Elektrobit

EB Assist solutions

EBHSCR

Documentation

Version 0.30

www.elektrobit.com

Elektrobit Automotive GmbH
Am Wolfsmantel 46
91058 Erlangen, Germany
Phone: +49 9131 7701 0
Fax: +49 9131 7701 6333
Email: info.automotive@elektrobit.com

Technical support

https://www.elektrobit.com/support

Legal disclaimer

Table of Contents

1. Typographic and style conventions

The signal word *WARNING* indicates information that is vital for the success of the configuration.

WARNING

⚠

Source and kind of the problem

What can happen to the product?

What are the consequences of the problem?

How does the user avoid the problem?

WARNING

⚠

ESD (Electrostatically Sensitive Device) warning

This warning informs of damages that may occur by electrostatic discharge.

WARNING

⚠

EMC (Electromagnetic Compatibility) warning

This warning informs about how to ensure electromagnetic compatibility with other devices.

The signal word *NOTE* indicates important information on a subject.

NOTE

ⓘ

Important information

Gives important information on a subject

The signal word *TIP* provides helpful hints, tips and shortcuts.

TIP

Helpful hints

Gives helpful hints

Throughout the documentation you find words and phrases that are displayed in **bold**, *italic*, or `mono-spaced` font.

To find out what these conventions mean, see the following table.

All default text is written in Arial Regular font.

Font	Description	Example
Arial italics	Emphasizes new or important terms	The *basic building blocks* of a configuration are module configurations.
Arial boldface	GUI elements and keyboard keys	1. In the **Project** drop-down list box, select Project_A.

1. Typographic and style conventions

Font	Description	Example
		2. Press the **Enter** key.
Monospaced font (Courier)	User input, code, and file directories	The module calls the `BswM_Dcm_RequestSessionMode()` function. For the project name, enter `Project_Test`.
Square brackets [ ]	Denotes optional parameters; for command syntax with optional parameters	`insertBefore [<opt>]`
Curly brackets {}	Denotes mandatory parameters; for command syntax with mandatory parameters	`insertBefore {<file>}`
Ellipsis ...	Indicates further parameters; for command syntax with multiple parameters	`insertBefore [<opt>…]`
A vertical bar |	Indicates all available parameters; for command syntax in which you select one of the available parameters	`allowinvalidmarkup {on|off}`

2. Documentation version information

EBHSCR documentation updates

EBHSCR version	Date	Changes
0.01	19.12.2018	Initial Version
0.02	16.01.2019	Removed status bit 8 in Ethernet EBHSCR packet header
0.03	07.02.2019	Added value information for Link Up/Down and Master/Slave in Ethernet EBHSCR packet header
0.04	13.02.2019	Added NMEA messages
0.05	03.04.2019	Added False Carrier status bit in Ethernet EBHSCR packet header; Added additional info to status section in Ethernet EBHSCR packet header
0.06	07.05.2019	Added TimeOffset messages
0.07	25.07.2019	Removed TimeOffset messages; Added TimeState messages
0.08	25.07.2019	Added description for using the channel field as a minor number
0.09	30.07.2019	Formatting changes TimeState messages
0.10	02.08.2019	Added prototype for CAN FD message; Payload based on socketcan format
0.11	13.09.2019	Added byte and bit information for Ethernet Major number specific header
0.12	13.09.2019	Video MIPI EBHSCR packet draft
0.13	01.10.2019	Released CAN FD message format; Minor changes in CAN FD
0.14	04.10.2019	Added new features to Ethernet packet format: Possibility to capture sent frames to the Ethernet packet description (Tx Error field); Support for capture hardware which does not support capture of FCS; Support for capture hardware which does not support taking a stop timestamp
0.15	21.02.2020	Changed max packet size (fragmentation limit) to 7 MB
0.16	04.05.2020	Added CAN Timesync source to EB TimeState packet
0.17	04.05.2020	Update timesource values so that they correspond to the FPGA values
0.18	04.09.2020	Added Can Error Logging Counter (CEL) to CAN / -FD protocol status content
0.19	16.09.2020	Added information about the start timestamp interpretation in the replay case for Ethernet
0.20	02.02.2021	Added Digital IO messages
0.21	08.11.2021	Added FlexRay and LIN messages

2. Documentation version information

EBHSCR version	Date	Changes
0.22	15.06.2024	Added MIPI_CSI2 messages; reorganize chapters
0.23	15.07.2024	Updated Digital IO messages
0.24	25.07.2024	Added EBX3 information
0.25	09.08.2024	Added CAN and FlexRay FIFO overflow flags
0.26	30.10.2024	Renamed product line to "EB Assist solutions"
0.27	28.11.2024	Updated major number specific header tables; Consistent format across different chapters.
0.28	29.11.2024	Unified header description; Fixed spelling and formatting mistakes.
0.29	05.03.2025	Update DSI3 channel description
0.30	25.03.2025	Update version information history

3. EBHSCR header description

The Elektrobit High Speed Capture and Replay (EBHSCR) protocol is produced by Elektrobit hardware for interfacing high speed automotive interfaces.

Figure 3.1. EBHSCR general major header description

```
 Protocol-Layout:
 0               1               2               3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Major number |Sl |  Channel  |SlG|Ver|         Status        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Off |                   Payload Length                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                       Start timestamp                         +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                         Stop timestamp                        +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                  Major number specific header                 +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- All multi-byte fields in the frame header are big endian.
- After the header comes payload of Payload Length bytes. Its content depends on the *Major number*.
- Every field marked as reserved in the EBHSCR protocol shall be handled in the following way: Set to `0` when generating, ignore when processing. This handling enables adding future extensions to the protocol.

Table 3.1. EBHSCR major header description

Header field	Description
Major number	Type of the frame
Slot (Sl)	Slot 0-3 Not used for the EBX3
Channel / Minor number	Each slot has channels: 0-63. The Channel number shall be zero based and dense. The structure and semantics of data received / transmitted may be different for

Header field	Description
	each channel. Therefore it is also possible to use this field as a Minor number, especially for protocols which don't support dedicated channels. Depending on the *Major number*, the slot field may have its original semantics when using the channel as a minor number, or it may be used to extend the range of the minor number by two bits.
Slot group (SlG)	Slot groups (0-3) may be used to further sub-group slots. Not used for the EBX3
Version (Ver)	Version of header format • `0`: The packet format defined in this table is used. Currently only protocol Version 0 is defined. • `<>0`: Everything except these four bits and the length in bytes may change with a new format. This ensures that unknown packets can be skipped if the new version is not yet supported.
Status	Flags used for status information and error reporting, specific for each *Major number*. • Unused bits are reserved for future use (set to `0` when generating, ignore when processing). • Bit numbering: Bit `0` is least significant Bit in 4th byte of EBHSCR header, Bit `11` is most significant bit of Status field in 3rd byte of EBHSCR header.
Payload Offset (Off)	Defines an offset in bytes when the payload begins. If different from zero payload offset bytes padding of undefined content are prepended before the payload field. Needed in order to keep the required alignment.
Payload Length	Number of payload bytes received until the end of the frame or until an error occurred. Not including header length. Please note that the maximum supported payload length an application should handle is 8 MB. See also Padding.
Start timestamp	• Containing the 64-bit timestamp value when received the start of the payload frame. • The timestamp is a nanoseconds counter. • The exact timestamping point is defined depending on the *Major number*.
Stop timestamp	• Containing the 64-bit timestamp value when received the end of the payload frame. • The timestamp is a nanoseconds counter.

3. EBHSCR header description

Header field	Description
	• The exact timestamping point is defined depending on the *Major number*.
Major number specific header	• Header specific for each *Major number* • Fixed-length of 8 bytes

The following sections describe the supported frame types and its usage of the EBHSCR protocol fields.

Table 3.2. Defined EBHSCR packet types

Major number	**Link to packet definition**
0x43 .. 0x4f	Custom EBHSCR packet types
0x43/0x44	???
0x45	???
0x46	???
0x50	Ethernet
0x51	NMEA
0x52	TimeState
0x53	CAN
0x54	Management packets
0x55	LIN
0x56	Digital IO
0x57	FlexRay
0x58	PDU
0x59	MIPI_CSI2
0x5A	I2C
0x5B	SimpleImageFrame
0x5C	DSI3
0x5D	SPI
0x5E	CSI2_FRAME

4. Padding

- Between EBHSCR packets padding bytes are added to optimize the access to the next packet header.
- Padding bytes have undefined content.
- The padding bytes are not included in the Payload Length header field.
- The number of padding bytes added depend on the implementation architecture of the FPGA design to reach the following alignment:
  - EB 7200 “MI5 base Architecture”: 8 byte alignment
  - EB 7200 “LVDS3G Architecture”: 4 byte alignment
  - EB 8200: “LVDS3G Architecture”: 4 byte alignment (in UDP frames)
  - EBX3 “EBReCap Api Reference”: 32 byte alignment
- Because the number of padding bytes depends on the implementation and the padding bytes have undefined content the user application shall not rely on it. E.g. it shall not use the padding bytes for application specific code.

5. Ethernet

Ethernet EBHSCR packets either provide logged data (Capture of Ethernet traffic) or are used to define data to be transmitted over Ethernet (Replay). Depending on these use cases the meaning of header fields and individual bits can differ between these use cases. Therefore the description of each header field contains two subsections (Capture and Replay) providing the use case specific definition of the header fields. Special care has to be taken, when "captured" EBHSCR packets containing logged data shall be used as base for Replay as some fields or bits might be used differently in both use cases. This is especially true for e.g. status bit indicating errors during reception as replay does not support fault injection and cannot reproduce such errors.

5.1. Header fields

5.1.1. Major number

0x50

5.1.2. Channel

Capture

- Bits 2 to 0: Indication of the physical Ethernet interface of the module on which the Ethernet frame has been received.
- Bit 3: Indication that the frame was not received from an external partner through the physical Ethernet interface indicated in the bits 2 to 0, but the transmission of the device itself over the physical Ethernet interface was logged ("self logging").
- Bit 4: Reserved
- Bit 5: Reserved

Replay

- Bits 2 to 0: Indication of the physical Ethernet interface of the module through which the Ethernet frame shall be sent.
- Bit 3: Reserved
- Bit 4: Reserved
- Bit 5: Reserved

5.1.3. Status

Capture

The **Status** field is used for Rx errors or events. If a bit is set (has the value 1) it indicates the following error/event:

- Bit 0: Ethernet CRC error
- Bit 1: Media-independent interface (GMII/RMII/MII etc.) FIFO overflow error
- Bit 2: Payload FIFO overflow error
- Bit 3: Header FIFO overflow error
- Bit 4: Receiver decoder error
  The receiver decoder error covers following cases:
  - Preamble and Start Frame Delimiter (SFD) are not received as stated in IEEE standard for Ethernet on specific Media Independent Interface.
  - If Data Reception Error occurs (RX_DV = 1 and RX_ER = 1)
- Bit 5: Symbol error
- Bit 6: Jabber event
- Bit 7: Polarity change event
- Bit 8: False carrier
- Bit 9: Truncation: The received frame has been truncated because it was too big for reception.
- Bit 10: Transmission Discarded Error: An Ethernet scheduled for transmission could not be sent within a configured time window. If this bit is set, then the EBHSCR packet contains no payload. This bit can only be set in conjunction with bit 3 of the Channel field. Otherwise: Reserved.
- Bit 11: Reserved

Replay

- Bits 9 to 0: Reserved
- Bit 10: START_FRAME_SEPARATION_BIT
- Bit 11: WAIT_FRAME_SEPARATION_BIT

5.1.4. Start timestamp

Capture

The timestamp is taken after the end of the SFD (Start of Frame Delimiter) detection, as specified in the IEEE 1588 Specification Sep-2004 (IEC 61588 First Edition). The timestamp is taken on the Media-Independent Interface (MII) between DATA LINK and PHY Layer, inside of the FPGA. The timestamp definition is valid for all capture packets, including the logged transmission packets ("self logging"). Detailed information: The MII is a parallel/non-serial MII (e.g. GMII/RMII/MII) and the data on the interface is not encoded (e.g. 8b10b, 64b66b).

If bit 10 (Transmission Discarded Error) in the Status field of the EBHSCR header is set, the start timestamp contains the time when the Ethernet frame was discarded.

Replay

The start timestamp describes the time when the transmission of first bit of the Destination MAC is on the Media- Independent Interface (MII) between DATA LINK and PHY Layer, inside of the FPGA. Detailed information: The MII is a parallel/non-serial MII (e.g. GMII/RMII/MII) and the data on the interface is not encoded (e.g. 8b10b, 64b66b).

The delay of the PHY Layer (e.g. PCS/PMA, Phy Device etc.) is not considered.

5.1.5. Stop timestamp

Capture

The stop timestamp is taken after the FCS of an Ethernet frame has been received.

If an implementation does not support taking a stop timestamp, then the stop timestamp will be set to the same value as the start timestamp.

If bit 10 (Transmission Discarded Error) in the Status field of the EBHSCR header is set, the stop timestamp contains the time when the Ethernet frame was discarded.

Replay

The stop timestamp is not considered during the replay of packets. The value written inside of the stop timestamp field can be ignored.

5.1.6. Major number specific header

The structure and content of the Major number specific header for Ethernet EBHSCR packets is different when packets have been captured or are used for replaying Ethernet data using EB hardware.

Capture

Figure 5.1. Major number specific header overview for Ethernet - Capture

```
Capture:
 0               1               2               3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Reserved   |  TXErrors     |    Reserved   | ChannelStatus |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Reserved                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **Byte 0:** Reserved
- **Byte 1:** TXErrors
  In case of capturing of frames transmitted by the device itself (“self logging”), each transmitted frame payload is returned together with the time of actual transmission and the **TXError** field is set to:
  - Bit 0 (LSB): If the value is `1`, a *Truncation* occurred. The frame is sent truncated.

  - Bit 1: If the value is `1`, a *Transmitter underrun* occurred. The Ethernet controller encountered a transmitter underrun condition while sending the associated Ethernet frame (e.g. caused by internal memory bus errors). This may cause a CRC error on the receiving node.
  - Bit 2: If the value is `1`, the *Retransmission Limit* was reached. The Ethernet controller failed to send a message successfully due to repeated collisions. This error terminates the transmission of the associated Ethernet frame.
  - Bit 3: If the value is `1`, a *Late collision* was detected. A collision occurred after 64 bytes are sent. This error terminates the transmission of the associated Ethernet frame.
- **Byte 2:** Reserved
- **Byte 3:** ChannelStatus - provides status information of the Ethernet interface
  - Bit 0 (LSB): Link Up/Down (Link Up = `1`, Link Down = `0`). Note that the link state changes may be sent without payload.
  - Bit 1: Master/Slave (if supported) (Master = `1`, Slave = `0`)
  - Bit 2: *FCS unavailable*
    - `0`: FCS appended to payload
    - `1`: FCS not appended to payload. This bit is set by the capture hardware which is not able to capture FCS.
  - Bit 3: Reserved
  - Bits 4..7: Speed
    - `0000`: Speed 10 M
    - `0001`: Speed 100 M
    - `0010`: Speed 1000 M
    - `0011`: Speed 2.5 G
    - `0100`: Speed 5 G
    - `0101`: Speed 10 G
    - `0110`: Speed 25 G
    - `0111`: Speed 40 G
    - `1000`: Speed 100 G
    - `1001` - `1110`: Reserved
    - `1111`: Speed unknown. This value can be used when the speed could not be detected, e.g. because of Ethernet transceiver access errors.
- **Byte 4..7:** Reserved

All reserved bytes and bits are set to `0`.

Replay

Figure 5.2. Major number specific header overview for Ethernet - Replay

```
Replay
 0               1               2               3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Reserved            |    Reserved   | ChannelStatus |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Reserved            |         SeparationTime        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **Byte 0..2:** Reserved
- **Byte 3:** ChannelStatus - provides status information on the Ethernet interface
  - Bits 0..1: Reserved
  - Bit 2: FCS unavailable
    - `0`: The payload contains the FCS.
    - `1`: The payload does NOT contain the FCS.

    Even if the payload contains an FCS, by default it will not be transmitted. The transmitting device will calculate and transmit an FCS on its own.
  - Bits 3 to 7: Reserved
- **Byte 4..5:** Reserved
- **Byte 6..7:** SeparationTime
  - These bytes represent a 16 bit unsigned integer. The most significant byte is the 7th byte by which the time for the frame separation gap can be defined.
  - The actual separation time is the result of multiplying the value of the SeparationTime field with 64 ns. Thus, it is possible to configure the separation time from 0 to 4194240 ns (4194 µs).
  - If `START_FRAME_SEPARATION_BIT` is set, the next best effort frame to be sent having the `WAIT_FRAME_SEPARATION_BIT` set, will not be transmitted before this time has elapsed after the transmission of the frame where `START_FRAME_SEPARATION_BIT` was set.
  - The minimum IFG defined in the Ethernet specification will always be considered and be added automatically to the frame separation time defined by the SeparationTime field.

Reserved bytes/bits should be set to `0`. Anyway, these will be ignored for transmission.

5.2. Payload

5.2.1. Frame Discarded

If the **Transmission Discarded Error** bit is set in the status field of the header, the EBHSCR packet does contain a payload of 8 bytes, only. The payload contains the original start time stamp (big endian - as in the EBHSCR header fields) of the packet which contained the frame which was discarded, i.e. the point in time, when the frame should have been sent.

5.2.2. Capture and Replay

802.3 Ethernet frame:

- The payload begins with the destination MAC and ends with the FCS of an Ethernet frame.
- The payload contains an FCS if the **FCS unavailable** bit is cleared.
- The payload does not contain an FCS if **FCS unavailable** bit is set.
- The payload may be handed over to Wireshark or Tcpdump Ethernet dissectors unmodified.

6. NMEA

6.1. Header fields

6.1.1. Major number

0x51

6.1.2. Channel

Controller ID (zero based)

6.1.3. Status

Reserved. Bits are set to `0` when generating and ignored when processing.

6.1.4. Start timestamp

- ZDA: Timestamp of 1PPS pulse
- All other: Undefined

6.1.5. Stop timestamp

Timestamp taken after the NMEA message was received from UART.

6.1.6. Major number specific header

Reserved. Bits are set to `0` when generating and ignored when processing.

6.2. Payload

ASCII NMEA data

7. TimeState

7.1. Header fields

7.1.1. Major number

0x52

7.1.2. Slot

0

7.1.3. Channel

0

7.1.4. Status

Validity bit mask: A bit mask that indicates which payload fields contain valid data.

- Bit 0: Time offset valid (Payload Byte `0..7`). This can be used for replay to set the time offset manually.
- Bit 1: Last offset change valid (Payload Byte `8..15`). Ignored for replay.
- Bit 2: Nanoseconds last jump valid (Payload Byte `16..23`). Ignored for replay.
- Bit 3: UTC leap seconds valid (Payload Byte `24..25`). This can be used for replay to set the UTC leap seconds manually.
- Bit 4: Sync state valid (Payload Byte `26..27`). Ignored for replay.

7.1.5. Start timestamp

Defines the point in time when the packet was created. Only valid for capture, ignored for replay.

7.1.6. Stop timestamp

Set to same value as start timestamp.

7.1.7. Major number specific header

Figure 7.1. Major number specific header overview for TimeState

```
 0               1               2               3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Reserved   |    Reserved   |    Reserved   |    Reserved   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Reserved              |  Time source  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **Byte 0..6:** Reserved
- **Byte 7:** Time source. One of the following values:
  - `0x00`: TimeSourceNone
  - `0x01`: TimeSourceEBTimesyncHard
  - `0x02`: TimeSourceXTSS
  - `0x03`: TimeSourcePTPHW
  - `0x10`: TimeSourcePTPSW
  - `0x20`: TimeSourceGPS
  - `0x30`: TimeSourceEBTimesyncSoft
  - `0x40`: TimeSourceCAN
  - `0x50`: TimeSourceEBVirt

Implementation note: The least significant four bits are time sources handled by FPGA. The most significant bits are time source handled by CPU.

7.2. Payload

All multibyte payload fields are unsigned and big endian:

Byte 0..7: EB time offset in nanoseconds.

- The offset is the difference of the zero-based start and stop timestamp in the EBHSCR header to TAI (1.1.1970 00:00:00, without leap seconds).

Byte 8..15: Last offset change in nanoseconds.

- Point in time of the last change of time offset. Can be used to track changes in time offset.

Byte 16..23: Nanoseconds last jump.

- Point in time of the last hard change/jump of time count after the jump. Hard changes in time count indicate an error. This only happens if a smooth change is not possible.

Byte 24..25: UTC leap-seconds

Byte 26..27: Sync state

- `0x0000`: Free running
- `0x0001`: Locked to master

8. CAN

8.1. Header fields

8.1.1. Major number

0x53

8.1.2. Channel

Controller ID (zero based)

8.1.3. Status

- Bit 0: CAN FD flag.
  - `0`: Classical CAN
  - `1`: CAN FD data frame
- Bit 1: protocol status flag (only relevant for capture, ignored for replay)
  - `0`: Protocol status not available
  - `1`: Protocol status available
- Bit 2: Overflow flag.
  - `0`: No FIFO overflow detected
  - `1`: FIFO overflow detected, data were lost

8.1.4. Start timestamp

Defines the point in time about 25 ns after the Start-of-frame (SOF) of the CAN message.

8.1.5. Stop timestamp

Defines the point in time about 25 ns after the End-of-frame (EOF) of the CAN message. Only relevant for capture, ignored for replay.

8.1.6. Major number specific header

Contains protocol status information and is only available if the protocol status flag in the status field is set. The protocol status may be sent without payload (Payload Length set to `0`). Only relevant for capture, ignored for replay.

Reserved bits are set to `0` when generating and ignored when processing. Bit 0 denotes the LSB in an octet. Byte 0 is the first octet.

Figure 8.1. Major number specific header overview for CAN

```
 0               1               2               3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      REC      |      TEC      |    res.  |DLEC |O|W|P|res| LEC |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   reserved                    |      CEL      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **Byte 0:** Receive Error Counter (REC)
  - Current value of the receive error counter `0..127`.
- **Byte 1:** Transmit Error Counter (TEC)
  - Current value of the transmit error counter `0..255`.
- **Byte 2:**
  - Bit 0..2: Data phase last error code (DLEC)
    - Error code during data phase of CAN FD frame with BRS flag set.
    - Coding is the same as LEC.
  - Bit 3..7: res. = reserved
- **Byte 3:**
  - Bit 0..2: last error code (LEC)
    - `0`: No Error. The last CAN message transfer was OK.
    - `1`: Stuff Error. More than five equal bits in a sequence have occurred in a part of a received message where this is not allowed.
    - `2`: Form Error. The fixed-format part of a frame has the wrong format.
    - `3`: Ack Error. The message transmitted was not acknowledged by another node.
    - `4`: Bit1 Error. During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value `1`), but the monitored bus value was dominant.
    - `5`: Bit0 Error. During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a dominant level (data or identifier bit logical value `0`), but the monitored bus value was recessive. During Bus_Off recovery this status is set each time a sequence of 11 recessive bits has been monitored. This enables the CPU to monitor the proceeding of the Bus_Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed).

        - `6`: CRC Error. The CRC checksum received for an incoming message does not match the CRC value that the controller calculated for the received data.
        - `7`: NoChange. No CAN bus event was detected since the last read. This event is currently unused.
    - Bit 3..4: res = reserved
    - Bit 5: ERRP (P)
        - `0`: Error counters are below the error passive limit (128).
        - `1`: One of the error counters has reached the error passive limit (128).
    - Bit 6: ERRW (W)
        - `0`: Error counters are below the error warning limit (96).
        - `1`: One of the error counters has reached the error warning limit (96).
    - Bit 7: BOFF (O)
        - `0`: Not in Bus_Off state
        - `1`: In Bus_Off state
- **Byte 4..6:** Reserved
- **Byte 7:** Can Error Logging Counter (CEL)
    - Current value of the CAN error logging counter 0..255

8.2. Payload

Reserved bits are set to 0 when generating and ignored when processing. Bit 0 denotes the LSB in an octet. Byte 0 is the first octet.

Figure 8.2. Overview of CAN payload structure

```
0               1               2               3
0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               id                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      len      |     flags     |            reserved           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                       data 0 .. 64 bytes                      +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Byte 0..3: id in little endian format.

- Bit 31: Extended frame format (EFF) flag

  - `0`: Standard 11 bit can identifier
  - `1`: Extended 29 bit can identifier
- Bit 30: Remote transmission request (RTR) flag
  - RTR frames are not supported by CAN FD.
  - RTR frames have a length (len field) set, but do not carry data.
- Bit 29: Reserved
- Bit 0..28: CAN identifier:
  - Standard: 11 bit (EFF flag 0)
  - Extended: 29 bit (EFF flag 1)

Byte 4: len - Frame data length in byte

- 0..8 for classic CAN (including RTR frames)
  RTR frames have a length set, but they do not carry payload!
- 0..8, 12, 16, 20, 24, 32, 48, 64 for CAN FD
- Bit 7 (most significant bit): set to 0 when generating, ignore when processing

Byte 5: flags - Used for CAN FD specific flags:

- Bit 0: Bit rate switch (BRS). If set, the second bit rate for data is used in CAN FD frame.
- Bit 1: Error state indicator (ESI). Set by the transmitting CAN FD controller to indicate errors.
- Bit 2..7: Set to `0` when generating, ignore when processing.

Byte 8..: data - CAN frame payload:

- `0..8` for classic CAN data frames
- `0` for classic CAN RTR frames
- `0..8`, `12`, `16`, `20`, `24`, `32`, `48`, `64` for CAN FD

WARNING ⚠

In case of RTR frames, or in case of errors on the CAN bus there may be less data bytes available than specified in the len field. Ensure to never read out more data bytes than available overall (as specified by the overall EBHSCR Header Payload Length field)!

9. LIN

9.1. Header fields

9.1.1. Major number

0x55

9.1.2. Channel

- 0-3 (Controller ID)

9.1.3. Status

- Bit 0: Set to `1` if a LIN 1.3 Classic Checksum received. During reception the checksum is validated to determine this bit. If the received checksum is invalid this bit can not be evaluated. Version 1.3 checksum is calculated over data bytes.
- Bit 1: Set to `1` if a LIN 2.0 Enhanced Checksum received. During reception the checksum is validated to determine this bit. If the received checksum is invalid this bit can not be evaluated. Version 2.0 checksum is calculated over ID and data byes.
- Bit 4: Set to `1` if a LIN Wake-Up Packet was received. A wakeup packet contains no payload (Payload length field is set to `0`). The wakeup length field in the major number specific header is set.
- Bit 10: Set to `1` if a time jump occurred near the edge and thus the timestamp was estimated. Only relevant for capture, ignored for replay.

9.1.4. Start timestamp

Taken at rising edge of Break field in the LIN message header.

9.1.5. Stop timestamp

Taken 30 bits after last rising edge of the LIN message. The LIN frame does not contain a length field, thus the end of the frame is detected using a timeout mechanism.

9.1.6. Major number specific header

Figure 9.1. Major number specific header overview for LIN

```
 0               1               2               3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        wakeup length          |       protocol status         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           reserved                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

All multibyte fields are unsigned and big endian.

- **Byte 0..1:** Wakeup length: Wakeup signal low phase length in us. Only valid if the wakeup bit in status header is set. Set to `0` otherwise.
- **Byte 2..3:** Protocol status: Bit 0 is LSB, bit 15 is MSB
  - Bit 0: Reserved (this seams to be superfluous - because error bits are set in case of error - VALID bit in current implementation)
  - Bit 1 '1': SYN - Received synchronization field is not 0x55
  - Bit 2 '1': PAR - Received parity does not match calculated parity
  - Bit 3 '1': RES - No response detected after LIN header (payload length field is 1 - i.e. only an identifier but no data was received)
  - Bit 4 '1': DAT - Too many data bytes received (more than 10, 8 data bytes + ID and checksum)
  - Bit 5 '1': CHK - Checksum is invalid: Received checksum does not match calculated checksum
  - Bit 6 '1': STA - Expected start bit, but detected recessive bus level
  - Bit 7 '1': STO - Expected stop bit, but detected recessive bus level
  - Bit 8 '1': EMP - Break and Sync received, but no further data
- **Byte 4..7:** Reserved

9.2. Payload

The layout matches the LIN frame as received on the bus, according to LIN Specification 2.2:

Byte 0: Protected identifier field

- Bit 0..5: LIN frame identifier
- Bit 6: Parity bit 0
- Bit 7: Parity bit 1

Byte 1..8: LIN data bytes

Byte [data(1-8) + 1]: After the last data byte comes one byte checksum.

10. Digital IO

10.1. Header fields

10.1.1. Major number

0x56

10.1.2. Channel

The interpretation of the channel field depends on the value of the "Value type" field of the major number specific header of this protocol. Please check the section below for more information.

10.1.3. Status

- Bit 0: Set to `1` in case of an overflow in the monitoring unit. In this case all remaining fields are invalid. Only relevant for capture, ignored for replay.
- Bit 10: Set to `1` if a time jump occurred near the edge and thus the timestamp was estimated. Only relevant for capture, ignored for replay.

10.1.4. Start timestamp

Defines the point in time about 25 ns after the edge specified in the Value type field.

10.1.5. Stop timestamp

Same value as start timestamp. Only relevant for capture, ignored for replay.

10.1.6. Major number specific header

Figure 10.1. Major number specific header overview for Digital IO

```
  0               1               2               3
  0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |  Value type   |   GPIO-ID     |          reserved             |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                            reserved                           |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **Byte 0:** Value type: Specifies the value and type of the Digital IO event.
  - Bit 0..1:
    - `0` Event triggered falling edge
    - `1` Event triggered rising edge
    - `2` GPIO is in low state (No state change)
    - `3` GPIO is in high state (No state change)
  - Bit 2:
    - `0` The EBHSCR header 'Channel' field refers to the Digital IO line number (zero based). EB 2200 / EB 5200 provides four input lines for DETI with the values 0-3.
    - `1` The EBHSCR header 'Channel' field refers to a accumulated GPIO data channel and the GPIO-ID field is used for further identification of the GPIO.
  - Bit 3..7: reserved
- **Byte 1:** GPIO ID: Specifies the ID of a GPIO. Based on the Physical Module this may refer to a physical GPIO-PIN on the interface module or a virtual GPIO-ID
- **Byte 2..7:** Reserved

11. FlexRay

11.1. Header fields

11.1.1. Major number

0x57

11.1.2. Channel

- Bit 0: Channel A
- Bit 1: Channel B
- Bit 2..4: Controller ID (zero based) big endian

11.1.3. Status

FlexRay EBHSCR packets come in different packet types:

- FlexRay Frames contain captured FlexRay data frames with their payload.
- FlexRay symbols are generated when a FlexRay symbol is captured.
- FlexRay slot status packets are generated when:
  - during FlexRay synchronous monitoring when an error occurs.
  - during FlexRay asynchronous and synchronous monitoring when a receive FIFO overflow is detected and data are lost.
- FlexRay start-of-cycle packets are only generated during FlexRay synchronous whenever the FlexRay cycle counter increments.

These four EBHSCR packet types are determined by the *Packet type* bits (status bits 3..2) , see section 11.1.3.1, General.

11.1.3.1. General

Bits 0..4 and Bit 11 are are identical for FlexRay frame, FlexRay symbol, slot status and start-of-cycle EBHSCR packets:

- Bit 0: Synchronous monitoring packet
  - `0`: Packet was generated by asynchronous monitoring
  - `1`: Packet was generated by FlexRay synchronous monitoring
- Bit 1: FlexRay cluster is in sync (only valid if bit 0 = `1`)
- Bit 3..2: *Packet type* (see section 11.1.6, Major number specific header for details)

  - `00`: Packet contains a FlexRay frame (see section 11.1.3.2, FlexRay frame)
  - `01`: Packet contains a symbol (see section 11.1.3.3, FlexRay symbol frame)
  - `10`: Packet contains a a slot status (see section 11.1.3.4, FlexRay slot status frame)
  - `11`: Packet contains a start-of-cycle packet (only valid if bit 1 = `1`) (see section 11.1.3.5, FlexRay start-of-cycle start)
- Bit 4..10: Depended on *Packet type*, see status bits 3..2
- Bit 11: Set to `1` if a time jump occurred near the edge and thus the timestamp was estimated. Only relevant for capture, ignored for replay.

11.1.3.2. FlexRay frame

- Bit 4: CODERR: Coding error. Indicates if a Frame Start Sequence Error (FSSERR) or a Byte Start Sequence error (BSSERR) occurred.
- Bit 5: TSSVIOL: TSS violation
- Bit 6: HCRCERR: Header CRC error
- Bit 7: FCRCERR: Frame CRC error
- Bit 8: FESERR: Frame end sequence error
- Bit 9: FSSERR: Frame start sequence error
- Bit 10: BSSERR: Byte start sequence error

11.1.3.3. FlexRay symbol frame

- Bit 4..10: reserved.

11.1.3.4. FlexRay slot status frame

- Bit 4: OVERFLOWERR: FIFO overflow error. Captured FlexRay data were lost.
- Bit 5..10: reserved.

11.1.3.5. FlexRay start-of-cycle start

- Bit 4..10: reserved.

11.1.4. Start timestamp

Defines the point in time about 80 ns after the falling edge of the TSS of the FlexRay frame.

Replay

- `0`: Do synchronous replay (use slot/cycle/supercycle information)

- Otherwise: Do asynchronous replay (not supported yet)

11.1.5. Stop timestamp

Defines the point in time about 25 ns after the rising edge in between the frame end sequence bits.

11.1.6. Major number specific header

Capture

Figure 11.1. Capture

```
Capture:
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Slot information       |         Frame status          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Symbol length |                 Reserved                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

In brackets you can find the value of status bits 3..0 for which the specified fields are valid.

- **Byte 0..1:** Slot information (0011, 0111, 1011)

15	14	13	12	11	10	9	8
SBV	ACI	CED	SED	VFR	SID10	SID9	SID8
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SID7	SID6	SID5	SID4	SID3	SID2	SID1	SID0

  SID [0..10]: Slot ID
  VFR: Valid Frame Received
  SED: Syntax Error Detected
  CED: Content Error Detected
  ACI: Additional Communication Indicator
  SBV: Slot Boundary Violation

- **Byte 2..3:** Frame status (0011, 0111, 1011)
  For status 0111 only CP1 and CP0 are valid

15	14	13	12	11	10	9	8
SPLERR	CCERR	FIDERR	SSERR	NERR	SOVERR	SWVIOL	NITVIOL

15	14	13	12	11	10	9	8
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BVIOL	PCD	SYNCERR	CP1	CP0	BRC2	BRC1	BRC0

BRC [0..2]: Byte Received Counter

- Number of bytes received by the decoder without coding error
- When more than 7 bytes are received, the counter is set to 7

CP [0..1] - Communication cycle part

- 0x00: Static part
- 0x01: Dynamic part
- 0x02: Symbol window
- 0x03: NIT

SYNCERR: Sync and/or startup bit wrongly set

PCD: Prolonged channel idle detection

- FES to CHIRP took longer than 11 bit times
- This is always true for dynamic frames because of the DTS.

BVIOL: Boundary violation

NITVIOL: NIT violation

SWVIOL: Symbol Window violation

SOVERR: Slot overbooked error

NERR: Null frame error

SSERR: Sync or startup error

FIDERR: Frame ID error

CCERR: Cycle counter error

SPLERR: Static Payload length error

- **Byte 4:** Symbol length and status (01xx)

7	6	5	4	3	2	1	0
SYERR	SL6	SL5	SL4	SL3	SL2	SL1	SL0

SYERR: The low phase was too long

SL [0..6]: Symbol length in units of bit cells

- Byte 0: POC-State (1111)
- Byte 1: Cycle counter of following cycle (1111)
- Byte 4..7: Supercycle counter (00xx, 10xx, 11xx)

Replay

Currently only replay of frames in synchronous mode is supported.

Byte 4..7: Supercycle counter (00xx, 10xx, 11xx) (big endian); if `0`, use next available supercycle.

11.2. Payload

Only packets of type *FlexRay frame* contain payload. The payload consists of a 5-byte header followed by up to 254 data bytes. The number of bytes may be calculated by multiplying the payload length information (PL) with 2. However, the actual number of bytes in data may be smaller than the calculated value because of truncation, e.g. because of an error on the bus. No padding is added after the data bytes.

The Frame CRC is not appended. If the Frame CRC is wrong the error bit FCRCERR in Error Flags shall be set.

Bits 39 through 32 appear in the first octet, Bits 31 through 24 in the second octet, Bits 23 through 16 in the third octet, and so on. In the octet bits with bits 39-32 bit 39 is the high order bit and bit 32 is the low order bit, all further bits use the same bit counting policy.

11.2.1. Frame header (5 bytes)

39	38	37	36	35	34	33	32
-	PPI	NFI	SFI	STFI	FID10	FID9	FID8
31	**30**	**29**	**28**	**27**	**26**	**25**	**24**
FID7	FID6	FID5	FID4	FID3	FID2	FID1	FID0
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PL6	PL5	PL4	PL3	PL2	PL1	PL0	HCRC10
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
HCRC9	HCRC8	HCRC7	HCRC6	HCRC5	HCRC4	HCRC3	HCRC2
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HCRC1	HCRC0	CC5	CC4	CC3	CC2	CC1	CC0

PPI: Payload preamble indicator

- Static segment: If this bit is set the first 0 to 12 bytes of the payload may optionally be used as a network management vector;
- Dynamic segment: If this bit is set the first two bytes of the payload may optionally be used as a message ID field.
- If the frame is a NULL frame, the PPI is always `0`.

NFI: Null frame indicator

- Indicates whether or not the frame is a NULL frame
- NFI = `0` -> payload segment contains no valid data
- NFI = `1` -> payload segment contains data

- Note: This is in line with the FlexRay specification.

SFI: Sync frame indicator

- Indicates whether or not the frame is a SYNC frame
- SFI = `0` -> no receiving node shall consider the frame for synchronization
- SFI = `1` -> all receiving nodes shall use the frame for synchronization
- Sync frames are only sent in the static segment!
- Each node is only allowed to send one single frame as a sync frame!

STFI: Startup frame indicator

- Indicates whether or not a frame is a STARTUP frame
- STFI = `0` -> frame is not a startup frame
- STFI = `1` -> frame is a startup frame
- A startup frame must always be a sync frame!

FID [0..10]: Frame ID

- Values between 1 and 2047
- FID10 is the most significant bit, FID0 is the least significant bit (big endian).
- Frame ID `0` is an invalid frame ID!

PL[0..6]: Payload length in words (2 bytes): Values between `0` and `127`

HCRC[0..10] - Header CRC: Values between `0` and `2047`. HCRC10 is the most significant bit. HCRC0 is the least significant bit (big endian).

CC[0..5] - Cycle count: Values between `0` and `63`

12. MIPI_CSI2

12.1. Header fields

12.1.1. Major number

0x59

12.1.2. Channel

- Indication of the physical MIPI interface of the module on which the MIPI CSI-2 Low Level Protocol packet has been received/shall be sent.
- The channel is dependent on the EB Logging Interface HW.

12.1.3. Status

The Status field is used for Rx errors or events. If a bit is set (has the value 1) it indicates the following error/event:

- Bit 0: MIPI CSI-2 long packet checksum error
- Bit 1: Reserved
- Bit 2: MIPI CSI-2 packet header contains an uncorrectable ECC error
- Bit 3: Payload FIFO Overflow error
- Bit 4: Header FIFO Overflow error
- Bit 5: Receiver Decoder error
  - Receiver Decoder Error covers following case(s):
    - Start of Transmission (SoT) Error according to MIPI Alliance specification for CSI-2
- Bit 6: Payload Truncated Error
- Bit 10: Transmission Rejected : If during transmission MIPI CSI-2 Low Level Protocol packet was not on-time or within grace period.

12.1.4. Start timestamp

The timestamp describes the point in time immediately after the end of the Start of Transmission sequence (SoT) of the MIPI CSI-2 Low Level Protocol packet at the FPGA MIPI interface.

If bit 10 in the Status field of the EBHSCR header is set, the Start timestamp contains the time when the MIPI CSI-2 Low-Level packet packet was discarded.

12.1.5. Stop timestamp

The timestamp describes the point in time immediately before the beginning of the End of Transmission sequence (EoT) of the MIPI CSI-2 Low-Level packet at the FPGA MIPI interface.

If a multi-lane configuration (e.g. 2-Lane MIPI, 4-lane MIPI etc.) is used, the EoT arriving at the later time shall be used for the timestamp reference.

If bit 10 in the Status field of the EBHSCR header is set, the Stop timestamp contains the time when the MIPI CSI-2 Low-Level packet packet was discarded.

12.1.6. Major number specific header

Figure 12.1. Major number specific header overview for MIPI CSI2

```
 0               1               2               3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Flags             |            Reserved           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          FrameCounter         |          LineCounter          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **Byte 0..1:** Flags
  - Bit 0: MIPI CSI-2 long packet checksum error
  - Bit 1: MIPI CSI-2 packet header contains an correctable ECC error
  - Bit 2: MIPI CSI-2 packet header contains an uncorrectable ECC error
  - Bit 3..13: Reserved
  - Bit 14: MIPI PHY Type
    - `0`: DPHY
    - `1`: CPHY
  - Bit 15: packet contains first line of a frame
- **Byte 2..3:** Reserved
- **Byte 4..5:** FrameCounter
  - Contains the value of the last frame counter received in a CSI-2 V-Sync short packet.
- **Byte 6..7:** LineCounter
  - Contains the value of the last line counter received in a CSI-2 H-Sync short packet.

12.2. Payload

12.2.1. Capture and Replay

CSI-2 Low Level Protocol packet Format:

- The payload begins with an eight byte packet header (PH) followed by an optional payload.
- The payload can consist of only a packet header for CSI-2 short packets.
- The payload can consist of a packet header and payload for CSI-2 long packets.
- The packet header (PH) is defined in the following.
- The packet data format is defined in the MIPI Alliance Specification for CSI-2 in the section "Recommended Memory Storage".

12.2.2. The packet Header (PH)

The packet Header (PH) is composed of five elements: a 6-bit DataType, an 2 bit Virtual Channel, a 16-bit Word Count field and an 8-bit ECC:

Figure 12.2. The Packet Header (PH)

```
  0               1               2               3
  0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |res|      DT    |      res.   |VC |      res.      |      ECC       |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |   CRC  LSB    |   CRC  MSB      | Word Count LSB| Word Count MSB|
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

DT

- MIPI CSI-2 Data Type.

VC

- MIPI CSI-2 Virtual Channel.

CRC LSB

- Least significant byte of the MIPI CSI-2 Checksum. In case of short packets, the value 0x00 00 is used.

CRC MSB

- Most significant byte of the MIPI CSI-2 Checksum. In case of short packets, the value 0x00 00 is used.

Word Count

- MIPI CSI-2 long packet: MIPI CSI-2 Word Count
- MIPI CSI-2 Short packet: MIPI CSI-2 Short packet Data Field

Word ECC

- MIPI CSI-2 packet Header ECC

13. I2C

13.1. Header fields

13.1.1. Major number

0x5A

13.1.2. Channel

- Indication of the I2C interface of the module on which the I2C Packet has been received/shall be sent.
- The channel is dependent on the EB Logging Interface HW.

13.1.3. Status

- Bit 0:FIFO overflow error
  - `0`: No FIFO overflow occurred
  - `1`: FIFO overflow occurred
- Bit 1: Packet Start Condition
  - `0`: Packet started with I2C "Start" condition
  - `1`: Packet started with I2C "Repeated Start" condition
- Bit 3..2: Packet Stop condition
  - `0`: packet ended with I2C "Stop" condition
  - `1`: packet ended with I2C "Repeated Start" condition
  - `2`: packet ended because of an packet oversize. Data might be lost.
  - `3`: packet ended because of an timeout. Data might be lost.
- Bit 4: Data integrity
  - `0`: Payload data is valid.
  - `1`: Payload data or ACK is invalid. I.e.: A packet end condition received outside a 8-bit boundary, a ACK/NACK is received differently than specified in the I2C specification.
- Bit 6..5: reserved, always 0

13.1.4. Start timestamp

Defines the point in time when a I2C Start or Repeated Start condition was detected.

13.1.5. Stop timestamp

Defines the point in time when the packet I2C Stop, Repeated Start Oversize or timeout was detected, leading to the packet end.

13.1.6. Major number specific header

Reserved. Bits are set to `0` when generating and ignored when processing.

13.2. Payload

13.2.1. Capture and Replay

I2C protocol packet format.

Each received byte is represented in a 16-bit word according to the following table.

```
 0               1
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Data      |    Control    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **Byte 0:** Data
- **Byte 1:** Control
  The second byte contains control information associated to the data byte.
  - Bit 0: Acknowledgement
    - `0`: Data byte was acknowledged on the bus (ACK).
    - `1`: Data byte was not acknowledged on the bus(NACK).
  - Bit 1..7: Reserved

14. DSI3

14.1. Header fields

14.1.1. Major number

0x5C

14.1.2. Channel

- 0: DSI-3 Slave channel. This is the channel which is connected to a DSI-3 master and voltage modulated commands are received and logged.
- 1: DSI-3 Master channel. This is the channel which is connected to a DSI-3 slave and current modulated responses are received and logged.
- 2..63: reserved

14.1.3. Status

- Bit 0: CRC error
  - `0`: No CRC error occurred
  - `1`: Indicates that the data packet associated with this header contains a CRC error
- Bit 1: Transition error
  - `0`: No transition error occurred
  - `1`: Indicates that either an unexpected transition on the data lines was detected during packet reception or a transition was not detected when it should have been
- Bit 2: Packet truncated error
  - `0`: Packet was not truncated
  - `1`: Indicates that the associated data packet has been truncated, refer to major specific header for further information on why this happened
- Bit 3: Packet dropped error
  - `0`: Packet has not been dropped
  - `1`: Indicates the associated data packet or an undefined number of packets since the last successful data reception has been dropped completely. Refer to major number specific header for further information on why this has happened
- Bit 11..4: reserved, always 0

14.1.4. Start timestamp

Defines the point in time when a DSI3 command/response packet start condition was detected.

14.1.5. Stop timestamp

Defines the point in time when the DSI3 command/response packet stop condition or error was detected, leading to a packet end.

14.1.6. Major number specific header

When channel is `0` (Slave):

Figure 14.1. Major number specific header overview for DSI3 - Slave

```
  0               1               2               3
  0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |    Status 0   |    Reserved   |    Status 2   |    Status 3   |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                            Reserved                           |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **Byte 0:** Status 0 of major number specific header
  - Bit 0: Type of DSI3 command
    - `0`: Data packet is a "Command and Response Mode" (CRM) command
    - `1`: Data packet is a "Periodic Data Collection Mode" (PDCM) command
  - Bit 6..4: Bit Count
    - Number of bits in the last byte of the data packet
- **Byte 1:** Reserved
- **Byte 2:** Status 2 of major specific header
  - Bit 0: Blanking error
    - Data packet truncated due to blanking signal being asserted during packet reception
  - Bit 1: No Transition
    - Data truncated due to no transition when there should have been a transition
- **Byte 3:** Status 3 of major specific header
  - Bit 0: CRC Error
    - Data packet CRC errors

- Bit 1: Counter Overflow
  - Data truncated due to byte counter overflow
- Bit 2: Disabled Error
  - Data dropped or truncated due to component being disabled during packet transmission
- Bit 3: Truncate Data
  - Back pressure on data AXIS during packet transmission
- Bit 4: Drop Data
  - Back pressure on data AXIS interface at the start of a packet
- **Byte 4..7:** Reserved

When channel is `1` (Master):

Figure 14.2. Major number specific header overview for DSI3 - Master

```
 0               1               2               3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Status 0    |    Reserved   |   Status 2    |    Status 3   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Reserved                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **Byte 0:** Status 0 of major number specific header
  - Bit 0: Type of DSI3 command
    - `0`: Data packet is a response to "Command and Response Mode" (CRM) command
    - `1`: Data packet is a response to "Periodic Data Collection Mode" (PDCM) command
  - Bit 4: Nibble Count
    - Number of nibbles in the last byte of the data packet
- **Byte 1:** Reserved
- **Byte 2:**Status 2 of major specific header
  - Bit 0: Blanking error
    - Data packet truncated due to blanking signal being asserted during packet reception
  - Bit 1: Bad Transition
    - Transition outside edge detection window timing violation
  - Bit 2: Bad Chip
    - Invalid sensor input. I_Sense_2 = 1 when I_Sense_1 = 0

  - Bit 3: No Transition
    - Data truncated or dropped since no valid transition detected for over 4 chip periods
- **Byte 3:** Status 3 of major specific header
  - Bit 0: CRC Error
    - Data packet CRC errors
  - Bit 1: Bad Decode
    - Invalid symbol, data cannot be decoded
  - Bit 2: Counter Overflow
    - Data truncated due to nibble counter overflow
  - Bit 3: Disable Error
    - Data dropped or truncated due to component being disabled during packet transmission
  - Bit 4: Truncate Data
    - Back pressure on data AXIS during packet transmission
  - Bit 5: Drop Data
    - Back pressure on data AXIS interface at the start of a packet
- **Byte 4..7:** Reserved

All other bits are set to zero.

15. SPI

15.1. Header fields

15.1.1. Major number

0x5D

15.1.2. Channel

- Indication of the physical interface of the module on which the image packet has been received/shall be sent.
- The channel is dependent on the EB Logging Interface hardware.

15.1.3. Status

- Bit 0: Overflow error
  - `0`: No Overflow error occurred
  - `1`: Indicates that due to backpressure it was not possible to transmit data, which blocked receiving new data
- Bit 1: Oversize Transmission error
  - `0`: No Oversize Transmission error occurred
  - `1`: Indicates that the amount of received data exceeded the transmission size limit, which causes that the SPI capture logic discards any further received data
- Bit 2: Timeout error
  - `0`: No Timeout error occurred
  - `1`: Indicates that a module internal timeout was triggered due to the internal decoder not forwarding any data nor closing the packet after a capture start was signaled
- Bit 11..4: reserved, always 0

15.1.4. Start timestamp

Defines the point in time when a SPI Start or Chip Selection change condition was detected.

15.1.5. Stop timestamp

Defines the point in time when the SPI clock stopped, Oversize or Timeout was detected, leading to the packet end.

15.1.6. Major number specific header

Figure 15.1. Slave

```
 0               1               2               3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      CS       |    Reserved   |    Reserved   |Last byte valid|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Reserved                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **Byte 0:** CS
  - Bit 0..7: Chip Select (CS) vector: Eight bits to signal which of the CS got asserted.
- **Byte 1:** Reserved
  - Reserved bytes are set to `0` when frame is generated and ignored when processed.
- **Byte 2:** Reserved
  - Reserved bytes are set to `0` when frame is generated and ignored when processed.
- **Byte 3:** Last Byte Valid Bits
  - Bit 0..7: Last Byte Valid Bits: Eight bits to signal which of the bits of the last captured byte are valid.
- **Byte 4..7:** Reserved
  - Reserved bytes are set to `0` when frame is generated and ignored when processed.

16. CSI2_FRAME

16.1. Header fields

16.1.1. Major number

0x5E

16.1.2. Channel

- Indication of the physical MIPI interface of the module on which the MIPI CSI-2 Low Level Protocol packet has been received/shall be sent.
- The channel is dependent on the EB Logging Interface HW.

16.1.3. Status

The Status field is used for Rx errors or events. If a bit is set (has the value 1) it indicates the following error/event:

- Bit 0: MIPI CSI-2 long packet checksum error. Set if at least one CRC error occurred in any line of the image.
- Bit 1: Reserved
- Bit 2: MIPI CSI-2 packet header contains an uncorrectable ECC error. Set if at least one uncorrectable ECC error was detected in any image line.
- Bit 3: Receiver Decoder error. Set if at least one decoder error occurred during the reception of the frame.
  - Receiver Decoder Error covers following case(s):
    - Start of Transmission (SoT) Error according to MIPI Alliance specification for CSI-2
      Termination of the HS state before the complete packet has been transmitted.
- Bit 4: Section Error: More sections than possible entries in the frame header are received.
- Bit 5: FIFO Overflow error
- Bit 6: Payload Truncated Error. Set if either a line exceeds the maximum supported length or the frame exceeds the maximum supported size. If this bit is set the payload must be considered invalid.
- Bit 10: Transmission Rejected : If during transmission MIPI CSI-2 Low Level Protocol packet was not on-time or within grace period.

16.1.4. Start timestamp

The timestamp describes the point in time immediately after the end of the Start of Transmission sequence (SoT) of the first MIPI CSI-2 long packet of a frame was received at the FPGA MIPI D-PHY interface.

If bit 10 in the Status field of the EBHSCR header is set, the Start timestamp contains the time when the MIPI CSI-2 Low-Level packet packet was discarded.

16.1.5. Stop timestamp

The timestamp describes the point in time immediately before the beginning of the End of Transmission sequence (EoT) of the last MIPI CSI-2 long packet of an image line received at the FPGA MIPI D-PHY interface.

If a multi-lane configuration (e.g. 2-Lane MIPI, 4-lane MIPI etc.) is used, the EoT arriving at the later time shall be used for the timestamp reference.

If bit 10 in the Status field of the EBHSCR header is set, the Stop timestamp contains the time when the MIPI CSI-2 Low-Level packet packet was discarded.

16.1.6. Major number specific header

Figure 16.1. Major number specific header overview for CSI2 Frame

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Flags             |  res. |   VC  |  ImH Version  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            reserved                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **Byte 0..1:** Flags
  - Bit 0: If set, the image has been processed (i.e. resized)
  - Bit 1: Reserved
  - Bit 2: Set if each line is padded to a multiple of 8 bytes and starts at an 8 byte alignment within the frame.
  - Bit 3..14: Reserved
  - Bit 15: MIPI PHY Type
    - `0`: DPHY
    - `1`: CPHY
- **Byte 2:**
  - Bit 0..3: VC (MIPI CSI-2 Virtual Channel)
  - Bit 4..7: reserved
- **Byte 3:** Image Header Version. The Image Header Version 0 is described in the payload section.

16.2. Payload

16.2.1. Capture and Replay

The following defines the Frame Header for Image Header Version 0.

CSI-2 Frame packet Format:

- The payload begins with an 128 byte frame header followed by an payload.
- The payload consist of image lines encoded according to MIPI CSI-2. Each line, including the last line of the image, is padded according to the status bits. This allows lines to start at a given memory alignment within the frame packet.

16.2.2. The Frame Header

The packet Header hold information of up to 8 sections (blocks) in the image. A section is defined by a consistent length and a consistent mipi datatype. All multi byte fields in the header are in big endian format. If a section is not present in the image, all fields of this section are set to 0.

Figure 16.2. The Frame Header

```
                       0               1               2               3
                       0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
          +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Word  0   |           |                       Payload Byte Offset                     |
          +           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Word  1   |           |                          Bytes Total                          |
          + Section 0 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Word  2   |           |        Bytes Per Line         |       Number Lines            |
          +           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Word  3   |           |          reserved             |  Error Bits   | MIPI Data Type|
          +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
          |                                                                           |
Word 4-27 |                              Section 1 to 6                               |
          |                                                                           |
          +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Word 28   |           |                       Payload Byte Offset                     |
          +           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Word 29   |           |                          Bytes Total                          |
          + Section 7 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Word 31   |           |        Bytes Per Line         |       Number Lines            |
          +           +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Word 32   |           |          reserved             |  Error Bits   | MIPI Data Type|
          +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

MIPI Data Type

- MIPI CSI-2 Data Type of this section.

Error Bits

- Bit 0: MIPI CSI-2 long packet checksum error. Set if at least one CRC error occurred in any line of this section.
- Bit 1: MIPI CSI-2 packet header contains an correctable ECC error. Set if at least any one correctable ECC error was detected in any image line of this section.
- Bit 2: MIPI CSI-2 packet header contains an uncorrectable ECC error. Set if at least any one uncorrectable ECC error was detected in any image line of this section.
- Bit 3: Receiver Decoder error. Set if at least one decoder error occurred during the reception of this section.
  - Receiver Decoder Error covers following case(s):
    - Start of Transmission (SoT) Error according to MIPI Alliance specification for CSI-2 Termination of the HS state before the complete packet has been transmitted.
- Bit 5..7 : reserved.

Number Lines

- Number of image lines in this section.

Bytes Per Line

- Number of bytes per image line.

Bytes Total

- Total number of bytes in this section including padding bytes.

Payload Byte Offset

- Byte offset of section start within the frame. The offsets starts with 0 right after this header.