Document Title	Specification of TCP/IP Stack
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	617
Document Classification	Standard
Document Version	1.1.1
Document Status	Final
Part of Release	4.1
Revision	3

Document Change History			
Date	**Version**	**Changed by**	**Change Description**
31.03.2014	1.1.1	AUTOSAR Release Management	• Clarifications and corrections of requirements • Editorial changes
31.10.2013	1.1.0	AUTOSAR Release Management	• Added control functions for ARP • Clarifications and corrections of requirements • Editorial changes • Removed chapter(s) on change documentation
28.02.2013	1.0.0	AUTOSAR Administration	Initial Release

Disclaimer

Advice for users

AUTOSAR Specification Documents 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 Specification Documents for illustration purposes only, and they themselves are not part of the AUTOSAR Standard. Neither their presence in such Specification Documents, 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

The AUTOSAR TCP/IP module offers functionality to send and receive Internet Protocol data.

The TCP/IP Stack (TCPIP) is located between the Socket Adaptor (SoAd) and the Ethernet Interface (EthIf) modules.

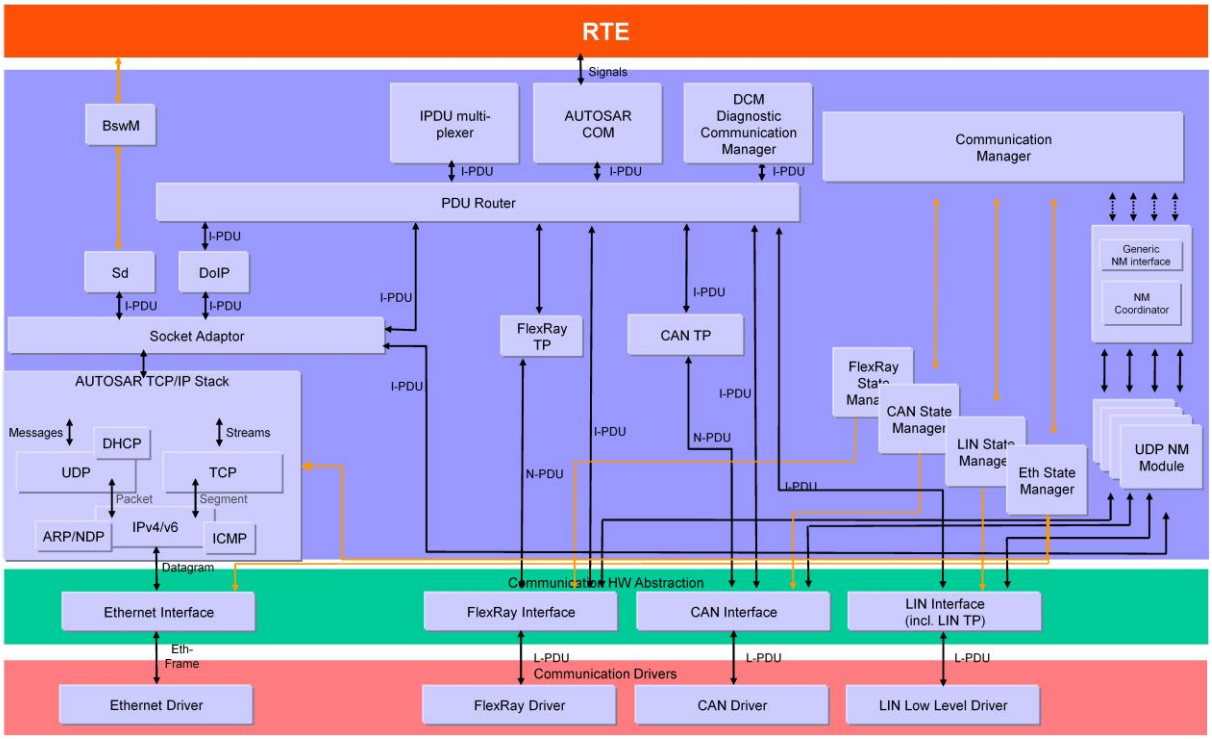


Figure 1: Extended AUTOSAR Communication Stack.

2 Acronyms and abbreviations

Abbreviation / Acronym:	*Description:*
ARP	Address Resolution Protocol
DEM	Diagnostic Event Manager
DET	Development Error Tracer
DHCP	Dynamic Host Configuration Protocol
ECU	Electronic Control Unit
EthIf	Ethernet Interface
EthSM	Ethernet State Manager
HTTP	HyperText Transfer Protocol
IANA	Internet Assigned Numbers Authority
ICMP	Internet Control Message Protocol
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
SoAd	Socket Adaptor
TCP	Transmission Control Protocol
TCP/IP	A family of communication protocols used in computer networks
TP	Transport Protocol
UDP	User Datagram Protocol

3 Related documentation

3.1 Input documents

[1] AUTOSAR Layered Software Architecture
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf

[2] AUTOSAR Basis Software Mode Manager
AUTOSAR_SWS_BSWModeManager.pdf

[3] AUTOSAR Socket Adaptor
AUTOSAR_SWS_SocketAdaptor.pdf

[4] AUTOSAR SRS BSW General
AUTOSAR_SRS_BSWGeneral.pdf

[5] AUTOSAR SRS Ethernet
AUTOSAR_SRS_Ethernet.pdf

[6] AUTOSAR General Specification for Basic Software Modules
AUTOSAR_SWS_BSWGeneral.pdf

[7] Specification of ECU Configuration
AUTOSAR_TPS_ECUConfiguration.pdf

[8] List of Basic Software Modules
AUTOSAR_TR_BSWModuleList.pdf

3.2 Related standards and norms

[9] IETF RFC 3927
http://tools.ietf.org/html/rfc3927

[10] IETF RFC 1122
http://tools.ietf.org/html/rfc1122

[11] IETF RFC 826
http://tools.ietf.org/html/rfc826

[12] IETF RFC 894
http://tools.ietf.org/html/rfc894

[13] IETF RFC 791
http://tools.ietf.org/html/rfc791

[14] IETF RFC 815
http://tools.ietf.org/html/rfc815

[15] IETF RFC 4632
http://tools.ietf.org/html/rfc4632

[16] IETF RFC 1112
http://tools.ietf.org/html/rfc1112

[17] IETF RFC 792
http://tools.ietf.org/html/rfc792

[18] IETF RFC 1191
http://tools.ietf.org/html/rfc1191

[19] IETF RFC 2131
http://tools.ietf.org/html/rfc2131

[20] IETF RFC 768
http://tools.ietf.org/html/rfc768

[21] IETF RFC 793
http://tools.ietf.org/html/rfc793

[22] IETF RFC 813
http://tools.ietf.org/html/rfc813

[23] IETF RFC 896
http://tools.ietf.org/html/rfc896

[24] IETF RFC 5681
http://tools.ietf.org/html/rfc5681

4 Constraints and assumptions

4.1 Limitations

This document does not cover the assignment of UDP or TCP port numbers. There is no reserved space within the IANA assigned number range. Each implementer is responsible for managing the used port numbers.

This document does not cover the management of IP addresses. This might be done dynamically, e.g. by using DHCP, or statically. It is the implementer's responsibility to prevent address conflicts and achieve compliance with IANA address assignments.

This specification does not prescribe a certain physical layer or data rate.

4.2 Applicability to car domains

No restrictions.

5 Dependencies to other modules

5.1 EthIf

The Ethernet Interface is the lower layer module of the TcpIp module.

5.2 EthSM

The Ethernet State Manager controls the communication mode of the TcpIp module by requesting communication modes from the TcpIp module. TcpIp notifies the EthSM about communication mode changes.

5.3 Socket Adaptor

The Socket Adaptor is the upper layer module of the TcpIp module.

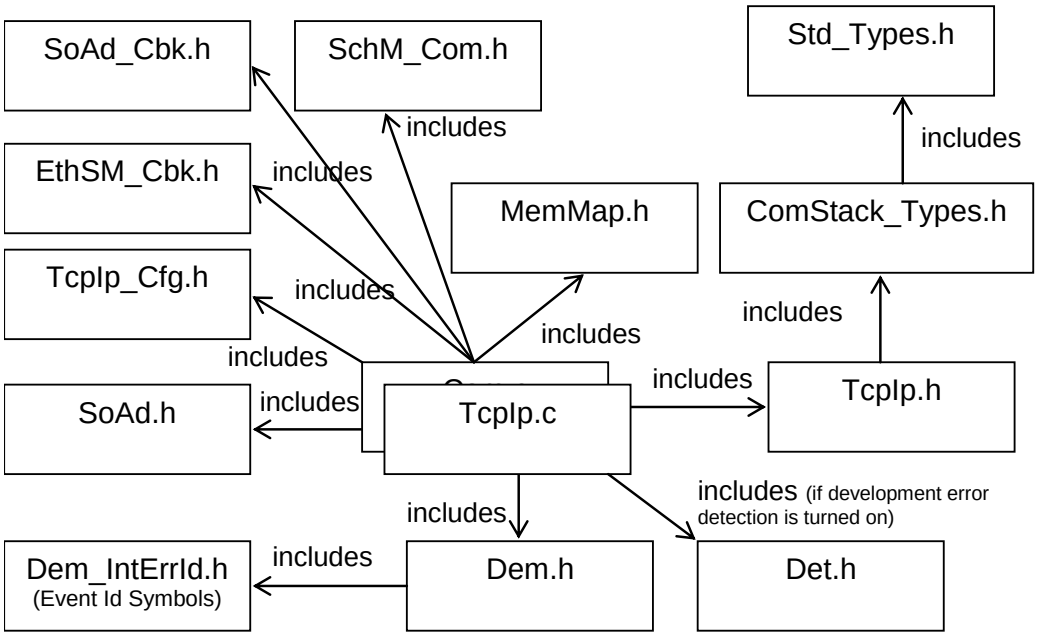
5.4 File structure

5.4.1 Code file structure

For details refer to the chapter 5.1.6 “Code file structure” in *SWS_BSWGeneral.*

5.4.2 Header file structure

This chapter shall contain the h –files especially the h-files which are necessary for configuration. The configuration c-file shall have a naming convention TcpIp_Cfg.h.



[SWS_TCPIP_00068][「The module shall include the Dem.h file. 」()
By this inclusion the APIs to report errors as well as the required Event Id symbols are included. This specification defines the name of the Event Id symbols which are provided by XML to the DEM configuration tool. The DEM configuration tool assigns ECU dependent values to the Event Id symbols and publishes the symbols in Dem_IntErrId.h.

5.5 Version check

For details refer to the chapter 5.1.8 “Version Check” in *SWS_BSWGeneral.*

6 Requirements traceability

Document: AUTOSAR requirements on Basic Software, general [x]

Requirement	***Satisfied by***
[BSW00344] Reference to link--time configuration	
[BSW00404] Reference to post build time configuration	Chapter 5
[BSW00405] Reference to multiple configuration sets	Not applicable
[BSW00345] Configuration at Compile time	
[BSW159] Automatic configuration	Chapter 10
[BSW167] Static configuration checking	Chapter 10
[BSW171] Configurability of optional functionality	Chapter 10
[BSW170] Data for reconfiguration of AUTOSAR SW--Components	Not applicable
[BSW00380] Separate C--Files for configuration parameters	
[BSW00419] Separate C--Files for pre--compile time configuration parameters	
[BSW00381] Separate configuration header file for pre--compile time parameters	
[BSW00412] Separate H--File for configuration parameters	
[BSW00383] List dependencies of configuration files	Chapter 5
[BSW00384] List dependencies to other modules	Chapter 5
[BSW00387] Specify the configuration class of callback function	Not applicable
[BSW00388] Introduce containers	Chapter 10
[BSW00389] Containers shall have names	Chapter 10
[BSW00390] Parameter content shall be unique within the module	Chapter 10
[BSW00391] Parameter shall have unique names	Chapter 10
[BSW00392] Parameters shall have a type	Chapter 10
[BSW00393] Parameters shall have a range	Chapter 10
[BSW00394] Specify the scope of the parameters	Chapter 10
[BSW00395] List the required parameters	Chapter 10

(per parameter)	
[BSW00396] Configuration classes	Chapter 10
[BSW00397] Pre--compile--time parameters	Chapter 10
[BSW00398] Link--time parameters	Chapter 10
[BSW00399] Loadable Post--build time parameters	Not applicable
[BSW00400] Selectable Post--build time parameters	Not applicable
[BSW00438] Post Build Configuration Data Structure	Chapter 10
[BSW00402] Published information	Chapter 10
[BSW101] Initialization interface	
[BSW00406] Check module initialization	
[BSW00407] Function to read out published parameters	
[BSW00423] Usage of SW--C template to describe BSW modules with AUTOSAR Interfaces	Chapter 7.7
[BSW00336] Shutdown interface	Not Applicable
[BSW00337] Classification of errors	Chapter 7.3
[BSW00338] Detection and Reporting of development errors	Chapter 7.4
[BSW00369] Do not return development error codes via API	Chapter 8
[BSW00339] Reporting of production relevant errors and exceptions	Not Applicable
[BSW00323] API parameter checking	Chapter 8
[BSW00409] Header files for production code error IDs	Not Applicable
[BSW00385] List possible error notifications	Chapter 7.3
[BSW00386] Configuration for detecting an error	Chapter 7.4
[BSW00415] User dependent include files	
[BSW00343] Specification and configuration of time	Chapter 10
[BSW00346] Basic set of module files	Chapter 5
[BSW158] Separation of configuration from implementation	Chapter 5
[BSW00370] Separation of callback interface from API	Chapter 8
[BSW00357] Standard API return type	Chapter 8
[BSW00377] Module specific API return types	Chapter 8
[BSW00371] Do not pass function pointers via API	Chapter 8
[BSW00358] Return type of init() functions	Chapter 8

[BSW00414] Parameter of init function	Chapter 8
[BSW00376] Return type and parameters of main processing functions	Chapter 8
[BSW00359] Return type of callback functions	Chapter 8
[BSW00360] Parameters of callback functions	Chapter 8
[BSW00440] Function prototype for callback functions of AUTOSAR Services	Chapter 7.7
[BSW00374] Module vendor identification	Chapter 10
[BSW00379] Module identification	Chapter 10
[BSW003] Version identification	Chapter 10
[BSW00318] Format of module version numbers	Chapter 10
[BSW00321] Enumeration of module version numbers	Chapter 10

Document: AUTOSAR requirements on Basic Software, SRS Ethernet [x]

Requirement	***Satisfied by***
SRS_Eth_00045 TCPIP automatic IP address assignment	
SRS_Eth_00014 TCPIP IPv4 implementation	
SRS_Eth_00066 TCPIP IPv6 implementation	
SRS_Eth_00015 TCPIP ARP implementation	
SRS_Eth_00016 TCPIP ICMP implementation	
SRS_Eth_00017 TCPIP TCP implementation	
SRS_Eth_00018 TCPIP UDP implementation	
SRS_Eth_00019 TCPIP TCP+UDP support	
SRS_Eth_00020 TCPIP DHCP implementation	
SRS_Eth_00021 TCPIP DHCP “host name option” implementation	
SRS_Eth_00022 TCPIP link local IP implementation	
SRS_Eth_00054 TCPIP minimum functionality	
SRS_Eth_00052 Interfaces to the TCP/IP stack	

7 Functional specification

Figure 2 provides an architecture overview of the AUTOSAR TCP/IP stack. The TCP/IP stack consists of the submodules within the red box. Furthermore the interaction with other AUTOSAR modules (beside Dem and Det) is shown.

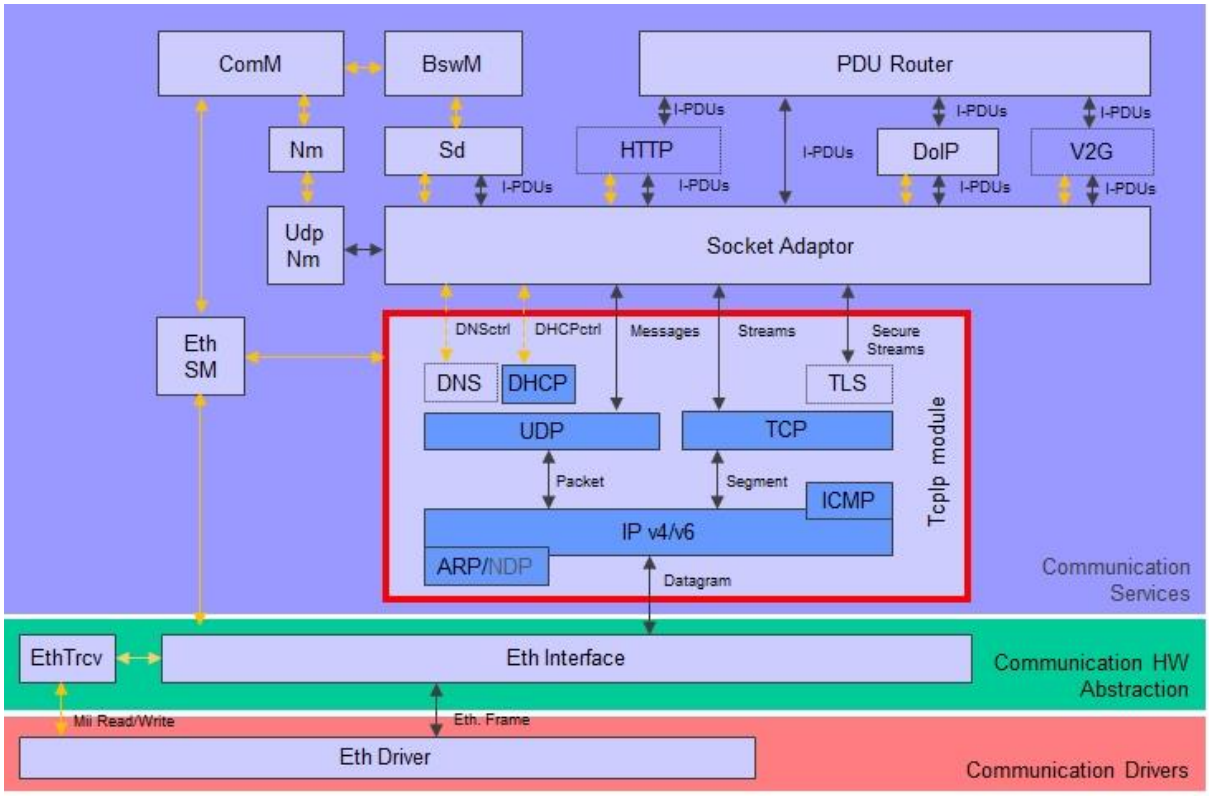


Figure 2: TCP/IP Architeture Overview

7.1 TcpIp protocol family

[SWS_TCPIP_00052]⌈ The TCP/IP stack shall consist of submodules implementing specific functionalities defined in the subchapters below. ⌋()

7.1.1 Address Resolution Protocol (ARP)

[SWS_TCPIP_00056]⌈ The TcpIp shall implement the Address Resolution Protocol (ARP) as defined in IETF RFC 826.⌋()

[SWS_TCPIP_00090]⌈ The TcpIp shall limit the number of ARP table (address resolution cache) entries to the number specified by the configuration parameter TcpIpArpTableSizeMax.⌋()

[SWS_TCPIP_00091]⌈ The TcpIp shall remove entries of the ARP table if they are not used for the timeout specified by the configuration parameter TcpIpArpTableEntryTimeout.⌋()

[SWS_TCPIP_00092]⌈ The TcpIp shall use the information from each received IP packet to update the ARP table in addition to received ARP packets.⌋()

[SWS_TCPIP_00142]⌈ The TcpIp shall call <Up_PhysAddrTableChg>() directly after each ARP table change:
(a) If TcpIp adds a new entry or updates an existing one, the parameter valid shall be set to TRUE and the parameters IpAddrPtr and PhysAddrPtr shall be set according to the new or updated entry.
(b) In case TcpIp removes an entry, valid shall be set to FALSE and the parameters IpAddrPtr and PhysAddrPtr shall be set according to the removed entry.⌋()

[SWS_TCPIP_00093]⌈ On assignment of a new IP address the TcpIp shall send a configurable number (TcpIpArpNumGratuitousARPonStartup) of gratuitous ARP replies according to IETF RFC 2002, section 4.6, second indent.⌋()

7.1.2 Internet Protocol (IP)

The Internet Protocol (IP) is the main protocol of the TCP/IP stack and is responsible for delivering datagrams from a source host identified by the source address to one or multiple destination hosts identified by the destination address. IP hides the underlying physical network interface, is an unreliable, best-effort, and connectionless packet delivery protocol.

[SWS_TCPIP_00053]⌈ The TcpIp shall implement the Internet Protocol as defined in IETF RFC 791 (Internet Protocol of version 4).⌋()

[SWS_TCPIP_00095]⌈ The TcpIp shall encapsulate IP packets in Ethernet frames according to IETF RFC 894.⌋()

[SWS_TCPIP_00096]⌈ The TcpIp shall support the identification of the network an IP address belongs to, by using a network mask (prefix) in addition to the IP address according to IETF RFC 4632, section 3.1.⌋()

[SWS_TCPIP_00102]⌈ The TcpIp shall fulfill the Internet Protocol related requirements specified by IETF RFC 1122, section 3.2.1.1 (Version number), 3.2.1.2 (Checksum), 3.2.1.3 (Addressing), 3.2.1.7 (TTL), and 3.3.2 (Reassembly).⌋()

[SWS_TCPIP_00097]⌈ The TcpIp shall be able to transmit IP datagrams to a group of hosts identified by a single IP destination address (multicast address) according to IETF RFC 1112, section 4, 6.2, and 6.4.⌋()

[SWS_TCPIP_00098]⌈ The TcpIp shall be able to receive multicast IP datagrams identified by a single IP destination address (multicast address) according to IETF RFC 1112, section 4 and 7.2 (excluding the requirement for IGMP).⌋()

[SWS_TCPIP_00054]⌈ The TcpIp shall be able to reassemble incoming datagrams that are fragmented according to IETF RFC 815 (IP Datagram Reassembly Algorithms).⌋()

[SWS_TCPIP_00055]⌈ The TcpIp shall discover the maximum transmission unit (MTU) for a path as defined in IETF RFC 1191 (Path MTU Discovery).⌋()

7.1.2.1 Local Address Table

[SWS_TCPIP_00099]⌈ The TcpIp shall maintain a table of local IP addresses, which can be assigned to an EthIf controller during runtime according to the configuration container TcpIpLocalAddr (including its subcontainers).⌋()

Note: Each entry of the local IP address table is uniquely identified by the configuration parameter TcpIpAddrId.

[SWS_TCPIP_00100]⌈ In case no TcpIpStaticAddressConfig is provided, the TcpIp shall enable to specify a multicast IP address during runtime via TcpIp_RequestIpAddrAssignment(). ⌋()

[SWS_TCPIP_00130]⌈ The Local IP address used for a socket is specified via TcpIp_Bind().⌋()

7.1.3 Dynamic Host Configuration Protocol (DHCP)

[SWS_TCPIP_00058]⌈ The TcpIp shall implement the client part of the Dynamic Host Configuration Protocol (DHCP) for the dynamic configuration of IPv4 addresses as defined in IETF RFC 2131 (Dynamic Host Configuration Protocol).⌋()

7.1.4 Dynamic Configuration of IPv4 Link-Local Addresses (Auto-IP)

[SWS_TCPIP_00057]⌈ The TcpIp shall support the dynamic configuration of IPv4 Link Local addresses as defined in IETF RFC 3927 (Dynamic Configuration of IPv4 Link-Local Addresses).⌋()

7.1.5 Internet Control Message Protocol (ICMP)

[SWS_TCPIP_00059]⌈ The TcpIp shall support the transmission and reception of Internet Control Message Protocol (ICMP) messages as defined in IETF RFC 792 (Internet Control Message Protocol in version 4).⌋()

7.1.6 User Datagram Protocol (UDP)

[SWS_TCPIP_00060]⌈ The TcpIp shall implement the User Datagram Protocol (UDP) as defined in IETF RFC 768 (User Datagram Protocol).⌋()

[SWS_TCPIP_00103]⌈ The TcpIp shall fulfill the UDP related requirements specified by IETF RFC 1122, section 4.1.3.1 (Ports), 4.1.3.4 (UDP Checksums), and 4.1.3.6 (Invalid Addresses).⌋()

7.1.7 Transmission Control Protocol (TCP)

[SWS_TCPIP_00061]⌈ The TcpIp shall implement the Transmission Control Protocol (TCP) as defined in IETF RFC 793 (Transmission Control Protocol)⌋()

[SWS_TCPIP_00104]⌈ The TcpIp shall fulfill the TCP related requirements specified by IETF RFC 1122, section 4.2.2.3 (Window Size), 4.2.2.5 (TCP Options), 4.2.2.6 (MSS), 4.2.2.7 (Checksum), 4.2.2.9 (Initial sequence number selection), 4.2.2.10 (Simultaneous Open Attempts), 4.2.2.11 (Recovery from Old Duplicate SYN), 4.2.2.13 (Closing a Connection, excluding “half-duplex close”), 4.2.2.15 (Retransmission Timeout), 4.2.2.16 (Managing the Window), 4.2.2.17 (Probing Zero Windows), 4.2.2.18 (Passive OPEN Calls), 4.2.2.19 (TTL), 4.2.3.2 (delayed ACK), 4.2.3.6 (TCP Keep Alive), and 4.2.3.10 (Remote Address Validation).⌋()

[SWS_TCPIP_00062]⌈ The TcpIp shall support the Window and Acknowledgment Strategy in TCP as defined in IETF RFC 813.⌋()

[SWS_TCPIP_00063]⌈ The TcpIp shall implement the Nagle Algorithm as defined in IETF RFC 896 (Congestion Control in IP/TCP Internetworks).⌋()

[SWS_TCPIP_00064]⌈ The TcpIp shall implement the congestion control strategies slow-start, congestion avoidance, fast retransmit and fast recovery as defined in IETF RFC 5681.⌋()

7.2 Message Reception

[SWS_TCPIP_00105]⌈ The TcpIp shall silently discard incoming messages, which do not pass the acceptance filter.⌋()

[SWS_TCPIP_00106]⌈ An incoming message shall pass the acceptance filter if any of the following conditions evaluates to true:

(a) a socket exists which is bound to a local IP address that is assigned to the EthIf controller the incoming message has been received from AND the destination IP address specified in the incoming message is either a broadcast address or matches exactly with the local IP address

(b) a socket exists which is bound to a wildcard IP address that is assigned to the EthIf controller the incoming message has been received

(c) a socket exists which is bound to a wildcard IP address on any EthIf controller

⌋()

Note: A socket may either be explicitly bound to a local IP address by using TcpIp_Bind() or implicitly as part of TcpIp_UdpTransmit() (if it is called without a previous call of TcpIp-Bind()).

Note: The TcpIp silently discard an incoming SYN segment that is addressed to a broadcast or multicast address.

[SWS_TCPIP_00107]⌈ The TcpIp shall select the related target socket according to the protocol and port number specified in the incoming message after applying the message acceptance filter.⌋()

Figure 3 provides an overview of the acceptance filter for incoming messages.

Acceptance filter for incoming messages		Local IP address selected at Bind()					
		Unicast	Multicast	Broadcast	ANY_IP	ANY	w/o bind
DestAddr	Unicast	exact match	never	never	If-match	ever	never*
	Multicast	never	exact match	never	If-match	ever	never*
	Broadcast	If-match	If-match	If-match	If-match	ever	never*

ANY_IP: wildcard for any IP address of a specific interface
ANY: wildcard for any IP address on any interface
exact match: IP address and interface (EthIfCtrl) match
If-match: interface (EthIfCtrl) match
never*: no message is accepted unless TcpIp_UdpTransmit is called which implicitely binds a local Unicast IP address according to source address selection for transmission

Figure 3: Acceptance filter for incoming messages

7.3 Message Transmission

An overview of the source address selection for transmission is shown in Figure 4.

source address selection for transmission	Local IP address selected at Bind					
	Unicast	Multicast	Broadcast	ANY_IP	ANY	w/o bind
selected source address	bound addr	AutoSrcSel(If)	AutoSrcSel(If)	AutoSrcSel(If)	AutoSrcSel	AutoSrcSel

ANY_IP: wildcard for any IP address of a specific interface
ANY: wildcard for any IP address on any interface
AutoSrcSel (If): find LocalAddr which is unicast, assigned and uses the same interface
AutoSrcSel: find LocalAddr which is unicast, assigned and in the same subnet, use DefaultSrcAddr otherwise (if assigned)
DefaultSrcAddr: first LocalAddr (TcpIpAddrId = 0)

Figure 4: Source Address selection for transmission

[SWS_TCPIP_00101]⌈ The TcpIp shall choose the correct next hop for each datagram it sends according to IETF RFC 1122, section 3.3.1.1.⌋()

[SWS_TCPIP_00131]⌈ TcpIp shall always call `EthIf_Transmit()` with parameter TxConfirmation set to FALSE.⌋()

7.4 TCP/IP Stack state handling

[SWS_TCPIP_00083]⌈ The TcpIp module shall maintain a separate state for each EthIf controller used by the TcpIp module, store the latest state request and distinguish at least the following states: TCPIP_STATE_OFFLINE, TCPIP_STATE_STARTUP, TCPIP_STATE_ONLINE, TCPIP_STATE_ONHOLD, and TCPIP_STATE_SHUTDOWN.⌋()

[SWS_TCPIP_00136]⌈ The TcpIp module shall initiate according actions to achieve the requested state if the stored state request is not the active state.⌋()

[SWS_TCPIP_00084]⌈ After each transition the TcpIp module shall report the new state to EthSM via `EthSM_TcpIpModeIndication()`.⌋()

[SWS_TCPIP_00075]⌈ If TCPIP_STATE_ONLINE is requested for an EthIf controller and the current state is TCPIP_STATE_OFFLINE for that EthIf controller, the TcpIp module shall

(a) enable all IP address assignments according to the configured assignment methods (TcpIpAssignmentMethod) and triggers (TcpIpAssignmentTrigger) for that EthIf controller. (Note: If the assignment trigger is configured to TCPIP_MANUAL no assignment is actually performed but initiation by the upper layer enabled) and

(b) enter the state TCPIP_STATE_STARTUP for the EthIf controller.」()

[SWS_TCPIP_00127]「 In case multiple IP address assignment methods are configured and a new address from an assignment method with a higher priority (1 is highest) becomes available, TcpIp shall use the new IP address and release the IP address previously assigned by an assignment method with a lower priority.」()

[SWS_TCPIP_00088]「 If TCPIP_STATE_OFFLINE is requested for an EthIf controller and the current state is TCPIP_STATE_STARTUP for that EthIf controller, the TcpIp module shall

(a) abort all ongoing IP address assignment actions appropriate and

(b) enter the state TCPIP_STATE_OFFLINE for the EthIf controller.」()

[SWS_TCPIP_00085]「 If at least one IP address has been successfully assigned to an EthIf controller and the current state is TCPIP_STATE_STARTUP for that EthIf controller, the TcpIp module shall enter the state TCPIP_STATE_ONLINE for the EthIf controller.」()

Note: After successfully assignment of an IP address to the EthIf controller the SoAd module will be notified via `SoAd_LocalIpAddrAssignmentChg()` with State TCPIP_IPADDR_STATE_ASSIGNED.

[SWS_TCPIP_00076]「 If TCPIP_STATE_ONHOLD is requested for an EthIf controller and the current state is TCPIP_STATE_ONLINE for that EthIf controller, the TcpIp module shall

(a) notify the SoAd via `SoAd_LocalIpAddrAssignmentChg()` with State TCPIP_IPADDR_STATE_ONHOLD for all assigned IP addresses of the related EthIf controller, and

(b) deactivate the communication within the TcpIp module for the related EthIf controller, and

(c) enter the state TCPIP_STATE_ONHOLD for the EthIf controller.」()

[SWS_TCPIP_00086]「 If TCPIP_STATE_ONLINE is requested for an EthIf controller and the current state is TCPIP_STATE_ONHOLD for that EthIf controller, the TcpIp module shall

(a) reactivate the communication within the TcpIp module for the related EthIf controller,

(b) call `SoAd_LocalIpAddrAssignmentChg()` with State TCPIP_IPADDR_STATE_ASSIGNED for all assigned IP addresses of the related EthIf controller, and

(c) enter the state TCPIP_STATE_ONLINE for the EthIf controller.」()

[SWS_TCPIP_00077]「 If TCPIP_STATE_OFFLINE is requested or all assigned IP address have been released for an EthIf controller and the current state is TCPIP_STATE_ONLINE or TCPIP_STATE_ONHOLD for that EthIf controller, the TcpIp module shall

(a) call `SoAd_LocalIpAddrAssignmentChg()` with State TCPIP_IPADDR_STATE_UNASSIGNED for all assigned IP addresses of the related EthIf controller,
(b) deactivate the communication within the TcpIp module for the related EthIf controller,
(c) release related resources, i.e. any socket using the EthIf controller shall be closed and thereafter any IP address assigned to the EthIf controller shall be unassigned,
(d) in case the no EthIf controller is assigned any more, all unbound sockets shall be released as well, and
(e) enter the state TCPIP_STATE_SHUTDOWN for the EthIf controller.」()

[SWS_TCPIP_00087]「 If the current state of an EthIf controller is TCPIP_STATE_SHUTDOWN and all related resources have been released, the TcpIp module shall enter the state TCPIP_STATE_OFFLINE for the EthIf controller.」()

[SWS_TCPIP_00094]「 The TcpIp module shall only accept new TCP connections if the related EthIf controller is in state TCPIP_STATE_ONLINE.」()

[SWS_TCPIP_00144]「 The TcpIp module shall indicate events related to sockets to the SoAd module by using the SoAd_TcpIpEvent API and the following events: TCPIP_TCP_RESET, TCPIP_TCP_CLOSED, TCPIP_TCP_FIN_RECEIVED and TCPIP_UDP_CLOSED.」()

7.5 Error classification

This section describes how the TcpIp module has to manage the error classes that may occur during the life cycle of this basic software.

The general requirements document of AUTOSAR [4] specifies that all basic software modules must distinguish (according to the product life cycle) two error types:

- **Development errors**: errors that should only occur during development and whose detection and/or reporting can be statically configured (on/off). These types of errors are reported to the DET module
- **Production errors**: exceptions that cannot be avoided and are expected to occur also in production code. These types of errors are reported to the DEM module.

[SWS_TCPIP_00042]⌈The following table lists development errors that shall be distinguished by the TcpIp module:

Type or error	*Relevance*	*Related error code*	*Value [hex]*
API service called before initializing the module	Development	TCPIP_E_NOTINIT	0x01
API service called with NULL pointer	Development	TCPIP_E_NULL_PTR	0x02
Invalid argument	Development	TCPIP_E_INV_ARG	0x03
No buffer space available	Development	TCPIP_E_NOBUFS	0x04
Invalid socket address	Development	TCPIP_E_INV_SOCKADDR	0x05
Destination address required	Development	TCPIP_E_DESTADDRREQ	0x06
Message too long	Development	TCPIP_E_MSGSIZE	0x07
Protocol wrong type for socket	Development	TCPIP_E_PROTOTYPE	0x08
Address already in use	Development	TCPIP_E_ADDRINUSE	0x09
Can't assign requested address	Development	TCPIP_E_ADDRNOTAVAIL	0x0A
Socket is already connected	Development	TCPIP_E_ISCONN	0x0B
Socket is not connected	Development	TCPIP_E_NOTCONN	0x0C
Protocol not available	Development	TCPIP_E_NOPROTOOPT	0x0D
Address family not supported by protocol family	Development	TCPIP_E_AFNOSUPPORT	0x0E

⌋()

[SWS_TCPIP_00043]⌈ The following table lists production errors that shall be distinguished by the TcpIp module:

Type or error	*Relevance*	*Related error code*	*Value [hex]*
Network is down	Extended Production	TCPIP_E_NETDOWN	Assigned by DEM
Network is unreachable	Extended Production	TCPIP_E_NETUNREACH	Assigned by DEM
Network dropped connection on reset	Extended Production	TCPIP_E_NETRESET	Assigned by DEM
Connection aborted by TcpIp stack because of an error	Extended Production	TCPIP_E_CONNABORTED	Assigned by DEM
Connection reset by peer	Extended Production	TCPIP_E_CONNRESET	Assigned by DEM
Operation timed out	Extended Production	TCPIP_E_TIMEDOUT	Assigned by DEM
Connection refused	Extended Production	TCPIP_E_CONNREFUSED	Assigned by DEM
Host is down	Extended Production	TCPIP_E_HOSTDOWN	Assigned by DEM

No route to host	Extended Production	TCPIP_E_HOSTUNREACH	Assigned by DEM

」()

Note: Values for production code Event Ids are assigned externally by the configuration of the Dem. They are published in the file Dem_IntErrId.h and included via Dem.h.

7.5.1 Error detection

For details refer to the chapter 7.3 “Error Detection” in *SWS_BSWGeneral.*

7.5.2 Error notification

For details refer to the chapter 7.4 “Error notification” in *SWS_BSWGeneral.*

7.6 Application notes

7.7 Debugging Concept

For details refer to the chapter 7.1.17 “Debugging support” in *SWS_BSWGeneral.*

7.8 Version checking

For details refer to the chapter 5.1.8 “Version Check” in *SWS_BSWGeneral.*

8 API specification

8.1 Imported types

The following types shall be imported by the TcpIp from the modules given:

[SWS_TCPIP_00008] ⌜

Module	*Imported Type*
ComStack_Types	BufReq_ReturnType
Dem	Dem_EventIdType
	Dem_EventStatusType
Eth_GeneralTypes	Eth_FilterActionType
	Eth_FrameType
Std_Types	Std_ReturnType
	Std_VersionInfoType

⌟()

8.2 Type definitions

[SWS_TCPIP_00067]⌜

Name:	`TcpIp_ConfigType`		
Type:	`--`		
Range:	`implementation specific`	`--`	The content of the configuration data structure is implementation specific.
Description:	Configuration data structure of the TcpIp module.		

⌟()

[SWS_TCPIP_00009] ⌜

Name:	`TcpIp_DomainType`		
Type:	`uint16`		
Range:	`TCPIP_AF_INET`	`0x02`	Use IPv4
	`TCPIP_AF_INET6`	`0x1c`	Use IPv6
Description:	TcpIp address families.		

⌟()

[SWS_TCPIP_00010] ⌜

Name:	`TcpIp_ProtocolType`	
Type:	`Enumeration`	
Range:	`TCPIP_IPPROTO_TCP`	0x06: Use TCP
	`TCPIP_IPPROTO_UDP`	0x11: Use UDP
Description:	Protocol type used by a socket.	

⌟()

[SWS_TCPIP_00012]⌜

Name:	TcpIp_SockAddrType		
Type:	Structure		
Element:	TcpIp_DomainType	domain	This is the code for the address format of this address
Description:	Generic structure used by APIs to specify an IP address. (A specific address type can be derived from this structure via a cast to the specific struct type.)		

⌋()

[SWS_TCPIP_00013]⌈

Name:	TcpIp_SockAddrInetType		
Type:	Structure		
Element:	TcpIp_DomainType	domain	This is the code for the address format of this address
	uint16	port	port number
	uint32[1]	addr	IPv4 address in network byte order
Description:	This structure defines an IPv4 address type which can be derived from the generic address structure via cast.		

⌋()

[SWS_TCPIP_00014]⌈

Name:	TcpIp_SockAddrInet6Type		
Type:	Structure		
Element:	TcpIp_DomainType	domain	This is the code for the address format of this address
	uint16	port	port number
	uint32[4]	addr	IPv6 address in network byte order
Description:	This structure defines a IPv6 address type which can be derived from the generic address structure via cast.		

⌋()

[SWS_TCPIP_00030]⌈

Name:	TcpIp_LocalAddrIdType
Type:	uint8
Description:	Address identification type for unique identification of a local IP address and EthIf Controller configured in the TcpIp module.

⌋()

[SWS_TCPIP_00038]⌈

Name:	TcpIp_SocketIdType
Type:	uint8, uint16
Description:	socket identifier type for unique identification of a TcpIp stack socket. TCPIP_SOCKETID_INVALID shall specify an invalid socket handle

⌋()

[SWS_TCPIP_00073]⌈

Name:	TcpIp_StateType	
Type:	Enumeration	
Range:	TCPIP_STATE_ONLINE	TCP/IP stack state for a specific EthIf controller is ONLINE, i.e. communication via at least one IP address is possible.
	TCPIP_STATE_ONHOLD	TCP/IP stack state for a specific EthIf controller is ONHOLD, i.e. no communication is currently possible (e.g. link down).
	TCPIP_STATE_OFFLINE	TCP/IP stack state for a specific EthIf controller is OFFLINE, i.e. no communication is possible.
	TCPIP_STATE_STARTUP	TCP/IP stack state for a specific EthIf controller is STARTUP, i.e. IP address assignment in progress or ready for manual start, communication is currently not possible.
	TCPIP_STATE_SHUTDOWN	TCP/IP stack state for a specific EthIf controller is SHUTDOWN, i.e. release of resources using the EthIf controller, release of IP address assignment.
Description:	Specifies the TcpIp state for a specific EthIf controller.	

⌋()

[SWS_TCPIP_00082]⌈

Name:	TcpIp_IpAddrStateType	
Type:	Enumeration	
Range:	TCPIP_IPADDR_STATE_ASSIGNED	local IP address is assigned
	TCPIP_IPADDR_STATE_ONHOLD	local IP address is assigned, but cannot be used as the network is not active
	TCPIP_IPADDR_STATE_UNASSIGNED	local IP address is unassigned
Description:	Specifies the state of local IP address assignment	

⌋()

[SWS_TCPIP_00031]⌈

Name:	TcpIp_EventType	
Type:	Enumeration	
Range:	TCPIP_TCP_RESET	TCP connection was reset, TCP socket and all related resources have been released.
	TCPIP_TCP_CLOSED	TCP connection was closed successfully, TCP socket and all related resources have been released.
	TCPIP_TCP_FIN_RECEIVED	A FIN signal was received on the TCP connection, TCP socket is still valid.
	TCPIP_UDP_CLOSED	UDP socket and all related resources have been released.
Description:	Events reported by TcpIp.	

⌋()

[SWS_TCPIP_00065]⌈

Name:	TcpIp_IpAddrAssignmentType	
Type:	Enumeration	
Range:	TCPIP_IPADDR_ASSIGNMENT_STATIC	--
	TCPIP_IPADDR_ASSIGNMENT_AUTOIP	--

	TCPIP_IPADDR_ASSIGNMENT_AUTOIP_DOIP	--
	TCPIP_IPADDR_ASSIGNMENT_DHCP	--
Description:	Specification of IP address assignment policy.	

⌋()

[SWS_TCPIP_00066]⌈

Name:	TcpIp_ReturnType	
Type:	Enumeration	
Range:	TCPIP_OK	operation completed successfully.
	TCPIP_E_NOT_OK	operation failed.
	TCPIP_E_ARP_CACHE_MISS	operation failed because of an ARP cache miss.
Description:	TcpIp specific return type.	

⌋()

[SWS_TCPIP_00126]⌈

Name:	TcpIp_ParamIdType		
Type:	uint8		
Range:	TCPIP_PARAMID_TCP_RXWND_MAX	0x00	Specifies the maximum TCP receive window for the socket.
	TCPIP_PARAMID_FRAMEPRIO	0x01	Specifies the frame priority for outgoing frames on the socket.
	TCPIP_PARAMID_TCP_NAGLE	0x02	Specifies if the Nagle Algorithm according to IETF RFC 896 is enabled or not.
	TCPIP_PARAMID_TCP_KEEPALIVE	0x03	Specifies if TCP Keep Alive Probes are sent on the socket connection.
	TCPIP_PARAMID_TTL	0x04	Specifies the time to live value for outgoing frames on the socket.
	TCPIP_PARAMID_VENDOR_SPECIFIC	0x80	Start of vendor specific range of parameter IDs.
Description:	Type for the specification of all supported Parameter IDs.		

⌋()

[SWS_TCPIP_00133]⌈

Name:	TcpIpIpAddrWildcardType		
Type:	uint32		
Range:	TCPIP_IPADDR_ANY	implementation specific	defines the value used as wildcard
Description:	IP address wildcard.		

⌋()

[SWS_TCPIP_00132]⌈

Name:	TcpIpIp6AddrWildcardType		
Type:	uint32		
Range:	TCPIP_IP6ADDR_ANY	implementation specific	defines the value used as wildcard for all IP6 address parts
Description:	IP6 address wildcard.		

⌋()

[SWS_TCPIP_00134]⌈

Name:	`TcpIpPortWildcardType`		
Type:	`uint16`		
Range:	`TCPIP_PORT_ANY`	`implementation specific`	defines the value used as wildcard
Description:	Port wildcard.		

⌋()

[SWS_TCPIP_00135]⌈

Name:	`TcpIpLocalAddrIdWildcardType`		
Type:	`TcpIp_LocalAddrIdType`		
Range:	`TCPIP_LOCALADDRID_ANY`	`implementation specific`	defines the value used as wildcard
Description:	LocalAddrId wildcard.		

⌋()

8.3 Function definitions

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

8.3.1 General

8.3.1.1 TcpIp_Init

[SWS_TCPIP_00002]⌈

Service name:	TcpIp_Init	

Syntax:	`void TcpIp_Init(`
	`  const TcpIp_ConfigType* ConfigPtr`
	`)`	

Service ID[hex]:	0x01	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ConfigPtr	Pointer to the configuration data of the TcpIp module
Parameters (inout):	None	
Parameters (out):	None	
Return value:	void	None
Description:	This service initializes the TCP/IP Stack. TcpIp_Init may not block the start-up process for an indefinite amount of time. Caveats: The call of this service is mandatory before using the TcpIp instance for further processing.	

⌋()

8.3.1.2 TcpIp_GetVersionInfo

[SWS_TCPIP_00004] ⌈

Service name:	TcpIp_GetVersionInfo	
Syntax:	`void TcpIp_GetVersionInfo( Std_VersionInfoType* versioninfo )`	
Service ID[hex]:	0x02	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	versioninfo	Pointer to where to store the version information of this module.
Return value:	None	
Description:	Returns the version information.	

」()

[SWS_TCPIP_00005][「 The function TcpIp_GetVersionInfo shall return the version information of this module. The version information includes:

- Module Id
- Vendor Id
- Vendor specific version numbers (BSW00407).

」()

[SWS_TCPIP_00006]「 The function TcpIp_GetVersionInfo shall be pre compile time configurable On/Off by the configuration parameter: TCPIP_VERSION_INFO_API」()

8.3.2 Core Communication Control

8.3.2.1 TcpIp_GetSocket

[SWS_TCPIP_00018]「

Service name:	TcpIp_GetSocket	
Syntax:	`Std_ReturnType TcpIp_GetSocket( TcpIp_DomainType Domain, TcpIp_ProtocolType Protocol, TcpIp_SocketIdType* SocketIdPtr )`	
Service ID[hex]:	0x03	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	Domain	IP address family.
	Protocol	Socket protocol as sub-family of parameter type.
Parameters (inout):	None	
Parameters (out):	SocketIdPtr	Pointer to socket identifier representing the requested socket. This socket identifier must be provided for all further API calls which requires a SocketId. Note: SocketIdPtr is only valid if return value is E_OK.
Return value:	Std_ReturnType	Result of operation E_OK The request has been accepted

		E_NOT_OK The request has not been accepted: no free socket
Description:	By this API service the TCP/IP stack is requested to allocate a new socket. Note: Each accepted incoming TCP connection also allocates a socket resource.	

⌋()

[SWS_TCPIP_00128]⌈ If development error detection is enabled, the service `TcpIp_GetSocket()` shall check the parameter Domain for being valid and raise the development error TCPIP_E_AFNOSUPPORT if it is invalid.⌋()

8.3.2.2 TcpIp_Close

[SWS_TCPIP_00017]⌈

Service name:	TcpIp_Close	

Syntax:	`Std_ReturnType TcpIp_Close(`
	`  TcpIp_SocketIdType SocketId,`
	`  boolean Abort`
	`)`	

Service ID[hex]:	0x04	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in):	SocketId	Socket handle identifying the local socket resource.
	Abort	TRUE: connection will immediately be terminated by sending a RST-Segment and releasing all related resources. FALSE: connection will be terminated after performing a regular connection termination handshake and releasing all related resources.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted.
Description:	By this API service the TCP/IP stack is requested to close the socket and release all related resources.	

⌋()

[SWS_TCPIP_00109]⌈ The service TcpIp_Close() shall perform the following actions for the socket specified by SocketId in case it is a TCP socket:
(a) if the connection is active and
(a1) abort = FALSE: the connection shall be terminated after performing a regular connection termination handshake and releasing all related resources.
(a2) abort = TRUE: connection shall immediately be terminated by sending a RST-Segment and releasing all related resources.
(b) if the socket is in the Listen state, the Listen state shall be left immediately and related resources shall be released.⌋()

[SWS_TCPIP_00110]⌈ The service TcpIp_Close() shall release all related resources immediately for the socket specified by SocketId in case it is a UDP socket .⌋()

Note: SoAd will be notified via SoAd_TcpIpEvent(TCPIP_TCP_CLOSED, TCPIP_TCP_RESET or TCPIP_UDP_CLOSED) after the socket and all related resources have been released. After this call the SocketId is invalid until allocated again with TcpIp_GetSocket().

8.3.2.3 TcpIp_Bind

[SWS_TCPIP_00015]⌜

Service name:	TcpIp_Bind	

Syntax:	`Std_ReturnType TcpIp_Bind(`
	`  TcpIp_SocketIdType SocketId,`
	`  TcpIp_LocalAddrIdType LocalAddrId,`
	`  uint16* PortPtr`
	`)`	

Service ID[hex]:	0x05	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in):	SocketId	Socket identifier of the related local socket resource.
	LocalAddrId	IP address identifier representing the local IP address and EthIf controller to bind the socket to. Note: to listen to all EthIf controller, TCPIP_LOCALADDRID_ANY has to be specified as LocalAddrId. Note: to listen on any IP addresss of a EthIf controller, the configuration parameter TcpIpStaticIpAddress referenced by LocalAddrId must be set to "ANY". The remote IP address of an incoming packet has no effect then. In case the socket shall be used as client socket, the IP address and EthIf controller represented by LocalAddrId is used for transmission. Note: for an automatic selection of the Local IP address and EthIf Controller, TCPIP_LOCALADDRID_ANY has to be specified as LocalAddrId.
Parameters (inout):	PortPtr	Pointer to memory where the local port to which the socket shall be bound is specified. In case the parameter is specified as TCPIP_PORT_ANY, the TCP/IP stack shall choose the local port automatically from the range 49152 to 65535 and shall update the parameter to the chosen value.
Parameters (out):	None	
Return value:	Std_ReturnType	Result of operation E_OK The request has been accepted E_NOT_OK The request has not been accepted (e.g. address in use)
Description:	By this API service the TCP/IP stack is requested to bind a UDP or TCP socket to a local resource.	

⌟()

[SWS_TCPIP_00111]⌜ The service TcpIp_Bind() shall bind the socket specified by parameter SocketId to the local resource specified by parameters LocalAddrId and PortPtr.⌟()

Note: Sockets that shall be switched in a listening state later on must be bound to a local resource. Optionally this API can be used to specify the local IP address and port used by later calls of TcpIp_TcpConnect() or TcpIp_UdpTransmit().

8.3.2.4 TcpIp_TcpConnect

[SWS_TCPIP_00022]⌈

Service name:	TcpIp_TcpConnect	
Syntax:	`Std_ReturnType TcpIp_TcpConnect( TcpIp_SocketIdType SocketId, TcpIp_SockAddrType* RemoteAddrPtr )`	
Service ID[hex]:	0x06	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in):	SocketId	Socket identifier of the related local socket resource.
	RemoteAddrPtr	IP address and port of the remote host to connect to.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted, e.g. connection is already established or no route to destination specified by remoteAddrPtr found.
Description:	By this API service the TCP/IP stack is requested to establish a TCP connection to the configured peer.	

⌋()

[SWS_TCPIP_00112]⌈ The service TcpIp_TcpConnect() shall establish a TCP connection between the local socket specified by parameter SocketId and the remote socket specified with parameter RemoteAddrPtr.⌋()

[SWS_TCPIP_00129]⌈ [If development error detection is enabled and the parameter RemoteAddrPtr equals NULL_PTR, the TcpIp_TcpConnect function shall raise the development error code TCPIP_E_NULL_PTR and the TcpIp_TcpConnect function shall return E_NOT_OK. ⌋()

8.3.2.5 TcpIp_TcpListen

[SWS_TCPIP_00023]⌈

Service name:	TcpIp_TcpListen	
Syntax:	`Std_ReturnType TcpIp_TcpListen( TcpIp_SocketIdType SocketId, uint16 MaxChannels )`	
Service ID[hex]:	0x07	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in):	SocketId	Socket identifier of the related local socket resource.
	MaxChannels	Maximum number of new parallel connections established on this listen connection.

Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted, the socket is not configured to be a server socket.
Description:	By this API service the TCP/IP stack is requested to listen on the TCP socket specified by the socket identifier.	

」()

[SWS_TCPIP_00113]「 The service TcpIp_TcpListen() shall put the socket specified by SocketId to the listen state (i.e. local socket is listening for incoming connections).」()

[SWS_TCPIP_00114]「 TcpIp shall derive a separate socket from the listen socket to establish a new connection from an incoming connection request on the listen socket and limit the number of new parallel connections to the value specified by MaxChannels.」()

8.3.2.6 TcpIp_TcpReceived

[SWS_TCPIP_00024]「

Service name:	TcpIp_TcpReceived	
Syntax:	`Std_ReturnType TcpIp_TcpReceived( TcpIp_SocketIdType SocketId, uint32 Length )`	
Service ID[hex]:	0x08	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in):	SocketId	Socket identifier of the related local socket resource.
	Length	Number of bytes finally consumed by the upper layer.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted
Description:	By this API service the reception of socket data is confirmed to the TCP/IP stack.	

」()

[SWS_TCPIP_00115]「 The service TcpIp_TcpReceived() shall increase the TCP receive window of the socket specified by SocketId considering the number of finally consumed bytes specified by Length.」()

8.3.2.7 TcpIp_RequestComMode

[SWS_TCPIP_00070]⌈

Service name:	TcpIp_RequestComMode	
Syntax:	`Std_ReturnType TcpIp_RequestComMode( uint8 CtrlIdx, TcpIp_StateType State )`	
Service ID[hex]:	0x09	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CtrlIdx	EthIf controller index to identify the communication network where the TcpIp state is requested.
	State	Requested TcpIp state.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Service accepted E_NOT_OK: Service denied
Description:	By this API service the TCP/IP stack is requested to change the TcpIp state of the communication network identified by EthIf controller index.	

⌋()

[SWS_TCPIP_00071]⌈ If TCPIP_STATE_ONLINE is requested, the TcpIp module shall initiate activation of the TcpIp communication on the related EthIf controller (e.g. start IP-Address assignment according to the configured IP address assignment policy for the EthIf controller).⌋()

[SWS_TCPIP_00072]⌈ If TCPIP_STATE_OFFLINE is requested, the TcpIp module shall initiate deactivation of the TcpIp communication on the related EthIf controller (e.g. close all sockets using the specified EthIf controller).⌋()

[SWS_TCPIP_00074]⌈ If TCPIP_STATE_ONHOLD is requested, the TcpIp module shall set the TcpIp communication to on hold, i.e. new transmit requests shall not be accepted, but sockets and assigned IP addresses shall be kept.⌋()

[SWS_TCPIP_00089]⌈ If TCPIP_STATE_STARTUP or TCPIP_STATE_SHUTDOWN is requested as state the function TcpIp_RequestComMode shall abort with E_NOT_OK and report TCPIP_E_INV_ARG to DET if DET is enabled.⌋()

Note: According to [SWS_TCPIP_00075] and [SWS_TCPIP_00077] TCPIP_STATE_STARTUP or TCPIP_STATE_SHUTDOWN are intermediate states arising from requesting TCPIP_STATE_OFFLINE or TCPIP_STATE_ONLINE. Requesting these intermediate states is not useful.

8.3.3 Extended Communication Control and Information

8.3.3.1 TcpIp_RequestIpAddrAssignment

[SWS_TCPIP_00037]⌈

Service name:	TcpIp_RequestIpAddrAssignment	
Syntax:	`Std_ReturnType TcpIp_RequestIpAddrAssignment( TcpIp_LocalAddrIdType LocalAddrId, TcpIp_IpAddrAssignmentType Type, TcpIp_SockAddrType* LocalIpAddrPtr )`	
Service ID[hex]:	0x0A	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	LocalAddrId	IP address index specifying the IP address for which an assignment shall be initiated.
	Type	type of IP address assignment which shall be initiated
	LocalIpAddrPtr	pointer to structure containing the IP address which shall be assigned to the EthIf controller indirectly specified via LocalAddrId. Note: This parameter is only used in case the parameters Type is set to TCPIP_IPADDR_ASSIGNMENT_STATIC.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted
Description:	By this API service the local IP address assignment for the IP address specified by LocalAddrId shall be initiated.	

⌋()

[SWS_TCPIP_00116]⌈ The service `TcpIp_RequestIpAddrAssignment()` shall initiate the local IP address assignment according to the IP address table entry specified by LocalAddId using the method specified by Type.⌋()

[SWS_TCPIP_00079]⌈In case `TcpIp_RequestIpAddrAssignment()` is called with parameter Type set to TCPIP_IPADDR_ASSIGNMENT_STATIC and no TcpIpStaticIpAddressConfig container is configured for the LocalAddr specified by parameter LocalAddrId, TcpIp shall assign the IP address specified by parameter LocalIpAddrPtr as soon as TCPIP_STATE_ONLINE is requested or immediately if already requested.⌋()

[SWS_TCPIP_00080]⌈ In case a multicast address is assigned, TcpIp shall derive the related physical address from the multicast IP address and add the derived address to the Eth MAC address filter by calling `EthIf_UpdatePhys-AddrFilter()` with action set to ETH_ADD_TO_FILTER.⌋()

8.3.3.2 TcpIp_ReleaseIpAddrAssignment

[SWS_TCPIP_00078]⌈

Service name:	TcpIp_ReleaseIpAddrAssignment
Syntax:	`Std_ReturnType TcpIp_ReleaseIpAddrAssignment(`

	`TcpIp_LocalAddrIdType LocalAddrId`)	
Service ID[hex]:	0x0B	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	LocalAddrId	IP address index specifying the IP address for which an assignment shall be released.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted
Description:	By this API service the local IP address assignment for the IP address specified by LocalAddrId shall be released.	

」()

[SWS_TCPIP_00117]「 The service `TcpIp_ReleasepAddrAssignment()` shall release the local IP address assignment related to the IP address table entry specified by LocalAddId.」()

8.3.3.3 TcpIp_IcmpTransmit

[SWS_TCPIP_00039]「

Service name:	TcpIp_IcmpTransmit	

Syntax:	`Std_ReturnType TcpIp_IcmpTransmit(`
	`  TcpIp_LocalAddrIdType LocalIpAddrId,`
	`  TcpIp_SockAddrType* RemoteAddrPtr,`
	`  uint8 Ttl,`
	`  uint8 Type,`
	`  uint8 Code,`
	`  uint16 DataLength,`
	`  uint8* DataPtr`
	)	

Service ID[hex]:	0x0C	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	LocalIpAddrId	IP address identifier representing the local IP address and EthIf controller which shall be used for transmission of the ICMP message.
	RemoteAddrPtr	pointer to struct representing the remote address
	Ttl	Time to live value to be used for the ICMP message. If 0 is specified the default value shall be used.
	Type	type field value to be used in the ICMP message (Note: the value of the type field determines the format of the remaining ICMP message data)
	Code	code field value to be used in the ICMP message
	DataLength	length of ICMP message
	DataPtr	Pointer to data which shall be sent as ICMP message data
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Result of operation E_OK The ICMP message has been sent successfully E_NOT_OK The ICMP message was not sent.

Description:	By this API service the TCP/IP stack sends an ICMP message according to the specified parameters.

⌋()

[SWS_TCPIP_00118]⌈ The service `TcpIp_IcmpTransmit()` shall (a) construct an ICMP message according to the parameters Type, Code, DataLength and DataPtr and (b) transmit the ICMP message using the local IP address and EthIf controller specified by LocalIpAddrId to the destination specified by RemoteAddrPtr using a time to live value according to the parameter Ttl.⌋()

8.3.3.4 TcpIp_DhcpReadOption

[SWS_TCPIP_00040]⌈

Service name:	TcpIp_DhcpReadOption	

Syntax:	`Std_ReturnType TcpIp_DhcpReadOption(`
	`  TcpIp_LocalAddrIdType LocalIpAddrId,`
	`  uint8 Option,`
	`  uint8* DataLength,`
	`  uint8* DataPtr`
	`)`	

Service ID[hex]:	0x0D	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	LocalIpAddrId	IP address identifier representing the local IP address and EthIf controller for which the DHCP option shall be read.
	Option	DHCP option
	DataPtr	Pointer to memory containing DHCP option data
Parameters (inout):	DataLength	As input parameter, contains the length of the provided data buffer. Will be overwritten with the length of the actual data.
Parameters (out):	None	
Return value:	Std_ReturnType	Result of operation E_OK requested data retrieved successfully. E_NOT_OK requested data could not be retrieved.
Description:	By this API service the TCP/IP stack retrieves DHCP option data identified by parameter option for already received DHCP options.	

⌋()

8.3.3.5 TcpIp_DhcpWriteOption

[SWS_TCPIP_00020]⌈

Service name:	TcpIp_DhcpWriteOption	

Syntax:	`Std_ReturnType TcpIp_DhcpWriteOption(`
	`  TcpIp_LocalAddrIdType LocalIpAddrId,`
	`  uint8 Option,`
	`  uint8 DataLength,`
	`  const uint8* DataPtr`
	`)`	

Service ID[hex]:	0x0E	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	LocalIpAddrId	IP address identifier representing the local IP address and EthIf

		controller for which the DHCP option shall be written.
	Option	DHCP option, e.g. Host Name
	DataLength	length of DHCP option data
	DataPtr	Pointer to memory containing DHCP option data
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Result of operation E_OK no error occured. E_NOT_OK DHCP option data could not be written.
Description:	By this API service the TCP/IP stack writes the DHCP option data identified by parameter option.	

)()

8.3.3.6 TcpIp_ChangeParameter

[SWS_TCPIP_00016](

Service name:	TcpIp_ChangeParameter	

Syntax:	`Std_ReturnType TcpIp_ChangeParameter(`
	`  TcpIp_SocketIdType SocketId,`
	`  TcpIp_ParamIdType ParameterId,`
	`  uint8* ParameterValue`
	`)`	

Service ID[hex]:	0x0F	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in):	SocketId	Socket identifier of the related local socket resource.
	ParameterId	Identifier of the parameter to be changed
	ParameterValue	Pointer to memory containing the new parameter value
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The parameter has been changed successfully. E_NOT_OK: The parameter could not be changed.
Description:	By this API service the TCP/IP stack is requested to change a parameter of a socket. E.g. the Nagle algorithm may be controlled by this API.	

)()

[SWS_TCPIP_00119](The service `TcpIp_ChangeParameter()` shall change the parameter specified by ParameterId with the value specified by ParameterValue of the socket specified by SocketId.)()

8.3.3.7 TcpIp_GetIpAddr

[SWS_TCPIP_00032](

Service name:	TcpIp_GetIpAddr

Syntax:	`Std_ReturnType TcpIp_GetIpAddr(`
	`  TcpIp_LocalAddrIdType LocalAddrId,`
	`  TcpIp_SockAddrType* IpAddrPtr,`
	`  uint8* NetmaskPtr,`

	`TcpIp_SockAddrType* DefaultRouterPtr`)	
Service ID[hex]:	0x10	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	LocalAddrId	Local address identifier referring to the local IP address which shall be obtained.
Parameters (inout):	None	
Parameters (out):	IpAddrPtr	Pointer to a struct where the IP address is stored. Struct members not related to the IP address are of arbitrary value and shall not be used.
	NetmaskPtr	Pointer to memory where Network mask of IPv4 address or address prefix of IPv6 address in CIDR Notation is stored
	DefaultRouterPtr	Pointer to struct where the IP address of the default router (gateway) is stored (struct member “port” is not used and of arbitrary value)
Return value:	Std_ReturnType	Result of operation E_OK The request was successful E_NOT_OK The request was not successful.
Description:	Obtains the local IP address actually used by LocalAddrId, the netmask and default router	

⌋()

8.3.3.8 TcpIp_GetPhysAddr

[SWS_TCPIP_00033]⌈

Service name:	TcpIp_GetPhysAddr	

Syntax:	`Std_ReturnType TcpIp_GetPhysAddr(`
	`  TcpIp_LocalAddrIdType LocalAddrId,`
	`  uint8* PhysAddrPtr`
	)	

Service ID[hex]:	0x11	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	LocalAddrId	Local address identifier implicitely specifing the EthIf controller for which the physical address shall be obtained.
Parameters (inout):	None	
Parameters (out):	PhysAddrPtr	Pointer to the memory where the physical source address (MAC address) in network byte order is stored
Return value:	Std_ReturnType	Result of operation E_OK The request was successful E_NOT_OK The request was not successful, e.g. no unique Ctrl specified via IpAddrId.
Description:	Obtains the physical source address used by the EthIf controller implicitly specified via LocalAddrId.	

⌋()

8.3.3.9 TcpIp_GetRemotePhysAddr

[SWS_TCPIP_00137]⌈

Service name:	TcpIp_GetRemotePhysAddr

Syntax:	`TcpIp_ReturnType TcpIp_GetRemotePhysAddr(`
	`  uint8 CtrlIdx,`
	`  TcpIp_SockAddrType* IpAddrPtr,`

	TcpIp_ReturnType PhysAddrPtr, boolean initRes)	
Service ID[hex]:	0x16	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CtrlIdx	EthIf controller index to identify the related ARP table.
	IpAddrPtr	specifies the IP address for which the physical address shall be retrieved
	initRes	specifies if the address resolution shall be initiated (TRUE) or not (FALSE) in case the physical address related to the specified IP address is currently unknown.
Parameters (inout):	None	
Parameters (out):	PhysAddrPtr	Pointer to the memory where the physical address (MAC address) related to the specified IP address is stored in network byte order.
Return value:	TcpIp_ReturnType	TCPIP_E_OK specified IP address resolved, physical address provided via PhysAddrPtr TCPIP_E_ARP_CACHE_MISS physical address currently unknown (address resolution initiated if initRes set to TRUE)
Description:	TcpIp_GetRemotePhysAddr queries the IP/physical address translation table specified by CtrlIdx and returns the physical address related to the IP address specified by IpAddrPtr. In case no physical address can be retrieved and parameter initRes is TRUE, address resolution for the specified IP address is initiated on the local network.	

()

[SWS_TCPIP_00138]⌈ TcpIp_GetRemotePhysAddr shall lookup the physical address for the IP address specified by IpAddrPtr at the IP/physical address translation table related to the controller identified by CtrlIdx.
(1) If the physical address is already known, PhysAddrPtr shall be set to the related physical address and the function shall return with TCPIP_E_OK.
(2) Otherwise it shall (a) initiate an address resolution if parameter initRes is set to TRUE and (b) return with TCPIP_E_ARP_CACHE_MISS. PhysAddrPtr is not updated in this case.⌋()

[SWS_TCPIP_00139]⌈ TcpIp_GetRemotePhysAddr shall immediately return with TCPIP_E_NOT_OK if it is called with an IP address that is not part of the same sub network as the local address currently assigned to the controller identified by CtrlIdx.⌋
()

8.3.3.10 TcpIp_GetCtrlIdx

[SWS_TCPIP_00140]⌈

Service name:	TcpIp_GetCtrlIdx
Syntax:	Std_ReturnType TcpIp_GetCtrlIdx(TcpIp_LocalAddrIdType LocalAddrId, uint8* CtrlIdxPtr)
Service ID[hex]:	0x17
Sync/Async:	Synchronous

Reentrancy:	Reentrant	
Parameters (in):	LocalAddrId	Local address identifier implicitely specifing the EthIf controller that shall be returned.
Parameters (inout):	None	
Parameters (out):	CtrlIdxPtr	Pointer to the memory where the index of the controller related to LocalAddrId is stored
Return value:	Std_ReturnType	Result of operation E_OK the request was successful E_NOT_OK the request was not successful.
Description:	TcpIp_GetCtrlIdx returns the index of the controller related to LocalAddrId.	

」()

[SWS_TCPIP_00141][「 TcpIp_GetCtrlIdx shall return the index of the controller related to LocalAddrId. 」()

8.3.4 Transmission

8.3.4.1 TcpIp_UdpTransmit

[SWS_TCPIP_00025]「

Service name:	TcpIp_UdpTransmit	

Syntax:	`TcpIp_ReturnType TcpIp_UdpTransmit(`
	`  TcpIp_SocketIdType SocketId,`
	`  uint8* DataPtr,`
	`  TcpIp_SockAddrType* RemoteAddrPtr,`
	`  uint16 TotalLength`
	`)`	

Service ID[hex]:	0x12	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in):	SocketId	Socket identifier of the related local socket resource.
	DataPtr	Pointer to a linear buffer of TotalLength bytes containing the data to be transmitted. In case DataPtr is a NULL_PTR, TcpIp shall retrieve data from SoAd via callback SoAd_CopyTxData().
	RemoteAddrPtr	IP address and port of the remote host to transmit to.
	TotalLength	indicates the payload size of the UDP datagram.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	TcpIp_ReturnType	TCPIP_OK: UDP message has been forwarded to EthIf for transmission. TCPIP_E_NOT_OK: UDP message could not be sent because of a permanent error, e.g. message is too long. TCPIP_E_ARP_CACHE_MISS: UDP message could not be sent because of an ARP cache miss, ARP request has been sent and upper layer may retry transmission by calling this function later again.
Description:	This service transmits data via UDP to a remote node. The transmission of the data is immediately performed with this function call by forwarding it to EthIf.	

」()

[SWS_TCPIP_00120]⌈ The service `TcpIp_UdpTransmit()` shall immediately transmit TotalLength data bytes via UDP and the socket specified by SocketId to a remote socket specified by RemoteAddrPtr according to the sequence diagram specified in section 9.5.⌋()

[SWS_TCPIP_00121]⌈ DataPtr shall either point to a linear buffer of TotalLength bytes containing the data for transmission or be a NULL_PTR. For data transmission the service `TcpIp_UdpTransmit()` shall either use all data from the linear buffer if DataPtr is not a NULL_PTR, or retrieve TotalLength data bytes from SoAd by calling `SoAd_CopyTxData()` one or multiple times in the context of this service otherwise.⌋ ()

[SWS_TCPIP_00122]⌈ The service `TcpIp_UdpTransmit()` shall select the local IP address and port for transmission if the socket specified by SocketId has not been bound to a local resource via a previous call to `TcpIp_Bind()`.⌋()

8.3.4.2 TcpIp_TcpTransmit

[SWS_TCPIP_00050]⌈

Service name:	TcpIp_TcpTransmit	
Syntax:	`Std_ReturnType TcpIp_TcpTransmit( TcpIp_SocketIdType SocketId, uint8* DataPtr, uint32 AvailableLength, boolean ForceRetrieve )`	
Service ID[hex]:	0x13	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different SocketIds. Non reentrant for the same SocketId.	
Parameters (in):	SocketId	Socket identifier of the related local socket resource.
	DataPtr	Pointer to a linear buffer of AvailableLength bytes containing the data to be transmitted. In case DataPtr is a NULL_PTR, TcpIp shall retrieve data from SoAd via callback SoAd_CopyTxData().
	AvailableLength	Available data for transmission in bytes.
	ForceRetrieve	This parameter is only valid if DataPtr is a NULL_PTR. Indicates how the TCP/IP stack retrieves data from SoAd if DataPtr is a NULL_PTR. TRUE: the whole data indicated by availableLength shall be retrieved from the upper layer via one or multiple SoAd_CopyTxData() calls within the context of this transmit function. FALSE: The TCP/IP stack may retrieve up to availableLength data from the upper layer. It is allowed to retrieve less than availableLength bytes. Note: Not retrieved data will be provided by SoAd with the next call to TcpIp_TcpTransmit (along with new data if available).
Parameters (inout):	None	

Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted, e.g. due to a lack of buffer space or the socket is not connected.
Description:	This service requests transmission of data via TCP to a remote node. The transmission of the data is decoupled. Note: The TCP segment(s) are sent dependent on runtime factors (e.g. receive window) and configuration parameter (e.g. Nagle algorithm) .	

」()

[SWS_TCPIP_00123]⌈ The service `TcpIp_TcpTransmit()` shall transmit data via TCP and the socket specified by SocketId to the connected remote socket according to the sequence diagram specified in section 9.4.」()

[SWS_TCPIP_00124]⌈ DataPtr shall either point to a linear buffer of AvailableLength bytes containing the data for transmission or be a NULL_PTR. For data transmission the service `TcpIp_TcpTransmit()` shall either use all data from the linear buffer if DataPtr is not a NULL_PTR, or retrieve up to AvailableLength data bytes from SoAd by calling SoAd_CopyTxData() one or multiple times in the context of this service otherwise.」()

[SWS_TCPIP_00125]⌈ The service `TcpIp_TcpTransmit()` shall retrieve exactly AvailableLength bytes from the SoAd if the parameter DataPtr is a NULL_PTR and ForceRetrieve is TRUE. (If DataPtr is a NULL_PTR and ForceRetrieve is FALSE, TcpIp may retrieve less data then available).」()

Note: The TCP segment(s) are sent dependent on runtime factors (e.g. receive window) and configuration parameter (e.g. Nagle algorithm).

8.4 Call-back notifications

This is a list of functions provided for other modules. The function prototypes of the callback functions shall be provided in the file TcpIp_Cbk.h.

8.4.1 TcpIp_RxIndication

[SWS_TCPIP_00029]⌈

Service name:	TcpIp_RxIndication

Syntax:	`void TcpIp_RxIndication(`
	`  uint8 CtrlIdx,`
	`  Eth_FrameType FrameType,`
	`  boolean IsBroadcast,`
	`  uint8* PhysAddrPtr,`
	`  uint8* DataPtr,`
	`  uint16 LenByte`
	`)`

Service ID[hex]:	0x14	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CtrlIdx	Index of the EthIf controller.
	FrameType	frame type of received Ethernet frame
	IsBroadcast	parameter to indicate a broadcast frame
	PhysAddrPtr	pointer to Physical source address (MAC address in network byte order) of received Ethernet frame
	DataPtr	Pointer to payload of the received Ethernet frame (i.e. Ethernet header is not provided).
	LenByte	Length of received data.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	By this API service the TCP/IP stack gets an indication and the data of a received frame.	

」()

8.5 Scheduled functions

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

8.5.1 Terms and definitions

For details refer to the chapter 8.5 “Scheduled functions” in *SWS_BSWGeneral.*

8.5.2 TcpIp_MainFunction

[SWS_TCPIP_00026]「

Service name:	TcpIp_MainFunction

Syntax:	`void TcpIp_MainFunction(`
	`  void`
	`)`

Service ID[hex]:	0x15
Description:	Schedules the TCP/IP stack. (Entry point for scheduling)

」()

8.6 Expected Interfaces

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

8.6.1 Mandatory Interfaces

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

[SWS_TCPIP_00027]⌈

API function	*Description*
Dem_ReportErrorStatus	Queues the reported events from the BSW modules (API is only used by BSW modules). The interface has an asynchronous behavior, because the processing of the event is done within the Dem main function. OBD Events Suppression shall be ignored for this computation.
EthIf_GetPhysAddr	Obtains the physical source address used by the indexed controller
EthIf_ProvideTxBuffer	Provides access to a transmit buffer of the specified Ethernet controller.
EthIf_SetPhysAddr	Sets the physical source address used by the indexed controller.
EthIf_Transmit	Triggers transmission of a previously filled transmit buffer
EthSM_TcpIpModeIndication	This service is called by the TcpIp to report the actual TcpIp state (e.g. online, offline).
SoAd_CopyTxData	This service requests to copy data for transmission to the buffer indicated. This call is triggered by TcpIp_Transmit(). Note: The call to SoAd_CopyTxData() may happen in the context of TcpIp_Transmit().
SoAd_LocalIpAddrAssignmentChg	SoAd_LocalIpAssignmentChg() gets called by the TCP/IP stack if an IP address assignment changes (i.e. new address assigned or assigned address becomes invalid).
SoAd_RxIndication	The TCP/IP stack calls this primitive after the reception of data on a socket. The socket identifier along with configuration information determines which module is to be called.
SoAd_TcpAccepted	SoAd_TcpAccepted() gets called if the stack put a socket into the listen mode before (as server) and a peer connected to it (as client). In detail: The TCP/IP stack calls this function after a socket was set into the listen state with TcpIp_TcpListen() and a TCP connection is requested by the peer.
SoAd_TcpConnected	SoAd_TcpConnected() gets called if the stack initiated a TCP connection before (as client) and the peer (the server) acknowledged the connection set up. In detail: The TCP/IP stack calls this function after a socket was requested to connect with TcpIp_TcpConnect() and a TCP connection is confirmed by the peer. The parameter value of SocketId equals the SocketId value of the preceeding TcpIp_TcpConnect() call.
SoAd_TcpIpEvent	SoAd_TcpIpEvent() gets called if the stack encounters a condition described by the values in TcpIpEvent.
SoAd_TxConfirmation	The TCP/IP stack calls this function after the data has been acknowledged by the peer for TCP. Caveats: The upper layer might not be able to determine exactly which data bytes have been confirmed.

⌋()

8.6.2 Optional Interfaces

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

[SWS_TCPIP_00028]⌈

API function	***Description***
Det_ReportError	Service to report development errors.
EthIf_UpdatePhysAddrFilter	Update the physical source address to/from the indexed controller filter. If the Ethernet Controller is not capable to do the filtering, the software has to do this.

⌋()

8.6.3 Configurable interfaces

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

The ServiceID of the functions defined in this chapter are specified at the upper layer module implementing the functions.

8.6.3.1 <Up_PhysAddrTableChg>

[SWS_TCPIP_00143]⌈

Service name:	<Up_PhysAddrTableChg>	

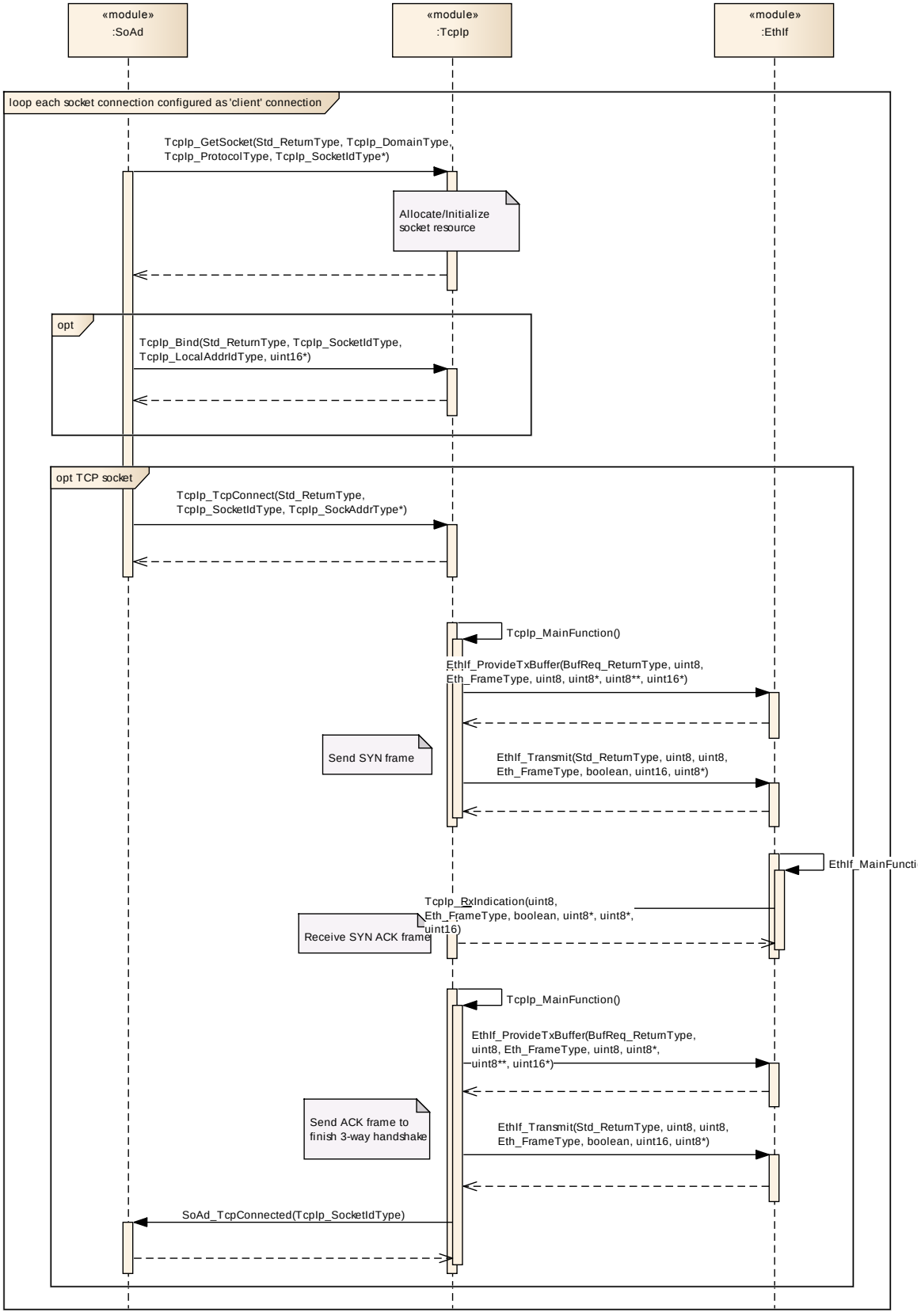
Syntax:	`void <Up_PhysAddrTableChg>(`
	`  uint8 CtrlIdx,`
	`  TcpIp_SockAddrType* IpAddrPtr,`
	`  uint8* PhysAddrPtr,`
	`  boolean valid`
	`)`	

Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CtrlIdx	EthIf controller index of the related ARP table.
	IpAddrPtr	specifies the IP address of the changed ARP table entry
	PhysAddrPtr	specifies the physical address of the changed ARP table entry
	valid	specifies if the ARP table entry is added or changed (TRUE) or has been removed (FALSE)
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	This API is called by TcpIp in case of a change in the ARP table related to the controller specified by CtrlIdx.	

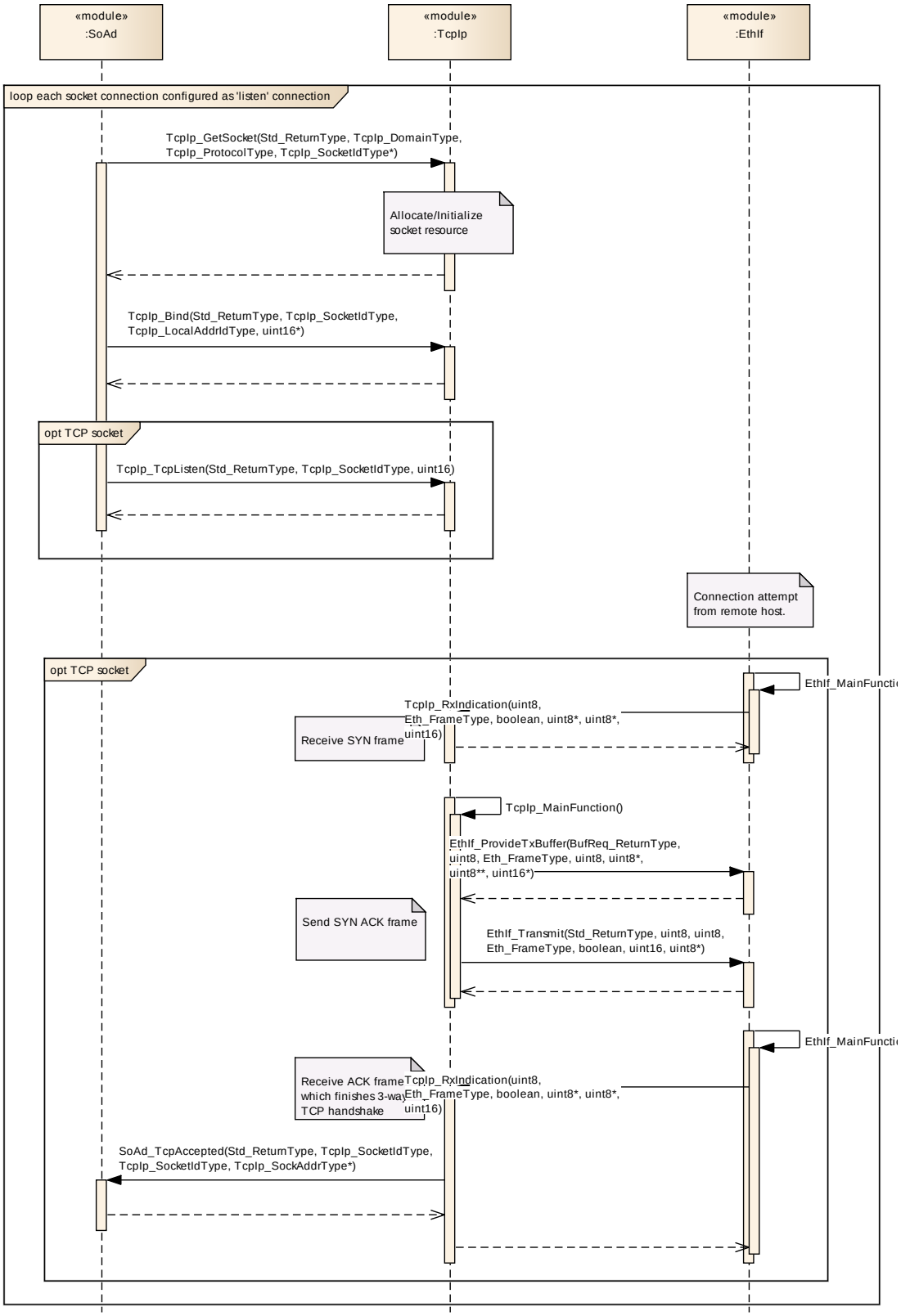
()

9 Sequence diagrams

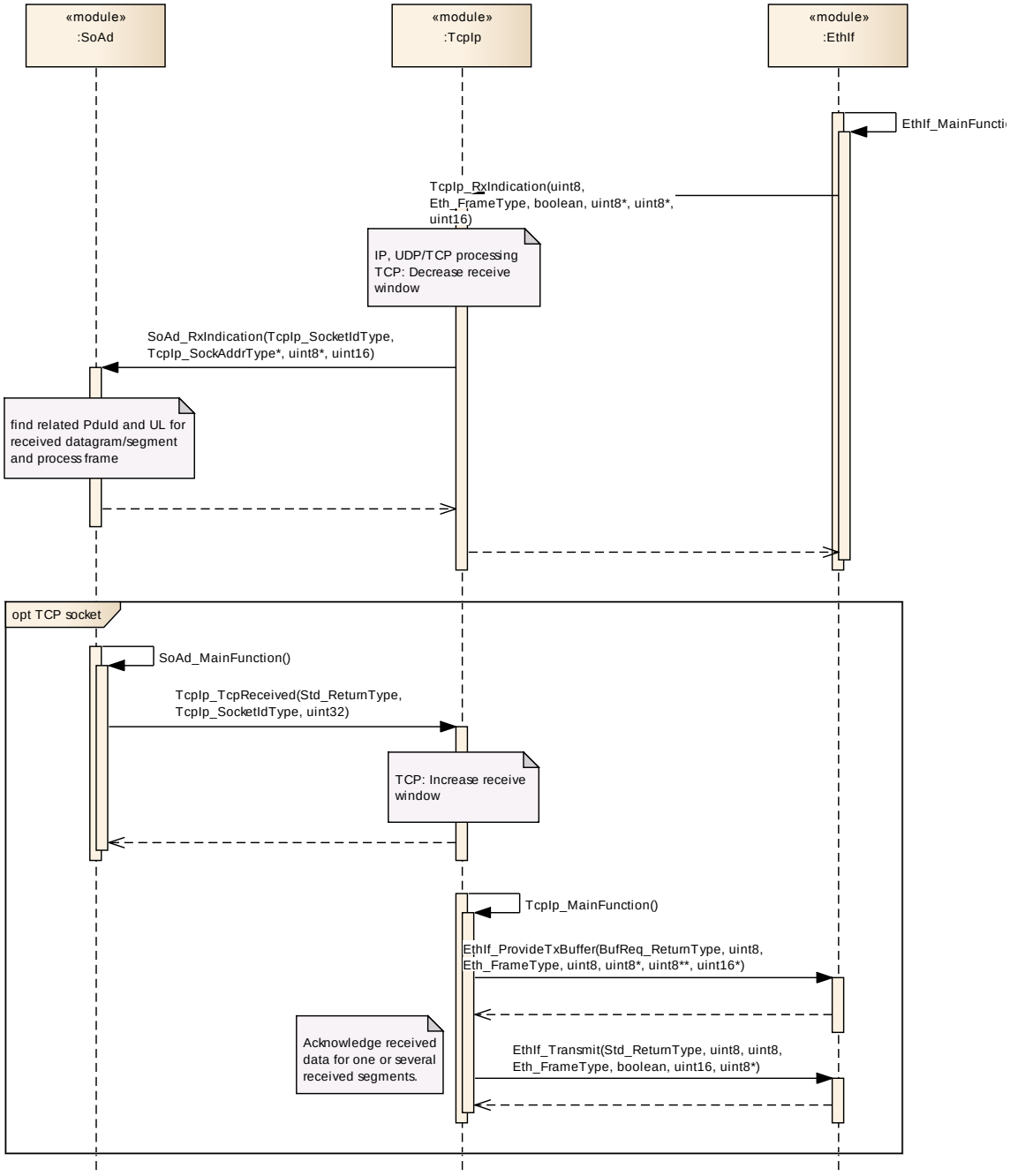
9.1 TCP Connection Setup – Client


«module»
:SoAd
«module»
:TcpIp
«module»
:EthIf
loop each socket connection configured as 'client' connection
TcpIp_GetSocket(Std_ReturnType, TcpIp_DomainType, TcpIp_ProtocolType, TcpIp_SocketIdType*)
Allocate/Initialize socket resource
opt
TcpIp_Bind(Std_ReturnType, TcpIp_SocketIdType, TcpIp_LocalAddrIdType, uint16*)
opt TCP socket
TcpIp_TcpConnect(Std_ReturnType, TcpIp_SocketIdType, TcpIp_SockAddrType*)
TcpIp_MainFunction()
EthIf_ProvideTxBuffer(BufReq_ReturnType, uint8, Eth_FrameType, uint8, uint8*, uint8**, uint16*)
Send SYN frame
EthIf_Transmit(Std_ReturnType, uint8, uint8, Eth_FrameType, boolean, uint16, uint8*)
EthIf_MainFuncti
TcpIp_RxIndication(uint8, Eth_FrameType, boolean, uint8*, uint8*, uint16)
Receive SYN ACK frame
TcpIp_MainFunction()
EthIf_ProvideTxBuffer(BufReq_ReturnType, uint8, Eth_FrameType, uint8, uint8*, uint8**, uint16*)
Send ACK frame to finish 3-way handshake
EthIf_Transmit(Std_ReturnType, uint8, uint8, Eth_FrameType, boolean, uint16, uint8*)
SoAd_TcpConnected(TcpIp_SocketIdType)

9.2 TCP Connection Setup – Server

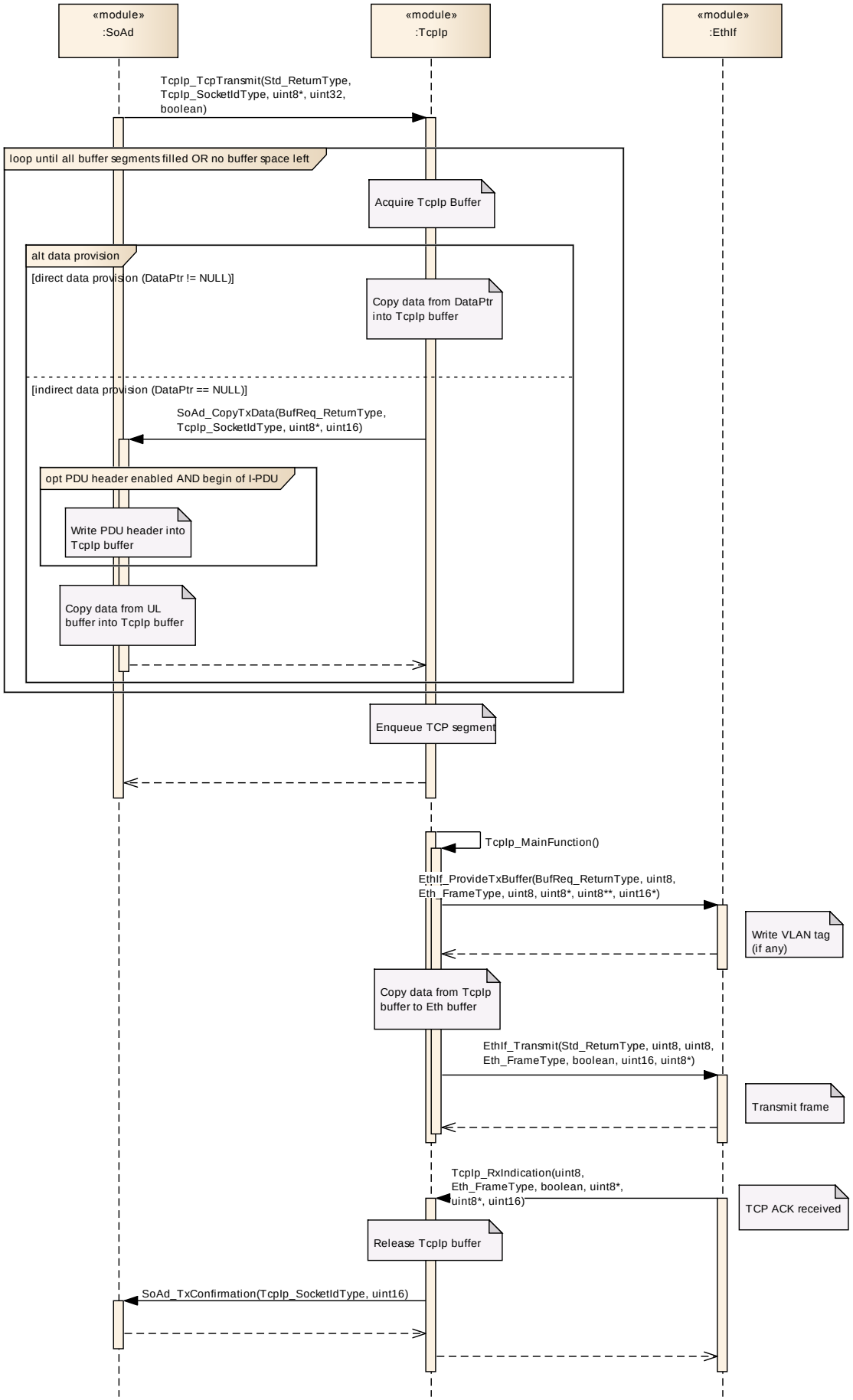


9.3 Reception

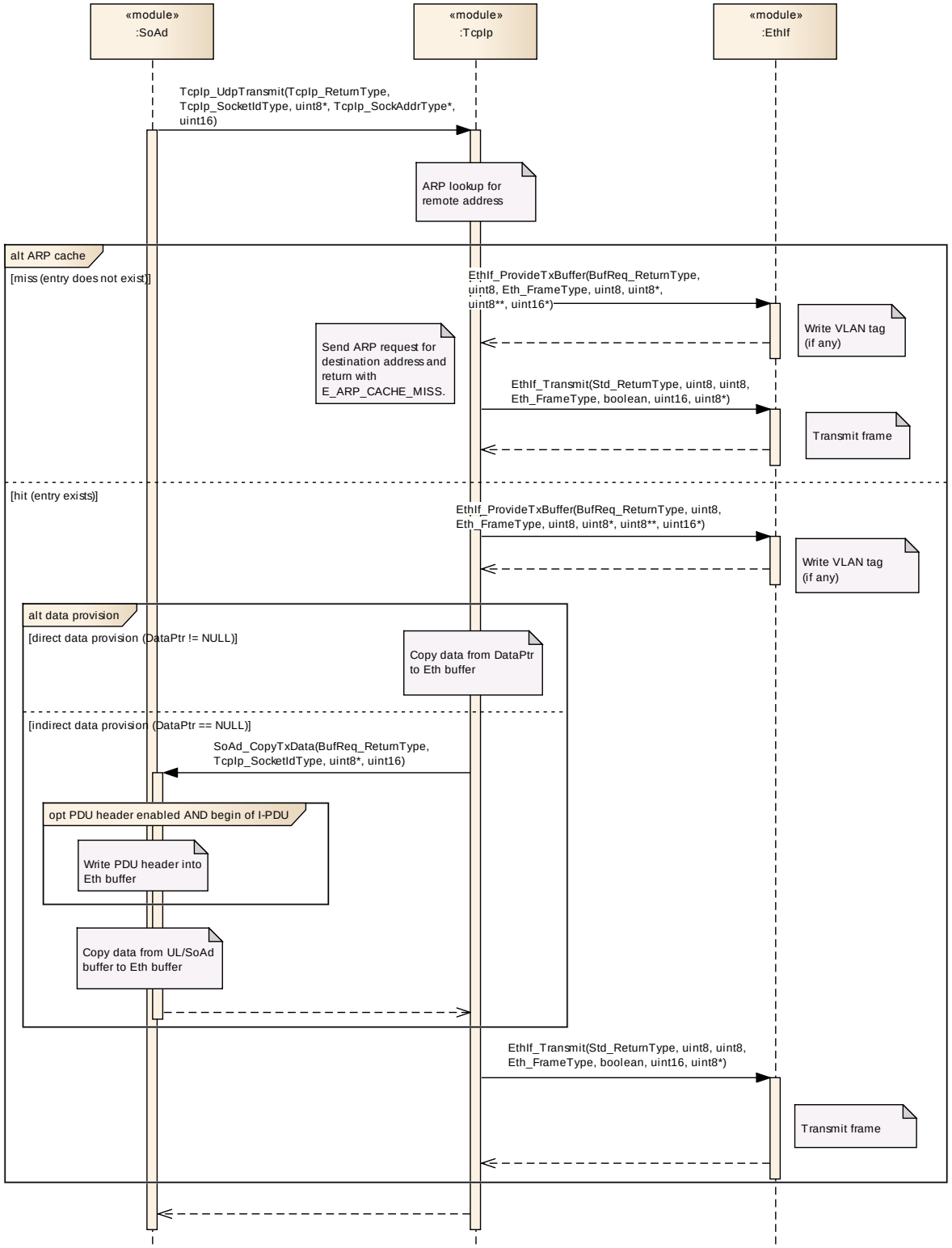


Note: Even it is not shown in the sequence diagram of section 9.3, TcpIp may decouple the data reception if required. E.g. for reassembling of incoming IP datagrams that are fragmented, TcpIp shall copy the received data to a TcpIp buffer and decouple TcpIp_RxIndication() from SoAd_RxIndication().

9.4 Transmission TCP


«module»
:SoAd
«module»
:TcpIp
«module»
:EthIf
TcpIp_TcpTransmit(Std_ReturnType, TcpIp_SocketIdType, uint8*, uint32, boolean)
loop until all buffer segments filled OR no buffer space left
Acquire TcpIp Buffer
alt data provision
[direct data provision (DataPtr != NULL)]
Copy data from DataPtr into TcpIp buffer
[indirect data provision (DataPtr == NULL)]
SoAd_CopyTxData(BufReq_ReturnType, TcpIp_SocketIdType, uint8*, uint16)
opt PDU header enabled AND begin of I-PDU
Write PDU header into TcpIp buffer
Copy data from UL buffer into TcpIp buffer
Enqueue TCP segment
TcpIp_MainFunction()
EthIf_ProvideTxBuffer(BufReq_ReturnType, uint8, Eth_FrameType, uint8, uint8*, uint8**, uint16*)
Write VLAN tag (if any)
Copy data from TcpIp buffer to Eth buffer
EthIf_Transmit(Std_ReturnType, uint8, uint8, Eth_FrameType, boolean, uint16, uint8*)
Transmit frame
TcpIp_RxIndication(uint8, Eth_FrameType, boolean, uint8*, uint8*, uint16)
TCP ACK received
Release TcpIp buffer
SoAd_TxConfirmation(TcpIp_SocketIdType, uint16)

9.5 Transmission UDP



10 Configuration specification

In general, this chapter defines configuration parameters and their clustering into containers. In order to support the specification Chapter 10.1 describes fundamentals. It also specifies a template (table) you shall use for the parameter specification. We intend to leave Chapter 10.1 in the specification to guarantee comprehension.

Chapter 10.2 specifies the structure (containers) and the parameters of the module TcpIp.

Chapter 10.3 specifies published information of the module TcpIp.

10.1 How to read this chapter

For details refer to the chapter 10.1 “Introduction to configuration specification” in *SWS_BSWGeneral.*

10.2 Containers and configuration parameters

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

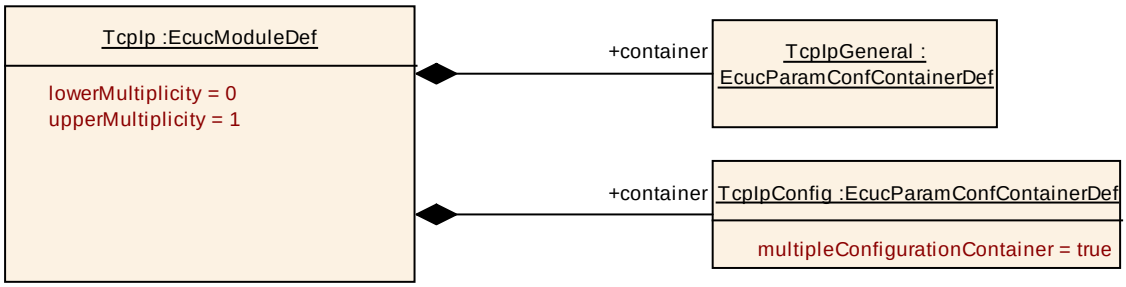
10.2.1 Variants

VARIANT-PRE-COMPILE only supports pre-compile time configurable parameters.
VARIANT-LINK-TIME includes mainly link-time and some pre-compile configurable parameters
VARIANT-POST-BUILD includes post-build-time, link-time and some pre-compile time configurable parameters.

10.2.2 TcpIp

SWS Item	**ECUC_TcpIp_00001 :**
Module Name	*TcpIp*
Module Description	Configuration of the TcpIp (TCP/IP stack) module.

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
TcpIpConfig	1	This container contains the configuration parameters and sub containers of the AUTOSAR TcpIp module. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.
TcpIpGeneral	1	This container is a subcontainer of TcpIp and specifies the general configuration parameters of the TCP/IP stack.



10.2.3 TcpIpGeneral

SWS Item	**ECUC_TcpIp_00002 :**
Container Name	TcpIpGeneral
Description	This container is a subcontainer of TcpIp and specifies the general configuration parameters of the TCP/IP stack.
Configuration Parameters	

SWS Item	**ECUC_TcpIp_00006 :**		
Name	TcpIpArpEnabled {TCPIP_ARP_ENABLED}		
Description	Enables (TRUE) or disables (FALSE) support of ARP (Address Resolution Protocol).		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00011 :**		
Name	TcpIpAutoIpEnabled {TCPIP_AUTOIP_SUPPORT}		
Description	Enables (TRUE) or disables (FALSE) the Auto-IP (automatic private IP addressing) sub-module.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00016 :**		
Name	TcpIpBufferMemory {TCPIP_BUFFER_MEMORY}		
Description	Memory size in bytes reserved for TCP/IP buffers.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00004 :**		
Name	TcpIpDevErrorDetect {TCPIP_DEV_ERROR_DETECT}		
Description	If true then TCP/IP will enable the error-reporting to the Development Error Tracer (DET).		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00010 :**		
Name	TcpIpDhcpClientEnabled {TCPIP_DHCP_CLIENT_SUPPORT}		
Description	Enables (TRUE) or disables (FALSE) the DHCP (Dynamic Host Configuration Protocol) Client.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		

ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00007 :**		
Name	TcpIpIcmpEnabled {TCPIP_ICMP_ENABLED}		
Description	Enables (TRUE) or disabled (FALSE) support of ICMP (Internet Control Message Protocol).		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00018 :**		
Name	TcpIpLocalAddrIpv4EntriesMax {TCPIP_LOCAL_ADDR_IPV4_ENTRIES_MAX}		
Description	Maximum number of LocalAddr table entries for IPv4.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00017 :**		
Name	TcpIpLocalAddrIpv6EntriesMax {TCPIP_LOCAL_ADDR_IPV6_ENTRIES_MAX}		
Description	Maximum number of LocalAddr table entries for IPv6.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00013 :**		
Name	TcpIpMainFunctionPeriod {TCPIP_MAIN_FUNCTION_PERIOD}		
Description	Period of TcpIp_MainFunction in [s].		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0 .. INF		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	

Scope / Dependency	scope: local

SWS Item	**ECUC_TcpIp_00012 :**		
Name	TcpIpPathMtuDiscoveryEnabled {TCPIP_PATHMTU_DISCOVERY_SUPPORT}		
Description	Enables (TRUE) or disables (FALSE) the discovery of the maximum transmission unit on a path according to IETF RfC 1191.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00008 :**		
Name	TcpIpTcpEnabled {TCPIP_TCP_ENABLED}		
Description	Enables (TRUE) or disabled (FALSE) support of TCP (Transmission Control Protocol).		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00014 :**		
Name	TcpIpTcpSocketMax {TCPIP_TCP_SOCKET_MAX}		
Description	Maximum number of TCP sockets		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: local		

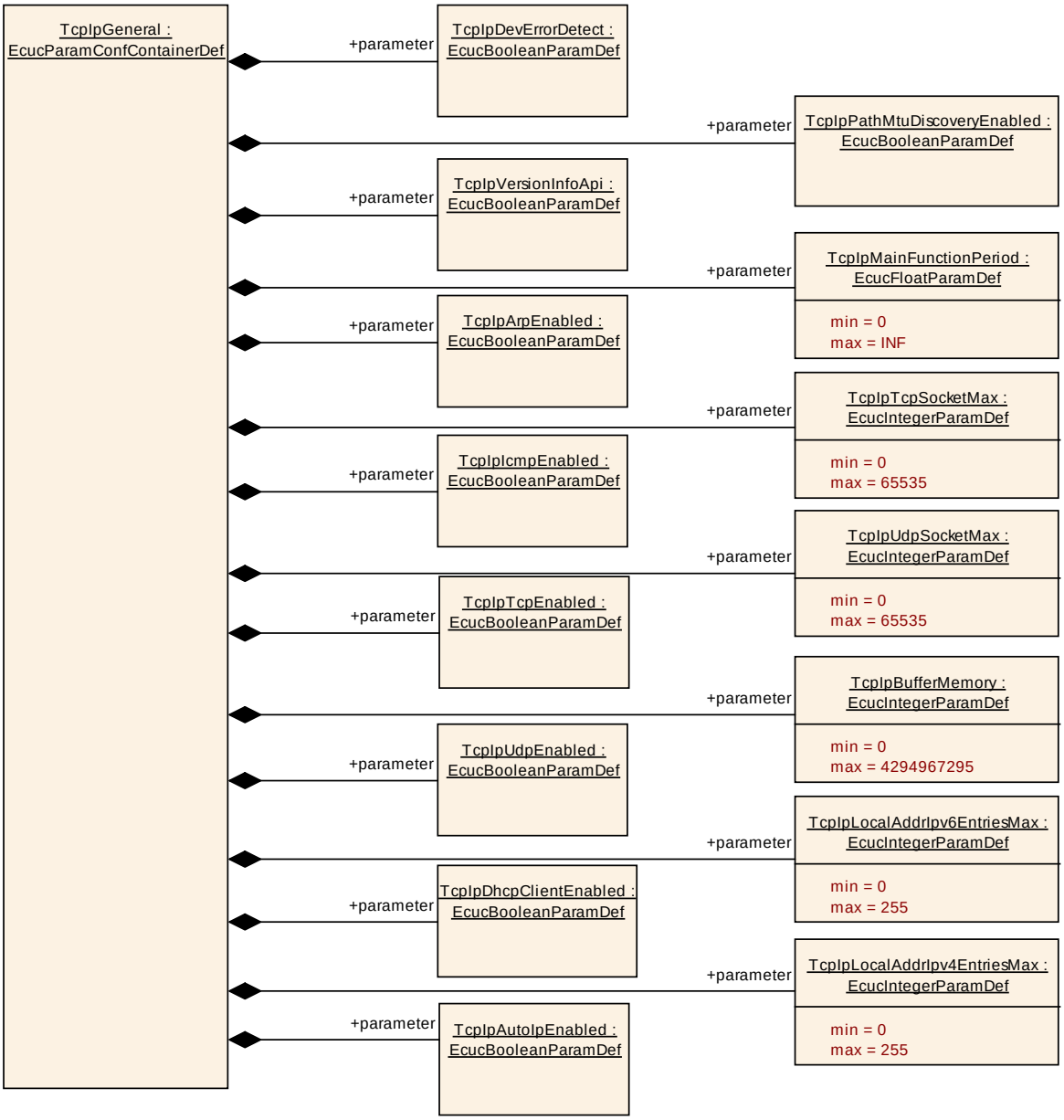
SWS Item	**ECUC_TcpIp_00009 :**		
Name	TcpIpUdpEnabled {TCPIP_UDP_ENABLED}		
Description	Enables (TRUE) or disabled (FALSE) support of UDP (User Datagram Protocol)		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00015 :**
Name	TcpIpUdpSocketMax {TCPIP_UDP_SOCKET_MAX}
Description	Maximum number of UDP sockets.
Multiplicity	1

Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00005 :**		
Name	TcpIpVersionInfoApi {TCPIP_VERSION_INFO_API}		
Description	If true the TcpIp_GetVersionInfo API is available.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers



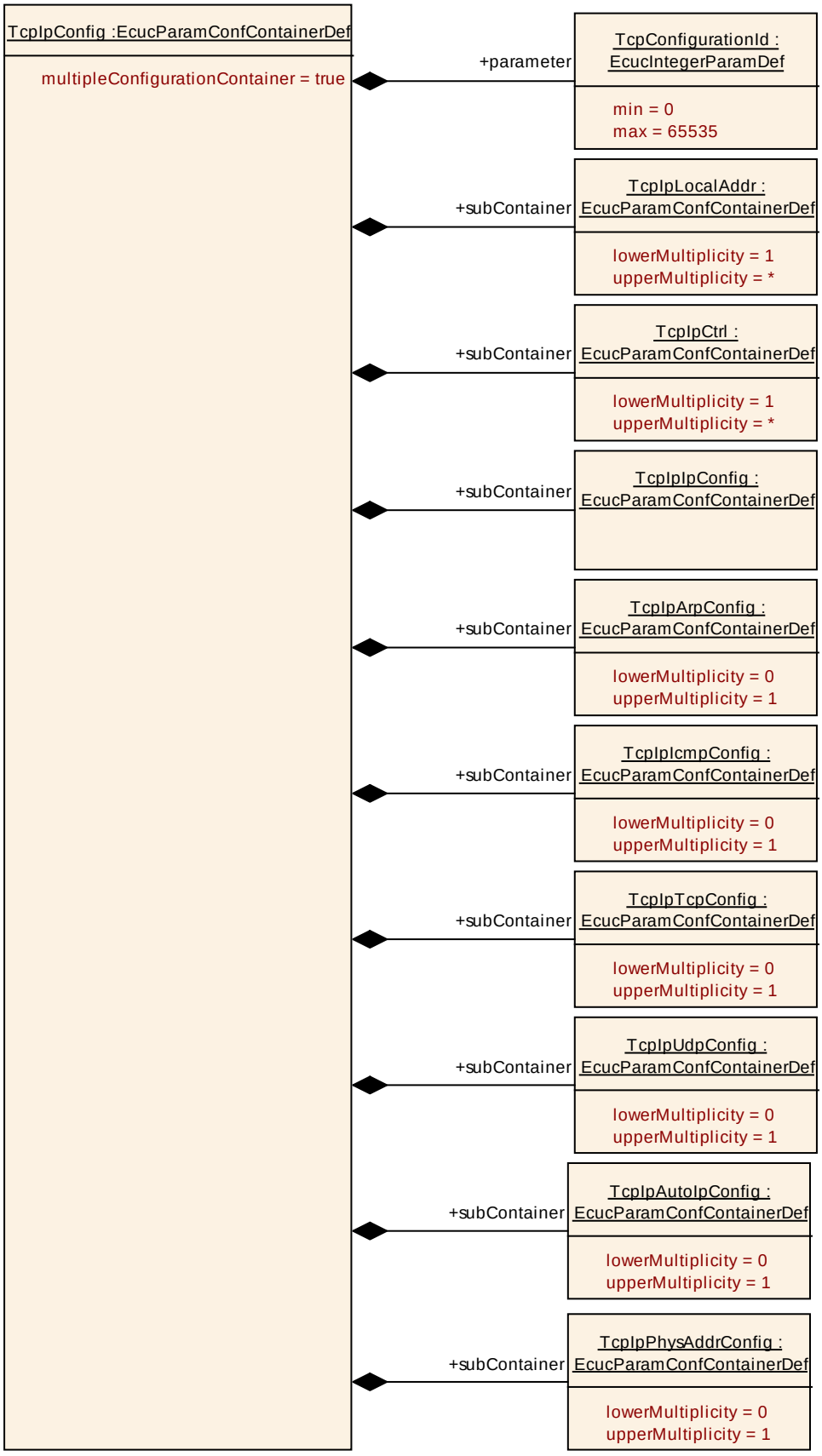
10.2.4 TcpIpConfig

SWS Item	**ECUC_TcpIp_00003 :**
Container Name	TcpIpConfig [Multi Config Container]
Description	This container contains the configuration parameters and sub containers of the AUTOSAR TcpIp module. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.
Configuration Parameters	

SWS Item	**ECUC_TcpIp_00019 :**
Name	TcpConfigurationId {TCPIP_CONFIGURATION_ID}
Description	Identification of the TcpIp configuration.
Multiplicity	1
Type	EcucIntegerParamDef

Range	0 .. 65535		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
TcpIpArpConfig	0..1	Specifies the configuration parameters of the ARP (Address Resolution Protocol) sub-module.
TcpIpAutoIpConfig	0..1	Specifies the configuration parameters of the Auto-IP (automatic private IP addressing) sub-module.
TcpIpCtrl	1..*	Specifies the EthIf controller used for IP communication and TcpIp errors that shall be reported to DEM.
TcpIpIcmpConfig	0..1	Specifies the configuration parameters of the ICMP (Internet Control Message Protocol) sub-module.
TcpIpIpConfig	1	Specifies the configuration parameters of the IP (Internet Protocol) sub-module
TcpIpLocalAddr	1..*	Specifies the local IP (Internet Protocol) addresses used for IP communication.
TcpIpPhysAddrConfig	0..1	Specifies the physical address configuration.
TcpIpTcpConfig	0..1	Specifies the configuration parameters of the TCP (Transmission Control Protocol) sub-module.
TcpIpUdpConfig	0..1	Specifies the configuration parameters of the UDP (User Datagram Protocol) sub-module


TcpIpConfig :EcucParamConfContainerDef
multipleConfigurationContainer = true
+parameter
TcpConfigurationId : EcucIntegerParamDef
min = 0
max = 65535
+subContainer
TcpIpLocalAddr : EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
+subContainer
TcpIpCtrl : EcucParamConfContainerDef
lowerMultiplicity = 1
upperMultiplicity = *
+subContainer
TcpIpIpConfig : EcucParamConfContainerDef
+subContainer
TcpIpArpConfig : EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
TcpIpIcmpConfig : EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
TcpIpTcpConfig : EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
TcpIpUdpConfig : EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
TcpIpAutoIpConfig : EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1
+subContainer
TcpIpPhysAddrConfig : EcucParamConfContainerDef
lowerMultiplicity = 0
upperMultiplicity = 1

10.2.5 TcpIpLocalAddr

SWS Item	**ECUC_TcpIp_00020 :**
Container Name	TcpIpLocalAddr
Description	Specifies the local IP (Internet Protocol) addresses used for IP communication.
Configuration Parameters	

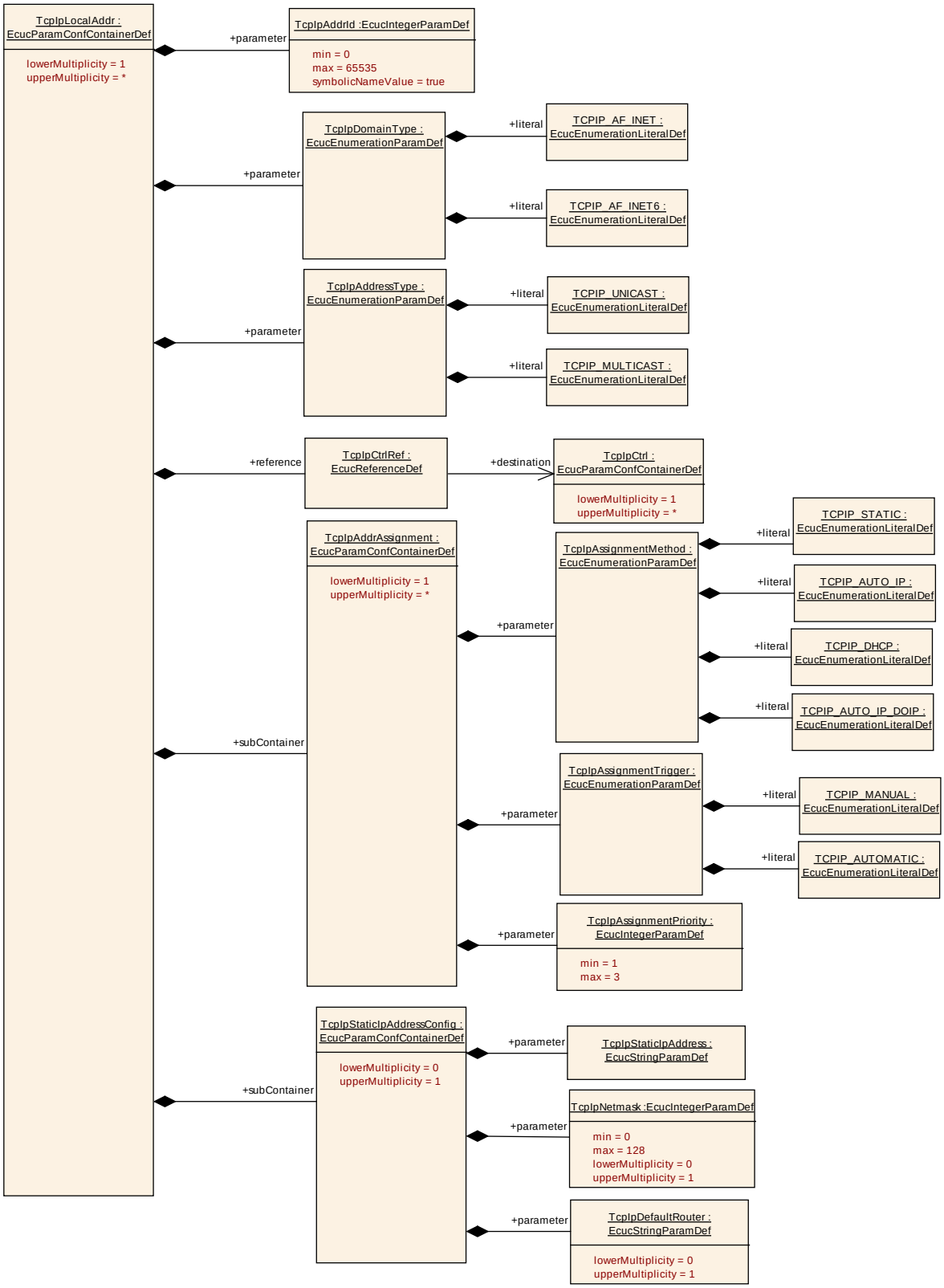
SWS Item	**ECUC_TcpIp_00029 :**		
Name	TcpIpAddrId {TCPIP_ADDR_IDX}		
Description	IP address table identifier assigned by TCP/IP stack.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

SWS Item	**ECUC_TcpIp_00031 :**		
Name	TcpIpAddressType {TCPIP_ADDRESS_TYPE}		
Description	Address type.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	TCPIP_MULTICAST		Multicast address.
	TCPIP_UNICAST		Unicast address
ConfigurationClass	***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_TcpIp_00030 :**		
Name	TcpIpDomainType {TCPIP_DOMAIN_TYPE}		
Description	Address family.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	TCPIP_AF_INET		IPv4 address
	TCPIP_AF_INET6		IPv6 address
ConfigurationClass	***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_TcpIp_00032 :**		
Name	TcpIpCtrlRef {TCP_IP_CTRL_REF}		
Description	Reference to a TcpIpCtrl specifying the EthIf Controller where the IP address shall be assigned and DEM errors that shall be reported in case of an error on this controller.		
Multiplicity	1		
Type	Reference to [TcpIpCtrl]		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
TcpIpAddrAssignment	1..*	This container is a subcontainer of TcpIpLocalAddr and specifies the assignment policy for the IP address.
TcpIpStaticIpAddressConfig	0..1	This container is a subcontainer of TcpIpLocalAddr and specifies a static IP address including directly related parameters.



10.2.6 TcpIpPhysAddrConfig

SWS Item	ECUC_TcpIp_00083 :

Container Name	TcpIpPhysAddrConfig
Description	Specifies the physical address configuration.
Configuration Parameters	

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
TcpIpPhysAddrChgHandler	0..1	This container is a subcontainer of TcpIpPhysAddrConfig and specifies the configuration parameters for physical address change handler.

10.2.7 TcpIpPhysAddrChgHandler

SWS Item	**ECUC_TcpIp_00084 :**
Container Name	TcpIpPhysAddrChgHandler
Description	This container is a subcontainer of TcpIpPhysAddrConfig and specifies the configuration parameters for physical address change handler.
Configuration Parameters	

SWS Item	**ECUC_TcpIp_00085 :**		
Name	TcpIpPhysAddrChgHandlerHeaderFileName		
Description	This parameter specifies the name of the header file containing the definition of the physical address change handler function.		
Multiplicity	1		
Type	EcucStringParamDef		
Default value	--		
maxLength	32		
minLength	1		
regularExpression	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00086 :**		
Name	TcpIpPhysAddrChgHandlerName		
Description	This parameter defines the name of the physical address change function <Up>_PhysAddrTableChg.		
Multiplicity	1		
Type	EcucFunctionNameDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.8 TcpIpAddrAssignment

SWS Item	**ECUC_TcpIp_00033 :**
Container Name	TcpIpAddrAssignment
Description	This container is a subcontainer of TcpIpLocalAddr and specifies the assignment policy for the IP address.
Configuration Parameters	

SWS Item	**ECUC_TcpIp_00035 :**		
Name	TcpIpAssignmentMethod {TCPIP_ASSIGNMENT_METHOD}		
Description	Method of address assignment		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	TCPIP_AUTO_IP	Address assignment via Auto-IP, timing according to IETF RFC 3927 – Dynamic Configuration of IPv4 Link-Local Addresses	
	TCPIP_AUTO_IP_DOIP	Address assignment via Auto-IP; timing according to ISO 13400 (DoIP)	
	TCPIP_DHCP	Address assignment via DHCP	
	TCPIP_STATIC	Static address assignment	
ConfigurationClass	***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_TcpIp_00037 :**		
Name	TcpIpAssignmentPriority {TCPIP_ASSIGNMENT_PRIORITY}		
Description	Priority of assignment (1 is highest). If a new address from an assignment method with a higher priority is available, it overwrites the IP address previously assigned by an assignment method with a lower priority.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 3		
Default value	--		
ConfigurationClass	***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_TcpIp_00036 :**		
Name	TcpIpAssignmentTrigger {TCPIP_ASSIGNMENT_TRIGGER}		
Description	Trigger of address assignment.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	TCPIP_AUTOMATIC	Assignment shall be initiated automatically by TCP/IP stack.	
	TCPIP_MANUAL	Assignment shall be initiated manually via TcpIp_StartIpAddrAssigment().	
ConfigurationClass	***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

10.2.9 TcpIpStaticIpAddressConfig

SWS Item	**ECUC_TcpIp_00034 :**
Container Name	TcpIpStaticIpAddressConfig
Description	This container is a subcontainer of TcpIpLocalAddr and specifies a static IP address including directly related parameters.
Configuration Parameters	

SWS Item	**ECUC_TcpIp_00040 :**		
Name	TcpIpDefaultRouter {TCPIP_DEFAULT_ROUTER}		
Description	IP address of default router (gateway)		
Multiplicity	0..1		
Type	EcucStringParamDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
ConfigurationClass	***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_TcpIp_00039 :**		
Name	TcpIpNetmask {TCPIP_NETMASK}		
Description	Network mask of IPv4 address or address prefix of IPv6 address in CIDR Notation, i.e. decimal value between 0 and 32 (IPv4) or 0 and 128 (IPv6) that describes the number of significant bits defining the network number or prefix of an IP address.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 128		
Default value	--		
ConfigurationClass	***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_TcpIp_00038 :**		
Name	TcpIpStaticIpAddress {TCPIP_STATIC_IP_ADDRESS}		
Description	Static IP Address. To specify any IP address for a certain EthIfCtrl, â€œANYâ€• has to be set as wildcard. See TcpIp_Bind() for more details.		
Multiplicity	1		
Type	EcucStringParamDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
ConfigurationClass	***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

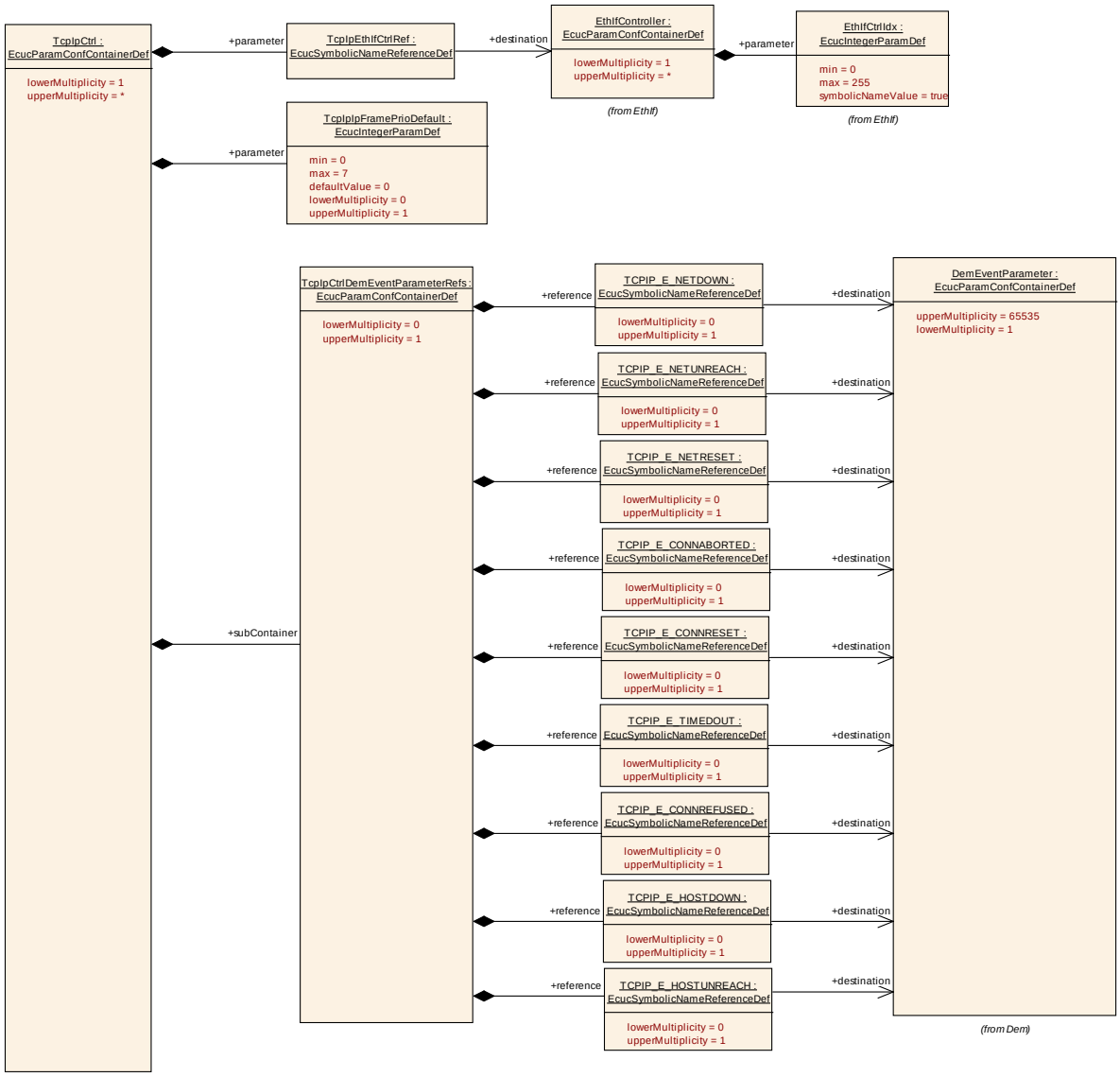
10.2.10 TcpIpCtrl

SWS Item	**ECUC_TcpIp_00021 :**
Container Name	TcpIpCtrl
Description	Specifies the EthIf controller used for IP communication and TcpIp errors that shall be reported to DEM.
Configuration Parameters	

SWS Item	**ECUC_TcpIp_00081 :**		
Name	TcpIpIpFramePrioDefault {TCPIP_IP_NUM_REASS_DGRAMS}		
Description	Specifies the default value for the frame priority used by all sockets. Note: the value can be changed for each socket individually via TcpIp_ChangeParameter() service. If this optional parameter is not available, 0 is used as default priority.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 7		
Default value	0		
ConfigurationClass	*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_TcpIp_00041 :**		
Name	TcpIpEthIfCtrlRef {TCPIP_ETHIF_CTRL_REF}		
Description	Reference to EthIf controller where the IP address shall be assigned.		
Multiplicity	1		
Type	Symbolic name reference to [EthIfController]		
ConfigurationClass	*Pre-compile time*	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

Included Containers		
Container Name	*Multiplicity*	*Scope / Dependency*
TcpIpCtrlDemEventParameterRefs	0..1	Specifies references to DEMEventParameter elements for TcpIp errors that shall be detected on the EthIf controller and reported to DEM.



10.2.11 TcpIpCtrlDemEventParameterRefs

SWS Item	**ECUC_TcpIp_00042 :**
Container Name	TcpIpCtrlDemEventParameterRefs
Description	This container is a subcontainer of TcpIpCtrl and specifies the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding TcpIp error occurs for communication on the EthIf Controller. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.
Configuration Parameters	

SWS Item	**ECUC_TcpIp_00046 :**
Name	TCPIP_E_CONNABORTED
Description	Reference to the DemEventParameter which shall be issued when the error "Connection aborted by TcpIp stack because of an errorâ€• has occurred.

Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
ConfigurationClass	***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_TcpIp_00049 :**		
Name	TCPIP_E_CONNREFUSED		
Description	Reference to the DemEventParameter which shall be issued when the error "Connection refusedâ€• has occurred.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
ConfigurationClass	***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_TcpIp_00047 :**		
Name	TCPIP_E_CONNRESET		
Description	Reference to the DemEventParameter which shall be issued when the error "Connection reset by peerâ€• has occurred.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
ConfigurationClass	***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_TcpIp_00050 :**		
Name	TCPIP_E_HOSTDOWN		
Description	Reference to the DemEventParameter which shall be issued when the error "Host is downâ€• has occurred.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
ConfigurationClass	***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_TcpIp_00051 :**		
Name	TCPIP_E_HOSTUNREACH		
Description	Reference to the DemEventParameter which shall be issued when the error "No route to hostâ€• has occurred.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
ConfigurationClass	***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_TcpIp_00043 :**		
Name	TCPIP_E_NETDOWN		
Description	Reference to the DemEventParameter which shall be issued when the error "Network is downâ€• has occurred.		
Multiplicity	0..1		

Type	Symbolic name reference to [DemEventParameter]		
ConfigurationClass	***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_TcpIp_00045 :**		
Name	TCPIP_E_NETRESET		
Description	Reference to the DemEventParameter which shall be issued when the error "Network dropped connection on resetâ€• has occurred.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
ConfigurationClass	***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_TcpIp_00044 :**		
Name	TCPIP_E_NETUNREACH		
Description	Reference to the DemEventParameter which shall be issued when the error "Network is unreachableâ€• has occurred.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
ConfigurationClass	***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_TcpIp_00048 :**		
Name	TCPIP_E_TIMEDOUT		
Description	Reference to the DemEventParameter which shall be issued when the error "Operation timed outâ€• has occurred.		
Multiplicity	0..1		
Type	Symbolic name reference to [DemEventParameter]		
ConfigurationClass	***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

10.2.12 TcpIpIpConfig

SWS Item	**ECUC_TcpIp_00022 :**
Container Name	TcpIpIpConfig
Description	Specifies the configuration parameters of the IP (Internet Protocol) sub-module
Configuration Parameters	

SWS Item	**ECUC_TcpIp_00077 :**
Name	TcpIpIpFragmentationRxEnabled {TCPIP_IP_FRAGMENTATION_RX_ENABLED}

Description	Enables (TRUE) or disables (FALSE) support for reassembling of incoming datagrams that are fragmented according to IETF RFC 815 (IP Datagram Reassembly Algorithms).		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

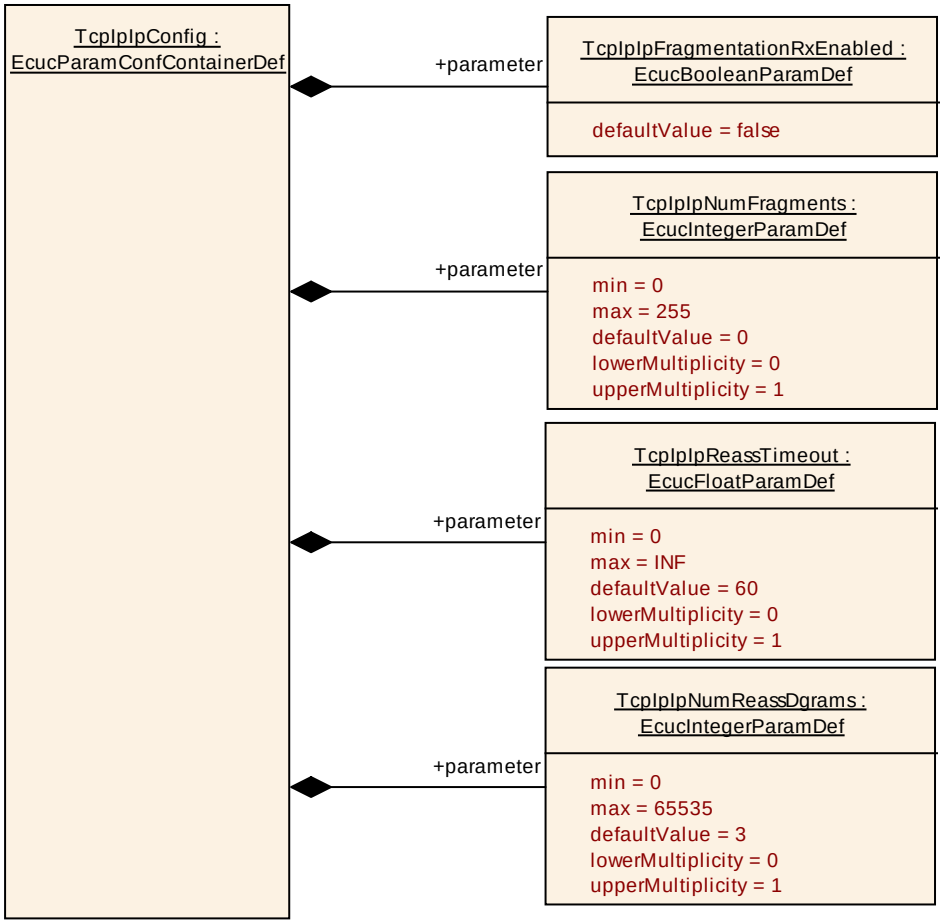
SWS Item	**ECUC_TcpIp_00078 :**		
Name	TcpIpIpNumFragments {TCPIP_IP_NUM_FRAGMENTS}		
Description	Specifies the maximum number of IP fragments per datagram. Note: this parameter is only relevant if TcpIpIpFragmentationRxEnabled is TRUE.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	0		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local dependency: TcpIpIpFragmentationRxEnabled		

SWS Item	**ECUC_TcpIp_00080 :**		
Name	TcpIpIpNumReassDgrams {TCPIP_IP_NUM_REASS_DGRAMS}		
Description	Specifies the maximum number of fragmented IP datagrams that can be reassembled in parallel. Note: this parameter is only relevant if TcpIpIpFragmentationRxEnabled is TRUE.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	3		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local dependency: TcpIpIpFragmentationRxEnabled		

SWS Item	**ECUC_TcpIp_00079 :**		
Name	TcpIpIpReassTimeout {TCPIP_IP_REASS_TIMEOUT}		
Description	Specifies the timeout in [s] after which an incomplete datagram gets discarded. Note: this parameter is only relevant if TcpIpIpFragmentationRxEnabled is TRUE.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	0 .. INF		
Default value	60		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

	dependency: TcpIpIpFragmentationRxEnabled

No Included Containers



10.2.13 TcpIpArpConfig

SWS Item	**ECUC_TcpIp_00023 :**
Container Name	TcpIpArpConfig
Description	Specifies the configuration parameters of the ARP (Address Resolution Protocol) sub-module.
Configuration Parameters	

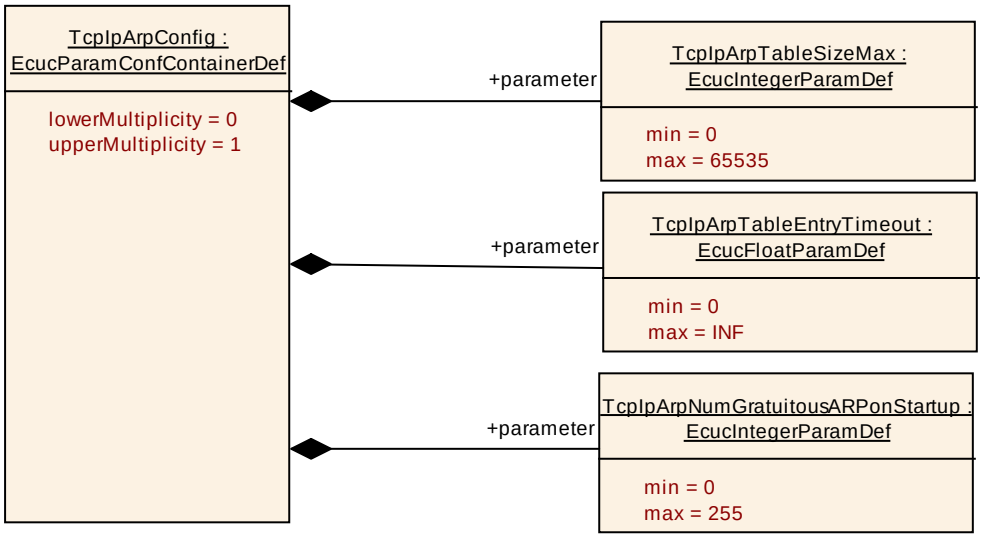
SWS Item	**ECUC_TcpIp_00054 :**	
Name	TcpIpArpNumGratuitousARPonStartup {TCPIP_ARP_NUM_GRATUITOUS_AR_PON_STARTUP}	
Description	Specifies the number of gratuitous ARP replies which shall be sent on assignment of a new IP address.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 255	
Default value	--	

ConfigurationClass	***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_TcpIp_00053 :**		
Name	TcpIpArpTableEntryTimeout {TCPIP_ARP_TABLE_ENTRY_TIMEOUT}		
Description	Timeout in seconds after which an unused ARP entry is removed.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0 .. INF		
Default value	--		
ConfigurationClass	***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_TcpIp_00052 :**		
Name	TcpIpArpTableSizeMax {TCPIP_ARP_TABLE_SIZE_MAX}		
Description	Maximum number of entries in the ARP table.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

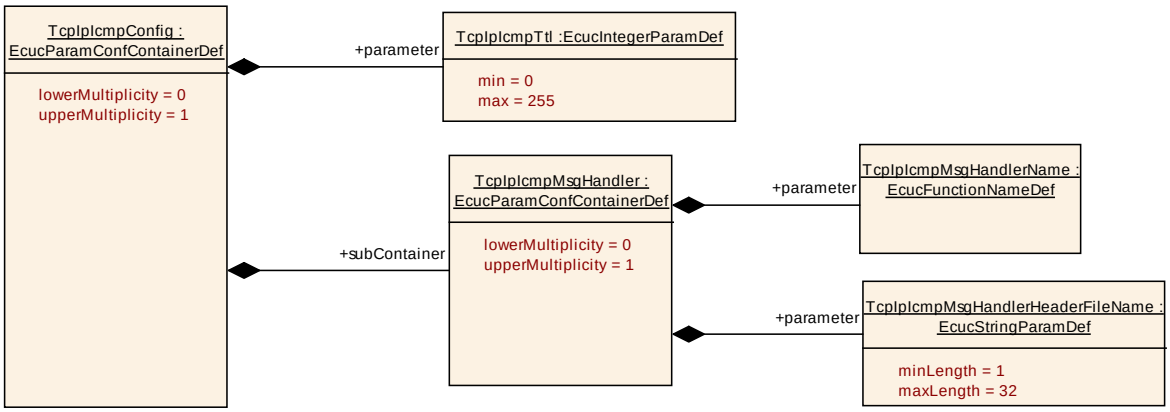


10.2.14 TcpIpIcmpConfig

SWS Item	ECUC_TcpIp_00024 :
Container Name	TcpIpIcmpConfig
Description	Specifies the configuration parameters of the ICMP (Internet Control Message Protocol) sub-module.
Configuration Parameters	

SWS Item	ECUC_TcpIp_00055 :		
Name	TcpIpIcmpTtl {TCPIP_ICMP_TTL}		
Description	Default Time-to-live value of outgoing ICMP packets.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
ConfigurationClass	Pre-compile time	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local		

Included Containers		
Container Name	Multiplicity	Scope / Dependency
TcpIpIcmpMsgHandler	0..1	This container is a subcontainer of TcpIpIcmpConfig and specifies the configuration parameters for the ICMP message handler.



10.2.15 TcpIpIcmpMsgHandler

SWS Item	ECUC_TcpIp_00056 :
Container Name	TcpIpIcmpMsgHandler
Description	This container is a subcontainer of TcpIpIcmpConfig and specifies the configuration parameters for the ICMP message handler.
Configuration Parameters	

SWS Item	ECUC_TcpIp_00058 :
Name	TcpIpIcmpMsgHandlerHeaderFileName {TCPIP_ICMP_MSG_HANDLER_HEADER_FILE_NAME}
Description	This parameter specifies the name of the header file containing the definition of the ICMP message handler function.
Multiplicity	1

Type	EcucStringParamDef		
Default value	--		
maxLength	32		
minLength	1		
regularExpression	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00057 :**		
Name	TcpIpIcmpMsgHandlerName {TCPIP_ICMP_MSG_HANDLER_NAME}		
Description	This parameter defines the name of the ICMP message handler function <User_IcmpMsgHandler>.		
Multiplicity	1		
Type	EcucFunctionNameDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME, VARIANT-POST-BUILD
	Post-build time	--	
Scope / Dependency	scope: local		

No Included Containers

10.2.16 TcpIpTcpConfig

SWS Item	**ECUC_TcpIp_00025 :**
Container Name	TcpIpTcpConfig
Description	Specifies the configuration parameters of the TCP (Transmission Control Protocol) sub-module.
Configuration Parameters	

SWS Item	**ECUC_TcpIp_00061 :**		
Name	TcpIpTcpCongestionAvoidanceEnabled {TCPIP_TCP_CONGESTIONAVOIDANCE_ENABLED}		
Description	Enables (TRUE) or disables (FALSE) support of TCP congestion avoidance algorithm according to IETF RFC 5681.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00063 :**
Name	TcpIpTcpFastRecoveryEnabled

	{TCPIP_TCP_FASTRECOVERY_ENABLED}		
Description	Enables (TRUE) or disables (FALSE) support of TCP Fast Recovery according to IETF RFC 5681.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00062 :**		
Name	TcpIpTcpFastRetransmitEnabled {TCPIP_TCP_FASTRETRANSMIT_ENABLED}		
Description	Enables (TRUE) or disables (FALSE) support of TCP Fast Retransmission according to IETF RFC 5681.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00066 :**		
Name	TcpIpTcpFinWait2Timeout {TCPIP_TCP_FIN_WAIT_TIMEOUT}		
Description	Timeout in [s] to receive a FIN from the remote node (after this node has initiated connection termination), i.e. maximum time waiting in FINWAIT-2 for a connection termination request from the remote TCP.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0 .. INF		
Default value	--		
ConfigurationClass	***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_TcpIp_00082 :**		
Name	TcpIpTcpKeepAliveEnabled {TCPIP_TCP_KEEP_ALIVE_ENABLED}		
Description	Enables (TRUE) or disables (FALSE) TCP Keep Alive Probes according to IETF RFC 1122 chapter 4.2.3.6		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00070 :**		
Name	TcpIpTcpKeepAliveInterval {TCPIP_TCP_KEEP_ALIVE_INTERVAL}		
Description	Specifies the interval in [s] between subsequent keepalive probes.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0 .. INF		

Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local dependency: TcpIpTcpKeepAliveEnabled		

SWS Item	**ECUC_TcpIp_00071 :**		
Name	TcpIpTcpKeepAliveProbesMax {TCPIP_TCP_KEEP_ALIVE_PROBES_MAX}		
Description	Maximum number of times that a TCP segment is retransmitted.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: local dependency: TcpIpTcpKeepAliveEnabled		

SWS Item	**ECUC_TcpIp_00069 :**		
Name	TcpIpTcpMaxRtx {TCPIP_TCP_MAX_RTX}		
Description	Maximum number of times that a TCP segment is retransmitted.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
ConfigurationClass	***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_TcpIp_00067 :**		
Name	TcpIpTcpMsl {TCPIP_TCP_MSL}		
Description	Maximum segment lifetime in [s]. (Note: TIME-WAIT = 2 x TcpIpTcpMsl â€“ to ensure that the remote node received the acknowledgment to its connection termination request.)		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0 .. INF		
Default value	--		
ConfigurationClass	***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_TcpIp_00059 :**		
Name	TcpIpTcpNagleEnabled {TCPIP_TCP_NAGLE_ENABLED}		
Description	Enables (TRUE) or disables (FALSE) support of Nagleâ€™s algorithm according to IETF RFC 896. If enabled the Nagleâ€™s algorithm is activated per default for all TCP sockets, but can be deactivated via TcpIp_ChangeParameter() API.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		

ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

SWS Item	**ECUC_TcpIp_00073 :**		
Name	TcpIpTcpReceiveWindowMax {TCPIP_RECEIVE_WINDOW_MAX}		
Description	Default value of maximum receive window in bytes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default value	--		
ConfigurationClass	***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_TcpIp_00068 :**		
Name	TcpIpTcpRetransmissionTimeout {TCPIP_TCP_RETRANSMISSION_TIMEOUT}		
Description	Timeout in [s] before an unacknowledged TCP segment is sent again.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0 .. INF		
Default value	--		
ConfigurationClass	***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_TcpIp_00060 :**		
Name	TcpIpTcpSlowStartEnabled {TCPIP_TCP_SLOWSTART_ENABLED}		
Description	Enables (TRUE) or disables (FALSE) support of TCP slow start algorithm according to IETF RFC 5681.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: local		

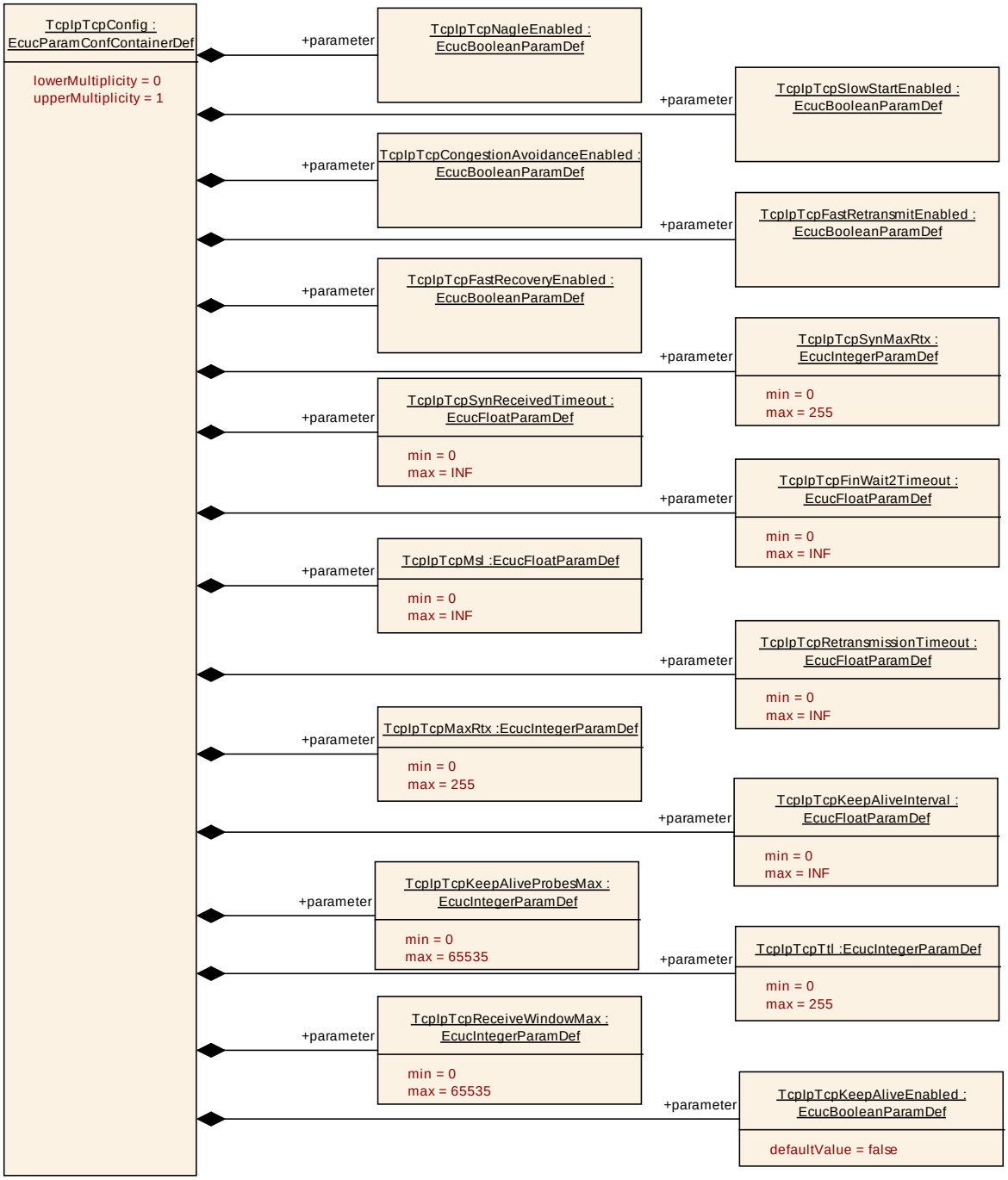
SWS Item	**ECUC_TcpIp_00064 :**		
Name	TcpIpTcpSynMaxRtx {TCPIP_TCP_SYN_MAX_RTX}		
Description	Maximum number of times that a TCP SYN is retransmitted. Note: SYN will be retried after TcpIpTcpRetransmissionTimeout. The connection will be dropped if no matching connection request has been received after the last TCP SYN has been sent and TcpIpTcpRetransmissionTimeout has been expired.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
ConfigurationClass	***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_TcpIp_00065 :**		
Name	TcpIpTcpSynReceivedTimeout {TCPIP_TCP_SYN_RECEIVED_TIMEOUT}		
Description	Timeout in [s] to complete a remotely initiated TCP connection establishment, i.e. maximum time waiting in SYN-RECEIVED for a confirming connection request acknowledgment after having both received and sent a connection request.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0 .. INF		
Default value	--		
ConfigurationClass	***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_TcpIp_00072 :**		
Name	TcpIpTcpTtl {TCPIP_TCP_TTL}		
Description	Default Time-to-live value of outgoing TCP packets.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
ConfigurationClass	***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



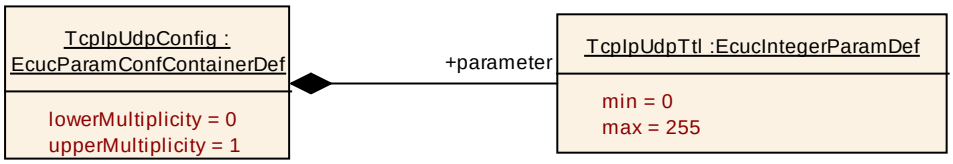
10.2.17 TcpIpUdpConfig

SWS Item	**ECUC_TcpIp_00026 :**
Container Name	TcpIpUdpConfig
Description	Specifies the configuration parameters of the UDP (User Datagram Protocol) sub-module
Configuration Parameters	

SWS Item	**ECUC_TcpIp_00075 :**
Name	TcpIpUdpTtl {TCPIP_UDP_TTL}

Description	Default Time-to-live value of outgoing UDP packets.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default value	--		
ConfigurationClass	***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

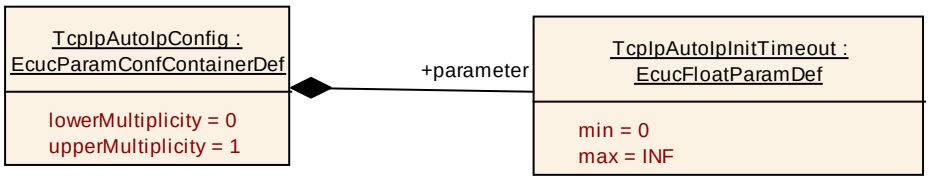


10.2.18 TcpIpAutoIpConfig

SWS Item	**ECUC_TcpIp_00028 :**
Container Name	TcpIpAutoIpConfig
Description	Specifies the configuration parameters of the Auto-IP (automatic private IP addressing) sub-module.
Configuration Parameters	

SWS Item	**ECUC_TcpIp_00074 :**		
Name	TcpIpAutoIpInitTimeout {TCPIP_AUTO_IP_INIT_TIMEOUT}		
Description	The time in seconds Auto-IP waits at startup, before beginning with ARP probing. This delay is used to give DHCP time to acquire a lease in case a DHCP server is present.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0 .. INF		
Default value	--		
ConfigurationClass	***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



10.3 Published Information

For details refer to the chapter 10.3 “Published Information” in *SWS_BSWGeneral.*