MICROSAR FrTp

Technical Reference

Implementation according to ISO 10681-2 for AUTOSAR 4.X communication stacks

Version 2.00.02

Authors	Knut Winkelbach, Oliver Reineke
Status	Released

1 Document Information

1.1 History

Author	Date	Version	Remarks
Knut Winkelbach	2008-09-05	0.1	First version for Pre-Release
Klaus Bergdolt	2008-09-16	0.2	Manual configuration when using ECUC files.
Knut Winkelbach	2009-01-26	1.0	Adapted to new features, explained limitations in more detail.
Knut Winkelbach	2009-03-28	1.1	Clarified interaction with FrIf Removed feature overview
Knut Winkelbach	2009-06-29	1.2	Added description of new features for code size optimization
Knut Winkelbach	2009-07-07	1.3	Rework after review
Knut Winkelbach	2009-11-20	1.4	Described new code optimizations in conjunction with their use cases. Adapted erroneous prototype of FrTp_Transmit() Extended description of GENy configurable options
Knut Winkelbach	2009-12-01	1.5	Rework after review
Knut Winkelbach	2010-04-22	1.6	Straightened description of "Multi Purpose Tp" Added error detection hints and support for "Multiple Identity" of Ecus Added description of feature "FrTp_ChangeParameterRequest" _(extended parameter change) and "FrTp_ReadParameterRequest"
Knut Winkelbach	2010-05-07	1.7	Rework after review
Knut Winkelbach	2010-05-21	1.8	Corrected prototype of FrTp_ReadParameterRequest
Knut Winkelbach	2010-08-20	1.9	Added description of optimizations for GW-operation
Knut Winkelbach	2010-08-26	1.10	Rework after review

Knut Winkelbach	2011-02-07	1.11	Clarified FrIf_Transmit(), FrTp_TriggerTransmit() and description of E_PENDING Optimizations for faster Rx- and Tx-Buffer-Retrieval now controlled by 'FrTp Disable Fast Buf Retrieval' instead of 'Have Fast Segm Stf Seg Rx:' and 'Have Fast Unsegm Stf Unseg Rx'.
Knut Winkelbach	2011-02-21	1.12	Corrected description of 'FrTp Disable Fast Buf Retrieval'
Knut Winkelbach	2011-05-23	1.13	Corrected revision history (aligned version with ALM tool)
Knut Winkelbach	2011-07-27	1.14	Removed description of FrIf_Transmit return code E_PENDING Removed feature providing AUTOSAR 4 prototypes & corrected all features depending on the feature removed. Overworked chapter Multi Identity (new feature Multi Config added)
Knut Winkelbach	2012-01-11	1.15	Added description of FrTp_CancelReceive, updated description of FrTp_CancelTransmit, FrTp_ChangeParameter, removed CancelReason Removed section “Configuration with *.GNY project files” because configuration with FIBEX files is no longer supported by GENy. Overworked section “Configuration” and “Important Hints” as a consequence of this.
Knut Winkelbach	2012-07-10	1.16	Corrected “Contents”
Knut Winkelbach	2013-02-21	1.17	Update to single source concept to be able to generate separate AUTOSAR 3 / 4 documents.
Knut Winkelbach	2013-05-17	1.18	No support of Acknowledge or Acknowledge and Retry
Knut Winkelbach	2014-01-20	1.19	1. No ChangeParameter Confirmation Callback 2. AUTOSAR-compliant prototype of FrTp_ChangeParameter 3. FrTp_Shutdown cancelling all transfers optionally

Knut Winkelbach	2014-05-21	1.20	AUTOSAR 4.1.2 compliant PduR-API (Replacement of NotifResultType by Std_ReturnType) Check separation cycles in FrTp_ChangeParameter() vs. FrTpTimeoutCr
Knut Winkelbach	2014-11-06	2.00.00	Separate development of AUTOSAR4 version. Support of postbuild selectable configurations Support of runtime measurement
Knut Winkelbach	2015-08-14	2.00.01	Added link to Technical Reference “Postbuild Selectable”, introduced new critical section.
Knut Winkelbach	2015-12-17	2.00.02	Updated Deviations Chapter

Table 1-1 History of the document

1.2 Reference Documents

No.	Title	Version
[1]	ISO_IS_10681-2_(E).doc	Date 2009-12-10
[2]	AUTOSAR_SWS_FlexRayISOTransportLayer.pdf	V4.0.0
[3]	AUTOSAR_SWS_DevelopmentErrorTracer.pdf	V3.2.0
[4]	AUTOSAR_SWS_PDURouter.pdf	V3.2.0
[5]	TechnicalReference_Asr_EcuM.pdf	3.00.00
[6]	TechnicalReference_Asr_Dbg.pdf	1.00.00
[7]	TechnicalReference_PostBuildLoadable.pdf	1.03.00
[8]	TechnicalReference_IdentityManager.pdf	1.01.07
[9]	TechnicalReference_Asr4Rtm.pdf	1.01.01
[10]	TechnicalReference_PostBuildSelectable.pdf	1.01.01

Table 1-2 Reference documents

1.3 Scope of the Document

This technical reference describes the general use of the Vector embedded software component Tp_Iso10681 which is an AUTOSAR FrTp basis software (BSW).

⚠ **Please note**
We have configured the programs in accordance with your specifications in the questionnaire. Whereas the programs do support other configurations than the one specified in your questionnaire, Vector´s release of the programs delivered to your company is expressly restricted to the configuration you have specified in the

	questionnaire.

Contents

Figures

Tables

2 Introduction

This document describes the functionality, API and configuration of the software module Tp_Iso10681 as specified in documents [1] and [2].

Supported AUTOSAR Release:	4.0	
Supported Configuration Variants:	pre-compile, link-time, post-build	
Vendor ID:	FrTp_VENDOR_ID	30 decimal (=Vector-Informatik, according to HIS)
Module ID:	FrTp_MODULE_ID	36 decimal (according to ref. [3])

The Tp_Iso10681 component offers data transfer of payload sizes from 1 byte to (2^16-1) bytes (i.e. 64 kByte - 1 byte) using the ISO10681-2 frame format and AUTOSAR 4.X Interfaces to neighbored software components like FrIf and PduR.

2.1 Architecture Overview

The following figure shows where the Tp_Iso10681 - alias FrTp - is located in the AUTOSAR architecture.

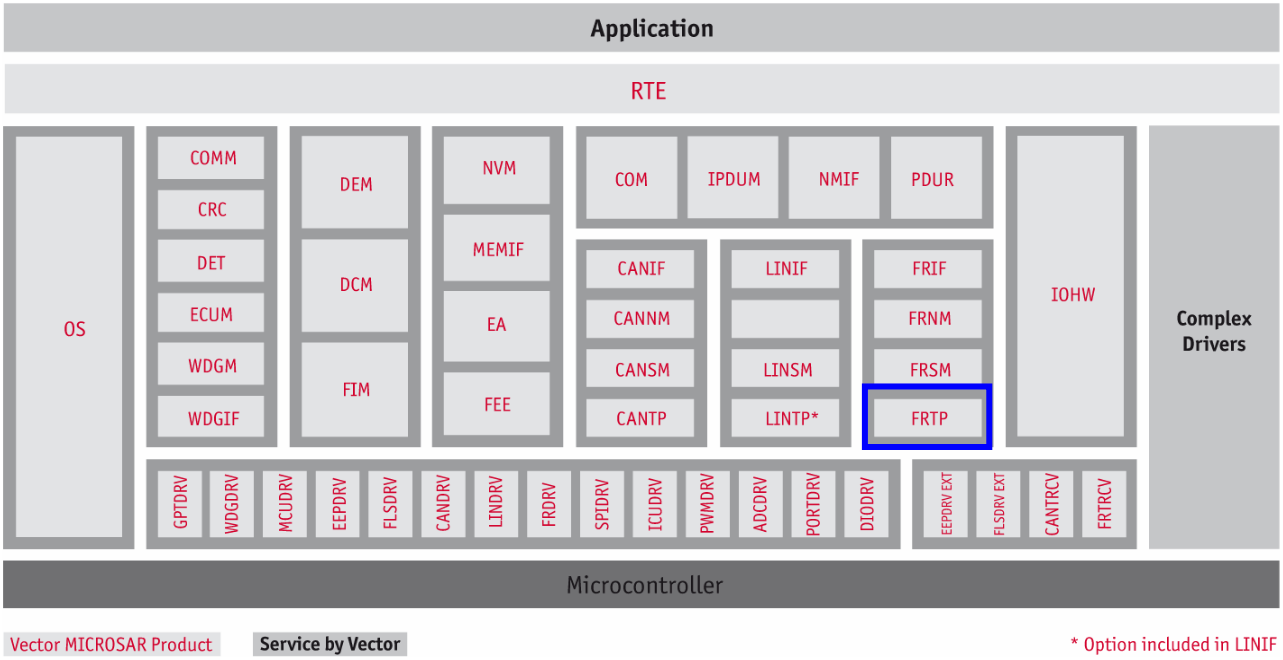

Figure 2-1 AUTOSAR architecture

The interfaces to adjacent modules of the Tp_Iso10681 are described in chapter 5.

3 Functional Description

Please refer to document [1] for details about the TP-protocol of the software-component that is used for communicating on the FlexRay-bus.

Please refer to document [2] for details about the interaction of the software-component with neighbored AUTOSAR-compatible software components like FrIf and PduR.

3.1 Features

The implemented features cover the functionality specified in [1]. Extensions and limitations concerning the AUTOSAR features are presented in chapter 7 in more detail. The following features are provided:

Feature	Summary	Reference
AUTOSAR Debugging	Internal variables can be accessed through the Vector MICROSAR DBG module	[6]
Postbuild Loadable	Configuration parameters can be changed and new connections can be added at post-build time.	[7]
Postbuild Selectable	Support of multiple ECU variants; the currently active variant is selected during initialization.	[8]
AUTOSAR Runtime Measurement	Runtime of functions FrTp_TxConfirmation, FrTp_RxIndication, FrTp_TriggerTransmit, FrTp_MainFunction can be measured	[9]

Table 3-1 Tp_Iso10681 Features

3.2 Error Handling

3.2.1 Development Error Reporting

By default, development errors are reported to the DET using the service `Det_ReportError()as` specified in [3], if development error reporting is enabled (i.e. pre-compile parameter `TP_ISO10681_DEV_ERROR_DETECT==STD_ON).`

If another module is used for development error reporting, the function prototype for reporting the error can be configured by the integrator, but must have the same signature as the service `Det_ReportError().`

The component ID reported in calls to `Det_ReportError()` is 36.

The service IDs reported identify the services which are described in 5.2. The following table presents the service IDs and the related services:

Service ID	Service
0x00	FrTp_Init
0x01	FrTp_Shutdown

Service ID	Service
0x02	FrTp_Transmit
0x03	FrTp_CancelTransmit
0x04	FrTp_ChangeParameter
0x08	FrTp_CancelReceive
0x10	FrTp_MainFunction
0x27	FrTp_GetVersionInfo
0x40	FrTp_TxConfirmation
0x41	FrTp_TriggerTransmit
0x42	FrTp_RxIndication

Table 3-2 Mapping of service IDs to services

The errors reported to DET are described in the following table:

Supported Error Code		Description
0x01	FRTP_E_UNINIT	API service called before initializing the module
0x02	FRTP_E_NULL_PTR	API service called with NULL pointer
0x03	FRTP_E_INVALID_PDU_SDU_ID	FrTp_Transmit called with an invalid parameter value
0x04	FRTP_E_INVALID_PARAMETER	API service called with an invalid parameter value
0x05	FRTP_E_SEG_ERROR	Segmentation would be required according given SduLength but a 1:n connection was selected for the FrTp_Transmit-call.
0x06	FRTP_E_UMSG_LENGTH_ERROR	Transmission of unknown message length was detected but is not supported by this component.
0x07	FRTP_E_NO_CHANNEL	No free rx- or tx-state-machine available

Table 3-3 Errors reported to DET

3.2.2 Production Error Reporting

According to document [2] the Tp_Iso10681 does not support production error reporting.

4 Integration

This chapter gives necessary information for the integration of the Tp_Iso10681 software component into an application environment of an ECU.

4.1 Scope of Delivery

The delivery of the Tp_Iso10681 contains or creates the files which are described in chapters 4.1.1 and 4.1.2:

4.1.1 Static Files

File Name	Description
FrTp.c, FrTp_Rscr.c, FrTp_FrIf.c, FrTp_RxSm.c, FrTp_TxSm.c, FrTp_Util.c	These are the source files of the FrTp
FrTp_Types.h	This header file defines the data types used by the Tp_Iso10681
FrTp_Common.h	This is an internal header file of the Tp_Iso10681
FrTp_Lcfg.h	This header file declares the Link-Time configuration in case the variant Link-Time-configuration is selected
FrTp_PBcfg.h	This header file declares the PB configuration in case the variant PB is selected
FrTp_Cbk.h	This header declares callback functions. The file might be named Tp_Iso10681_Cbk.h in case your integrator has prepared a non-Autosar integration.
FrTp.h	This header declares the Tp_Iso10681 function prototypes. The file might be named Tp_Iso10681.h in case your integrator has prepared a non-Autosar integration.

Table 4-1 Static files

4.1.2 Dynamic Files

The dynamic files are generated by the configuration tool.

File Name	Description
FrTp_PBcfg.c	This file contains generated data defining the PB configuration in case the variant PB is selected
FrTp_Lcfg.c	This file contains generated data defining the Link-Time configuration in case the variant Link-Time-configuration is selected
FrTp_cfg.h	This file is a generated header file defining the FrTp-specific settings of the FrTp
FrTp_GlobCfg.h	This file is a generated header file declaring the global configuration struct of the FrTp.

Table 4-2 Generated files

4.1.3 Version of Files

Please note that each source file has its own version. FrTp.c checks the versions of the subcomponents to be valid. The versions are decimal coded.

4.2 Include Structure

Please refer to the AUTOSAR Basic Software Specifications [2] for detailed information about the include structure.

4.3 Interrupt Locking

In an AUTOSAR environment the FrTp uses the following macros for the integration with the AUTOSAR SchM:

```
#define FRTPISO_ENTER_CRITICAL_SECTION() SchM_Enter_FrTp(
FRTP_EXCLUSIVE_AREA_0 )

#define FRTPISO_LEAVE_CRITICAL_SECTION() SchM_Exit_FrTp(
FRTP_EXCLUSIVE_AREA_0 )

#define FRTPISO_ENTER_CRITICAL_SECTION() SchM_Enter_FrTp(
FRTP_EXCLUSIVE_AREA_1 )

#define FRTPISO_LEAVE_CRITICAL_SECTION() SchM_Exit_FrTp(
FRTP_EXCLUSIVE_AREA_1 )

#define FRTPISO_ENTER_CRITICAL_SECTION() SchM_Enter_FrTp(
FRTP_EXCLUSIVE_AREA_2 )

#define FRTPISO_LEAVE_CRITICAL_SECTION() SchM_Exit_FrTp(
FRTP_EXCLUSIVE_AREA_2 )

#define FRTPISO_ENTER_CRITICAL_SECTION()
SchM_Enter_FrTp_FRTP_EXCLUSIVE_AREA_0()

#define FRTPISO_LEAVE_CRITICAL_SECTION()
SchM_Exit_FrTp_FRTP_EXCLUSIVE_AREA_0()

#define FRTPISO_ENTER_CRITICAL_SECTION()
SchM_Enter_FrTp_FRTP_EXCLUSIVE_AREA_1()

#define FRTPISO_LEAVE_CRITICAL_SECTION()
SchM_Exit_FrTp_FRTP_EXCLUSIVE_AREA_1()

#define FRTPISO_ENTER_CRITICAL_SECTION()
SchM_Enter_FrTp_FRTP_EXCLUSIVE_AREA_2()

#define FRTPISO_LEAVE_CRITICAL_SECTION()
SchM_Exit_FrTp_FRTP_EXCLUSIVE_AREA_2()
```

The exclusive area `FRTP_EXCLUSIVE_AREA_0` is used to prevent simultaneous access to rx- or tx-state-machines' data and member-variables at the same time.

The exclusive area `FRTP_EXCLUSIVE_AREA_1` is used to prevent simultaneous allocation of rx- or tx-state-machines at the same time.

The exclusive area `FRTP_EXCLUSIVE_AREA_2` is used to prevent de-allocation of rx- or tx-state-machines at the same time.

Rx-state-machine allocation is done in the context of calls to function FrTp_RxIndication.

Tx-state-machine allocation is done in the context of calls to function FrTp_Transmit.

State-machine de-allocation is done in the contexts of calls to:

- FrTp_CancelReceive

- FrTp_CancelTransmit

- FrTp_RxIndication (in case an incoming FrTp-pdu ends a running reception)

- FrTp_MainFunction (in case a timeout of an rx- or tx-state-machine occurs)

Info

By default the demo application that is delivered with your software locks all interrupts of the processor in case the "Enter" macro / function is called and releases all interrupts of the processor in case the "Exit" macro is called.

It is recommended to define all exclusive areas (‘0’, ‘1’ and ‘2’) in a way that they lock the communication interrupts of the bus systems affected. These depend on the use case the FrTp is used in:

For use case “Standard Ecu” all critical sections should lock FlexRay-Timer interrupt because this interrupt drives the FlexRay-interfaces JobListExecution (JLE) which issues the FrTp-related communication events FrTp_RxIndication, FrTp_TxConfirmation and FrTp_TriggerTransmit.

For use case “Gateway Ecu” all critical sections should lock the interrupts of all bus-systems that affect the FrTp, e.g:

In case there are CanTp-FrTp-Routings configured then the interrupts that lead to calls of CanTp_RxIndication, CanTp_TxConfirmation should also be locked (in addition to FlexRay-timer-interrupts).

In case there are LinTp-FrTp-Routings configured then the interrupts that lead to calls of LinTp_RxIndication, LinTp_TxConfirmation should also be locked (in addition to FlexRay-timer-interrupts).

This assures that no rx- or tx-events from busses that affect FrTp-communication bring the FrTp statemachine into an inconsistent state.

4.4 Compatibility Checks

The Tp_Iso10681 component conducts a compatibility check during initialization in case the configuration variant is “postbuild”. A failure of this check is caused by generated data that does not fit to the FrTp implementation. In this case the function EcuM_BswErrorHook is called with error code “ECUM_BSWERROR_MAGICNUMBER”.

4.5 Steps to Start Up

In case of problems during starting up the Tp_Iso10681, please read the following two chapters and check the facts described there.

4.5.1 Checklist

These are the "must haves" that have to be fulfilled in order to get the Tp_Iso10681 "up and running":

1. Caution
2. In case no reception or transmission can be done using the Tp_Iso10681 the DET should be set active for the Tp_Iso10681 in the configuration tool.
3. The functions FrTp_InitMemory() and FrTp_Init() have to be called before operation of the Tp_Iso10681.
4. The FrTp_MainFunction() has to be called:
   - once per FlexRay cycle and also
   - synchronous to the FlexRay time.
   - before or after the time of transmission or reception of the FrIf-Pdus used by the FrTp

   Please refer to document [1] to understand the necessity of these rules.
5. The Tp_Iso10681 supports the mixed usage of FrIf-Tx-Pdus that are configured for both decoupled and immediate transmission that are contained within one Tx-Pdu-Pool. However it is strongly recommended to not use such a scenario because the transmission sequences of TP-frames that are sent cleanly in order but that have different transmission types can be swapped on the FlexRay-bus depending on the FrIf-Job configuration.

4.5.2 Transmission

In case you want to transmit data from an ECU with the Tp_Iso10681 integrated, these facts should be fulfilled, i.e. in case of problems each fact should be checked by applying breakpoints in the ECU-software using a debugging-tool:

1. *FrTp_Transmit()* should return E_OK (‘0’).
2. During the next scheduling of the *FrTp_MainFunction()* the Tp_Iso10681 should request the upper layer (typically the “PduR”) to copy the Tx-data into the TP buffer by a call to *PduR_FrTpCopyTxData()*.
3. After the successful provision of Tx-data in that function the transmission should begin. You can verify the start of the transmission of TP-Tx-frames by assuring that the function *FrIf_Transmit()* is called by the Tp_Iso10681 and the functions return value is E_OK..
4. After that (i.e. after the successfull call to *FrTp_TriggerTransmit* for decoupled Pdus) the function *FrTp_TxConfirmation()* is called by the *FrIf*-component that is used. There is only one location in Tp_Iso10681.c where *FrIf_Transmit()* is called.
5. Each call to *FrIf_Transmit()* of the Tp_Iso10681 should be acknowledged by a call to *FrTp_TxConfirmation()* done by the FrIf component.
6. In any case the callback *<Owner>_FrTpTxConfirmation()* should be called after the transmission request via *FrTp_Transmit()* was accepted by the Tp_Iso10681 by

returning E_OK. The notification-code given to *<Owner>_FrTpTxConfirmation()* will provide additional information about success or failure of the transmission.

Please refer to document [2] on how to interpret parameters and notification-codes of the functions mentioned.

4.5.3 Reception

In case you want to receive data on an ECU with the Tp_Iso10681 integrated, these facts should be fulfilled, i.e. in case of problems each fact should be checked e.g. by applying breakpoints in the ECU-software using a debugging-tool:

1. After the first Tp_Iso10681-frame has been “detected” on the bus by an appropriate analyser-tool, the function *FrTp_RxIndication()* should be called by the *FrIf*-component that is used.
2. During the next scheduling of the *FrTp_MainFunction()* the Tp_Iso10681 should indicate to the upper layer that new Rx-data is available by calling *PduR_FrTpStartOfReception()*. Afterwards FrTp calls *PduR_FrTpCopyRxData()* in order to store the Rx-data.
3. After the successful provision of Rx-buffer-space the reception should begin, e.g. by the transmission of a flow control frame using *FrIf_Transmit()* in case of an 1:1 connection. You can verify the transmission of flow control frames by assuring that the function *FrIf_Transmit()* is called by the Tp_Iso10681 and the function *FrTp_TxConfirmation()* is called by the *FrIf*-component that is used. There is only one location in Tp_Iso10681.c where *FrIf_Transmit()* is called.
4. Each call to *FrIf_Transmit()* of the Tp_Iso10681 should be acknowledged by a call to *FrTp_TxConfirmation()* done by the FrIf component.
5. In any case the callback *<Owner>_FrTp_RxIndication()* should be called after the successful reception of at least one Iso10681-frame. The notification-code given to *<Owner>_FrTp_RxIndication()* will provide additional information about success or failure of the reception.

Please refer to document [2] on how to interpret parameters and notification-codes of the functions mentioned.

4.5.4 Buffer Handling

During one call of the *FrTp_MainFunction()* the retrieval of Tx-data or Rx-buffer-space is done. For Rx-buffer-retrieval no limitations apply, because the ISO 10681-2 protocol allows the FrTp to communicate to the sender the exact buffer size available at the receiving application. However for Tx-buffer-retrieval the following restriction applies:

Caution
Please note, that the FrTp does not support several calls to PduR_FrTpCopyTxData / PduR_FrTpCopyRxData in case the application does not supply the Tx-data / Rx-buffer-space of **functional connections** in one piece. I.e. in case of a 246 byte unsegmented functional STF the application has to process the complete payload within exactly one call to PduR_FrTpCopyTxData / PduR_FrTpCopyRxData.

4.6 Postbuild Variants

4.6.1 Post-build selectable

The MICROSAR Identity Manager (refer to [8]) is an implementation of the AUTOSAR 4 post-build selectable concept. It allows the ECU manufacturer to include several FrTp configurations within one ECU. With post-build selectable and the Identity Manager the ECU variants are downloaded within the ECUs non-volatile memory (e.g. flash) at ECU build time. Post-build selectable does not allow modification of FrTp aspects after ECU build time. At the same time, this limitation allows some of the optimization strategies still to be effective - FrTp static code part will be optimized for the variant with maximum configuration size. The variant selection is performed at runtime by passing the corresponding configuration root during the module initialization (refer to chapter 5.2.1.1).

4.6.2 Post-build loadable

All FrTp configuration parameters, that are classified to be post-build selectable, also do support post-build loadable variant. The differences to the post-build-selectable case relisted upon their qualification:

- advantages:
  - The module’s configuration can be updated after the module’s compile time without reprogramming the whole ECU software.
- disadvantages:
  - Since all of the affected configuration-parameters may change after modules compile time, the optimization level of the source code is very low.
  - Since no maximum configuration size can be pre-calculated, some scalable RAM blocks are referred not by a direct linker symbol, but through a pointer.
  - Only one configuration variant is supported at a time (no variant selection at runtime possible). This disadvantage is avoided if the post -build loadable selectable variant is chosen instead (refer to chapter 4.6.3).
  - Greater risks of passing an invalid pointer during module initialization time. For details about the post-build loadable feature, please refer to [10].

4.6.3 Post-build loadable selectable

This variant actually combines both post-build selectable and loadable variants, allowing a variant selection at runtime and at the same time post-build calibration of parameters. For details on the two mentioned variants, please refer correspondingly to chapters 4.6.1 and 4.6.2.

4.6.4 Post-build delete-able

This variant is actually a specific sub-variant of the post-build loadable variants. It allows deleting of containers that were created at link time, by guaranteeing at the same time the preservation of other post-build capable parameters' values. For details about this feature, refer to document [10].

4.7 Recommended compiler switches

Note

In case you want to use the configuration variant post build together with a Greenhills® compiler on platform NEC® V850™ and compatible then you should take attention on the following compiler switch combination:

In case you want to use option `-pack=1` to reduce code size you should always use `–misalign_pack` at the same time in order to prevent a strong increase of the size of the static code of the Tp_Iso10681.

5 API Description

5.1 Type Definitions

All AUTOSAR compliant type definitions of the AUTOSAR FrTp 3.0 are also used by the Tp_Iso10681 component. Please refer to document [2] for details.

5.2 Services provided by Tp_Iso10681

5.2.1.1 FrTp_Init

FrTp_Init()

Prototype	
Depending on your order there are two variants of that service available: AUTOSAR-compliant variant: void **FrTp_Init** (void) The Vector-specific variant lets the Tp_Iso10681 support Post-build and Multi-Identity configurations: void **FrTp_Init** (void* CfgPtr)	
Parameters [in/out/both]	
void[in] void*[in]	AUTOSAR-compliant variant: n/a Vector-specific variant: Pointer to the post-build-configuration structure that shall be used for initialization.
Return code	
void	n/a
Service ID	
Service ID	0x00
Functional Description	
This method initializes all administration-data of the Tp_Iso10681 and prepares the Tp_Iso10681 component for operation. It has to be called prior to any other function-call to a Tp_Iso10681-API. Otherwise the respective other API-call will report an error to the DET.	
Preconditions	
FrTp_InitMemory() has to be called prior to FrTp_Init() in case the startup code does not set the RAM used by Tp_Iso10681 to zero.	
Postconditions	
After the call to this method the Tp_Iso10681-component is ready to operate.	

Particularities and Limitations
During its runtime this method temporarily locks interrupts using the API-calls of the AUTOSAR BSW Scheduler (SchM). This is because this function reuses internal functions of the Tp_Iso10681 that are also used during runtime and that require interrupt-locking. In case of post build configuration this method checks the validity of the given configuration versus the version of the embedded code of the Tp_Iso10681 component. In case the given configuration is invalid or not suitable for the embedded code then the callout function EcuM_GeneratorCompatibilityError is called.
Call context
This function has to be called during the initialization of the FlexRay-Stack, i.e. during ECU-startup.

5.2.1.2 FrTp_InitMemory

FrTp_Init()

Prototype	
void **FrTp_InitMemory** (void)	
Parameters [in/out/both]	
void[in]	n/a
Return code	
void	n/a
Service ID	
Service ID	This method has no service ID because it is not an AUTOSAR compliant API.
Functional Description	
This method initializes all administration-data of the Tp_Iso10681 that has to be set to zero before FrTp_Init() is called. FrTp_InitMemory() has to be called in case the startup code does not set the RAM used by Tp_Iso10681 to zero.	
Preconditions	
n/a	
Postconditions	
After the call to this method the Tp_Iso10681-component is ready to operate.	
Particularities and Limitations	
n/a	
Call context	
This function has to be called during the initialization of the FlexRay-Stack, i.e. during ECU-startup in case the startup code does not initialize the RAM.	

5.2.1.3 FrTp_Shutdown

FrTp_Shutdown()

Prototype	
void **FrTp_Shutdown** (void)	
Parameters [in/out/both]	
void [in]	n/a
Return code	
void	n/a
Service ID	
Service ID	0x01
Functional Description	
This method shuts down the Tp_Iso10681 and prevents the initiation of transmissions in future. After the call the Tp_Iso10681 is in a non-initialized state.	
Preconditions	
At least *FrTp_Init()* has to be called prior to *FrTp_Shutdown()*. Otherwise an error will be indicated to the DET.	
Postconditions	
After the call to this function no further API-calls are possible without rising a DET error in such a call.	
Particularities and Limitations	
This function solely changes a Tp_Iso10681-internal status-variable. It does not cancel or stop any ongoing transmissions. During its runtime this method temporarily locks interrupts using the API-calls of the AUTOSAR BSW Scheduler (SchM). In case of feature “Have Full FrTp_Shutdown” being active then this function terminates all active connections, including callouts to all affected upper layers. In case of feature “Disable FrTp_Shutdown” being active then this function does nothing because its source code then is disabled in order to save ressources.	
Call context	
This function can be called at any time after Tp_Iso10681-initialization. This function may not be called in interrupt context because this call might interrupt another Tp_Iso10681-API-call which will lead to an unpredictable behavior of the API call which was interrupted.	

5.2.1.4 FrTp_Transmit

FrTp_Transmit()

Prototype	
Std_ReturnType **FrTp_Transmit** (PduIdType FrTpTxSduId, const PduInfoType* FrTpTxSduInfoPtr)	
Parameters [in/out/both]	
PduIdType [in]	This parameter identifies the connection to be used for transmission.
PduInfoType* [in]	This is a pointer to a struct of type PduInfoType whereas the structs parameter PduLength states the length of the data to be transmitted. Unlike the prototype described in the AUTOSAR SWS the Vector-prototype declares the pointer to be constant in order to reduce compiler-warnings.

Return code	
Std_ReturnType	E_OK: This value is returned in case the transmission request is going to be executed by the Tp_Iso10681 component. E_NOT_OK: This value is returned in case the transmission request is not going to be executed by the Tp_Iso10681 component. This can happen under these conditions: 1. The length-value given in the struct-parameter is invalid, i.e. either larger than 64 kBytes or, in case of a 1:n connection, it might be larger than the payload of the smallest FrIf-Pdu that is assigned to the affected connection. 2. A connection identifier is provided that is invalid. Then the value usually is larger or equal than the overall number of connections configured in the current Tp_Iso10681 instance. 3. In case 1. or 2. are not applicable then typically all channels of the Tp_Iso10681 are busy at the moment of the call.
Service ID	
Service ID	0x02
Functional Description	
This method initiates a transmission in case the given parameters are valid (see section “Return code” above). In this case a free channel of the Tp_Iso10681 is reserved to conduct the transmission requested. Within the next call to the *FrTp_MainFunction()* the steps required for the transmission will be taken, i.e. a buffer with Tx-Data is written by the application via *<Owner>_ FrTpCopyTxData()* (For AUTOSAR <Owner> is PduR), and after successful transmission of the provided data a *<Owner>_FrTpTxConfirmation()*-callback is called. These steps are conducted in the same way as described in document [2].	
Preconditions	
FrTp_Init() has to be called prior to any calls to this method, i.e. the Tp_Iso10681 has to be initialized. Otherwise an error will be indicated to the DET. It is not required that the physical layer is in sync with the networks since this function does not check the status of the physical FlexRay bus.	
Postconditions	
The PduR / caller should be prepared to receive a call to the callback-function *PduR_FrTpCopyTxData()* afterwards, in order to provide the Tx-data to be transmitted.	
Particularities and Limitations	
During its runtime this method temporarily locks interrupts using the API-calls of the AUTOSAR BSW Scheduler (SchM).	
Call context	
This function can be called at any time after Tp_Iso10681-initialization.	

5.2.1.5 FrTp_CancelTransmit

FrTp_CancelTransmit()

Prototype
Std_ReturnType FrTp_CancelTransmit (PduIdType FrTpTxPduId)

Parameters [in/out/both]	
PduIdType [in]	This parameter identifies the connection a cancellation request shall be applied to.
Return code	
Std_ReturnType	E_OK: This value is returned in case the request for transmission-cancellation is going to be executed by the component. E_NOT_OK: This value is returned in case the request for transmission-cancellation is invalid. This can happen in case the affected connection is not active at all.
Service ID	
Service ID	0x03
Functional Description	
This method requests the cancellation of an active transmission as described in document [1].	
Preconditions	
FrTp_Init() has to be called prior to any calls to this method, i.e. the Tp_Iso10681 has to be initialized. Otherwise an error will be indicated to the DET. It is not required that the physical layer is in sync with the networks since this function does not check the status of the physical FlexRay bus.	
Postconditions	
After successful cancellation of the transmission no additional calls to *PduR_FrTp_TxConfirmation()* or *<Owner>_FrTp_TxConfirmation()* are done by the Tp_Iso10681 in order to finish the transmission.	
Particularities and Limitations	
During its runtime this method temporarily locks interrupts using the API-calls of the AUTOSAR BSW Scheduler (SchM). This API is optional. It is enabled by the GUI switch ‘Transmit Cancellation’.	
Call context	
This function can be called at any time after Tp_Iso10681-initialization.	

5.2.1.6 FrTp_CancelReceive

FrTp_CancelReceive()

Prototype	
Std_ReturnType FrTp_CancelReceive (PduIdType FrTpRxPduId)	
Parameters [in/out/both]	
PduIdType [in]	This parameter identifies the connection a cancellation request shall be applied to.
Return code	
Std_ReturnType	E_OK: This value is returned in case the request for reception-cancellation is going to be executed by the component. E_NOT_OK: This value is returned in case the request for reception-cancellation is invalid e.g. because the affected connection is not active at all.

Service ID	
Service ID	0x08
Functional Description	
This method requests the cancellation of an active reception as described in document [1].	
Preconditions	
FrTp_Init() has to be called prior to any calls to this method, i.e. the Tp_Iso10681 has to be initialized. Otherwise an error will be indicated to the DET. It is not required that the physical layer is in sync with the networks since this function does not check the status of the physical FlexRay bus.	
Postconditions	
After successful cancellation of the reception no additional calls to *PduR_FrTp_RxIndication()* or *<Owner>_FrTp_RxIndication()* are done by the Tp_Iso10681 in order to finish the reception.	
Particularities and Limitations	
During its runtime this method temporarily locks interrupts using the API-calls of the AUTOSAR BSW Scheduler (SchM). This API is optional. It is enabled by the GUI switch 'Receive Cancellation'.	
Call context	
This function can be called at any time after Tp_Iso10681-initialization.	

5.2.1.7 Reading and Writing Parameters during Runtime

The following methods can be used to read and write connection specific parameters during runtime.

FrTp_ChangeParameter ()

Prototype	
Std_ReturnType **FrTp_ChangeParameter** (PduIdType FrTpSduId, TPParameterType FrTpParameterType, unit16 FrTpParameterValue)	
Parameters [in/out/both]	
PduIdType [in]	This parameter identifies the connection a request to change parameters shall be applied to.
TPParameterType [in]	This parameter contains the information about which parameter of the given connection shall be changed.
unit16 [in]	This parameter contains the new value to be set as new parameter-value.
Return code	
Std_ReturnType	E_OK: Request was accepted E_NOT_OK: Request was not accepted
Service ID	
Service ID	0x04

Functional Description
This method changes the parameter of a Tp_Iso10681 connection identified by the type-parameter using the given value-parameter as value. In case the SCexp part of the given BC-parameter leads to a larger inter-pdu-gap than configured in the FrTpTimeoutCr-parameter of the affected connection the parameter will not be changed and E_NOT_OK will be returned.
Preconditions
FrTp_Init() has to be called prior to any calls to this method, i.e. the Tp_Iso10681 has to be initialized. Otherwise an error will be indicated to the DET. It is not required that the physical layer is in sync with the networks since this function does not check the status of the physical FlexRay bus.
Postconditions
n/a
Particularities and Limitations
During its runtime this method temporarily locks interrupts using the API-calls of the AUTOSAR BSW Scheduler (SchM).
Call context
This function can be called at any time after Tp_Iso10681-initialization.

5.2.1.7.1 FrTp_ReadParameter

FrTp_ReadParameter ()

Prototype	
Std_ReturnType **FrTp_ReadParameter**(PduIdType FrTpSduId, TPParameterType FrTpParameterType, uint16 FrTpParameterValuePtr)	
Parameters [in/out/both]	
PduIdType [in]	This parameter identifies the connection a request to read parameters shall be applied to.
TPParameterType FrTpParameterType [in]	This parameter contains the information which type of parameter shall be read.
uint16 FrTpParameterValuePtr [out]	This parameter contains the pointer to a variable of type uint16 where the value of the parameter to be read shall be stored in.
Return code	
void	-
Service ID	
Service ID	0x81
Functional Description	
In its basic extent this method reads the BC-parameter of a Tp_Iso10681 connection. Only the value "`TPPARAMTYPE_FRTP_BC`" is allowed for parameter FrTpParameterType. Depending on the preparation of your delivery this method also supports the reading of other parameters of the connections of the Tp_Iso10681 component. Please refer to chapter 5.2.1.7.2 for more details.	

Preconditions
FrTp_Init() has to be called prior to any calls to this method, i.e. the Tp_Iso10681 has to be initialized. Otherwise an error will be indicated to the DET. It is not required that the physical layer is in sync with the networks since this function does not check the status of the physical FlexRay bus.
Postconditions
n/a
Particularities and Limitations
During its runtime this method temporarily locks interrupts using the API-calls of the AUTOSAR BSW Scheduler (SchM). This parameter is optional. It is included in your delivery only in case you have ordered the full extent of functionality defined by the ISO10681-2.
Call context
This function can be called at any time after Tp_Iso10681-initialization.

5.2.1.7.2 TPParameterType

Parameters can be read and written using the methods in the previous 2 chapters.These prototypes are using the following #defines and type definitions from TPParameterType in header ComStack_Types.h**:**

- TP-independent type definition:

```
typedef enum
{
  TP_STMIN, /* Separation Time, NOT USED in a TP acc. to ISO10681-2 */
  TP_BS, /* Block Size, NOT USED in a TP acc. to ISO10681-2 */
  TP_BC /* The Bandwidth-Control parameter acc. to ISO10681-2 */
} TPParameterType;
```

5.2.1.8 FrTp_MainFunction

FrTp_MainFunction()

Prototype	
void **FrTp_MainFunction** (void)	
Parameters [in/out/both]	
void [in]	n/a
Return code	
void	n/a
Service ID	
Service ID	0x10

Functional Description

This method conducts these actions:

- It supervises the timers of each TP-channel.
- It evaluates the actions that have happened since the last call to *FrTp_MainFunction()*, i.e. callbacks to the methods *FrTp_RxIndication()*, *FrTp_TxConfirmation()*.
- It conducts all calls to *FrIf_Transmit()* required by a Tp_Iso10681 communication.

Preconditions

FrTp_Init() has to be called prior to any calls to this method, i.e. the Tp_Iso10681

has to be initialized. Otherwise an error will be indicated to the DET.

Postconditions

n/a

Particularities and Limitations

By design this method conducts most of the actions, in order to keep the runtime of the functions *FrTp_RxIndication(), FrTp_TxConfirmation()* and *FrTp_TriggerTransmit()* small which might be called in interrupt context.

During its runtime this method temporarily locks interrupts using the API-calls of the AUTOSAR BSW Scheduler (SchM).

This function has to be called synchronous to the FlexRay bus-time.

Call context

This function can be called at any time after initialization as long as the scheduling limitations in the section above are obeyed.

This function has to be called at a different time than the execution-time of the FrIf-job-execution (Both Rx and Tx !!!). Both points in time are possible: Prior or after the point of time of the FrIf-job-execution.

Caution

The *FrTp_MainFunction()* has to be called:

- once per FlexRay cycle and also
- synchronous to the FlexRay time.

Please refer to document [1] to understand the necessity of this rule.

Caution

This function has to be called at a different time than the execution-time of the FrIf-job-execution that operates the FrIf-Rx- and FrIf-Tx-Pdus of the Tp_Iso10681.

Example for the scheduling of the *FrTp_MainFunction()*:

In case of 5 ms FlexRay cycle length and all FrIf-Tx-Pdus assigned to the Tp_Iso10681 being in the dynamic segment then the *FrTp_MainFunction()* should be executed in the static segment.

The offset within the cycle should be:

- earlier than the Tx-Job of the FrIf responsible for Tp_Iso10681 FrIf-Tx-Pdus
- later than the Rx-Job of the FrIf responsible for Tp_Iso10681 FrIf-Rx-Pdus

	Other offsets are possible but will result in poor performance of the Tp_Iso10681.

5.2.1.9 FrTp_GetVersionInfo

FrTp_GetVersionInfo()

Prototype	
void **FrTp_GetVersionInfo** (Std_VersionInfoType *pVersionInfo)	
Parameters [in/out/both]	
Std_VersionInfoType* [in]	The version-info-struct the parameter “pVersionInfo” points to is filled with the version-information by this method.
Return code	
Void	n/a
Service ID	
Service ID	0x27
Functional Description	
This method fills the version-info-struct that parameter “pVersionInfo” points to with the Tp_Iso10681 version as described in document [2].	
Preconditions	
FrTp_Init() has to be called prior to any calls to this method, i.e. the Tp_Iso10681 has to be initialized. Otherwise an error will be indicated to the DET. It is not required that the physical layer is in sync with the networks since this function does not check the status of the physical FlexRay bus.	
Postconditions	
n/a	
Particularities and Limitations	
n/a	
Call context	
This function can be called at any time after Tp_Iso10681-initialization.	

5.3 Services used by FrTp

The following tables list services, which are provided by other components and used by Tp_Iso10681. For details about prototype and functionality refer to the documentation of the component providing the service that is of interest.

5.3.1 Services with AUTOSAR compliant prototypes

Component	**API**
Variant without TP_ISO10681_HAVE_MULTI_PURPOSE_TP	
AUTOSAR PduR	`BufReq_ReturnType PduR_FrTpCopyTxData(`

	PduIdType DcmTxPduId, PduInfoType* info, RetryInfoType* retry, PduLengthType* availableDataPtr)
AUTOSAR PduR	BufReq_ReturnType PduR_FrTpCopyRxData(PduIdType DcmRxPduId, PduInfoType* dataAtTp, PduLengthType* bufferSizePtr)
AUTOSAR PduR	BufReq_ReturnType PduR_FrTpStartOfReception(PduIdType DcmRxPduId, const PduInfoType* info, PduLengthType TpSduLength, PduLengthType* bufferSizePtr)
AUTOSAR PduR	void PduR_FrTpRxIndication(PduIdType FrTpRxPduId, Std_ReturnType Result)
AUTOSAR PduR	void PduR_FrTpTxConfirmation (PduIdType FrTpTxPduId, Std_ReturnType Result)
AUTOSAR PduR	PduR_FrTpCancelTransmitConfirmation(PduIdType connId)
Variant with TP_ISO10681_HAVE_MULTI_PURPOSE_TP	
User-defined application	void <Owner>_FrTpRxIndication(PduIdType FrTpRxPduId, Std_ReturnType Result)
User-defined application	void <Owner>_FrTpTxConfirmation (PduIdType FrTpTxPduId, Std_ReturnType Result)
User-defined application	void <Owner>_FrTpCancelTransmitConfirmation(PduIdType connId)

Table 5-1: Services used by Tp_Iso10681

In case variant TP_ISO10681_HAVE_MULTI_PURPOSE_TP is used the complete name of the interaction functions can be changed by the user for up to 3 different upper layer units. Only return-type and parameters (i.e. the signature) have to be the same.

Note
In case the feature TP_ISO10681_HAVE_MULTI_PURPOSE_TP is used then no distinction is done between particular Sdu-owners for these methods:

- PduR_FrTpCopyRxData
- PduR_FrTpCopyTxData

i.e. the token PduR is not replaced by <Owner> and the name of the method is not configurable at all.

Note
Some deliveries do not offer all services listed above.
In case the feature TP_ISO10681_HAVE_MULTI_PURPOSE_TP is used then you at first are forced to use each owner of a FrTpConnection that is shown in the GUI of the configuration tool. You can change this by disabling a particular connection owner by these means:

1. Delete the name of its include header in the GUI.
2. Delete the names of all function prototypes of the services listed above in the GUI.

You must not delete the name of the owner structure itself because it is used as an AUTOSAR Short-Name in the EcuC-file of your configuration.

5.3.2 Services that are MICROSAR extensions

In case of incompatibilities of configuration and embedded code then the function `EcuM_BswErrorHook()` is called in the context of FrTp_Init():

Component	API
AUTOSAR EcuM	`EcuM_BswErrorHook(uint16 moduleId, uint8 errorCode)`

5.4 Callback Functions

This chapter describes the callback functions that are implemented by the Tp_Iso10681 and can be invoked by other modules. The prototypes of the callback functions are declared in the header file FrTp_Cbk.h by the Tp_Iso10681.

5.4.1 FrTp_TxConfirmation

FrTp_TxConfirmation()

Prototype	
void **FrTp_TxConfirmation** (PduIdType FrIfTxPduId)	
Parameters [in/out/both]	
PduIdType [in]	This parameter identifies the FrIf-Tx-Pdu assigned to the Tp_Iso10681 that has been transmitted successfully on the FlexRay-bus by the FrIf.
Return code	
void	n/a
Service ID	
Service ID	0x40
Functional Description	
This method has to be called after the transmission of all FrIf-Tx-Pdus assigned to the Tp_Iso10681.	
Preconditions	
FrTp_Init() has to be called prior to any calls to this method, i.e. the Tp_Iso10681 has to be initialized. Otherwise an error will be indicated to the DET.	
Postconditions	
After the call to this method the Tp_Iso10681 regards all other Tx-Pdus of the affected channel as transmitted. Of course only these Tx-Pdus are affected that have been requested by the Tp_Iso10681 to be transmitted by the FrIf in the current FlexRay-cycle or this.	
Particularities and Limitations	
During its runtime this method temporarily locks interrupts using the API-calls of the AUTOSAR BSW Scheduler (SchM).	
Call context	
This function can be called at any time after Tp_Iso10681-initialization.	

5.4.2 FrTp_RxIndication

FrTp_RxIndication()

Prototype	
void **FrTp_RxIndication**(PduIdType FrRxPduId, const PduInfoType* FrIf_PduInfoPtr)	
Parameters [in/out/both]	
PduIdType [in]	This parameter indicates to the Tp_Iso10681 which Rx-Pdu has been received successfully by the underlying FlexRay-Interface component.
const PduInfoType* [in]	This parameter points to a C-struct of type PduInfoType. The C-struct contains a pointer to the Rx-Pdu received and a value of type PduLengthType that indicates the length of the Pdus payload.
Return code	
void	n/a

Service ID	
Service ID	0x42
Functional Description	
This method is a callback that is called by the FrIf in case an Rx-Pdu assigned to the Tp_Iso10681 has been received successfully by the FrIf. The method copies the data pointed to by the member DataPtr of the PduInfoStruct to a Tp_Iso10681-internal buffer or directly to the Rx-buffer of the application. For this the member PduLength of the PduInfoStruct is used. This is done depending on the frame-type received.	
Preconditions	
FrTp_Init() has to be called prior to any calls to this method, i.e. the Tp_Iso10681 has to be initialized. Otherwise an error will be indicated to the DET.	
Postconditions	
n/a	
Particularities and Limitations	
During its runtime this method temporarily locks interrupts using the API-calls of the AUTOSAR BSW Scheduler (SchM).	
Call context	
This function can be called at any time after Tp_Iso10681-initialization.	

5.4.3 FrTp_TriggerTransmit

FrTp_TriggerTransmit()

Prototype	
Std_ReturnType **FrTp_TriggerTransmit** (PduIdType FrIfTxPduId, PduInfoType* FrTpTxPduInfoPtr)	
Parameters [in/out/both]	
PduIdType FrIfTxPduId [in]	This parameter contains the Id of the Tx-Pdu (FrIf-L-PduId of the FrIf) whose transmission was requested and whose payload-data shall be provided by the Tp_Iso10681 to the FrIf.
PduInfoType* FrTpTxPduInfoPtr [in]	Pointer to PduInfoType-structure containing a pointer to memory of the FrIf (or of the Fr-driver) the Tp_Iso10681 shall copy the payload and containing the information about the length of the Tp_Iso10681-frame.
Return code	
Std_ReturnType	E_OK: The request has been accepted E_NOT_OK: The request has not been accepted, e. g. parameter check has failed or cancellation is requested.
Service ID	
Service ID	0x41
Functional Description	
This method is called by the FrIf-Job-Execution in order to retrieve the payload-data of a FrIf-Tx-Pdu assigned to and previously transmitted by the Tp_Iso10681 via *FrIf_Transmit()*. This function copies the complete FlexRay TP-frame to a memory-area pointed to by the FrIf.	

Preconditions
FrTp_Init() has to be called prior to any calls to this method, i.e. the Tp_Iso10681 has to be initialized. Otherwise an error will be indicated to the DET. The transmission of the Tx-Pdu indicated by parameter FrIfTxPduId must have been requested by the Tp_Iso10681 in a preceding call to the FrIf-method FrIf_Transmit() whereas the latter call has to return E_OK in this case.
Postconditions
FrTp_TxConfirmation() has to be called by the FrIf in order to finalize the transmission of the affected FrIf-Tx-Pdu assigned to the Tp_Iso10681.
Particularities and Limitations
During its runtime this method temporarily locks interrupts using the API-calls of the AUTOSAR BSW Scheduler (SchM). This method is optional. It can be disabled by GUI switch 'Disable Decoupled Transmit' in case you plan to solely use FrIf-Pdus with transmission type 'immediate' in future.
Call context
This function can be called at any time after Tp_Iso10681-initialization. The only caller is the FrIf-Job-Execution which is executed in the context of the FlexRay-CCs timer-interrupt.

5.5 Services provided by Tp_Iso10681

The Tp_Iso10681 provides services solely to the PduR which is the only software unit "above" the Tp_Iso10681. Please refer to document [4] on how and when *PduR_FrTpRxIndication()*, *PduR_FrTpTxConfirmation()*,*PduR_CopyRxData()*, *PduR_CopyTxData()* are called by the Tp_Iso10681.

Although these services are only used in a AUTOSAR 3.0 compliant way the interpretation of certain NofityResultCodes is different, because the component implements the FrTp according to ISO10681-2 internally which has different error and success situations as well as a different buffer-handling regarding the buffer-sizes and the frequentness of calls to get Rx-buffer or Tx-data.

These are the differences and mappings between Tp_Iso10681 and AUTOSAR FrTp:

ISO 10681-2 return code	AUTOSAR NotifResultCode
C_ML_MISMATCH	As there is no appropriate AUTOSAR return code E_NOT_OK will be issued to *PduR_FrTpRxIndication()*.
C_ABORT	As there is no appropriate AUTOSAR return code E_NOT_OK will be issued to *PduR_FrTpTxConfirmation()*.
C_WRONG_BP	As there is no appropriate AUTOSAR return code E_NOT_OK will be issued to *PduR_FrTpRxIndication()*. By navigating the call-stack internally the code NTFRSLT_E_FR_INVALID_BS can be found for better error-analysis.
C_TX_ON	*FrTp_Transmit()* will issue the AUTOSAR return code E_NOT_OK in this case.

Table 5-2 Mappings from ISO10681-2 return codes to AUTOSAR FrTp 3.0 return codes.

6 Troubleshooting

6.1 Embedded Code

Symptom	Solution
FrTp **frames** / Pdus on the FlexRay bus seem to be **inconsistent**, e.g.: FPL (Frame Payload Length) is 4 bytes, ML (Message Length) is 1 byte - i.e. both values are inconsistent in relation to each other already - but in addition no payload is contained in the frame at all!	The frame has been ported from a CAN database and still has a length of 8 bytes which is not suitable for FrTp communication. Change the value to a suitable value, e.g. at least 10 bytes.
An FrTp **transmission / reception stops** that comes from or goes to a Gateway-ECU before the data is transferred completely.	Check the exchanging of Rx- or Tx-buffer-on the Gateway-ECU. Often the PduR or the transport-layer on the other bus-system has configurative problems (e.g. timing) that lead to the stopping of the Rx or Tx processes on the particular other bus-system.
The FrTp on a gateway Ecu **stops transmitting**	Check the Std_ReturnType code of the callback PduR_FrTpTxConfirmation() or (better:) the FrTp-internal result of type FrTp_NotifResultType that is used in FrTp-function FrTpIso_TxInit().
The FrTp on a gateway Ecu **stops receiving** (i.e. another <Bus>Tp stops sending (e.g. CanTp)	Check the Std_ReturnType code of the callback PduR_FrTpRxIndication() or (better:) the FrTp-internal result of type FrTp_NotifResultType that is used in FrTp-function FrTpIso_RxInit().
In lines where the function-like macro **FrTpIso_IsDecoupPdu()** is used the compiler issues warnings that "the condition is always true" or that code is "unreachable code".	In case of the warnings described the configuration of the Tp_Iso10681
The **size of the binary** (respectively of the Tp_Iso10681 object file) suddenly **increases** after the change of optimization switches of the compiler when using **post build** configuration.	In case you want to use the configuration variant post build together with a Greenhills® compiler on platform NEC® V850™ and compatible then you should take attention on the following compiler switch combination: In case you want to use option -pack=1 to reduce code size you should always use -misalign_pack at the same time in order to prevent a strong increase of the size of the static code of the Tp_Iso10681.
In case of the parameter E_NOT_OK given to PduR_FrTp[RxIndication | TxConfirmation] the **detailed, internal return code** might help.	Using the call-stack, try to navigate to the Tp_Iso10681-functions FrTpIso_[Rx | Tx]Init. There the internal code starting with token NTFRSLT_ tells you, what problem has happened when [receiving | transmitting] TP-data.

Table 6-1 Embedded Troubleshooting

7 AUTOSAR Standard Compliance

7.1 Deviations

The following features are not supported or deviate from the AUTOSAR SWS FrTp 4.0.3:

Category	Description	Version
Functional	“Acknowledgement and Retry” is not supported.	4.0.3
Functional	“FrIf Retry” i.e. retry of calls to FrIf_Transmit() after FrIf errors is not supported.	4.0.3
Functional	The FrTp does not implement the buffer handling of AUTOSAR SWS FrTp 4.0.3 (Version 4.0.3 specifies using parameters FrTpTimeBuffer and FrTpMaxFcWait as a timing basis for data exchange with the PduR. Instead the FrTp tries to copy Rx- or Tx-data for a duration defined by parameters FrTpTimeBr and FrTpTimeCs as specified by version 4.2.1).	4.0.3 < 4.2.1
Config	Parameter FrTpMaxFCWait of container FrTpConnectionControl: The full range of values: [0..255] of that parameter is not supported (Instead the FrTp supports a range of [0..254]).	4.0.3
Config	Parameter FrTpTimeBuffer of container FrTpConnectionControl: This parameter is ignored (Instead the FrTp supports the asking for Tx-data or Rx-buffer at each call to FrTp_MainFunction() as described in AUTOSAR SWS FrTp 4.2.1).	4.0.3
Config	Parameter FrTpMaxFrIf of container FrTpConnectionControl: This parameter is ignored because the related feature is not implemented. In addition the full range of values: [0..255] of that parameter is not supported (Instead the FrTp supports a range of [1..255]).	4.0.3
Config	FrTpTxPdu The maximum number of containers of type FrTpTxPdu is not supported (Instead the FrTp supports 254 Tx-Pdus).	4.0.3
Config	FrTpTxPduPool The maximum number of containers of type FrTpTxPduPool is not supported (Instead the FrTp supports 254 Tx-Pdu-Pools).	4.0.3
Config	FrTpRuntime The maximum number of runtime ressources of type FrTpChannel is not supported (Instead the FrTp supports 254 FrTpChannels).	4.0.3

Table 7-1 Deviations from AUTOSAR SWS FrTp 4.0.3

8 Glossary and Abbreviations

8.1 Glossary

Term	Description
EAD	Embedded Architecture Designer; generation tool for MICROSAR components
GENy	Generation tool for AUTOSAR3 MICROSAR components
Cfg5	Generation tool for AUTOSAR4 MICROSAR components

Table 8-1 Glossary

8.2 Abbreviations

Abbreviation	Description
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
BC	Bandwidth Control
BSW	Basis Software
DEM	Diagnostic Event Manager
DET	Development Error Tracer
EAD	Embedded Architecture Designer
ECU	Electronic Control Unit
FC	Flow Control
FPL	Frame Payload Length
HIS	Hersteller Initiative Software
ISR	Interrupt Service Routine
MICROSAR	Microcontroller Open System Architecture (the Vector AUTOSAR solution)
ML	Message Length
MNPC	Maximum Number of Pdus per Cycle
PDU	Protocol Data Unit
PB	Post Build
PCI	Protocol Control Information
PPort	Provide Port
RPort	Require Port
RTE	Runtime Environment
SRS	Software Requirement Specification

SWC	Software Component
SWS	Software Specification
STF	Start Frame
STFsegm, segmented STF	STF that contains the first data bytes of a segmented transfer
STFunseg, unsegmented STF	STF that contains the complete payload of an unsegmented transfer

Table 8-2 Abbreviations

9 Contact

Visit our website for more information on

- News
- Products
- Demo software
- Support
- Training data
- Addresses

www.vector-informatik.com