Document Title	Specification of Module Wireless Ethernet Transceiver Driver
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	799
Document Classification	Standard

Document Status	Final
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	4.3.0

Document Change History

Date	Release	Changed by	Change Description
2016-11-30	4.3.0	AUTOSAR Release Management	• Initial Release

Disclaimer

This specification 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 specification.

The material contained in this specification is protected by copyright and other types of Intellectual Property Rights. The commercial exploitation of the material contained in this specification requires a license to such Intellectual Property Rights.

This specification 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 specification may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The AUTOSAR specifications have been developed for automotive applications only. They have neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Advice for users

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

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

Table of Contents

1 Introduction and functional overview

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

In the AUTOSAR Layered Software Architecture, the Wireless Ethernet Transceiver driver belongs to the *Microcontroller Abstraction Layer*, or more precisely, to the *Communication Drivers*.

This indicates the main task of the Wireless Ethernet Transceiver driver:
Provide to the upper layer (Ethernet Interface) a hardware independent interface comprising multiple equal transceivers. This interface shall be uniform for all transceivers. Thus, the upper layer (Ethernet Interface) may access the underlying bus system in a uniform manner. The interface provides functionality for initialization, configuration and data transmission. The configuration of the Wireless Ethernet Transceiver driver however is bus specific, since it takes into account the specific features of the communication controller.

A single Wireless Ethernet Transceiver driver module supports only one type of transceiver hardware. The Wireless Ethernet Transceiver driver's prefix requires a unique namespace. The Ethernet Interface can access different Wireless Ethernet controller types using different Wireless Ethernet Transceiver drivers using this prefix. The decision which driver to use to access a particular transceiver is a configuration parameter of the Ethernet Interface.

Figure 1.1 depicts the lower part of the Wireless Ethernet stack. One Ethernet Interface can access several transceivers using several Wireless Ethernet Transceiver drivers. Each transceiver may support multiple radio configurations.

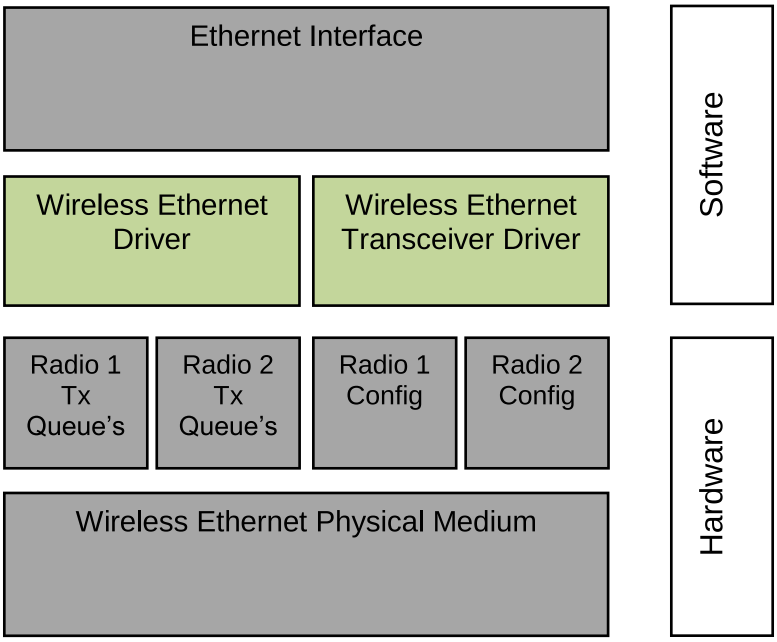

Figure 1.1: Wireless Ethernet module overview

2 Acronyms and abbreviations

Abbreviation / Acronym:	***Description:***
AIFS	Arbitration Inter Frame Space
CBR	Channel Busy Ratio
CIT	Channel Idle Time
CW	Contention Window
DP	DCC Profile
EthIf	Ethernet Interface (AUTOSAR BSW module)
Eth	Ethernet Driver (AUTOSAR BSW module)
EthTrcv	Ethernet Transceiver Driver (AUTOSAR BSW module)
ISR	Interrupt Service Routine
MCG	Module Configuration Generator
WEth	Wireless Ethernet Driver (AUTOSAR BSW module)
WEthTrcv	Wireless Ethernet Transceiver (AUTOSAR BSW module)

3 Related documentation

3.1 Input documents

[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] Specification of Communication
AUTOSAR_SWS_COM.pdf

[5] Specification of Ethernet Interface
AUTOSAR_SWS_EthernetInterface.pdf

[6] Specification of Wireless Ethernet Driver
AUTOSAR_SWS_WirelessEthernetDriver.pdf

[7] Specification of Ethernet Transceiver Driver
AUTOSAR_SWS_EthernetTransceiverDriver.pdf

[8] BSW Scheduler Specification
AUTOSAR_SWS_Scheduler.pdf

[9] Specification of ECU Configuration
AUTOSAR_TPS_ECUConfiguration.pdf

[10] Specification of Memory Mapping
AUTOSAR_SWS_MemoryMapping.pdf

[11] Specification of Standard Types
AUTOSAR_SWS_StandardTypes.pdf

[12] Specification of Default Error Tracer
AUTOSAR_SWS_DefaultErrorTracer.pdf

[13] Specification of Diagnostics Event Manager
AUTOSAR_SWS_DiagnosticEventManager.pdf

[14] Requirements on Vehicle-2-X communication
AUTOSAR_SRS_V2XCommunication.pdf

3.2 Related standards and norms

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

[16] IEEE 802.11-2012

3.3 Related specification

AUTOSAR provides a General Specification on Basic Software (SWS BSW General) [3] which is also valid for Wireless Ethernet Transceiver.

Thus, the specification SWS BSW General [3] shall be considered as additional and required specification for Wireless Ethernet Transceiver.

Furthermore, this document uses the Ethernet Transceiver Driver as a base for the requirements, APIs and configuration, because the wired and the wireless use case have many things (but not all) in common. The term “Ethernet Transceiver Driver“ as used in this document describes the class of Ethernet drivers regardless of the used physical layer and means Wireless as well as Wired Ethernet Transceiver Drivers.

4 Constraints and assumptions

4.1 Limitations

No limitations.

4.2 Applicability to car domains

The Wireless Ethernet Driver is intended to be used for wireless access of customer hardware (Access Point mode) and for wireless access of Vehicle-2-X (V2X) applications / BSW Modules (using a meshed network).

5 Dependencies to other modules

This chapter lists the modules interacting with the Wireless Ethernet Transceiver Driver module.

Modules that use Wireless Ethernet Transceiver Driver module:

- Ethernet Interface (EthIf)

Modules used by the Wireless Ethernet Transceiver Driver module:

- Wireless Ethernet Controller Driver (WEth) for transceiver access via an transceiver dependent interface
- Typically the transceiver hardware is an external device that is accessed by an existing communication driver such as SPI.

5.1 File structure

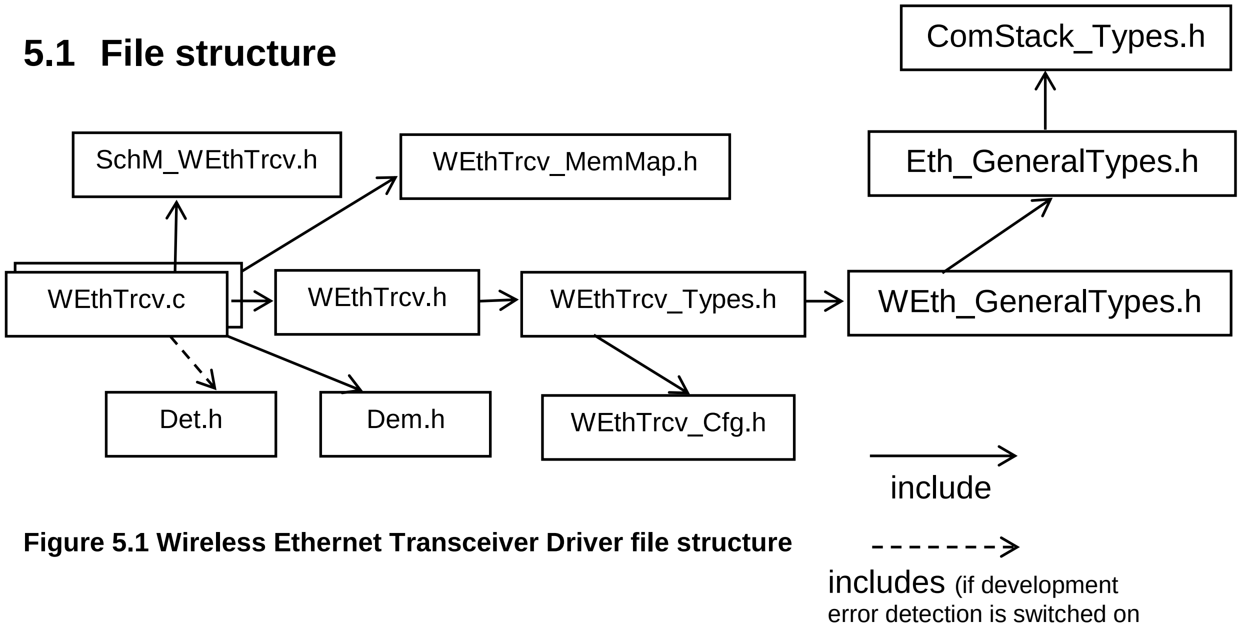

Figure 5.1 Wireless Ethernet Transceiver Driver file structure

[SWS_WEthTrcv_10072] ⌈
WEthTrcv.h shall include EthTrcv_GeneralTypes.h for the inclusion of general EthTrcv type declarations. ⌋()

[SWS_WEthTrcv_10006] ⌈
WEthTrcv.h shall include WEth_GeneralTypes.h for the inclusion of general WEth and WEthTrcv type declarations. ⌋()

[SWS_WEthTrcv_00096] ⌈
The types specified in SWS_WirelessEthernetTransceiverDriver shall be declared in WEth_GeneralTypes.h. ⌋()

6 Requirements traceability

Note:
Requirement IDs within this document have an encoding to state where each requirement has its origin:

- SWS items starting with a leading 0 (SWS_WEth_0xxxx) are inherited from the SWS Ethernet Driver [7].
- SWS items starting with a leading 1 (SWS_WEth_1xxxx) are module specific and not inherited.
- SWS items starting with a leading 2 (SWS_WEth_2xxxx) are inherited from C2C-CC Basic System Profile

Requirement	Description	Satisfied by
SRS_V2X_00010	The implementation of the V2X system shall follow additional guidance given by C2C-CC requirements	SWS_WEthTrcv_20226, SWS_WEthTrcv_20244
SRS_V2X_00245	The V2X system shall support per-packet transmission power control	SWS_WEthTrcv_20246
SRS_V2X_00451	The V2X system's access layer shall be compliant to the ETSI Harmonized Channel Specifications	SWS_WEthTrcv_10071

7 Functional specification

The Wireless Ethernet Transceiver driver sets up the radio for wireless communications.

7.1 Wireless Ethernet BSW stack

As part of the AUTOSAR Layered Software Architecture (see Figure 1.1), the Wireless Ethernet BSW modules also form a layered software stack. The Ethernet Interface module accesses several transeivers using the Wireless Ethernet tranceiver Driver layer, which can be made up of several Wireless Ethernet Transceiver Drivers modules.

7.1.1 Indexing scheme

Users of the Wireless Ethernet Driver identify controller resources using an indexing scheme as described in the Ethernet Transceiver Driver, [7].

[SWS_WEthTrcv_00003] ⌈
The Wireless Ethernet Transceiver Driver is using a zero-based index to abstract the access for upper software layers. The parameter WEthTrcvId within configuration corresponds to parameter TrcvId used in the API. ⌋()

[SWS_WEthTrcv_10001] ⌈
The Wireless Ethernet Transceiver Driver is using a zero-based index to abstract the access to Radios for upper software layers. The parameter WEthTrcvRadioId within configuration corresponds to parameter RadioId used in the API. ⌋()

7.1.2 Requirements

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

[SWS_WEthTrcv_00007] ⌈
In case development error detection is enabled for the Wireless Ethernet Transceiver Driver module: The Wireless Ethernet Transceiver Driver module shall check API parameters for validity and report detected errors to the DET. ⌋()

DET API functions are specified in [12].

7.1.3 Transceiver Parameters

[SWS_WEthTrcv_10026] ⌈

The function WEthTrcv_SetRadioParams shall set properties of type WEthTrcv_SetRadioParamIdType to the access layer of a specific wireless radio indexed by RadioId. ⌋()

[SWS_WEthTrcv_10039] ⌈
The function WEthTrcv_SetChanRxParams shall set properties of type WEthTrcv_SetChanRxParamIdType to a specific wireless channel within a wireless radio indexed by RadioId. ⌋()

[SWS_WEthTrcv_10041] ⌈
The function WEthTrcv_SetChanTxParams shall set of type WEthTrcv_SetChanTxParamIdType to a specific wireless channel within a wireless radio indexed by RadioId. ⌋()

[SWS_WEthTrcv_10049] ⌈
The function EthTrcv_GetChanRxParams shall provide properties of type WEthTrcv_GetChanRxParamIdType of a specific wireless channel within a wireless radio indexed by RadioId. ⌋()

7.1.4 Key/Value Parameter Mapping

[SWS_WEthTrcv_10066] ⌈
For unique reference to transmission and reception parameters, unique enumeration IDs shall be used within this module.
⌋()

[SWS_WEthTrcv_10058] ⌈
Functions using the type WEthTrcv_SetRadioParamIdType shall use a generic list of uint32 values for the list of corresponding values.
⌋()

[SWS_WEthTrcv_10059] ⌈
Functions using the WEthTrcv_SetRadioParamIdType shall use the following type mapping for the corresponding values:

ParamId	*ParamValue Type*
WETHTRCV_SETRADIOPID_SEL_TRCV_CHCFG	uint8
WETHTRCV_SETRADIOPID_SET_CHCFGID	uint8
WETHTRCV_SETRADIOPID_TOLLINGZONE_INFO	uint8

⌋()

[SWS_WEthTrcv_10060] ⌈
Functions using the type WEthTrcv_SetChanRxParamIdType shall use a generic list of uint32 values for the list of corresponding values.
⌋()

[SWS_WEthTrcv_10061] ⌈
Functions using the WEthTrcv_SetChanRxParamIdType shall use the following type mapping for the corresponding values:

ParamId	*ParamValue Type*
WETHTRCV_SETCHRXPID_BITRATE	uint8
WETHTRCV_SETCHRXPID_BANDWIDTH	WEthTrcv_BandwidthType
WETHTRCV_SETCHRXPID_FREQ	uint16
WETHTRCV_SETCHRXPID_CSPWRTRESH	WEthTrcv_RssiType
WETHTRCV_SETCHRXPID_RADIO_MODE	WEthTrcv_RadioModeType
WETHTRCV_SETCHRXPID_ANTENNA	uint8

⌋()

[SWS_WEthTrcv_10062] ⌈
Functions using the type WEthTrcv_SetChanTxParamIdType shall use a generic list of uint32 values for the list of corresponding values.
⌋()

[SWS_WEthTrcv_10063] ⌈
Functions using the WEthTrcv_SetChanTxParamIdType shall use the following type mapping for the corresponding values:

ParamId	*ParamValue Type*
WETHTRCV_SETCHTXPID_BITRATE	uint8
WETHTRCV_SETCHTXPID_BANDWIDTH	WEthTrcv_BandwidthType
WETHTRCV_SETCHTXPID_TXPOWER	WEthTrcv_TxPwrLvlType
WETHTRCV_SETCHTXPID_DCC_CBR	uint8
WETHTRCV_SETCHTXPID_TXQSEL	uint8
WETHTRCV_SETCHTXPID_TXQCFG_AIFSN	uint8
WETHTRCV_SETCHTXPID_TXQCFG_CWMIN	uint8
WETHTRCV_SETCHTXPID_TXQCFG_CWMAX	uint16
WETHTRCV_SETCHTXPID_TXQCFG_TXOP	uint8
WETHTRCV_SETCHTXPID_RADIO_MODE	WEthTrcv_RadioModeType
WETHTRCV_SETCHTXPID_ANTENNA	uint8
WETHTRCV_SETCHTXPID_PACKET_INTERVAL	uint16
WETHTRCV_SETCHTXPID_DCC_STATE	uint8

⌋()

[SWS_WEthTrcv_10064] ⌈
Functions using the type WEthTrcv_GetChanRxParamIdType shall use a generic list of uint32 values for the list of corresponding values.
⌋()

[SWS_WEthTrcv_10065] ⌈
Functions using the WEthTrcv_GetChanRxParamIdType shall use the following type mapping for the corresponding values:

ParamId	*ParamValue Type*
WETHTRCV_GETCHRXPID_CBR	uint8
WETHTRCV_GETCHRXPID_CIT	uint16

⌋()

7.1.5 MainFunction

[SWS_WEthTrcv_10057] ⌈
The MainFunction is used for hardware / software implementation specific execution of cyclic tasks. In case of V2X the MainFunction is used to trigger queue transmission via WEth_TriggerPriorityQueueTransmit and to get Information of the current channel status (CBR). ⌋()

7.1.6 V2X Specific Transceiver Requirements

[SWS_WEthTrcv_10071] ⌈
The following requirements are only valid for WEth Transceivers used within the V2X Communication Stack [14].
⌋ (SRS_V2X_00451)

[SWS_WEthTrcv_20226] ⌈
RF output power of the WEthTrcv module shall be adjustable. ⌋ (SRS_V2X_00010)

[SWS_WEthTrcv_20244] ⌈
The WEthTrcv module shall abide by the following maximum message rates:

- For the relaxed state: the sum of all messages sent on DP1, DP2 and DP3 while in relaxed state shall not surpass $R_{max\_relaxed}$ = 16.7 messages per second. Message bursts are allowed for DP0 with R_{Burst} = 20 messages per second, with a maximum duration of T_{Burst} = 1 seconds, and may only take place every $T_{BurstPeriod}$ = 10 seconds. Thus, adding DP0 messages, the maximum message rate amounts to $R_{max\_relaxed}$ = 36.7 messages per second.

⌋ (SRS_V2X_00010)

[SWS_WEthTrcv_20246] ⌈
The WEthTrcv module shall reduce its transmission power to P_{Toll} = 10 dBm as soon as the protected communication zone is entered, and without changing any other DCC transmission parameters. DP0 messages are excluded from this restriction.⌋
(SRS_V2X_00245)

7.1.7 Wake-up support

There is currently no efficient concept for technologies like Wake on Wireless LAN. Wireless Wake-up is therefore not supported.

7.2 Error classification

7.2.1 Development Errors

[SWS_WEthTrcv_00017] ⌈

Type of error	***Related error code***	***Value [hex]***
Invalid transceiver index	WETHTRCV_E_INV_TRCV_ID	0x01
WEthTrcv module was not initialized	WETHTRCV_E_NOT_INITIALIZED	0x02
Invalid pointer in parameter list	WETHTRCV_E_PARAM_POINTER	0x03

⌋()

7.2.2 Runtime Errors

There are no runtime errors.

7.2.3 Transient Faults

There are no transient faults.

7.2.4 Production Errors

There are no production errors.

7.2.5 Extended Production Errors

Extended production errors are handled as events of the Diagnostic Event Manager. The event IDs are defined in the following tables, while the actual values are assigned externally by the configuration of the Diagnostic Event Manager, and are included in the module via Dem.h.

[SWS_WEthTrcv_00105] ⌈

Error Name:	WETHTRCV_E_ACCESS	
Short Description:	Wireless Ethernet Transceiver Access Failure.	
Long Description:	Monitors the access to the Wireless Ethernet Transceiver.	
Detection Criteria:	Fail	When access to the Wireless Ethernet Transceiver fails the module shall report the extended production error with event status DEM_EVENT_STATUS_PREFAILED to DEM.
	Pass	When access to the Wireless Ethernet Transceiver succeds the module shall report the extended production error with event status DEM_EVENT_STATUS_PREPASSED to DEM.
Secondary Parameters:	None.	
Time Required:	None.	
Monitor Frequency	None.	

⌋()

8 API specification

8.1 Imported types

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

[SWS_WEthTrcv_00027] ⌈

Module	*Imported Type*
Dem	Dem_EventIdType
	Dem_EventStatusType
Eth_GeneralTypes	EthTrcv_LinkStateType
	EthTrcv_ModeType
Std_Types	Std_ReturnType
	Std_VersionInfoType
WEth_GeneralTypes	WEthTrcv_ConfigType
	WEthTrcv_GetChanRxParamIdType
	WEthTrcv_SetChanRxParamIdType
	WEthTrcv_SetChanTxParamIdType
	WEthTrcv_SetRadioParamIdType

⌋ ()

8.2 Type definitions

8.2.1 WEthTrcv_ConfigType

[SWS_WEthTrcv_00098] ⌈

Name:	WEthTrcv_ConfigType
Type:	Structure
Range:	Implementation specific.
Description:	Implementation specific structure of the post build configuration

⌋ ()

8.2.2 WEthTrcv_SetRadioParamIdType

[SWS_WEthTrcv_10008] ⌈

Name:	WEthTrcv_SetRadioParamIdType		
Type:	Enumeration		
Range:	WETHTRCV_SETRADIOPID_SEL_TRCV_CHCFG	0x01	Select which channel config within the transceiver should be set within multichannel context mode
	WETHTRCV_SETRADIOPID_SET_CHCFGID	0x02	Switch to another channel config in single channel context mode or in multichannel context mode (requires previous selection of channel

			config in transceiver)
	WETHTRCV_SETRADIOPID_TOLLINGZONE_INFO	0x03	Information of entering and leaving a Tolling Zone Area
Description:	Wireless radio settings for the transceiver		

⌋ ()

8.2.3 WEthTrcv_SetChanRxParamIdType

[SWS_WEthTrcv_10009] ⌈

Name:	WEthTrcv_SetChanRxParamIdType		
Type:	Enumeration		
Range:	WETHTRCV_SETCHRXPID_BITRATE	0x00	Bitrate
	WETHTRCV_SETCHRXPID_BANDWIDTH	0x01	Bandwidth
	WETHTRCV_SETCHRXPID_FREQ	0x02	Center frequency of a channel
	WETHTRCV_SETCHRXPID_CSPWRTRESH	0x03	Parameter for Rx busy detection
	WETHTRCV_SETCHRXPID_RADIO_MODE	0x04	Param for Rx Radio Mode
	WETHTRCV_SETCHRXPID_ANTENNA	0x05	Rx Antenna Id
Description:	Wireless channel settings for the receive side		

⌋ ()

8.2.4 WEthTrcv_SetChanTxParamIdType

[SWS_WEthTrcv_10011] ⌈

Name:	WEthTrcv_SetChanTxParamIdType		
Type:	Enumeration		
Range:	WETHTRCV_SETCHTXPID_BITRATE	0x00	Bitrate
	WETHTRCV_SETCHTXPID_BANDWIDTH	0x01	Bandwidth
	WETHTRCV_SETCHTXPID_TXPOWER	0x02	Transmission power
	WETHTRCV_SETCHTXPID_DCC_CBR	0x03	Param for Channel Busy Ratio for DCC
	WETHTRCV_SETCHTXPID_TXQSEL	0x04	Selection of the transmit queue for that the settings should be set
	WETHTRCV_SETCHTXPID_TXQCFG_AIFSN	0x05	Arbitration inter-frame-spacing number (multiplier with value of 0 to 15)
	WETHTRCV_SETCHTXPID_TXQCFG_CWMIN	0x06	Contention window min
	WETHTRCV_SETCHTXPID_TXQCFG_CWMAX	0x07	Contention window max
	WETHTRCV_SETCHTXPID_TXQCFG_TXOP	0x08	TXOP duration limit [µs] divided by 32
	WETHTRCV_SETCHTXPID_RADIO_MODE	0x09	Param for Tx Radio Mode
	WETHTRCV_SETCHTXPID_ANTENNA	0x0A	Tx Antenna Id
	WETHTRCV_SETCHTXPID_PACKET_INTERVAL	0x0C	Packet interval for transmission interspace
	WETHTRCV_SETCHTXPID_DCC_STATE	0x0D	State of DCC state machine
Description:	--		

⌋ ()

8.2.5 WEthTrcv_GetChanRxParamIdType

[SWS_WEthTrcv_10007] ⌈

Name:	WEthTrcv_GetChanRxParamIdType		
Type:	Enumeration		
Range:	WETHTRCV_GETCHRXPID_CBR	0x00	Parameter Id for Channel Busy Ratio
	WETHTRCV_GETCHRXPID_CIT	0x01	Parameter Id for Channel Idle Time
Description:	Wireless channel properties of the receive side		

⌋ ()

8.2.6 WEthTrcv_BandwidthType

[SWS_WEthTrcv_10012] ⌈

Name:	WEthTrcv_BandwidthType		
Type:	uint32		
Range:	0x0000004..0xFFFFFFF	--	Invalid
	WETHTRCV_BW_5MHz	0x00	Indicates 5 MHz
	WETHTRCV_BW_10MHz	0x01	Indicates 10 MHz
	WETHTRCV_BW_20MHz	0x02	Indicates 20 MHz
	WETHTRCV_BW_40MHz	0x03	Indicates 40 MHz
Description:	Bandwidth of a radio channel		

⌋ ()

8.2.7 WEthTrcv_TxPwrLvlType

[SWS_WEthTrcv_10014] ⌈

Name:	WEthTrcv_TxPwrLvlType		
Type:	uint16		
Range:	0..399	--	Valid values of 0.5db with an offset of -100dBm
	400..65535	--	Invalid
Description:	Power of frame, in 0.5 dBm units, raw value 0 equals -100 dBm		

⌋ ()

8.2.8 WEthTrcv_RssiType

[SWS_WEthTrcv_10016] ⌈

Name:	WEthTrcv_RssiType		
Type:	uint16		
Range:	0..399	--	Valid values of 0.5db with an offset of -100dBm
	400..65535	--	Invalid
Description:	Power of frame, in 0.5 dBm units, raw value 0 equals -100 dBm		

⌋ ()

8.2.9 WEthTrcv_RadioModeType

[SWS_WEthTrcv_10018] ⌈

Name:	WEthTrcv_RadioModeType		
Type:	uint32		
Range:	0x00000005..0xFFFFFFFF	--	Invalid

	`WETHTRCV_MODE_OFF`	`0x00`	Radio is off
	`WETHTRCV_MODE_RX`	`0x01`	Receive is on
	`WETHTRCV_MODE_TX`	`0x02`	Transmit is on
	`WETHTRCV_MODE_RX_TX`	`0x03`	Receive and Transmit is on
	`WETHTRCV_MODE_SWITCHED`	`0x04`	Radio channel switching is on
Description:	Radio operation mode with multiple radio channel configurations		

⌋ ()

8.3 Function definitions

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

8.3.1 WEthTrcv_Init

[SWS_WEthTrcv_00028] ⌈

Service name:	WEthTrcv_Init	

Syntax:	`void WEthTrcv_Init(`
	`    const WEthTrcv_ConfigType* CfgPtr`
	`)`	

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

⌋ ()

[SWS_WEthTrcv_10022] ⌈
The function shall behave as EthTrcv_Init in [7], **SWS_EthTrcv_00028**. Instead of ETHTRCV_E_XXX the corresponding development error WETHTRCV_E_XXX shall be used. **SWS_EthTrcv_00115** does not apply. ⌋()

8.3.2 WEthTrcv_SetTransceiverMode

[SWS_WEthTrcv_00042] ⌈

Service name:	WEthTrcv_SetTransceiverMode	

Syntax:	`Std_ReturnType WEthTrcv_SetTransceiverMode(`
	`    uint8 TrcvId,`
	`    EthTrcv_ModeType CtrlMode`
	`)`	

Service ID[hex]:	0x03	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	TrcvId	Index of the transceiver within the context of the Ethernet Transceiver Driver
	CtrlMode	ETHTRCV_MODE_DOWN: disable the transceiver ETHTRCV_MODE_ACTIVE: enable the transceiver
Parameters (inout):	None	

Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Service accepted E_NOT_OK: Service denied
Description:	Enables / disables the indexed transceiver	

⌋ ()

[SWS_WEthTrcv_10023] ⌈
The function shall behave as EthTrcv_SetTransceiverMode in [7], **SWS_EthTrcv_00042**. Instead of EthTrcv_XXX, the corresponding WEthTrcv_XXX functions shall be used. Instead of ETHTRCV_E_YYY the corresponding development error WETHTRCV_E_YYY shall be used. Instead of EthTrcvSetTransceiverModeApi, WEthTrcvSetTransceiverModeApi shall be used. **SWS_EthTrcv_00117** and **SWS_EthTrcv_00118** do not apply. ⌋()

8.3.3 WEthTrcv_GetTransceiverMode

[SWS_WEthTrcv_00048] ⌈

Service name:	WEthTrcv_GetTransceiverMode	

Syntax:	`Std_ReturnType WEthTrcv_GetTransceiverMode(`
	`    uint8 TrcvId,`
	`    EthTrcv_ModeType* TrcvModePtr`
	`)`	

Service ID[hex]:	0x04	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	TrcvId	Index of the transceiver within the context of the Wireless Ethernet Transceiver Driver
Parameters (inout):	None	
Parameters (out):	TrcvModePtr	ETHTRCV_MODE_DOWN: the transceiver is disabled ETHTRCV_MODE_ACTIVE: the transceiver is enable
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: transceiver could not be initialized
Description:	Obtains the state of the indexed transceiver	

⌋ ()

[SWS_WEthTrcv_10024] ⌈
The function shall behave as EthTrcv_GetTransceiverMode in [7], **SWS_EthTrcv_00048**. Instead of EthTrcv_Init, the WEthTrcv_Init function shall be used. Instead of ETHTRCV_E_XXX the corresponding development error WETHTRCV_E_XXX shall be used. Instead of EthTrcvGetTransceiverModeApi, WEthTrcvGetTransceiverModeApi shall be used. ⌋()

8.3.4 WEthTrcv_GetLinkState

[SWS_WEthTrcv_00061] ⌈

Service name:	WEthTrcv_GetLinkState

Syntax:	`Std_ReturnType WEthTrcv_GetLinkState(`
	`    uint8 TrcvId,`
	`    EthTrcv_LinkStateType* LinkStatePtr`
	`)`

Service ID[hex]:	0x06

Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	TrcvId	Index of the transceiver within the context of the Ethernet Transceiver Driver
Parameters (inout):	None	
Parameters (out):	LinkStatePtr	ETHTRCV_LINK_STATE_DOWN: transceiver is disconnected ETHTRCV_LINK_STATE_ACTIVE: transceiver is connected
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: transceiver could not be initialized
Description:	Obtains the link state of the indexed transceiver	

⌋ ()

[SWS_WEthTrcv_10073] ⌈
The function shall behave as EthTrcv_GetLinkState in [7], **SWS_EthTrcv_00061**. Instead of EthTrcv_Init, the WEthTrcv_Init function shall be used. Instead of ETHTRCV_E_XXX the corresponding development error WETHTRCV_E_XXX shall be used. Instead of EthTrcvGetLinkStateApi, WEthTrcvGetLinkStateApi shall be used. ⌋()

8.3.5 WEthTrcv_SetRadioParams

[SWS_WEthTrcv_10025] ⌈

Service name:	WEthTrcv_SetRadioParams	

Syntax:	`Std_ReturnType WEthTrcv_SetRadioParams(`
	`uint8 TrcvId,`
	`const WEthTrcv_SetRadioParamIdType* ParamIds,`
	`const uint32* ParamValue,`
	`uint8 NumParams`
	`)`	

Service ID[hex]:	0x30	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	TrcvId	Index of the transceiver
	ParamIds	IDs of the Parameters to set
	ParamValue	Values of the Parameters to set
	NumParams	Number of Parameters to set
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: failed writing parameters
Description:	Set values related to a transceiver's wireless radio. For example, this could be the selection of the radio settings (channel, ...).	

⌋ ()

[SWS_WEthTrcv_10067] ⌈
The function shall use the type mapping from **SWS_WEthTrcv_10059** for the ParamIds and ParamValues parameters. ⌋()

[SWS_WEthTrcv_10027] ⌈

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

[SWS_WEthTrcv_10028] ⌈
If development error detection is enabled: the function shall check the parameter TrcvId for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_INV_TRCV_ID otherwise (if DET is disabled) return E_NOT_OK. ⌋()

[SWS_WEthTrcv_10029] ⌈
If development error detection is enabled: the function shall check the parameter RadioId for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_INV_PARAM otherwise (if DET is disabled) return E_NOT_OK. ⌋()

[SWS_WEthTrcv_10030] ⌈
If development error detection is enabled: the function shall check the parameter ParamIds for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_PARAM_POINTER. ⌋()

[SWS_WEthTrcv_10031] ⌈
If development error detection is enabled: the function shall check the parameter ParamValues for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_PARAM_POINTER. ⌋()

8.3.6 WEthTrcv_SetChanRxParams

[SWS_WEthTrcv_10033] ⌈

Service name:	WEthTrcv_SetChanRxParams	

Syntax:	`Std_ReturnType WEthTrcv_SetChanRxParams(`
	`    uint8 TrcvId,`
	`    uint8 RadioId,`
	`    const WEthTrcv_SetChanRxParamIdType* ParamIds,`
	`    const uint32* ParamValues,`
	`    uint8 NumParams`
	`)`	

Service ID[hex]:	0x31	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	TrcvId	Index of the transceiver
	RadioId	Index of the Transceiver's Radio (including channel)
	ParamIds	IDs of the Parameters to set
	ParamValues	Values of the Parameters to set
	NumParams	Number of Parameters to set
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: success E_NOT_OK: failed writing parameters
Description:	Set values related to the receive direction of a transceiver's wireless channel.For example, this could be a channel parameter like the frequency.	

⌋ ()

[SWS_WEthTrcv_10068] ⌈
The function shall use the type mapping from **SWS_WEthTrcv_10061** for the ParamIds and ParamValues parameters. ⌋()

[SWS_WEthTrcv_10034] ⌈
If development error detection is enabled: the function shall check that the service WEthTrcv_Init was previously called. If the check fails, the function shall raise the development error WETHTRCV_E_NOT_INITIALIZED. ⌋()

[SWS_WEthTrcv_10035] ⌈
If development error detection is enabled: the function shall check the parameter TrcvId for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_INV_TRCV_ID otherwise (if DET is disabled) return E_NOT_OK. ⌋()

[SWS_WEthTrcv_10036] ⌈
If development error detection is enabled: the function shall check the parameter RadioId for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_INV_PARAM otherwise (if DET is disabled) return E_NOT_OK. ⌋()

[SWS_WEthTrcv_10037] ⌈
If development error detection is enabled: the function shall check the parameter ParamIds for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_PARAM_POINTER. ⌋()

[SWS_WEthTrcv_10038] ⌈
If development error detection is enabled: the function shall check the parameter ParamValues for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_PARAM_POINTER. ⌋()

8.3.7 WEthTrcv_SetChanTxParams

[SWS_WEthTrcv_10040] ⌈

Service name:	WEthTrcv_SetChanTxParams	

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

Service ID[hex]:	0x32	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	TrcvId	Index of the transceiver
	RadioId	Index of the Transceiver's Radio (including channel)
	TxParamIds	IDs of the Parameters to set
	ParamValues	Values of the Parameters to set
	NumParams	Number of Parameters to set
Parameters	None	

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

⌋ ()

[SWS_WEthTrcv_10069] ⌈
The function shall use the type mapping from **SWS_WEthTrcv_10063** for the TxParamIds and ParamValues parameters. ⌋()

[SWS_WEthTrcv_10042] ⌈
If development error detection is enabled: the function shall check that the service WEthTrcv_Init was previously called. If the check fails, the function shall raise the development error WETHTRCV_E_NOT_INITIALIZED. ⌋()

[SWS_WEthTrcv_10043] ⌈
If development error detection is enabled: the function shall check the parameter TrcvId for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_INV_TRCV_ID otherwise (if DET is disabled) return E_NOT_OK. ⌋()

[SWS_WEthTrcv_10044] ⌈
If development error detection is enabled: the function shall check the parameter RadioId for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_INV_PARAM otherwise (if DET is disabled) return E_NOT_OK. ⌋()

[SWS_WEthTrcv_10045] ⌈
If development error detection is enabled: the function shall check the parameter TxParamIds for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_PARAM_POINTER. ⌋()

[SWS_WEthTrcv_10046] ⌈
If development error detection is enabled: the function shall check the parameter ParamValues for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_PARAM_POINTER. ⌋()

8.3.8 WEthTrcv_GetChanRxParams

[SWS_WEthTrcv_10048] ⌈

Service name:	WEthTrcv_GetChanRxParams

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

Service ID[hex]:	0x33
Sync/Async:	Synchronous

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

⌋ ()

[SWS_WEthTrcv_10070] ⌈
The function shall use the type mapping from **SWS_WEthTrcv_10065** for the ParamIds and ParamValues parameters. ⌋()

[SWS_WEthTrcv_10050] ⌈
If development error detection is enabled: the function shall check that the service WEthTrcv_Init was previously called. If the check fails, the function shall raise the development error WETHTRCV_E_NOT_INITIALIZED. ⌋()

[SWS_WEthTrcv_10051] ⌈
If development error detection is enabled: the function shall check the parameter TrcvId for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_INV_TRCV_ID otherwise (if DET is disabled) return E_NOT_OK. ⌋()

[SWS_WEthTrcv_10052] ⌈
If development error detection is enabled: the function shall check the parameter RadioId for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_INV_PARAM otherwise (if DET is disabled) return E_NOT_OK. ⌋()

[SWS_WEthTrcv_10053] ⌈
If development error detection is enabled: the function shall check the parameter ParamIds for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_PARAM_POINTER. ⌋()

[SWS_WEthTrcv_10054] ⌈
If development error detection is enabled: the function shall check the parameter ParamValues for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_PARAM_POINTER. ⌋()

8.3.9 WEthTrcv_GetVersionInfo

[SWS_WEthTrcv_00082] ⌈

Service name:	WEthTrcv_GetVersionInfo

Syntax:	`void WEthTrcv_GetVersionInfo(`
	`    Std_VersionInfoType* VersionInfoPtr`
	`)`

Service ID[hex]:	0x0b	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	VersionInfoPtr	Version information of this module
Return value:	None	
Description:	Returns the version information of this module	

⌋ ()

[SWS_WEthTrcv_00093] ⌈
If development error detection is enabled: the function shall check the parameter VersionInfoPtr for being valid. If the check fails, the function shall raise the development error WETHTRCV_E_PARAM_POINTER. ⌋()

8.4 Call-back notifications

The Wireless Ethernet Transceiver Driver does not provide any callback functions.

8.5 Interrupt service routines

The Wireless Ethernet Transceiver Driver does not provide any interrupt service routines.

8.6 Scheduled functions

8.6.1 WEthTrcv_MainFunction

[SWS_WEthTrcv_00106] ⌈

Service name:	WEthTrcv_MainFunction

Syntax:	`void WEthTrcv_MainFunction(`
	`    void`
	`)`

Service ID[hex]:	0x0c
Description:	Used for polling state changes. Calls EthIf_TrcvModeIndication when the transceiver mode changed.

⌋ ()

8.7 Expected Interfaces

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

8.7.1 Mandatory Interfaces

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

[SWS_WEthTrcv_00085] ⌈

API function	***Description***
Dem_SetEventStatus	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.
EthIf_TrcvModeIndication	Called asynchronously when mode has been read out. Triggered by previous Eth_SetTransceiverMode call. Can directly be called within the trigger functions.
SchM_Enter_WEthTrcv	Invokes the SchM_Enter function to enter a module local exclusive area.
SchM_Exit_WEthTrcv	Invokes the SchM_Exit function to exit an exclusive area.
WEth_ReadTrcvRegs	Reads a transceiver register
WEth_WriteTrcvRegs	Configures a transceivers registers or triggers a function offered by the receiver

⌋ ()

8.7.2 Optional Interfaces

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

[SWS_WEthTrcv_00120] ⌈

API function	***Description***
Det_ReportError	Service to report development errors.

⌋ ()

8.7.3 Configurable interfaces

The Wireless Ethernet Transceiver Driver does not use configurable interfaces.

9 Sequence diagrams

The Wireless Ethernet Transceiver driver will interact with Ethernet Interface in the same way as the Ethernet Transceiver driver, see sequence diagrams in [5]. Note: there is no Link State Change event in Wireless Ethernet Transceiver driver.

10 Configuration specification

Chapter 10.1 specifies the structure (containers) and the parameters of the module Wireless Ethernet Transceiver Driver.

Chapter 10.2 specifies additionally published information of the module Wireless Ethernet Transceiver Driver.

10.1 Containers and configuration parameters

The following chapters summarize all configuration parameters.

10.1.1 WEthTrcv

SWS Item	**ECUC_WEthTrcv_10023 :**
Module Name	*WEthTrcv*
Module Description	Configuration of Ethernet Transceiver Driver module
Post-Build Variant Support	true
Supported Config Variants	VARIANT-LINK-TIME, VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

Included Containers		
Container Name	*Multiplicity*	*Scope / Dependency*
WEthTrcvAntennaConfigSet	1	This container contains the antenna configurations.
WEthTrcvConfigSet	1	This container contains the configuration parameters and sub containers of the AUTOSAR WEthTrcv module.
WEthTrcvGeneral	1	General configuration of Wireless Ethernet Transceiver Driver module
WEthTrcvRadioConfigSet	1..*	This container contains the radio configurations.

10.1.2 WEthTrcvConfigSet

SWS Item	**ECUC_WEthTrcv_00016 :**
Container Name	WEthTrcvConfigSet
Description	This container contains the configuration parameters and sub containers of the AUTOSAR WEthTrcv module.
Configuration Parameters	

Included Containers		
Container Name	*Multiplicity*	*Scope / Dependency*
WEthTrcvConfig	1..*	Configuration of the individual transceiver

10.1.3 WEthTrcvConfig

SWS Item	**ECUC_WEthTrcv_00012 :**
Container Name	WEthTrcvConfig
Description	Configuration of the individual transceiver

Configuration Parameters

SWS Item	**ECUC_WEthTrcv_00015 :**		
Name	WEthTrcvBusId		
Description	Specifies the hardware id used for lower level bus interface access (e.g. MII/SPI) to the transceiver's hardware module. For example the MII index if MII would have been used.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
Post-Build Variant Value	true		
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_00013 :**		
Name	WEthTrcvId		
Description	Specifies the instance ID of the configured transceiver.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 255		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_WEthTrcv_00024 :**		
Name	WEthTrcvPhysLayerType		
Description	Specifies the physical layer type of the Wireless Ethernet transceiver link.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	TRCV_PHYS_LAYER_TYPE_80211_P	802.11p physical layer	
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_10022 :**
Name	WEthTrcvCtrlRef
Description	Specifies a reference to the wireless ethernet controller used for lower layer bus interface access to the transceiver.
Multiplicity	1
Type	Symbolic name reference to [WEthCtrlConfig]
Post-Build Variant Value	true

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

SWS Item	**ECUC_WEthTrcv_10001 :**		
Name	WEthTrcvRadioConfigSetRef		
Description	Reference to a WEthTrcvRadioConfigSet.		
Multiplicity	1		
Type	Reference to [WEthTrcvRadioConfigSet]		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency			

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
WEthTrcvDemEventParameterRefs	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.

10.1.4 WEthTrcvDemEventParameterRefs

SWS Item	**ECUC_WEthTrcv_00017 :**
Container Name	WEthTrcvDemEventParameterRefs
Description	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.
Configuration Parameters	

SWS Item	**ECUC_WEthTrcv_00018 :**		
Name	WETHTRCV_E_ACCESS		
Description	Reference to the DemEventParameter which shall be issued when the error "Transceiver access failed" has occurred.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD

Scope / Dependency	scope: local

No Included Containers

10.1.5 WEthTrcvRadioConfigSet

SWS Item	**ECUC_WEthTrcv_10002 :**
Container Name	WEthTrcvRadioConfigSet
Description	This container contains the radio configurations.
Configuration Parameters	

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
WEthTrcvRadioConfig	1..*	Configuration of the individual radio (PHY + MAC).

10.1.6 WEthTrcvRadioConfig

SWS Item	**ECUC_WEthTrcv_10003 :**
Container Name	WEthTrcvRadioConfig
Description	Configuration of the individual radio (PHY + MAC).
Configuration Parameters	

SWS Item	**ECUC_WEthTrcv_10007 :**		
Name	WEthTrcvRadioChannelBandwidth		
Description	Specifies the bandwidth of the physical channel.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	BW_10MHZ	--	
	BW_20MHZ	--	
	BW_40MHZ	--	
	BW_5MHZ	--	
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_10012 :**		
Name	WEthTrcvRadioChannelCsPowerThreshold		
Description	Specifies the threshold for carrier sense (CS) power of the physical channel [dBm].		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[-100 .. 100]		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants

	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_10006 :**		
Name	WEthTrcvRadioChannelFreq		
Description	Specifies the frequency of the physical channel [Hz].		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 18446744073709551615		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_10011 :**		
Name	WEthTrcvRadioChannelMaxTxPower		
Description	Specifies the transmit power of the physical channel [dBm].		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[-100 .. 100]		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_10010 :**		
Name	WEthTrcvRadioChannelTxDatarate		
Description	Specifies the transmit datarate of the physical channel. [bit/s]		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	]0 .. INF[		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_10004 :**		
Name	WEthTrcvRadioId		
Description	Specifies the instance ID of the configured radio.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 255		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_WEthTrcv_10005 :**		
Name	WEthTrcvRadioMode		
Description	Specifies the mode of the radio within a WEthTrcvRadioConfig. Inside of a WEthTrcvRadioConfigSet different modes for the respective WEthTrcvRadioConfigs are possible. The WEthTrcvRadioConfigSet can be selected at runtime.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	OFF	--	
	RX_ON	--	
	RX_TX_ON	--	
	TX_ON	--	
	USED_FOR_CHANNEL_SWITCHING	--	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_10009 :**		
Name	WEthTrcvRadioChannelRxAntenna		
Description	Specifies the antenna used for reception of packets of the physical channel.		
Multiplicity	1		
Type	Reference to [WEthTrcvAntennaConfig]		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_10008 :**		
Name	WEthTrcvRadioChannelTxAntenna		
Description	Specifies the antenna used for transmission of packets to the physical channel.		
Multiplicity	1		
Type	Reference to [WEthTrcvAntennaConfig]		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
WEthTrcvRadioChannelTxQueueConfig	1..*	Configuration of the individual EDCA transmit queue of a channel.

10.1.7 WEthTrcvRadioChannelTxQueueConfig

SWS Item	**ECUC_WEthTrcv_10013 :**

Container Name	WEthTrcvRadioChannelTxQueueConfig		
Description	Configuration of the individual EDCA transmit queue of a channel.		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Configuration Parameters			

SWS Item	**ECUC_WEthTrcv_10015 :**		
Name	WEthTrcvRadioChannelTxQueueAifsn		
Description	Specifies the arbitration inter frame space number (AIFSN) of the queue.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 15		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_10017 :**		
Name	WEthTrcvRadioChannelTxQueueCwMax		
Description	Specifies the maximum size of the contention windows (CW) of the queue.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_10016 :**		
Name	WEthTrcvRadioChannelTxQueueCwMin		
Description	Specifies the minimum size of the contention windows (CW) of the queue.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_10014 :**	
Name	WEthTrcvRadioChannelTxQueueId	
Description	Specifies the ID (equals priority) of the queue.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 255	
Default value	--	

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

SWS Item	**ECUC_WEthTrcv_10018 :**		
Name	WEthTrcvRadioChannelTxQueueTxOpDurationLimit		
Description	Specifies the transmit operation duration limit of the queue in [s].		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[3.2E-5 .. 0.00816]		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

10.1.8 WEthTrcvAntennaConfigSet

SWS Item	**ECUC_WEthTrcv_10019 :**
Container Name	WEthTrcvAntennaConfigSet
Description	This container contains the antenna configurations.
Configuration Parameters	

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
WEthTrcvAntennaConfig	1..*	Configuration of the individual antenna.

10.1.9 WEthTrcvAntennaConfig

SWS Item	**ECUC_WEthTrcv_10020 :**
Container Name	WEthTrcvAntennaConfig
Description	Configuration of the individual antenna.
Configuration Parameters	

SWS Item	**ECUC_WEthTrcv_10021 :**		
Name	WEthTrcvAntennaId		
Description	Specifies the instance ID of the configured antenna.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 255		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	

	Post-build time	--	
Scope / Dependency	scope: ECU		

No Included Containers

10.1.10 WEthTrcvGeneral

SWS Item	**ECUC_WEthTrcv_00001 :**
Container Name	WEthTrcvGeneral
Description	General configuration of Wireless Ethernet Transceiver Driver module
Configuration Parameters	

SWS Item	**ECUC_WEthTrcv_00003 :**		
Name	WEthTrcvDevErrorDetect		
Description	Switches the Default Error Tracer (Det) detection and notification ON or OFF. • true: detection and notification is enabled. • false: detection and notification is disabled.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_00009 :**		
Name	WEthTrcvGetLinkStateApi		
Description	Enables / Disables WEthTrcv_GetLinkState API		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_00007 :**		
Name	WEthTrcvGetTransceiverModeApi		
Description	Enables / Disables WEthTrcv_GetTransceiverMode API		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_00020 :**		
Name	WEthTrcvIndex		
Description	Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_00032 :**		
Name	WEthTrcvMainFunctionPeriod		
Description	Specifies the period of main function WEthTrcv_MainFunction in seconds.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	]0 .. INF[		
Default value	--		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_00002 :**		
Name	WEthTrcvMaxTrcvsSupported		
Description	--		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	1		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_00006 :**		
Name	WEthTrcvSetTransceiverModeApi		
Description	Enables / Disables WEthTrcv_SetTransceiverMode API		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_00004 :**		
Name	WEthTrcvVersionInfoApi		
Description	Enables / Disables version info API		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_WEthTrcv_00005 :**		
Name	WEthTrcvVersionInfoApiMacro		
Description	Enables / Disables version info API macro implementation		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

10.2 Published Information

Additional module-specific published parameters are listed below if applicable.

11 Not applicable requirements