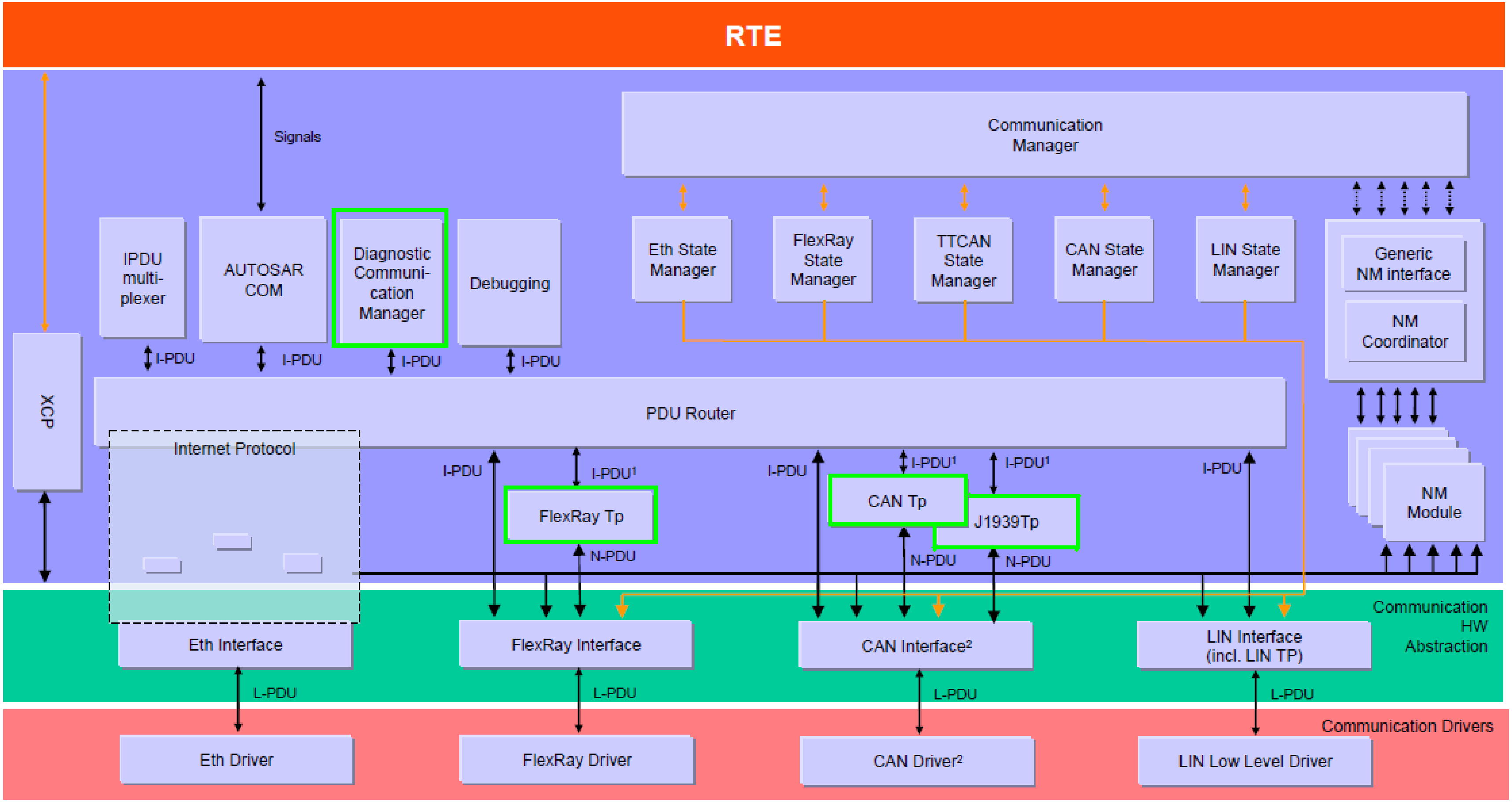
CanTp Training
CDG-SMT

- The Solutions Network -

Agenda

- CLASSIC CAN
  - Introduction to ISO 15765 – Part2
  - ISO 15765 “Transport Protocol” Features
  - Introduction to AUTOSAR CanTp
- CANFD
  - Additional AUTOSAR CanTp Features with CanFD
- AEEE-Pro
  - CanTp Configuration Example

Autosar Communication Stack



Communication Stack / OSI Layers / ISO Specs

<table>
<tr><th>Applicability</th><th>OSI 7 layer</th><th colspan="5">Enhanced diagnostics services</th><th>WWH-OBD</th></tr>
<tr><td rowspan="7">Seven layer according to ISO/IEC 7498-1 and ISO/IEC 10731</td><td>Application (layer 7)</td><td colspan="5">ISO 14229-1, ISO 14229-3 UDSonCAN, ISO 14229-4 UDSonFR, ISO 14229-5 UDSonIP, ISO 14229-6 UDSonK-Line, further standards</td><td>ISO 27145-3</td></tr>
<tr><td>Presentation (layer 6)</td><td colspan="5">vehicle manufacturer specific</td><td>ISO 27145-2</td></tr>
<tr><td>Session (layer 5)</td><td colspan="6">ISO 14229-2</td></tr>
<tr><td>Transport (layer 4)</td><td rowspan="2">ISO 15765-2</td><td rowspan="2">ISO 10681-2</td><td rowspan="2">ISO 13400-2</td><td rowspan="2">Not applicable</td><td>further standards</td><td rowspan="4">ISO 27145-4</td></tr>
<tr><td>Network (layer 3)</td><td>further standards</td></tr>
<tr><td>Data link (layer 2)</td><td rowspan="2">ISO 11898-1, ISO 11898-2</td><td rowspan="2">ISO 17458</td><td rowspan="2">ISO 13400-3, IEEE 802.3</td><td>ISO 14230-2</td><td>further standards</td></tr>
<tr><td>Physical (layer 1)</td><td>ISO 14230-1</td><td>further standards</td></tr>
</table>

ISO 15765 – Part 2 (Network Layer Services)

- ISO 15765 – Part 2 specifies an unconfirmed communication protocol tailored to meet the requirements of CAN based diagnostic systems.

- ISO/CD 15765 – Part 2 is providing the following services:
  - Segmentation of data, which do not fit into a single CAN frame, in transmit direction
  - Reassembling of data in receive direction
  - Control of data flow
  - Detection of errors in segmentation sessions
  - Flexible Data rate support

BOSCH

Service Primitives defined by ISO 15765-2

N_USData.request (Address Information, MessageData, Length)

N_USData.confirm (Address Information, N_Result)

N_USData_FF.indication (Address Information, Length)

N_USData.indication (Address Information, MessageData, Length, N_Result)

N_ChangeParameter.request (Address Information, Parameter, Parameter_Value)

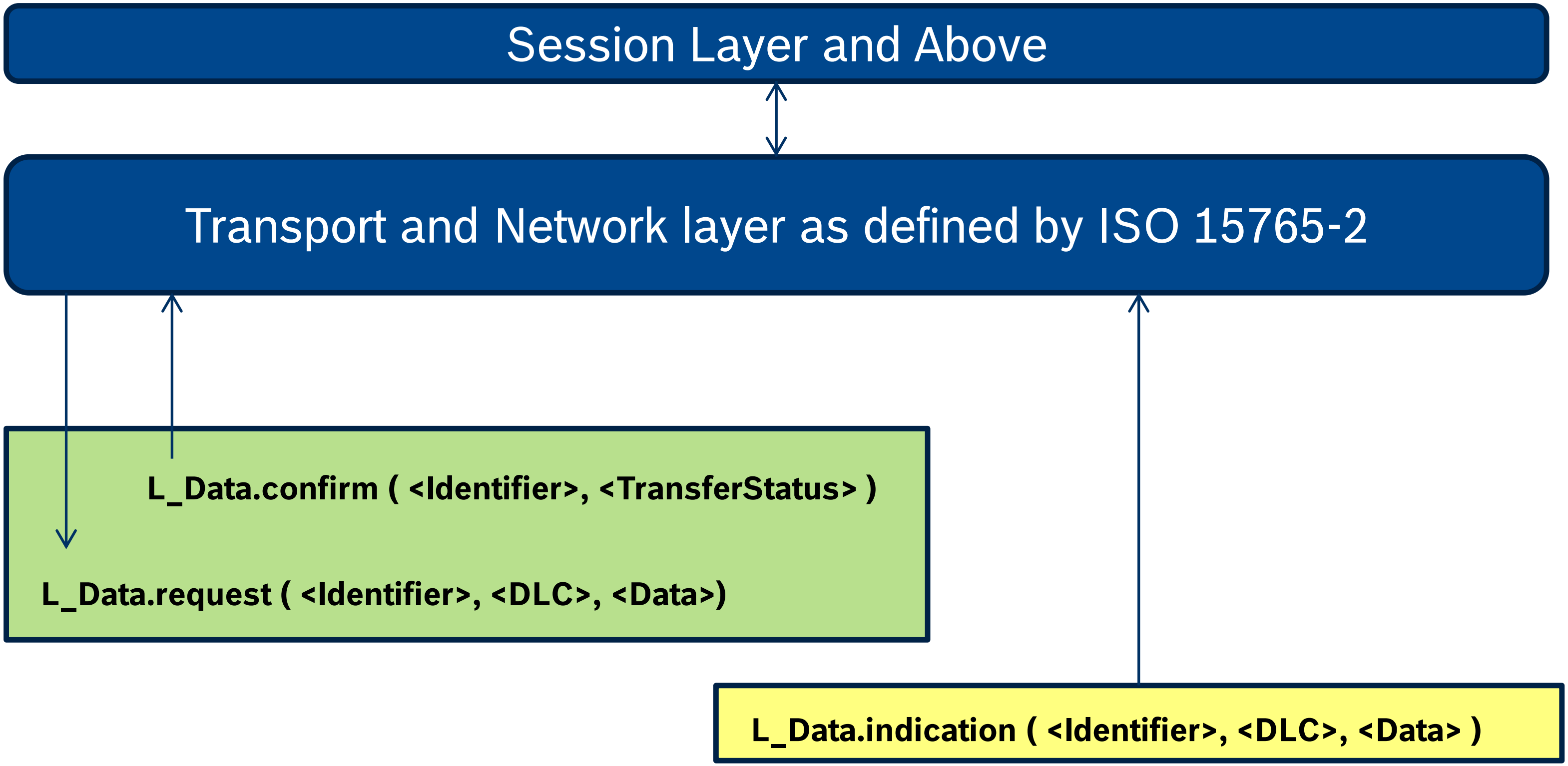
N_ChangeParameter.confirm (Address Information, Parameter, Result_ChangeParameter)

Transport and Network layer as defined by ISO 15765-2

Data Link Layer and Physical Layer

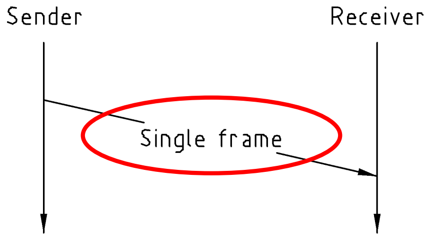
BOSCH

Service Primitives defined by ISO 15765-2

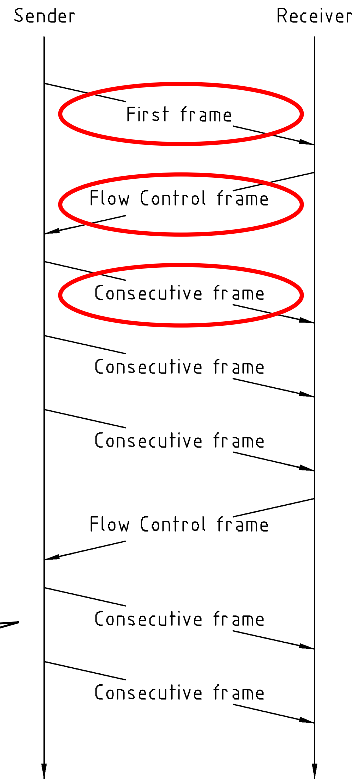


ISO 15765 – Message Transmission

Un-Segmented message transmission



Segmented message transmission



Unsegmented Transmission

- Single Frame (SF)

Segmented Transmission

- Sender starts with First Frame (FF)
- Receiver confirms with Flow Control (FC)
- Sender sends remaining transmission data with Consecutive Frames (CF)

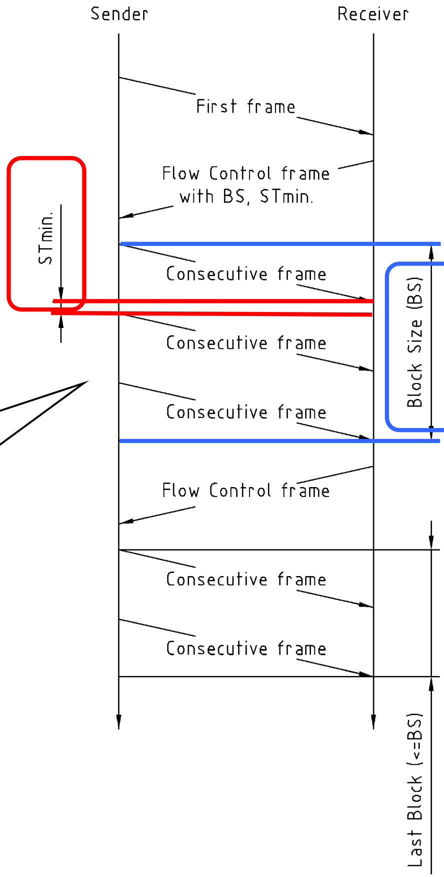
ISO 15765 – STmin, BS

Flow Control Parameter

Minimum Separation Time (STmin)
Minimum Time the sender shall wait between 2 CFs

Block Size (BS)
Maximum number of CFs the receiver allows the sender to send in one block, before waiting for the next CF.

BS = 0 (infinite)
All Consecutive Frames (CF) will be send without any further Flow Control frames



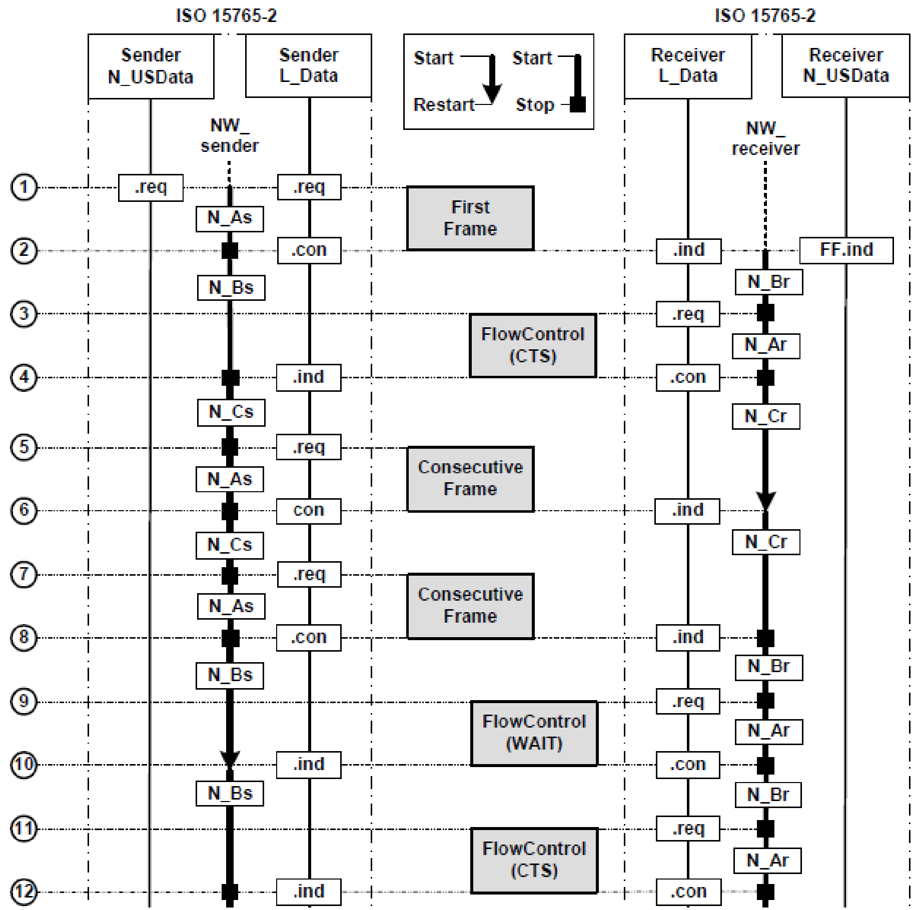
ISO 15765 – Flow Control Status

Flow Status

FC.CTS
Request to continue sending the next block of CFs

FC.WAIT
request to wait

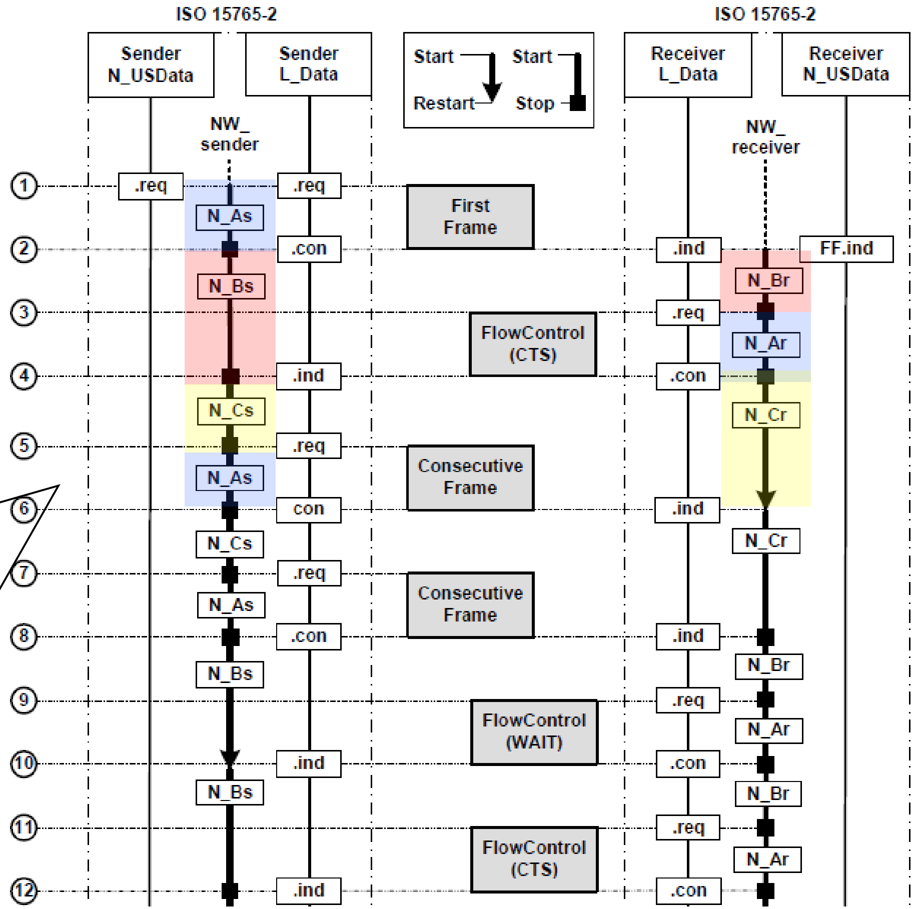
FC.OVFLW
Receive buffer overflow.



ISO 15765 – Timing-Parameter

Timing Parameter

- **As/Ar**
  Time for transmission of the CAN frame
- **Bs**
  Time until reception of the next Flow Control Frame
- **Br**
  Time until transmission of the next Flow Control Frame
- **Cs**
  Time until transmission of the next Consecutive Frame
- **Cr**
  Time until reception of the next Consecutive Frame



ISO 15765 – CLASSIC CAN - Normal Addressing

Frame Type	CAN Identifier	CAN Frame Data field				
		Byte 1		Byte 2	Byte 3	Byte 4-8
		Bit 7- 4	**Bit 3 – 0**			
Single Frame (SF)	N_AI	SF = 0x00	SF_DL	Data		
First Frame (FF)	N_AI	FF = 0x01	FF_DL		Data	
Consecutive Frame (CF)	N_AI	CF = 0x02	SN	Data		
Flow Control (FC)	N_AI	FC = 0x03	FS	BS	STmin	N/A

Flow Status
0x00: Clear to Send
0x01: Wait
0x02: Overflow

Legend

SF_DL:	*4 Bit Single Frame Data Length*	*FF_DL:*	*12 Bit First Frame Data Length*
SN:	*Sequence Number*	*FS:*	*Flow Status*
BS:	*Block Size*	*STmin:*	*Minimum Separation Time*

PCI - Protocol Control Information

ISO 15765 – CLASSIC CAN - Extended Addressing

Frame Type	CAN Identifier	CAN Frame Data field					
		Byte 1	Byte 2		Byte 3	Byte 4	Byte 5-8
			Bit 7- 4	**Bit 3 – 0**			
Single Frame (SF)	N_AI, except N_TA	N_TA	SF = 0x00	SF_DL	Data		
First Frame (FF)	N_AI, except N_TA	N_TA	FF = 0x01	FF_DL		Data	
Consecutive Frame (CF)	N_AI, except N_TA	N_TA	CF = 0x02	SN	Data		
Flow Control (FC)	N_AI, except N_TA	N_TA	FC = 0x03	FS	BS	STmin	N/A

Flow Status
0x00: Clear to Send
0x01: Wait
0x02: Overflow

Legend

SF_DL:	*4 Bit Single Frame Data Length*	*FF_DL:*	*12 Bit First Frame Data Length*
SN:	*Sequence Number*	*FS:*	*Flow Status*
BS:	*Block Size*	*STmin:*	*Minimum Separation Time*

TA - Target Address

PCI - Protocol Control Information

ISO 15765 – CLASSIC CAN - Mixed Addressing

Frame Type	CAN Identifier	CAN Frame Data field					
		Byte 1	Byte 2		Byte 3	Byte 4	Byte 5-8
			Bit 7- 4	**Bit 3 – 0**			
Single Frame (SF)	N_AI	N_AE	SF = 0x00	SF_DL	Data		
First Frame (FF)	N_AI	N_AE	FF = 0x01	FF_DL		Data	
Consecutive Frame (CF)	N_AI	N_AE	CF = 0x02	SN	Data		
Flow Control (FC)	N_AI	N_AE	FC = 0x03	FS	BS	STmin	N/A

Flow Status
0x00: Clear to Send
0x01: Wait
0x02: Overflow

Legend

SF_DL:	*4 Bit Single Frame Data Length*	*FF_DL:*	*12 Bit First Frame Data Length*
SN:	*Sequence Number*	*FS:*	*Flow Status*
BS:	*Block Size*	*STmin:*	*Minimum Separation Time*

TA - Target Address

PCI - Protocol Control Information

BOSCH

ISO 15765 – Target Address Type

Target Address Type

The parameter Target Address Type shall be used to encode the communication model used by the communicating entities.

Communication Models

Two communication models are specified:

- Physical addressing (1 to 1 communication)
  shall be supported for all types of network layer messages.
- Functional addressing (1 to n communication, broadcast)
  shall only be supported for Single Frame communication.

BOSCH

ISO 15765 – CLASSIC CAN - Padding

CAN Frame Data Padding

- The DLC (data Length Code) is always set to 8. If the CAN Frame to be transmitted is shorter than 8 bytes, then the sender has to set the DLC to the maximum value 8 (padding of unused data bytes).

- In particular this can be the case for a Single Frame, a Flow Control Frame or the last Consecutive Frame of a segmented message.

ISO 15765 – Half/Full - Duplex

Network layer design decision to support full- or half-duplex communication:

- Half-duplex:
  Point-to-point communication between two nodes is only possible in one direction at a time.
- Full-duplex:
  Point-to-point communication between two nodes is possible in both directions at once.

This design decision has a side-effect on the interpretation of “unexpected” frames!

Trace – Extended Addressing Request

Tester Address
FF

ECU Address
11

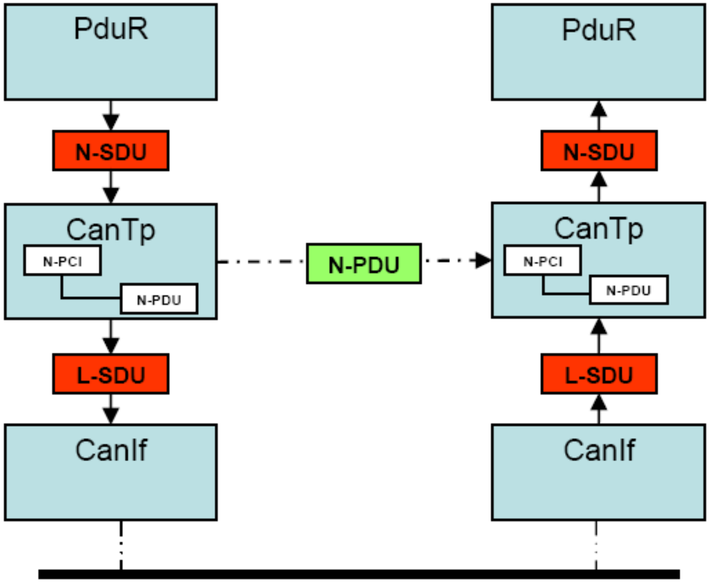
Tester Request
22 10 00

ECU Response
62 10 00 03 04 05
06 07 08 09 0A 0B
0C 0D 0E 0F 10
11 12 13

AUTOSAR CanTp - PDUs and SDUs

PDU (Protocol Data Unit)
In layered systems, it refers to a data unit that is specified in the protocol of a given layer. This contains user data of that layer (SDU) plus possible protocol control information (PCI). Furthermore, the PDU of layer X is the SDU of its lower layer X-1 (i.e. (X)-PDU = (X-1)-SDU).

SDU (Service Data Unit)
In layered systems, this refers to a set of data that is sent by a user of the services of a given layer, and is transmitted to a peer service user.



AUTOSAR CanTp - Interfaces to/with CAN-IF

Std_ReturnType **CanIf_Transmit** (PduIdType CanTxPduId, const PduInfoType* PduInfoPtr)

➔ This service initiates a request for transmission of the CAN L-PDU specified by the CanTxPduId and CAN related data in the L-PDU structure.

void **CanTp_RxIndication** (PduIdType RxPduId, PduInfoType* PduInfoPtr)

➔ This function is called by the CAN Interface after a successful reception of a Rx CAN L-PDU which is configured in CanIf/Can for CanTp.

void **CanTp_TxConfirmation** (PduIdType CanTpTxPduId)

➔ The main function for scheduling the CAN TP (Entry point for scheduling).

AUTOSAR CanTp - Interfaces to/with PduR (Rx)

BufReq_ReturnType **PduR_CanTpStartOfReception**
(PduIdType id, PduLengthType TpSduLength, PduLengthType* bufferSizePtr)

➔ This function is called by the CANTP to indicate PduR about a start of new reception (i.e. SF or FF is received in CanTp)

BufReq_ReturnType **PduR_CanTpCopyRxData**
(PduIdType id, PduInfoType* info, PduLengthType* bufferSizePtr)

➔ This function is called when a transport protocol module has data to copy for the receiving module.

void **PduR_CanTpRxIndication**
(PduIdType CanTpRxPduId, NotifResultType Result)

➔ This function is called by the CAN TP after successful/unsuccessful reception of data.

BOSCH

AUTOSAR CanTp - Interfaces to/with PduR (Tx)

Std_ReturnType **CanTp_Transmit**
(PduIdType CanTpTxSduId,const PduInfoType* CanTpTxInfoPtr)

- This service is used to request the transfer of (un)segmented data.

BufReq_ReturnType **PduR_CanTpCopyTxData**
(PduIdType id, PduInfoType* info, RetryInfoType* retry, PduLengthType* availableDataPtr)

- This function is called by the transport protocol module to query the transmit data of an I-PDU segment. PduR shall need to copy the required data to transmit (SF, FF or CF) accordingly to CanTp Buffer which is provided along with this call-back.

void **PduR_CanTpTxConfirmation** (PduIdType CanTpTxPduId, NotifResultType Result)

- This function is called by the CAN Transport Protocol after successful/unsuccesful transmision of a transmit request

AUTOSAR CanTp - Interfaces to Scheduler

void **CanTp_Init** (const CanTp_ConfigType* CfgPtr)

➔ This API is responsible for initialisation of CanTp

void **CanTp_Shutdown** (void)

➔ This API is responsible to close all the active Channels/Connections of CanTp and move CanTp to shutdown state.

void **CanTp_MainFunction** (void)

➔ This API is the main function which has to be called from a fixed cyclic process by the OS Scheduler.

void **CanTp_GetVersionInfo** (Std_VersionInfoType* versioninfo)

➔ This API can be used by any component to check or validate the version compatibility of CanTp with the other dependant components.

AUTOSAR CanTp - Br / Cs - Timeout

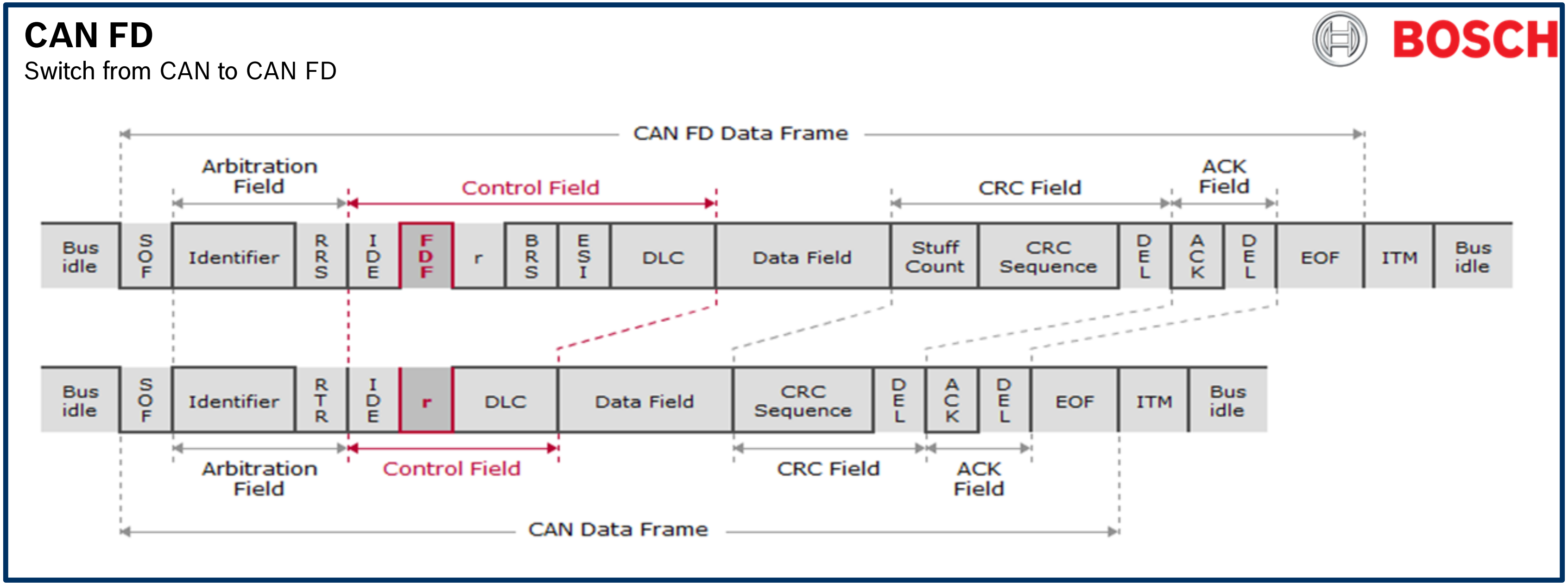
CanTp166:
At the reception of a FF, last CF of a block or a SF, the CanTp module shall start a time-out N_Br before requesting an Rx buffer

CanTp167:
After a transmission request from upper layer, the CanTp module shall start time-out N_Cs before the call of PduR_CanTpCopyTxData. If a buffer is not available before the timer elapsed, the CanTp module shall abort the communication.

N_Br and N_Cs are defined as "performance parameters in ISO15765" but AUTOSAR CanTp is "redefining" N_Br and N_Cs as timeout parameter for buffer handling with PduR !

BOSCH

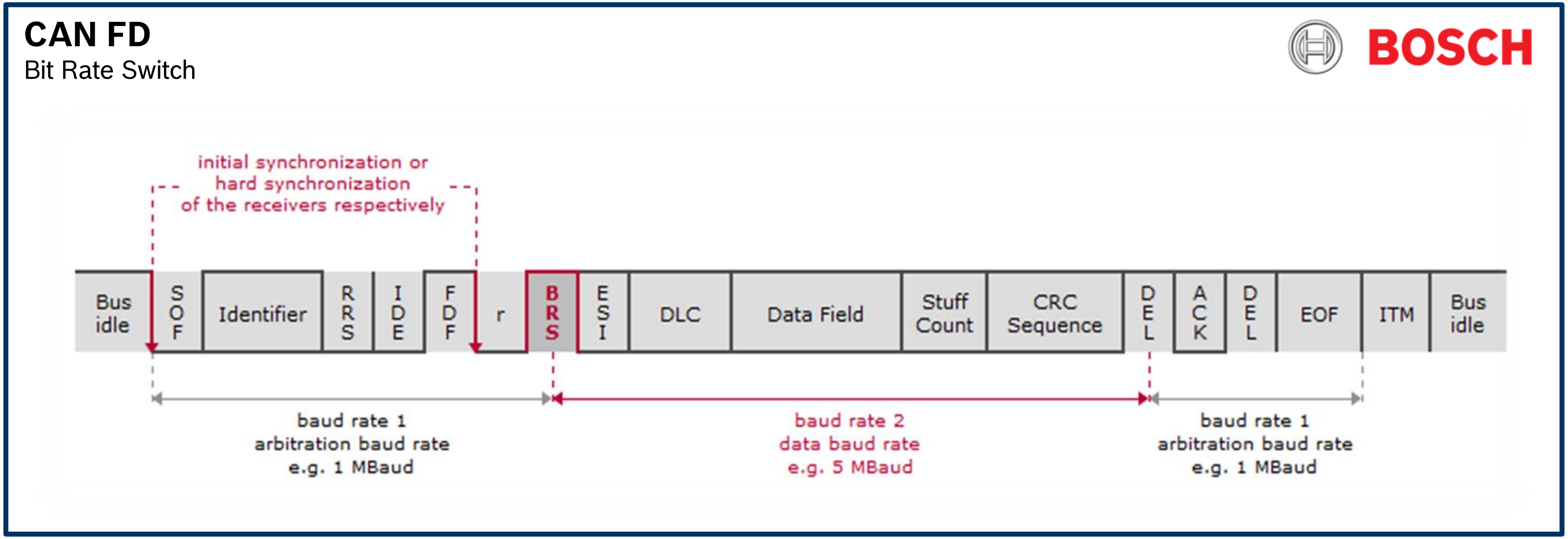
Switch from Can To CanFD [Flexible Data Rate]



The reserve bit of the traditional CAN frame now becomes the switch for the CAN FD format. If transferred dominant as 0, it denotes a classical CAN frame. If it contains the recessive value 1, it denotes a CAN FD frame.

The bit's new name is **FDF** for Flexible Data Rate Format which establishes the opportunity to transmit a much larger payload. The actual length of the data field and the decision whether to switch to a higher transmission rate or not, follow later in the frame.

Can with Flexible Data Rate in Data Field



After a reserve bit **r0** for future extensions to CAN FD a bit called Bit Rate Switch (**BRS**) follows. When **BRS** is dominant, baud rate 2 is equal to baud rate 1, so no accelerated transmission will occur.
If **BRS** is recessive, the part of the CAN FD frame marked red in the graphic, will be transmitted at the higher baud rate 2. For all CAN FD controllers on one bus both transmission rates have to be configured uniformly.

For more information regarding Flexible data rate feature in Can Controller please refer below training slides.
http://www.bosch-semiconductors.de/media/pdf_1/canliteratur/can_fd_spec.pdf

CanTp with Flexible Data Rate Support

- **Up to 64 Bytes payload**
  A CAN FD (Flexible Data Rate) protocol device can transmit/receive frames with payload sizes from 1 to 64 bytes. A CAN FD protocol device is also capable of transmitting/receiving CLASSIC CAN frames.

- **Up to 4GB Bytes messages**
  Using CanFD the maximum segmented message length supported is equal to 4294967295(4GB) bytes of user data, but currently CUBAS CANTP is supporting only 64K bytes of user data.

- **Mandatory Padding**
  For N_PDU length values up to 8 bytes either the data padding or the DLC data optimization are applicable. To prevent the transmission of uninitialized data the padding of CAN frame data is mandatory for DLC values greater than eight ,when the length of the N_PDU size to be transmitted is not equal to one of the discrete length values defined in the ISO 11898-1:2014 DLC table.

- **Backwards Compatibility**
  An Optional parameter **CanTpFlexibleDataRateSupport** is provided to support this feature in CanTp. This parameter is defined by AR4.2.x.

BOSCH

CAN 2.0 / CAN FD Data length comparison table

Table 3 — CAN 2.0 / CAN FD data length comparison table

data length code (DLC)	CAN 2.0 data length (CAN_DL)	CAN FD data length (CAN_DL)
0	0	0
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	8[a]	12
10	8[a]	16
11	8[a]	20
12	8[a]	24
13	8[a]	32
14	8[a]	48
15	8[a]	64

[a] For CAN 2.0 the DLC values 9..15 are automatically reduced to the value of 8 which leads to the maximum possible CAN_DL for CAN 2.0.

CAN 2.0 / CAN FD Frame Format comparison

N_PDU name	N_PCI bytes						
	Byte #1		Byte #2	Byte #3	Byte #4	Byte #5	Byte #6
	Bits 7 – 4	Bits 3 – 0					
SingleFrame (SF) (CAN_DL ≤ 8)	0000_2	SF_DL					
SingleFrame (SF) (CAN_DL > 8) [a]	0000_2	0000_2	SF_DL				
FirstFrame (FF) (FF_DL ≤ 4095)	0001_2	FF_DL					
FirstFrame (FF) (FF_DL > 4095) [b]	0001_2	0000_2	$0000\ 0000_2$	FF_DL			
ConsecutiveFrame (CF)	0010_2	SN					
FlowControl (FC)	0011_2	FS	BS	ST_{min}	N/A	N/A	N/A

[a] Messages with CAN_DL > 8 shall use an escape sequence where the lower nibble of Byte #1 is set to 0 (invalid length). This signifies to the network layer that the value of SF_DL is determined based on the next byte in the frame (Byte #2). As CAN_DL is defined to be greater than 8 this definition is only valid for CAN FD type frames.

[b] Messages larger than 4095 bytes shall use an escape sequence where the lower nibble of Byte#1 and all bits in Byte #2 are set to 0 (invalid length). This signifies to the network layer that the value of FF_DL is determined based on the next 32 bits in the frame (Byte #3 is the MSB and Byte #6 the LSB)

CAN 2.0 / CAN FD: SF_DL versus CAN_DL

Table 12 — Allowed SF_DL values for a given addressing scheme with optimized CAN_DL

Addressing type	CAN_DL value							
	0 .. 1	2	3	4	5	6	7	8
Normal	invalid	SF_DL = 1	SF_DL = 2	SF_DL = 3	SF_DL = 4	SF_DL = 5	SF_DL = 6	SF_DL = 7
Mixed or Extended	invalid	invalid	SF_DL = 1	SF_DL = 2	SF_DL = 3	SF_DL = 4	SF_DL = 5	SF_DL = 6

Table 13 — Allowed SF_DL values for a given CAN_DL greater than 8 and addressing scheme

Addressing type	CAN_DL value						
	12	16	20	24	32	48	64
Normal	8 ≤ SF_DL ≤ 10	11 ≤ SF_DL ≤ 14	15 ≤ SF_DL ≤ 18	19 ≤ SF_DL ≤ 22	23 ≤ SF_DL ≤ 30	31 ≤ SF_DL ≤ 46	47 ≤ SF_DL ≤ 62
Mixed or Extended	7 ≤ SF_DL ≤ 9	10 ≤ SF_DL ≤ 13	14 ≤ SF_DL ≤ 17	18 ≤ SF_DL ≤ 21	22 ≤ SF_DL ≤ 29	30 ≤ SF_DL ≤ 45	46 ≤ SF_DL ≤ 61

ISO/CD 15765-2 – Normal Addressing in CanFD

Frame Type	CAN Identifier	CAN Frame Data field					
		Byte 1		Byte 2	Byte 3	Byte 4-6	Byte 7-64
		Bit 7- 4	**Bit 3 – 0**				
Single Frame (SF)	N_AI	SF = 0x00	0x00	SF_DL	Data		
First Frame (FF)	N_AI	FF = 0x01	0x00		FF_DL		Data
Consecutive Frame (CF)	N_AI	CF = 0x02	SN	Data			
Flow Control (FC)	N_AI	FC = 0x03	FS	BS	STmin	N/A	

- Messages with CAN_DL > 8 can use an ***escape sequence*** *where the lower nibble of Byte #1 is set to 0 (invalid length). This signifies to the network layer that the value of SF_DL is determined based on the next byte in the frame (Byte #2). As CAN_DL is defined to be greater than 8 this definition is only valid for CAN FD type frames.*
- Messages larger than 4095 bytes shall use an ***escape sequence*** *where the lower nibble of Byte#1 and all bits in Byte #2 are set to 0 (invalid length). This signifies to the network layer that the value of FF_DL is determined based on the next 32 bits in the frame (Byte #3 is the MSB and Byte #6 the LSB)*

PCI - Protocol Control Information

ISO/CD 15765-2– Extended Addressing in CanFD

Frame Type	CAN Identifier	CAN Frame Data field						
		Byte 1	Byte 2		Byte 3	Byte 4	Byte 5-7	Byte 8-64
			Bit 7- 4	**Bit 3 – 0**				
Single Frame (SF)	N_AI, except N_TA	N_TA	SF = 0x00	0x00	SF_DL	Data		
First Frame (FF)	N_AI, except N_TA	N_TA	FF = 0x01	0x00		FF_DL		Data
Consecutive Frame (CF)	N_AI, except N_TA	N_TA	CF = 0x02	SN	Data			
Flow Control (FC)	N_AI, except N_TA	N_TA	FC = 0x03	FS	BS	STmin	N/A	

Flow Status
0x00: Clear to Send
0x01: Wait
0x02: Overflow

Legend

SF_DL:	*8 Bit Single Frame Data Length*	*FF_DL:*	*32 Bit First Frame Data Length*
SN:	*Sequence Number*	*FS:*	*Flow Status*
BS:	*Block Size*	*STmin:*	*Minimum Separation Time*

TA - Target Address

PCI - Protocol Control Information

BOSCH

ISO/CD 15765-2– Mixed Addressing in CanFD

Frame Type	CAN Identifier	CAN Frame Data field						
		Byte 1	Byte 2		Byte 3	Byte 4	Byte 5-7	Byte 8-64
			Bit 7- 4	**Bit 3 – 0**				
Single Frame (SF)	N_AI	N_AE	SF = 0x00	0x00	SF_DL	Data		
First Frame (FF)	N_AI	N_AE	FF = 0x01	0x00		FF_DL		Data
Consecutive Frame (CF)	N_AI	N_AE	CF = 0x02	SN	Data			
Flow Control (FC)	N_AI	N_AE	FC = 0x03	FS	BS	STmin	N/A	

Flow Status
0Clear to Send
0x01: Wait
0x02: Overflow
x00:

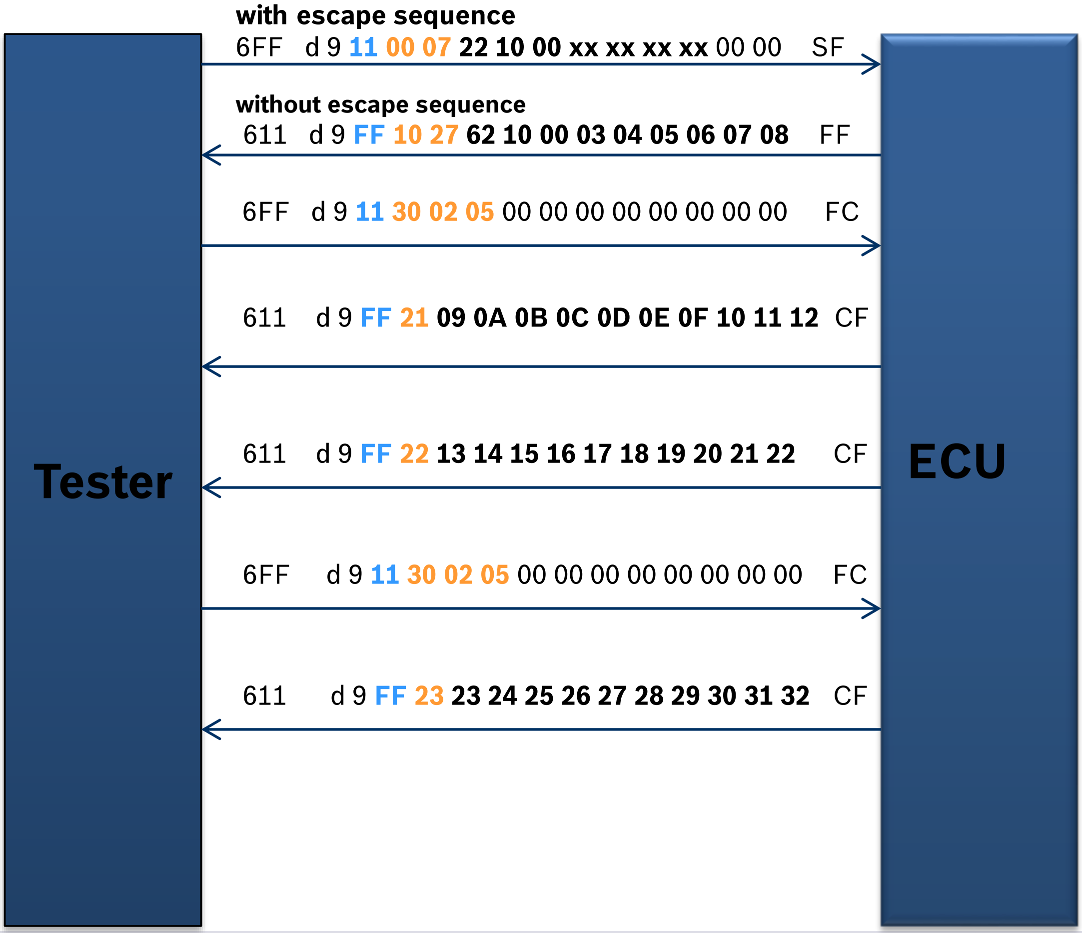
Legend

SF_DL: 8 Bit Single Frame Data Length
FF_DL: 32 Bit First Frame Data Length
SN: Sequence Number
FS: Flow Status
BS: Block Size
STmin: Minimum Separation Time

TA - Target Address

PCI - Protocol Control Information

Extended Addressing Request for CanFD

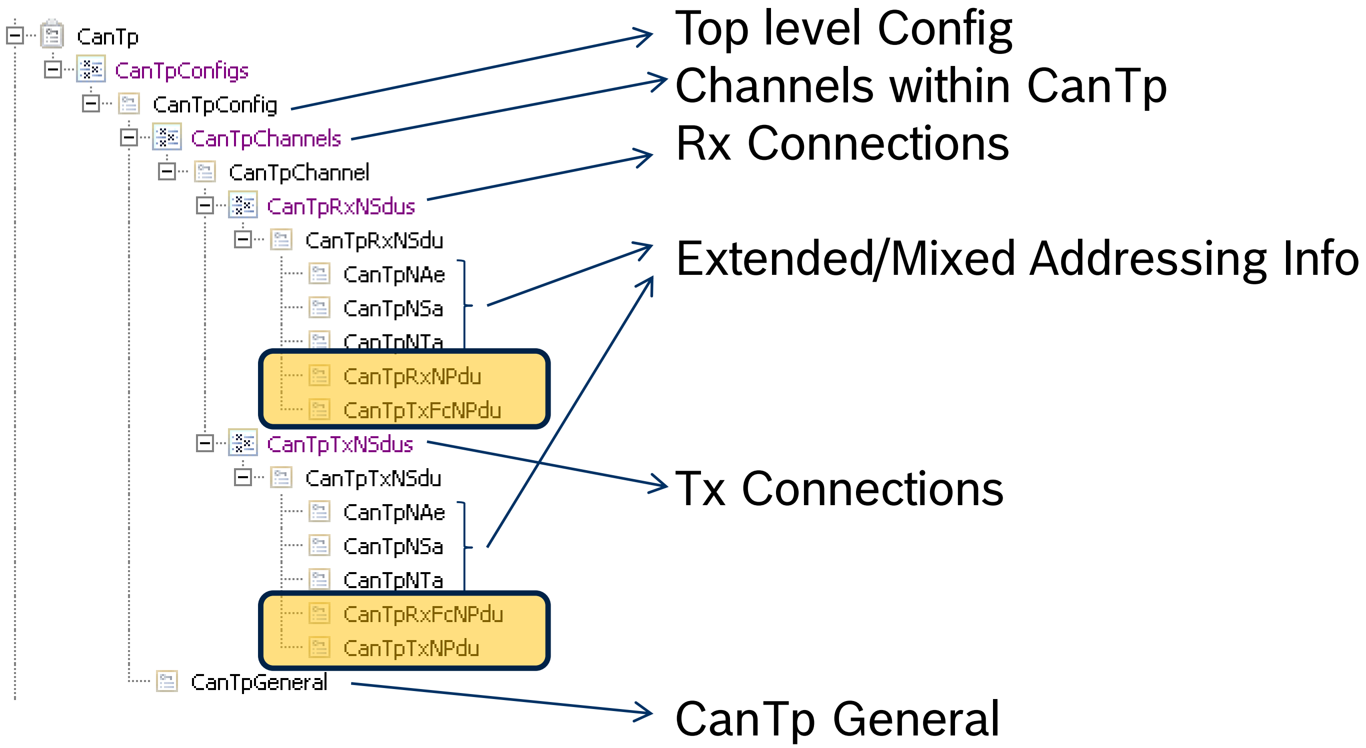


Tester Request
(7 Bytes)
22 10 00 xx xx xx xx

ECU Response
(39 Bytes)
62 10 00 03 04 05 06 07
08 09 0A 0B 0C 0D 0E 0F
10 11 12 13 14 15 16 17
18 19 20 21 22 23 24 25
26 27 28 29 30 31 32

Constraints
CAN_DL = 12

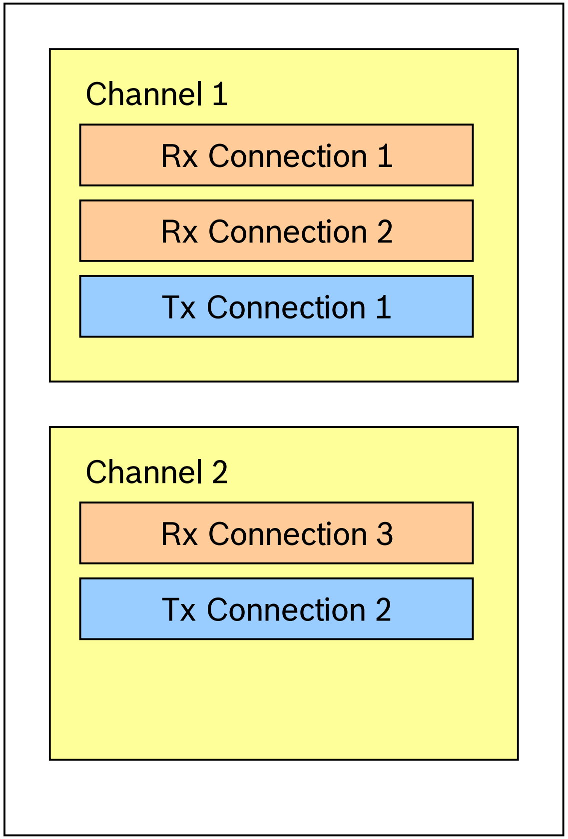
CanTp Configuration Container View (BCT)



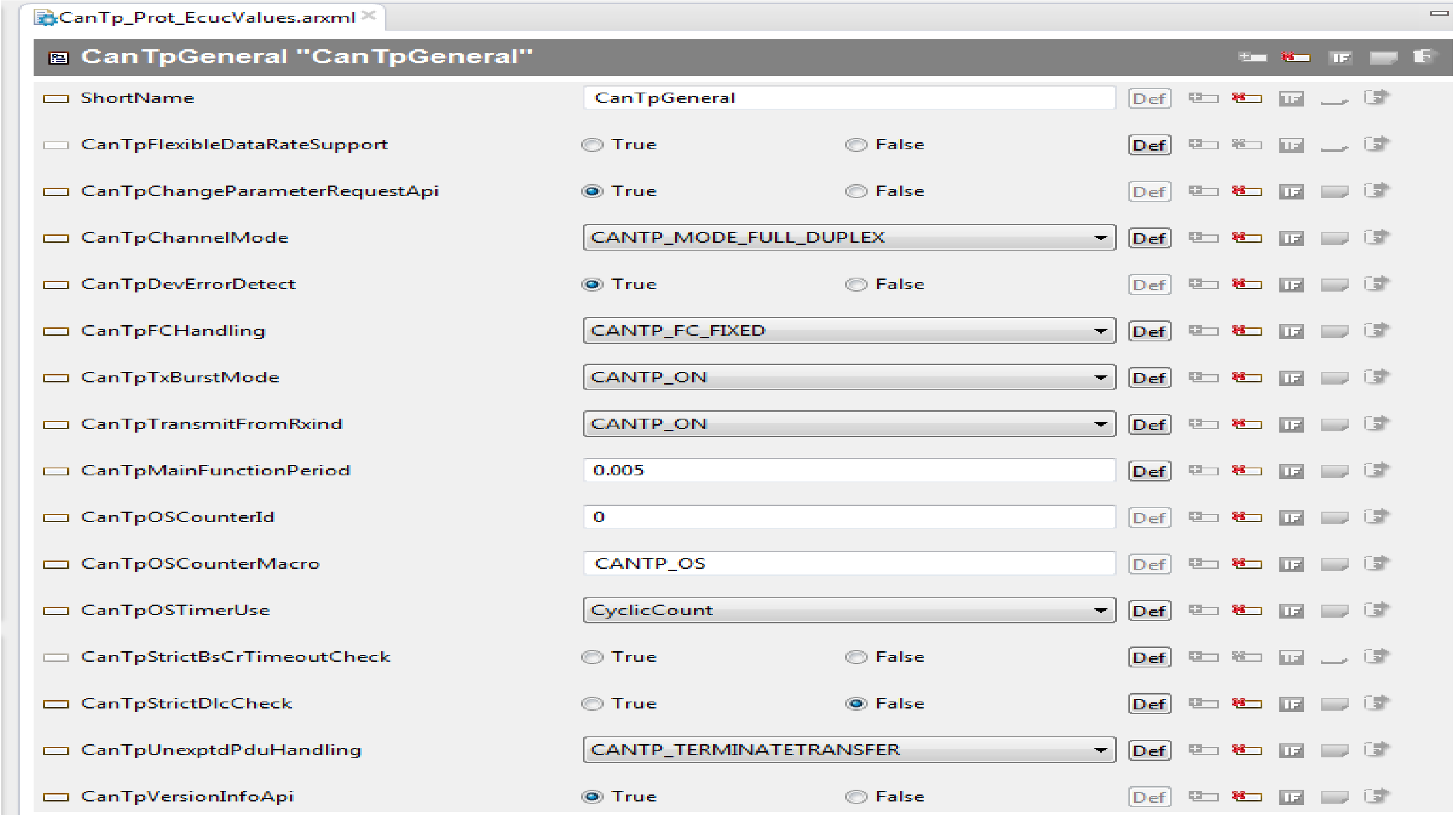
Pdu references to CanIf

CanTp – Channels

- Each Rx/Tx Connection (Tx-N-SDU/Rx-N-SDU) is statically linked to one connection channel. Sharing of Rx Connection or Tx Connection across Channels is basically not allowed.
- Each instance of CanTp Channel is an independent entity. This means that a CanTp channel uses its own resources, such as internal buffer, timer, or state machine.
- If a CanTp Channel is assigned to multiple connections, then resources are shared between different connections, and the CAN Transport Layer will reject transmission or abort receiving, if no free connection channels are available.
- Half-Duplex: Only one connection (including all the Rx and Tx) can be active within a Channel at any given point of time.
- Full-Duplex: Only one Rx connection and one Tx Connection can be active within a Channel at any given point of time. Other Rx and Tx connections would have to wait until the existing connections complete the usage of channel.



CanTp General - Container View (BCT)



CanTp General - Container View (BCT) with CanFD

CanTp_Prot_EcucValues.arxml

CanTpGeneral "CanTpGeneral"

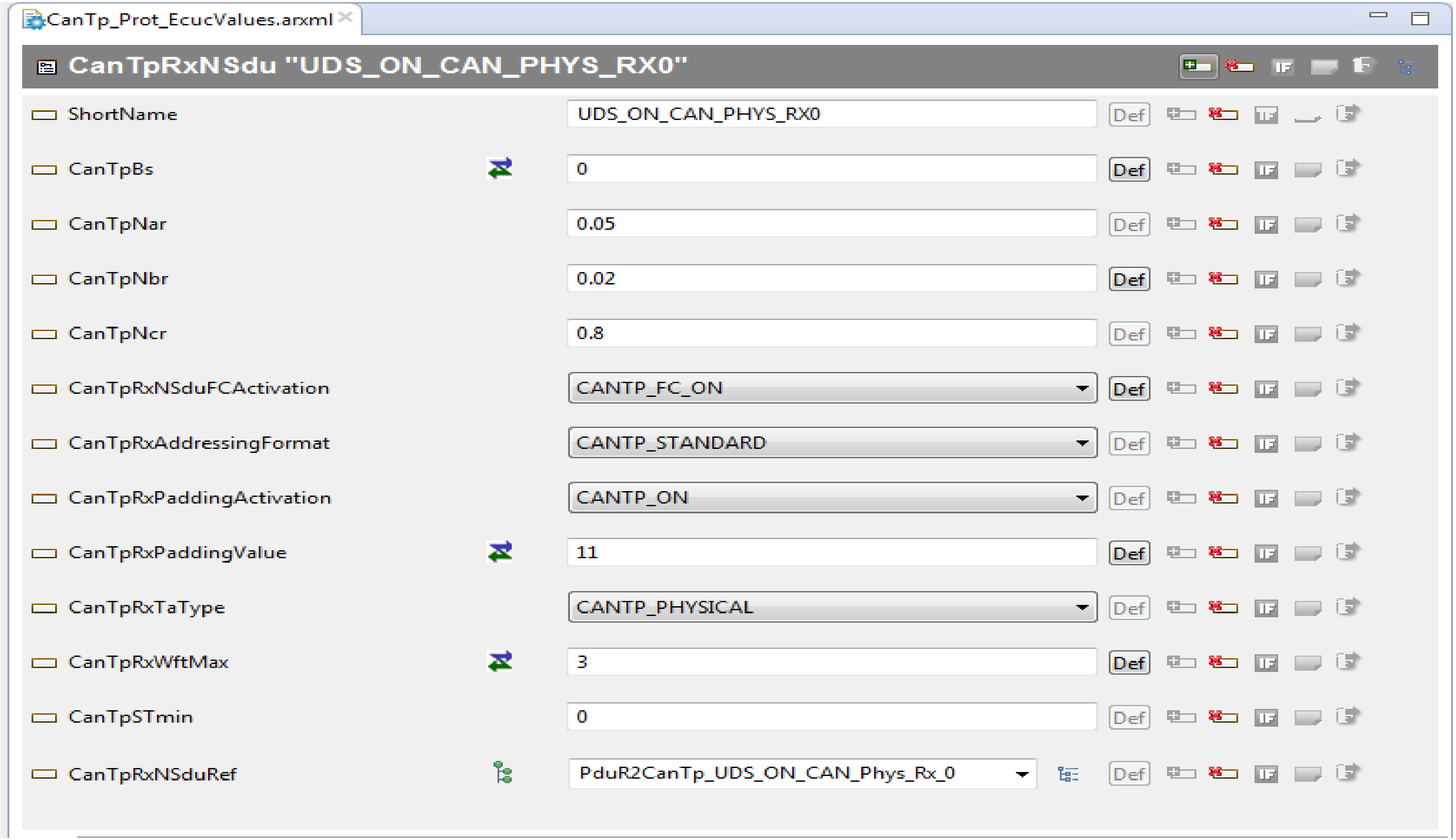
Parameter	Value		Def
ShortName	CanTpGeneral		Def
CanTpFlexibleDataRateSupport	(•) True	( ) False	Def
CanTpChangeParameterRequestApi	(•) True	( ) False	Def
CanTpChannelMode	CANTP_MODE_HALF_DUPLEX		Def
CanTpDevErrorDetect	(•) True	( ) False	Def
CanTpFCHandling	CANTP_FC_ADAPTABLE		Def
CanTpTxBurstMode	CANTP_OFF		Def
CanTpTransmitFromRxind	CANTP_OFF		Def
CanTpMainFunctionPeriod	0.005		Def
CanTpOSCounterId	0		Def
CanTpOSCounterMacro	CANTP_OS		Def
CanTpOSTimerUse	CyclicCount		Def
CanTpStrictBsCrTimeoutCheck	( ) True	( ) False	Def
CanTpStrictDlcCheck	(•) True	( ) False	Def
CanTpUnexptdPduHandling	CANTP_TERMINATETRANSFER		Def
CanTpVersionInfoApi	(•) True	( ) False	Def

Some other requisites after enabling **CanTpFlexibleDataRateSupport** parameter.

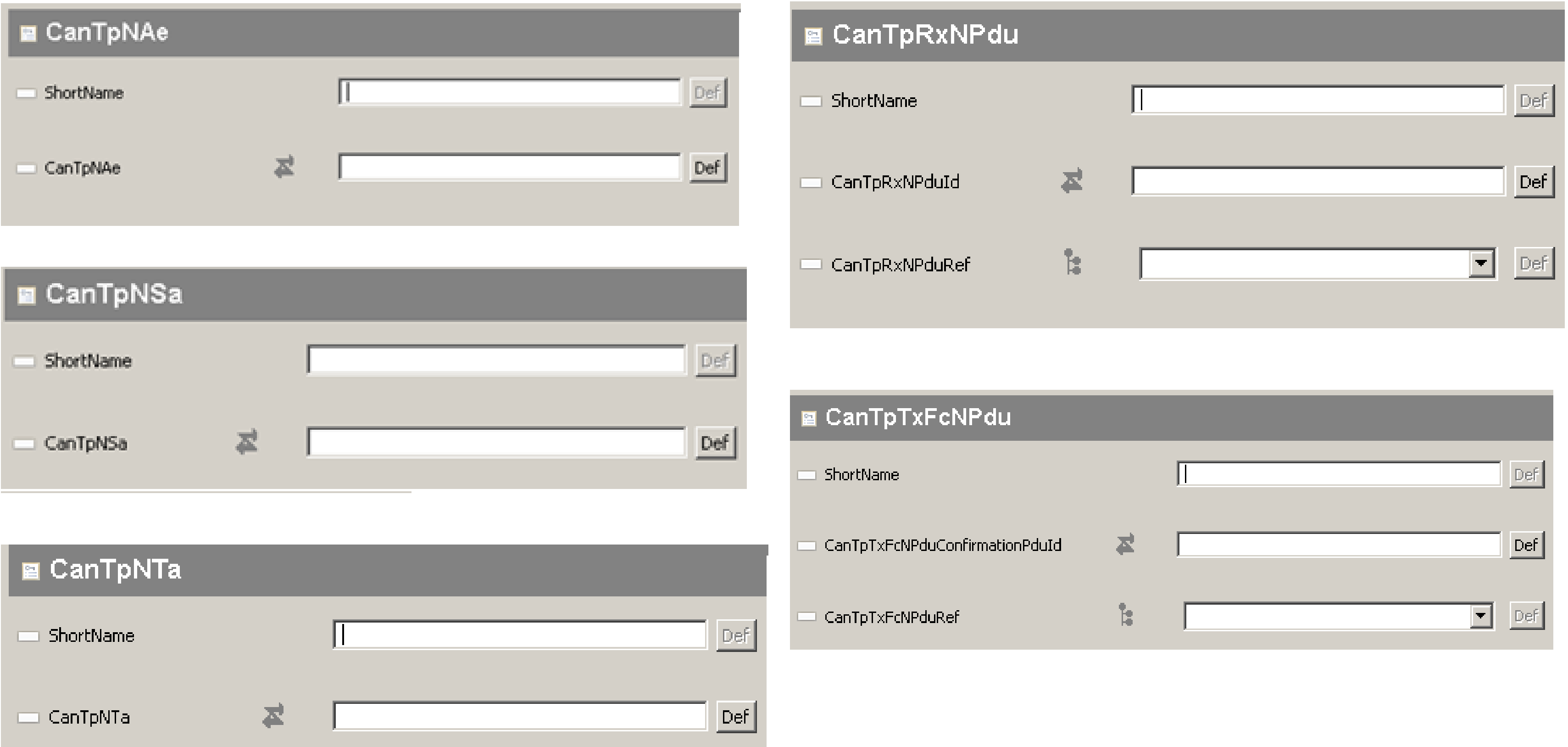
1. **Rx,Tx padding activation** should be ON.
2. **CanTpStrictDlcCheck** should be TRUE.
3. **CanTpTxBurstMode** should be OFF.
4. **CanTpTransmitFromRxind** should be OFF.
5. **PduLength** should be configured from CAN_DL 8,12,16,20…64.

BOSCH

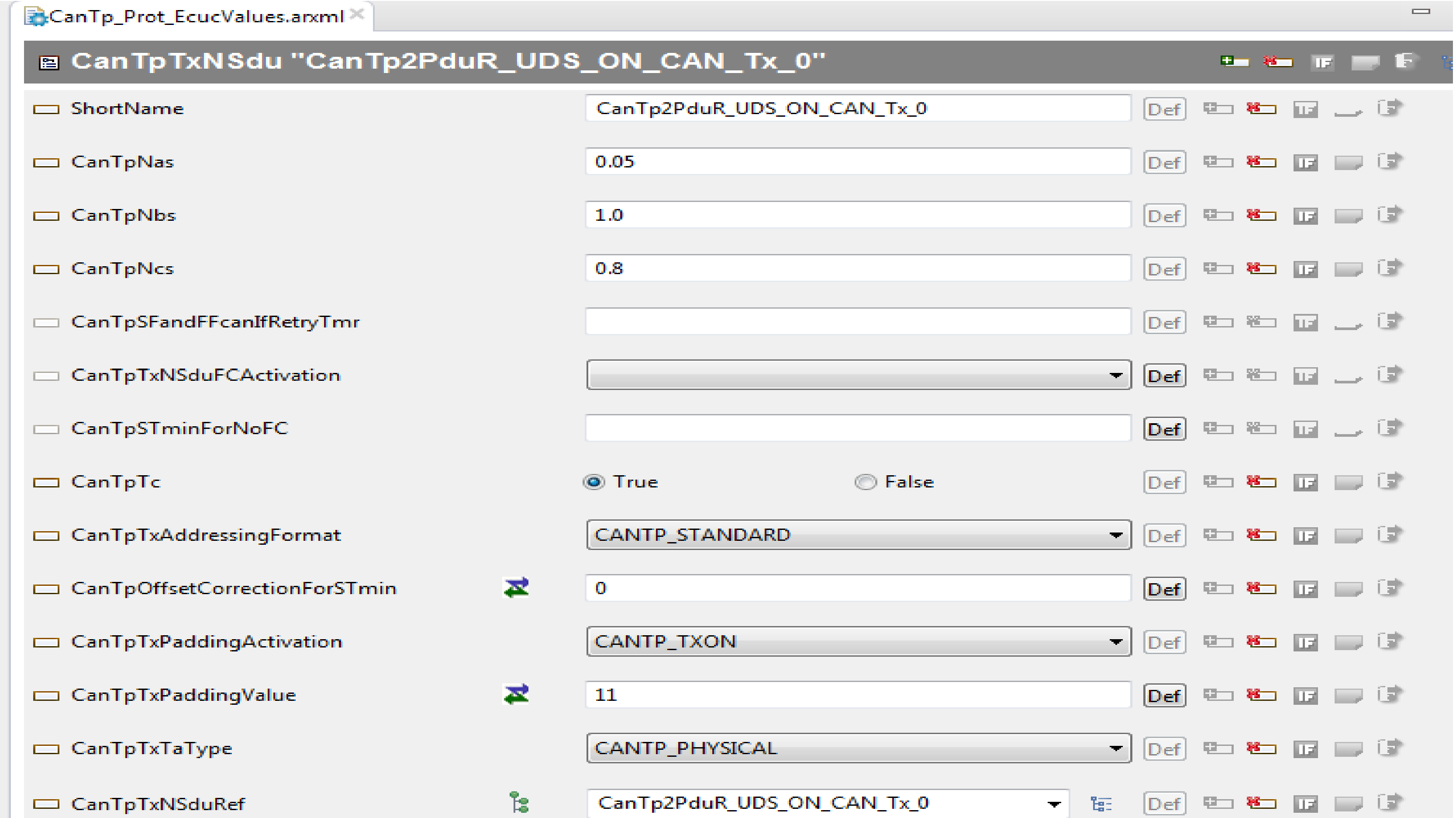
CanTp Rx N Sdu - Container View (BCT)



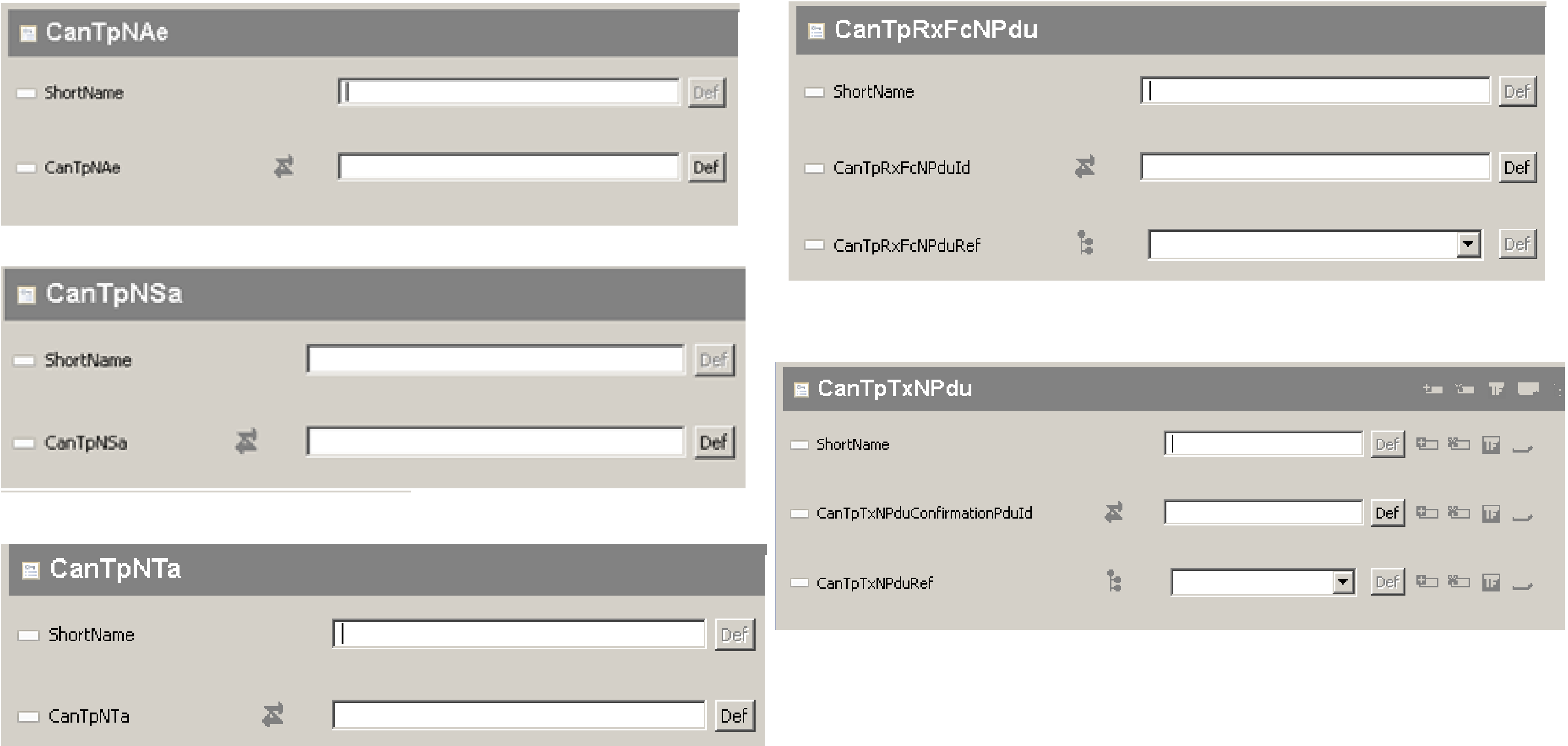
CanTp Rx N Sdu – Sub Container View (BCT)



CanTp Tx N Sdu - Container View (BCT)



CanTp Tx N Sdu – Sub Container View (BCT)




Thank you