



SURFACE VEHICLE RECOMMENDED PRACTICE

J2534™-2/7_0500

JAN2022

Issued

2022-01

Pass-Thru Extended Feature - TP2.0

RATIONALE

Each of the features contained in the original SAE J2534-2 document (API version 04.04) is being published as a separate document for ease of maintenance and extensibility. At the same time, these features are being updated to be compatible with SAE J2534-1_0500 (API version 05.00) and to address feedback from the field.

TABLE OF CONTENTS

1.	SCOPE.....	3
2.	REFERENCES.....	3
2.1	Applicable Documents	3
2.1.1	SAE Publications.....	3
3.	DEFINITIONS	3
4.	ACRONYMS	4
5.	DOCUMENTATION CONVENTIONS.....	4
6.	PASS-THRU INTERFACE REQUIREMENTS.....	4
6.1	Specific Protocol Requirements.....	4
6.2	Pin Usage.....	5
6.3	Data Buffering	5
6.3.1	Message Reception	5
6.3.2	Message Transmission	6
6.4	Error Handling	6
6.4.1	Device Not Connected	6
6.4.2	Receive Buffer Overflow	6
6.4.3	Message Termination.....	6
6.4.4	Network Errors	6
6.4.5	Message Timeout.....	7
7.	WIN32 APPLICATION PROGRAMMING INTERFACE.....	7
7.1	API Changes	7
7.2	Message Structure	8
7.2.1	C/C++ Definition	8
7.2.2	Description of Structure Elements	8
7.2.3	Protocol Specific Details	10
7.2.4	Format Checks for Messages Passed to the API	10
7.2.5	Conventions for Returning Messages from the API.....	11

SAE Executive Standards Committee Rules provide that: "This report is published by SAE to advance the state of technical and engineering sciences. The use of this report is entirely voluntary, and its applicability and suitability for any particular use, including any patent infringement arising therefrom, is the sole responsibility of the user."

SAE reviews each technical report at least every five years at which time it may be revised, reaffirmed, stabilized, or cancelled. SAE invites your written comments and suggestions.

Copyright © 2022 SAE International

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of SAE.

TO PLACE A DOCUMENT ORDER: Tel: 877-606-7323 (inside USA and Canada)
Tel: +1 724-776-4970 (outside USA)
Fax: 724-776-0790
Email: CustomerService@sae.org
http://www.sae.org

SAE WEB ADDRESS:

Full document details: SAE J2534-2_7_5_00

For more information on this standard, visit
https://www.sae.org/standards/content/J2534-2/7_5_00

Documents

Document Sharing Hub

7.2.6	Indications	11
7.3	API Functions - Detailed Information	12
7.3.1	PassThruLogicalConnect	12
7.3.2	PassThruLogicalDisconnect	13
7.3.3	PassThruQueueMsgs	14
7.3.4	PassThruStartPeriodicMsg	15
7.3.5	PassThruIoctl	15
7.4	IOCTL Section	16
7.4.1	GET_CONFIG/SET_CONFIG	16
7.4.2	CLEAR_TX_QUEUE	17
7.4.3	CLEAR_RX_QUEUE	18
7.4.4	CLEAR_PERIODIC_MSGS	18
7.4.5	REQUEST_CONNECTION	18
7.4.6	TEARDOWN_CONNECTION	18
8.	DISCOVERY MECHANISUM	19
8.1	GET_PROTOCOL_INFO	19
8.2	GET_RESOURCE_INFO	24
9.	NOTES	24
9.1	Revision Indicator	24
APPENDIX A		25
Table 1	Required connector/pin combinations	5
Table 2	Minimum receive buffer size	6
Table 3	Minimum transmit buffer size	6
Table 4	Network errors	7
Table 5	SAE J2534 API functions	8
Table 6	RxStatus bit definitions	9
Table 7	TxFlags bit definitions	9
Table 8	List of indications	11
Table 9	List of description codes for the error indication	12
Table 10	Connect flag values	13
Table 11	Return values for PassThruLogicalConnect	13
Table 12	Return values for PassThruLogicalDisconnect	14
Table 13	Return values for PassThruQueueMsgs	14
Table 14	Return values for PassThruStartPeriodicMsgs	15
Table 15	Return values for PassThruIoctl	16
Table 16	IOCTL details	16
Table 17	GET_CONFIG/SET_CONFIG details	17
Table 18	GET_CONFIG/SET_CONFIG parameters details	17
Table 19	REQUEST_CONNECTION	18
Table 20	TEARDOWN_CONNECTION	19
Table 21	GET_PROTOCOL_INFO	19
Table 22	GET_PROTOCOL_INFO details for CAN physical channel	20
Table 23	GET_PROTOCOL_INFO details	20

1. SCOPE

This SAE Recommended Practice is part of the SAE J2534-2/X_0500 set of documents that extends the SAE J2534-1_0500 API (version 05.00) specification, and defines how to implement TP2.0 within the SAE J2534 API framework. This document details only the changes from SAE J2534-1_0500 and items not specifically detailed in this document are assumed to have not changed.

An SAE J2534-2/7_0500 interface shall be compliant to the TP2.0 feature only when all the required functionality in this SAE Recommended Practice is implemented. Any functionality not required for compliance will be specifically marked as “optional” in this document.

This document must be used in conjunction with the SAE J2534-2/BA_0500 and SAE J2534-2/RE_0500 documents.

2. REFERENCES

2.1 Applicable Documents

The following publications form a part of this specification to the extent specified herein. Unless otherwise indicated, the latest issue of SAE publications shall apply.

2.1.1 SAE Publications

Available from SAE International, 400 Commonwealth Drive, Warrendale, PA 15096-0001, Tel: 877-606-7323 (inside USA and Canada) or +1 724-776-4970 (outside USA), www.sae.org.

SAE J1962	Diagnostic Connector
SAE J2534-1_0500	Recommended Practice for Pass-Thru Vehicle Programming (API version 05.00)
SAE J2534-2/BA_0500	Pass-Thru Extended Features - Base Document (API version 05.00)
SAE J2534-2/RE_0500	Pass-Thru Extended Features - Resource Document (API version 05.00)
SAE J2819	TP2.0 Vehicle Diagnostic Protocol

3. DEFINITIONS

ACTIVE MESSAGE: Refers to a message that has started the transmission process on the physical vehicle bus but has not completed it.

LOGICAL COMMUNICATION CHANNEL: A specific path through the pass-thru interface to the physical vehicle bus that uses an existing physical communication channel but adds a network and transport layer (as defined in the OSI seven layers).

PASS-THRU DEVICE: Refers only to the hardware portion of the SAE J2534 interface.

PASS-THRU INTERFACE: Refers to all components of a SAE J2534 interface, including both hardware and software.

PHYSICAL COMMUNICATION CHANNEL: A specific path through the pass-thru interface to the physical vehicle bus that encompasses the physical layer and data link layer (as defined in the OSI seven layers) and includes all the associated physical resources (such as the data link controller, connector, pins, etc.) in the pass-thru device.

QUEUED MESSAGE: Refers to the acceptance of a full message from the application by the pass-thru interface (via the API), which is intended to become an active message at some point in the future.

TERMINATE A MESSAGE: Refers to the process of stopping reception or transmission of a message by the pass-thru interface.

TRANSMITTED MESSAGE: Refers to a previously active message that has been successfully transmitted on the physical vehicle bus.

4. ACRONYMS

API	Application Programming Interface
CAN	Controller Area Network
CRC	Cyclic Redundancy Check
DLL	Dynamic Link Library
ECU	Electronic Control Unit
IOCTL	Input/Output Control
MNTC	Maximum Repeats of the Connecting Structure Telegrams
MSB	Most Significant Byte

5. DOCUMENTATION CONVENTIONS

In the document body and in tables:

- All SAE J2534 API functions are in bold print (for example, **PassThruOpen**).
- All SAE J2534 API variables are placed between “<” and “>” (for example, <DeviceID>).
- All SAE J2534 API DEFINES shall be in upper-case print (for example, STATUS_NOERROR).
- All numbers are in decimal unless otherwise indicated. (Hexadecimal values have a leading “0x”; for example, 0x7AF or 0x800 are hexadecimal values.)

6. PASS-THRU INTERFACE REQUIREMENTS

The TP2.0 feature shall be implemented as a logical channel that communicates over a CAN physical communication channel.

TP2.0 is defined in SAE J2819.

6.1 Specific Protocol Requirements

Key functionality of TP2.0 are the establishment and maintenance of a channel between two nodes. The maintenance of a connection on an established channel and the ability to send broadcast messages that repeat five times at a configurable time spacing.

A pass-thru interface that supports the TP2.0 feature shall:

- a. Support data rates of 250 kbps and 500 kbps.
- b. Support 11 or 29 bit CAN identifiers.

- c. Support a CAN physical communication channel that will allow four logical channels to establish and maintain connections simultaneously. Support a single broadcast message that is repeated five times with the last 2 bytes alternating between 0x55 and 0xAA with configurable time spacing between the messages.
- d. Send a re-trigger broadcast message which starts out as stated above with additional messages sent at a slower configured rate.
- e. Support stopping a re-triggered broadcast message.
- f. Support the ability to establish and maintain a single connection on each logical channel.
- g. Send indication messages to the application when a connection is established or disconnected.
- h. Fragment and reassemble messages.
- i. Buffer one outgoing and one incoming messages as they are being fragmented or reassembled on each logical channel.
- j. Handle the acknowledge with retry and break events.
- k. Automatically transmit the connection test telegram to keep a connection active.
- l. Maintain the sequence number as part of the data telegram and data acknowledge telegram for each connection.
- m. Support one connection as a passive device (a module on the network requests a connection with the interface). This passive connection is configurable meaning it can be disabled. A passive connection counts as one of the 4 connections supported by the pass-thru interface. Allow TP2.0 and ISO 15765 to coexist on the same CAN physical communication channel.
- n. When a logical channel is created, it shall not have a connection address and shall be unable to initiate any TP2.0 transport communications until an IOCTL requesting an address is successful. The application shall be able to transmit single frame messages even if no connection address is established.
- o. Support full duplex communication.

6.2 Pin Usage

Pin usage is determined by the physical communication channel. [Table 1](#) defines the required connector/pin usages for the TP2.0 feature as they appear in the structure RESOURCE_STRUCT, where the order of elements in the array ResourceListPtr defines its purpose. Other connector/pin combinations may optionally be supported by the pass-thru interface and must be queried the using the discovery mechanism.

Table 1 - Required connector/pin combinations

RESOURCE_STRUCT Contents
<ResourceList.Connector> = J1962_CONNECTOR;
<ResourceList.NumOfResources> = 2;
<ResourceList.ResourceListPtr[0]> = 6; // CAN_H
<ResourceList.ResourceListPtr[1]> = 14; // CAN_L

6.3 Data Buffering

6.3.1 Message Reception

The TP2.0 feature shall provide a receive buffer as specified in [Table 2](#) and shall be capable of receiving and buffering messages while the receive buffer is not full. The number of messages buffered will depend on the size of the messages received. This capability shall be available on each channel at all specified baud rates.

Table 2 - Minimum receive buffer size

Maximum Size of Each Message	Minimum Receive Buffer Size
4096 bytes	4128 bytes

6.3.2 Message Transmission

The TP2.0 feature shall provide a transmit buffer as specified in [Table 3](#) for individual messages (using **PassThruQueueMsgs**) and shall be capable of buffering messages while the transmit buffer is not full. The number of messages buffered will depend on the size of the messages to be transmitted. This capability shall be available on each channel at all specified baud rates.

Table 3 - Minimum transmit buffer size

Maximum Size of Each Message	Minimum Transmit Buffer Size
4096 bytes	4128 bytes

NOTE: This buffer is only for messages filled using **PassThruQueueMsgs**.

6.4 Error Handling

6.4.1 Device Not Connected

As defined in SAE J2534-1_0500.

6.4.2 Receive Buffer Overflow

As defined in SAE J2534-1_0500.

6.4.3 Message Termination

As defined for CAN in SAE J2534-1_0500. The message termination boundary is a CAN frame.

6.4.4 Network Errors

The pass-thru interface shall be capable of detecting transmit and receive errors. To deal with these errors, recovery strategies have been defined SAE J2534-1_0500. [Table 4](#) details the types of errors to be detected and the associated recovery strategy as well as the required user notification that shall occur at the end of the designated recovery strategy.

Table 4 - Network errors

Error	Application Notification Mechanism	Recovery Strategy
Acknowledgement error or loss of arbitration	NONE or TxFailed indication (if a TxDone was requested)	Controller retry as defined in SAE J2534-1_0500
Bus warning	NONE	Controller retry as defined in SAE J2534-1_0500
Overrun error	Set BUFFER_OVERFLOW in <RxStatus>	Drop strategy as defined in SAE J2534-1_0500
Bus off	Generate LINK_DOWN error indication	Passive operation strategy as defined in SAE J2534-1_0500
During message reception: form error, stuff error, CRC error, bit 0 error, bit 1 error, or errors defined in TP2.0 (T1 timeout)	NONE	NONE
During message transmission: form error, stuff error, CRC error, bit 0 error, bit 1 error	NONE or TxFailed indication (if a TxDone was requested)	Controller retry strategy
During message transmission errors defined in TP2.0	NONE or TxFailed indication (if a TxDone was requested)	Drop strategy as defined in SAE J2534-1_0500

6.4.5 Message Timeout

A node that times out while trying to send a message shall repeat the last telegram MNCT times (specified using the IOCLTID SET_CONFIG with the parameter TP2_0_MNCT). If, at the end of the attempts, the messages has still not successfully been sent then the pass-thru interface shall send the disconnect telegram (refer to error handling in SAE J2819). The failed pass-thru message is then removed from the transmit queue and a TxFail is queued in the Rx buffer, if a TxDone was requested.

7. WIN32 APPLICATION PROGRAMMING INTERFACE

7.1 API Changes

[Table 5](#) summarizes the changes to the SAE J2534-1_0500 API functions for this feature. The pass-thru API DLL must include entry points for all functions, regardless of whether or not the API function call is used for this feature.

Table 5 - SAE J2534 API functions

Function	Description
PassThruScanForDevices	Function implementation independent of this feature
PassThruGetNextDevice	Function implementation independent of this feature
PassThruOpen	As defined in SAE J2534-1_0500
PassThruClose	As defined in SAE J2534-1_0500
PassThruConnect	This function shall always return an error (applicable error values defined in SAE J2534-1_0500)
PassThruDisconnect	As defined in SAE J2534-1_0500
PassThruLogicalConnect	Added new ProtocolID value
PassThruLogicalDisconnect	As defined in SAE J2534-1_0500
PassThruSelect	As defined in SAE J2534-1_0500
PassThruReadMsgs	As defined in SAE J2534-1_0500
PassThruQueueMsgs	Added new return value, ERR_NO_CONNECTION_ESTABLISHED
PassThruStartPeriodicMsg	Added new return value, ERR_NO_CONNECTION_ESTABLISHED
PassThruStopPeriodicMsg	As defined in SAE J2534-1_0500
PassThruStartMsgFilter	This function shall always return an error (applicable error values defined in SAE J2534-1_0500)
PassThruStopMsgFilter	As defined in SAE J2534-1_0500
PassThruSetProgrammingVoltage	Function implementation independent of this feature
PassThruReadVersion	Function implementation independent of this feature
PassThruGetLastError	Function implementation independent of this feature
PassThruIoctl	Added discovery mechanism, new IOCTL ID, and configuration parameters

The TP2.0 feature does not allow pass and block filters, thus PassThruStartMsgFilter is not supported.

7.2 Message Structure

7.2.1 C/C++ Definition

The PASSTHRU_MSG structure, defined in SAE J2534-1_0500, shall be used for TP2.0 messages.

```
typedef struct {
    unsigned long ProtocolID;
    unsigned long MsgHandle;
    unsigned long RxStatus;
    unsigned long TxFlags;
    unsigned long Timestamp;
    unsigned long DataLength;
    unsigned long ExtraDataIndex;
    unsigned char *DataBuffer;
    unsigned long DataBufferSize;
} PASSTHRU_MSG;
```

7.2.2 Description of Structure Elements

- <ProtocolID> Value shall be TP2_0_LOGICAL.
- <MsgHandle> As defined by SAE J2534-1_0500.
- <RxStatus> The receive message status. See [7.2.2.1](#) for more details.
- <TxFlags> The transmit message flags. See [7.2.2.2](#) for more details.
- <Timestamp> As defined by SAE J2534-1_0500.

<DataLength> As defined by SAE J2534-1_0500.

<ExtraDataIndex> As defined by SAE J2534-1_0500.

<DataBuffer> As defined by SAE J2534-1_0500.

<DataBufferSize> As defined by SAE J2534-1_0500.

7.2.2.1 RxStatus

The TP2.0 feature shall support only the <RxStatus> bits defined in [Table 6](#). All unused bits shall be set to zero.

Table 6 - RxStatus bit definitions

Name	Description	Value
TX_FAILED	Indicates that the associated message is a TxFailed indication (refer to SAE J2534-1_0500 for more details)	0 = No TxFailed indication 1 = TxFailed indication
CAN_29BIT_ID	Indicates the CAN ID type	0 = 11-bit identifier 1 = 29-bit identifier
BUFFER_OVERFLOW	Indicates that the channel had a buffer overflow (refer to SAE J2534-1_0500 for more details)	0 = No buffer overflow 1 = Buffer overflow occurred prior to this message - data was lost
ERROR_INDICATION	Indicates that there was an error detected on the channel (refer to SAE J2534-1_0500 for more details)	0 = No ErrInd indication 1 = ErrInd indication
TX_SUCCESS	Indicates that the associated message is a TxDone indication (refer to SAE J2534-1_0500 for more details)	0 = No TxDone indication 1 = TxDone indication
TX_MSG_TYPE	Indicates that the source of the message was a transmission	0 = Received message 1 = Message transmitted by this channel
START_OF_MESSAGE	Indicates that the associated message is a start indication (refer to SAE J2534-1_0500 for more details)	0 = No start indication 1 = Start indication
CONNECTION_LOST	Indicates that a connection was lost or a claim attempt failed	0 = No ConnectionLost indication 1 = ConnectionLost indication
CONNECTION_ESTABLISHED	Indicates that a connection was successfully established indication	0 = No ConnectionEstablished indication 1 = ConnectionEstablished indication

7.2.2.2 TxFlags

The TP2.0 feature shall support only the <TxFlags> defined in [Table 7](#). All unused bits shall be set to zero.

Table 7 - TxFlags bit definitions

Name	Description	Value
CAN_29BIT_ID	Indicates the CAN ID type	0 = 11-bit identifier 1 = 29-bit identifier
TP2_0_BROADCAST_MSG	Send this message as a broadcast which means it will be sent five times with an interval of TP2_0_T_BR_INT The pass-thru interface shall insure that the last 2 data bytes alternate between 0x55 and 0xAA	0 = normal message 1 = TP2.0 broadcast message