Document Title	Specification of Bus Mirroring Protocol
Document Category	Protocol specification (PRS)
Document UID	1151
Document Status	published
Part of AUTOSAR Standard	Foundation
Part of Standard Release	R26-11

Document Change History			
Date	**Release**	**Changed by**	**Description**
2026-11-26	R26-11	AUTOSAR Release Management	• Initial release • Based on the bus mirroring protocol definition extracted from CP SWS Bus Mirroring

Disclaimer

This work (specification and/or software implementation) and the material contained in it, as released by AUTOSAR, is for the purpose of information only. AUTOSAR and the companies that have contributed to it shall not be liable for any use of the work.

The material contained in this work is protected by copyright and other types of intellectual property rights. The commercial exploitation of the material contained in this work requires a license to such intellectual property rights.

This work may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only. For any other purpose, no part of the work may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The work has been developed for automotive applications only. It has neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Table of Contents

1 Introduction and Overview 5
1.1 Protocol Purpose and Objectives . . . 5
1.2 Applicability of the Protocol . . . 5
1.2.1 Constraints and Assumptions . . . 5
1.2.2 Limitations . . . 5
1.3 Dependencies . . . 6
1.3.1 Dependencies to other Protocol Layers . . . 6
1.3.2 Dependencies to other Standards and Norms . . . 6
1.3.3 Dependencies to the Application Layer . . . 6
2 Use Cases 7
3 Related Documentation 9
3.1 Input Documents & Related Standards and Norms . . . 9
4 Protocol Requirements 10
5 Definition of Acronyms, Abbreviations, and Terms 11
5.1 Acronyms and Abbreviations . . . 11
5.2 Definition of Terms . . . 11
6 Protocol Specification 12
6.1 Serialization Protocol . . . 12
6.1.1 Header Layout . . . 12
6.1.1.1 ProtocolVersion . . . 13
6.1.1.2 SequenceNumber . . . 13
6.1.1.3 HeaderTimestamp . . . 13
6.1.1.4 DataLength . . . 14
6.1.2 Data Item Layout . . . 14
6.1.2.1 Timestamp . . . 15
6.1.2.2 NetworkStateAvailable . . . 15
6.1.2.3 FrameIDAvailable . . . 15
6.1.2.4 PayloadAvailable . . . 16
6.1.2.5 NetworkType . . . 16
6.1.2.6 NetworkID . . . 16
6.1.2.7 NetworkState . . . 17
6.1.2.7.1 NetworkStateCAN . . . 17
6.1.2.7.2 NetworkStateLIN . . . 18
6.1.2.7.3 NetworkStateFlexRay . . . 19
6.1.2.8 FrameID . . . 21
6.1.2.8.1 FrameIDCAN . . . 21
6.1.2.8.2 FrameIDLIN . . . 22
6.1.2.8.3 FrameIDFlexRay . . . 22
6.1.2.9 PayloadLength . . . 23

6.1.2.10 Payload 23
6.2 Status Protocol 23
6.2.1 Status Header Layout 24
6.2.1.1 SHProtocolVersion 24
6.2.2 Status Item Layout 24
6.2.2.1 SINetworkStateAvailable 25
6.2.2.2 SIFrameIDAvailable 26
6.2.2.3 SINetworkType 26
6.2.2.4 SINetworkID 26
6.2.2.5 SINetworkState 26
6.2.2.6 SIFrameID 26
A Change History of AUTOSAR Traceable Items 27
A.1 Traceable Item History of this Document according to AUTOSAR Release R26-11 27
A.1.1 Added Specification Items in R26-11 27
A.1.2 Changed Specification Items in R26-11 27
A.1.3 Deleted Specification Items in R26-11 27

1 Introduction and Overview

This protocol specification specifies the format, message sequences and semantics of the AUTOSAR Bus Mirroring Protocol.

The Bus Mirroring Protocol is used to serialize classic bus frames and status information. It is used by the Bus Mirroring BSW module.

1.1 Protocol Purpose and Objectives

When mirroring frames from a classic bus like CAN 2.0 to an Ethernet bus or other high-speed bus, it would be a waste of bandwidth to simply forward source frames of destination frames, because on high-speed buses the frame headers are typically much larger and therefore the overhead of the header compared to the payload would be considerable. Instead, several source frames are serialized into one destination frame, including status information.

When mirroring frames between similar buses, the data frames can be forwarded directly, saving the overhead of the serialization protocol. But in this case, status information about the source bus is still required on the destination bus, and therefore serialized into a specific status frame.

Both serialization formats are described by the AUTOSAR Bus Mirroring Protocol.

1.2 Applicability of the Protocol

The AUTOSAR Bus Mirroring Protocol is used on internal buses and on external links, to which a specific tester is connected that can interpret the AUTOSAR Bus Mirroring Protocol. Mirrored bus traffic is always uni-directionally forwarded from internal buses towards an external link.

1.2.1 Constraints and Assumptions

Mirrored bus traffic will always change the load of the destination bus. It is therefore essential to limit the amount of mirrored bus traffic by using filtering mechanisms similar to those implemented by the Bus Mirroring BSW module.

1.2.2 Limitations

The AUTOSAR Bus Mirroring Protocol does not support fragmentation, therefore mirroring of traffic is only possible if the frame size used on the source bus does not exceed the frame size supported by the destination bus.

1.3 Dependencies

1.3.1 Dependencies to other Protocol Layers

The AUTOSAR Bus Mirroring Protocol is self-contained, it does not depend on any other protocol.

1.3.2 Dependencies to other Standards and Norms

The AUTOSAR Bus Mirroring Protocol is self-contained, it does not depend on any other standards or norms.

1.3.3 Dependencies to the Application Layer

Applications do not access the information transported by the AUTOSAR Bus Mirroring Protocol.

2 Use Cases

ID	*Name*	*Description*
[UC_Mirror_00001]	CAN to CAN Mirroring	This use case supports the forwarding of frames from one or more CAN buses (source buses) to another CAN bus (destination bus). The content of the frames is not altered during forwarding, but the `Bus Mirroring` allows for remapping of the CAN IDs to avoid conflicts on the destination bus. If the destination bus is a CAN FD bus, the source buses may also be CAN FD buses.
[UC_Mirror_00002]	LIN to CAN Mirroring	This use case behaves like `[UC_Mirror_00001]` except frames from one or more LIN source buses are forwarded to a destination CAN bus by the `Bus Mirroring`, which assigns a CAN ID to each LIN frame.
[UC_Mirror_00003]	CAN, LIN, and FlexRay Mirroring to Ethernet	Compared to the previous two use cases, the bandwidth of Ethernet is much higher, and the available payload size of an Ethernet frame is also much larger than the possible frame lengths on the source buses. Therefore, frames of different source buses can be collected and forwarded together. To ensure timing consistency, the `Bus Mirroring` will wrap the forwarded frames with a `Bus Mirroring Protocol` that includes fields for the current time, the source bus, the identifier, and the content of the frames.
[UC_Mirror_00004]	CAN, LIN, and FlexRay Mirroring to FlexRay	Mirroring to FlexRay is comparable to Ethernet `[UC_Mirror_00003]`, in that the FlexRay bus can carry larger frames. Therefore, when routing to FlexRay, an encoding scheme similar to Ethernet will be applied.

▽

△

[UC_Mirror_00005]	CAN, LIN, and FlexRay Mirroring to a proprietary bus	Mirroring to a proprietary bus uses the same protocol that is used for mirroring to Ethernet [UC_Mirror_00003].

3 Related Documentation

3.1 Input Documents & Related Standards and Norms

[1] Technical Report on AUTOSAR Features
AUTOSAR_FO_TR_Features

[2] Glossary
AUTOSAR_FO_TR_Glossary

[3] Specification of Bus Mirroring
AUTOSAR_CP_SWS_BusMirroring

[4] Specification of CAN Driver
AUTOSAR_CP_SWS_CANDriver

[5] Specification of FlexRay Driver
AUTOSAR_CP_SWS_FlexRayDriver

4 Protocol Requirements

No requirements are maintained for the `Bus Mirroring Protocol`. Instead, Chapter 2 lists the use cases for this protocol. And [1, TR Features] lists the features of AUTOSAR, including the `Bus Mirroring`.

5 Definition of Acronyms, Abbreviations, and Terms

5.1 Acronyms and Abbreviations

This document does not define any specific acronyms or abbreviations. A general list of acronyms and abbreviations is available in the AUTOSAR glossary [2].

The table below contains all document-independant acronyms or abbreviations that are defined in the AUTOSAR glossary [2].

Acronym/ Abbreviation	Description
PDU	Protocol Data Unit

Table 5.1: Acronyms and Abbreviations used from AUTOSAR Glossary

5.2 Definition of Terms

The table below contains all definitions of terms that are used in this document.

Glossary Term	Explanation
Bus	A communication channel, synonymous to Network in this specification
Bus Mirroring	Bus Mirroring implementation, see e.g. [3, CP SWS Bus Mirroring]. Uses the Bus Mirroring Protocol defined in this document for serialization of frames and status information.
Bus Mirroring Protocol	The protocol defined in this document, which is used by the Bus Mirroring implementation for serialization of frames and status information.
Bus Mirroring Serialization Protocol	The part of the Bus Mirroring Protocol that describes the serialization of frames for destination buses that support large frames.
Bus Mirroring Status Protocol	The part of the Bus Mirroring Protocol that describes the serialization of status information for destination buses where the frame size is similar size to that of the source buses.
Destination Bus	The Bus to which (serialized) PDUs are transmitted
Meta Data	Meta data transferred alongside a PDU, consisting of a set of meta data items
Meta Data Item	A single item of meta data of defined type and size
Network	A communication channel, synonymous to bus in this specification
Network ID	A unique identifier for each source bus, by which it is represented on the destination bus
Source Bus	The Bus from which PDUs are received

Table 5.2: Glossary Terms

6 Protocol Specification

6.1 Serialization Protocol

The protocol that is applied by the Bus Mirroring for IP, FlexRay, and proprietary destination buses is shown in Figure 6.1, in this example for an Ethernet destination bus.



Figure 6.1: Bus Mirroring Serialization Protocol

The protocol consists of a header (see Chapter 6.1.1) followed by several data items (see Chapter 6.1.2).

In the tables and descriptions of this section, the byte numbers increase in the same sequence as the bytes are transmitted on the destination bus, starting from 0. The bit numbers decrease, the most significant bit of a byte being bit 7 and the least significant bit 0.

6.1.1 Header Layout

Every destination frame starts with a header, which is shown in Figure 6.2.

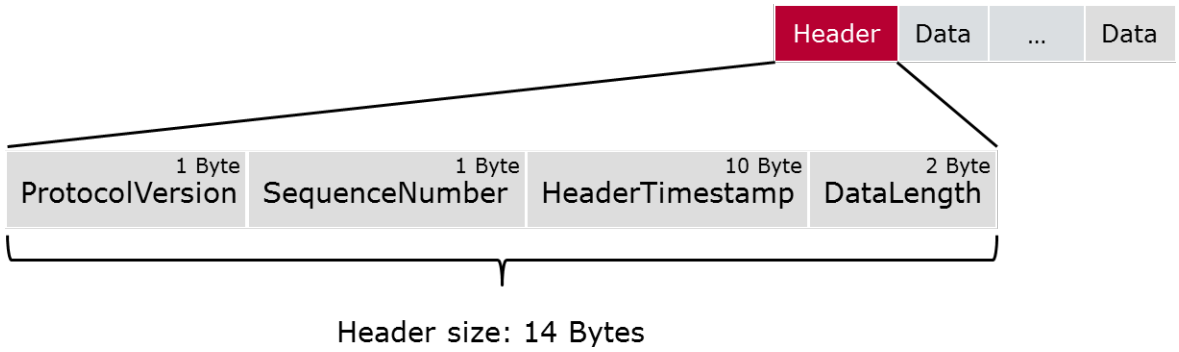


Figure 6.2: Bus Mirroring Serialization Protocol Protocol Header

[FO_PRS_Mirror_00055] Serialization Header Layout ⌈The header of a Bus Mirroring destination frame shall contain the following fields in this order:

1. ProtocolVersion
2. SequenceNumber
3. HeaderTimestamp
4. DataLength

⌋

The fields of the header are described in detail in the following subsections.

6.1.1.1 ProtocolVersion

[FO_PRS_Mirror_00056] Definition of `ProtocolVersion` ⌈The `ProtocolVersion` shall indicate the layout of the header and the data items. The layout currently defined in this section is identified by `ProtocolVersion` 1. The range $[2 \,..\, 127]$ is reserved for future extensions of the AUTOSAR defined protocol, the range $[128 \,..\, 255]$ is available for customer specific protocols.⌋

The protocol version allows the tester tool to interpret the protocol correctly, and to enable different layouts of the protocol.

[FO_PRS_Mirror_00057] Width of `ProtocolVersion` ⌈The width of the `ProtocolVersion` field shall be 8 bits.⌋

6.1.1.2 SequenceNumber

[FO_PRS_Mirror_00058] Definition of `SequenceNumber` ⌈The `SequenceNumber` shall increase with each transmission of a destination frame. It shall start from 0 when a `destination bus` is (re-)enabled.⌋

The sequence number allows the tester tool to identify lost destination frames.

[FO_PRS_Mirror_00059] Width of `SequenceNumber` ⌈The width of the `SequenceNumber` field shall be 8 bits.⌋

This means that the `SequenceNumber` will wrap around to 0 after it reached 255. A tester has to cope with this behavior and still sort the frames correctly.

6.1.1.3 HeaderTimestamp

[FO_PRS_Mirror_00060] Definition of `HeaderTimestamp` ⌈The `HeaderTimestamp` shall reflect the time when collection of data items into the destination frame started. This time shall be given as the absolute number of seconds and nanoseconds since January 1st of 1970.⌋

[FO_PRS_Mirror_00061] Width of `HeaderTimestamp` ⌈The total width of the `HeaderTimestamp` field shall be 10 bytes, where the seconds take the upper 48 Bits and the nanoseconds take the lower 32 Bits. Both elements of the the `HeaderTimestamp` field shall be encoded in network byte order (MSB first).⌋

Table 6.1 shows the layout of the `HeaderTimestamp`.

HeaderTimestamp									
Byte 0	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Byte 5**	**Byte 6**	**Byte 7**	**Byte 8**	**Byte 9**
Seconds (48 bits, MSB first)						Nanoseconds (32 bits, MSB first)			

Table 6.1: Layout of `HeaderTimestamp`

6.1.1.4 DataLength

[FO_PRS_Mirror_00062] Definition of `DataLength` ⌈The `DataLength` shall give the number of bytes following the header. It is the sum of the length of all data items in the destination frame.⌋

[FO_PRS_Mirror_00063] Width of `DataLength` ⌈The width of the `DataLength` field shall be 16 bits. It shall be encoded in network byte order (MSB first).⌋

6.1.2 Data Item Layout

Every source frame is placed in a data item, which is shown in Figure 6.3.

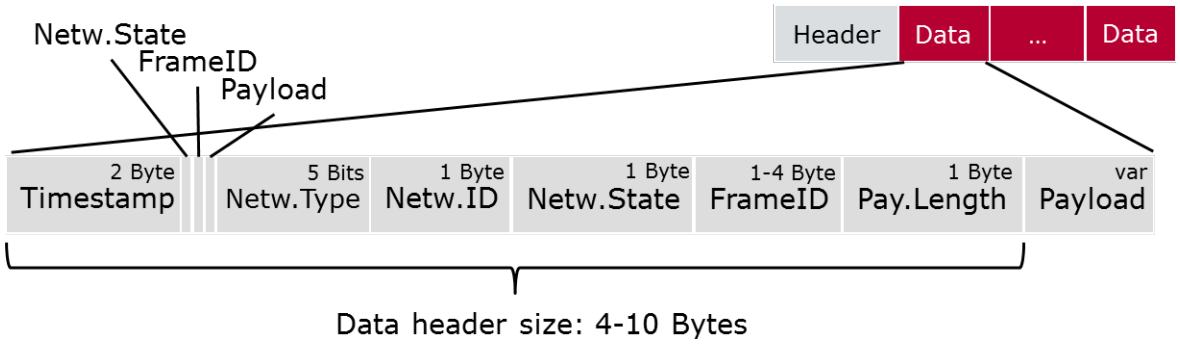


Figure 6.3: `Bus Mirroring Serialization Protocol` Data Item

[FO_PRS_Mirror_00064] Serialization Data Item Layout ⌈Data items of a `Bus Mirroring` destination frame shall contain the following fields in this order:

1. `Timestamp`
2. `NetworkStateAvailable`
3. `FrameIDAvailable`
4. `PayloadAvailable`
5. `NetworkType`
6. `NetworkID`

7. NetworkState (optional)
8. FrameID (optional)
9. PayloadLength (optional)
10. Payload (optional)

⌋

The fields of the data item are described in detail in the following sub sections.

6.1.2.1 Timestamp

[FO_PRS_Mirror_00065] Definition of Timestamp ⌈The Timestamp shall reflect the temporal offset of the source frame reception from the HeaderTimestamp, i.e. the time that passed since collection of data items into the destination frame started. It shall be given in multiples of $10\,\mu s$.⌋

[FO_PRS_Mirror_00066] Width of Timestamp ⌈The width of the Timestamp field shall be 16 bits. It shall be encoded in network byte order (MSB first).⌋

6.1.2.2 NetworkStateAvailable

[FO_PRS_Mirror_00067] Definition of NetworkStateAvailable ⌈The NetworkStateAvailable shall indicate whether the field NetworkState is present in the data item. If NetworkStateAvailable is 1, that field shall be present. If it is 0, that field shall be omitted.⌋

[FO_PRS_Mirror_00068] Width of NetworkStateAvailable ⌈The width of the NetworkStateAvailable field shall be 1 bit.⌋

6.1.2.3 FrameIDAvailable

[FO_PRS_Mirror_00069] Definition of FrameIDAvailable ⌈The FrameIDAvailable shall indicate whether the field FrameID is present in the data item. If FrameIDAvailable is 1, that field shall be present. If it is 0, that field shall be omitted.⌋

[FO_PRS_Mirror_00070] Width of FrameIDAvailable ⌈The width of the FrameIDAvailable field shall be 1 bit.⌋

6.1.2.4 PayloadAvailable

[FO_PRS_Mirror_00071] Definition of `PayloadAvailable` ⌈The `PayloadAvailable` shall indicate whether the fields `PayloadLength` and `Payload` are present in the data item. If `PayloadAvailable` is 1, these fields shall be present. If it is 0, these fields shall be omitted.⌋

[FO_PRS_Mirror_00072] Width of `PayloadAvailable` ⌈The width of the `PayloadAvailable` field shall be 1 bit.⌋

6.1.2.5 NetworkType

[FO_PRS_Mirror_00073] Definition of `NetworkType` ⌈The `NetworkType` shall indicate the type of the `source bus`.⌋

[FO_PRS_Mirror_00074] Width and Reserved Bits of `NetworkType` ⌈The width of the `NetworkType` field shall be 5 bits, the possible values are defined in [FO_PRS_Mirror_00170]. The range $[5 \,..\, 15]$ is reserved for future extensions of the AUTOSAR defined protocol, the range $[16 \,..\, 31]$ is available for customer specific `bus` types.⌋

[FO_PRS_Mirror_00170] Values of `NetworkType` ⌈

Invalid	0
Network Type	**Numerical**
CAN or CAN FD	1
LIN	2
FlexRay	3
Ethernet	4

Table 6.2

⌋

6.1.2.6 NetworkID

[FO_PRS_Mirror_00075] Definition of `NetworkID` ⌈The `NetworkID` shall uniquely identify a `bus` of a certain `NetworkType`.⌋

This means that the same `NetworkID` can be used for two `buses` with different `NetworkTypes`, but not with the same `NetworkType`.

[FO_PRS_Mirror_00076] Width of `NetworkID` ⌈The width of the `NetworkID` field shall be 8 bits.⌋

6.1.2.7 NetworkState

[FO_PRS_Mirror_00077] Definition of `NetworkState` ⌈The `NetworkState` shall provide information about the `source bus` state. It shall only be present when the source bus state has changed since the last time it was reported or frames were lost since the last successful transmission on the `destination bus`. The presence of the `NetworkState` shall be indicated by `NetworkStateAvailable`.⌋

[FO_PRS_Mirror_00078] Width and Layout of `NetworkState` ⌈The width of the `NetworkState` field shall be 8 bits, the layout is `bus` specific and is defined separately for each `bus` as `NetworkStateCAN`, `NetworkStateLIN`, and `NetworkStateFlexRay`.⌋

Please note that bits 6 and 7 of the `NetworkState` field are the same for all currently defined `bus` types.

[FO_PRS_Mirror_00079] Frames Lost Bit of `NetworkState` ⌈Bit 7 (the most significant bit) of the `NetworkState` shall always contain the Frames Lost state. This is a sporadic error that is not related to the source frame that is reported in the same data item, but shall not be reported in a separate data item. The Frames Lost state shall be set once to 1 after one or more source frames that passed the filters were lost because the queue of the `destination bus` was full or the transmission failed. Afterwards it shall be set to 0 again.⌋

Please note that re-transmission after failed transmission of destination frames is not required. The frame loss indication is sufficient to cover this case.

[FO_PRS_Mirror_00080] Bus Online Bit of `NetworkState` ⌈Bit 6 of the `NetworkState` shall always contain the Bus Online state. This is a continuous state that is not related to the source frame that is reported in the same data item, and may also be reported in a data item where the `FrameIDAvailable` and `PayloadAvailable` fields are set to 0. The Bus Online state shall be set to 1 when the `source bus` is online, i.e. when both the controller and the transceiver are able to communicate. Otherwise it shall be set to 0.⌋

6.1.2.7.1 NetworkStateCAN

The layout of the `NetworkState` for a CAN or CAN FD bus is shown in Table 6.3.

NetworkState							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Frames Lost	Bus Online	Error-Passive	Bus-Off	Tx error counter, divided by 8			

Table 6.3: Layout of CAN `NetworkState`

[FO_PRS_Mirror_00081] Error-Passive Bit of `NetworkStateCAN` ⌈Bit 5 of the `NetworkStateCAN` shall contain the Error-Passive state. This is a continuous state that is not related to the source frame that is reported in the same data item, and may also be reported in a data item where the `FrameIDAvailable` and `PayloadAvailable` fields are set to 0.

The Error-Passive state shall be set to 1 when the CAN controller is in the Error-Passive state, and to 0 when it is in the Error-Active or Bus-Off state.⌋

[FO_PRS_Mirror_00082] Bus-Off Bit of `NetworkStateCAN` ⌈Bit 4 of the `NetworkStateCAN` shall contain the Bus-Off state. This is a continuous state that is not related to the source frame that is reported in the same data item, and may also be reported in a data item where the `FrameIDAvailable` and `PayloadAvailable` fields are set to 0.

The Bus-Off state shall be set to 1 when the CAN controller is in the Bus-Off state, and to 0 when it is in the Error-Active or Error-Passive state.⌋

[FO_PRS_Mirror_00083] Tx Error Counter Bits of `NetworkStateCAN` ⌈Bits 3 – 0 of the `NetworkStateCAN` shall contain the Tx error counter of the can controller divided by 8. This is a continuous state that is not related to the source frame that is reported in the same data item, and may also be reported in a data item where the `FrameIDAvailable` and `PayloadAvailable` fields are set to 0.⌋

6.1.2.7.2 NetworkStateLIN

The layout of the `NetworkState` for a LIN bus is shown in Table 6.4.

NetworkState							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Frames Lost	Bus Online	*reserved*		Header Tx Error	Tx Error	Rx Error	Rx No Response

Table 6.4: Layout of LIN `NetworkState`

[FO_PRS_Mirror_00084] Reserved Bits of `NetworkStateLIN` ⌈Bits 5 and 4 of the `NetworkStateLIN` are currently reserved. They shall always be set to 0.⌋

[FO_PRS_Mirror_00085] Header Tx Error Bit of `NetworkStateLIN` ⌈Bit 3 of the `NetworkStateLIN` shall contain the Header Tx Error state. This is an error that is related to the source frame that is reported in the same data item.

The Header Tx Error state shall be set to 1 when the LIN controller detected an error during transmission of a LIN header. Otherwise it shall be set to 0.⌋

[FO_PRS_Mirror_00086] Tx Error Bit of `NetworkStateLIN` ⌈Bit 2 of the `NetworkStateLIN` shall contain the Tx Error state. This is an error that is related to the source frame that is reported in the same data item.

The Tx Error state shall be set to 1 when the LIN controller detected an error during transmission of a LIN frame. Otherwise it shall be set to 0.⌋

[FO_PRS_Mirror_00087] Rx Error Bit of `NetworkStateLIN` ⌈Bit 1 of the `NetworkStateLIN` shall contain the Rx Error state. This is an error that is related to the source frame that is reported in the same data item.

The Rx Error state shall be set to 1 when the LIN controller detected an error during reception of a LIN frame. Otherwise it shall be set to 0.⌋

[FO_PRS_Mirror_00088] Rx No Response Bit of `NetworkStateLIN` ⌈Bit 0 of the `NetworkStateLIN` shall contain the Header Rx No Response state. This is an error that is related to the source frame that is reported in the same data item.

The Rx No Response state shall be set to 1 when the LIN controller did not receive the expected LIN frame after transmission of a LIN header. Otherwise it shall be set to 0.⌋

6.1.2.7.3 NetworkStateFlexRay

The layout of the `NetworkState` for a FlexRay bus is shown in Table 6.5.

NetworkState							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Frames Lost	Bus Online	Bus Synchronous	Normal Active	Syntax Error	Content Error	Boundary Violation	Tx Conflict

Table 6.5: Layout of FlexRay `NetworkState`

[FO_PRS_Mirror_00089] Bus Synchronous Bit of `NetworkStateFlexRay` ⌈Bit 5 of the `NetworkStateFlexRay` shall contain the Bus Synchronous state. This is a continuous state that is not related to the source frame that is reported in the same data item, and may also be reported in a data item where the `FrameIDAvailable` and `PayloadAvailable` fields are set to 0.

The Bus Synchronous state shall be set to 1 when all FlexRay controllers connected to that `bus` are synchronous to the network time. Otherwise it shall be set to 0.⌋

[FO_PRS_Mirror_00090] Normal Active Bit of `NetworkStateFlexRay` ⌈Bit 4 of the `NetworkStateFlexRay` shall contain the Normal Active state. This is a continuous state that is not related to the source frame that is reported in the same data item, and may also be reported in a data item where the `FrameIDAvailable` and `PayloadAvailable` fields are set to 0.

The Normal Active state shall be set to 1 when all FlexRay controllers connected to that bus are synchronous and in the normal active state. Otherwise it shall be set to 0.⌋

[FO_PRS_Mirror_00091] Syntax Error Bit of NetworkStateFlexRay ⌈Bit 3 of the NetworkStateFlexRay shall contain the Syntax Error state. This is an aggregated error flag of the FlexRay channels that is related to the channel assignment of the FrameID, but not to a source frame and its FrameID that is reported in the same data item. It may also be reported in a data item where the PayloadAvailable field is set to 0 and the FrameIDAvailable is set to 1 with the slot valid flag of the FrameID set to 0.

The Syntax Error state shall be set to 1 once after a FlexRay controller detected a syntax error. Otherwise it shall be set to 0.⌋

[FO_PRS_Mirror_00092] Content Error Bit of NetworkStateFlexRay ⌈Bit 2 of the NetworkStateFlexRay shall contain the Content Error state. This is an aggregated error flag of the FlexRay channels that is related to the channel assignment of the FrameID, but not to a source frame and its FrameID that is reported in the same data item. It may also be reported in a data item where the PayloadAvailable field is set to 0 and the FrameIDAvailable is set to 1 with the slot valid flag of the FrameID set to 0.

The Content Error state shall be set to 1 once after a FlexRay controller detected a content error. Otherwise it shall be set to 0.⌋

[FO_PRS_Mirror_00093] Boundary Violation Bit of NetworkStateFlexRay ⌈Bit 1 of the NetworkStateFlexRay shall contain the Boundary Violation state. This is an aggregated error flag of the FlexRay channels that is related to the channel assignment of the FrameID, but not to a source frame and its FrameID that is reported in the same data item. It may also be reported in a data item where the PayloadAvailable field is set to 0 and the FrameIDAvailable is set to 1 with the slot valid flag of the FrameID set to 0.

The Boundary Violation state shall be set to 1 once after a FlexRay controller detected a boundary violation. Otherwise it shall be set to 0.⌋

[FO_PRS_Mirror_00094] Tx Conflict Bit of NetworkStateFlexRay ⌈Bit 0 of the NetworkStateFlexRay shall contain the Tx Conflict state. This is an error that is related to the previous source frame that was reported with the same FrameID and is always reported in a data item where the FrameIDAvailable field is set to 1 and the PayloadAvailable field is set to 0.

The Tx Conflict state shall be set to 1 when a FlexRay controller detected a transmission conflict. Otherwise it shall be set to 0.⌋

6.1.2.8 FrameID

[FO_PRS_Mirror_00095] Definition of `FrameID` ⌈The `FrameID` shall provide the identification of the source frame. This identification shall be unique for one `source bus` identified by `NetworkType` and `NetworkID`. The `FrameID` may be omitted when reporting a `source bus` state change, the presence shall be indicated by `FrameIDAvailable`.⌋

[FO_PRS_Mirror_00096] Width of `FrameID` ⌈The width and layout of the `FrameID` field is `bus` specific and is defined separately for each `bus` as `FrameIDCAN`, `FrameIDLIN`, and `FrameIDFlexRay`.⌋

6.1.2.8.1 FrameIDCAN

The layout of the `FrameID` for a CAN or CAN FD bus is shown in Table 6.6.

FrameID						
Byte 0				Byte 1	Byte 2	Byte 3
Bit 7	Bit 6	Bit 5	Bits 4 .. 0			
Ext.ID/ Std.ID	FD/ 2.0	*res.*	CAN ID (Bits 28 .. 24)	CAN ID (Bits 23 .. 16)	CAN ID (Bits 15 .. 8)	CAN ID (Bits 7 .. 0)

Table 6.6: Layout of CAN `FrameID`

Please note: The layout of the `FrameIDCAN` corresponds to the `Can_IdType` defined in the [4, CP SWS CAN Driver].

[FO_PRS_Mirror_00097] Width of `FrameIDCAN` ⌈The width of the `FrameIDCAN` field shall be 4 bytes.⌋

[FO_PRS_Mirror_00098] Extended CAN ID Bit of `FrameIDCAN` ⌈Bit 7 of Byte 0 of the `FrameIDCAN` shall be set to 1 for an Extended CAN ID and to 0 for a Standard CAN ID.⌋

[FO_PRS_Mirror_00099] CAN FD Bit of `FrameIDCAN` ⌈Bit 6 of Byte 0 of the `FrameIDCAN` shall be set to 1 for a CAN FD frame and to 0 for a CAN 2.0 frame.⌋

[FO_PRS_Mirror_00100] Reserved Bits of `FrameIDCAN` ⌈Bit 5 of Byte 0 of the `FrameIDCAN` is currently reserved. It shall always be set 0.⌋

[FO_PRS_Mirror_00101] CAN ID Bits of `FrameIDCAN` ⌈Bits 4 – 0 of Byte 0 and Bytes 1 – 3 of the `FrameIDCAN` shall contain the CAN ID of the reported CAN frame in network byte order (MSB first).⌋

6.1.2.8.2 FrameIDLIN

The layout of the FrameID for a LIN bus is shown in Table 6.7.

FrameID
Byte 0
LIN PID

Table 6.7: Layout of LIN FrameID

[FO_PRS_Mirror_00102] Width of FrameIDLIN ⌈The width of the FrameIDLIN field shall be 1 byte.⌋

[FO_PRS_Mirror_00103] LIN PID Bits of FrameIDLIN ⌈Byte 0 of the FrameIDLIN shall contain the LIN PID of the reported LIN frame.⌋

6.1.2.8.3 FrameIDFlexRay

The layout of the FrameID for a FlexRay bus is shown in Table 6.8.

FrameID						
Byte 0					Byte 1	Byte 2
Bit 7	**Bit 6**	**Bit 5 .. 4**	**Bit 3**	**Bits 2 .. 0**		
Channel B	Channel A	*reserved*	Slot Valid	Slot ID (Bits 10 .. 8)	Slot ID (Bits 7 .. 0)	Cycle

Table 6.8: Layout of FlexRay FrameID

[FO_PRS_Mirror_00104] Width of FrameIDFlexRay ⌈The width of the FrameIDFlexRay field shall be 3 bytes.⌋

[FO_PRS_Mirror_00105] Channel Assignment Bits of FrameIDFlexRay ⌈Bits 7 – 6 of Byte 0 of the FrameIDFlexRay shall contain the channel assignment of the reported FlexRay frame. Bit 7 shall be set to 1 if the reported FlexRay frame is available on channel B of the FlexRay controller, otherwise it shall be set to 0. Bit 6 shall be set to 1 if the reported FlexRay frame is available on channel A of the FlexRay controller, otherwise it shall be set to 0. A reported FlexRay frame is either assigned exclusively to channel A or B or to both channels.⌋

Please note: The layout of the channel assignment corresponds to the Fr_ChannelType defined in the [5, CP SWS FlexRay Driver].

[FO_PRS_Mirror_00106] Reserved Bits of FrameIDFlexRay ⌈Bits 5 – 4 of Byte 0 of the FrameIDFlexRay are currently reserved. They shall always be set 0.⌋

[FO_PRS_Mirror_00159] Valid Cycle and Slot Bit of `FrameIDFlexRay` ⌈Bit 3 of Byte 0 of the `FrameIDFlexRay` shall contain a flag indicating whether the reported slot ID and cycle are valid (flag is 1) or unused (flag is 0). It shall only be set to 0 when an aggregated error of the FlexRay channels is reported independently of a source frame or transmission conflict. Otherwise it shall always be set to 1.⌋

[FO_PRS_Mirror_00107] Slot ID Bits of `FrameIDFlexRay` ⌈Bits 2 – 0 of Byte 0 and Byte 1 of the `FrameIDFlexRay` shall contain the slot ID of the reported FlexRay frame in network byte order (MSB first).⌋

[FO_PRS_Mirror_00108] Cycle Bits of `FrameIDFlexRay` ⌈Byte 2 of the `FrameIDFlexRay` shall contain the cycle in which the reported FlexRay frame was sent or received.⌋

Please note: For received frames and for frames sent in the static segment, the cycle is always reliable. For frames sent in the dynamic segment, the actual cycle cannot be known in advance, because the frame might not be transmitted in the planned cycle.

6.1.2.9 PayloadLength

[FO_PRS_Mirror_00109] Definition of `PayloadLength` ⌈The `PayloadLength` shall provide the length of the payload of the source frame. It may be omitted when reporting a `source bus` state change, the presence shall be indicated by `PayloadAvailable`.⌋

[FO_PRS_Mirror_00110] Width of `PayloadLength` ⌈The width of the `PayloadLength` field shall be 8 bits.⌋

6.1.2.10 Payload

[FO_PRS_Mirror_00111] Definition of `Payload` ⌈The `Payload` shall provide the actual payload of the source frame. It may be omitted when reporting a `source bus` state change, the presence shall be indicated by `PayloadAvailable`.⌋

[FO_PRS_Mirror_00112] Width of `Payload` ⌈The width of the `Payload` field shall correspond to the reported source frame. The maximum values are 8 bytes for LIN and CAN 2.0, 64 bytes for CAN FD, and 254 for FlexRay.⌋

6.2 Status Protocol

The protocol that is applied by the `Bus Mirroring` for transmission of status frames on CAN consists of a header (see Chapter 6.2.1) followed by several data items (see Chapter 6.2.2).

In the tables and descriptions of this section, the byte numbers increase in the same sequence as the bytes are transmitted on the destination bus, starting from 0. The bit numbers decrease, the most significant bit of a byte being bit 7 and the least significant bit 0.

6.2.1 Status Header Layout

Every status frame starts with a header, which is shown in Figure 6.4.

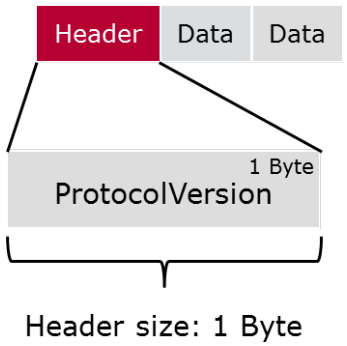


Figure 6.4: Bus Mirroring Status Protocol Header

[FO_PRS_Mirror_00127] Status Header Layout ⌈The header of a Bus Mirroring status frame shall contain the SHProtocolVersion.⌋

6.2.1.1 SHProtocolVersion

[FO_PRS_Mirror_00128] Definition of SHProtocolVersion ⌈The SHProtocol-Version shall be identical to the ProtocolVersion of a serialized destination frame.⌋

The ProtocolVersion is defined in Chapter 6.1.1.1.

6.2.2 Status Item Layout

Every source bus state is placed in a status item, which is shown in Figure 6.5.

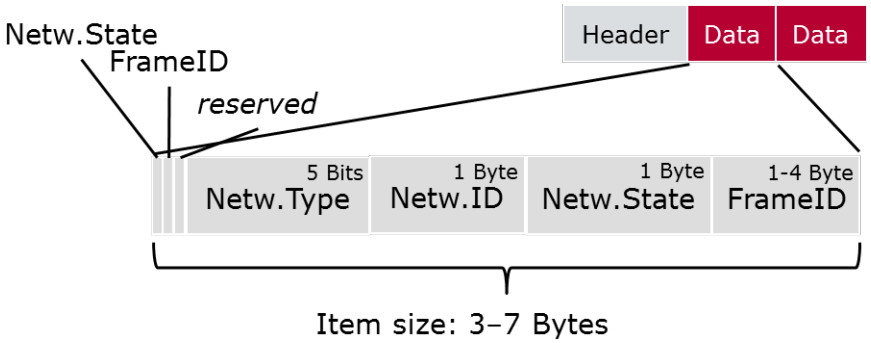


Figure 6.5: `Bus Mirroring Status Protocol` Item

[FO_PRS_Mirror_00129] Status Item Layout ⌈Status items of a `Bus Mirroring` status frame shall contain the following fields in this order:

1. `SINetworkStateAvailable`
2. `SIFrameIDAvailable`
3. *reserved*
4. `SINetworkType`
5. `SINetworkID`
6. `SINetworkState`
7. `SIFrameID` (optional)

⌋

[FO_PRS_Mirror_00132] Reserved Bits of Status Item ⌈Bit 5 of Byte 0 of the status item is currently reserved and shall always be set to 0.⌋

The fields of the status item are described in detail in the following sub sections.

6.2.2.1 SINetworkStateAvailable

[FO_PRS_Mirror_00149] Definition of `SINetworkStateAvailable` ⌈The layout and semantics of the `SINetworkStateAvailable` shall be identical to the `NetworkStateAvailable` used in a serialized data item. It shall always be set to 1.⌋

The `NetworkStateAvailable` is defined in Chapter 6.1.2.2. The receiver of a `Bus Mirroring` status frame can use the `SINetworkStateAvailable` to check for a valid status item: If this bit is 0, the remainder of the frame can be ignored, it is probably just padding.

6.2.2.2 SIFrameIDAvailable

[FO_PRS_Mirror_00131] Definition of `SIFrameIDAvailable` ⌈The layout and semantics of the `SIFrameIDAvailable` shall be identical to the `FrameIDAvailable` used in a serialized data item.⌋

The `FrameIDAvailable` is defined in Chapter 6.1.2.3.

6.2.2.3 SINetworkType

[FO_PRS_Mirror_00133] Definition of `SINetworkType` ⌈The layout and semantics of the `SINetworkType` shall be identical to the `NetworkType` used in a serialized data item.⌋

The `NetworkType` is defined in Chapter 6.1.2.5.

6.2.2.4 SINetworkID

[FO_PRS_Mirror_00134] Definition of `SINetworkID` ⌈The layout and semantics of the `SINetworkID` shall be identical to the `NetworkID` used in a serialized data item.⌋

The `NetworkID` is defined in Chapter 6.1.2.6.

6.2.2.5 SINetworkState

[FO_PRS_Mirror_00135] Definition of `SINetworkState` ⌈The layout and semantics of the `SINetworkState` shall be identical to the `NetworkState` used in a serialized data item.⌋

The `NetworkState` is defined in Chapter 6.1.2.7.

6.2.2.6 SIFrameID

[FO_PRS_Mirror_00136] Definition of `SIFrameID` ⌈The layout and semantics of the `SIFrameID` shall be identical to the `FrameID` used in a serialized data item.⌋

Please note: The `SIFrameID` is only used to report transmission or reception errors. In contrast to the the `FrameID`, the `SIFrameID` is never related to mirrored data, because the latter is provided by the directly mirrored data frames.

The `FrameID` is defined in Chapter 6.1.2.8.

A Change History of AUTOSAR Traceable Items

Please note that the lists in this chapter also include traceable items that have been removed from the specification in a later version. These items do not appear as hyperlinks in the document.

A.1 Traceable Item History of this Document according to AUTOSAR Release R26-11

A.1.1 Added Specification Items in R26-11

[FO_PRS_Mirror_00055] [FO_PRS_Mirror_00056] [FO_PRS_Mirror_00057] [FO_PRS_Mirror_00058] [FO_PRS_Mirror_00059] [FO_PRS_Mirror_00060] [FO_PRS_Mirror_00061] [FO_PRS_Mirror_00062] [FO_PRS_Mirror_00063] [FO_PRS_Mirror_00064] [FO_PRS_Mirror_00065] [FO_PRS_Mirror_00066] [FO_PRS_Mirror_00067] [FO_PRS_Mirror_00068] [FO_PRS_Mirror_00069] [FO_PRS_Mirror_00070] [FO_PRS_Mirror_00071] [FO_PRS_Mirror_00072] [FO_PRS_Mirror_00073] [FO_PRS_Mirror_00074] [FO_PRS_Mirror_00075] [FO_PRS_Mirror_00076] [FO_PRS_Mirror_00077] [FO_PRS_Mirror_00078] [FO_PRS_Mirror_00079] [FO_PRS_Mirror_00080] [FO_PRS_Mirror_00081] [FO_PRS_Mirror_00082] [FO_PRS_Mirror_00083] [FO_PRS_Mirror_00084] [FO_PRS_Mirror_00085] [FO_PRS_Mirror_00086] [FO_PRS_Mirror_00087] [FO_PRS_Mirror_00088] [FO_PRS_Mirror_00089] [FO_PRS_Mirror_00090] [FO_PRS_Mirror_00091] [FO_PRS_Mirror_00092] [FO_PRS_Mirror_00093] [FO_PRS_Mirror_00094] [FO_PRS_Mirror_00095] [FO_PRS_Mirror_00096] [FO_PRS_Mirror_00097] [FO_PRS_Mirror_00098] [FO_PRS_Mirror_00099] [FO_PRS_Mirror_00100] [FO_PRS_Mirror_00101] [FO_PRS_Mirror_00102] [FO_PRS_Mirror_00103] [FO_PRS_Mirror_00104] [FO_PRS_Mirror_00105] [FO_PRS_Mirror_00106] [FO_PRS_Mirror_00107] [FO_PRS_Mirror_00108] [FO_PRS_Mirror_00109] [FO_PRS_Mirror_00110] [FO_PRS_Mirror_00111] [FO_PRS_Mirror_00112] [FO_PRS_Mirror_00127] [FO_PRS_Mirror_00128] [FO_PRS_Mirror_00129] [FO_PRS_Mirror_00131] [FO_PRS_Mirror_00132] [FO_PRS_Mirror_00133] [FO_PRS_Mirror_00134] [FO_PRS_Mirror_00135] [FO_PRS_Mirror_00136] [FO_PRS_Mirror_00149] [FO_PRS_Mirror_00159] [FO_PRS_Mirror_00170]

A.1.2 Changed Specification Items in R26-11

none

A.1.3 Deleted Specification Items in R26-11

none