MICROSAR SAE J1939 Transport Layer

Technical Reference

Version 1.2.1

Authors	Martin Schlodder, Thomas Albrecht, Matthias Müller
Status	Released

Document Information

History

Author	Date	Version	Remarks
Thomas Albrecht	2013-09-25	0.1.0	Created initial version
Martin Schlodder	2014-06-20	0.2.0	Updated
Martin Schlodder	2014-11-28	1.0.0	First released version
Martin Schlodder	2015-01-23	1.1.0	ETP support added
Martin Schlodder	2015-09-04	1.1.1	Restrictions of parallel transmissions
Martin Schlodder	2016-11-24	1.2.0	FPP support added
Matthias Müller	2017-02-24	1.2.1	Introduced runtime errors

Reference Documents

No.	Source	Title	Version
[1]	AUTOSAR	AUTOSAR_SWS_SAEJ1939TransportLayer.pdf	4.2.1
[2]	AUTOSAR	AUTOSAR_SWS_DefaultErrorTracer.pdf	4.2.1
[3]	AUTOSAR	AUTOSAR_SWS_BSWGeneral.pdf	4.2.1
[4]	AUTOSAR	AUTOSAR_TR_BSWModuleList.pdf	4.2.1
[5]	Vector	TechnicalReference_PostBuildLoadable.pdf	1.0.0
[6]	ISO 11783	ISO_FDIS_11783_3.pdf	2013
[7]	NMEA 2000	NMEA2000_Main.pdf	1.210

Caution
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

Illustrations

Tables

1 Component History

The component history gives an overview over the important milestones that are supported in the different versions of the component.

Component Version	New Features
[0.1.0]	Initial Version (BETA)
[0.2.0]	Implemented timeout and state handling, and parameter checks
[0.3.0]	Added support for post-build configuration
[1.0.0]	Added support for variant handling; component released
[1.1.0]	ETP support added
[1.2.0]	Receive direct frames with less than 8 bytes
[1.3.0]	FPP support added

Table 1-1 Component History

2 Introduction

This document describes the functionality, API, and configuration of the AUTOSAR BSW module J1939Tp as specified in [1], with additional support for the ISO 11783 extended transport protocol (ETP, [6]) and the NMEA 2000 fast packet protocol (FPP, [7]).

Supported AUTOSAR Release:	4	
Supported Configuration Variants:	pre-compile, post-build-loadable, post-build-selectable	
Vendor ID:	J1939TP_VENDOR_ID	30 decimal (= Vector-Informatik, according to HIS)
Module ID:	J1939TP_MODULE_ID	37 decimal (according to [4])

The MICROSAR SAE J1939 Transport Layer implements the transport protocol defined by the SAE in the document J1939-21, icluding both the broadcast (BAM) and the point-to-point (CMDT, RTS/CTS) variant, and additionally the extended transport protocol (ETP) of ISO 11783-3 and the fast packet protocol of NMEA 2000.

In the AUTOSAR architecture, the SAE J1939 Transport Layer interacts with the CanIf to transmit and receive the TP.DT and TP.CM messages, as well as direct messages in case the total message size drops below 9 bytes. And it interacts with the PduR to transmit and receive the assembled (large) messages.

J1939Tp makes heavy use of the Meta Data support, which was introduced with AUTOSAR 4.1.1. This allows for flexible handling of the protocols, such that an N-SDU can be transmitted or received via BAM, CMDT, ETP, and directly, depending on the current destination address and length. FPP messages, in contrast, are always transmitted with the same protocol, even when they consist only of one segment. FPP supports both broadcast and destination specific transmission.

2.1 Architecture Overview

The following figure highlights the position of the J1939Tp in the AUTOSAR architecture.

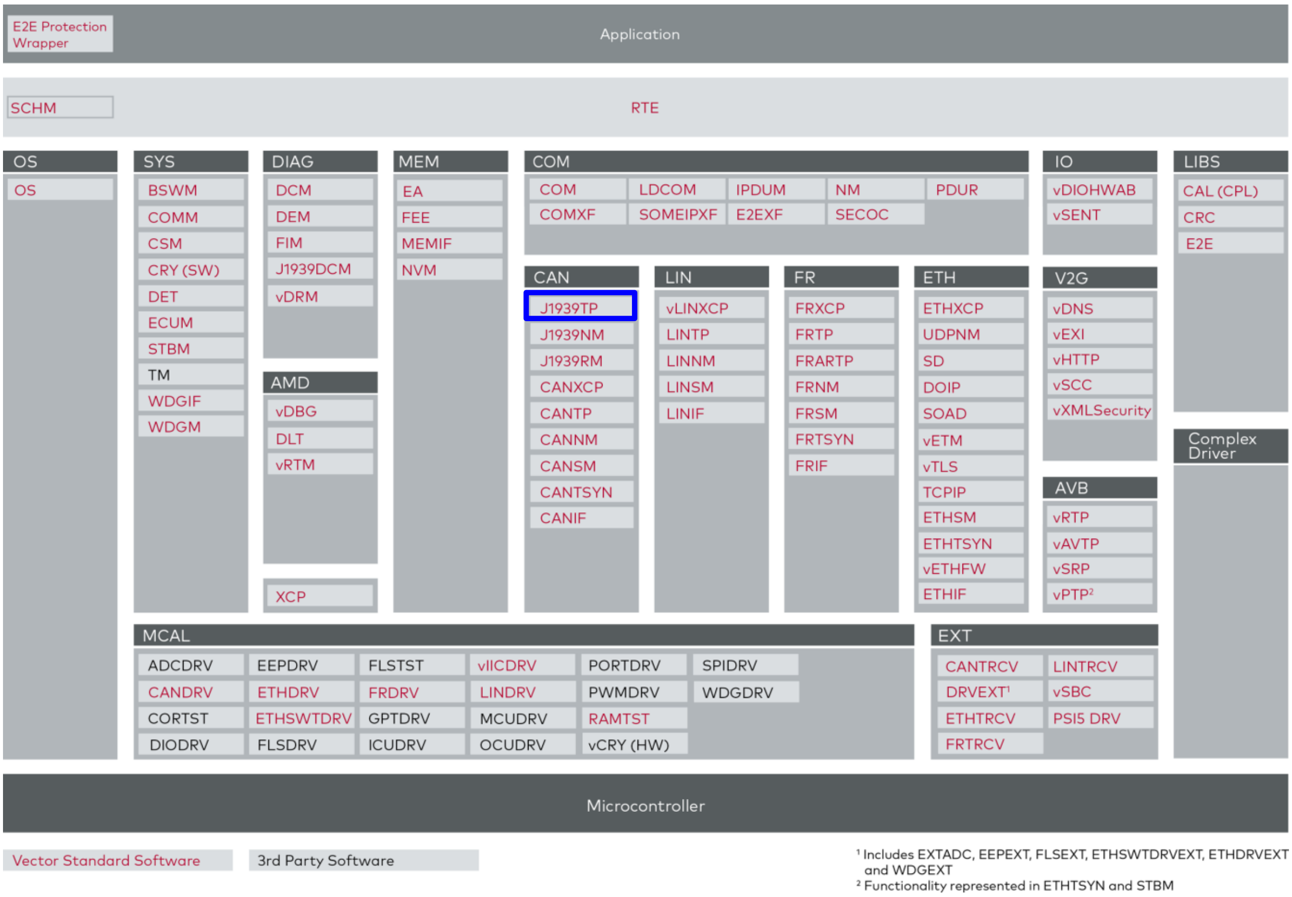

Figure 2-1 AUTOSAR 4.2 Architecture Overview

The next figure shows the interfaces to adjacent modules of the J1939Tp. These interfaces are described in chapter 5.

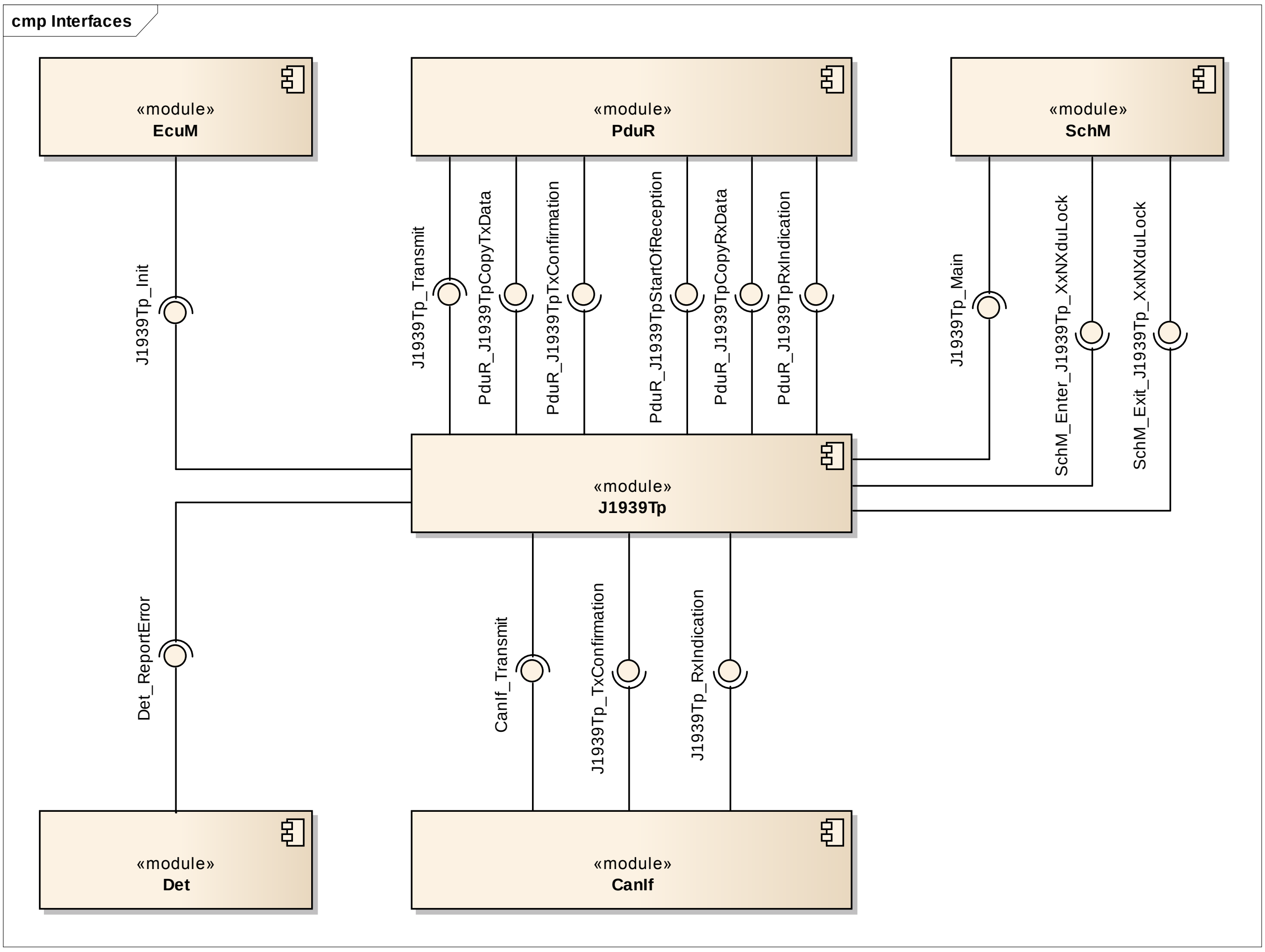

Figure 2-2 Interfaces to adjacent modules of the J1939Tp

3 Functional Description

3.1 Features

The features listed in the following tables cover the complete functionality specified for the J1939Tp.

The AUTOSAR standard functionality is specified in [1], the corresponding features are listed in the tables

> Table 3-1 Supported AUTOSAR standard conform features

> Table 3-2 Not supported AUTOSAR standard conform features

Vector Informatik provides further J1939Tp functionality beyond the AUTOSAR standard. The corresponding features are listed in the table

> Table 3-3 Features provided beyond the AUTOSAR standard

The following features specified in [1] are supported:

Supported AUTOSAR Standard Conform Features
Initialization and shutdown of the module
Timing supervision
Reception and transmission of segmented messages
Reception and transmission of direct messages (current size ≤ 8)
TP connections with multiple receivers using BAM
TP connections with a single receiver using CMDT
Meta data handling

Table 3-1 Supported AUTOSAR standard conform features

3.1.1 Deviations from AUTOSAR 4.2.1

The following features specified in [1] are not supported:

Not Supported AUTOSAR Standard Conform Features
Dynamic block calculation
Retry support
Cancellation support

Table 3-2 Not supported AUTOSAR standard conform features

3.1.1.1 Dynamic Block Calculation

Dynamic block calculation encompasses the calculation of the “maximum number of packets” value of the TP.CM_RTS message and the adaptation of the “number of packets” value of the TP.CM_CTS according to the currently available buffer and to the value provided via the J1939Tp_ChangeParameter API.

Affected AUTOSAR specification items: SWS_J1939Tp_00165, SWS_J1939Tp_00180, SWS_J1939Tp_00206, SWS_J1939Tp_00207, SWS_J1939Tp_00208, SWS_J1939Tp_00210, SWS_J1939Tp_00211, SWS_J1939Tp_00212, SWS_J1939Tp_00226, SWS_J1939Tp_00227, SWS_J1939Tp_00229, ECUC_J1939Tp_00187, ECUC_J1939Tp_00188, ECUC_J1939Tp_00191, ECUC_J1939Tp_00190

3.1.1.2 Retry Support

Retry support encompasses the handling of TP transmission or reception errors by requesting retransmission of already transmitted TP.DT frames via TP.CM_CTS, and by sending the requested TP.DT frames again.

Affected AUTOSAR specification items: SWS_J1939Tp_00217, SWS_J1939Tp_00220, SWS_J1939Tp_00221, SWS_J1939Tp_00222, ECUC_J1939Tp_00185, ECUC_J1939Tp_00193

3.1.1.3 Cancellation Support

Cancellation support encompasses the termination of TP transmissions or receptions when the upper layer calls the J1939Tp_CancelTransmit or J1939Tp_CancelReceive APIs.

Affected AUTOSAR specification items: SWS_J1939Tp_00040, SWS_J1939Tp_00048

3.1.2 Additions / Extensions

The following features are provided beyond the AUTOSAR standard:

Features Provided Beyond The AUTOSAR Standard
Extended Error Reporting
ETP Support
FPP Support

Table 3-3 Features provided beyond the AUTOSAR standard

3.1.2.1 Extended Error Reporting

The J1939Tp reports additional development errors that are not specified by AUTOSAR. See Table 3-5 in section 3.6.1.

3.1.2.2 ETP Support

The J1939Tp supports the Extended Transport Protocol, defined in [6].

3.1.2.3 FPP Support

The J1939Tp supports the Fast Packet Protocol, defined in [7].

3.2 Initialization

The J1939Tp uses a global state (J1939Tp_ModuleInitialized) to determine whether the module is initialized and operational. This state is initially set to J1939TP_UNINIT. If initialization by startup code is not supported, the initialization routine should call J1939Tp_InitMemory() to set the global state to J1939TP_UNINIT.

By calling J1939Tp_Init(), the J1939Tp module is set to the state J1939TP_INIT, and internal states are set to their initial states. The module is now operational.

To stop the J1939Tp module, J1939Tp_Shutdown() may be called, which sets the global state to J1939TP_UNINIT again.

3.3 States

The J1939Tp module has a global state, and separate states for each Tx and Rx N-SDU (communication with upper layers) and each Tx N-PDU (communication with CanIf).

3.3.1 Global State

The global state is switched by the services J1939Tp_InitMemory(), J1939Tp_Init(), and J1939Tp_Shutdown().

In the state J1939TP_UNINIT, all services of J1939Tp return immediately. If they have a return value, an error (typically E_NOT_OK) is returned.

In the state J1939TP_INIT, services are operational.

3.3.2 Tx N-PDU State

Each transmitted N-PDU has its own state to ensure that a value provided to CanIf is not overwritten with a new one before the CanIf was able to transmit it on the CAN bus (e.g. a TP.CM_BAM overwriting a preceding TP.CM_CTS). This state is protected by an exclusive area.

The Tx N-PDU state depends on the J1939Tp_TxConfirmation() being called after each call to CanIf_Transmit(). Because this is not always the case, the J1939Tp monitors this state and resets it after a timeout configurable via “Tx Conf Timeout”, assuming the N-PDU was not transmitted.

3.3.3 Tx/Rx N-SDU State

Each transmitted and received N-SDU has its own state machine which represents the state of a transport layer session. J1939Tp uses separate session states for direct transfer and for transfer using BAM, CMDT, or ETP, so that one N-SDU can use any of these protocols (though not at the same time).

3.4 Main Function

The J1939Tp_MainFunction() is used by the J1939Tp module to supervise all kinds of timing. Therefore, it is essential that this main function is called with the timing configured via “Main Function Period”.

If the timing is not exact, a typical error will be that some BAM frames are sent with noticeably less than or far more than 50ms.

3.5 Parallel Transmission

In principle, J1939Tp supports transmission of direct frames, BAM, CMDT, ETP, and FPP in parallel. Of course, there are the restrictions imposed by the usage of CAN identifiers by J1939Tp. Because of these, only one BAM connection can be open at any time for each source address, and CMDT connections can only coexist if they have different source and destination addresses, which applies also to ETP and FPP connections.

An additional restriction is due to the way J1939Tp transmits CMDT messages, where each TP.DT frame is sent from the transmit confirmation of the last frame. Because of this, and if the receiver responds also immediately to the last TP.DT frame, CMDT messages can block the bus completely for some time. This, in turn, may lead to timeout of BAM and other CMDT connections, and also of simple messages and ETP connections, depending on the relative priority of the CAN IDs and the allocation of CAN hardware buffers.

3.6 Error Handling

3.6.1 Development and Runtime Error Reporting

By default, development errors are reported to the DET using the service `Det_ReportError()` as specified in [2], if development error reporting is enabled (i.e. pre-compile parameter `J1939TP_DEV_ERROR_REPORT == STD_ON`). Runtime errors are reported on the same way, if runtime error reporting is enabled (i.e. pre-compile parameter `J1939RM_RUNTIME_ERROR_REPORT == STD_ON`).

The reported J1939Tp module ID is 37.

The reported service IDs identify the services which are described in section 5.1 as well as the callback functions described in section 5.3. The following table presents the service IDs and the related services and callback functions:

Service ID		Service
0x01	J1939TP_SID_INIT	J1939Tp_Init()
0x02	J1939TP_SID_SHUTDOWN	J1939Tp_Shutdown()
0x03	J1939TP_SID_GETVERSIONINFO	J1939Tp_GetVersionInfo()
0x04	J1939TP_SID_MAINFUNCTION	J1939Tp_MainFunction()
0x05	J1939TP_SID_TRANSMIT	J1939Tp_Transmit()
0x08	J1939TP_SID_CHANGEPARAMETER	J1939Tp_ChangeParameter()
0x09	J1939TP_SID_CANCELTRANSMIT	J1939Tp_CancelTransmit()
0x0A	J1939TP_SID_CANCELRECEIVE	J1939Tp_CancelReceive()
0x40	J1939TP_SID_TXCONFIRMATION	J1939Tp_TxConfirmation()
0x42	J1939TP_SID_RXINDICATION	J1939Tp_RxIndication()
0x80	*J1939TP_SID_INITMEMORY*	J1939Tp_InitMemory()

Table 3-4 Service IDs

Note
The service IDs above 0x80 are not specified by AUTOSAR.

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

Error Code		Description
0x01	J1939TP_E_UNINIT	API service called before J1939Tp_Init or after J1939Tp_Shutdown

Error Code		Description
0x02	J1939TP_E_REINIT	J1939Tp_Init called after J1939Tp_Init and before J1939Tp_Shutdown
0x03	J1939TP_E_INIT_FAILED	J1939Tp_Init called with invalid init structure
0x10	J1939TP_E_PARAM_POINTER	API service called with null pointer
0x11	J1939TP_E_INVALID_PDU_SDU_ID	API service called with wrong ID
0x80	*J1939TP_E_INVALID_LENGTH*	Invalid length of received or transmitted N-SDU
0x83	*J1939TP_E_INVALID_CHANGE_PARAM*	Invalid parameter argument to ChangeParameter
0x84	*J1939TP_E_INVALID_CHANGE_VALUE*	Invalid value argument to ChangeParameter
0x9B	*J1939TP_E_DUMMY_API*	A dummy API was called

Table 3-5 Development errors reported to DET

Note
The development error codes above 0x80 are not specified by AUTOSAR.

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

Error Code		Description
0x30	J1939TP_E_TIMEOUT_T1	Timeout occurred on receiver side after reception of an intermediate (E)TP.DT frame of a block or an FPP frame
0x31	J1939TP_E_TIMEOUT_T2	Timeout occurred on receiver side after transmission of a (E)TP.CM_CTS frame
0x32	J1939TP_E_TIMEOUT_T3	Timeout occurred on transmitter side after transmission of the last (E)TP.DT frame of a block
0x33	J1939TP_E_TIMEOUT_T4	Timeout occurred on transmitter side after reception of a (E)TP.CM_CTS(0) frame
0x34	J1939TP_E_TIMEOUT_TR	Timeout occurred on transmitter or receiver side while trying to send the next (E)TP.DT or (E)TP.CM frame
0x35	J1939TP_E_TIMEOUT_TH	Timeout occurred on receiver side while trying to send the next (E)TP.CM_CTS frame after a (E)TP.CM_CTS(0) frame
0x40	J1939TP_E_INVALID_TMS	Invalid value for “total message size” in received TP.CM/RTS frame
0x41	J1939TP_E_INVALID_TNOP	Value for “total number of packets” in received TP.CM/RTS frame does not match the “total message size”
0x42	J1939TP_E_INVALID_MNOP	Invalid value for “maximum number of packets” in received TP.CM/RTS frame

Error Code		Description
0x43	J1939TP_E_INVALID_PGN	Unexpected PGN in received TP.CM frame
0x44	J1939TP_E_INVALID_NOP	Invalid value for “number of packets” in received TP.CM/CTS frame
0x45	J1939TP_E_INVALID_NPN	Invalid value for “next packet number” in received TP.CM/CTS frame
0x46	J1939TP_E_INVALID_CAR	Invalid value for “connection abort reason” in received TP.ConnAbort frame
0x47	J1939TP_E_INVALID_SN	Unexpected sequence number in received TP.DT frame
0x81	*J1939TP_E_INVALID_SA*	Invalid source address in received frame
0x82	*J1939TP_E_INVALID_DA*	Invalid destination address in received frame
0x85	*J1939TP_E_UNTIMELY_RTS*	TP.CM_RTS frame received while a CMDT transmission was still active on the same connection
0x86	*J1939TP_E_IGNORED_CTS*	Ignored untimely TP.CM_CTS frame
0x87	*J1939TP_E_IGNORED_EOMACK*	Ignored untimely TP.CM_EndOfMsgAck frame
0x88	*J1939TP_E_IGNORED_ABORT*	Ignored untimely TP.ConnAbort frame
0x89	*J1939TP_E_NO_CONNECTION*	TP.CM_RTS frame received, but no free connection found
0x8A	*J1939TP_E_INVALID_DLC*	Invalid length of received N-PDU
0x8B	*J1939TP_E_INVALID_ATMS*	Value for “total message size” in received TP.CM_EOMAck frame differs from the same value in TP.CM_RTS
0x8C	*J1939TP_E_INVALID_ATNOP*	Value for “total number of packets” in received TP.CM_EOMAck frame differs from the same value in TP.CM_RTS
0x90	*J1939TP_E_INVALID_NBT*	Invalid value for “number of bytes to transfer” in received ETP.CM_RTS frame
0x91	*J1939TP_E_INVALID_ANBT*	Value for “number of bytes transferred” in received ETP.CM_EOMA frame differs from the same value in ETP.CM_RTS
0x92	*J1939TP_E_UNEXPECTED_ECTS*	Unexpected ETP.CM_CTS frame (wrong PGN)
0x93	*J1939TP_E_INVALID_DPO*	Invalid value for “data packet offset” in received ETP.CM_DPO frame
0x94	*J1939TP_E_INVALID_NPO*	Invalid value for “number of packets to which to apply the offset” in received ETP.CM_DPO frame
0x95	*J1939TP_E_UNEXPECTED_DPO*	Unexpected ETP.CM_DPO frame (wrong PGN)
0x96	*J1939TP_E_INVALID_CONTROL_BYTE*	(E)TP.CM used with wrong addressing or invalid control byte, e.g. TP.CM_BAM with DA != 0xFF or ETP.CM with an unknown CB
0x97	*J1939TP_E_TIMEOUT_TXCONF*	Timeout of transmission confirmation callback
0x98	*J1939TP_E_UNTIMELY_BAM*	TP.CM_BAM frame received while a BAM reception was still active on the same connection
0x99	*J1939TP_E_EARLY_EOMACK*	TP.CM_EndOfMsgAck frame received before all data

Error Code		Description
		was transmitted
0x9A	*J1939TP_E_INVALID_SIZE*	Invalid length of received N-SDU
0x9C	*J1939TP_E_IGNORED_ECTS*	Ignored untimely ETP.CM_CTS frame
0x9D	*J1939TP_E_INVALID_SC*	Unexpected sequence counter of received FF or AF frame
0x9E	*J1939TP_E_INVALID_FC*	Unexpected frame counter of received AF frame
0x9F	*J1939TP_E_UNTIMELY_FF*	First frame received while an FPP reception was still active on the same connection
0xA0	*J1939TP_E_TIMEOUT_FP*	Timeout on transmitter side while trying to send the next FPP frame

Table 3-6 Runtime errors reported to DET

Note
The runtime error codes above 0x80 are not specified by AUTOSAR.

4 Integration

This chapter gives necessary information for the integration of the MICROSAR J1939Tp into an application environment of an ECU.

4.1 Scope of Delivery

The delivery of the J1939Tp contains the files which are described in the sections 4.1.1 and 4.1.2:

4.1.1 Static Files

File Name	Description
J1939Tp.c	Implementation of the J1939Tp module
J1939Tp.h	Main header of the J1939Tp module
J1939Tp_Cbk.h	Callback header of the J1939Tp module
J1939Tp_Types.h	Global types header of the J1939Tp module
J1939Tp_Int.h	Internal header of the J1939Tp module
J1939Tp_Bam.c	Implementation of the BAM sub-module of the J1939Tp module
J1939Tp_Bam.h	Header of the BAM sub-module of the J1939Tp module
J1939Tp_Cmdt.c	Implementation of the CMDT sub-module of the J1939Tp module
J1939Tp_Cmdt.h	Header of the CMDT sub-module of the J1939Tp module
J1939Tp_Direct.c	Implementation of the direct sub-module of the J1939Tp module
J1939Tp_Direct.h	Header of the direct sub-module of the J1939Tp module
J1939Tp_Etp.c	Implementation of the ETP sub-module of the J1939Tp module
J1939Tp_Etp.h	Header of the ETP sub-module of the J1939Tp module
J1939Tp_Fpp.c	Implementation of the FPP sub-module of the J1939Tp module
J1939Tp_Fpp.h	Header of the FPP sub-module of the J1939Tp module

Table 4-1 Static files

4.1.2 Dynamic Files

The dynamic files are generated by DaVinci Configurator.

File Name	Description
J1939Tp_Cfg.h	Generated header file of J1939Tp containing pre-compile switches and providing symbolic defines
J1939Tp_Cfg.c	Generated source file of J1939Tp containing pre-compile time configurable parameters
J1939Tp_Lcfg.h	Generated header file of J1939Tp containing link time configurable preprocessor symbols

File Name	Description
J1939Tp_Lcfg.c	Generated source file of J1939Tp containing link time configurable parameters
J1939Tp_PBcfg.h	Generated header file of J1939Tp containing post-build time configurable preprocessor symbols
J1939Tp_PBcfg.c	Generated source file of J1939Tp containing post-build time configurable parameters

Table 4-2 Generated files

4.2 Critical Sections

The J1939Tp module uses three critical sections to protect the state machines for the Tx N-PDUs, the Tx N-SDUs and the Rx N-SDUs:

- J1939Tp_TxNPduLock
- J1939Tp_TxNSduLock
- J1939Tp_RxNSduLock

All these critical sections have a very short locking time, and do not cover any function calls.

5 API Description

For an interfaces overview please see Figure 2-2.

5.1 Services provided by J1939Tp

This section describes the service functions that are implemented by the J1939Tp and can be invoked by other modules. The prototypes of the service functions are provided in the header file `J1939Tp.h`.

5.1.1 J1939Tp_InitMemory

Prototype	
`void` **`J1939Tp_InitMemory`** `(void)`	
Parameter	
None	
Return code	
void	None
Functional Description	
Sets the global J1939Tp state to uninitialized.	
Particularities and Limitations	
This function should be used if the J1939Tp is not initialized by startup code.	
Call Context	
Only to be called from initialization code.	

Table 5-1 J1939Tp_InitMemory

5.1.2 J1939Tp_Init

Prototype	
`void` **`J1939Tp_Init`** `(const J1939Tp_ConfigType *config)`	
Parameter	
config	Pointer to configuration data structure.
Return code	
void	None
Functional Description	
Initializes the J1939Tp module.	
Particularities and Limitations	
The config parameter is only required if the configuration is variant or changed at post-build time.	
Call Context	
Only to be called from task level.	
Preconditions	

The module must be in the uninitialized state.

Table 5-2 J1939Tp_Init

5.1.3 J1939Tp_Shutdown

Prototype	
void **J1939Tp_Shutdown** (void)	
Parameter	
None	
Return code	
void	None
Functional Description	
Shuts the J1939Tp module down.	
Particularities and Limitations	
The module is not truly shut down before all services and callback functions have terminated.	
Call Context	
Only to be called from task level.	
Preconditions	
The module must be in the initialized state.	

Table 5-3 J1939Tp_Shutdown

5.1.4 J1939Tp_MainFunction

Prototype	
void **J1939Tp_MainFunction** (void)	
Parameter	
None	
Return code	
void	None
Functional Description	
Main function of the J1939Tp. Used for scheduling purposes and timeout supervision.	
Particularities and Limitations	
The main function must be called cyclically with a timing corresponding to the configured Main Function Period.	
Call Context	
Only to be called from task level.	

Table 5-4 J1939Tp_MainFunction

5.1.5 J1939Tp_GetVersionInfo

Prototype	
`void` **`J1939Tp_GetVersionInfo`** `(Std_VersionInfoType *VersionInfo)`	
Parameter	
VersionInfo	Pointer to the location where the version information of J1939Tp shall be stored.
Return code	
void	None
Functional Description	
Returns the version information of J1939Tp.	
Particularities and Limitations	
none	
Call Context	
May be called from interrupt or task level.	
Preconditions	
The VersionInfo parameter must not be NULL.	

Table 5-5 J1939Tp_GetVersionInfo

5.1.6 J1939Tp_Transmit

Prototype	
`Std_ReturnType` **`J1939Tp_Transmit`** `(PduIdType txSduId, const PduInfoType *pduInfoPtr)`	
Parameter	
txSduId	ID of the J1939Tp N-SDU to be transmitted. The available IDs are configured via J1939TpTxNSduId.
pduInfoPtr	Pointer to a structure with length and content of the J1939Tp N-SDU that shall be transmitted. The content of this structure is used to transfer addressing information and priority of N-SDU in the MetaData.
Return code	
Std_ReturnType	E_OK: The request has been accepted. E_NOT_OK: The request failed. This happens when a resource could not be allocated, e. g. when the requested transmission would use a channel that is currently active.
Functional Description	
Requests transmission of a J1939Tp N-SDU.	
Particularities and Limitations	
none	
Call Context	
May be called from interrupt or task level.	
Preconditions	

The pduInfoPtr parameter and its field SduDataPtr must not be NULL.

Table 5-6 J1939Tp_Transmit

5.1.7 J1939Tp_CancelTransmit

Prototype	
`Std_ReturnType` **`J1939Tp_CancelTransmit`** `(PduIdType id)`	
Parameter	
id	ID of the J1939Tp N-SDU to be canceled. The available IDs are configured via J1939TpTxNSduId.
Return code	
Std_ReturnType	E_OK: The request has been accepted. E_NOT_OK: The request failed. This happens when the provided N-SDU is currently not transmitted.
Functional Description	
Cancels the ongoing transmission of a J1939Tp N-SDU.	
Particularities and Limitations	
This function is not yet implemented, and returns always E_NOT_OK.	
Call Context	
May be called from interrupt or task level.	
Preconditions	
The N-SDU is currently being transmitted.	

Table 5-7 J1939Tp_CancelTransmit

5.1.8 J1939Tp_CancelReceive

Prototype	
`Std_ReturnType` **`J1939Tp_CancelReceive`** `(PduIdType id)`	
Parameter	
id	ID of the J1939Tp N-SDU to be canceled. The available IDs are configured via J1939TpRxNSduId.
Return code	
Std_ReturnType	E_OK: The request has been accepted. E_NOT_OK: The request failed. This happens when the provided N-SDU is currently not received.
Functional Description	
Cancels the ongoing reception of a J1939Tp N-SDU.	
Particularities and Limitations	
This function is not yet implemented, and returns always E_NOT_OK.	
Call Context	
May be called from interrupt or task level.	
Preconditions	

The N-SDU is currently being received.

Table 5-8 J1939Tp_CancelReceive

5.1.9 J1939Tp_ChangeParameter

Prototype	
Std_ReturnType **J1939Tp_ChangeParameter** (PduIdType id, TPParameterType parameter, uint16 value)	
Parameter	
id	ID of the N-SDU for which parameters should be changed. The available IDs are configured via J1939TpRxNSduId.
parameter	ID of parameter that should be changed.
value	New value for changed parameter.
Return code	
Std_ReturnType	E_OK: The request has been accepted. E_NOT_OK: The request failed. This happens when the provided parameter does not exist.
Functional Description	
Changes reception parameters of J1939Tp for a specific N-SDU.	
Particularities and Limitations	
This function is not yet implemented, and returns always E_NOT_OK.	
Call Context	
May be called from interrupt or task level.	

Table 5-9 J1939Tp_ChangeParameter

5.2 Services used by J1939Tp

The following table lists the services provided by other components, which are used by the J1939Tp. For details about prototype and functionality refer to the documentation of the providing component.

Component	API
CAN Interface	CanIf_Transmit
Default Error Tracer	Det_ReportError
PDU Router	PduR_J1939TpCopyRxData
	PduR_J1939TpCopyTxData
	PduR_J1939TpRxIndication
	PduR_J1939TpStartOfReception
	PduR_J1939TpTxConfirmation

Table 5-10 Services used by the J1939Tp

5.3 Callback Functions

This section describes the callback functions that are implemented by the J1939Tp and can be invoked by other modules. The prototypes of the callback functions are provided in the header file `J1939Tp_Cbk.h`.

5.3.1 J1939Tp_RxIndication

Prototype	
`void` **`J1939Tp_RxIndication`** `(PduIdType RxPduId, const PduInfoType *PduInfoPtr)`	
Parameter	
RxPduId	ID of the received N-PDU.
PduInfoPtr	Contains the length (SduLength) of the received N-PDU and a pointer to a buffer (SduDataPtr) containing the N-PDU and MetaData.
Return code	
void	None
Functional Description	
Indicates the reception of an N-PDU from the CanIf.	
Particularities and Limitations	
none	
Call Context	
May be called from interrupt or task level.	
Preconditions	
J1939Tp_RxIndication is not currently executed with the same RxPduId.	

Table 5-11 J1939Tp_RxIndication

5.3.2 J1939Tp_TxConfirmation

Prototype	
`void` **`J1939Tp_TxConfirmation`** `(PduIdType TxPduId)`	
Parameter	
TxPduId	ID of the N-PDU that has been transmitted.
Return code	
void	none
Functional Description	
Confirms the successful transmission of an N-PDU by the CanIf.	
Particularities and Limitations	
none	
Call Context	
May be called from interrupt or task level.	
Preconditions	

J1939Tp_RxIndication is not currently executed with the same TxPduId.

Table 5-12 J1939Tp_TxConfirmation

6 Configuration

6.1 Configuration Variants

The J1939Tp supports the configuration variants

- `VARIANT-PRE-COMPILE`
- `VARIANT-POST-BUILD-LOADABLE`
- `VARIANT-POST-BUILD-SELECTABLE`

The configuration classes of the J1939Tp parameters depend on the supported configuration variants. For their definitions please see the J1939Tp_bswmd.arxml file.

6.2 Post-Build Configuration

The configuration of post-build loadable is described in [5].

7 Glossary and Abbreviations

7.1 Glossary

Term	Description
DaVinci Configurator	Generation tool for MICROSAR components.

Table 7-1 Glossary

7.2 Abbreviations

Abbreviation	Description
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
BAM	Broadcast Announce Message, broadcast variant of SAE J1939 transport protocol
BSW	Basis Software
CMDT	Connection Mode Data Transfer, peer-to-peer variant of SAE J1939 transport protocol
DET	Default Error Tracer
ECU	Electronic Control Unit
ETP	ISO 11783 Extended Transport Protocol
FPP	NMEA2000 Fast Packet Protocol
I-PDU	PDU of the PDU Router, forms an N-SDU if handled by a TP layer
MICROSAR	Microcontroller Open System Architecture (Vector’s AUTOSAR solution)
N-PDU	PDU of the network layer, exchanged between J1939Tp and CanIf
N-SDU	SDU of the network layer, exchanged between J1939Tp and PduR
PDU	Protocol Data Unit, consisting of an SDU and control information
PPORT	Provide Port
RPORT	Require Port
RTE	Runtime Environment
SDU	Service Data Unit
SRS	Software Requirement Specification
SWS	Software Specification

Table 7-2 Abbreviations

8 Contact

Visit our website for more information on

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

www.vector.com