirroredSwitchIdx	Index of the switch within the context of the Ethernet Switch Driver.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	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)
<b>Description:</b>	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.	
<b>Available via:</b>	EthSwt.h	

] ()

**[SWS\_EthSwt\_00425]**

The function EthSwt\_DeletePortMirrorConfiguration shall mark the stored port mirror configuration in the shadow buffer of the given MirroredSwitchIdx as "to be deleted".

] (SRS\_Eth\_00123)

**[SWS\_EthSwt\_00426]**

If a port mirroring for the given MirroredSwitchIdx is enabled, the request to delete the configuration shall be rejected by returning E\_NOT\_OK. Only those port configurations are allowed to be deleted, where the port mirroring of the given MirroredSwitchIdx is disabled.

] (SRS\_Eth\_00123)

**[SWS\_EthSwt\_00427]**

The function EthSwt\_DeletePortMirrorConfiguration shall be pre compile time configurable On/Off by the configuration parameter: EthSwtDeletePortMirrorConfigurationApi.] ( SRS\_BSW\_00171)

### 8.3.43 EthSwt\_GetPortMirrorState

**[SWS\_EthSwt\_91021]** [

<b>Service name:</b>	EthSwt_GetPortMirrorState	
<b>Syntax:</b>	Std_ReturnType EthSwt_GetPortMirrorState( uint8 SwitchIdx, uint8 PortIdx, EthSwt_PortMirrorStateType* PortMirrorStatePtr )	
<b>Service ID[hex]:</b>	0x38	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	

<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	PortMirrorStatePtr	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.
<b>Return value:</b>	Std_ReturnType	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)
<b>Description:</b>	Obtain the current status of the port mirroring for the indexed Ethernet switch port	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00123)

[SWS\_EthSwt\_00318] The function EthSwt\_GetPortMirrorState shall return the port mirroring state of the indexed ethernet switch port.] ( SRS\_Eth\_00123)

[SWS\_EthSwt\_00322] The function EthSwt\_GetPortMirrorState shall be pre compile time configurable On/Off by the configuration parameter: EthSwt\_GetPortMirrorStateApi.] ( SRS\_BSW\_00171)

### 8.3.44 EthSwt\_SetPortMirrorState

[SWS\_EthSwt\_91022] [

<b>Service name:</b>	EthSwt_SetPortMirrorState	
<b>Syntax:</b>	Std_ReturnType EthSwt_SetPortMirrorState ( uint8 MirroredSwitchIdx, EthSwt_PortMirrorStateType PortMirrorState )	
<b>Service ID[hex]:</b>	0x39	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	MirroredSwitchIdx	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, respectively.
	PortMirrorState	Contain the requested port mirroring state either PORT_MIRRORING_ENABLED or PORT_MIRRORING_DISABLED
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	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 mirror configuration is available)
<b>Description:</b>	Request to set the given port mirroring state of the port mirror configuration for the given Ethernet switch.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00123)

**[SWS\_EthSwT\_00323]** The function EthSwT\_SetPortMirrorState shall request the given port mirroring state for the port mirroring configuration of the indexed Ethernet switch, and store the requested port mirror state in a shadow buffer. ] ( SRS\_Eth\_00123)

**[SWS\_EthSwT\_00327]** The function EthSwT\_SetPortMirrorState shall be pre compile time configurable On/Off by the configuration parameter: EthSwT\_SetPortMirrorStateApi. ] ( SRS\_BSW\_00171)

### 8.3.45 EthSwT\_SetPortTestMode

**[SWS\_EthSwT\_91029]** [

<b>Service name:</b>	EthSwT_SetPortTestMode	
<b>Syntax:</b>	Std_ReturnType EthSwT_SetPortTestMode( uint8 SwitchIdx, uint8 PortIdx, EthTrcv_PhyTestModeType Mode )	
<b>Service ID[hex]:</b>	0x3a	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	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
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	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)
<b>Description:</b>	Activates a given test mode of the indexed Ethernet switch port.	
<b>Available via:</b>	EthSwT.h	

] (SRS\_Eth\_00123)

**[SWS\_EthSwT\_00328]** The function EthSwT\_SetPortTestMode shall forward the call with the given test mode by calling the function EthTrcv\_SetPhyTestMode() of the referenced Ethernet Transceiver Driver. ] ( SRS\_Eth\_00123)

**[SWS\_EthSwT\_00332]** The function EthSwT\_SetPortTestMode shall be pre compile time configurable On/Off by the configuration parameter: EthSwT\_SetPortTestModeApi. ] ( SRS\_BSW\_00171)

### 8.3.46 EthSwT\_SetPortLoopbackMode

**[SWS\_EthSwT\_91023]** [

<b>Service name:</b>	EthSwT_SetPortLoopbackMode
----------------------	----------------------------

<b>Syntax:</b>	Std_ReturnType EthSwt_SetPortLoopbackMode ( uint8 SwitchIdx, uint8 PortIdx, EthTrcv_PhyLoopbackModeType Mode )	
<b>Service ID[hex]:</b>	0x3b	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	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
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	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)
<b>Description:</b>	Activates a given test loop-back mode of the indexed Ethernet switch port.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00123)

**[SWS\_EthSwt\_00334]** [ The function EthSwt\_SetPortLoopbackMode() shall forward the call with the given loop-back mode by calling the function EthTrcv\_SetPhyLoopbackMode() of the referenced Ethernet Transceiver Driver.] ( SRS\_Eth\_00123)

**[SWS\_EthSwt\_00338]** [ The function EthSwt\_SetPortLoopbackMode shall be pre compile time configurable On/Off by the configuration parameter: EthSwt\_SetPortLoopbackModeApi.] ( SRS\_BSW\_00171)

### 8.3.47 EthSwt\_SetPortTxMode

**[SWS\_EthSwt\_91024]** [

<b>Service name:</b>	EthSwt_SetPortTxMode	
<b>Syntax:</b>	Std_ReturnType EthSwt_SetPortTxMode ( uint8 SwitchIdx, uint8 PortIdx, EthTrcv_PhyTxModeType Mode )	
<b>Service ID[hex]:</b>	0x3c	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	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
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	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)
<b>Description:</b>	Activates a given transmission mode of the indexed Ethernet switch port.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00123)

**[SWS\_EthSwt\_00340]** [ The function EthSwt\_SetPortTxMode() shall forward the call with the given transmission mode by calling the function EthTrcv\_SetPhyTxMode() of the referenced Ethernet Transceiver Driver. ] ( SRS\_Eth\_00123)

**[SWS\_EthSwt\_00344]** [ The function EthSwt\_SetPortTxMode shall be pre compile time configurable On/Off by the configuration parameter: EthSwt\_SetPortTxModeApi. ] ( SRS\_BSW\_00171)

### 8.3.48 EthSwt\_RunPortCableDiagnostic

**[SWS\_EthSwt\_91011]** [

<b>Service name:</b>	EthSwt_RunPortCableDiagnostic	
<b>Syntax:</b>	Std_ReturnType EthSwt_RunPortCableDiagnostic( uint8 SwitchIdxIdx, uint8 PortIdx )	
<b>Service ID[hex]:</b>	0x45	
<b>Sync/Async:</b>	Asynchronous	
<b>Reentrancy:</b>	Reentrant for different SwitchIdx and PortIdx. Non reentrant for the same SwitchIdx and PortIdx.	
<b>Parameters (in):</b>	SwitchIdxIdx	Index of the switch within the context of the Ethernet Switch Driver.
	PortIdx	Index of the port at the addressed switch.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	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
<b>Description:</b>	Trigger the cable diagnostics of the given Ethernet Switch port (PortIdx) by calling EthTrcv_RunCableDiagnostic of the referenced Ethernet transceiver.	
<b>Available via:</b>	EthSwt.h	

] ()

**[SWS\_EthSwt\_00429]** [ The function EthSwt\_RunPortCableDiagnostic() shall forward the call by calling EthTrcv\_RunCableDiagnostic() of the referenced Ethernet Transceiver Driver.] ()

### 8.3.49 EthSwt\_GetPortCableDiagnosticsResult

#### [SWS\_EthSwt\_91025] [

<b>Service name:</b>	EthSwt_GetPortCableDiagnosticsResult	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_GetPortCableDiagnosticsResult(     uint8 SwitchIdx,     uint8 PortIdx,     EthTrcv_CableDiagResultType* ResultPtr )</pre>	
<b>Service ID[hex]:</b>	0x3f	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	ResultPtr	Pointer to the location where the cable diagnostics result shall be stored
	Std_ReturnType	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)
<b>Description:</b>	Retrieves the cable diagnostics result of the indexed Ethernet switch port respectively the referenced Ethernet Transceiver Driver.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00123)

[SWS\_EthSwt\_00346] [ The function EthSwt\_GetPortCableDiagnosticsResult() shall obtain the cable diagnostics result by calling the function EthTrcv\_GetCableDiagnosticsResult() of the referenced Ethernet Transceiver Driver. ] ( SRS\_Eth\_00123)

[SWS\_EthSwt\_00350] [ The function EthSwt\_GetPortCableDiagnosticsResult shall be pre compile time configurable On/Off by the configuration parameter: EthSwt\_GetPortCableDiagnosticsResultApi.] ( SRS\_BSW\_00171)

### 8.3.50 EthSwt\_GetCfgDataRow

#### [SWS\_EthSwt\_91030] [

<b>Service name:</b>	EthSwt_GetCfgDataRow	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_GetCfgDataRow(     uint8 SwitchIdx,     uint32 Offset,     uint16 Length,     uint8* BufferPtr )</pre>	
<b>Service ID[hex]:</b>	0x41	
<b>Sync/Async:</b>	Asynchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	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
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	BufferPtr	Pointer to the location where the data shall be copied
<b>Return value:</b>	Std_ReturnType	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)
<b>Description:</b>	Retrieves the data in memory of the indexed Ethernet switch in variable length	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00123)

**[SWS\_EthSwt\_00403]** [ The function EthSwt\_GetCfgDataRaw() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.] ( SRS\_BSW\_00171)

**[SWS\_EthSwt\_00404]** [ When calling the function EthSwt\_GetCfgDataRaw(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT\_E\_ACCESS and return E\_NOT\_OK, otherwise pass the extended production error ETHSWT\_E\_ACCESS and return E\_OK.] ( )

### 8.3.51 EthSwt\_GetCfgDataInfo

**[SWS\_EthSwt\_91031]** [

<b>Service name:</b>	EthSwt_GetCfgDataInfo	
<b>Syntax:</b>	Std_ReturnType EthSwt_GetCfgDataInfo( uint8 SwitchIdx, uint32* DataSizePtr, uint32* DataAddressPtr )	
<b>Service ID[hex]:</b>	0x42	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the Ethernet switch within the context of the Ethernet Switch Driver
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	DataSizePtr	Pointer to the location where the total size of the configuration data shall be copied
	DataAddressPtr	Pointer to the location where the start address of the configuration registers shall be copied
<b>Return value:</b>	Std_ReturnType	E_OK: the data was obtained successfully E_NOT_OK: the data was not obtained successfully. (i.e. indexed Ethernet switch is not available)
<b>Description:</b>	Retrieves the total size of data and the memory start address of the indexed Ethernet Switch.	
<b>Available via:</b>	EthSwt.h	

] (SRS\_Eth\_00123)

[SWS\_EthSwt\_00405] [ The function EthSwt\_GetCfgDataInfo() shall only be available if parameter EthSwtGetCfgRaw is set to TRUE.] ( SRS\_BSW\_00171)

[SWS\_EthSwt\_00406] [ When calling the function EthSwt\_GetCfgDataInfo(), the function shall check the access to the Ethernet switch driver. If the check fails, the function shall raise the extended production error ETHSWT\_E\_ACCESS and return E\_NOT\_OK, otherwise pass the extended production error ETHSWT\_E\_ACCESS and return E\_OK.] ()

### 8.3.52 EthSwt\_PortLinkStateRequest

[SWS\_EthSwt\_91123] [

<b>Service name:</b>	EthSwt_PortLinkStateRequest	
<b>Syntax:</b>	Std_ReturnType EthSwt_PortLinkStateRequest ( uint8 SwitchIdx, uint8 PortIdx, EthTrcv_LinkStateType PortLinkState )	
<b>Service ID[hex]:</b>	--	
<b>Sync/Async:</b>	Asynchronous	
<b>Reentrancy:</b>	Reentrant for different SwitchIdx and PortIdx. Non reentrant for the same SwitchIdx and PortIdx.	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver.
	PortIdx	Index of the port at the addressed switch.
	PortLinkState	The Ethernet link state of a physical Ethernet connection.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: Request has been accepted and if the function call is in state ETHSWT_STATE_PORTINIT_COMPLETED or ETHSWT_STATE_ACTIVE E_NOT_OK: Request has not been accepted. (e.g. the indexed Ethernet switch port does not reference an EthTrcv)
<b>Description:</b>	Request a link state by calling EthTrcv_TransceiverLinkStateRequest with the TrcvIdx of the Ethernet transceiver which is referenced by the Ethernet Switch port (PortIdx).	
<b>Available via:</b>	EthSwt.h	

] ()

[SWS\_EthSwt\_00415] [ The function EthSwt\_PortLinkStateRequest shall request the given link state for the indexed Ethernet switch port of the switch by calling the EthTrcv\_TransceiverLinkStateRequest with the given EthTrcv\_LinkStateType. If the EthSwtPort does not reference an EthTrcv, then the function shall return E\_NOT\_OK.] ()

### 8.3.53 EthSwt\_GetMaxFIFOBufferFillLevel

[SWS\_EthSwt\_91050] [

<b>Service name:</b>	EthSwt_GetMaxFIFOBufferFillLevel
<b>Syntax:</b>	Std_ReturnType EthSwt_GetMaxFIFOBufferFillLevel (

	<pre>uint8 SwitchIdx, uint8 SwitchPortIdx, uint8 SwitchPortEgressFifoIdx, uint32* SwitchPortEgressFifoBufferLevelPtr ) </pre>	
<b>Service ID[hex]:</b>	--	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant for different SwitchIdx and PortIdx. Non reentrant for the same SwitchIdx and PortIdx.	
<b>Parameters (in):</b>	SwitchIdx	Index of the Ethernet switch within the context of the Ethernet Switch Driver.
	SwitchPortIdx	Index of the Ethernet switch egress port at the addressed Ethernet switch.
	SwitchPortEgressFifoIdx	Index of the egress FIFO of the addressed Ethernet switch port
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	SwitchPortEgressFifoBufferLevelPtr	Pointer to a memory location, where the maximum amount of allocated FIFO buffer (in bytes) since the last read out shall be stored
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: The maximal FIFO buffer level could not be obtained
<b>Description:</b>	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 SwitchPortEgressFifoBufferLevelPtr shall be set to 0xFFFFFFFF. This API may be called by e.g. a CDD.	
<b>Available via:</b>	EthSwt.h	

] ()

**[SWS\_EthSwt\_00430]** The function EthSwt\_GetMaxFIFOBufferFillLevel shall read out the maximum amount of allocated FIFO buffer since the last read out.] (SRS\_ETH\_00119)

**[SWS\_EthSwt\_00431]** When the maximum amount of allocated FIFO buffer is read out, the value shall be reset to 0x00000000 explicitly, if it is not done by the hardware.] (SRS\_ETH\_00119)

**[SWS\_EthSwt\_00432]** The function EthSwt\_GetMaxFIFOBufferFillLevel shall be pre compile time configurable On/Off by the configuration parameter: EthSwtGetMaxFIFOBufferFillLevelApi.] (SRS\_BSW\_00171)

### 8.3.54 EthSwt\_GetRxMgmtObject

**[SWS\_EthSwt\_91038]** [

<b>Service name:</b>	EthSwt_GetRxMgmtObject
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_GetRxMgmtObject( uint8 CtrlIdx, Eth_DataType* DataPtr, EthSwt MgmtObjectType** MgmtObjectPtr ) </pre>

	)	
<b>Service ID[hex]:</b>	0x43	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	CtrlIdx	Index of an Ethernet Interface controller
	DataPtr	Ethernet data pointer
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	MgmtObjectPtr	Pointer to the management object.
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: management object could not be obtained
<b>Description:</b>	Obtains the MgmtObject of the (in this context) unique DataPtr.	
<b>Available via:</b>	EthSwt.h	

| ()

### 8.3.55 EthSwt\_GetTxMgmtObject

[SWS\_EthSwt\_91039] [

<b>Service name:</b>	EthSwt_GetTxMgmtObject	
<b>Syntax:</b>	<pre>Std_ReturnType EthSwt_GetTxMgmtObject(     uint8 CtrlIdx,     Eth_BufIdxType BufIdx,     EthSwt_MgmtObjectType** MgmtObjectPtr )</pre>	
<b>Service ID[hex]:</b>	0x44	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	CtrlIdx	Index of an Ethernet Interface controller
	BufIdx	Ethernet Rx Buffer index
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	MgmtObjectPtr	Pointer to the management object.
<b>Return value:</b>	Std_ReturnType	E_OK: success E_NOT_OK: management object could not be obtained
<b>Description:</b>	Obtains the MgmtObject of the (in this context) unique BufIdx.	
<b>Available via:</b>	EthSwt.h	

| ()

## 8.4 Call-back notifications

### 8.4.1 <user>\_PersistentConfigurationResult

[SWS\_EthSwt\_00193] [

<b>Service name:</b>	<User>_PersistentConfigurationResult	
<b>Syntax:</b>	<pre>void &lt;User&gt;_PersistentConfigurationResult(     NvM_RequestResultType JobResult</pre>	

	)
<b>Service ID[hex]:</b>	0x1b
<b>Sync/Async:</b>	Synchronous /Asynchronous
<b>Reentrancy:</b>	Non Reentrant
<b>Parameters (in):</b>	JobResult  Covers the job result of the previous processed single block job.
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	None
<b>Description:</b>	Shall be called by the EthSwt_NvmSingleBlockCallback
<b>Available via:</b>	<none>

] (SRS\_ETH\_00122, SRS\_ETH\_00087)

#### [SWS\_EthSwt\_00194] [

The function <User>\_PersistentConfigurationResult shall be called by the NvmSingleBlockCallback to inform the caller of EthSwt\_StoreConfiguration or EthSwt\_ResetConfiguration about the state of the past calls.] (SRS\_ETH\_00122, SRS\_ETH\_00087)

#### [SWS\_EthSwt\_00195] [

The function <User>\_PersistentConfigurationResult shall be pre compile time configurable by the <user> with content of EthSwtPersistentConfigurationResult. ] (SRS\_BSW\_00171)

## 8.5 Scheduled functions

#### [SWS\_EthSwt\_00114] [

<b>Service name:</b>	EthSwt_MainFunction
<b>Syntax:</b>	void EthSwt_MainFunction( void )
<b>Service ID[hex]:</b>	0x1c
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Non Reentrant
<b>Parameters (in):</b>	None
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	None
<b>Description:</b>	Service to support asynchronous behavior of API calls
<b>Available via:</b>	EthSwt_SchM.h

] (SRS\_BSW\_00433)

#### [SWS\_EthSwt\_00115] [

The EthSwt\_MainFunction support asynchronous behavior of API calls. This function is directly called by Basic Software Scheduler. ] (SRS\_BSW\_00433)

## 8.6 Expected Interfaces

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

### 8.6.1 Mandatory Interfaces

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

No mandatory Interfaces defined.

### 8.6.2 Optional Interfaces

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

#### [SWS\_EthSwT\_00098] [

API function	Header File	Description
Dem_SetEventStatus	Dem.h	Called by SW-Cs or BSW modules to report monitor status information to the Dem. BSW modules calling Dem_SetEventStatus can safely ignore the return value.
Det_ReportError	Det.h	Service to report development errors.
Eth_ReadMii	Eth.h	Reads a transceiver register
Eth_WriteMii	Eth.h	Configures a transceiver register or triggers a function offered by the receiver
EthTrcv_GetBaudRate	EthTrcv.h	Obtains the baud rate of the indexed transceiver
EthTrcv_GetDuplexMode	EthTrcv.h	Obtains the duplex mode of the indexed transceiver
EthTrcv_GetLinkState	EthTrcv.h	Obtains the link state of the indexed transceiver
EthTrcv_GetTransceiverMode	EthTrcv.h	Obtains the state of the indexed transceiver
EthTrcv_SetTransceiverMode	EthTrcv.h	Enables / disables the indexed transceiver
EthTrcv_StartAutoNegotiation	EthTrcv.h	Restarts the negotiation of the transmission parameters used by the indexed transceiver
NvM_GetErrorStatus	NvM.h	Service to read the block dependent error/status information.
NvM_ReadBlock	NvM.h	Service to copy the data of the NV block to its corresponding RAM block.
NvM_WriteBlock	NvM.h	Service to copy the data of the RAM block to its corresponding NV block.
Spi_AsyncTransmit	Spi.h	Service to transmit data on the SPI bus.
Spi_Cancel	Spi.h	Service cancels the specified on-going sequence transmission.
Spi_ReadIB	Spi.h	Service for reading synchronously one or more data from an IB SPI Handler/Driver Channel specified by parameter.
Spi_SetAsyncMode	Spi.h	Service to set the asynchronous mechanism mode for SPI busses handled asynchronously.
Spi_SetupEB	Spi.h	Service to setup the buffers and the length of data for the EB SPI Handler/Driver Channel specified.
Spi_SyncTransmit	Spi.h	Service to transmit data on the SPI bus
Spi_WriteIB	Spi.h	Service for writing one or more data to an IB SPI Handler/Driver Channel specified by parameter.

] (SRS\_Eth\_00122, SRS\_ETH\_00118, SRS\_ETH\_00119, SRS\_ETH\_00120, SRS\_ETH\_00087, SRS\_ETH\_00125, SRS\_BSW\_00375)

**[SWS\_EthSwt\_00192]**

[ The NvM APIs will only be used if the respective block is not configured for NvM\_ReadAll() and NvM\_WriteAll().  
] (SRS\_Eth\_00122)

**8.6.3 Configurable interfaces**

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

**8.6.3.1 <EthSwtLinkDownCallout>**

**[SWS\_EthSwt\_00117]** [

<b>Service name:</b>	<EthSwtLinkDownCallout>	
<b>Syntax:</b>	void <EthSwtLinkDownCallout>( uint8 SwitchIdx, uint8 PortIdx )	
<b>Service ID[hex]:</b>	0x19	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdx	Index of the port at the addressed switch
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	None	
<b>Description:</b>	Is called, if a link which is configured goes down.	
<b>Available via:</b>	EthSwt_Externals.h	

] (SRS\_ETH\_00119, SRS\_ETH\_00087)

**[SWS\_EthSwt\_00118]** [

The function <EthSwtLinkDownCallout> shall be called if a link, which is configured, goes down (link loss). The function provides the Switch index and the Port index, such that the port which went down can be identified.] (SRS\_ETH\_00119, SRS\_ETH\_00087)

**8.6.3.2 <EthSwtLinkUpCallout>**

**[SWS\_EthSwt\_00203]** [

<b>Service name:</b>	<EthSwtLinkUpCallout>	
<b>Syntax:</b>	void <EthSwtLinkUpCallout>( uint8 SwitchIdx, uint8 PortIdx )	
<b>Service ID[hex]:</b>	0x1a	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the switch within the context of the Ethernet Switch Driver
	PortIdx	Index of the port at the addressed switch
<b>Parameters</b>	None	



<b>(inout):</b>	
<b>Parameters (out):</b>	None
<b>Return value:</b>	None
<b>Description:</b>	Is called, if a link which is configured goes up
<b>Available via:</b>	EthSwt_Externals.h

] (SRS\_ETH\_00119, SRS\_ETH\_00087)

#### [SWS\_EthSwt\_00204] [

The function <EthSwtLinkUpCallout> shall be called if a link, which is configured, goes up. The function provides the Switch index and the Port index, such that the port which went up can be identified.] (SRS\_ETH\_00119, SRS\_ETH\_00087)

Note: If the hardware cannot signal a link up with an interrupt, the status of the link has to be determined in polling mode by checking the state of the link.

### 8.6.3.3 <GetCfgDataRawDone>

#### [SWS\_EthSwt\_91032] [

<b>Service name:</b>	<GetCfgDataRawDone>	
<b>Syntax:</b>	void <GetCfgDataRawDone>( uint8 SwitchIdx )	
<b>Service ID[hex]:</b>	0x43	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	SwitchIdx	Index of the Ethernet switch where the Configuration is read.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	None	
<b>Description:</b>	The call of the function EthSwt_GetCfgDataRaw() triggers a asynchrony read of a certain memory section of the Ethernet switch driver. If the read is done, the configured callout function <GetCfgDataRawDone> shall be called]	
<b>Available via:</b>	EthSwt_Externals.h	

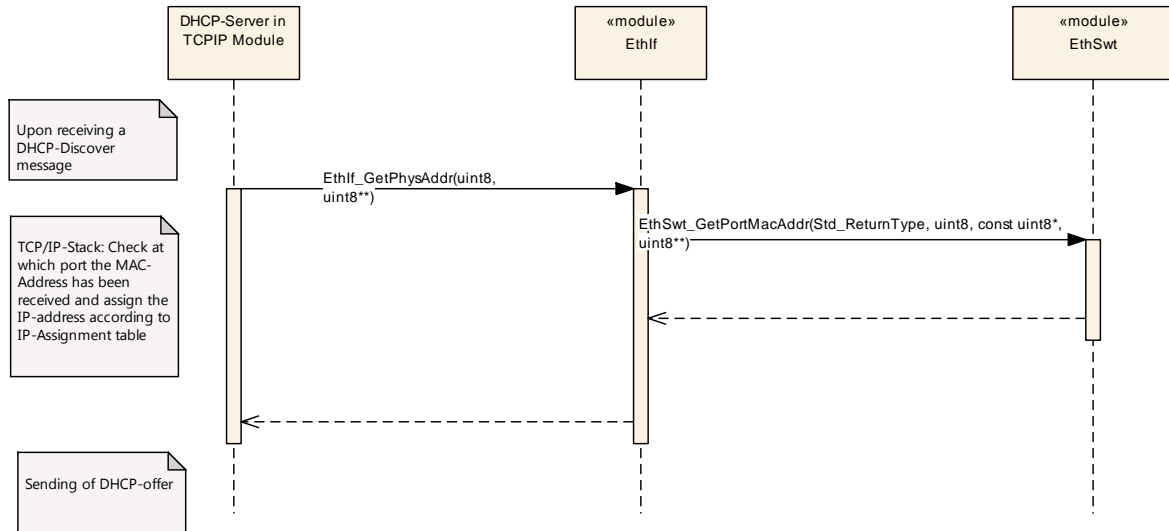
] (SRS\_Eth\_00123)

## 8.7 Service Interfaces

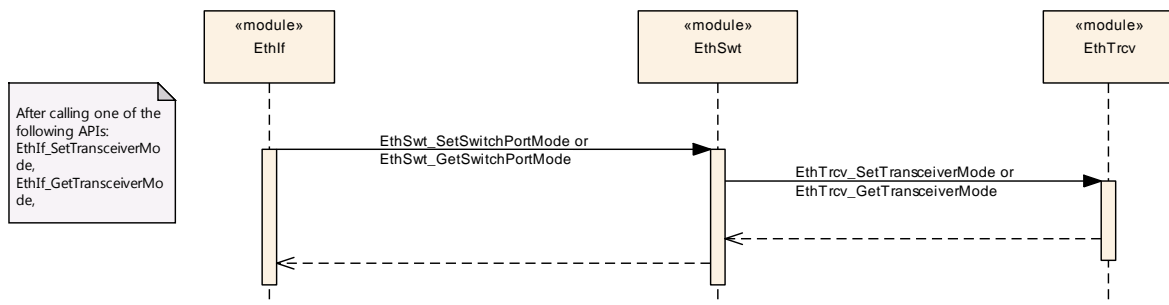
No direct access is necessary from the application layer.

## 9 Sequence diagrams

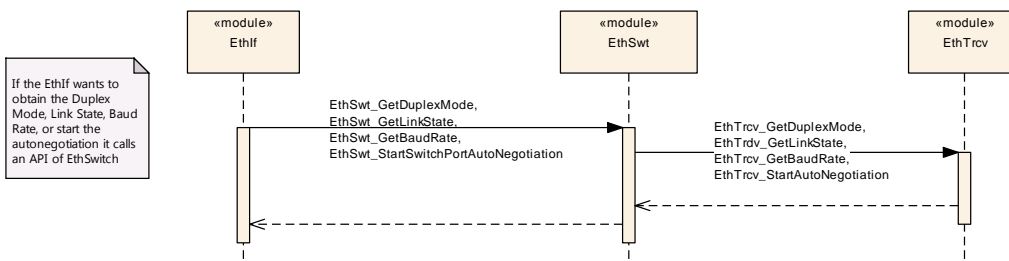
The following sequence diagram shows the interaction between the DHCP-Server in the TCP/IP-module and the Ethernet Switch Driver:



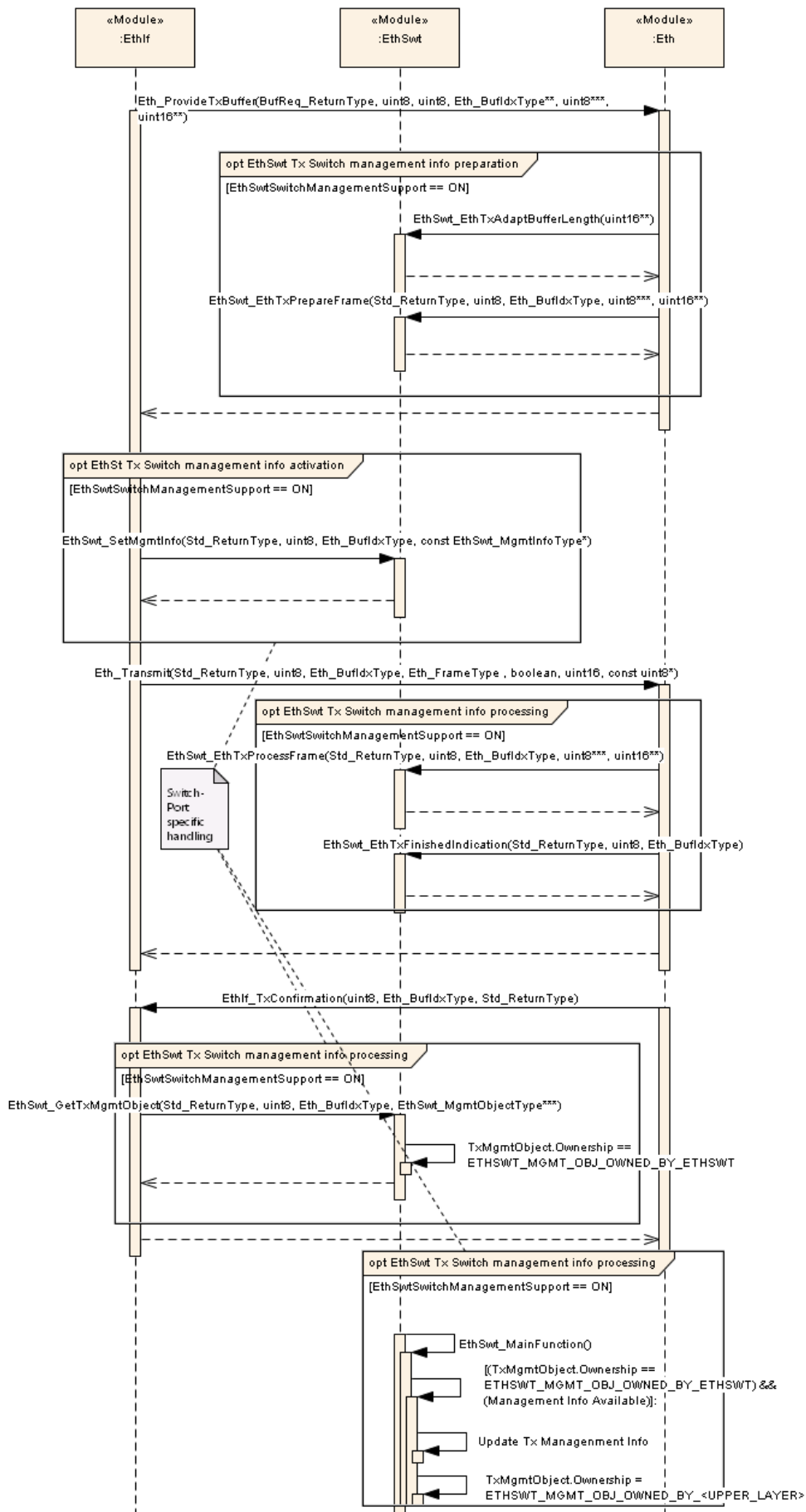
The following sequence diagram shows the interaction between the EthIf, EthSwT and the EthTrcv for API calls to the EthIf:



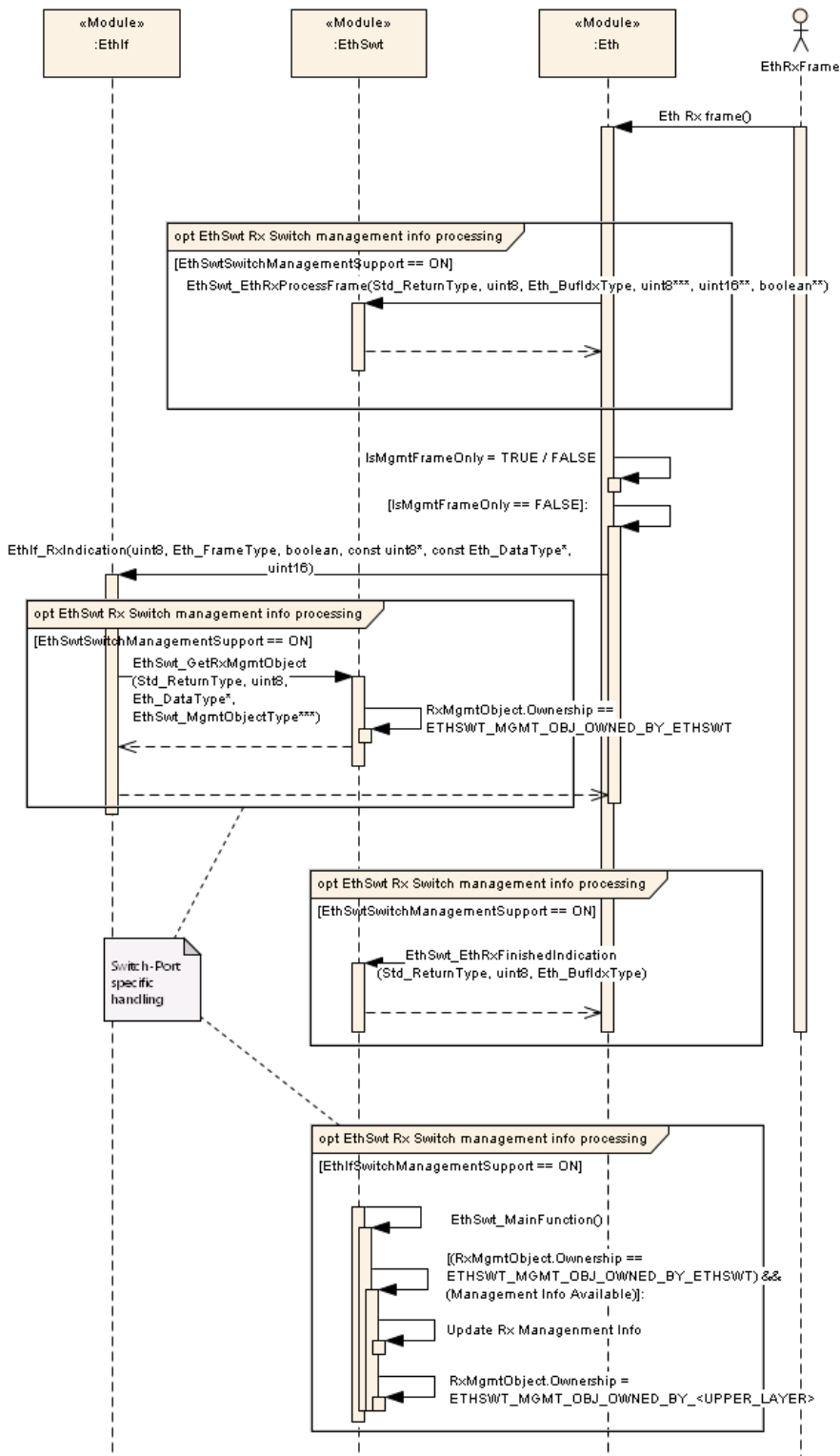
The following sequence diagram shows the interaction between the EthIf, EthSwT, and the EthTrcv for API calls which are initiated by the EthIf:



## 9.1 Switch Management support



**Figure 8 Switch Management support for transmission**



**Figure 9 Management support for reception**

## 10 Configuration specification

Chapter 10.1 specifies the structure (containers) and the parameters of the module EthSwt.

### 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.

**[SWS\_EthSwt\_00414]** DRAFT [ The Ethernet Switch Driver module shall reject configurations with partition mappings which are not supported by the implementation.] ()

#### 10.1.1 EthSwt

<b>SWS Item</b>	<b>ECUC_EthSwt_00046 :</b>
<b>Module Name</b>	EthSwt
<b>Module Description</b>	Configuration of the EthSwt (Ethernet Switch Driver) module.
<b>Post-Build Variant Support</b>	true
<b>Supported Config Variants</b>	VARIANT-LINK-TIME, VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
EthSwtConfig	1..*	Configuration of one Ethernet Switch.
EthSwtGeneral	1	General configuration of Ethernet Switch Driver module.

#### 10.1.2 EthSwtConfig

<b>SWS Item</b>	<b>ECUC_EthSwt_00001 :</b>		
<b>Container Name</b>	EthSwtConfig		
<b>Description</b>	Configuration of one Ethernet Switch.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00127 :</b>		
<b>Name</b>	EthSwtArItableEntryTimeout		
<b>Parent Container</b>	EthSwtConfig		
<b>Description</b>	If present, this parameter specifies the timeout in seconds for removing unused entries from the ARL table of the Ethernet switch. Otherwise, entries are not removed automatically.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucFloatParamDef		

<b>Range</b>	[1 .. 65535]		
<b>Default value</b>	300		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00128 :</b>		
<b>Name</b>	EthSwtClockSynchronizationSupport		
<b>Parent Container</b>	EthSwtConfig		
<b>Description</b>	<p>This parameter defines, if a Ethernet switch shall enable clock synchronization with another Ethernet switch to which it is connected via uplink port.</p> <p>If this parameter is set to TRUE the clock synchronization between connected Ethernet switches is activated and the clocks of the Ethernet switches are synchronized.</p> <p>If this parameter is set to FALSE the clock synchronization between connected Ethernet switches is deactivated.</p> <p>This parameter shall only be set to TRUE if the Ethernet switch hardware supports clock synchronization.</p>		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00073 :</b>		
<b>Name</b>	EthSwtDropDoubleTagged		
<b>Parent Container</b>	EthSwtConfig		
<b>Description</b>	<p>This parameter defines if a switch shall drop double tagged (Q in Q) frames.</p> <p>If this parameter is set to TRUE double tagged frames are dropped at all ports.</p> <p>If this parameter is set to FALSE, then double tagged frames are forwarded. If double tagging is used as a feature, this parameter must be set to FALSE.</p> <p>This parameter shall only be set to TRUE when Switch-HW supports the filtering of double tagged frames as filtering by SW is NOT possible!</p>		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD



<b>Scope / Dependency</b>	scope: local
---------------------------	--------------

<b>SWS Item</b>	<b>ECUC_EthSwt_00004 :</b>		
<b>Name</b>	EthSwtIdx		
<b>Parent Container</b>	EthSwtConfig		
<b>Description</b>	Specifies the instance ID of the configured Ethernet Switch.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
<b>Range</b>	0 .. 255		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwt_00130 :</b>		
<b>Name</b>	EthSwtConfigEcucPartitionRef		
<b>Parent Container</b>	EthSwtConfig		
<b>Description</b>	Maps the configuration of one single Ethernet switch to zero or one ECUC partitions. The ECUC partition referenced is a subset of the ECUC partitions where the Ethernet switch driver is mapped to. <b>Tags:</b> atp.Status=draft		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Reference to [ EcucPartition ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwt_00110 :</b>		
<b>Name</b>	EthSwtManagementEthCtrlRef		
<b>Parent Container</b>	EthSwtConfig		
<b>Description</b>	Reference to the Ethernet controller connected to the management port where the management frames will be transmitted/received.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Symbolic name reference to [ EthCtrlConfig ]		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00111 :</b>		
-----------------	----------------------------	--	--

<b>Name</b>	EthSwtManagementPortRef		
<b>Parent Container</b>	EthSwtConfig		
<b>Description</b>	Reference to the port where the management CPU is connected to.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Reference to [ EthSwtPort ]		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
EthSwtDemEventParameterRefs	0..1	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId symbolic value. The standardized errors are provided in this container and can be extended by vendor-specific error references.
EthSwtNvm	0..1	Configuration of one Ethernet Switch Nvm usage in case the module requires non volatile memory in the Ecu to store switch configuration.
EthSwtPort	1..*	Configuration of one Ethernet Switch Port.
EthSwtSpi	0..1	Configuration of one Ethernet Switch SPI access (if SPI is used).

### 10.1.3 EthSwtDemEventParameterRefs

<b>SWS Item</b>	<b>ECUC_EthSwt_00016 :</b>		
<b>Container Name</b>	EthSwtDemEventParameterRefs		
<b>Description</b>	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId symbolic value. The standardized errors are provided in this container and can be extended by vendor-specific error references.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00006 :</b>		
<b>Name</b>	ETHSWT_E_ACCESS		
<b>Parent Container</b>	EthSwtDemEventParameterRefs		

<b>Description</b>	Reference to the DemEventParameter which shall be issued when the error "Ethernet Switch Access Failure" has occurred.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Symbolic name reference to [ DemEventParameter ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00125 :</b>		
<b>Name</b>	ETHSWT_E_SYNCPORT2PHY		
<b>Parent Container</b>	EthSwtDemEventParameterRefs		
<b>Description</b>	Reference to the DemEventParameter which shall be issued when the error "Ethernet switch port and the referenced Ethernet transceiver are in contradicting modes" has occurred.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Symbolic name reference to [ DemEventParameter ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

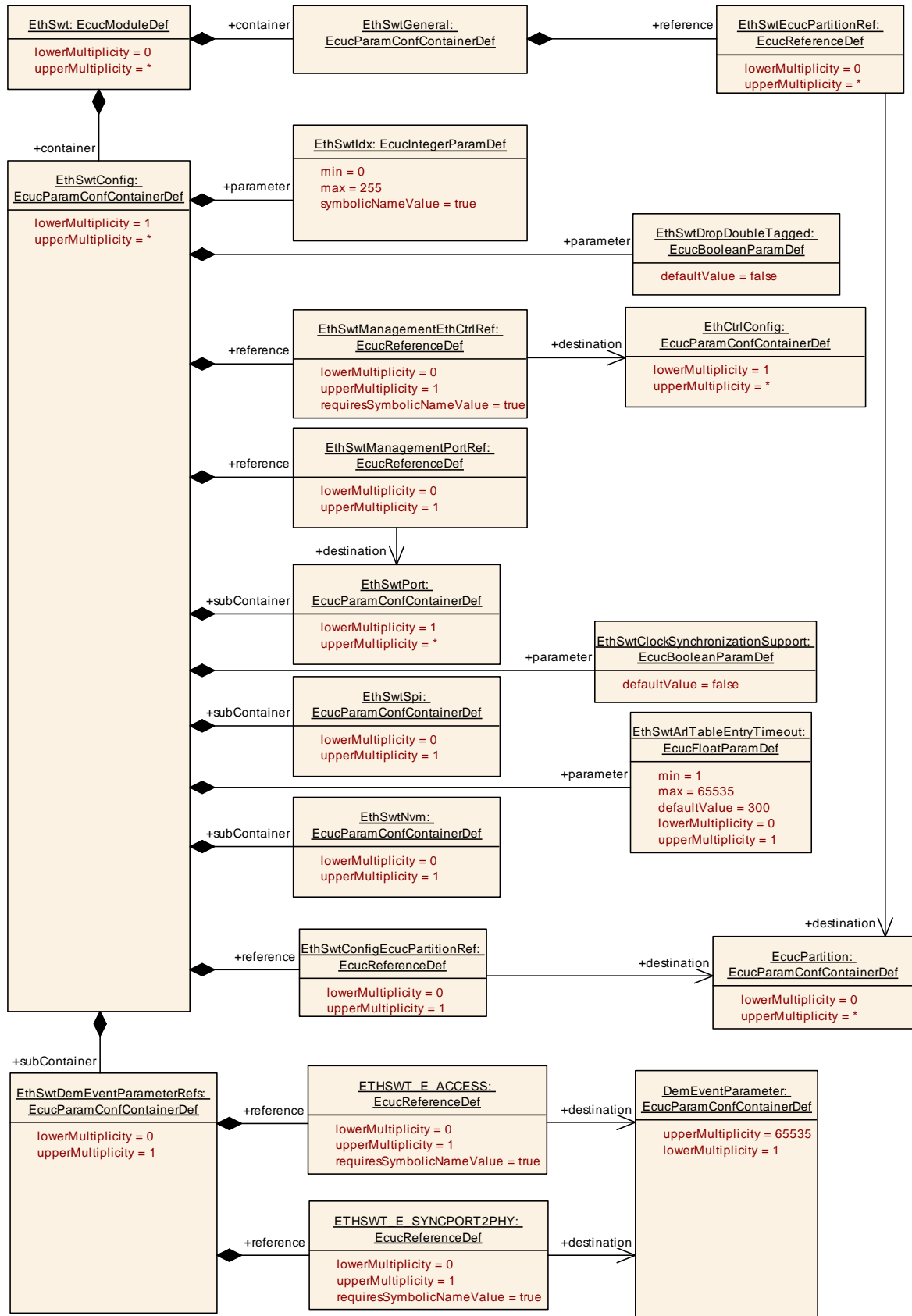


Figure 10: EthSwt

### 10.1.4 EthSwtGeneral

<b>SWS Item</b>	<b>ECUC_EthSwt_00003 :</b>		
<b>Container Name</b>	EthSwtGeneral		
<b>Description</b>	General configuration of Ethernet Switch Driver module.		
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00133 :</b>		
<b>Name</b>	EthSwtDeletePortMirrorConfigurationApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_DeletePortMirrorConfiguration API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00002 :</b>		
<b>Name</b>	EthSwtDevErrorDetect		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Switches the development error detection and notification on or off. <ul style="list-style-type: none"> <li>• true: detection and notification is enabled.</li> <li>• false: detection and notification is disabled.</li> </ul>		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00135 :</b>		
<b>Name</b>	EthSwtEnableCableDiagnosticApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enable/disable the APIs for cable diagnostic: EthSwt_RunPortCableDiagnostic, EthSwt_GetPortCableDiagnosticsResult		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00055 :</b>		
-----------------	----------------------------	--	--

<b>Name</b>	EthSwtEnableVlanApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_EnableVLAN API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00052 :</b>		
<b>Name</b>	EthSwtGetAriTableApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetAriTable API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00121 :</b>		
<b>Name</b>	EthSwtGetBaudRateApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetBaudRate API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00124 :</b>		
<b>Name</b>	EthSwtGetCfgDataRawDone		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Defines the function name for <GetCfgDataRawDone>		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucFunctionNameDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants

	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local dependency: The function GetCfgDataRawDone shall only be configured if parameter EthSwtGetCfgRaw is set to TRUE.		

<b>SWS Item</b>	<b>ECUC_EthSwt_00123 :</b>		
<b>Name</b>	EthSwtGetCfgRaw		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Disable /Enable support of reading raw data from switch memory		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00053 :</b>		
<b>Name</b>	EthSwtGetCounterValuesApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetCounterValues API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00122 :</b>		
<b>Name</b>	EthSwtGetDuplexModeApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetDuplexMode API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00120 :</b>		
<b>Name</b>	EthSwtGetLinkStateApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetLinkState API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	



<b>Scope / Dependency</b>	scope: local
---------------------------	--------------

<b>SWS Item</b>	<b>ECUC_EthSwt_00061 :</b>		
<b>Name</b>	EthSwtGetMacLearningModeApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetMacLearningMode API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00131 :</b>		
<b>Name</b>	EthSwtGetMaxFIFOBufferFillLevelApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetMaxFIFOBufferFillLevel API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00092 :</b>		
<b>Name</b>	EthSwtGetPortCableDiagnosticsResultApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetPortCableDiagnosticsResult API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00083 :</b>		
<b>Name</b>	EthSwtGetPortIdentifierApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetPortIdentifier API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00051 :</b>		
-----------------	----------------------------	--	--

<b>Name</b>	EthSwtGetPortMacAddrApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetPortMacAddr API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00087 :</b>		
<b>Name</b>	EthSwtGetPortMirrorStateApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetPortMirrorState API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00082 :</b>		
<b>Name</b>	EthSwtGetPortSignalQualityApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetPortSignalQuality API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00065 :</b>		
<b>Name</b>	EthSwtGetRxStatsApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetRxStats API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00084 :</b>		
<b>Name</b>	EthSwtGetSwitchIdentifierApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetSwitchIdentifier API		

<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00118 :</b>		
<b>Name</b>	EthSwtGetSwitchPortModeApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetSwitchPortMode API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00066 :</b>		
<b>Name</b>	EthSwtGetSwitchRegApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_GetSwitchReg API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00100 :</b>		
<b>Name</b>	EthSwtGetTxErrorCounterValuesApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables/Disables Eth_GetTxErrorCounterValues API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00099 :</b>		
<b>Name</b>	EthSwtGetTxStatsApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables/Disables Eth_GetTxStats API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		

<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00107 :</b>		
<b>Name</b>	EthSwtGlobalTimeSupportApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables/Disables the Global Time APIs used amongst others by Global Time Synchronization over Ethernet.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00033 :</b>		
<b>Name</b>	EthSwtIndex		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 255		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00115 :</b>		
<b>Name</b>	EthSwtLinkDownCallout		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Defines the function name for the <EthSwtLinkDownCallout> callout.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucFunctionNameDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00116 :</b>		
<b>Name</b>	EthSwtLinkUpCallout		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Defines the function name for the <EthSwtLinkUpCallout> callout.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucFunctionNameDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00102 :</b>		
<b>Name</b>	EthSwtLowPowerModeSupport		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Disable / Enable support of low power mode.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00071 :</b>		
<b>Name</b>	EthSwtMainFunctionPeriod		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	The cycle time of the periodic main function of EthSwt. Defined in seconds.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucFloatParamDef		
<b>Range</b>	]0 .. INF[		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00108 :</b>		
<b>Name</b>	EthSwtManagementSupportApi		

<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables/Disables the Switch management APIs to support a Switch-port specific communication attribute access.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00109 :</b>		
<b>Name</b>	EthSwtMgmtInfoIndicationTimeout		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	<p>This parameter specifies the timeout while the Switch driver is waiting for management information out of the Switch for reception. The value 0 deactivates the timeout supervision.</p> <p>Unit: seconds</p>		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucFloatParamDef		
<b>Range</b>	[0 .. INF[		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00062 :</b>		
<b>Name</b>	EthSwtPersistentConfigurationResult		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables the callback API <User>_PersistentConfigurationResult.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00063 :</b>		
<b>Name</b>	EthSwtPersistentConfigurationResultUser		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Defines the <User> function name for the <User>_PersistentConfigurationResult callback.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucFunctionNameDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	--		
<b>Post-Build Variant</b>	false		

<b>Multiplicity</b>			
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00064 :</b>		
<b>Name</b>	EthSwtPublicCddHeaderFile		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Defines header files for callback functions which shall be included in case of CDDs.		
<b>Multiplicity</b>	0..*		
<b>Type</b>	EcucStringParamDef		
<b>Default value</b>	--		
<b>maxLength</b>	32		
<b>minLength</b>	1		
<b>regularExpression</b>	--		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00086 :</b>		
<b>Name</b>	EthSwtReadPortMirrorConfigurationApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_ReadPortMirrorConfiguration API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00069 :</b>		
<b>Name</b>	EthSwtReadTrcvRegisterApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_ReadTrcvRegister API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	



<b>Scope / Dependency</b>	scope: local
---------------------------	--------------

<b>SWS Item</b>	<b>ECUC_EthSwt_00049 :</b>		
<b>Name</b>	EthSwtResetConfigurationApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_ResetConfiguration API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00104 :</b>		
<b>Name</b>	EthSwtSetForwardingModeApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables /disables EthSwt_SetForwardingMode API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00060 :</b>		
<b>Name</b>	EthSwtSetMacLearningModeApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_SetMacLearningMode API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00090 :</b>		
<b>Name</b>	EthSwtSetPortLoopbackModeApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_SetPortLoopbackModeApi API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00088 :</b>		
-----------------	----------------------------	--	--

<b>Name</b>	EthSwtSetPortMirrorStateApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_SetPortMirrorState API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00089 :</b>		
<b>Name</b>	EthSwtSetPortTestModeApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_SetPortTestMode API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00091 :</b>		
<b>Name</b>	EthSwtSetPortTxModeApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_SetPortTxModeApi API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00117 :</b>		
<b>Name</b>	EthSwtSetSwitchPortModeApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_SetSwitchPortMode API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00067 :</b>		
<b>Name</b>	EthSwtSetSwitchRegApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_SetSwitchReg API.		

<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00119 :</b>		
<b>Name</b>	EthSwtStartSwitchPortAutoNegotiationApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_StartSwitchPortAutoNegotiation API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00050 :</b>		
<b>Name</b>	EthSwtStoreConfigurationApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_StoreConfiguration API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00105 :</b>		
<b>Name</b>	EthSwtVerifyConfigApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables /disables EthSwt_VerifyConfig API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00031 :</b>		
<b>Name</b>	EthSwtVersionInfoApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables version info API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		

<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00085 :</b>		
<b>Name</b>	EthSwtWritePortMirrorConfigurationApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_WritePortMirrorConfiguration API		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00070 :</b>		
<b>Name</b>	EthSwtWriteTrcvRegisterApi		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	Enables / Disables EthSwt_WriteTrcvRegister API.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00129 :</b>		
<b>Name</b>	EthSwtEcucPartitionRef		
<b>Parent Container</b>	EthSwtGeneral		
<b>Description</b>	<p>Maps the Ethernet switch driver to zero or multiple ECUC partitions to make the modules API available in this partition. The Ethernet switch driver will operate as an independent instance in each of the partitions.</p> <p><b>Tags:</b> atp.Status=draft</p>		
<b>Multiplicity</b>	0..*		
<b>Type</b>	Reference to [ EcucPartition ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

**No Included Containers**

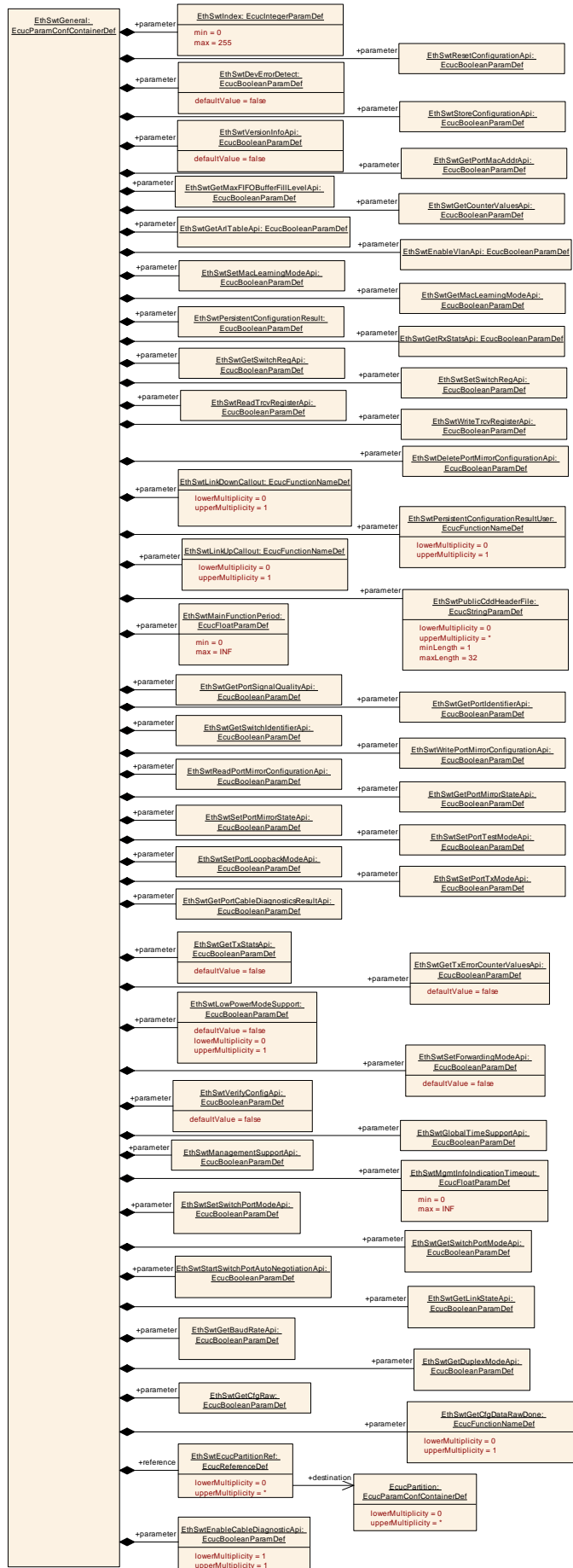


Figure 11: EthSwtGeneral

### 10.1.5 EthSwtPort

<b>SWS Item</b>	<b>ECUC_EthSwt_00005 :</b>		
<b>Container Name</b>	EthSwtPort		
<b>Description</b>	Configuration of one Ethernet Switch Port.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00013 :</b>		
<b>Name</b>	EthSwtPortIdx		
<b>Parent Container</b>	EthSwtPort		
<b>Description</b>	Specifies the instance ID of the configured Ethernet Switch Port.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
<b>Range</b>	0 .. 255		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwt_00114 :</b>		
<b>Name</b>	EthSwtPortMacLayerSpeed		
<b>Parent Container</b>	EthSwtPort		
<b>Description</b>	Defines the baud rate of the MAC layer.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	ETH_MAC_LAYER_SPEED_100M	--	
	ETH_MAC_LAYER_SPEED_10G	--	
	ETH_MAC_LAYER_SPEED_10M	--	
	ETH_MAC_LAYER_SPEED_1G	--	
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwT_00113 :</b>		
<b>Name</b>	EthSwTPortMacLayerSubType		
<b>Parent Container</b>	EthSwTPort		
<b>Description</b>	Defines the MAC layer subtype of this EthSwTPort.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	REDUCED		Reduced media-independent interface
	REVERSED		reversed media-independent interface (to provide direct connection between two Ethernet MACs)
	SERIAL		low-power and low pin-count serial 8b/10b-coded media-independent interface
	STANDARD		standard media-independent interface
	UNIVERSAL_SERIAL		Universal low-power and low pin-count serial 8b/10b-coded media-independent interface
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwT_00072 :</b>		
<b>Name</b>	EthSwTPortMacLayerType		
<b>Parent Container</b>	EthSwTPort		
<b>Description</b>	Defines the MAC layer type of this EthSwTPort.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	ETHSWT_PORT_MAC_LAYER_TYPE_XGMII		MAC layer interface (data) bandwidth class 1Gbit/s (e.g. GMII, RGMII, SGMII, RvGMII, USGMII)
	ETHSWT_PORT_MAC_LAYER_TYPE_XMII		MAC layer interface (data) bandwidth class 100Mbit/s (e.g. RMII, RvMII, SMII, RvMII)
	ETHSWT_PORT_MAC_LAYER_TYPE_XXGMII		MAC layer interface (data) bandwidth class 10Gbit/s
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	



<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwt_00054 :</b>		
<b>Name</b>	EthSwtPortPhysicalLayerType		
<b>Parent Container</b>	EthSwtPort		
<b>Description</b>	Defines the physical layer type of this EthSwtPort.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	ETHSWT_PORT_1000BASE_T	physical layer interface 1000BASE-T (1Gbit/s, 4 pairs). Used for consumer electronic.	
	ETHSWT_PORT_1000BASE_T1	physical layer interface 1000BASE-T1 (1Gbit/s, 1 pair). Used for automotive.	
	ETHSWT_PORT_100BASE_T1	physical layer interface 100BASE-T1 (100Mbit/s, 1 pair). Used for automotive.	
	ETHSWT_PORT_100BASE_TX	physical layer interface 100BASE-TX (100Mbit/s, 2 pairs). Used for consumer electronic.	
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU dependency: If a EthSwtPort has an EthSwtPortPhysicalLayerType then EthSwtPort shall reference an EthTrcv.		

<b>SWS Item</b>	<b>ECUC_EthSwt_00032 :</b>		
<b>Name</b>	EthSwtPortPredefinedMacAddresses		
<b>Parent Container</b>	EthSwtPort		
<b>Description</b>	Specifies a list of 48-bit physical addresses (MAC addresses) which can be reached via this port in network byte order. Note that further addresses can be learned during runtime.		
<b>Multiplicity</b>	0..*		
<b>Type</b>	EcucStringParamDef		
<b>Default value</b>	--		
<b>maxLength</b>	--		
<b>minLength</b>	--		
<b>regularExpression</b>	[0-9a-fA-F]{2}[:-][0-9a-fA-F]{2}{5}		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE

<b>Class</b>	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00101 :</b>		
<b>Name</b>	EthSwtPortRole		
<b>Parent Container</b>	EthSwtPort		
<b>Description</b>	Set a special role of the Ethernet switch port. It is either a host port or a up link port. If not configured it is a standard port.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	ETHSWT_HOST_PORT	The hostPort is connected to an ECU (host ecu). The host ECU controls the connected CouplingElement (e.g. Ethernet switch).	
	ETHSWT_UP_LINK_PORT	A CouplingPort can be connected to another CouplingPort of a CouplingElement located on the same ECU (CouplingElement.ecuInstance) using the CouplingPortConnection. This is used to model a cascaded switch.	
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local dependency: One Ethernet switch shall have either exactly one host port or at least one up link port. In case of having a host port also multiple up link port can exist.  A master switch shall be connected by one host port with the host ecu.  A slave switch shall be connected to a master switch by one up link port.		

<b>SWS Item</b>	<b>ECUC_EthSwt_00112 :</b>		
<b>Name</b>	EthSwtPortTimeStampSupport		
<b>Parent Container</b>	EthSwtPort		
<b>Description</b>	Enables/Disables the Switch-port specific timestamping.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local dependency: EthSwtPortTimeStampSupport can only be set to TRUE, * if (EthSwtClockSynchronizationSupport is FALSE) OR * if ((EthSwtClockSynchronizationSupport is TRUE) AND (EthSwtPortRole		

	is NOT ETHSWT_UP_LINK_PORT))
--	------------------------------

<b>SWS Item</b>	<b>ECUC_EthSwt_00041 :</b>		
<b>Name</b>	EthSwtPortTrcvRef		
<b>Parent Container</b>	EthSwtPort		
<b>Description</b>	Reference to the Ethernet transceiver driver this EthSwtPort is connected with.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Symbolic name reference to [ EthTrcvConfig ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU dependency: If EthSwtPortPhysicalLayerType is defined, then EthSwtPortTrcvRef holds the reference to the corresponding EthTrcv.		

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
EthSwtPortEgress	1	Configuration of one Ethernet Switch Port Egress behavior.
EthSwtPortIngress	1	Configuration of one Ethernet Switch Port ingress behavior.
EthSwtPortVlanMembership	0..4095	Description Determines the membership of this port to the virtual network, i.e. frames with this VID can be received and transmitted via this port.

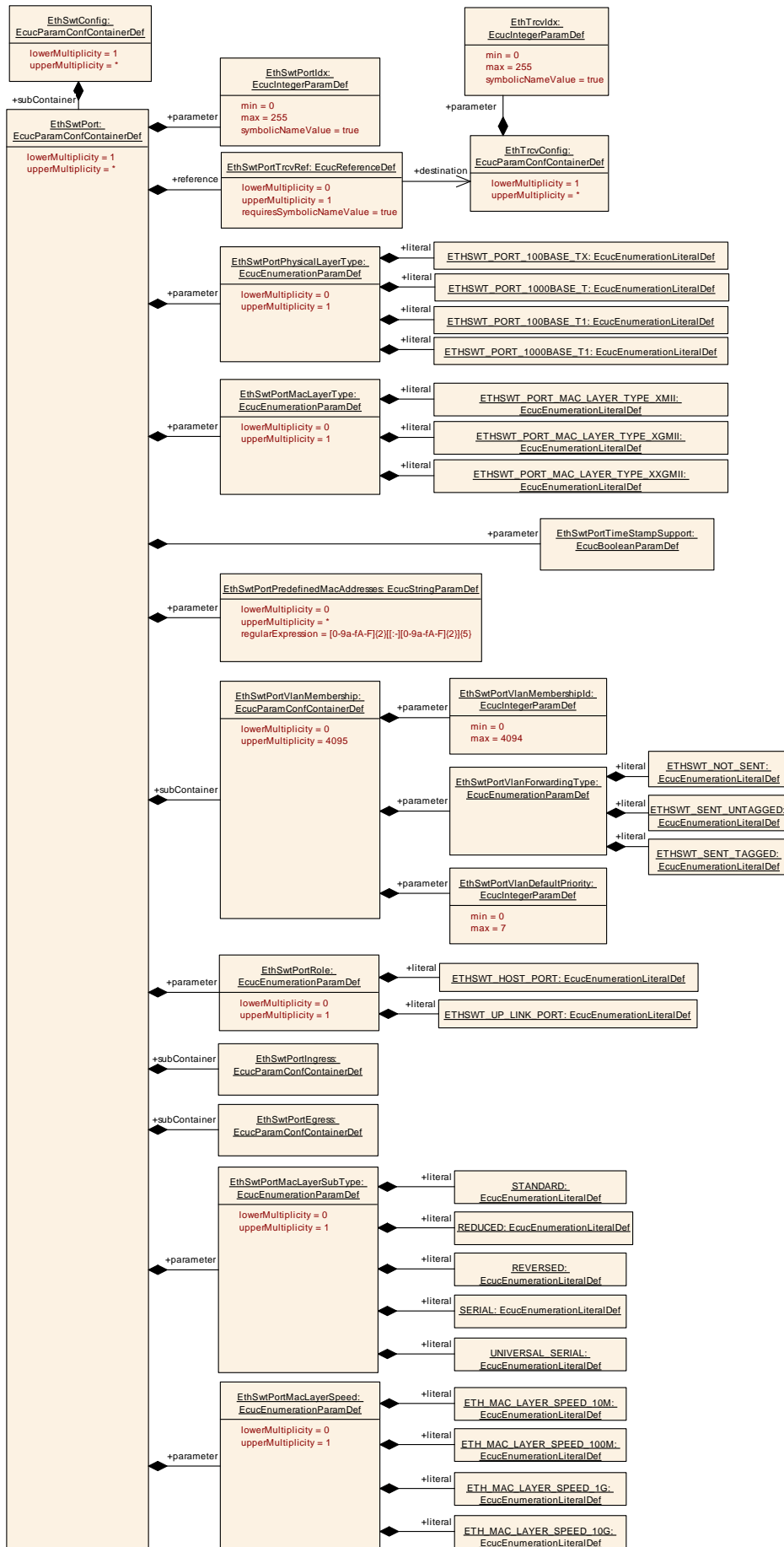


Figure 12: EthSwtPort

Please note that the functional behavior of the ingress and egress port of a switch is implemented in hardware in the switch devices (see 0). Thus, the configuration of **EthSwtPort** and described in the following has to be written to the switch device or is related to the switch configuration.

### 10.1.6 EthSwtPortIngress

<b>SWS Item</b>	<b>ECUC_EthSwt_00014 :</b>
<b>Container Name</b>	EthSwtPortIngress
<b>Description</b>	Configuration of one Ethernet Switch Port ingress behavior.
<b>Configuration Parameters</b>	

<b>SWS Item</b>	<b>ECUC_EthSwt_00096 :</b>		
<b>Name</b>	EthSwtPortIngressDefaultPriority		
<b>Parent Container</b>	EthSwtPortIngress		
<b>Description</b>	Default priority for ingress.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	0 .. 7		
<b>Default value</b>	0		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local dependency: If EthSwtPortIngressDefaultPriority is configured (multiplicity set to 1) then EthSwtPortIngressDefaultVlan shall be configured.  If EthSwtPortIngressDefaultVlan is configured EthSwtPortIngressDropUntagged shall be set to FALSE.		

<b>SWS Item</b>	<b>ECUC_EthSwt_00095 :</b>		
<b>Name</b>	EthSwtPortIngressDefaultVlan		
<b>Parent Container</b>	EthSwtPortIngress		
<b>Description</b>	Default VLAN for ingress.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	0 .. 4094		
<b>Default value</b>	1		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE

	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	<p>scope: local            dependency: If EthSwtPortIngressDefaultVlan is configured (multiplicity set to 1) then EthSwtPortIngressDefaultPriority shall be configured.</p> <p>If EthSwtPortIngressDefaultVlan is configured            EthSwtPortIngressDropUntagged shall be set to FALSE.</p>		

<b>SWS Item</b>	<b>ECUC_EthSwt_00097 :</b>		
<b>Name</b>	EthSwtPortIngressDropUntagged		
<b>Parent Container</b>	EthSwtPortIngress		
<b>Description</b>	Defines the ingress behavior for untagged frames.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	<p>scope: local            dependency: If EthSwtPortIngressDropUntagged is set to TRUE then EthSwtPortIngressDefaultVlan and EthSwtPortIngressDefaultPriority parameters shall not be configured.</p>		

<b>SWS Item</b>	<b>ECUC_EthSwt_00015 :</b>		
<b>Name</b>	EthSwtPortIngressVlanModification		
<b>Parent Container</b>	EthSwtPortIngress		
<b>Description</b>	<p>If this parameter is defined all messages which arrive at this ingress port will be tagged with this VLAN Id. This tagging happen also if the arriving message already has a VLAN Id, it will be overwritten by the defined one.</p> <p>If this parameter is not defined no changes to the VLAN Id shall happen at this ingress port.</p>		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 4095		
<b>Default value</b>	--		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwt_00023 :</b>		
<b>Name</b>	EthSwtPortTrafficClassAssignment		
<b>Parent Container</b>	EthSwtPortIngress		
<b>Description</b>	<p>If this parameter is defined all arriving messages at this ingress port shall be assigned this traffic class.</p> <p>If this parameter is not defined no general port based traffic class assignment is done.</p>		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucIntegerParamDef		

<b>Range</b>	0 .. 7		
<b>Default value</b>	--		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
EthSwtPortPolicer	0..32760	Definition of Rate Policing parameters.
EthSwtPriorityRegeneration	0..8	<p>Defines a priority regeneration where the EthSwtPriorityRegenerationIngressPriority is replaced by EthSwtPriorityRegenerationRegeneratedPriority. The EthSwtPriorityRegeneration is optional in case no priority regeneration shall be performed.</p> <p>In case a EthSwtPriorityRegeneration is defined it shall have 8 mappings, one for each priority.</p>
EthSwtPriorityTrafficClassAssignment	0..8	<p>Defines a priority based traffic class assignment. All messages with a specific priority (EthSwtPriorityTrafficClassAssignmentPriority) arriving at this ingress port or, if enabled regenerated priorities (EthSwtPriorityRegeneration), shall be assigned to a traffic class (EthSwtPriorityTrafficClassAssignmentTrafficClass).</p>



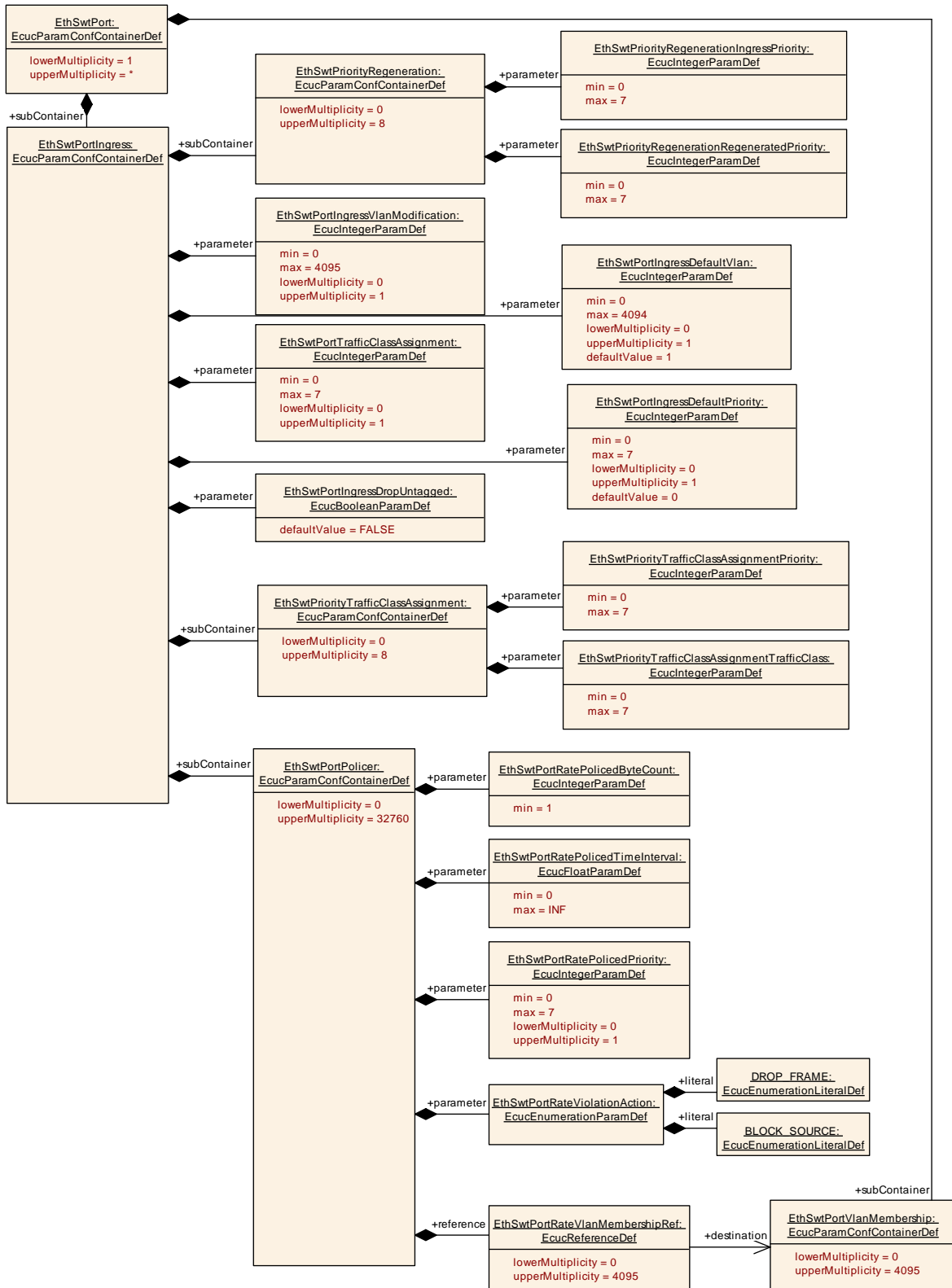


Figure 13: EthSwtPortIngress

### 10.1.7 EthSwtPortPolicer

<b>SWS Item</b>	<b>ECUC_EthSwt_00074 :</b>		
<b>Container Name</b>	EthSwtPortPolicer		
<b>Description</b>	Definition of Rate Policing parameters.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00075 :</b>		
<b>Name</b>	EthSwtPortRatePolicedByteCount		
<b>Parent Container</b>	EthSwtPortPolicer		
<b>Description</b>	Amount of Byte Counts (excluding Header information) which can be received in a configured EthSwtPortRatePolicedTimeInterval.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	1 .. 18446744073709551615		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00077 :</b>		
<b>Name</b>	EthSwtPortRatePolicedPriority		
<b>Parent Container</b>	EthSwtPortPolicer		
<b>Description</b>	Defines the priority which this rate policy shall be limited on. If no priority is given this rate policy is not considering priority.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 7		
<b>Default value</b>	--		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local dependency: If no priority is configured the rate policing only applies to the configured EthSwtPortRateVlanMembershipRef.		

<b>SWS Item</b>	<b>ECUC_EthSwt_00076 :</b>		
<b>Name</b>	EthSwtPortRatePolicedTimeInterval		
<b>Parent Container</b>	EthSwtPortPolicer		
<b>Description</b>	Time interval in seconds where a configured EthSwtPortRatePolicedByteCount can be received without a rate limitation.		
<b>Multiplicity</b>	1		

<b>Type</b>	EcucFloatParamDef		
<b>Range</b>	]0 .. INF[		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00078 :</b>		
<b>Name</b>	EthSwtPortRateViolationAction		
<b>Parent Container</b>	EthSwtPortPolicer		
<b>Description</b>	Action to be taken when the rate policy criteria defined for this EthSwtPortPolicer are met.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	BLOCK_SOURCE	All incoming traffic from the violating Source based on the MAC-Address is blocked.	
	DROP_FRAME	The received frame which led to the violation of the rate policy is dropped.	
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00081 :</b>		
<b>Name</b>	EthSwtPortRateVlanMembershipRef		
<b>Parent Container</b>	EthSwtPortPolicer		
<b>Description</b>	References the Vlans this rate policy shall apply to. If no EthSwtPortRateVlanMembershipRef is configured the rate policing applies only on the configured EthSwtPortRatePolicedPriority.		
<b>Multiplicity</b>	0..4095		
<b>Type</b>	Reference to [ EthSwtPortVlanMembership ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

### 10.1.8 EthSwtPriorityRegeneration

<b>SWS Item</b>	<b>ECUC_EthSwt_00057 :</b>		
<b>Container Name</b>	EthSwtPriorityRegeneration		

<b>Description</b>	<p>Defines a priority regeneration where the EthSwtPriorityRegenerationIngressPriority is replaced by EthSwtPriorityRegenerationRegeneratedPriority.</p> <p>The EthSwtPriorityRegeneration is optional in case no priority regeneration shall be performed.</p> <p>In case a EthSwtPriorityRegeneration is defined it shall have 8 mappings, one for each priority.</p>		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00058 :</b>		
<b>Name</b>	EthSwtPriorityRegenerationIngressPriority		
<b>Parent Container</b>	EthSwtPriorityRegeneration		
<b>Description</b>	Message priority of the incoming message.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 7		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwt_00059 :</b>		
<b>Name</b>	EthSwtPriorityRegenerationRegeneratedPriority		
<b>Parent Container</b>	EthSwtPriorityRegeneration		
<b>Description</b>	Message priority the incoming message will be tagged with.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 7		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

**No Included Containers**

### 10.1.9 EthSwtPriorityTrafficClassAssignment

<b>SWS Item</b>	<b>ECUC_EthSwt_00027 :</b>		
<b>Container Name</b>	EthSwtPriorityTrafficClassAssignment		
<b>Description</b>	Defines a priority based traffic class assignment. All messages with a specific priority (EthSwtPriorityTrafficClassAssignmentPriority) arriving at		

	this ingress port or, if enabled regenerated priorities (EthSwtPriorityRegeneration), shall be assigned to a traffic class (EthSwtPriorityTrafficClassAssignmentTrafficClass).		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00028 :</b>		
<b>Name</b>	EthSwtPriorityTrafficClassAssignmentPriority		
<b>Parent Container</b>	EthSwtPriorityTrafficClassAssignment		
<b>Description</b>	Message priority.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 7		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwt_00029 :</b>		
<b>Name</b>	EthSwtPriorityTrafficClassAssignmentTrafficClass		
<b>Parent Container</b>	EthSwtPriorityTrafficClassAssignment		
<b>Description</b>	Traffic Class value.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 7		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

**No Included Containers**

### 10.1.10 EthSwtPortEgress

<b>SWS Item</b>	<b>ECUC_EthSwt_00007 :</b>		
<b>Container Name</b>	EthSwtPortEgress		
<b>Description</b>	Configuration of one Ethernet Switch Port Egress behavior.		
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00008 :</b>		
<b>Name</b>	EthSwtPortEgressLastSchedulerRef		
<b>Parent Container</b>	EthSwtPortEgress		
<b>Description</b>	Reference to the port scheduler which is the last in the egress port		

	structure.		
<b>Multiplicity</b>	1		
<b>Type</b>	Reference to [ EthSwtPortScheduler ]		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

Included Containers		
Container Name	Multiplicity	Scope / Dependency
EthSwtPortFifo	1..*	Represents a Fifo in the egress port.
EthSwtPortScheduler	1..*	Represents a Scheduler in the egress port.
EthSwtPortShaper	0..*	Represents a Shaper in the egress port.

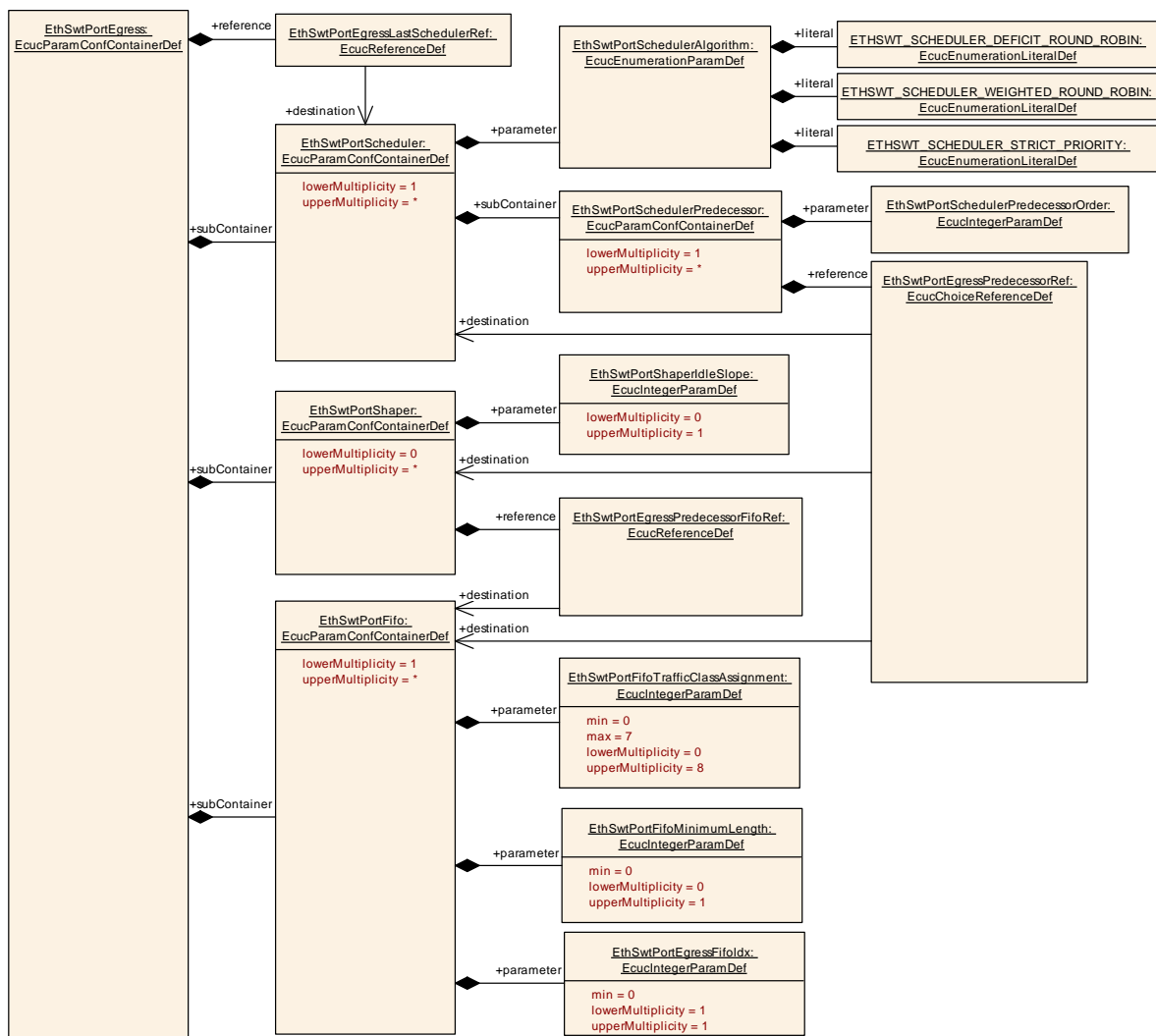


Figure 14: EthSwtPortEgress

### 10.1.11 EthSwtPortScheduler

<b>SWS Item</b>	<b>ECUC_EthSwt_00017 :</b>
-----------------	----------------------------

<b>Container Name</b>	EthSwtPortScheduler		
<b>Description</b>	Represents a Scheduler in the egress port.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00018 :</b>		
<b>Name</b>	EthSwtPortSchedulerAlgorithm		
<b>Parent Container</b>	EthSwtPortScheduler		
<b>Description</b>	Defines the scheduler algorithm.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	ETHSWT_SCHEDULER_DEFICIT_ROUND_ROBIN		deficit round robin
	ETHSWT_SCHEDULER_STRICT_PRIORITY		strict priority
	ETHSWT_SCHEDULER_WEIGHTED_ROUND_ROBIN		weighted round robin
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
EthSwtPortSchedulerPredecessor	1..*	Defines an ordered list of predecessors for this scheduler.

### 10.1.12 EthSwtPortSchedulerPredecessor

<b>SWS Item</b>	<b>ECUC_EthSwt_00019 :</b>		
<b>Container Name</b>	EthSwtPortSchedulerPredecessor		
<b>Description</b>	Defines an ordered list of predecessors for this scheduler.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00020 :</b>		
<b>Name</b>	EthSwtPortSchedulerPredecessorOrder		
<b>Parent Container</b>	EthSwtPortSchedulerPredecessor		
<b>Description</b>	Defines the order of the scheduler predecessors. This value has to be understood as a relative value, i.e. the value shows only the relative ordering of the elements. The highest value has the		



	highest priority and gaps are allowed (not dense based). The values need to be unique within one EthSwtPortScheduler.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 ..		18446744073709551615
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwt_00010 :</b>		
<b>Name</b>	EthSwtPortEgressPredecessorRef		
<b>Parent Container</b>	EthSwtPortSchedulerPredecessor		
<b>Description</b>	Choice reference to the scheduler predecessor.		
<b>Multiplicity</b>	1		
<b>Type</b>	Choice reference to [ EthSwtPortFifo , EthSwtPortScheduler , EthSwtPortShaper ]		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

### 10.1.13 EthSwtPortShaper

<b>SWS Item</b>	<b>ECUC_EthSwt_00021 :</b>		
<b>Container Name</b>	EthSwtPortShaper		
<b>Description</b>	Represents a Shaper in the egress port.		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00042 :</b>		
<b>Name</b>	EthSwtPortShaperIdleSlope		
<b>Parent Container</b>	EthSwtPortShaper		
<b>Description</b>	Defines the increase of credit in bits per second for the AVB shaper.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 ..		18446744073709551615
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME

	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_EthSwt_00009 :</b>		
<b>Name</b>	EthSwtPortEgressPredecessorFifoRef		
<b>Parent Container</b>	EthSwtPortShaper		
<b>Description</b>	Reference to the fifo which is the predecessor for this shaper.		
<b>Multiplicity</b>	1		
<b>Type</b>	Reference to [ EthSwtPortFifo ]		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

### 10.1.14 EthSwtPortFifo

<b>SWS Item</b>	<b>ECUC_EthSwt_00011 :</b>		
<b>Container Name</b>	EthSwtPortFifo		
<b>Description</b>	Represents a Fifo in the egress port.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00132 :</b>		
<b>Name</b>	EthSwtPortEgressFifoldx		
<b>Parent Container</b>	EthSwtPortFifo		
<b>Description</b>	Specifies the instance ID of the fifo of the configured Ethernet switch egress port		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 ..		
	18446744073709551615		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwt_00098 :</b>		
<b>Name</b>	EthSwtPortFifoMinimumLength		
<b>Parent Container</b>	EthSwtPortFifo		
<b>Description</b>	FIFO minimum length in Byte. This assignment is used to configure a guaranteed size of a configured FIFO.		
<b>Multiplicity</b>	0..1		

<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 18446744073709551615		
<b>Default value</b>	--		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwt_00012 :</b>		
<b>Name</b>	EthSwtPortFifoTrafficClassAssignment		
<b>Parent Container</b>	EthSwtPortFifo		
<b>Description</b>	Defines which traffic classes are assigned to this Fifo.		
<b>Multiplicity</b>	0..8		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 7		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

**No Included Containers**

### 10.1.15 EthSwtPortVlanMembership

<b>SWS Item</b>	<b>ECUC_EthSwt_00079 :</b>		
<b>Container Name</b>	EthSwtPortVlanMembership		
<b>Description</b>	Description Determines the membership of this port to the virtual network, i.e. frames with this VID can be received and transmitted via this port.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_EthSwt_00056 :</b>		
<b>Name</b>	EthSwtPortVlanDefaultPriority		
<b>Parent Container</b>	EthSwtPortVlanMembership		
<b>Description</b>	Determines the standard output-priority outgoing messages will be tagged with.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 7		
<b>Default value</b>	--		

<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwt_00026 :</b>		
<b>Name</b>	EthSwtPortVlanForwardingType		
<b>Parent Container</b>	EthSwtPortVlanMembership		
<b>Description</b>	Defines how the message with a specific VLAN Id shall be handled.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	ETHSWT_NOT_SENT	The message with the specific VLAN Id shall not be sent at this port.	
	ETHSWT_SENT_TAGGED	The message with the specific VLAN Id shall be sent with its VLAN Id at this port.	
	ETHSWT_SENT_UNTAGGED	The message with the specific VLAN Id shall sent untagged.	
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwt_00080 :</b>		
<b>Name</b>	EthSwtPortVlanMembershipId		
<b>Parent Container</b>	EthSwtPortVlanMembership		
<b>Description</b>	Determines the VID of the virtual network this port belongs to.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 4094		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	X	VARIANT-LINK-TIME
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: ECU		

**No Included Containers**

### 10.1.16 EthSwtSpi

<b>SWS Item</b>	<b>ECUC_EthSwt_00030 :</b>		
<b>Container Name</b>	EthSwtSpi		
<b>Description</b>	Configuration of one Ethernet Switch SPI access (if SPI is used).		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	

	<b>Post-build time</b>	--	
--	------------------------	----	--

**Configuration Parameters**

**Included Containers**

Container Name	Multiplicity	Scope / Dependency
EthSwtSpiSequence	1..*	Container gives EthSwt driver information about one SPI sequence. One SPI sequence used by EthSwt driver is in exclusive use for it. No other driver is allowed to access this sequence. EthSwt driver may use one sequence to access n EthSwt hardware chips of the same type or n sequences are used to access one single EthSwt hardware chip. If a EthSwt hardware has no SPI interface, there is no instance of this container.

### 10.1.17 EthSwtSpiSequence

<b>SWS Item</b>	<b>ECUC_EthSwt_00034 :</b>		
<b>Container Name</b>	EthSwtSpiSequence		
<b>Description</b>	Container gives EthSwt driver information about one SPI sequence. One SPI sequence used by EthSwt driver is in exclusive use for it. No other driver is allowed to access this sequence. EthSwt driver may use one sequence to access n EthSwt hardware chips of the same type or n sequences are used to access one single EthSwt hardware chip. If a EthSwt hardware has no SPI interface, there is no instance of this container.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	

**Configuration Parameters**

<b>SWS Item</b>	<b>ECUC_EthSwt_00036 :</b>		
<b>Name</b>	EthSwtSpiAccessSynchronous		
<b>Parent Container</b>	EthSwtSpiSequence		
<b>Description</b>	This parameter is used to define whether the access to the Spi sequence is synchronous or asynchronous. true: SPI access is synchronous. false: SPI access is asynchronous.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	--		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_EthSwt_00035 :</b>		
-----------------	----------------------------	--	--

<b>Name</b>	EthSwtSpiSequenceName		
<b>Parent Container</b>	EthSwtSpiSequence		
<b>Description</b>	Reference to a Spi sequence configuration container.		
<b>Multiplicity</b>	0..*		
<b>Type</b>	Symbolic name reference to [ SpiSequence ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

**No Included Containers**

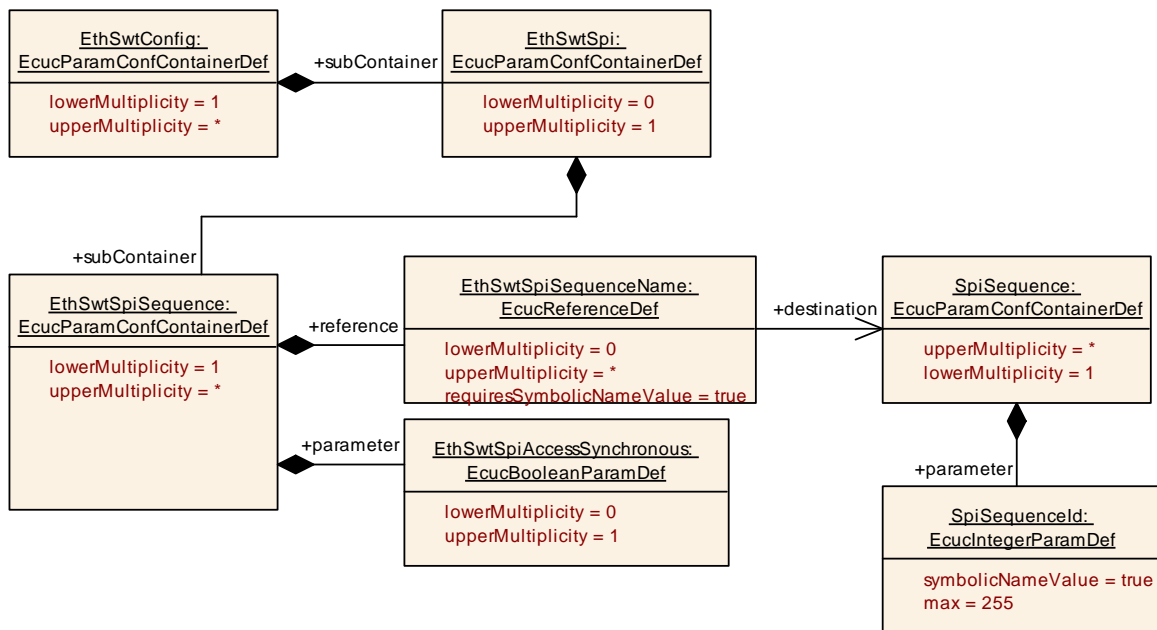


Figure 15: EthSwt Spi interaction

### 10.1.18 EthSwtNvm

<b>SWS Item</b>	ECUC_EthSwt_00043 :		
<b>Container Name</b>	EthSwtNvm		
<b>Description</b>	Configuration of one Ethernet Switch Nvm usage in case the module requires non volatile memory in the Ecu to store switch configuration.		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	

	<b>Post-build time</b>	--	
<b>Configuration Parameters</b>			
<b>SWS Item</b>	<b>ECUC_EthSwt_00134 :</b>		
<b>Name</b>	EthSwtConfigurationNvmBlockDescriptorRef		
<b>Parent Container</b>	EthSwtNvm		
<b>Description</b>	Reference to the Nvm block description in the Nvm module configuration to store e.g. the port mirror configurations		
<b>Multiplicity</b>	1		
<b>Type</b>	Symbolic name reference to [ NvMBlockDescriptor ]		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		
<b>SWS Item</b>	<b>ECUC_EthSwt_00044 :</b>		
<b>Name</b>	EthSwtTableNvmBlockDescriptorRef		
<b>Parent Container</b>	EthSwtNvm		
<b>Description</b>	Reference to the Nvm block description in the Nvm module configuration to store e.g. the learned ARL table		
<b>Multiplicity</b>	1		
<b>Type</b>	Symbolic name reference to [ NvMBlockDescriptor ]		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		
<b>No Included Containers</b>			

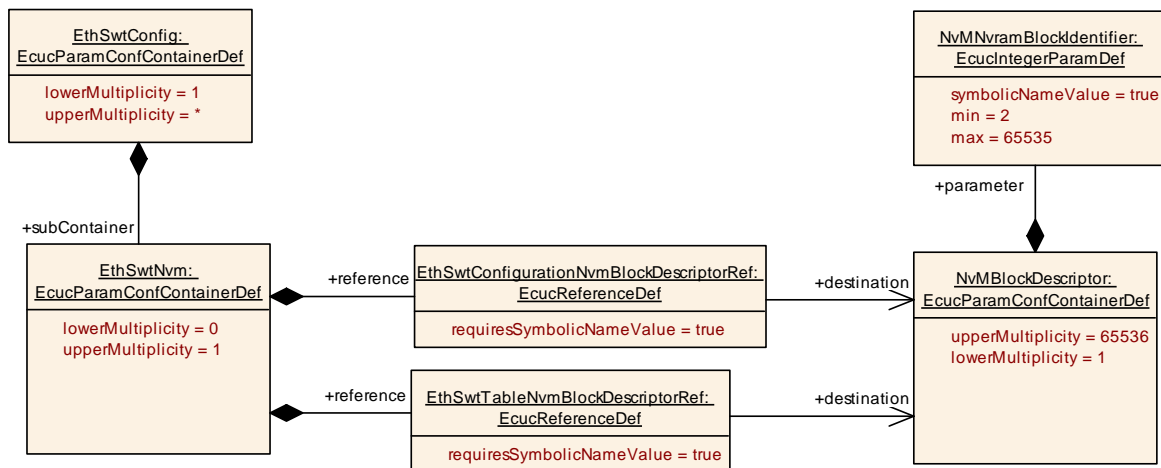


Figure 16: EthSwt NvM interaction



## 10.2 Constraints

[SWS\_EthSwt\_CONSTR\_00413] DRAFT | The module will operate as an independent instance in each of the partitions (see `ECUC_EthSwt_00129` : ), means the called API will only target the partition it is called in.  
| ()

[SWS\_EthSwt\_CONSTR\_00411] DRAFT | The ECUC partitions referenced by `EthSwtConfigEcucPartitionRef` shall be a subset of the ECUC partitions referenced by `EthSwtEcucPartitionRef`. | ()

[SWS\_EthSwt\_CONSTR\_00412] DRAFT | `EthSwtConfig`, `EthCtrlConfig` and `EthTrcvConfig` of one communication channel shall all reference the same ECUC partition. | ()