


SCOPE OF APPLICATION All Project/Engineering		SHT/SHTS 1 / 28
Responsibility: Classic AUTOSAR Team	AUTOSAR IpduM User Manual	DOC. NO
AUTOSAR IpduM User Manual		

Document Change History			
Date (YYYY-MM-DD)	Ver.	Editor	Content
2021-03-07	1.0.0.0	TuanNM30	IpduM UM Initial Version
2021-09-15	1.0.0.1	DuyNHP2	Update Module version Update Change Log
2022-03-10	1.0.1.0	DuyNHP2	Update Module version Update Change Log
2022-04-29	1.0.2.0	DuyNHP2	Update Module version Update Change Log
2022-06-28	1.0.2.1	DuyNHP2	Update Change Log
2022-07-22	1.0.3.0	DuyNHP2	Update Module version Update Change Log
2022-08-12	1.0.3.1	DuyNHP2	Update Change Log
2022-11-06	1.0.4.0	NhatTD2	Update Module version Update Change Log Update Generator Error Message
2022-11-15	1.0.4.1	NhatTD2	Update Module version Update Change Log
2022-12-15	1.0.5.0	NhatTD2	Update Module version Update Change Log
2023-05-12	1.0.6.0	HaoTA4	Update Module version Update Change Log Update Configuration Guide
2023-07-07	1.0.7.0	HaoTA4	Update Module version Update Change Log
2024-01-26	1.0.7.1	Jacqueline Catley	Update Module version Update Change Log
2024-03-27	1.0.8.0	DH Kim	Apply New UM Template Update Generator Error Message

Edition Date: 2024/03/27	File Name IpduM_UM.pdf	Creation DH Kim	Check HM Kim	Approval JS Jang
Document Management System		2024/03/27	2024/03/27	2024/03/27

## Table of Contents

1. Overview .....	4
2. Reference.....	4
3. AUTOSAR System.....	5
3.1 Overview of Software Layers .....	5
4. Limitations and Deviations .....	6
4.1 Limitations.....	6
4.2 Deviations.....	6
5. Configuration Guide .....	7
5.1. IpduMGeneral Container.....	7
5.2. IpduMPublishedInformation Container .....	10
5.3. IpduMConfig Container.....	10
5.3.1. IpduMConfig-IpduMTxPathway setting.....	10
5.3.2. IpduMConfig-IpduMRxPathway setting.....	13
5.3.3. IpduMConfig-IpduMContainerTxPdu setting .....	15
5.3.4. IpduMConfig-IpduMContainedTxPdu setting .....	16
5.3.5. IpduMConfig-IpduMContainerRxPdu .....	17
5.3.6. IpduMConfig-IpduMContainedRxPdu setting .....	18
6. Application Programming Interface (API) .....	19
6.1 Type Definitions .....	19
6.2 Macro Constants .....	19
6.3 Function .....	19
6.3.1 Initialization.....	19
6.3.2 Transmission and reception .....	20
6.3.3 Scheduled function .....	22
7. Generator .....	24
7.1 Generator Option .....	24
7.1 Generator Message .....	24
7.1.1 Error Messages.....	24

7.1.2 Warning Messages .....	27
7.1.3 Information Messages .....	27
8. SWP Error Code .....	28
9. Appendix .....	28

## 1. Overview

It is written based on AUTOSAR standard SRS / SWS. If more detailed functional explanation is needed when using the module, see the Reference Manual. The interpretation of setting related category is as follows:

- Changeable (C): Items that can be set by the user
- Fixed (F): Items that cannot be changed by the user.
- Not Supported (N): Deprecated item

## 2. Reference

Sl. No.	Title	Version
1	AUTOSAR_SWS_IPDUMultiplexer.pdf	4.4.0
2	AUTOSAR_SWS_COM.pdf	4.4.0

## 3. AUTOSAR System

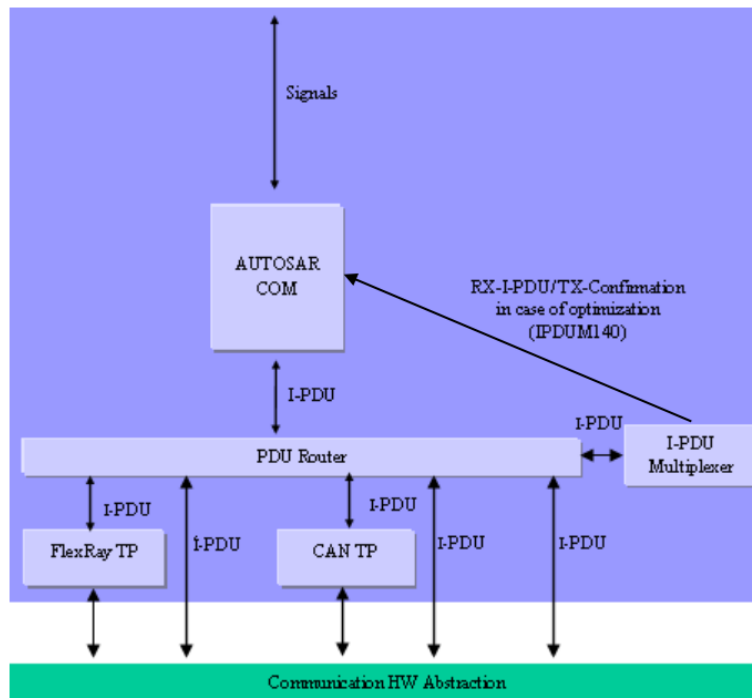
### 3.1 Overview of Software Layers

This specification describes the functionality, APIs and the configuration of the AUTOSAR Basic Software module I-PDU Multiplexer IpduM.

PDU multiplexing means using the same PCI (Protocol Control Information) of a PDU (Protocol Data Unit) with more than one unique layout of its SDU (Service Data Unit). A selector field is a piece of the SDU of the multiplexed PDU. It is used to distinguish the contents of the multiplexed PDUs from each other.

Multiplexing of PDUs is currently known from CAN, but is not restricted to this communication system.

On sender-side, the I-PDU Multiplexer module is responsible to combine appropriate I-PDUs from COM to new, multiplexed I-PDUs and send them back to the PDU Router. On receiver-side, it is responsible to interpret the content of multiplexed I-PDUs and provide COM with its appropriate separated I-PDUs taking into account the value of the selector field.



## **4. Limitations and Deviations**

### **4.1 Limitations**

None

### **4.2 Deviations**

None

## 5. Configuration Guide

The IpduM setting of the AUTOSAR platform distributed by Hyundai AutoEver is a setting reflecting Hyundai AutoEver Policy's policy. Therefore, you should consult with Hyundai AutoEver.

The following chapters summarize all configuration parameters of module

### 5.1. IpduMGeneral Container

Contains the general configuration parameters of IpduM.

Parameter Name	Value	Category
IpduMContainedTxPduPriorityHandling	True/False	C
IpduMMetaDataSupport	True/False	C
IpduMRxTimeBase	0..Inf	C
IpduMTxTimeBase	0..Inf	C
IpduMHeaderByteOrder	User Defined	C
IpduMDevErrorDetect	True/False	C
IpduMStaticPartExists	True/False	C
IpduMVersionInfoApi	True/False	C
IpduMEnableModule	True/False	C

- 1) Contained Tx Pdu Priority Handling:** This parameter enables/disables handling of priority for IpduMContainedTxPdu's with IpduMContainedTxPduCollectionSemantics IPDUM\_LAST\_IS\_BEST.
  - + true: enabled
  - + false: disabled
- 2) Meta Data Support:** This parameter enables/disables the support of meta-data feature.
  - + true: enabled
  - + false: disabled
- 3) Rx Time Base:** The period between successive calls to IpduM\_MainFunctionRx in seconds. This parameter may be used by the IpduM generator to transform the values of the reception related timing configuration parameters of the IpduM module to internal implementation specific counter or tick values. The IpduM module's internal timing handling is implementation specific.  
The IpduM module (generator) may rely on the fact that IpduM\_MainFunctionRx is scheduled according to the value configured here.
- 4) Tx Time Base:** The period between successive calls to IpduM\_MainFunctionTx in seconds. This parameter may be used by the IpduM generator to transform the values of the reception related timing configuration parameters of the IpduM module to internal implementation specific counter or tick values. The IpduM module's internal timing handling is implementation specific.  
The IpduM module (generator) may rely on the fact that IpduM\_MainFunctionTx is scheduled according to the value configured here.
- 5) Header Byte Order:** This parameter defines the ByteOrder of the headers inside a Container I-PDU.
  - + IPDUM\_BIG\_ENDIAN: Headers inside a Container I-PDU shall be ordered big endian.
  - + IPDUM\_LITTLE\_ENDIAN: Headers inside a Container I-PDU shall be ordered little endian.
- 6) Dev Error Detect:** Switches the development error detection and notification on or off.

- + true: detection and notification is enabled.
- + false: detection and notification is disabled.

**7) Static Part Exists:** This is to allow optimizations in the case the IpduM will never be used with a static part. Note that this is a pre-compile option. If this is set to False then it will not be possible to add static parts after compilation.

- + True: A static part may exist.
- + False: A static part will never exist.

**8) Version Info Api:** Active/Deactivate the version information API.

- + true: version information activated
- + false: version information deactivated

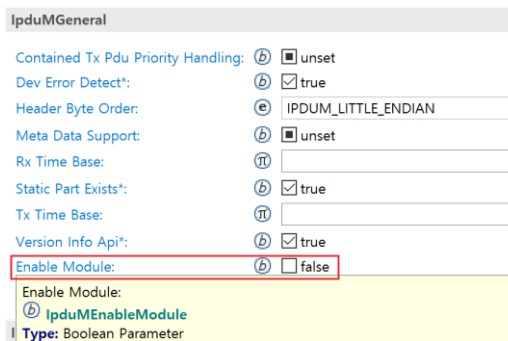
**9) Enable Module:** Settings that allow you to decide whether to use IpduM itself or not, even if IpduM is integrated.

This can support in PB too and “unset”, “false” will be operated like Not Used, and “true” can use the module normally.

- + true: enabled
- + false: disabled

For using this, “Ecud\_PduR”, “Ecud\_IpduM” have to config like following as.

1. Untick the “Enable Module” in “Ecud\_IpduM”



2. For PDUs below “IpduMConfig”, if Some container or parameter is referring in PduR, delete them in “Ecud\_PduR”

Ex. In “Rx Pathway” of “IpduMConfig”, If PduR has some container that refer in “Pdu\_MultiplexedIPdu\_HS\_CAN1\_EMS22”, “Pdu\_ISignalIPdu\_HS\_CAN1\_EMS22\_m<0,1,2,3>”, “Pdu\_ISignalIPdu\_HS\_CAN1\_EMS22” of belows configurations, Have to remove in “PduR\_RoutingPath”, “PduR\_SrcPdu”, “PduR\_DestPdu”. “Tx Pathway” too.

[“Pdu\_MultiplexedIPdu\_HS\_CAN1\_EMS22” in IpduM]



The screenshot shows the IpduM configuration tree on the left and its properties on the right. The tree structure is as follows:

- IpduM
  - IpduMConfig
    - Rx Pathway [1]
      - IpduMRxPathWay\_MultiplexedIPdu\_HS\_CAN1\_EMS22
        - IpduMRxIndication\_MultiplexedIPdu\_HS\_CAN1\_EMS22
          - Rx Dynamic Part [4]
            - IpduMRxDynamicPart\_HS\_CAN1\_EMS22\_m0
            - IpduMRxDynamicPart\_HS\_CAN1\_EMS22\_m1
            - IpduMRxDynamicPart\_HS\_CAN1\_EMS22\_m2
            - IpduMRxDynamicPart\_HS\_CAN1\_EMS22\_m3
          - Rx Dynamic Segment [1]
          - Rx Static Segment [7]
            - IpduMSelectorField
            - IpduMRxStaticPart\_HS\_CAN1\_EMS22
  - Tx Pathway [1]
    - IpduMTxPathWay\_MultiplexedIPdu\_HS\_CAN1\_EMS12
      - IpduMTxRequest\_MultiplexedIPdu\_HS\_CAN1\_EMS12

The properties on the right are:

- Short Name\*: IpduMRxIndication\_MultiplexedIPdu\_HS\_CAN1\_EMS22
- Byte Order\*: LITTLE\_ENDIAN
- Rx Handle Id\*: 0
- Pdu Ref\*: Pdu\_MultiplexedIPdu\_HS\_CAN1\_EMS22 [/AUTOSAR/EcuC/EcuCConfigSet/EcuCConfigSet]

Property	Value
Rx Dynamic Part	4 [0..*]
Rx Dynamic Segment	1 [0..*]
Rx Static Part	1 [0..1]
Rx Static Segment	7 [0..*]
Selector Field	1 [1]

[“Pdu\_ISignalIPdu\_HS\_CAN1\_EMS22\_m<0,1,2,3>” in IpduM]

The screenshot shows the IpduM configuration tree on the left and the Container Details for IpduMRxDynamicPart on the right. The tree structure is the same as in the previous screenshot. The Container Details table is as follows:

Index	Short Name	Rx Selector Value	Outgoing Dynamic Pdu Ref
0	IpduMRxDynamicPart_HS_CAN1_EMS22_m0	0	Pdu_ISignalIPdu_HS_CAN1_EMS22_m0
1	IpduMRxDynamicPart_HS_CAN1_EMS22_m1	1	Pdu_ISignalIPdu_HS_CAN1_EMS22_m1
2	IpduMRxDynamicPart_HS_CAN1_EMS22_m2	2	Pdu_ISignalIPdu_HS_CAN1_EMS22_m2
3	IpduMRxDynamicPart_HS_CAN1_EMS22_m3	3	Pdu_ISignalIPdu_HS_CAN1_EMS22_m3

[“Pdu\_ISignalIPdu\_HS\_CAN1\_EMS22” in IpduM]

The screenshot shows the IpduM configuration tree on the left and the Container Details for IpduMRxStaticPart on the right. The tree structure is the same as in the previous screenshot. The Container Details table is as follows:

Property	Value
Short Name*	IpduMRxStaticPart_HS_CAN1_EMS22
Outgoing Static Pdu Ref*	Pdu_ISignalIPdu_HS_CAN1_EMS22 [/AUTOSAR/EcuC/EcuCConfigSet/EcuCConfigSet]

[Delete Container Item in PduR]

The screenshot shows the PduR configuration tree with several items highlighted in red, indicating they are to be deleted. The items are:

- PduR\_MultiplexedIPdu\_HS\_CAN1\_EMS22\_De
- PduR\_HS\_CAN1\_EMS22\_m0\_DestPdu
- PduR\_HS\_CAN1\_EMS22\_m1\_DestPdu
- PduR\_HS\_CAN1\_EMS22\_m2\_DestPdu
- PduR\_HS\_CAN1\_EMS22\_m3\_DestPdu
- PduR\_HS\_CAN1\_EMS22\_DestPdu
- PduR\_MultiplexedIPdu\_HS\_CAN1\_EMS12\_De
- PduR\_HS\_CAN1\_EMS12\_m0\_DestPdu
- PduR\_HS\_CAN1\_EMS12\_m1\_DestPdu
- PduR\_HS\_CAN1\_EMS12\_m2\_DestPdu
- PduR\_HS\_CAN1\_EMS12\_m3\_DestPdu
- PduR\_HS\_CAN1\_EMS12\_DestPdu
- IN\_Multiplexed\_HS\_CAN1\_EMS22\_RoutingPa
- IN\_m0\_HS\_CAN1\_EMS22\_RoutingPath
- IN\_m1\_HS\_CAN1\_EMS22\_RoutingPath
- IN\_m2\_HS\_CAN1\_EMS22\_RoutingPath
- IN\_m3\_HS\_CAN1\_EMS22\_RoutingPath
- IN\_HS\_CAN1\_EMS22\_RoutingPath
- OUT\_Multiplexed\_HS\_CAN1\_EMS12\_Routingf
- OUT\_m0\_HS\_CAN1\_EMS12\_RoutingPath
- OUT\_m1\_HS\_CAN1\_EMS12\_RoutingPath
- OUT\_m2\_HS\_CAN1\_EMS12\_RoutingPath
- OUT\_m3\_HS\_CAN1\_EMS12\_RoutingPath
- OUT\_HS\_CAN1\_EMS12\_RoutingPath
- PduR\_HS\_CAN1\_EMS22\_m0\_SrcPdu
- PduR\_HS\_CAN1\_EMS22\_m1\_SrcPdu
- PduR\_HS\_CAN1\_EMS22\_m2\_SrcPdu
- PduR\_HS\_CAN1\_EMS22\_m3\_SrcPdu
- PduR\_HS\_CAN1\_EMS22\_SrcPdu
- PduR\_MultiplexedIPdu\_HS\_CAN1\_EMS12\_Src
- PduR\_HS\_CAN1\_EMS12\_m0\_SrcPdu
- PduR\_HS\_CAN1\_EMS12\_m1\_SrcPdu
- PduR\_HS\_CAN1\_EMS12\_m2\_SrcPdu
- PduR\_HS\_CAN1\_EMS12\_m3\_SrcPdu
- PduR\_HS\_CAN1\_EMS12\_SrcPdu

3. Then, Generate PduR, IpduM and Compile.

## 5.2. IpduMPublishedInformation Container

Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.

Parameter Name	Value	Category
IpduMRxDirectComInvocation	True/False	C

- 1) **Rx Direct Com Invocation:** If set to TRUE the COM invocation optimization as defined in IPDUM140 is implemented.

## 5.3. IpduMConfig Container

This container contains the sub containers of the IpduM module.

Container Name	Value	Category
IpduMTxPathway	User Defined	C
IpduMRxPathway	User Defined	C
IpduMContainerTxPdu	User Defined	C
IpduMContainedTxPdu	User Defined	C
IpduMContainerRxPdu	User Defined	C
IpduMContainedRxPdu	User Defined	C

- 1) **IpduMTxPathway:** The IpduMTxPathway subcontainer includes information about sent I-PDUs.
- 2) **IpduMRxPathway:** The IpduMRxPathway includes information about received IPDUs.
- 3) **IpduMContainerTxPdu:** The IpduMContainerTxPdu include information about the sending of ContainerPdus
- 4) **IpduMContainedTxPdu:** The IpduMContainedTxPdu include information about the sending of ContainerPdus
- 5) **IpduMContainerRxPdu:** The IpduMContainerRxPdu include information about the reception of ContainerPdus.
- 6) **IpduMContainedRxPdu:** The IpduMContainedRxPdu include information about the reception of ContainerPdus.

### 5.3.1. IpduMConfig-IpduMTxPathway setting

Contains the configuration parameters transmitted I-PDUs by the IpduM module.

Parameter Name	Value	Category
IpduMTxRequest	User Defined	C
IpduMByteOrder	Automated	C
IpduMTxConfirmationPduId	0..65535	C
IpduMTxTriggerMode	Automated	C

IpduMIPduUnusedAreasDefault	0..255	C
IpduMInitialDynamicPart	Automated	C
IpduMOutgoingPduRef	Automated	C
IpduMSelectorField	User Defined	C
IpduMSelectorFieldLength	1..16	C
IpduMSelectorFieldPosition	0..2031	C
IpduMTxDynamicPart	User Defined	C
IpduMJitUpdate	True/False	C
IpduMTxDynamicConfirmation	True/False	C
IpduMTxDynamicHandleId	0..65535	C
IpduMTxDynamicPduRef	Automated	C
IpduMTxDynamicSegment	User Defined	C
IpduMSegmentLength	1..2032	C
IpduMSegmentPosition	0..2031	C
IpduMTxStaticPart	User Defined	C
IpduMJitUpdate	True/False	C
IpduMTxStaticConfirmation	True/False	C
IpduMTxStaticHandleId	0..65535	C
IpduMTxStaticPduRef	Automated	C
IpduMTxStaticSegment	User Defined	C
IpduMSegmentLength	1..2032	C
IpduMSegmentPosition	0..2031	C

**1) IpduMTxRequest:** This container is used to specify the configuration for Transmit requests.

There will be one instance of this container for each I-PDU that can be requested for transmission (the outgoing I-PDUs) by the IpduM.

**2) IpduMByteOrder:** This parameter defines the ByteOrder for all segments (static and dynamic part) and for the selectorField within the MultiplexedPdu.

The absolute position of a segment in the MultiplexedIPdu is determined by the definition of the ByteOrder parameter:

If BIG\_ENDIAN is specified, the SegmentPosition indicates the bit position of the most significant bit in an IPDU.

If LITTLE\_ENDIAN is specified, the SegmentPosition indicates the bit position of the least significant bit in an IPDU.

**3) IpduMTxConfirmationPduId:** Handle Id used by the PduR for confirmation (IpduM\_TxConfirmation) and for TriggerTransmit (IpduM\_TriggerTransmit).

The existence of this parameter is essential for the PduR generation tool to actually find a symbolicNameValue for the OutgoingPdu.

**4) IpduMTxTriggerMode:** Selects whether to send the multiplexed I-PDU immediately or at some later date

- 5) IpduMPduUnusedAreasDefault:** IpduM module fills not used areas of an I-PDU with this bit-pattern.  
If this attribute is omitted the IpduM module does not fill the I-PDU.
- 6) IpduMInitialDynamicPart:** Reference to the dynamic part that shall be used to initialize this multiplexed TX-I-PDU.
- 7) IpduMOutgoingPduRef:** Reference to the PDU defining the outgoing I-PDU.  
When the outgoing I-PDU is sent this is the I-PDU ID to give it. It is the IpduM I-PDU ID of the assembled I-PDU.
- 8) IpduMSelectorField:** This contains the location and the length of the selector field.
- 9) IpduMSelectorFieldLength:** Length of the selector field in bits.
- 10) IpduMSelectorFieldPosition:** Selector field bit position in the multiplexed Pdu.
- 11) IpduMTxDynamicPart:** Configuration parameters for an instance of a TxRequest call into the IpduM. When a Tx Request with the IpduMTxDynamicHandleId is received by the IpduM, all segments (defined in the IpduMDynamicSegment container) are copied from the incoming I-PDU into the outgoing I-PDU buffer and then the send mode honored. This container is used by the dynamic part of a TxRequest configuration. Therefore, for each outgoing I-PDU there will be one instance of this container for the dynamic part.
- 12) IpduMJitUpdate:** This Parameter determines whether to update Jit value.  
If configured to true, fetch the data of this part Just-In-Time via the triggerTransmit API of the PduR.
- 13) IpduMTxDynamicConfirmation:** This Parameter determines whether to generate confirmation for dynamic part. A transmit request can be confirmed by the lower layer. If this parameter is set to true, confirmation of the I-PDU in COM representing the dynamic part is generated.
- 14) IpduMTxDynamicHandleId:** This defines an incoming handle id. When the handle of an incoming Tx Request matches this id, the configured dynamic segments are copied and the IpduMTxTriggerMode is honored.
- 15) IpduMTxDynamicPduRef:** Reference to the Pdu representation in the ECU Configuration Description exchange file to be transmitted.
- 16) IpduMTxDynamicSegment:** The dynamic part of the multiplexed outgoing I-Pdu (referenced by IpduMOutgoingPduRef) can be separated into several segments.  
  
For each segment one IpduMTxDynamicSegment container shall be created that contains the location and the length of the segment.  
  
Please note that each configured segment will be copied out of the source I-Pdu that is referenced in the IpduMTxDynamicPart container and will be copied to the same location in the multiplexed outgoing I-Pdu. The segment layout for all dynamic Parts is always identical.
- 17) IpduMSegmentLength:** Length of the segment in bits.

**18) IpduMSegmentPosition:** Segments bit position in the multiplexed Pdu.

**19) IpduMTxStaticPart:** Configuration parameters for an instance of a Tx\_Request call into the IpduM. When a Tx Request with the IpduMTxStaticHandleId is received by the IpduM, all segments (defined in the IpduMStaticSegment container) are copied from the incoming I-PDU into the outgoing I-PDU buffer and then the send mode honored. This container is used for the static part of a TxRequest configuration. Therefore, for each outgoing I-PDU there will be one instance of this container for the static part if it exists.

**20) IpduMJitUpdate:** This Parameter determines whether to update Jit value.  
If configured to true, fetch the data of this part Just-In-Time via the triggerTransmit API of the PduR.

**21) IpduMTxStaticConfirmation:** This Parameter determines whether to generate confirmation for static part. A transmit request can be confirmed by the lower layer. If this parameter is set to true, confirmation of the I-PDU in COM representing the static part is generated.

**22) IpduMTxStaticHandleId:** This defines an incoming handle id. When the handle of an incoming Tx Request matches this id, the configured static segments are copied and the IpduMTxTriggerMode is honored.

**23) IpduMTxStaticPduRef:** Reference to the Pdu representation in the ECU Configuration Description exchange file to be transmitted.

**24) IpduMTxStaticSegment:** The static part of the multiplexed outgoing I-Pdu (referenced by IpduMOutgoingPduRef) can be separated into several segments.

For each segment one IpduMTxStaticSegment container shall be created that contains the location and the length of the segment.

Please note that each segment in the source I-Pdu that is referenced in the IpduMTxStaticPart container will be copied to the same location in the multiplexed outgoing I-Pdu.

**25) IpduMSegmentLength:** Length of the segment in bits.

**26) IpduMSegmentPosition:** Segments bit position in the multiplexed Pdu.

### 5.3.2. IpduMConfig-IpduMRxPathway setting

Contains the configuration parameters received I-PDUs by the IpduM module.

Parameter Name	Value	Category
IpduMRxIndication	User Defined	C
IpduMByteOrder	Automated	C
IpduMRxHandleId	0..65535	C
IpduMRxIndicationPduRef	Automated	C
IpduMRxDynamicPart	User Defined	C
IpduMRxSelectorValue	0..65535	C
IpduMOutgoingDynamicPduRef	Automated	C

IpduMRxDynamicSegment	User Defined	C
IpduMSegmentLength	1..2032	C
IpduMSegmentPosition	0..2031	C
IpduMRxStaticPart	User Defined	C
IpduMOutgoingStaticPduRef	Automated	C
IpduMRxStaticSegment	User Defined	C
IpduMSegmentLength	1..2032	C
IpduMSegmentPosition	0..2031	C
IpduMSelectorField	User Defined	C
IpduMSelectorFieldLength	1..16	C
IpduMSelectorFieldPosition	0..2031	C

- 1) **IpduMRxIndication:** Contains the configuration for incoming RxIndication calls.
- 2) **IpduMByteOrder:** This parameter defines the ByteOrder for all segments (static and dynamic part) and for the selectorField within the MultiplexedPdu.  
The absolute position of a segment in the MultiplexedIPdu is determined by the definition of the ByteOrder parameter:  
If BIG\_ENDIAN is specified, the SegmentPosition indicates the bit position of the most significant bit in an IPDU.  
If LITTLE\_ENDIAN is specified, the SegmentPosition indicates the bit position of the least significant bit in an IPDU.
- 3) **IpduMRxHandleId:** This is the I-PDU ID of the incoming I-PDU. If an incoming RxIndication's IPDU ID matches this value then it is unpacked according to the specification in this container.
- 4) **IpduMRxIndicationPduRef:** Reference to the received Pdu representation in the ECU Configuration Description exchange file.
- 5) **IpduMRxDynamicPart:** This container contains the configuration for the dynamic part of incoming RxIndication calls. When an incoming received I-PDU's selector field matches the IpduMRxSelectorValue, the new outgoing I-PDU for the dynamic part is constructed as defined by the segments (defined in the IpduMDynamicSegment container) and sent out with the I-PDU ID referenced by IpduMOutgoingDynamicPduRef.  
In case no dynamic part shall be extracted from this received I-PDU this container does not exist. This use-case can occur in case a MultiplexedIPdu is received by an ECU which is only interested in the static part of the MultiplexedIPdu.
- 6) **IpduMRxSelectorValue:** This is the selector value that this container refers to.
- 7) **IpduMOutgoingDynamicPduRef:** When the new I-PDU is sent out it is sent with this I-PDU ID. Reference to the sent PDU representation in the ECU Configuration Description exchange file.
- 8) **IpduMRxDynamicSegment:** The dynamic part of the multiplexed incoming I-Pdu (referenced by IpduMRxIndicationPduRef) can be separated into several segments.  
For each segment one IpduMRxDynamicSegment container shall be created that contains the location and the length of the segment.  
Please note that each configured segment will be copied into the destination I-Pdu that is referenced in the IpduMRxDynamicPart container and will be copied from the same location in the multiplexed incoming I-Pdu. The segment layout for all dynamic Parts is always identical.

- 9) **IpduMSegmentLength:** Length of the segment in bits.
- 10) **IpduMSegmentPosition:** Segments bit position in the multiplexed Pdu.
- 11) **IpduMRxStaticPart:** This container contains the configuration for the static part of incoming RxIndication calls. On reception, the new outgoing I-PDU for the static part is constructed as defined by the segments (defined in the IpduMStaticSegment container) and sent out with the I-PDU ID referenced by IpduMOutgoingStaticPduRef.
- 12) **IpduMOutgoingStaticPduRef:** When the new I-PDU is sent out it is sent with this I-PDU ID. Reference to the sent Pdu representation in the ECU Configuration Description exchange file.
- 13) **IpduMRxStaticSegment:** The static part of the multiplexed incoming I-Pdu (referenced by IpduMRxIndicationPduRef) can be separated into several segments.  
For each segment one IpduMRxStaticSegment container shall be created that contains the location and the length of the segment.  
Please note that each configured segment will be copied into the destination I-Pdu that is referenced in the IpduMRxStaticPart container and will be copied from the same location in the multiplexed incoming I-Pdu.
- 14) **IpduMSegmentLength:** Length of the segment in bits.
- 15) **IpduMSegmentPosition:** Segments bit position in the multiplexed Pdu.
- 16) **IpduMSelectorField:** This is the selector value that this container refers to.
- 17) **IpduMSelectorFieldLength:** Length of the selector field in bits.
- 18) **IpduMSelectorFieldPosition:** Selector field bit position in the multiplexed Pdu.

### 5.3.3. IpduMConfig-IpduMContainerTxPdu setting

Configuration of a transmitted container Pdu.

Parameter Name	Value	Category
IpduMContainerHeaderSize	User Defined	C
IpduMContainerQueueSize	1..255	C
IpduMContainerTxFirstContainedPduTrigger	True/False	C
IpduMContainerTxHandleId	0..65535	C
IpduMContainerTxSendTimeout	0..65.535	C
IpduMContainerTxTriggerMode	User Defined	C
IpduMContainerTxSizeThreshold	0.. 4294967295	C
IpduMUnusedAreasDefault	0..255	C
IpduMContainerTxPduRef	User Defined	C

- 1) **IpduMContainerHeaderSize:** Defines the layout of the header information (header id and length).  
+ IPDUM\_HEADERTYPE\_LONG: Header size is 64 bit, Header Id 32 bit, Dlc 32 bit  
+ IPDUM\_HEADERTYPE\_NONE: Static Container Layout  
+ IPDUM\_HEADERTYPE\_SHORT: Header size is 32 bit, Header Id 24 bit, Dlc 8 bit
- 2) **IpduMContainerQueueSize:** Defines a local queue for handling of each ContainerPdu. Defined in number of instances of this ContainerPdu.
- 3) **IpduMContainerTxFirstContainedPduTrigger:** Defines if the transmission of this IpduMContainerTxPdu shall be requested right after the first IpduMContainedTxPdu was put into it.



- 4) **IpduMContainerTxHandleId:** Handle Id used by the PduR for TxConfirmation and for TriggerTransmit of the ContainerPdu.
- 5) **IpduMContainerTxSendTimeout:** When this timeout expires the ContainerPdu is triggered for sending. The respective timer is started when the first Pdu is put into the ContainerPdu. Defined in seconds.
- 6) **IpduMContainerTxTriggerMode:** Defines whether this ContainerPdu is fetched via trigger transmit.  
+ IPDUM\_DIRECT: The IpduM sends this ContainerPdu when this ContainerPdu is triggered  
+ IPDUM\_TRIGGERTRANSMIT: This ContainerPdu is stored in the IpduM and fetched via trigger transmit.
- 7) **IpduMContainerTxSizeThreshold:** Defines the size threshold in bytes which, when exceeded, triggers the sending of the ContainerPdu although the maximum Pdu size (PduLength parameter of Pdu object) has not been reached yet.
- 8) **IpduMUnusedAreasDefault:** IpduM fills not updated areas of the Container PDU with this byte-pattern.
- 9) **IpduMContainerTxPduRef:** Reference to the Pdu which represents the container and is used for transmission.

#### 5.3.4. IpduMConfig-IpduMContainedTxPdu setting

Configuration of a sender ContainedPdu.

Parameter Name	Value	Category
IpduMContainedPduHeaderId	1.. 4294967295	C
IpduMContainedPduOffset	0.. 4294967295	C
IpduMContainedTxPduCollectionSemantics	User Defined	C
IpduMContainedTxPduConfirmation	True/False	C
IpduMContainedTxPduHandleId	0..65535	C
IpduMContainedTxPduPriority	0..255	C
IpduMContainedTxPduSendTimeout	0..65.535	C
IpduMContainedTxPduTrigger	User Defined	C
IpduMPduUpdateBitPosition	0.. 4294967295	C
IpduMContainedTxInContainerPduRef	User Defined	C
IpduMContainedTxPduRef	User Defined	C

- 1) **IpduMContainedPduHeaderId:** Header Id which is part of the ContainerPdu when this ContainedPdu is inside.
- 2) **IpduMContainedPduOffset:** Static offset (in bytes) of the ContainedPdu.
- 3) **IpduMContainedTxPduCollectionSemantics:** Defines whether this IpduMContainedTxPdu shall be collected using a last-is-best or queued semantics.  
+ IPDUM\_COLLECT\_LAST\_IS\_BEST: The IpduMContainedTxPdu data will be fetched via TriggerTransmit just before the transmission executes.  
+ IPDUM\_COLLECT\_QUEUED: The IpduMContainedTxPdu data will instantly be stored to the IpduMContainerTxPdu in the context of the Transmit API.
- 4) **IpduMContainedTxPduConfirmation:** This Parameter determines whether for this contained I-PDU a TxConfirmation shall be provided. If set to TRUE a TxConfirmation is issued. It is not used when an I-PDU is requested using the trigger transmit API. If this Parameter is omitted, the default value shall be used.
- 5) **IpduMContainedTxPduHandleId:** Handle Id of the ContainedPdu.



- 6) **IpduMContainedTxPduPriority:** Defines a priority of a ContainedTxPdu. 255 represents the lowest priority and 0 represent the highest priority.
- 7) **IpduMContainedTxPduSendTimeout:** Defines a ContainedPdu specific sender timeout which can reduce the ContainerPdu timer when this ContainedPdu is put inside the ContainerPdu. Defined in seconds.
- 8) **IpduMContainedTxPduTrigger:** Defines whether this Pdu triggers the sending of the ContainerPdu.  
+ IPDUM\_TRIGGER\_ALWAYS: This Pdu directly triggers the sending of the ContainerPdu.  
+ IPDUM\_TRIGGER\_NEVER: This Pdu does not triggers the sending of the ContainerPdu (other trigger criteria might still trigger sending of the ContainerPdu).
- 9) **IpduMPduUpdateBitPosition:** This value specifies where the PDU's Update-Bit is stored in the Container PDU (bit location of PDU's Update-Bit in the Container PDU).
- 10) **IpduMContainedTxInContainerPduRef:** Reference to the container Pdu which this contained Pdu shall be collected in.
- 11) **IpduMContainedTxPduRef:** Reference to the Pdu which represents this ContainedPdu and is used for transmission.

### 5.3.5. IpduMConfig-IpduMContainerRxPdu

Configuration of a receiver ContainerPdu which may collect several ContainedPdu.

Parameter Name	Value	Category
IpduMContainerHeaderSize	User Defined	C
IpduMContainerPduProcessing	User Defined	C
IpduMContainerQueueSize	1..255	C
IpduMContainerRxAcceptContainedPdu	User Defined	C
IpduMContainerRxHandleId	0..65535	C
IpduMContainerRxPduRef	User Defined	C

- 1) **uMContainerHeaderSize:** Defines the layout of the header information (header id and length).  
+ IPDUM\_HEADERTYPE\_LONG: Header size is 64 bit: Header Id 32 bit, Dlc 32 bit  
+ IPDUM\_HEADERTYPE\_NONE: Static Container Layout  
+ IPDUM\_HEADERTYPE\_SHORT: Header size is 32 bit, Header Id 24 bit, Dlc 8 bit
- 2) **uMContainerPduProcessing:** Defines whether the handling of this ContainerPdu shall be done in the context of the caller (IMMEDIATE) or in the next call to IpduM\_MainFunctionRx (DEFERRED).
- 3) **IpduMContainerQueueSize:** Defines a local queue for handling of each ContainerPdu. Defined in number of instances of this ContainerPdu.
- 4) **IpduMContainerRxAcceptContainedPdu:** Defines for the received IpduMContainerRxPdu whether the list of referencing IpduMContainedRxPdu (via the reference IpduMContainedPduContainerRefRx) is a closed set.  
+ IPDUM\_ACCEPT\_ALL: The IpduMContainedRxPdu which are referencing this pduMContainerRxPdu are expected inside this IpduMContainerRxPdu, but there may also occur other Pdu inside this IpduMContainerRxPdu as well. This also supports the case where no IpduMContainedRxPdu references the IpduMContainerRxPdu.  
+ IPDUM\_ACCEPT\_CONFIGURED: Only the IpduMContainedRxPdu which are referencing this IpduMContainerRxPdu are expected inside this IpduMContainerRxPdu.
- 5) **MContainerRxHandleId:** Handle Id used by the PduR for RxIndication.
- 6) **IpduMContainerRxPduRef:** Reference to the Pdu which represents the container and is used for reception.

### 5.3.6. IpduMConfig-IpduMContainedRxPdu setting

Configuration of a received contained Pdu.

Parameter Name	Value	Category
IpduMContainedPduOffset	0.. 4294967295	C
IpduMContainedRxPduLongHeaderId	1.. 4294967295	C
IpduMContainedRxPduShortHeaderId	1.. 16777215	C
IpduMPduUpdateBitPosition	0.. 4294967295	C
IpduMContainedRxInContainerPduRef	User Defined	C
IpduMContainedRxPduRef	User Defined	C

- 1) IpduMContainedPduOffset:** Static offset (in bytes) of the ContainedPdu.
- 2) IpduMContainedRxPduLongHeaderId:** LongHeader Id which is part of the ContainerPdu when this ContainedPdu is inside.
- 3) IpduMContainedRxPduShortHeaderId:** ShortHeader Id which is part of the ContainerPdu when this ContainedPdu is inside.
- 4) IpduMPduUpdateBitPosition:** This value specifies where the PDU's Update-Bit is stored in the Container PDU (bit location of PDU's Update-Bit in the Container PDU).
- 5) IpduMContainedRxInContainerPduRef:** Optional reference to an IpduMContainerRxPdu this IpduMContainedRxPdu may be received in.  
If this IpduMContainedRxPdu shall be received in exactly one IpduMContainerRxPdu with IpduMContainerRxAcceptContainedPdu=IPDUM\_ACCEPT\_CONFIGURED then the IpduMContainedRxInContainerPduRef shall be defined.  
If this IpduMContainedRxPdu can be received in any IpduMContainerRxPdu with IpduMContainerRxAcceptContainedPdu=IPDUM\_ACCEPT\_ALL then the IpduMContainedRxInContainerPduRef shall NOT be defined.
- 6) IpduMContainedRxPduRef:** Reference to the Pdu which represents this ContainedPdu and is used for reception indication.

## 6. Application Programming Interface (API)

### 6.1 Type Definitions

None

### 6.2 Macro Constants

None

### 6.3 Function

#### 6.3.1 Initialization

<b>Function Name</b>	IpduM_Init	
<b>Syntax</b>	FUNC(void, IPDUM_CODE) IpduM_Init( P2CONST(IpduM_ConfigType, AUTOMATIC, IPDUM_APPL_CONST) config)	
<b>Service ID [Hex]</b>	0x00	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Non-Reentrant	
<b>Parameters (In)</b>	config	Implementation specific structure with configuration parameters.
<b>Parameters (Inout)</b>	None	
<b>Parameters (Out)</b>	None	
<b>Return Value</b>	None	
<b>Description</b>	Initializes the I-PDU Multiplexer. This function is used by BSW.	
<b>Preconditions</b>	None	
<b>Configuration Dependency</b>	None	
<b>Available via</b>	IpduM.h	

<b>Function Name</b>	IpduM_GetVersionInfo	
<b>Syntax</b>	FUNC(void, IPDUM_CODE) IpduM_GetVersionInfo( P2VAR(Std_VersionInfoType, AUTOMATIC, IPDUM_APPL_DATA) versioninfo)	
<b>Service ID [Hex]</b>	0x01	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Non-Reentrant	
<b>Parameters (In)</b>	None	
<b>Parameters (Inout)</b>	None	
<b>Parameters (Out)</b>	versioninfo	Pointer to where to store the version information of this module.
<b>Return Value</b>	None	
<b>Description</b>	Service returns the version information of this module This function is used by user. But it needs configuration. (It cannot be called directly by user).	

<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>Available via</b>	IpduM.h

### 6.3.2 Transmission and reception

<b>Function Name</b>	IpduM_Transmit	
<b>Syntax</b>	FUNC(Std_ReturnType, IPDUM_CODE) IpduM_Transmit( PduIdType TxPduId, P2CONST(PduInfoType, AUTOMATIC, IPDUM_APPL_DATA) PduInfoPtr)	
<b>Service ID [Hex]</b>	0x02	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant for different PduIds. Non reentrant for the same PduId.	
<b>Parameters (In)</b>	TxPduId	Identifier of the PDU to be transmitted
	PduInfoPtr	Length of and pointer to the PDU data and pointer to MetaData.
<b>Parameters (Inout)</b>	None	
<b>Parameters (Out)</b>	None	
<b>Return Value</b>	Std_ReturnType	E_OK: Transmit request has been accepted. E_NOT_OK: Transmit request has not been accepted.
<b>Description</b>	Requests transmission of a PDU. This function is used by BSW.	
<b>Preconditions</b>	None	
<b>Configuration Dependency</b>	None	
<b>Available via</b>	IpduM.h	

<b>Function Name</b>	IpduM_RxIndication	
<b>Syntax</b>	FUNC(void, IPDUM_CODE) IpduM_RxIndication( PduIdType RxPduId, P2CONST(PduInfoType, AUTOMATIC, IPDUM_APPL_DATA) PduInfoPtr)	
<b>Service ID [Hex]</b>	0x42	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant for different PduIds. Non reentrant for the same PduId.	
<b>Parameters (In)</b>	RxPduId	ID of the received PDU.
	PduInfoPtr	Contains the length (SduLength) of the received PDU, a pointer to a buffer (SduDataPtr) containing the PDU, and the MetaData related to this PDU.
<b>Parameters (Inout)</b>	None	
<b>Parameters (Out)</b>	None	
<b>Return Value</b>	None	
<b>Description</b>	Indication of a received PDU from a lower layer communication interface module. This function is used by BSW.	

<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>Available via</b>	IpduM.h

<b>Function Name</b>	IpduM_TxConfirmation	
<b>Syntax</b>	FUNC(void, IPDUM_CODE) IpduM_TxConfirmation( PduldType TxPduld, Std_ReturnType result)	
<b>Service ID [Hex]</b>	0x40	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant for different Pdulds. Non reentrant for the same Pduld.	
<b>Parameters (In)</b>	TxPduld	ID of the PDU that has been transmitted.
	result	E_OK: The PDU was transmitted. E_NOT_OK: Transmission of the PDU failed.
<b>Parameters (Inout)</b>	None	
<b>Parameters (Out)</b>	None	
<b>Return Value</b>	None	
<b>Description</b>	The lower layer communication interface module confirms the transmission of a PDU, or the failure to transmit a PDU. This function is used by BSW.	
<b>Preconditions</b>	None	
<b>Configuration Dependency</b>	None	
<b>Available via</b>	IpduM.h	

<b>Function Name</b>	IpduM_TriggerTransmit	
<b>Syntax</b>	FUNC(Std_ReturnType, IPDUM_CODE) IpduM_TriggerTransmit( PduldType TxPduld, P2VAR(PdulInfoType, AUTOMATIC, IPDUM_APPL_DATA) PdulInfoPtr)	
<b>Service ID [Hex]</b>	0x41	
<b>Sync/Async</b>	Synchronous	
<b>Reentrancy</b>	Reentrant for different Pdulds. Non reentrant for the same Pduld.	
<b>Parameters (In)</b>	TxPduld	ID of the SDU that is requested to be transmitted. ID of the PDU that has been transmitted.
	PdulInfoPtr	Contains a pointer to a buffer (SduDataPtr) to where the SDU data shall be copied, and the available buffer size in SduLength. On return, the service will indicate the length of the copied SDU data in SduLength.
<b>Parameters (Out)</b>	None	
<b>Return Value</b>	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. PdulInfoPtr must not be used since it may contain a NULL pointer or point to invalid data.

<b>Description</b>	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. This function is used by BSW.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>Available via</b>	IpduM.h

### 6.3.3 Scheduled function

<b>Function Name</b>	IpduM_MainFunctionTx
<b>Syntax</b>	FUNC(Std_ReturnType, IPDUM_CODE) IpduM_TriggerTransmit( PduIdType TxPduId, P2VAR(PduInfoType, AUTOMATIC, IPDUM_APPL_DATA) PduInfoPtr)
<b>Service ID [Hex]</b>	0x12
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None
<b>Description</b>	This function performs the processing of the transmission activities that are not directly handled within the calls from PduR. This function is used by BSW.
<b>Preconditions</b>	None
<b>Configuration Dependency</b>	None
<b>Available via</b>	IpduM.h

<b>Function Name</b>	IpduM_MainFunctionRx
<b>Syntax</b>	FUNC(Std_ReturnType, IPDUM_CODE) IpduM_TriggerTransmit( PduIdType TxPduId, P2VAR(PduInfoType, AUTOMATIC, IPDUM_APPL_DATA) PduInfoPtr)
<b>Service ID [Hex]</b>	0x11
<b>Sync/Async</b>	Synchronous
<b>Reentrancy</b>	Non Reentrant
<b>Parameters (In)</b>	None
<b>Parameters (Inout)</b>	None
<b>Parameters (Out)</b>	None
<b>Return Value</b>	None

<b>Description</b>	This function performs the processing of the reception activities that are not directly handled within the calls from PduR. This function is used by BSW.
<b>Preconditions</b>	None
<b>Configuration</b>	None
<b>Dependency</b>	
<b>Available via</b>	IpduM.h

## 7. Generator

### 7.1 Generator Option

Options	Description
-G,--Generation	Symbolic parameters to be used for fore generation (skip validation).
-H,--Help	Display this help message.
-I,--Input <I>	ECU description file path of the module for which generation tool need to run.
-L,--Log	Symbolic parameters to be used for generation error log.
-M,--Module <M>	Specify module name and version to be generated code for.
-O,--Output <O>	Project-relative path to location where the generated code is to be placed.
-T,--Top_path <T>	Symbolic parameters to be used for set path of module.
-V,--Validate	Symbolic parameters to be used for invoking validation checks.

### 7.1 Generator Message

#### 7.1.1 Error Messages

**ERR0520001:** 'MandatoryModule' Component is not present in the input file(s).

**ERR0520002:** 'COM' Component is not present in the input file(s) when parameter 'IpduMRxDirectComInvocation' is configured as <true/1>.

**ERR0520003:** 'The Module Description of IpduM Ecuc Module Configuration Values should be configured.'

**ERR0520004:** The value configured for the parameter 'Parameter Name' in the container 'Container Name' should follow the pattern: <Pattern>

**ERR0520005:** At least one 'IpduMTxPathway' or 'IpduMRxPathway' or 'IpduMContainerTxPdu' or 'IpduMContainerRxPdu' container should be configured.

**ERR0520006:** Parameter 'IpduMRxTimeBase' in container 'IpduMGeneral' should be configured greater than <0>."

**ERR0520007:** Parameter 'IpduMTxTimeBase' in container 'IpduMGeneral' should be configured greater than <0>."

**ERR0520008:** Parameter 'IpduMStaticPartExists' should be configured as <true/1>, since IpduMTxStaticPart/ or IpduMRxStaticPart is configured in the IpduM module.

**ERR0520009:** Parameter 'IpduMTxConfirmationPduld' in container 'IpduMTxPathway' having short name <shortname of IpduMTxPathway> should be configured when parameter 'IpduMTxTriggerMode' is configured as NONE.

**ERR0520010:** Parameter 'IpduMTxTriggerMode' of TxPathawy having short name <shotname of IpduMTxPathway> should not be configured as STATIC\_PART\_TRIGGER since there is no Static Part configured in this TxPathway.

**ERR0520011:** Handle Id used by the PduR for TxConfirmation and for TriggerTransmit of the ContainerPdu and multiplexed I-PDU should start with <0> and should be unique and sequential.



ERR0520012: Handle Id used by the PduR for Tx Request of the ContainedPdu and Static/Dynamic part of multiplexed I-PDU should start with <0> and should be unique and sequential.

ERR0520013: Handle Id used by the PduR for Rx Indication of the ContainerPdu and multiplexed I-PDU should start with <0> and should be unique and sequential.

ERR0520014: Parameter 'IpduMInitialDynamicPart' in TxPathway having short name <shortname of IpduMTxPathway> should be configured as reference path to one Dynamic Part in same TxPathway.

ERR0520015: The Selector Field in TxPathway or RxPathway having short name <shortname of IpduMTxPathway or IpduMRxPathway> should be configured how it is included in one dynamic segment.

ERR0520016: Selector Value of Dynamic Part in RxPathway having short name <shortname of IpduMRxPathway> should be unique and not greater than <2^NoOfBit>.

ERR0520017: The Dynamic and Staic Segments in TxPathway or RxPathway having short name <shortname of IpduMTxPathway or IpduMRxPathway> should not be overlap.

ERR0520018: The calculated PDU length from configured segments in TxPathway or RxPathway having short name <shortname of IpduMTxPathway or IpduMRxPathway> is <Calculated length> greater than the maximum PDU length <MaxLength> configured in EcuC.

ERR0520019: Value of reference parameter 'Parameter Name' having short name <shortname of Pdu in EcuC> in 'Container Name' does not have an equivalent I-PDU in the < Module Name> module.

ERR0520020: Value of reference parameter 'IpduMTxStaticPduRef/IpduMTxDynamicPduRef' in the I-PDU IpduMTxStaticPart/IpduMTxDynamicPart does not have an equivalent I-PDU in the Com module.

ERR0520021: Parameter 'ComIPduHandleId' in the container 'ComIPdu' should be configured, since it is referred by the reference path configured in parameter 'Parameter Name' in the container 'Container Name'.

ERR0520022: Parameter 'PduRDestPduHandleId' in container 'PduRDestPdu' should be configured, since it is referred by the reference path configured in parameter 'Parameter Name' in the container 'Container Name'.

ERR0520023: <IpduMContainerTxPdu Short Name> contains <IpduMContainedTxPdu Short Name> and <IpduMContainedTxPdu Short Name> with different IpduMContainedTxPduCollectionSemantics.

ERR0520024: IpduM shall reject configurations in which contained I-PDU supporting MetaData have a different MetaDataType from the MetaDataType of the Container PDU. <IpduMContainedTxPdu Short Name> supporting MetaData have a different MetaDataType from the MetaDataType of the <IpduMContainerTxPdu Short Name>."

ERR0520025: For a Container PDU with IpduMContainerHeaderSize set to IPDUM\_HEADERTYPE\_NONE, all contained I-PDUs shall have IpduMContainedTxPduCollectionSemantics set to IPDUM\_COLLECT\_LAST\_IS\_BEST.

<IpduMContainerTxPdu Short Name> with IpduMContainerHeaderSize is IPDUM\_HEADERTYPE\_NONE, but <IpduMContainedTxPdu Short Name> with IpduMContainedTxPduCollectionSemantics is not IPDUM\_COLLECT\_LAST\_IS\_BEST."

**ERR0520026:** For a Container PDU with IpduMContainerHeaderSize set to IPDUM\_HEADERTYPE\_NONE, all contained I-PDUs shall have a configured IpduMContainedTxPduOffset.

<IpduMContainerTxPdu Short Name> with IpduMContainerHeaderSize set to IPDUM\_HEADERTYPE\_NONE, but <IpduMContainedTxPdu Short Name> with IpduMContainedTxPduOffset is not configured."

**ERR0520027:** For a Container PDU with IpduMContainerHeaderSize set to IPDUM\_HEADERTYPE\_NONE and IpduMUnusedAreasDefault not set, all contained I-PDUs shall have a configured IpduMPduUpdateBitPosition.

<IpduMContainerTxPdu Short Name> with IpduMContainerHeaderSize set to IPDUM\_HEADERTYPE\_NONE and IpduMUnusedAreasDefault not set, but <IpduMContainedTxPdu Short Name> with IpduMPduUpdateBitPosition is not configured."

**ERR0520030:** IpduMPduUpdateBitPosition of IpduMContainedRxPdu../IpduMContainedTxPdu... only have range from 0 to (length of pdu)\*8 - 1

**ERR0520032:** IpduMContainedRxPdu.. and IpduMContainedRxPdu.. have the same IpduMContainedRxPdu-LongHeaderId/IpduMContainedRxPduShortHeaderId

**ERR0520033:** IpduMContainedRxPdu.. and IpduMContainedRxPdu.. have the same IpduMContainedRxPdu-LongHeaderId/IpduMContainedRxPduShortHeaderId

**ERR0520034:** When IpduMContainerPduProcessing is IPDUM\_PROCESSING\_DEFERRED, IpduMContainerQueueSize should be configured.

**ERR0520035:** Static Contained Pdu should be configured IpduMContainedPduOffset.

**ERR0520036:** Wrong IpduMContainedPduOffset value. IpduMContainedPduOffset should be greater than IpduMContainedPduOffset + PduLength of previous containedRxPdu.

**ERR0520037:** This ContainedRxPdu should has a IpduMContainedRxPduLongHeaderId because it refer to an ContainerRxPdu, which has IpduMContainerHeaderSize IPDUM\_HEADERTYPE\_LONG.

**ERR0520038:** The IpduMContainedRxPduLongHeaderId must be unique.

**ERR0520039:** This ContainedRxPdu should has a IpduMContainedRxPduShortHeaderId because it refer to an ContainerRxPdu, which has IpduMContainerHeaderSize IPDUM\_HEADERTYPE\_SHORT.

**ERR0520040:** The IpduMContainedRxPduShortHeaderId must be unique.

**ERR0520042:** The IpduMContainedRxInContainerPduRef should refer to a ContainerRxPdu, which have IpduMContainerRxAcceptContainedPdu is IPDUM\_ACCEPT\_CONFIGURED.

**ERR0520047:** The IpduMContainedRxInContainerPduRef should refer to a ContainerRxPdu, which have IpduMContainerRxAcceptContainedPdu is IPDUM\_ACCEPT\_CONFIGURED.

**ERR0520050:** A IpduMContainerTxPdu must have at least 1 IpduMContainedTxPdu.

<IpduMContainerTxPdu Short Name> must have at least 1 IpduMContainedTxPdu."

ERR0520060: The IpduM module supported Post-Build but there are no variants configured in EcuC.

ERR0520065: Mismatch post-build variant with EcuC module, the module's post-build variants should include all EcuC post-build variants.

## 7.1.2 Warning Messages

None

## 7.1.3 Information Messages

INF0520001: Parameter 'IpduMStaticPartExists' in container 'IpduMGeneral' should be configured as <false/0> when there is no configuration of 'IpduMTxStaticPart' or 'IpduMRxStaticPart'. Hence, the Generator shall reset the value of parameter 'IpduMStaticPartExists' to <false/0>.

INF0520002: Parameter 'IpduMJitUpdate' in containers 'IpduMTxDynamicPart' and 'IpduMTxStaticPart' should not configured as <true/1> when parameter 'IpduMTxTriggerMode' of TxPathway having short name <shortname of IpduMTxPathway> is configured as STATIC\_OR\_DYNAMIC\_PART\_TRIGGER. Hence the Generator shall reset the value of 'IpduMJitUpdate' to <false/0>.

INF0520003: Parameter 'IpduMTxDynamicConfirmation' in containers 'IpduMTxDynamicPart' and parameter 'IpduMTxStaticConfirmation' in container 'IpduMTxStaticPart' should configured as <false/0> when parameter 'IpduMTxConfirmationPduld' in TxPathway having short name <shortname of IpduMTxPathway> is not configured. Hence the Generator shall reset the value of 'IpduMTxDynamicConfirmation' or 'IpduMTxStaticConfirmation' to <false/0>.

INF0520004: Since the segments are configured in continuous manner in TxPathway or RxPathway having short name <shortname of IpduMTxPathway or IpduMRxPathway>, the Generator shall collect them to one segment having position is <minimum position of IpduMSegmentPosition> and length is <total length of IpduMSegmentLength>.

8. SWP Error Code

None

9. Appendix

None