Document Title	Specification of Linklayer Sdu Routing Module
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	1094

Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R23-11

Document Change History			
Date	**Release**	**Changed by**	**Description**
2023-11-23	R23-11	AUTOSAR Release Management	• Initial release

Disclaimer

Contents

Known Limitations

- No content

1 Introduction and functional overview

This specification describes the functionality, API and the configuration for the AUTOSAR Basic Software module L-SDU Router. The L-SDU Router module provides services for routing L-SDUs (Link Layer Service Data Units) using the following module types:

- Communication interface modules, that use the `<Provider:Up>` or `<Provider:Lo>` APIs, e.g. EthIf, IEEE1722Tp;

The routing of L-SDUs is performed based on statically defined L-SDU identifiers. Thus, no L-SDU is routed dynamically during run-time, e.g. dependent on its payload.

The location of related modules can be "upper" (e.g. IEEE1722Tp) and/or "lower" (e.g. EthIf).

Please note:

- R23-11: The L-SDU Router act as pass-through module between the Ethernet Interface and the IEEE1722Tp module.
- After R23-11: The L-SDU Router will be extended to act as mandatory upper layer of all communication interface modules and as mandatory lower layer of all direct linked modules: PduR, <Bus>Nm, <Bus>Tp, <Bus>TSyn, XCPon<Bus>. The L-SDU Router will be extended to provide similar gateway functionality as the PDU Router.

The L-SDU Router module is based on a generic approach of interfaced modules. The module that is interfaced is configured in the L-SDU Router module configuration.

1.1 AUTOSAR architecture

The L-SDU Router act as pass-through module between the Ethernet Interface and the IEEE1722Tp module(see [1]).

Please note: After R23-11 the L-SDU Router module will be extended to act as central module in the AUTOSAR communicaton structure between the communication hardware abstraction layer and communication service layer.

Figure 1.1 gives an overview of the AUTOSAR communication structure

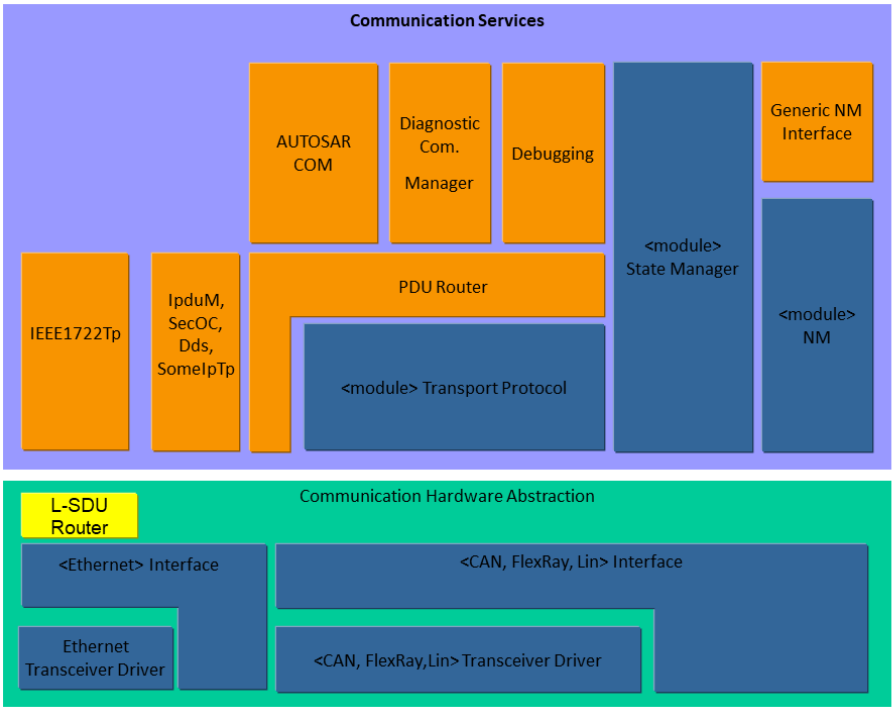

Figure 1.1: Communication Structure

1.2 L-SDU Router module function overview

The detailed L-SDU Router module structure is shown in Figure 1.2.

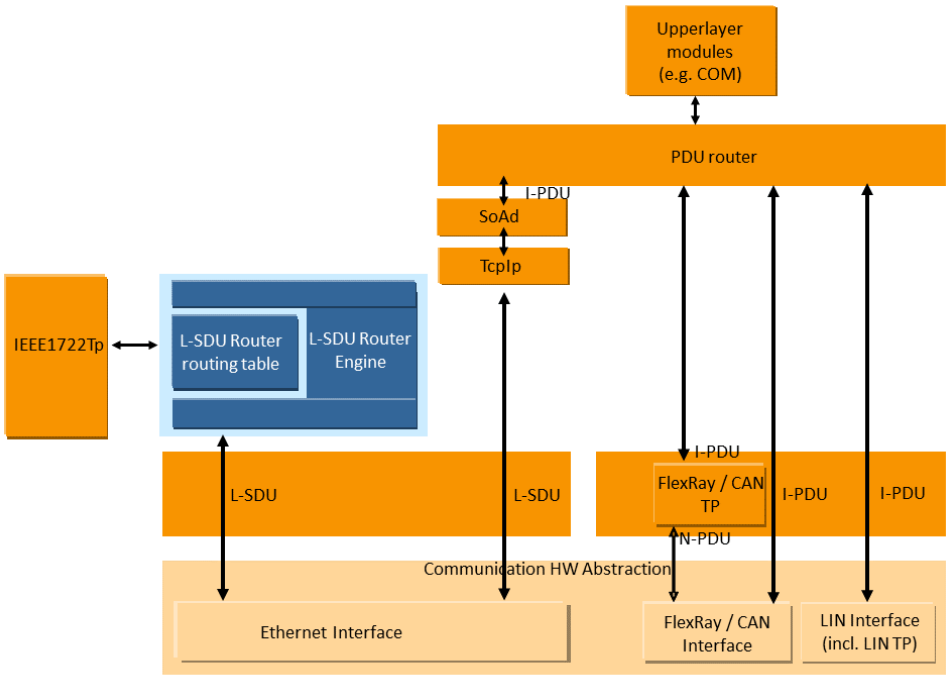

Figure 1.2: Detailed L-SDU Router Structure

The L-SDU Router module mainly consists of two parts:

- The **L-SDU Router routing paths**: static routing paths describing the routing attributes for each L-SDU to be routed. The routing paths can be (if supported) updated post-build loadable in the programming state of the ECU or selected when initializing the L-SDU Router by post-build selectable (see section 10.1.1).
- The **L-SDU Router Engine**: the actual code performing routing actions according to the L-SDU Router routing paths. The L-SDU Router Engine has to deal with:
  - Routing L-SDU from source(s) to destination(s),
  - Translating the source L-SDU ID to the destination L-SDU ID (e.g. `LSduR_IEEE1722Tp_RxIndication` to `IEEE1722Tp_RxIndication`,

1.3 L-SDU handling

L-SDUs are identified by static L-SDU IDs. The L-SDU Router module determines the destination of an L-SDU by using the L-SDU ID in a static configuration table. L-SDUs are used for the data exchange of the modules directly above the L-SDU Router module, e.g. the IEEE1722Tp module. The routing operation of the L-SDU Router module does not modify the L-SDU, it simply forwards the L-SDU to the destination module.

The L-SDU ID is set in the configuration that also implements the API. This will allow an efficient implementation of look-up tables in each module receiving an L-SDU ID (e.g. the L-SDU Router module's configuration contains the L-SDU ID for the `LSduR_EthIfTxConfirmation`, while EthIf module's configuration contains the L-SDU ID for the `EthIf_Transmit`).

The following list summarizes the routing capabilities of LSduR:

1. L-SDU Forwarding
   - Transmission from upper layer
     - Communication Interface
       - Singlecast (1:1) an L-SDU from a local module to a communication interface module.
   - Reception to upper layer
     - Communication Interface
       - Singlecast (1:1) an L-SDU from a communication interface module to a local module.

2 Acronyms and Abbreviations

The glossary below includes acronyms and abbreviations relevant to the LSduR module that are not included in the [2, AUTOSAR glossary].

Abbreviation / Acronym:	Description:
PDU	Protocol Data Unit.
I-PDU	Interaction Layer PDU. An I-PDU consists of data (buffer), length and I-PDU ID.
I-PDU ID	I-PDU Identifier.
L-PDU	Data Link Layer PDU. One or more I-PDUs are packed into one L-PDU. The L-PDU is bus specific, e.g. Ethernet frame.
L-PDU ID	L-PDU Identifier.
SDU	Service Data Unit.
L-SDU	Data link layer service data unit. An L-SDU consists of data(buffer), length, L-SDU ID and may L-PDU specific information transported via meta data.
L-SDU ID	L-SDU identifier
L-SDU Router	Module that transfers L-SDUs from one module to another module. The L-SDU Router module can be utilized for internal routing purposes.
Upper Layer Modules (Up)	Modules above the L-SDU Router. This layer usually includes IEEE1722Tp.
Lower Layer Modules (Lo)	Modules below the L-SDU Router. This layer includes the Ethernet Communication Interface module.
<SrcLo>	Lower layer Communication Interface module acting as a source of the L-SDU. The SrcLo is always one.
<DstLo>	Lower layer Communication Interface module acting as a destination of the L-SDU. The DstLo may by one to many.
<Lo>	Lower layer communication interface module.
<Up>	Upper layer communication Interface module

Table 2.1: Acronyms and abbreviations used in the scope of this Document

3 Related documentation

3.1 Input documents & related standards and norms

[1] Layered Software Architecture
AUTOSAR_CP_EXP_LayeredSoftwareArchitecture

[2] Glossary
AUTOSAR_FO_TR_Glossary

[3] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral

[4] General Requirements on Basic Software Modules
AUTOSAR_CP_SRS_BSWGeneral

[5] Requirements on Gateway
AUTOSAR_CP_SRS_Gateway

[6] Specification of IEEE1722 Transport Protocol Module
AUTOSAR_CP_SWS_IEEE1722TransportLayer

[7] List of Basic Software Modules
AUTOSAR_CP_TR_BSWModuleList

[8] Specification of ECU Configuration
AUTOSAR_CP_TPS_ECUConfiguration

3.2 Related specification

AUTOSAR provides a General Specification on Basic Software modules [3, SWS BSW General], which is also valid for LSduR.

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

4 Constraints and assumptions

4.1 Limitations

The L-SDU Router module does not:

- have mechanisms for signal extraction or conversion,
- have mechanisms for data integrity checking (like checksums),
- change or modify the L-SDU,
- make any L-SDU payload dependent routing decisions,

4.1.1 Limitations on supported functionality

In R23-11 the L-SDU Router is considered to act a pass-trough module between the IEEE1722Tp and the EthIf

- Gateway functionality is excluded from the L-SDU router
- The L-SDU router interacts only with IEEE1722Tp and the EthIf

4.2 Applicability to car domains

R23-11: The L-SDU Router is used in all ECUs where communication via IEEE1722Tp module is necessary.

After R23-11: The L-SDU Router is used in all ECUs where communication is necessary.

5 Dependencies to other modules

The L-SDU Router module depends on the APIs and capabilities of the used communication hardware abstraction layer modules and the used communication service layer modules. Basically the API functions required by the L-SDU Router module are:

Communication Interface modules:

- `<Lo>_Transmit` (e.g. `EthIf_Transmit`)

Upper layer modules which process I-PDUs originating from Communication Interface modules:

- `<Up>_RxIndication` (e.g. `IEEE1722Tp_RxIndication`),
- `<Up>_TxConfirmation` (e.g. `IEEE1722Tp_TxConfirmation`),)

5.1 File structure

5.1.1 Code file structure

For details refer to the Chapter 5.1.6 "Code file structure" in [3, SWS_BSWGeneral].

The code file structure is not defined within this specification completely. However to allow integration to other modules the following structure is needed.

5.1.2 Header file structure

[CP_SWS_LSduR_00001]{DRAFT} ⌈The L-SDU Router module shall provide the functions used by the different modules in separate header files.⌋*(SRS_BSW_00415)*

Example: If EthIf is used then the L-SDU Router module shall provide `LSduR_EthIf.h`.

[CP_SWS_LSduR_00002]{DRAFT} ⌈The L-SDU Router implementation shall include `Det.h`.⌋*(SRS_BSW_00350)*

[CP_SWS_LSduR_00003]{DRAFT} ⌈All L-SDU Router header files shall contain a software and specification version number.⌋*(SRS_BSW_00003)*

This structure allows the separation between platform, compiler and implementation specific definitions and declarations from general definitions as well as the separation of source code and configuration.

5.2 Version check

For details refer to the chapter 5.1.8 "Version Check" in [3, SWS_BSWGeneral].

6 Requirements Tracing

The following tables reference the requirements specified in [4, CP_SRS_BSWGeneral] and [5, CP_SRS_Gateway] and links to the fulfillment of these. Please note that if column “Satisfied by” is empty for a specific requirement this means that this requirement is not fulfilled by this document.

Requirement	Description	Satisfied by
[SRS_BSW_00003]	All software modules shall provide version and identification information	[CP_SWS_LSduR_00003]
[SRS_BSW_00101]	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	[CP_SWS_LSduR_91006] [CP_SWS_LSduR_91007]
[SRS_BSW_00305]	Data types naming convention	[CP_SWS_LSduR_91003] [CP_SWS_LSduR_91004] [CP_SWS_LSduR_91005]
[SRS_BSW_00310]	API naming convention	[CP_SWS_LSduR_00033] [CP_SWS_LSduR_91006] [CP_SWS_LSduR_91007] [CP_SWS_LSduR_91008] [CP_SWS_LSduR_91009] [CP_SWS_LSduR_91010] [CP_SWS_LSduR_91011] [CP_SWS_LSduR_91012] [CP_SWS_LSduR_91013] [CP_SWS_LSduR_91014]
[SRS_BSW_00323]	All AUTOSAR Basic Software Modules shall check passed API parameters for validity	[CP_SWS_LSduR_00034] [CP_SWS_LSduR_91001]
[SRS_BSW_00335]	Status values naming convention	[CP_SWS_LSduR_91005]
[SRS_BSW_00337]	Classification of development errors	[CP_SWS_LSduR_91001]
[SRS_BSW_00350]	All AUTOSAR Basic Software Modules shall allow the enabling/ disabling of detection and reporting of development errors.	[CP_SWS_LSduR_00002]
[SRS_BSW_00358]	The return type of init() functions implemented by AUTOSAR Basic Software Modules shall be void	[CP_SWS_LSduR_91006] [CP_SWS_LSduR_91007]
[SRS_BSW_00384]	The Basic Software Module specifications shall specify at least in the description which other modules they require	[CP_SWS_LSduR_91015] [CP_SWS_LSduR_91016]
[SRS_BSW_00400]	Parameter shall be selected from multiple sets of parameters after code has been loaded and started	[CP_SWS_LSduR_91003]
[SRS_BSW_00404]	BSW Modules shall support post-build configuration	[CP_SWS_LSduR_00035] [CP_SWS_LSduR_00037] [CP_SWS_LSduR_00038] [CP_SWS_LSduR_00039] [CP_SWS_LSduR_91003]
[SRS_BSW_00405]	BSW Modules shall support multiple configuration sets	[CP_SWS_LSduR_91004]
[SRS_BSW_00406]	A static status variable denoting if a BSW module is initialized shall be initialized with value 0 before any APIs of the BSW module is called	[CP_SWS_LSduR_00026] [CP_SWS_LSduR_00027] [CP_SWS_LSduR_00028] [CP_SWS_LSduR_00029] [CP_SWS_LSduR_00030] [CP_SWS_LSduR_00031] [CP_SWS_LSduR_00032] [CP_SWS_LSduR_91005]
[SRS_BSW_00414]	Init functions shall have a pointer to a configuration structure as single parameter	[CP_SWS_LSduR_91006] [CP_SWS_LSduR_91007]
[SRS_BSW_00415]	Interfaces which are provided exclusively for one module shall be separated into a dedicated header file	[CP_SWS_LSduR_00001]

△

Requirement	Description	Satisfied by
[SRS_BSW_00438]	Configuration data shall be defined in a structure	[CP_SWS_LSduR_00035] [CP_SWS_LSduR_91003]
[SRS_BSW_00452]	Classification of runtime errors	[CP_SWS_LSduR_91001]
[SRS_GTW_06001]	Gateway shall be only be reconfigured while the configuration table to be reconfigured is not in use	[CP_SWS_LSduR_00031] [CP_SWS_LSduR_00038]
[SRS_GTW_06002]	The `Routing Configuration` shall be updateable at post-build time	[CP_SWS_LSduR_00037]
[SRS_GTW_06097]	A `Routing Configuration` shall be identified by an unique ID number	[CP_SWS_LSduR_00036] [CP_SWS_LSduR_00039] [CP_SWS_LSduR_91004] [CP_SWS_LSduR_91008]
[SRS_GTW_06141]	L-SDU Router transparent routing	[CP_SWS_LSduR_00005] [CP_SWS_LSduR_00006] [CP_SWS_LSduR_00007] [CP_SWS_LSduR_00008] [CP_SWS_LSduR_00010] [CP_SWS_LSduR_00011] [CP_SWS_LSduR_00012] [CP_SWS_LSduR_00013] [CP_SWS_LSduR_00014] [CP_SWS_LSduR_00015] [CP_SWS_LSduR_00016] [CP_SWS_LSduR_00017] [CP_SWS_LSduR_00018] [CP_SWS_LSduR_00019] [CP_SWS_LSduR_00020] [CP_SWS_LSduR_00021] [CP_SWS_LSduR_00022] [CP_SWS_LSduR_00030]
[SRS_GTW_06142]	L-SDU Router error handling for unknown PDU-ID	[CP_SWS_LSduR_00034]
[SRS_GTW_06143]	L-SDU Router error handling for local reception or transmission	[CP_SWS_LSduR_00023]
[SRS_GTW_06144]	L-SDU Router interface (API) for IEEE1722Tp	[CP_SWS_LSduR_91009] [CP_SWS_LSduR_91010] [CP_SWS_LSduR_91011] [CP_SWS_LSduR_91012] [CP_SWS_LSduR_91013] [CP_SWS_LSduR_91014]
[SRS_GTW_06145]	L-SDU Router interface (API) for bus and network interfaces	[CP_SWS_LSduR_91009] [CP_SWS_LSduR_91010] [CP_SWS_LSduR_91011] [CP_SWS_LSduR_91012] [CP_SWS_LSduR_91013] [CP_SWS_LSduR_91014]
[SRS_GTW_06146]	L-SDU Router resource usage shall be scalable to zero	[CP_SWS_LSduR_00010] [CP_SWS_LSduR_00024] [CP_SWS_LSduR_00025]

Table 6.1: RequirementsTracing

7 Functional specification

The L-SDU Router module is an L-SDU transfer unit placed above Ethernet Interface module (lower layer module) and below IEEE1722Tp module (upper layer module), see Figure 1.1.

Beside the L-SDU Router module there is the IEEE1722Tp module [6] that support IEEE1722-stream related communication. The IEEE1722Tp module is the upper layer module, which request transmission by calling `Transmit` of the the L-SDU Router module. The L-SDU Router module forward reception of data by calling `RxIndication` or indicate transmission confirmation by calling `TxConfirmation` of the IEEE1722Tp module.

From the ECU point of view, the L-SDU Router module can perform two different classes of operations:

- **PDU Reception to local module(s)**:
  - receive L-SDUs from one lower layer module and forward them to one or more upper layer modules,
- **PDU Transmission from local module(s)**: transmit L-SDUs to one lower layer module on request of one upper layer module,

[CP_SWS_LSduR_00004]{DRAFT} ⌈When the LSduR reports a development, runtime, or transient error, it shall use the `moduleId` of the caller module as `instanceId` when calling the Default Error Tracer module.⌋*()*

For example: When an error is detected during the `LSduR_EthIfRxIndication`, `Det_ReportError`(51 (Module id of LSduR), 65 (ModuleId (used as InstanceId) of EthIf), 0x42, `LSDUR_E_PDU_INSTANCES_LOST`) shall be called.

Note: The standardized module ID is found in the List of Basic Software Modules document [7]. The parameter `LSduRBswModuleRef` identifies the module used. With this information the `moduleId` can be retrieved in the `BswModuleDescription.moduleId`.

7.1 L-SDU handling

[CP_SWS_LSduR_00005]{DRAFT} ⌈The L-SDU Router module shall transfer an L-SDU without modification in a consistent manner from the source module to the destination module(s).⌋*(SRS_GTW_06141)*

An L-SDU is identified by the L-SDU ID and/or the symbolic name (i.e. the `SymbolicNameValue` of the container of the PDU [8, Specification of ECU Configuration]). For post-build the L-SDU ID is required because the L-SDU must be identified after the L-SDU Router module is compiled. If the L-SDU Router module is pre-compile (i.e. in

source code) the symbolic names may be used, see [8, Specification of ECU Configuration].

Each BSW module that handles L-SDUs and provides an API for L-SDUs must contain a list of L-SDU IDs [8]. This means that each called module will have a look-up table identifying the PDU.

Example: The IEEE1722Tp module calls `LSduR_IEEE1722TpTransmit` (here the L-SDU Router module configuration contains the L-SDU ID), the L-SDU Router module will call `EthIf_Transmit` (here the EthIf module configuration contains the L-SDU ID), the EthIf will call `LSduR_EthIfTxConfirmation` (here the L-SDU Router module configuration contains the L-SDU ID), and L-SDU Router module will call `IEEE1722Tp_TxConfirmation` (here the IEEE1722Tp module configuration contains the L-SDU ID). The example is illustrated in the following Figure 7.1 (only L-SDU ID is shown as parameter):

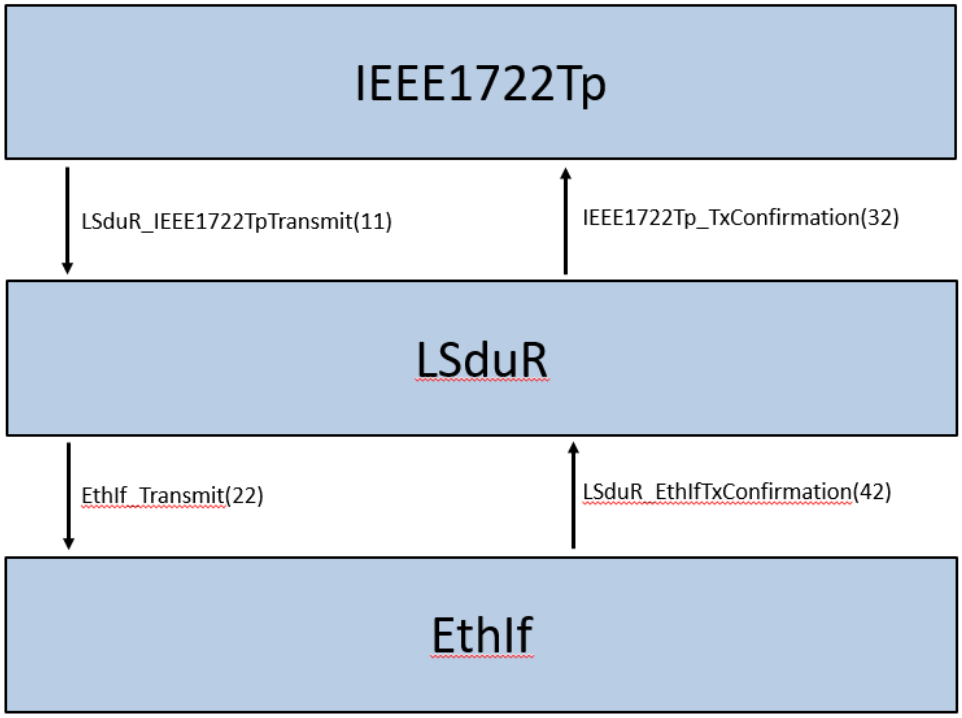

Figure 7.1: I-SDU ID Example

[CP_SWS_LSduR_00006]{DRAFT} ⌈The L-SDU Router module shall identify a routing path uniquely by the combination of source module L-SDU ID (located in the L-SDU Router configuration) and destination L-SDU IDs (located in the called destination module configurations).⌋*(SRS_GTW_06141)*

[CP_SWS_LSduR_00007]{DRAFT} ⌈The L-SDU Router module shall convert the L-SDU ID to the destination module(s) for both `Transmit` path and `TxConfirmation`/`RxIndication` path.⌋*(SRS_GTW_06141)*

Example: The IEEE1722Tp module transmits an L-SDU to EthIf. The `LSduR_IEEE1722TpTransmit` is called. The L-SDU Router module will convert the source

L-SDU ID (L-SDU Router module configuration) to one L-SDU ID for EthIf (EthIf module configuration). The PduInfoType value received from the IEEE1722Tp module is copied to the EthIf module without change.

Example: The EthIf module will call LSduR_EthIfTxConfirmation with an L-SDU ID and, dependent on the success of the transmission, with a result E_OK (successful transmission) or E_NOT_OK (not successful transmission). Then the L-SDU Router module will convert this L-SDU ID and forward the call to IEEE1722Tp using IEEE1722Tp_TxConfirmation with the converted L-SDU ID and the received result.

[CP_SWS_LSduR_00008]{DRAFT} ⌈The L-PDU Router module shall only route L-SDUs according to the routing paths given in the configuration.⌋*(SRS_GTW_06141)*

[CP_SWS_LSduR_00009]{DRAFT} ⌈LSduR generator (validation) shall deny configurations where L-SDUs with different MetaDataTypes are connected by a routing path.⌋*()*

7.1.1 L-SDU Reception to upper layer module

The receive operation of the L-SDU Router module is either finalized by an RxIndication (LSduR_<User:Lo>RxIndication) from a lower layer module (Communication Interface) or , if configured, by a call of ReleaseRxBuffer (LSduR_<User:Up>ReleaseRxBuffer) from the receiving upper layer module (e.g. IEEE1722 application) after RxIndication from a lower layer module has been called.

The RxIndication function is originated from the lower layer either in the context of a cyclic function after polling a communication driver or in the context of an interrupt.

The ReleaseRxBuffer function is originated from the upper layer either in context of the RxIndication or in the context context of a cyclic function after RxIndication has been called.

7.1.1.1 Communication Interface

The source Communication Interface module indicates a received L-SDU by calling LSduR_<User:Lo>RxIndication. The L-SDU may have multiple local destination modules configured by the routing path.

[CP_SWS_LSduR_00010]{DRAFT} ⌈The L-SDU Router module shall provide 1:n routing for an L-SDU received from a Communication Interface module and routed to one or more upper layer module(s).⌋*(SRS_GTW_06141, SRS_GTW_06146)*

Example: An L-SDU is received on EthIf and forwarded to IEEE1722Tp.

[CP_SWS_LSduR_00011]{DRAFT} ⌈When the LSduR_<User:Lo>RxIndication is called the L-SDU Router module shall call <Up>_RxIndication for each destination upper layer module.⌋*(SRS_GTW_06141)*

[CP_SWS_LSduR_00012]{DRAFT} ⌈If an L-SDU received by a local module is directly forwarded, the L-SDU Router shall not check the length of the L-SDU.⌋*(SRS_GTW_06141)*

Since the L-SDU Router module will not buffer this L-SDU it does not have to reject L-SDU that are longer/shorter than configured.

[CP_SWS_LSduR_00013]{DRAFT} ⌈In case of a singlecast (1:1) reception, when upper layer source module calls LSduR_<User:Up>ReleaseRxBuffer the L-SDU Router shall call <Lo>_ReleaseRxBuffer of the corresponding Communication Interface module.⌋*(SRS_GTW_06141)*

[CP_SWS_LSduR_00014]{DRAFT} ⌈In case of a multicast (1:n, n>1) reception, the L-SDU Router shall call the <Lo>_ReleaseRxBuffer of the corresponding Communication Interface module when the last LSduR_<User:Up>ReleaseRxBuffer call of the corresponding upper layer module has been received.⌋*(SRS_GTW_06141)*

7.1.2 L-SDU Transmission from upper layer module(s)

The transmit operations of the lower layer destination modules are always asynchronous. This means that a transmission service request returns immediately after the I-PDU has been passed by the L-SDU Router module to the lower layer destination(s). If the L-SDU Router module is notified by lower layer destination modules via LSduR_<User:Lo>TxConfirmation (Communication Interface) after successful or failed transmission of the L-SDU, the L-PDU Router module will forward this confirmation to the upper layer module via <Up>_TxConfirmation (Communication Interface).

The transmit operation of the L-SDU Router module is triggered by a L-SDU Transmit request from an upper layer source module and the L-SDU Router forwards the request to lower layer destination(s).

[CP_SWS_LSduR_00015]{DRAFT} ⌈The L-SDU shall not be buffered in the L-SDU Router module in case of L-SDU transmission from an upper layer source module.⌋*(SRS_GTW_06141)*

7.1.2.1 Communication Interface

There are four ways that L-SDUs can be transmitted on Communication Interface:

1. Direct data provision - where the upper layer module is calling the LSduR_<User:Up>Transmit function, the L-SDU Router module forwards the call

to <Lo>_Transmit and the data is copied by the lower Communication Interface module in the call.

2. Direct data provision - where the upper layer module is calling the LSduR_<User:Up>ImmediateTransmit function, the L-SDU Router module forwards the call to <Lo>_ImmediateTransmit and the data is copied by the lower Communication Interface module in the call.

3. Trigger transmit provision - where the lower Communication Interface module requests transmission of an L-SDU by using the LSduR_<User:Lo>TriggerTransmit, and L-SDU Router module forwards the call to <Up>_TriggerTransmit and the data is copied to the destination's buffer by the upper layer module.

4. Trigger transmit provision - Where the upper layer module calls the LSduR_<User:Up>Transmit function, the L-SDU Router module forwards the call to <Lo>_Transmit and the data is not copied by the lower module (Communication Interface module). The data will later be requested by the lower layer using LSduR_<User:Lo>TxConfirmation.

The confirmation of the transmission of the L-SDU is the same for the direct and trigger transmit data provision:

[CP_SWS_LSduR_00016]{DRAFT} ⌈When the Communication Interface module calls LSduR_<User:Lo>TxConfirmation the L-SDU Router shall call <Up>_TxConfirmation in the upper layer module and forward the transmission result from the lower to the upper layer module.⌋*(SRS_GTW_06141)*

[CP_SWS_LSduR_00017]{DRAFT} ⌈If the L-SDU is transmitted by an upper layer module the L-SDU Router module shall not check the length of the L-SDU.⌋*(SRS_GTW_06141)*

[CP_SWS_LSduR_00018]{DRAFT} ⌈When upper layer source module calls LSduR_<User:Up>Transmit the L-SDU Router shall call <Lo>_Transmit for each Communication Interface destination module.⌋*(SRS_GTW_06141)*

[CP_SWS_LSduR_00019]{DRAFT} ⌈If singlecast (1:1) the return value of the <Lo>_Transmit call shall be forwarded to the upper layer source module.⌋*(SRS_GTW_06141)*

[CP_SWS_LSduR_00020]{DRAFT} ⌈When upper layer source module calls LSduR_<User:Up>ImmediateTransmit the L-SDU Router shall call <Lo>_ImmediateTransmit for each Communication Interface destination module.⌋*(SRS_GTW_06141)*

[CP_SWS_LSduR_00021]{DRAFT} ⌈If singlecast (1:1) the return value of the <Lo>_ImmediateTransmit call shall be forwarded to the upper layer source module.⌋*(SRS_GTW_06141)*

7.1.2.1.1 Trigger transmit data provision

The upper layer module must be informed whether it has to reset the update-bits.

[CP_SWS_LSduR_00022]{DRAFT} ⌈The L-SDU Router module shall forward a LSduR_<User:Lo>TriggerTransmit request by the Communication Interface lower layer module to the upper layer module by calling <Up>_TriggerTransmit.⌋*(SRS_GTW_06141)*

[CP_SWS_LSduR_00023]{DRAFT} ⌈The L-SDU Router module shall copy the return value from the <Up>_TriggerTransmit to the lower layer module.⌋*(SRS_GTW_06143)*

7.1.2.1.2 Error handling

For errors occurred using singlecast or multicast over Communication Interface modules, no specific error handling is done. Errors in return values are forwarded to the upper layer source module.

7.2 Zero Cost Operation

Zero cost operation is an optimization that may be done where source and destination modules are single and in source code (one of the modules must be in source code otherwise the L-SDU Router must create glue-code for the function call). For example an ECU with a IEEE1722Tp module and a single Ethernet network, the LSduR_IEEE1722Transmit may directly call the EthIf_Transmit without any logic inside the L-SDU Router Module. The L-SDU Router becomes a macro layer.

This optimization is only possible where routing paths are of configuration class Pre-Compile.

[CP_SWS_LSduR_00024]{DRAFT} ⌈If LSduRZeroCostOperation is set to true and all routing paths are of configuration class Pre-Compile; modules directly above or below the L-SDU Router may directly call each other without using LSduR module functions.⌋*(SRS_GTW_06146)*

[CP_SWS_LSduR_00025]{DRAFT} ⌈If LSduRZeroCostOperation is set to true and at least one routing path is not of configuration class Pre-Compile; the L-SDU Router module configuration generator shall report an error.⌋*(SRS_GTW_06146)*

7.3 State Management

The state machine of the L-SDU Router module is depicted in Figure 7.2.

[CP_SWS_LSduR_00026]{DRAFT} ⌈Only one instance of the state machine shall exist in the L-SDU Router module.⌋*(SRS_BSW_00406)*

[CP_SWS_LSduR_00027]{DRAFT} ⌈The L-SDU Router module shall consist of two states, `LSDUR_UNINIT` and `LSDUR_ONLINE` as defined in `LSduR_StateType`⌋*(SRS_BSW_00406)*

[CP_SWS_LSduR_00028]{DRAFT} ⌈The L-SDU Router module shall be in the state `LSDUR_UNINIT` after power up the L-SDU Router module (i.e. before calling the `LSduR_Init` function).⌋*(SRS_BSW_00406)*

[CP_SWS_LSduR_00029]{DRAFT} ⌈The L-SDU Router module shall change to the state `LSDUR_ONLINE` when the L-SDU Router has successfully been initialized via the function `LSduR_Init`⌋*(SRS_BSW_00406)*

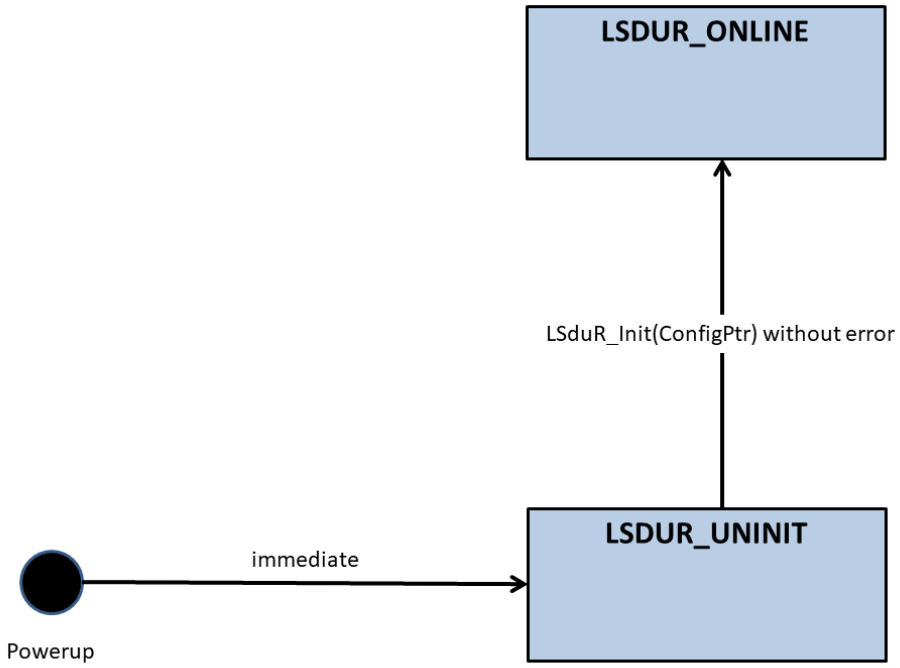

Figure 7.2: L-SDU Router states

[CP_SWS_LSduR_00030]{DRAFT} ⌈The L-SDU Router module shall perform routing of L-SDUs according to the L-SDU Router routing tables only when it is in the online state `LSDUR_ONLINE`⌋*(SRS_BSW_00406, SRS_GTW_06141)*

[CP_SWS_LSduR_00031]{DRAFT} ⌈The L-SDU Router module shall perform no routing when it is in the uninitialized state `LSDUR_UNINIT`⌋*(SRS_BSW_00406, SRS_GTW_06001)*

[CP_SWS_LSduR_00032]{DRAFT} ⌈If the L-SDU Router module has not been initialized (state `LSDUR_UNINIT` all functions except `LSduR_Init` and `LSduR_GetVersionInfo` shall report the error `LSDUR_E_UNINIT` via the DET when called, when `LSduRDevErrorDetect` is enabled.⌋*(SRS_BSW_00406)*

7.4 Complex Driver Interaction

Besides the AUTOSAR IEEE1722Tp module, Complex Drivers (CDD) are also possible as upper or lower layer modules for the LSduR.

Whether a CDD is an upper layer or a lower layer module for the LSduR is configurable via the LSduRUpperModule or LSduRLowerModule configuration parameters of the LSduRBswModules configuration.

A CDD can require Communication Interface API, depending on the configuration parameters LSduRCommunicationInterface (e.g. LSduRTransmit) of the LSduRBswModules configuration. The API functions provided by the LSduR for the CDD interaction contain the CDD's service prefix as specified by the apiServicePrefix configuration parameter, see [CP_SWS_LSduR_00033].

The LSduR provides the unique transmit function LSduR_<Cdd>Transmit for each upper layer CDD. When a callout function of the LSduR is invoked from a lower layer module for a L-SDU that is transmitted or received by an upper layer CDD, the LSduR invokes the corresponding target function of the CDD.

For a lower layer CDD that requires a Communication Interface API, the LSduR provides a unique set of Communication Interface API functions LSduR_<Cdd>Rx Indication and - if configured - LSduR_<Cdd>TxConfirmation and LSduR_<Cdd>TriggerTransmit, see Section 8.3.3.

When an API function of the LSduR is invoked from an upper layer module for a L-SDU that is transmitted or received by a lower layer CDD, the L-SduR invokes the corresponding target function of the CDD.

To determine if a L-SDU is transmitted or received by a CDD, the LSduR has to examine the origin of the references to the PDU list in the EcuC module:

- If the source L-SDU of a routing path references a PDU in the PDU list that is also referenced by an upper layer CDD, the L-SDU is transmitted by the CDD.
- If the destination L-SDU of a routing path references a PDU in the PDU list that is also referenced by an upper layer CDD, the L-SDU is received by the CDD.
- If the source L-SDU of a routing path references a PDU in the PDU list that is also referenced by a lower layer CDD, the L-SDU is received from the CDD.
- If the destination L-SDU of a routing path references a PDU in the PDU list that is also referenced by a lower layer CDD, the L-SDU is transmitted via the CDD.

[CP_SWS_LSduR_00033]{DRAFT} ⌈The LSduR shall use the apiServicePrefix attribute of the CDD's vendor specific module definition (EcucModuleDef element) to replace the <Lo> and <Up> tags of the GenericComServices APIs. The CDD's vendor specific module definition can be indirectly accessed via the configuration parameter LSduRBswModuleRef which references the top-level element of the concrete configuration of the CDD (i.e., EcucModuleConfigurationValues element)

which references the CDD's vendor specific module definition (`EcucModuleDef element`).⌋(*SRS_BSW_00310*)

7.5 Security Events

The module does not report security events.

7.6 API parameter checking

[CP_SWS_LSduR_00034]{DRAFT} ⌈If development error detection is enabled, a PDU identifier is not within the specified range, and the PDU identifier is configured to be used by the L-SDU Router module, the L-SDU Router module shall report the error `LSDUR_E_PDU_ID_INVALID` to the DET module, when `LSduRDevErrorDetect` is enabled.⌋(*SRS_GTW_06142*, *SRS_BSW_00323*)

7.7 Error Classification

Section "Error Handling" of the document [3, SWS BSW General] "General Specification of Basic Software Modules" describes the error handling of the Basic Software in detail. Above all, it constitutes a classification scheme consisting of five error types which may occur in BSW modules.

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

7.7.1 Development Errors

[CP_SWS_LSduR_91001]{DRAFT} **Definiton of development errors in module LSduR** ⌈

Type of error	*Related error code*	*Error value*
Invalid configuration pointer	LSDUR_E_INIT_FAILED	0x00
API service (except LSduR_GetVersionInfo) used without module initialization or LSduR_Init called in any state other than LSDUR_UNINIT	LSDUR_E_UNINIT	0x01
Invalid PDU identifier	LSDUR_E_PDU_ID_INVALID	0x02
Null pointer has been passed as an argument	LSDUR_E_PARAM_POINTER	0x03

⌋(*SRS_BSW_00337*, *SRS_BSW_00323*, *SRS_BSW_00452*)

7.7.2 Runtime Errors

[CP_SWS_LSduR_91002]{DRAFT} **Definiton of runtime errors in module LSduR** ⌈

Type of error	*Related error code*	*Error value*
Loss of a PDU instance (buffer overrun in gateway operation)	LSDUR_E_PDU_INSTANCES_LOST	0x04

⌋()

7.7.3 Transient Faults

The LSduR module does not define transient faults.

7.7.4 Production Errors

The IEEE1722Tp module does not define production errors.

7.7.5 Extended Production Errors

The LSduR module does not define extended production errors.

8 API specification

8.1 Imported types

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

[CP_SWS_LSduR_91017] Definition of imported datatypes of module LSduR ⌈

Module	*Header File*	*Imported Type*
ComStack_Types	ComStack_Types.h	PduIdType
	ComStack_Types.h	PduInfoType
	ComStack_Types.h	PduLengthType
Std	Std_Types.h	Std_ReturnType
	Std_Types.h	Std_VersionInfoType

⌋()

8.2 Type definitions

8.2.1 LSduR_PBConfigType

The post-build-time configuration fulfills two functionalities:

- Post-build selectable, where more than one configuration is located in the ECU, and one is selected at init of the L-SDU Router module
- Post-build loadable, where one configuration is located in the ECU. This configuration may be reprogrammed after compile-time

Basically there is no restriction to mix both selectable and loadable. Typically the post-build loadable is located in its own flash sector where it can be reprogrammed without affecting other modules/applications.

[CP_SWS_LSduR_91003]{DRAFT} **Definition of datatype LSduR_PBConfigType** ⌈

Name	LSduR_PBConfigType (draft)	
Kind	Structure	
Elements	Implementation specific	
	Type	–
	Comment	–
Description	Data structure containing post-build-time configuration data of the L-SDU Router. **Tags:** atp.Status=draft	
Available via	LSduR.h	

⌋(*SRS_BSW_00400*, *SRS_BSW_00438*, *SRS_BSW_00404*, *SRS_BSW_00305*)

[CP_SWS_LSduR_00035]{DRAFT} ⌈The type `LSduR_PBConfigType` is an external data structure containing post-build-time configuration data of the L-SDU Router mod-

ule which shall be implemented in LSduR_PBcfg.c.⌋(*SRS_BSW_00438*, *SRS_BSW_00404*)

Note: see chapter section 5.1

8.2.2 LSduR_PBConfigIdType

This type is returned by the `LSduR_GetConfigurationId` API.

[CP_SWS_LSduR_91004]{DRAFT} **Definition of datatype LSduR_PBConfigIdType** ⌈

Name	LSduR_PBConfigIdType (draft)
Kind	Type
Derived from	uint16
Description	Identification of the post-build configuration currently used for routing L-SDUs. An ECU may contain several configurations (post-build selectable), each have unique Id. **Tags:** atp.Status=draft
Available via	LSduR.h

⌋(*SRS_BSW_00405*, *SRS_BSW_00305*, *SRS_GTW_06097*)

8.2.3 LSduR_StateType

This type is returned by the `LSduR_GetConfigurationId` API.

[CP_SWS_LSduR_91005]{DRAFT} **Definition of datatype LSduR_StateType** ⌈

Name	LSduR_StateType (draft)		
Kind	Enumeration		
Range	LSDUR_UNINIT	–	L-SDU Router not initialized
	LSDUR_ONLINE	–	L-SDU Router initialized successfully
Description	States of the L-SDU Router **Tags:** atp.Status=draft		
Available via	LSduR.h		

⌋(*SRS_BSW_00305*, *SRS_BSW_00335*, *SRS_BSW_00406*)

8.3 Function definitions

8.3.1 General functions provided by the L-SDU Router

8.3.1.1 LSduR_Init

[CP_SWS_LSduR_91006]{DRAFT} **Definition of API function LSduR_Init** ⌈

Service Name	LSduR_Init (draft)	

Syntax	`void LSduR_Init (`
	`  const LSduR_PBConfigType* ConfigPtr`
	`)`	

Service ID [hex]	0x1	
Sync/Async	Synchronous	
Reentrancy	Non Reentrant	
Parameters (in)	ConfigPtr	Pointer to post build configuration
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Initializes the L-SDU Router **Tags:** atp.Status=draft	
Available via	LSduR.h	

⌋(*SRS_BSW_00101*, *SRS_BSW_00358*, *SRS_BSW_00414*, *SRS_BSW_00310*)

Integration note: To avoid problems calling the PDU Router module uninitialized it is important that the PDU Router module is initialized before interfaced modules.
Note: NULL pointer checking is specified within document [3, SWS BSW General].

8.3.1.2 LSduR_GetVersionInfo

[CP_SWS_LSduR_91007]{DRAFT} **Definition of API function LSduR_GetVersion Info** ⌈

Service Name	LSduR_GetVersionInfo (draft)	

Syntax	`void LSduR_GetVersionInfo (`
	`  Std_VersionInfoType versionInfo`
	`)`	

Service ID [hex]	0x2	
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 of this module. **Tags:** atp.Status=draft
Available via	LSduR.h

⌋(*SRS_BSW_00101*, *SRS_BSW_00358*, *SRS_BSW_00414*, *SRS_BSW_00310*)

8.3.1.3 LSduR_GetConfigurationId

[CP_SWS_LSduR_91008]{DRAFT} **Definition of API function LSduR_GetConfigurationId** ⌈

Service Name	LSduR_GetConfigurationId (draft)	
Syntax	`LSduR_PBConfigIdType LSduR_GetConfigurationId ( void )`	
Service ID [hex]	0x3	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	None	
Return value	LSduR_PBConfigIdType	Identifier of the post-build time configuration
Description	Returns the unique identifier of the post-build time configuration of the L-SDU Router **Tags:** atp.Status=draft	
Available via	LSduR.h	

⌋(*SRS_GTW_06097*, *SRS_BSW_00310*)

[CP_SWS_LSduR_00036]{DRAFT} ⌈The function `LSduR_GetConfigurationId` shall return the unique identifier of the post-build time configuration of the L-SDU Router module.⌋(*SRS_GTW_06097*)

8.3.2 Configurable interfaces definitions for interaction with upper layer module

8.3.2.1 LSduR_<User:Up>Transmit

[CP_SWS_LSduR_91009]{DRAFT} **Definition of API function LSduR_<User:Up>Transmit** ⌈

Service Name	LSduR_<User:Up>Transmit (draft)	
Syntax	`Std_ReturnType LSduR_<User:Up>Transmit ( PduIdType TxPduId, const PduInfoType* PduInfoPtr )`	
Service ID [hex]	0x49	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	TxPduId	Identifier of the PDU to be transmitted
	PduInfoPtr	Length of and pointer to the PDU data and pointer to MetaData.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: Transmit request has been accepted. `E_NOT_OK`: Transmit request has not been accepted.
Description	Requests transmission of a PDU. **Tags:** atp.Status=draft	
Available via	LSduR_<module>.h	

⌋(*SRS_GTW_06144*, *SRS_GTW_06145*, *SRS_BSW_00310*)

8.3.2.2 LSduR_<User:Up>ImmediateTransmit

[CP_SWS_LSduR_91010]{DRAFT} **Definition of API function LSduR_<User:Up>ImmediateTransmit** ⌈

Service Name	LSduR_<User:Up>ImmediateTransmit (draft)	
Syntax	`Std_ReturnType LSduR_<User:Up>ImmediateTransmit ( PduIdType TxPduId, const PduInfoType* PduInfoPtr )`	
Service ID [hex]	0x5	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId	
Parameters (in)	TxPduId	Identifier of the PDU to be transmitted
	PduInfoPtr	Length of and pointer to the PDU data and pointer to MetaData.
Parameters (inout)	None	
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: Transmit request has been accepted. `E_NOT_OK`: Transmit request has not been accepted.

▽

△

Description	Requests transmission of a PDU for an immediate transmit. **Tags:** atp.Status=draft
Available via	LSduR_<module>.h

⌋(*SRS_GTW_06144*, *SRS_GTW_06145*, *SRS_BSW_00310*)

8.3.2.3 LSduR_<User:Up>ReleaseRxBuffer

[CP_SWS_LSduR_91012]{DRAFT} **Definition of API function LSduR_<User:Up>ReleaseRxBuffer** ⌈

Service Name	LSduR_<User:Up>ReleaseRxBuffer (draft)	

Syntax	`void LSduR_<User:Up>ReleaseRxBuffer (`
	`  PduIdType RxPduId`
	`)`	

Service ID [hex]	0x7	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId	
Parameters (in)	RxPduId	Identifier of the received PDU.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Indication from the upper layer to release the lower layer reception buffer. **Tags:** atp.Status=draft	
Available via	LSduR_<module>.h	

⌋(*SRS_GTW_06144*, *SRS_GTW_06145*, *SRS_BSW_00310*)

8.3.3 Configurable interfaces definitions for lower layer communication interface module interaction

Since the API description now has a generic approach, the `serviceId`s of the lower layer API functions are generic as well. To differentiate between several lower layers, the LSduR uses the `moduleId`s of the lower layer modules as the `instanceId` argument in the Det call originated from APIs listed in this section.

8.3.3.1 LSduR_<User:Lo>RxIndication

[CP_SWS_LSduR_91011]{DRAFT} **Definition of callback function LSduR_<User:Lo>RxIndication** ⌈

Service Name	LSduR_<User:Lo>RxIndication (draft)	
Syntax	`void LSduR_<User:Lo>RxIndication ( PduIdType id, Std_ReturnType result )`	
Service ID [hex]	0x45	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	id	Identification of the received I-PDU.
	result	E_OK: The PDU was received. E_NOT_OK: Reception of the PDU failed.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	Called after an I-PDU has been received via the TP API, the result indicates whether the transmission was successful or not. **Tags:** atp.Status=draft	
Available via	LSduR_<module>.h	

⌋(*SRS_GTW_06144*, *SRS_GTW_06145*, *SRS_BSW_00310*)

8.3.3.2 LSduR_<User:Lo>TxConfirmation

[CP_SWS_LSduR_91013]{DRAFT} **Definition of callback function LSduR_<User:Lo>TxConfirmation** ⌈

Service Name	LSduR_<User:Lo>TxConfirmation (draft)	
Syntax	`void LSduR_<User:Lo>TxConfirmation ( PduIdType TxPduId, Std_ReturnType result )`	
Service ID [hex]	0x40	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	TxPduId	ID of the PDU that has been transmitted.
	result	E_OK: The PDU was transmitted. E_NOT_OK: Transmission of the PDU failed.
Parameters (inout)	None	
Parameters (out)	None	
Return value	None	
Description	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU. **Tags:** atp.Status=draft	
Available via	LSduR_<module>.h	

⌋(*SRS_GTW_06144*, *SRS_GTW_06145*, *SRS_BSW_00310*)

8.3.3.3 LSduR_<User:Lo>TriggerTransmit

[CP_SWS_LSduR_91014]{DRAFT} **Definition of callback function LSduR_<User:Lo>TriggerTransmit** ⌈

Service Name	LSduR_<User:Lo>TriggerTransmit (draft)	
Syntax	`Std_ReturnType LSduR_<User:Lo>TriggerTransmit ( PduIdType TxPduId, PduInfoType* PduInfoPtr )`	
Service ID [hex]	0x41	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different PduIds. Non reentrant for the same PduId.	
Parameters (in)	TxPduId	ID of the SDU that is requested to be transmitted.
Parameters (inout)	PduInfoPtr	Contains a pointer to a buffer (SduDataPtr) to where the SDU data shall be copied, and the available buffer size in SduLengh. On return, the service will indicate the length of the copied SDU data in SduLength.
Parameters (out)	None	
Return value	Std_ReturnType	`E_OK`: SDU has been copied and SduLength indicates the number of copied bytes. `E_NOT_OK`: No SDU data has been copied. PduInfoPtr must not be used since it may contain a NULL pointer or point to invalid data.
Description	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr. **Tags:** atp.Status=draft	
Available via	LSduR_<module>.h	

⌋(*SRS_GTW_06144*, *SRS_GTW_06145*, *SRS_BSW_00310*)

8.4 Callback notifications

There are no callback notifications defined.

8.5 Scheduled functions

As any L-SDU Router operation is triggered by an adjacent communication module the L-SDU Router does not require scheduled functions.

8.6 Expected interfaces

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

The L-SDU router module is modeled as a generic module that can interface to different upper and lower modules. The approach taken to model this generic approach is to

have a virtual module called GenericComServices. This virtual module contains a set of APIs that the L-SDU router will call in upper layer or lower layer modules. These APIs are generic in the way that they contain a tag <Lo> and <Up> that is replaced with the interfaced module. The tag is set by the configuration in the LSduRBswModules container using the LSduRBswModuleRef reference parameter.

8.6.1 Mandatory interfaces

The L-SDU Router does not require mandatory interfaces. The required API functions depend on the configuration.

[CP_SWS_LSduR_91015] Definition of mandatory interfaces in module LSduR ⌈

API Function	Header File	Description
Det_ReportRuntimeError	Det.h	Service to report runtime errors. If a callout has been configured then this callout shall be called.

⌋(*SRS_BSW_00384*)

8.6.2 Optional interfaces

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

[CP_SWS_LSduR_91016] Definition of optional interfaces in module LSduR ⌈

API Function	Header File	Description
<Provider:Lo>_Transmit	–	Requests transmission of a PDU.
<Provider:Up>_RxIndication	–	Indication of a received PDU from a lower layer communication interface module.
<Provider:Up>_TriggerTransmit	–	Within this API, the upper layer module (called module) shall check whether the available data fits into the buffer size reported by PduInfoPtr->SduLength. If it fits, it shall copy its data into the buffer provided by PduInfoPtr->SduDataPtr and update the length of the actual copied data in PduInfoPtr->SduLength. If not, it returns E_NOT_OK without changing PduInfoPtr.
<Provider:Up>_TxConfirmation	–	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU.

⌋(*SRS_BSW_00384*)

8.7 Service Interfaces

There are no service interfaces defined.

9 Sequence diagrams

The goal of this chapter is to make the understanding of the PDU Router easier. For this purpose sequence diagrams which show different communication scenarios are used. Please consider that the sequence diagrams are not exhaustive and are only used to support the functional specification (Chapter 7) and API specification (Chapter 8)

Focus of the sequence diagrams is the L-SDU Router and therefore interactions between other modules (e.g. between an interface and its driver) are not shown.

Note: The diagrams in this chapter show specific use-cases. They do not reflect requirements for an implementation of the L-SDU Router module.

9.1 L-SDU transmission

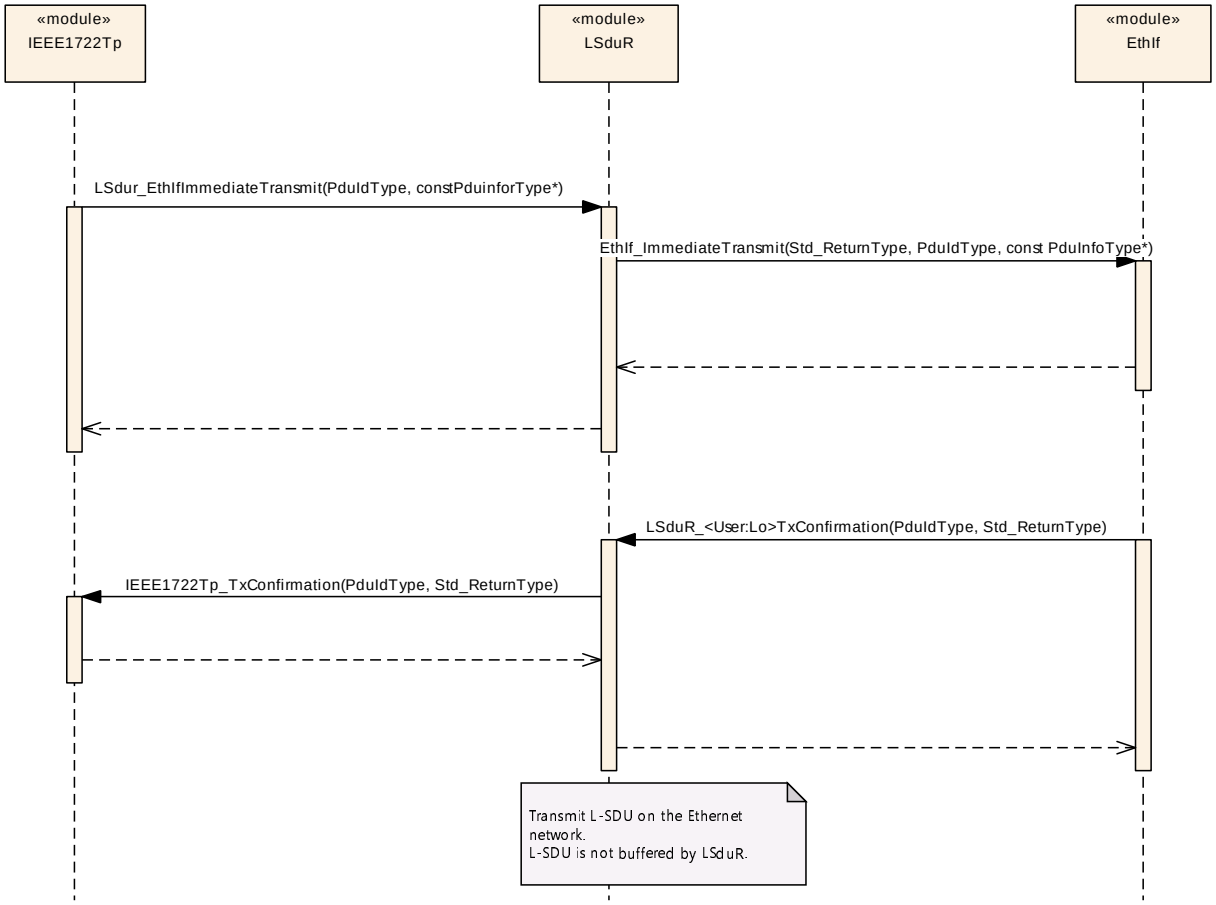

Figure 9.1: IEEE1722Tp to EthIf L-SDU transmission

9.2 L-SDU reception

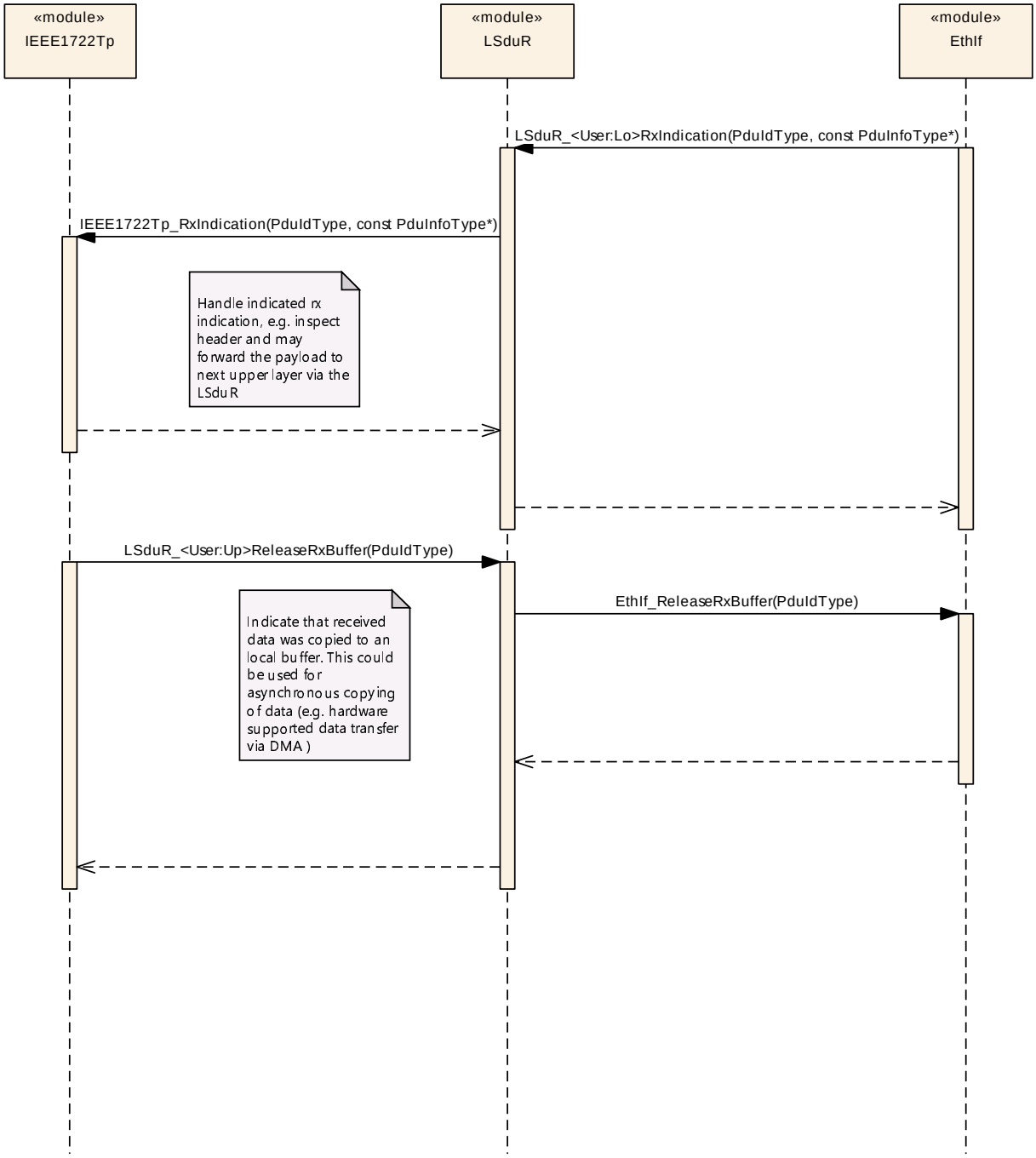

Figure 9.2: EthIf to IEEE1722Tp L-SDU reception

10 Configuration specification

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

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

Chapter 10.3 specifies published information of the module LSduR.

10.1 How to read this chapter

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

10.1.1 Variants

[CP_SWS_LSduR_00037]{DRAFT} ⌈The L-SDU Router module shall support the update of the routing configuration (i.e. the L-SDU Router routing tables) at post build-time if this variant is supported.⌋*(SRS_GTW_06002, SRS_BSW_00404)*

Support of post-build update of the routing table is not always desired. Therefore post-build update of the routing table is only supported in the variant post-build of the L-SDU Router module, see further section 10.1.1.

The post-build comes in two flavors: Selectable and Loadable, there is no restriction on using any of them in the L-SDU Router module or even a combination of them.

[CP_SWS_LSduR_00038]{DRAFT} ⌈If the variant post-build is supported, the update of the routing tables shall only be possible when the L-SDU Router module is uninitialized.⌋*(SRS_GTW_06001, SRS_BSW_00404)*

Remark: The process how the update of the routing tables is performed is not restricted. Most likely a reflashing of the memory segment that holds the table will be done by the bootloader - a separate program which may be loaded after a reboot to update the ECU.

[CP_SWS_LSduR_00039]{DRAFT} ⌈The post-build time configuration of the L-SDU Router module shall be identifiable by the unique configuration identifier: `LSduRConfigurationId`⌋*(SRS_GTW_06097, SRS_BSW_00404)*

Remark: The unique configuration identifier is not used to select one of multiple post-build configuration sets of the L-SDU Router module, but for unique identification of the current L-SDU Router module post-build configuration, e.g. for Diagnostics or for checking at runtime that the post-build configurations of related communication mod-

ules match. The configuration identifier can be read via the API `LSduR_GetConfigurationId` see section 8.3.1.3.

10.2 Containers and configuration parameters

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

10.2.1 LSduR

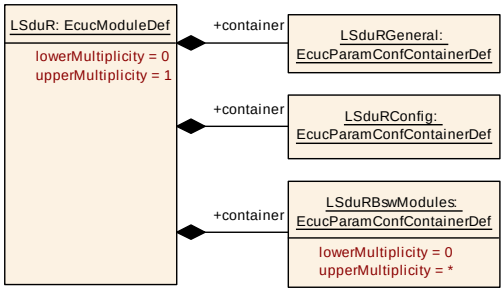

Figure 10.1: LSduR

SWS Item	[ECUC_LSduR_00001]
Module Name	LSduR
Description	Configuration of the LSduR module.
Post-Build Variant Support	true
Supported Config Variants	VARIANT-LINK-TIME, VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
LSduRBswModules	0..*	Each container describes a specific BSW module (upper/CDD/ lower/IEEE1722Tp) that the L-SDU Router shall interface to. The reason to have it as own configuration container instead of implication of the routing path is to be able to configure CDDs properly and to force modules to be used in a post-build situation even though no routing is made to/from this module (future configurations may include these modules). **Tags:** atp.Status=draft
LSduRConfig	1	This container contains the configuration parameters and sub containers of the AUTOSAR LSduR module. **Tags:** atp.Status=draft
LSduRGeneral	1	Specifies the general configuration parameters of the LSduR. **Tags:** atp.Status=draft

10.2.2 LSduRGeneral

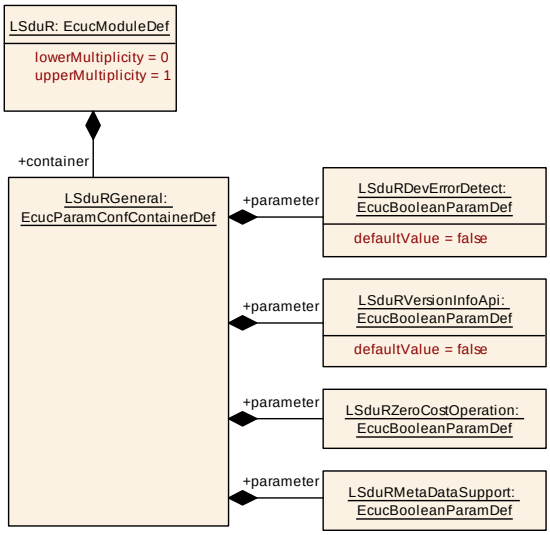

Figure 10.2: LSduRGeneral

SWS Item	**[ECUC_LSduR_00002]**
Container Name	LSduRGeneral
Parent Container	LSduR
Description	Specifies the general configuration parameters of the LSduR. **Tags:** atp.Status=draft
Configuration Parameters	

SWS Item	**[ECUC_LSduR_00003]**		
Parameter Name	LSduRDevErrorDetect		
Parent Container	LSduRGeneral		
Description	Switches the development error detection and notification on or off. • true: detection and notification is enabled. • false: detection and notification is disabled. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_LSduR_00014]**
Parameter Name	LSduRMetaDataSupport
Parent Container	LSduRGeneral

▽

△

Description	Enable support for MetaData handling. The MetaData is defined by the referenced MetaDataType of the global PDU definitions. This feature may be used for efficient forwarding of frame attributes (e.g. EtherType), where the MetaData contains the Ether Type. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_LSduR_00004]**		
Parameter Name	LSduRVersionInfoApi		
Parent Container	LSduRGeneral		
Description	If true the LSduR_GetVersionInfo API is available. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

SWS Item	**[ECUC_LSduR_00013]**		
Parameter Name	LSduRZeroCostOperation		
Parent Container	LSduRGeneral		
Description	If set, the LSduR configuration generator will report an error if zero-cost-operation cannot be fulfilled. This parameter shall be seen as an input requirement to the configuration generator. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

10.2.3 LSduRConfig

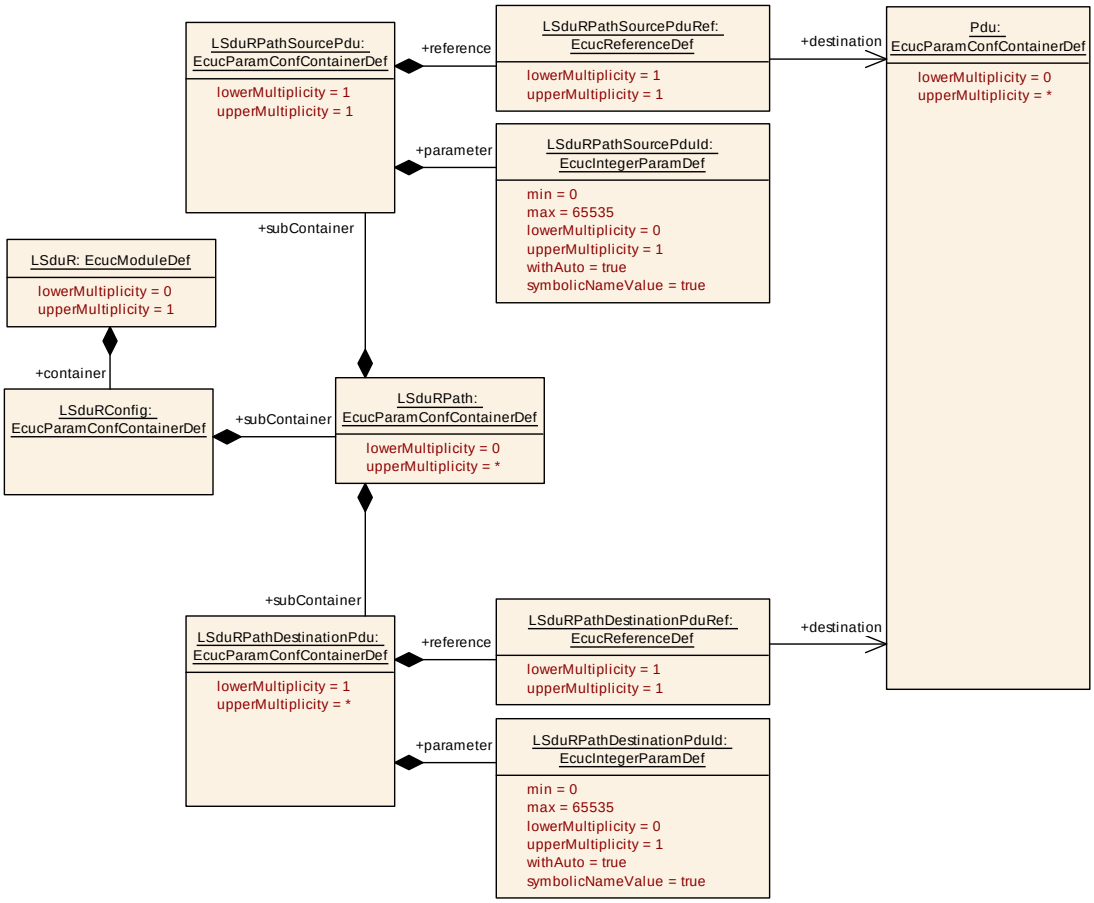

Figure 10.3: LSduRConfig

SWS Item	[ECUC_LSduR_00005]
Container Name	LSduRConfig
Parent Container	LSduR
Description	This container contains the configuration parameters and sub containers of the AUTOSAR LSduR module. **Tags:** atp.Status=draft
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
LSduRPath	0..*	This container defines a LSduR path (1:1 or 1:n) for one source Pdu to 1 or n destination Pdus. **Tags:** atp.Status=draft

10.2.4 LSduRPath

SWS Item	[ECUC_LSduR_00006]		
Container Name	LSduRPath		
Parent Container	LSduRConfig		
Description	This container defines a LSduR path (1:1 or 1:n) for one source Pdu to 1 or n destination Pdus. **Tags:** atp.Status=draft		
Post-Build Variant Multiplicity	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Configuration Parameters			

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
LSduRPathDestinationPdu	1..*	This container defines the EcuC Pdu representing one or more destinations of the routing path. **Tags:** atp.Status=draft
LSduRPathSourcePdu	1	This container defines the EcuC Pdu representing the source of the routing path. **Tags:** atp.Status=draft

SWS Item	[ECUC_LSduR_00007]
Container Name	LSduRPathSourcePdu
Parent Container	LSduRPath
Description	This container defines the EcuC Pdu representing the source of the routing path. **Tags:** atp.Status=draft
Configuration Parameters	

SWS Item	[ECUC_LSduR_00009]		
Parameter Name	LSduRPathSourcePduId		
Parent Container	LSduRPathSourcePdu		
Description	Definition of the Handle Pdu Id representing the source of the routing path. **Tags:** atp.Status=draft		
Multiplicity	0..1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD

△

Scope / Dependency	scope: ECU withAuto = true

SWS Item	**[ECUC_LSduR_00008]**		
Parameter Name	LSduRPathSourcePduRef		
Parent Container	LSduRPathSourcePdu		
Description	Reference to the EcuC Pdu representing the source of the routing path. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	Reference to Pdu		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

No Included Containers

SWS Item	**[ECUC_LSduR_00010]**		
Container Name	LSduRPathDestinationPdu		
Parent Container	LSduRPath		
Description	This container defines the EcuC Pdu representing one or more destinations of the routing path. **Tags:** atp.Status=draft		
Post-Build Variant Multiplicity	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Configuration Parameters			

SWS Item	**[ECUC_LSduR_00012]**		
Parameter Name	LSduRPathDestinationPduId		
Parent Container	LSduRPathDestinationPdu		
Description	Definition of the Handle Pdu Id representing the destination of the routing path. **Tags:** atp.Status=draft		
Multiplicity	0..1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default value	–		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD

▽

△

Scope / Dependency	scope: ECU withAuto = true

SWS Item	**[ECUC_LSduR_00011]**		
Parameter Name	LSduRPathDestinationPduRef		
Parent Container	LSduRPathDestinationPdu		
Description	Reference to the EcuC Pdu representing one destination of the routing path. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	Reference to Pdu		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	X	VARIANT-LINK-TIME
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

No Included Containers

10.2.5 LSduRBswModules

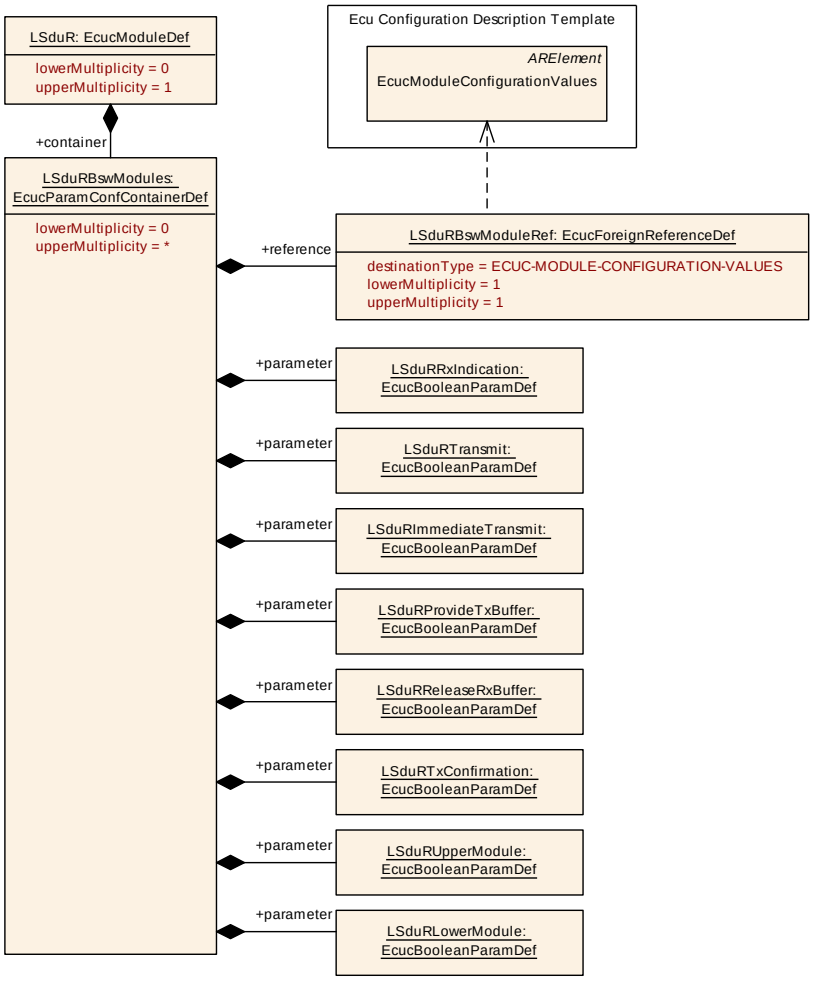

Figure 10.4: LSduRBswModules

SWS Item	**[ECUC_LSduR_00015]**		
Container Name	LSduRBswModules		
Parent Container	LSduR		
Description	Each container describes a specific BSW module (upper/CDD/lower/IEEE1722Tp) that the L-SDU Router shall interface to. The reason to have it as own configuration container instead of implication of the routing path is to be able to configure CDDs properly and to force modules to be used in a post-build situation even though no routing is made to/from this module (future configurations may include these modules). **Tags:** atp.Status=draft		
Post-Build Variant Multiplicity	false		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE, VARIANT-POST-BUILD
	Link time	–	
	Post-build time	–	
Configuration Parameters			

SWS Item	[ECUC_LSduR_00018]		
Parameter Name	LSduRImmediateTransmit		
Parent Container	LSduRBswModules		
Description	Specifies if BSW module supports the (IF) ImmediateTransmit API or not. Value true the API is supported. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

SWS Item	[ECUC_LSduR_00024]		
Parameter Name	LSduRLowerModule		
Parent Container	LSduRBswModules		
Description	The LSduRLowerModule will decide who will call the APIs and who will implement the APIs. For example, if the EthIf module is referenced then the L-SDU Router module will implement the LSduR_EthIfRxIndication API and the L-SDU Router module will call the EthIf_Transmit API. Other APIs are of course also covered. An upper module can also be an lower module (e.g. the IEEE1722Tp module). **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

SWS Item	[ECUC_LSduR_00019]		
Parameter Name	LSduRProvideTxBuffer		
Parent Container	LSduRBswModules		
Description	Specifies if BSW module supports the (IF) ProvideTxBuffer API or not. Value true the API is supported. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

SWS Item	[ECUC_LSduR_00020]		
Parameter Name	LSduRReleaseRxBuffer		
Parent Container	LSduRBswModules		
Description	Specifies if BSW module supports the ReleaseRxBuffer API or not. Value true the API is supported. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

SWS Item	[ECUC_LSduR_00016]		
Parameter Name	LSduRRxIndication		
Parent Container	LSduRBswModules		
Description	Specifies if BSW module supports the RxIndication API or not. Value true the API is supported. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

SWS Item	[ECUC_LSduR_00017]		
Parameter Name	LSduRTransmit		
Parent Container	LSduRBswModules		
Description	Specifies if BSW module supports the (IF) Transmit API or not. Value true the API is supported. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

SWS Item	[ECUC_LSduR_00021]
Parameter Name	LSduRTxConfirmation
Parent Container	LSduRBswModules

▽

△

Description	Specifies if the BSW module supports the TxConfirmation API or not. Value true the API is supported. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

SWS Item	**[ECUC_LSduR_00023]**		
Parameter Name	LSduRUpperModule		
Parent Container	LSduRBswModules		
Description	The LSduRUpperModule will decide who will call the APIs and who will implement the APIs. For example, if the IEEE1722Tp module is referenced then the L-SDU Router module will implement the LSduR_Transmit API and the L-SDU Router module will call the IEEE1722_RxIndication API. Other APIs are of course also covered. An upper module can also be an lower module (e.g. the IEEE1722Tp module). **Tags:** atp.Status=draft		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	–		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

SWS Item	**[ECUC_LSduR_00022]**		
Parameter Name	LSduRBswModuleRef		
Parent Container	LSduRBswModules		
Description	This is a reference to one BSW module's configuration (i.e. not the ECUC parameter definition template). Example, there could be several configurations of EthIf and this reference selects one of them. **Tags:** atp.Status=draft		
Multiplicity	1		
Type	Foreign reference to ECUC-MODULE-CONFIGURATION-VALUES		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.3 Published Information

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

A Not applicable requirements

[SWS_LSduR_NA]{DRAFT} ⌈These requirements are not applicable to this specification.⌋*(SRS_BSW_00493, SRS_BSW_00492, SRS_BSW_00491, SRS_BSW_00490, SRS_BSW_00489, SRS_BSW_00488, SRS_BSW_00487, SRS_BSW_00486, SRS_BSW_00485, SRS_BSW_00484, SRS_BSW_00483, SRS_BSW_00482, SRS_BSW_00481, SRS_BSW_00480, SRS_BSW_00479, SRS_BSW_00478, SRS_BSW_00477, SRS_BSW_00473, SRS_BSW_00472, SRS_BSW_00471, SRS_BSW_00470, SRS_BSW_00469, SRS_BSW_00467, SRS_BSW_00466, SRS_BSW_00465, SRS_BSW_00464, SRS_BSW_00463, SRS_BSW_00462, SRS_BSW_00461, SRS_BSW_00460, SRS_BSW_00459, SRS_BSW_00458, SRS_BSW_00457, SRS_BSW_00456, SRS_BSW_00454, SRS_BSW_00453, SRS_BSW_00451, SRS_BSW_00450, SRS_BSW_00449, SRS_BSW_00448, SRS_BSW_00447, SRS_BSW_00441, SRS_BSW_00440, SRS_BSW_00439, SRS_BSW_00437, SRS_BSW_00004, SRS_BSW_00005, SRS_BSW_00006, SRS_BSW_00007, SRS_BSW_00009, SRS_BSW_00010, SRS_BSW_00159, SRS_BSW_00160, SRS_BSW_00161, SRS_BSW_00162, SRS_BSW_00164, SRS_BSW_00167, SRS_BSW_00168, SRS_BSW_00170, SRS_BSW_00171, SRS_BSW_00172, SRS_BSW_00300, SRS_BSW_00301, SRS_BSW_00302, SRS_BSW_00304, SRS_BSW_00306, SRS_BSW_00307, SRS_BSW_00308, SRS_BSW_00309, SRS_BSW_00312, SRS_BSW_00314, SRS_BSW_00318, SRS_BSW_00321, SRS_BSW_00325, SRS_BSW_00327, SRS_BSW_00328, SRS_BSW_00330, SRS_BSW_00331, SRS_BSW_00333, SRS_BSW_00334, SRS_BSW_00336, SRS_BSW_00339, SRS_BSW_00341, SRS_BSW_00342, SRS_BSW_00343, SRS_BSW_00344, SRS_BSW_00345, SRS_BSW_00346, SRS_BSW_00347, SRS_BSW_00348, SRS_BSW_00351, SRS_BSW_00353, SRS_BSW_00357, SRS_BSW_00359, SRS_BSW_00360, SRS_BSW_00369, SRS_BSW_00373, SRS_BSW_00374, SRS_BSW_00375, SRS_BSW_00377, SRS_BSW_00378, SRS_BSW_00379, SRS_BSW_00380, SRS_BSW_00383, SRS_BSW_00385, SRS_BSW_00386, SRS_BSW_00388, SRS_BSW_00389, SRS_BSW_00390, SRS_BSW_00392, SRS_BSW_00393, SRS_BSW_00394, SRS_BSW_00395, SRS_BSW_00396, SRS_BSW_00397, SRS_BSW_00398, SRS_BSW_00399, SRS_BSW_00401, SRS_BSW_00402, SRS_BSW_00403, SRS_BSW_00407, SRS_BSW_00408, SRS_BSW_00409, SRS_BSW_00410, SRS_BSW_00411, SRS_BSW_00413, SRS_BSW_00416, SRS_BSW_00417, SRS_BSW_00419, SRS_BSW_00422, SRS_BSW_00423, SRS_BSW_00424, SRS_BSW_00425, SRS_BSW_00426, SRS_BSW_00427, SRS_BSW_00428, SRS_BSW_00429, SRS_BSW_00432, SRS_BSW_00433, SRS_BSW_00494)*

B Change history of AUTOSAR traceable items

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

B.1 Traceable item history of this document according to AUTOSAR Release R23-11

B.1.1 Added Specification Items in R23-11

[CP_SWS_LSduR_00001] [CP_SWS_LSduR_00002] [CP_SWS_LSduR_00003]
[CP_SWS_LSduR_00004] [CP_SWS_LSduR_00005] [CP_SWS_LSduR_00006]
[CP_SWS_LSduR_00007] [CP_SWS_LSduR_00008] [CP_SWS_LSduR_00009]
[CP_SWS_LSduR_00010] [CP_SWS_LSduR_00011] [CP_SWS_LSduR_00012]
[CP_SWS_LSduR_00013] [CP_SWS_LSduR_00014] [CP_SWS_LSduR_00015]
[CP_SWS_LSduR_00016] [CP_SWS_LSduR_00017] [CP_SWS_LSduR_00018]
[CP_SWS_LSduR_00019] [CP_SWS_LSduR_00020] [CP_SWS_LSduR_00021]
[CP_SWS_LSduR_00022] [CP_SWS_LSduR_00023] [CP_SWS_LSduR_00024]
[CP_SWS_LSduR_00025] [CP_SWS_LSduR_00026] [CP_SWS_LSduR_00027]
[CP_SWS_LSduR_00028] [CP_SWS_LSduR_00029] [CP_SWS_LSduR_00030]
[CP_SWS_LSduR_00031] [CP_SWS_LSduR_00032] [CP_SWS_LSduR_00033]
[CP_SWS_LSduR_00034] [CP_SWS_LSduR_00035] [CP_SWS_LSduR_00036]
[CP_SWS_LSduR_00037] [CP_SWS_LSduR_00038] [CP_SWS_LSduR_00039]
[CP_SWS_LSduR_91001] [CP_SWS_LSduR_91002] [CP_SWS_LSduR_91003]
[CP_SWS_LSduR_91004] [CP_SWS_LSduR_91005] [CP_SWS_LSduR_91006]
[CP_SWS_LSduR_91007] [CP_SWS_LSduR_91008] [CP_SWS_LSduR_91009]
[CP_SWS_LSduR_91010] [CP_SWS_LSduR_91011] [CP_SWS_LSduR_91012]
[CP_SWS_LSduR_91013] [CP_SWS_LSduR_91014] [CP_SWS_LSduR_91015]
[CP_SWS_LSduR_91016] [CP_SWS_LSduR_91017] [SWS_LSduR_NA]

B.1.2 Changed Specification Items in R23-11

none

B.1.3 Deleted Specification Items in R23-11

none