GM	WORLDWIDE ENGINEERING STANDARDS	General Specification	GMW17753

GM Extension to SAE J2534 Standard to Support CAN FD

1 Introduction

Note: Nothing in this standard supercedes applicable laws and regulations.

Note: In the event of conflict between the English and domestic language, the English language shall take precedence.

1.1 Purpose. This purpose of this standard is to define an implementation of the Controller Area Network with Flexible Data (CAN FD) protocol within the framework of SAE J2534-2, V04.04. This implementation utilizes the manufacture discretionary features defined in the SAE J2534 standard. The intent is to add support of the CAN FD protocol with minimal changes to existing applications and interfaces. The CAN FD protocol implemented two (2) strategies to increase data throughput of a Controller Area Network (CAN) 2.0 network:

- Increased frame payload size to 64 Bytes.
- Higher Baudrates (5 Mpbs).

Also included in this standard are enhancements to the ISO 15765-2 protocol that were necessary to support CAN FD:

- Configurable Frame Size.
- Increased Message Size.

1.2 Applicability. This document describes hardware and software changes required to add CAN FD to an SAE J2534-1/SAE J2534-2 interface. It also serves a reference guide for application developers that wish to add CAN FD support while staying within the framework of J2534, V04.04.

1.3 Remarks.

- References to “J1962” throughout the document relate to the SAE J1962 Diagnostic Connecter for On-Board Diagnostic purposes.
- References to “J2284” throughout the document relate to SAE J2284 and are use-case specific. Care should be exercised to apply the proper specification (see 2.1) for each application.
- References to “J2534” throughout the document relate to SAE J2534 and are use-case specific. Care should be exercised to apply the proper specification (see 2.1) for each application.

2 References

Note: Only the latest approved standards are applicable unless otherwise specified.

2.1 External Standards/Specifications.

ISO 11898-1	SAE J1962	SAE J2284-1	SAE J2284-4
ISO 11898-2	SAE J2534-1	SAE J2284-2	SAE J2284-5
ISO 15765-2	SAE J2534-2	SAE J2284-3	

2.2 GM Standards/Specifications.

GMW3059

2.3 Additional References.

Bosch CAN FD specification: CAN with Flexible Data-Rate, Specification Version 1.0 (released April 17, 2012)

3 Requirements

Unless indicated otherwise, the SAE J2534, V04.04 definitions of requirements, restrictions and error conditions for Controller Area Network Pin Select (CAN_PS) and ISO15765_PS protocols will apply to CAN_FD_PS and ISO15765_FD_PS channels. For CAN FD the ISO 15765-2 Extended Data (ED) standard has been extended to support larger message sizes.

3.1 Hardware Requirements.

3.1.1 CAN FD Channels. The Pass-Thru Interface shall have the ability of supporting at least two (2) concurrent ISO 11898-1/ISO 11898-2 CAN FD channels.

3.1.2 J1962 Pin Usage. See Table 1.

Table 1: Pin Usage

Channel	Required	CAN(+)	CAN(-)
CH1	Yes	6	14
CH2	Yes	1	9
CH3	Optional	3	11
CH4	Optional	12	13

3.1.3 Termination Resistor. The Pass-Thru Interface have the capability of connecting a 120 Ω termination resistor across the J1962 pins of each channel. The state of the termination resistor shall be controlled by the application.

3.2 Protocol Identification (ProtocolID) Values. In order to identify Pass-Thru Device(s) that support CAN FD protocol, the device manufacture shall include the following Protocol Registry key in the J2534_V0404 registry hive. The value of the KEY shall be selected from the manufacture specific range and assigned to the Key in the J2534 registry. See Table 2.

Table 2: ProtocolID Registry Keys

Registry Key	Value	Comment
ISO15765_FD_PS	0×1000C	This is the value used by the GM Pass-Thru Interface
CAN_FD_PS	0×1000D	This is the value used by the GM Pass-Thru Interface

3.3 Message Structure. The following defines how the J2534 Pass-Thru message type definition (typedef) structure (struct) elements that are used by the CAN_FD_PS and ISO15765_FD_PS protocol:

```
typedef struct {
    unsigned long ProtocolID;
    unsigned long RxStatus;
    unsigned long TxFlags;
    unsigned long Timestamp;
    unsigned long DataSize;
    unsigned long ExtraDataIndex;
    unsigned char Data[4128];
} PASSTHRU_MSG;
```

3.3.1 CAN_FD_PS Message Structure. The message structure parameters used for a CAN_FD_PS channel are very similar to CAN_PS (SAE J2534-2, V04.04). The following describes valid settings for each parameter; using any other settings will cause the Pass-Thru Interface to return an ERR_INVALID_MSG.

- **ProtocolID:** Shall be set to CAN_FD_PS.
- **DataSize:** The <DataSize> for a CAN FD formatted message must set to a valid value 4 thru 12, 16, 20, 24, 28, 36, 52, or 68 bytes, else the J2534 Application Programming Interface (API) shall return ERR_INVALID_MSG. The <DataSize> for a CAN 2.0 formatted message must be within the range of 4 bytes to 12 bytes.
- **Transmit (Tx) Flags:** Table 3 identifies the valid TxFlags for a CAN_FD_PS message. Setting any other Flag for a CAN_FD_PS message will cause the Pass-Thru Interface to return an ERR_INVALID_MSG.

Table 3: CAN_FD_PS TxFlags

Definition	Bit	Value
CAN_FD_PS	24	0 = The message shall be transmitted with the data phase at the arbitration speed. Note 1 1 = The message shall be transmitted with the data phase at the CAN_FD_DATA_PHASE_RATE. Note 1
CAN_FD_FORMAT	25	0 = The message shall be transmitted the all using the CAN 2.0 format. 1 = The message shall be transmitted the all using the CAN FD format.
CAN_29BIT_ID	8	As defined by CAN in J2534, V04.04.

Note 1: This flag will be ignored unless CAN_FD_FORMAT flag is set to 1.

- **Receive (Rx) Status:** Table 4 identifies the valid RxStatus bits for a received message, and shows, which bits are used as indications.

Table 4: CAN_FD_PS RxStatus Bits

Definition	Bit	IND	Value
CAN_FD_BRS	24	-	This flag reflects the value of the BRS bit of the last frame in the received message. Note 1 0 = The data phase of the last frame in the message was received at arbitration speed. 1 = The data phase of the last frame in the message was received at the CAN FD data phase rate.
CAN_FD_FORMAT	25	-	0 = The message was received in CAN 2.0 format. 1 = The message was received in CAN FD format.
CAN_FD_ESI	26	-	This flag reflects the value of the ESI bit of the message. Note 2 0 = ESI bit is active in received CAN FD frame. The sender is in the Error Active State. 1 = ESI bit is passive in received CAN FD frame. The sender is in the Error Passive State.
CAN_29BIT_ID	8	-	As defined by CAN in J2534, V04.04.
ERROR_IND	4	X	Indicates that the Pass-Thru Interface detected an error while receiving or transmitting a message.
TX_MSG_TYPE	0	-	As defined by CAN in J2534, V04.04.

Note: BRS = Baud Rate Switch, ESI = Error State Indicator, IND = Indication, MSG = Message.
Note 1: It is possible to receive a Multi-Frame message where the BRS bit is different for some frames. This flag shall be zero (0) if CAN_FD_FORMAT is not set.
Note 2: This flag must be zero when CAN_FD_FORMAT is 0.

3.3.2 ISO15765_FD_PS Message Structure. The message structure used for ISO15765_FD_PS is identical to ISO15765_PS (SAE J2534-1, V04.04) with the exception of:

1. The Pass-Thru Interface shall permit the usage of Mixed Format Frames on an ISO15765_FD_PS channel. This allows the simultaneous reception and transmission of ISO15765_FD_PS messages and unformatted CAN FD frames on the ISO15765_FD channel. In Mixed mode, the ProtocolID (in the PASSTHRU_MSG structure) is used to identify if the ISO_15765 transport layer is applicable for the associated message.
2. API calls that cause a message transmission of an ISO15765_FD_PS message shall limit the maximum frames size to the value defined by CAN_FD_TX_DATA_LENGTH. All frames containing more than 8 (eight) data bytes shall be padded out to the next valid frame boundary with the value of ISO15765_PAD_VALUE. Frames with less than 8 (eight) data bytes may also be padded out to 8 (eight) data bytes if the TxFlag ISO15765_FRAME_PAD is set.

- **ProtocolID:** Shall be set to ISO15765_FD_PS.
- **DataSize:** The minimum value for the <DataSize> of an ISO15765_FD_PS message is dependent on the <TxFlag> ISO15765_ADDR_TYPE. See Table 5.

Table 5: <DataSize> ISO15765_FD_PS

ISO15765_ADDR_TYPE	Minimum Value of <DataSize>	Maximum Value of <DataSize>
0	4	4128 Note 1
1	5	4128 Note 1

Note 1: The maximum value for the <DataSize> of an ISO15765_FD_PS message is limited by the Pass-Thru message structure to 4128 bytes. However, it is possible for an application to transmit and receive larger messages through an extended message buffer.

- **TxFlags:** The following table identifies the valid TxFlags for an ISO15765_FD_PS message. Setting any other Flag for a CAN_FD_PS message will cause the Pass-Thru Interface to return an ERR_INVALID_MSG. See Table 6.

Table 6: <TxFlags> ISO15765_FD_PS

Definition	Bit	Value
CAN_FD_BRS	24	0 = All the frames of the message shall be transmitted with the data phase at the arbitration speed. Note 1 1 = All the frames of the message shall be transmitted with the data phase at the CAN_FD_DATA_PHASE_RATE. Note 1
CAN_FD_FORMAT	25	0 = All the frames of the message shall be transmitted using the CAN 2.0 format. 1 = All the frames of the message shall be transmitted using the CAN FD format.
CAN_29BIT_ID	8	As defined by CAN in J2534, V04.04.
ISO15765_ADDR_TYPE	7	As defined by CAN in J2534, V04.04.
ISO15765_FRAME_PAD	6	As defined by CAN in J2534, V04.04 for frames less than 8 (eight) data bytes. Frames with more than 8 (eight) data bytes shall use default CAN FD padding. (The frames are padded to the next CAN FD frame boundary.)

Note 1: This flag will be ignored unless CAN_FD_FORMAT flag is set to 1.

- **RxStatus:** Table 7 identifies the valid RxStatus bits for a received message and shows which bits are used as indications.

Table 7: <RxStatus> ISO15765_FD_PS

Definition	Bit	IND	Value
CAN_FD_BRS	24	-	This flag reflects the value of the BRS bit of the last frame in the received message. Note 1 0 = The data phase of the last frame in the message was received at arbitration speed. 1 = The data phase of the last frame in the message was received at CAN_FD_DATA_PHASE_RATE.
CAN_FD_FORMAT	25	-	0 = All frames of the message were received in CAN 2.0 format. 1 = All frames of the message were received in CAN FD format.
CAN_FD_ESI	26	-	This flag reflects the value of the ESI bit of the last frame in the received message. Note 2 0 = ESI bit is active in received CAN FD frame. The sender is in the Error Active State. 1 = ESI bit is passive in received CAN FD frame. The sender is in the Error Passive State.
RX_MSG_TRUNCATED	27	-	The Rx message length exceeded 4128 bytes of data and the Pass-Thru Interface has truncated the message. The remaining data bytes can be read from the extended Rx Buffer. 0 = The entire message has been returned in the <Data> array. 1 = 4128 bytes of the message have returned in the <Data> array the remaining bytes must be read from the Extended Data array.
CAN_28BIT_ID	8	-	As defined by ISO 15765 in J2534, V04.04.
ISO15765_ADDR_TYPE	7	-	As defined by ISO 15765 in J2534, V04.04.
ERROR_IND	4	X	This bit was redefined from J2534, ISO 15765 Padding Error. Indicates that the Pass-Thru Interface detected an error while receiving or transmitting a message.
TX_INDICATION	3	X	As defined by ISO 15765 in J2534, V04.04.
TX_MSG_TYPE	0	-	As defined by ISO 15765 in J2534, V04.04.
START_OF_MESSAGE	1	X	Indicates the first frame of a received message has been received.

Note: BRS = Baud Rate Switch, ESI = Error State Indicator, IND = Indication, MSG = Message, Rx = Receive.
Note 1: It is possible to receive a Multi-Frame message where the BRS bit is different for some frames. This flag shall be zero (0) if CAN_FD_FORMAT is not set.
Note 2: This flag must be zero (0) when CAN_FD_FORMAT bit is 0.

3.3.2.1 If the <DataSize> of a Pass-Thru message exceeds 4128 bytes then the application must use an Extended Data array to exchange the additional data with the Pass-Thru Interface.

Refer to the Input/Output Control <ioctlID> WRITE_MSG_EXTENSION for details on transferring extended data of a transmit message to the Pass-Thru Interface. After the application has transferred then extended data PassThruWriteMsgs() will allow the <DataSize> of the message to exceed 4128 bytes by the number of bytes in the extended data.

Refer to the <ioctlID> READ_MSG_EXTENSION for details obtaining the extended data of a received message.

3.4 Indications. The Pass-Thru Interface shall only set one indication bit in the <RxStatus>.

- <Timestamp> shall reflect the time the associated indication occurred.
- <ProtocolID> shall match the <ProtocolID> of the channel.
- <TxFlags> shall be 0.
- <ExtraDataIndex> shall be 0.
- <DataSize> minimum value shall be 4.

3.4.1 CAN_FD_PS Indications.

ERROR_IND

This Indication shall be generated when the Pass-Thru Interface has detected an error condition. See Table 8.

With this Indication, the first 4 bytes of the array <Data> in the PASSTHRU_MSG structure shall identify the ErrorCode for the indicator. (Most Significant Byte First).

<DataSize> shall be 4.

<RxStatus> All status bits shall be 0.

Table 8: Error Codes for ERROR_IND

Error Codes for ERROR_IND	Value	Description
CAN_OVERFLOW	1	The Pass-Thru interface was not able to keep up with the network traffic and frames have been dropped.
BUS_OFF	2	The Pass-Thru interface has gone BUS_OFF.

3.4.2 ISO15765_FD_PS Indications. ISO15765_FD_PS also inherits CAN_FD_PS indications regardless if Mixed Mode is not enabled. See Table 9.

START_OF_MESSAGE

The Start Indication shall be generated after receiving the First Frame (FF) of a Segmented Message that is able to be queued in the Receive Buffer.

<RxStatus> shall reflect the attributes associated with the First Frame (FF) of the message.

<DataSize> shall indicate the number of bytes in the indication, 8 bytes for Normal Addressing, 9 bytes for Extended Addressing.

The array <Data> shall contain the CAN ID and possibly the extended address of the message being received, followed by the number of data bytes required to receive the message.

The Pass-Thru interface shall report the number of data bytes that are required to receive the message as a 4 byte unsigned integer most significant byte first.

TX_INDICATION

As defined by ISO 15765 in SAE J2534-1

ERROR_IND

This Indication shall be generated when the Pass-Thru Interface has detected an error condition.

With this Indication, the first 4 bytes of the array <Data> in the PASSTHRU_MSG structure shall identify the ErrorCode for the indicator. (Most Significant Byte First).

<DataSize> shall be 4.

<RxStatus> All status bits shall be 0.

Table 9: Error Codes for ERROR_IND

Error Codes for ERROR_IND	Value	Description
CAN_OVERFLOW	1	As defined by CAN_FD_PS
BUS_OFF	2	As defined by CAN_FD_PS
RX_ERROR	3	The Pass-Thru Interface terminated the message reception because the maximum frame separation time was exceeded. (N_CR_MAX) or it detected one (1) of the following conditions on an incoming message: • Invalid Frame Size: The consecutive frame size of the Rx message does not match the size of the first frame. Note • Invalid Frame Format: The consecutive frame format does not match the format of the first frame. • Invalid Sequence number in a consecutive frame.
TX_FAILED	4	The Pass-Thru Interface terminated message transmission for one (1) of the following reasons. • There was No Flow Control frame received or the Flow Control message was improperly formatted. • Wait Flow Control (FC) limit was exceeded.

Note: The last frame of the message may be shorter than the first frame.

3.5 J2534 API Functions. The following does not define new API functions, it simply identifies where changes are required to support CAN FD.

3.5.1 PassThruConnect(). PassThruConnect opens a CAN_FD_PS or ISO15765_FD_PS channel on the Pass-Thru Interface.

Note: Currently there are two (2) version of CAN FD available Bosch CAN FD (see 2.3) and International Standards Organization (ISO) CAN FD. ISO CAN FD offers improved error detection and will eventually supersede the Bosch version. By default, the Pass-Thru Device shall support ISO CAN FD, but may optionally provide provisions to enable Bosch CAN FD compatibility mode. (See <ioctlID> CAN_FD_TYPE for additional details).

When PassThruConnect is called with either the CAN_FD_PS or ISO15765_FD_PS <ProtocolID>, the physical layer remains disconnected until <iIoclId> J1962_PINS is set with valid pins.

It is recommended that applications specify CAN_FD_DATA_PHASE_RATE and CAN_FD_TYPE if using Bosch CAN FD prior to setting the <ioctlID> J1962_PINS to avoid bus errors.

PROTOCOLID The Pass-Thru Interface may be configured to transmit and receive Raw CAN FD or use a transport layer as specified by ISO 15765.

- CAN_FD_PS
- ISO15765_FD_PS

FLAGS The flags shown in Table 10 are applicable for ISO15765_FD_PS and CAN_FD_PS.

Table 10: Connect Flags

Flag	Bit Position	Description
CAN_ID_BOTH	11	As defined by SAE J2534-1
CAN_29_BIT_ID	8	As defined by SAE J2534-1

BaudRate Specifying a value for the <BaudRate> provides a method for the application to initialize the Arbitration data rate of the network. The <BaudRate> is expressed in Bits per Second (bps). See Table 11.

Table 11: Arbitration Baud Rates

Value	Description
500 000	500 Kbps
250 000	250 Kbps
125 000	125 Kbps

3.5.1.1 Return Values. As defined by SAE J2534-1.

3.5.2 PassThruStartMsgFilter(). For the most part PassThruStartMsgFilter shall function as defined by SAE J2534-2. CAN_FD_PS will follow the same logic as CAN_PS and ISO15765_FD_PS will follow the same logic flows and ISO15765_PS. The following only defines the differences.

3.5.2.1 Return Values. As defined by SAE J2534-1.

3.5.2.2 PassThruStartMsgFilter(): Parameters. See Table 12.

Table 12: Supported Filter Types

Filter Type	Usage
PASS_FILTER	As defined by SAE J2534-1
BLOCK_FILTER	As defined by SAE J2534-1
FLOW_CONTROL_FILTER	As defined by SAE J2534-1 (ISO15765_FD_PS only)

pPatternMsg RXstatus flags of the Pattern Message are not part of the filtering algorithm.

The TxFlag CAN_29BIT_ID is used to determine if the Pattern Message's CANID is 29bit or 11bit.

If the <ProtocolID> is ISO15765_FD_PS then the TxFlag ISO15765_ADDR_TYPE is used to determine if Extended Addressing is valid for the CANID.

All filters are intended to be insensitive to the CAN Format of the message being filtered. The Pass-Thru Interface shall confirm the <DataSize> of an incoming message is greater than the <DataSize> of the Pattern Message.

pMaskMsg RXstatus flags of the Mask Message are not part of the filtering algorithm. The <DataSize>, <TxFlags> and <ProtocolID> of the Mask message, shall be equal to the values provided in the Pattern Message.

pFlowControlMsg This parameter shall always be set to NULL for a pattern message with a <ProtocolID> of CAN_FD_PS.

For ISO15765_FD_PS, this structure is used to define the properties of the flow control handshake from the Pass-Thru Interface. If set to NULL the Pass-Thru Interface shall not send a flow control.

The <ProtocolID> shall be ISO15765_FD_PS.

An ERR_INVALID_MSG shall be returned if the Flow Control Message is configured to use a different Address Type than the Pattern Message or if <DataSize> is set to an undefined value (see 3.5.2.2.1).

The Flow control message transmitted by the Pass-Thru Interface shall match the same CAN FD format as the First Frame of the received message.

pFilterID As defined by J2534

3.5.2.2.1 Dynamic Flow Control. Application that sends all node requests would normally be required to create a Flow Control Filter for each node capable of replying. A Dynamic Flow Control filter reduces the number of FLOW_CONTROL_FILTERS that are required to support this use case.

To enable dynamic flow control responses the application would specify a <DataSize> for the <pFlowControlMsg> message to as specified by Table 13 and Table 14.

Table 13: ISO 15765 Normal Addressing

Dynamic	<DataSize>	CAN Address Size	Flow Control Message Construction (Normal Addressing)
Yes	2	11bit	Invalid Flow Control Message: ERR_INVALID_MSG
		29bit	FC.Data[0 thru 1] shall be from the <pFlowControlMsg>. FC.Data[2] shall be set to RxMsg.Data[3] of the first frame. FC.Data[3] shall be set to RxMsg.Data[2] of the first frame.
No	3	11bit or 29bit	Invalid Flow Control Message: ERR_INVALID_MSG
No	4	11bit or 29bit	Data [0 thru 3] shall be from the <pFlowControlMsg>

Table 14: ISO 15765 Extended Messaging

Dynamic	<DataSize>	CAN Address Size	Flow Control Message Construction (Extended Addressing)
Yes	2	11bit	Invalid Flow Control Message: ERR_INVALID_MSG
		29bit	FC.Data[0 thru 1] shall be from the <pFlowControlMsg>. FC.Data[2] shall be set to RxMsg.Data[3] of the first frame. FC.Data[3] shall be set to RxMsg.Data[2] of the first frame. FC.Data[4] shall be set to RxMsg.Data[4] of the first frame.
No	3	11bit or 29bit	Invalid Flow Control Message: ERR_INVALID_MSG
Yes	4	11bit or 29bit	FC.Data[0 thru 3] shall be from the <pFlowControlMsg>. FC.Data[4] shall be set to RxMsg.Data[4] of the first frame.
No	5	11bit or 29bit	FC.Data[0 thru 4] shall be from the <pFlowControlMsg>.

3.5.2.3 Return Values. As defined by SAE J2534-1, however; the condition checks for the ERR_INVALID_MSG have been modified to account for dynamic Filter <DataSize>.

3.5.3 PassThruStartPeriodicMsg(). For the most part PassThruStartPeriodicMsg remains as defined by SAE J2534-1. CAN_FD_PS inherits functionality from CAN, and ISO15765_FD_PS inherits functionality from ISO15765_PS. An ERR_INVALID_MSG shall be returned if the <DataSize> of the Periodic Message is not within the ranges defined for the <ProtocolID>. See Table 15.

Table 15: <DataSize> CAN_FD_PS

CAN_FD_FORMAT	Minimum Value of <DataSize>	Maximum Value of <DataSize>
0	4	12
1	4	68

If the <ProtocolID> of the message is ISO15765_FD_PS then the <DataSize> shall be limited to a single frame with 8 (eight) data bytes. See Table 16.

Table 16: <DataSize> ISO15765_PS

ISO15765_ADDR_TYPE	Minimum Value of <DataSize>	Maximum Value of <DataSize>
0	4	11
1	5	11

3.5.3.1 Return Values. As defined by SAE J2534-1.

3.5.4 PassThruWriteMsg(). For the most part CAN_FD_PS and ISO15765_FD_PS inherit all the basic functionality as defined by CAN_PS and ISO15765_PS respectively.

Refer to 3.3 message validation checks and TxFlags definitions.

3.5.4.1 ISO 15765 Multi-Frame Messages. For multi-frame ISO15765_FD_PS messages, the Pass-Thru Interface must have a filter configured to receive the Flow Control response from the receiving node. If multiple messages have been queued for transmit, the Pass-Thru Interface must associate each received flow control message with a message that it sending. For standard Flow Control Filters the <pFlowControlMsg> CANID is matched to the Transmit Message CANID. Since ISO15765_FD_PS also supports dynamic flow control filters, the Pass-Thru Interface must perform an additional step to determine, which transmit message is associated with the received flow control message.

The received flow control message must have the same CAN_FD_FORMAT as the message that is being transmitted.

The following checks are done to associate a received flow control message with a multi-frame message that is being transmitted. See also Table 17and Table 18.

pFlowControlMsg->Data[0] == Tx_msg.Data[0]

pFlowControlMsg->Data[1] == Tx_msg.Data[1]

pRxFC_Msg->Data[4] == 0x30 // Flow Control Message

Table 17: ISO 15765 Address Type Normal

pFlowControlMsg <DataSize>	Compare to Tx CANID	Comment
2	pRxFC_Msg->Data[2] == TX_msg.Data[3] pRxFC_Msg->Data[3] ==TX_msg.Data[2]	Swapping lower two bytes of the Rx flow control message for the CANID compare.
4	pFlowControlMsg->Data[2] ==TX.Data[2] pFlowControlMsg->Data[3] == TX.Data[3]	As defined by J2534

Table 18: ISO 15765 Address Type Extended

pFlowControlMsg <DataSize>	Compare to Tx CANID and Include the Extended Address	Comment
2	pRxFC_Msg->Data[2] == TX_msg.Data[3] pRxFC_Msg->Data[3] == TX_msg.Data[2] pRxFC_Msg->Data[4] == TX_msg.Data[4]	Swap the lower 2 bytes of Rx flow control message for the CANID compare. Also include the Extended Address from the Rx Flow Control message in the compare.
4	pFlowControlMsg->Data[2] == TX.Data[3] pFlowControlMsg->Data[3] == TX.Data[3] pRxFC_Msg->Data[4] == TX_msg.Data[4]	Use the Extended Address from the Rx Flow Control message in the compare.
5	pFlowControlMsg->Data[2] == TX.Data[2] pFlowControlMsg->Data[3] == TX.Data[3] pFlowControlMsg->Data[4] == TX.Data[4]	As defined by J2534

3.5.4.2 Extended Tx Messages. PassThruWriteMsgs only allows the first message in the message list to have an extended message buffer. If any of the following conditions exist, PassThruWriteMsgs shall return value ERR_INVALID_MSG and the message will not be queued for transmission.

- A message in the list has a <DataSize> greater than 4128 bytes and it is not the first message in the message list.
- The first message in the Message list has a <DataSize> greater than 4128 bytes and the application did not transfer extended data to the Pass-Thru Interface.
- The <DataSize> does not equal 4128 bytes + Number of bytes in the extended data.

If the <DataSize> equals 4128 bytes + Number of bytes in the extended data, then the extended data shall be appended to the message and the message queued for transmission else the extended data shall be cleared from the temporary buffer.

Refer to the <ioctlID> WRITE_MSG_EXTENSION for details on extending the Data array of a transmit message.

3.5.4.3 Return Values. As defined by J2534 with the addition of the following tests.

- ERR_INVALID_MSG shall be extended to allow extended Tx messages.
- ERR_NO_FLOW_CONTROL shall be extended to allow Dynamic flow Control Filters.

3.5.5 PassThruReadMsgs(). As defined by J2534, V04.04 except that support for messages beyond 4128 bytes is supported for ISO15765_FD_PS.

3.5.5.1 Extended Rx Messages. The Pass-Thru Interface shall set the RX_MSG_TRUNCATED status bit if <DataSize> of the received message exceeds 4128 bytes. The <DataSize> shall reflect the length of the received message even though the data field can only hold 4128 bytes.

The Pass-Thru Interface shall also store the truncated data in a temporary storage buffer to give the application an opportunity to read the data. Refer to <ioctlID> READ_MSG_EXTENSION for details on reading the truncated data.

If the first frame indicated a message length that was too large to save in the extended data buffer or the extended data buffer is full, then the Pass-Thru Interface shall not store the truncated data. The RX_MSG_TRUNCATED status bit shall be set and the <DataSize> shall indicate there is no extended data (<DataSize> = 4128 bytes).

Note: PassThruReadMsgs() function clears the previously created extended storage buffer each time it is called.

3.5.5.2 Return Values. As defined by SAE J2534-1.

3.5.6 PassThruIOCTL(). The following SAE J2534-1 ioctlIDs are valid for CAN_FD_PS and ISO15765_FD_PS. See Table 19.

Table 19: Supported ioctlIDs

ioctlID	Value	InputPtr	OutputPtr
GET_CONFIG	0x01	Pointer to SCONFIG_LIST	NULL Pointer
SET_CONFIG	0x02	Pointer to SCONFIG_LIST	NULL Pointer
CLEAR_TX_QUEUE	0x07	NULL Pointer	NULL Pointer
CLEAR_RX_QUEUE	0x08	NULL Pointer	NULL Pointer
CLEAR_PERIODIC_MSGS	0x09	NULL Pointer	NULL Pointer
CLEAR_MSG_FILTERS	0x0A	NULL Pointer	NULL Pointer
WRITE_MSG_EXTENSION	0x00010006	Pointer to SBYTE_ARRAY	NULL Pointer
READ_MSG_EXTENSION	0x00010007	Pointer to unsigned long	Pointer to SBYTE_ARRAY

3.5.6.1 <ioctlID> WRITE_MSG_EXTENSION. The <ioctlID> WRITE_MSG_EXTENSION provides a method for an application to transmit a message that exceeds the PASS_THRU_MSG <DataSize> limit of 4128 data bytes. WRITE_MSG_EXTENSION is not supported by a CAN_FD_PS channel.

Prior to calling to PassThruWriteMsgs, the application is required to call <ioctlID> WRITE_MSG_EXTENSION with an SBYTE_ARRAY that contains the data beyond 4128 bytes. The Pass-Thru Interface shall store the extended data in a temporary buffer. See Table 20.

Table 20: Write_Msg_Extension Parameters

Parameter	Typedef	Description
InputPtr	Pointer to SBYTE_ARRAY	Pointer to a buffer that contains the data that the application wishes to have appended to the next transmit message. typedef struct { unsigned long NumOfBytes; unsigned char *BytePtr; } SBYTE_ARRAY NumOfBytes = Number of Data Bytes that are in the extended buffer. BytePtr = Points to a buffer containing the data that is to be appended.
OutputPtr	NULL	Not used

Note: char = character, struct = structure, typedef = type definition.

PassThruWriteMsgs shall append the extended data to the transmit message only if the <DataSize> is correct (<DataSize> = 4128 bytes + NumOfBytes). If the <DataSize> is not correct, then the extended data shall be cleared from the temporary buffer.

Note: PassThruWriteMsgs only allows the first message of the message array to be extended by the application.

<iocltID> CLEAR_TX_QUEUE shall also clear the extended data

3.5.6.1.1 Return Values. For the most part the return values are defined as J2534 however; ERR_NOT_SUPPORTED has been modified to include the following conditions:

- NumOfBytes of the SBYTE_ARRAY exceeds 28640 bytes (32768 bytes – 4128 bytes).

3.5.6.2 <ioctlID> READ_MSG_EXTENSION. The Pass-Thru Interface shall store any data that was truncated by PassThruReadMsgs in a temporary buffer. The <IoclID> READ_MSG_EXTENSION provides a method for an application to read the data bytes of a message that was truncated by the last call to PassThruReadMsgs. See Table 21.

Truncated bytes = <DataSize> - 4128 bytes.

Table 21: Read Msg Extension Parameters

Parameter	Typedef	Description
InputPtr	Pointer to unsigned long	Pointer to the Message Index of a truncated message returned in the PassThruReadMsgs message array.
OutputPtr	Pointer to SBYTE_ARRAY	Pointer to a buffer where the Pass-Thru Interface can write the data bytes that were truncated by the PassThruReadMsgs function. typedef struct { unsigned long NumOfBytes; unsigned char *BytePtr; } SBYTE_ARRAY NumOfBytes = Number of Data Bytes that were truncated. BytePtr = Points to a buffer large enough for the Pass-Thru Interface to copy the data.

Note: <ioctlID> CLEAR_RX_QUEUE shall also clear the truncated data storage.
Note: char = character, struct = structure, typedef = type definition.

3.5.6.2.1 Return Values. The return value of ERR_NOT_SUPPORTED shall be used to identify the following conditions:

- Message Index set to a message that was not truncated.
- Message Index is greater than the NumMsgs returned by PassThruReadMsgs.
- Message Index does identify a message that was truncated.
- Message index identifies a truncated message but the NumOfBytes of the SBYTE_ARRAY does not equal the number of truncated bytes.

3.5.6.3 <ioctlID> CLEAR_TX_QUEUE. As defined by SAE J2534-1. However, also includes clearing the Extended Tx Data array if present.

3.5.6.4 <ioctlID> CLEAR_RX_QUEUE. As defined by SAE J2534-1. However, also includes clearing the Extended Rx Data arrays if present.

3.5.6.5 <ioctlID> SET_CONFIG and GET_CONFIG. See Table 22 and Table 24.

Table 22: SET_CONFIG and GET_CONFIG ParameterIDs

Parameter	ID Value	Description
CAN_FD_DATA_PHASE_RATE	0x10010	Specifies the baud rate for the data phase of a CAN FD frame in bits per second (bps). To minimize the possibility of Bus Errors, the Data phase rate should be set prior to setting J1962_PINS.
CAN_FD_TERMINATION	0x10012	The termination resistor shall be applied to across the DLC pins specified by J1962_PINS. To minimize the possibility of Bus Errors termination should be set prior to setting J1962_PINS. 0 = no termination (Default). 3 = 120 Ω termination.
CAN_FD_TYPE	0x10013	Switches between Bosch CAN FD and ISO CAN FD definitions. To minimize the possibility of Bus Errors, CAN FD type should be set prior to setting J1962_PINS. 0 = ISO CAN FD (Default). 1 = Bosch CAN FD.
CAN_FD_TC_DATA_LENGTH	0x10011	Supported by ISO15765_FD_PS only. The frame size the Pass-Thru Interface shall use when transmitting a CAN FD formatted segmented message. Transmit flag CAN_FD_FORMAT determines the frame format of the message. For the CAN 2.0 format, the frame size is always 8 bytes. 8 bytes (default), 12, 16, 20, 24, 32, 48, 64 bytes
N_CR_MAX	0x10014	Supported by ISO15765_FD_PS. The maximum time between consecutive frames of a ISO 15765 Message that is being received. If this time limit is detected, the Pass-Thru Device shall discard the message. Resolution: 1 µs. Default value: 150 000.
CAN_MIXED_FORMAT	0x8000	Supported by ISO15765_FD_PS. CAN_MIXED_FORMAT behaves as defined in SAE J2534-2. This implementation allows usage of CAN_FD_PS on an ISO15765_FD_PS channel.

Note: DLC = Data Link Connector.

Table 23 contains the valid combinations of <BaudRate> and CAN_FD_DATA_PHASE_RATE.

Table 23: Data Phase Baud Rates

<BaudRate>	Valid CAN_FD_DATA_PHASE_RATE Values (in bits per second (bps))				
500 000	500 000	1 000 000	2 000 000	4 000 000	5 000 000 (Default)
250 000	250 000	500 000	1 000 000	2 000 000	2 500 000 (Default)
125 000	125 000	250 000	500 000	1 000 000	1 250 000 (Default)

Table 24: SET_CONFIG and GET_CONFIG from CAN

Parameter Name	Default Setting	Description
J1962_PINS	0x0000	As defined by J2534, V04.04 Connects the CAN Transceiver to the specified pins of the SAE J1962 Connector. If the termination resistor is enabled it shall also be connected across the specified pins. Initial value for this <ioctlID> is 0x0000: Disconnected. Settings required by GM: 0x060E: Pins 6 and 14. 0x0109: Pins 1 and 9.
DATA_RATE	500 000	As defined by J2534, V04.04 DATA_RATE is initialized with the value of <BaudRate> from PassThruConnect (GET_CONFIG only).
CAN_MIXED_FORMAT	0 (OFF) Default	Supported by ISO15765_FD_PS only. Allows CAN_FD_PS messages to be transmitted and received on an ISO15765_FD_PS channel.
LOOPBACK	0 (OFF) Default 1 (ON)	As defined by J2534, V04.04
SYNC_JUMP_WIDTH	SAE J2284	As defined in SAE J2284 (GET_CONFIG only), (Arbitration Bit Timing).
BIT_SAMPLE_POINT	SAE J2284	As defined in SAE J2284 (GET_CONFIG only), (Arbitration Bit Timing).
ISO15765_PAD_VALUE	$CC	Supported by ISO15765_FD_PS only. The pad byte value is used extend the Last Frame of an ISO15765_FD_PS message to the next valid CAN FD frame size, (8, 12, 16, 20, 24, 32, 48 or 64 bytes).
ISO15765_BS	0	Supported by ISO15765_FD_PS only. As defined by J2534, V04.04.
ISO15765_STMIN	0	Supported by ISO15765_FD_PS only. As defined by J2534, V04.04.
BS_TX	$FFFF	Supported by ISO15765_FD_PS only. As defined by J2534, V04.04.
STMIN_TX	$FFFF	Supported by ISO15765_FD_PS only. As defined by J2534, V04.04.
ISO15765_WFT_MAX	0	Supported by ISO15765_FD_PS only. As defined by J2534, V04.04.

Note: BS = Block Size, STMIN = Minimum Frame Separation time.

4 Validation

Not applicable.

5 Provisions for Shipping

Not applicable.

6 Notes

6.1 Glossary.

Pass-Thru Device: Refers only to the physical hardware/circuitry portion of the J2534 Interface.

Pass-Thru Interface: Refers to all components of a J2534 Interface, including both hardware and software.

6.2 Acronyms, Abbreviations, and Symbols.

API	Application Programming Interface
bps	Bits Per Second
BRS	Baud Rate Switch
BS	Block Size (Frames)
CAN	Controller Area Network
CAN FD	Controller Area Network with Flexible Data
char	character
DLC	Data Link Connector
ED	Extended Data
ESI	Error State Indicator
FC	Flow Control
FD	Flexible Data-Rate
FF	First Frame
ID	Identification
ioctl	Input/Output Control
IND	Indication
ISO	International Standards Organization
msg	Message
PS	Pin Select
Rx	Receive
STMIN	Minimum Frame Separation time
struct	structure
Tx	Transmit
typedef	type definition

7 Additional Paragraphs

7.1 All parts or systems supplied to this standard must comply with the requirements of GMW3059, **Restricted and Reportable Substances.**

8 Coding System

This standard shall be referenced in other documents, drawings, etc., as follows:

GMW17753

9 Release and Revisions

This standard was originated in December 2016. It was first approved by Service Engineering in January 2017. It was first published in January 2017

Issue	Publication Date	Description (Organization)
1	JAN 2017	Initial publication.