FlexRay

BLF Logging Format

Specification

Version 1.8 of 2020-02-07

Status	Completed
Publisher	Vector Informatik GmbH © 2020 All rights reserved. Any distribution or copying is subject to prior written approval by Vector. Note: Hardcopy documents are not subject to change management.

Document Management

Revision list

Version	Date	Editor	Section	Changes, comments
1.0	2008-06-03	Wbn	All	Initial version created
1.1	2008-09-16	Wbn	2.10	POC state for CC2
1.2	2009-12-04	Wbn	2.3.5	Bit 22 added, description for Bit 17 updated
1.3	2009-05-19	Wbn	1	Added Disclaimer
1.4	2009-09-08	Wbn	3.8, 3.3.4	Tx Conflict (Bit 12) for VN interfaces and member <mBlfLogMask> and <mReservedW> added for CANoe/CANalyzer 7.2
1.5	2011-05-09	Wbn	3.8	Corrected mDir parameter type
1.6	2012-02-06	Hb	3.10	Clarifications
1.7	2015-04-27	Hb	3.3.4	Additional error flags (since CANoe/CANalyzer 8.5 SP3)
1.7.1	2017-04-19	Mom	All	CI and layout
1.7.2	2017-08-22	Yav	3.10.1	VBLFLEXRAYVFrStatus, POC State
1.7.3	2019-12-04	Bma	3.8	Corrected mCycle and mReserved parameters
1.8	2020-02-07	vsn	3	API has changed to standard types, e.g. uint32_t instead of DWORD, added libbinlog.so

Contents

1 Disclaimer

Severability clause - Restrictions for the usage of Vector logging data formats outside of Vector products

The format specification / access functions for the Vector BLF and ASC logging data formats are made available under the restrictions and conditions cited hereafter.

Please note that Vector Informatik neither gives any guarantee nor assumes any liability beyond compulsory legal regulations for the BLF or ASC logging format respectively as well as for the access functions to the single objects.

Vector Informatik disclaims all liability for errors which might be contained in the access functions or the format specification itself.

Vector Informatik does neither provide support for the integration into your software nor for problems occurring inside your software on the customer side.

Beyond that Vector Informatik reserves the right to change the BLF or ASC data format respectively anytime without prior notification. Therefore, the compatibility of the format is not ensured.

2 Overview

The document specifies the format of FlexRay events in the CANoe/CANalyzer BLF logging. The described structures can be used to read and write BLF logging files using the binlog.dll or libbinlog.so, which can be found in the CANoe/CANalyzer User Data folder:

<UserDataFolder>\Programming\BLF_Logging

3 Format Description

3.1 Terms and Acronyms

Term	Definition
CC	Communication controller

3.2 General

Note: unused members or flags are not used yet and must be always set to 0 if logging object is written by another application.

3.3 Common Data Types

3.3.1 Direction Flags

0 = Rx
1 = Tx
2 = Tx Request
3 and 4 are for internal use only.

3.3.2 Channel Mask

0 = Reserved or invalid
1 = FlexRay Channel A
2 = FlexRay Channel B
3 = FlexRay Channels A and B

3.3.3 CC-Types

Communication controllers (CC-Types):

0 = Architecture independent
1 = Invalid CC type (for internal use only)
2 = Cyclone I
3 = BUSDOCTOR
4 = Cyclone II
5 = Vector VN interface
6 = VN-Sync-Pulse (only in Status Event, for debugging purposes only)

3.3.4 Controller Specific Frame State Information

Note: unused bits in frame status field are not used yet and must be always set to 0 if logging object is written by another application.

Bit	Cyclone I	BUSDOCTOR	Cyclone II	VN
0	TX Conflict (TXCON)	Decoding Error (CODERR)	Syntax Error (SERR)	Syntax Error (SERR)
1	Boundary Violation (BVIOL)	Violation Error (TSSVIOL)	Content Error (CERR)	Content Error (CERR)
2	Content Error (CERR)	Header CRC Error (HCRCERR)	Slot BoundaryViolation (BVIOL)	Slot BoundaryViolation (BVIOL)
3	Syntax Error (SERR)	Frame CRC Error (FCRCERR)	Empty Slot (SLEMPTY)	Empty Slot (SLEMPTY)

Bit	Cyclone I	BUSDOCTOR	Cyclone II	VN
4	StartUP Frame indication (SUPF)	Frame End Sequence Error (FESERR)	Message Lost (MLOST)	Message Lost (MLOST)
5	NULL Frame indication (NF)	Symbol (SYMB)	Valid Frame (VAL)	Valid Frame (VAL)
6	SYNC Frame indication (SF)	Valid Frame (VAL)		TX Conflict (TXCON)
7	Valid Communication Element (VCE)	Boundary Violation Error (MASB)		Framing Error (FrmERR)
8		NIT Violation Error (NITVIOL)		Header CRC Error (HdrERR)
9		Symbol Window Violation Error (SWVIOL)		Frame CRC Error (FrmCRC)
10		Slot Overbooked Error (SOVERR)		Reserved Bit Error
11		Null Frame Error (INFE)		Tx Conflict (bus signal level failure during transmission)
12		Syncframe or Start-up Error (ISFE)		Redundancy Error (dual channel frame with different payload or header flags detected)
13		Frame ID Error (FIDE)		Bus Error (Spikes detected)
14		Cycle Counter Error (CCE)		Unknown error
15		Static Payload Length Error (PLSE)		

3.3.5 Frame Flags

Description of frame flags.

Bit	Description
0	1 = Null frame.
1	1 = Data segment contains valid data
2	1 = Sync bit
3	1 = Startup flag
4	1 = Payload preamble bit
5	1 = Reserved bit
6	1 = Error flag (error frame or invalid frame)
7	Reserved
8	Internally used in CANoe/CANalyzer
9	Internally used in CANoe/CANalyzer
10	Internally used in CANoe/CANalyzer
11	Internally used in CANoe/CANalyzer
12	Internally used in CANoe/CANalyzer

Bit	Description
13	Internally used in CANoe/CANalyzer
14	Internally used in CANoe/CANalyzer
15	1 = Async. monitoring has generated this event
16	1 = Event is a PDU
17	Valid for PDUs only. The bit is set if the PDU is valid (either if the PDU has no update bit, or the update bit for the PDU was set in the received frame).
18	Reserved
19	1 = Raw frame (only valid if PDUs are used in the configuration). A raw frame may contain PDUs in its payload
20	1 = Dynamic segment 0 = Static segment
21	This flag is only valid for frames and not for PDUs. 1 = The PDUs in the payload of this frame are logged in separate logging entries. 0 = The PDUs in the payload of this frame must be extracted out of this frame. The logging file does not contain separate PDU-entries.
22	Valid for PDUs only. The bit is set if the PDU has an update bit

The reserved bits and the bits which are for internally CANoe/CANalyzer usage must be ignored from other applications. Other applications must set these bits to 0 when writing logging files.

3.4 Obsolete Types

The types listed below are not provided by CANoe and CANalyzer applications any more.

1. VBLFLEXRAYData
2. VBLFLEXRAYSync
3. VBLFLEXRAYStatusEvent

3.5 VBLFLEXRAYV6StartCycleEvent

Description: Start of cycle event transmitted by the hardware interface on a FlexRay channel.

Note: this type is provided only for compatibility with previews logging file formats. Applications should use the VBLFLEXRAYVFrStartCycle (section 3.9) type instead.

Parameter	Type	Description
mHeader	VBLObjectHeader	Common header type
mChannel	uint16_t	Application channel
mDir	uin8_t	See 3.3.1
mLowTime	uin8_t	Additional time field in simulation
mFPGATick	uint32_t	Timestamp generated from xModule
mFPGATickOverflow	uint32_t	Overflow counter of the timestamp
mClientIndex	uint32_t	Client index of send node. Must be set to 0 if file is written from other applications
mClusterTime	uint32_t	Relative cluster time, from 0 to cycle length
mDataBytes[2]	uin8_t	Array of data bytes
mReserved	uint16_t	Reserved

3.6 VBLFLEXRAYV6Message

Description: FlexRay Message received or transmitted on a FlexRay channel.

Note: this type is provided only for compatibility with previews logging file formats. Applications should use the VBLFLEXRAYVFrReceiveMsgEx type (section 3.8) instead.

Parameter	Type	Description
mHeader	VBLObjectHeader	Common header type
mChannel	uint16_t	Application channel
mDir	uin8_t	See 3.3.1
mLowTime	uin8_t	Additional time field in simulation
mFPGATick	uint32_t	Timestamp generated from xModule
mFPGATickOverflow	uint32_t	Overflow counter of the timestamp
mClientIndex	uint32_t	Client index of send node
mClusterTime	uint32_t	Relative cluster time, from 0 to cycle length
mFrameId	uint16_t	slot identifier
mHeaderCRC	uint16_t	CRC of the frame header
mFrameState	uint16_t	V6 framestate: 0 Payload preample indicator bit 1 Sync. frame indicator 2 Reserved bit 3 Null frame indicator 4 Startup frame indicator 5-7 Frame state format mask (see below) Bit 5-7 meaning: 0 (0x00) Motorola V.6 1 (0x20) reserved 2 (0x40) BusDoctor 3 (0x60) reserved 4 (0x80) FlexCard Cyclone 5 (0xA0) reserved 6 (0xC0) reserved 7 (0xE0) reserved
mLength	uin8_t	Payload length
mCycle	uin8_t	Current cycle number
mHeaderBitMask	uin8_t	Bit 0 = NMBit, Bit 1 = SyncBit, Bit 2 = Reserved
mReserved1	uin8_t	Reserved
mReserved2	uint16_t	Reserved
mDataBytes[64]	uin8_t	Payload

3.7 VBLFLEXRAYVFrReceiveMsg

Description: FlexRay message received or transmitted on FlexRay bus.

Note: this type is provided only for compatibility with previews logging file formats. Applications should use the VBLFLEXRAYVFrReceiveMsgEx type (section 3.8) instead.

Parameter	Type	Description
mHeader	VBLObjectHeader	Common header type

Parameter	Type	Description
mChannel	uint16_t	Application channel
mVersion	uint16_t	Object version, for internal use
mChannelMask	uint16_t	See 3.3.2
mDir	uin8_t	See 3.3.1
mClientIndex	uint32_t	Client index of send node. Must be set to 0 if file is written from other applications.
mClusterNo	uint32_t	Number of cluster: channel number - 1
mFrameId	uint16_t	Slot identifier
mHeaderCRC1	uint16_t	Header CRC FlexRay channel 1 (A)
mHeaderCRC2	uint16_t	Header CRC FlexRay channel 2 (B)
mByteCount	uint16_t	Payload length in bytes
mDataCount	uint16_t	Number of bytes of the payload stored in mDataBytes. If the CC-frame buffer was too small to receive the complete payload, then mDataCount is smaller than mByteCount.
mCycle	uin8_t	Cycle number
mTag	uint32_t	Type of communication controller, see 3.3.3
mData	uint32_t	Controller specific frame state information, see 3.3.4
mFrameFlags	uint32_t	See description of flags, see 3.3.5
mAppParameter	uint32_t	Not used, reserved
mDataBytes[254]	uin8_t	Payload

3.8 VBLFLEXRAYVFrReceiveMsgEx

Description: FlexRay message or PDU received or transmitted on FlexRay bus.

Parameter	Type	Description
mHeader	VBLObjectHeader	Common header type
mChannel	uint16_t	Application channel
mVersion	uint16_t	Object version, for internal use
mChannelMask	uint16_t	See 3.3.2
mDir	uint16_t	See 3.3.1
mClientIndex	uint32_t	Client index of send node. Must be set to 0 if file is written from other applications.
mClusterNo	uint32_t	Number of cluster: channel number - 1
mFrameId	uint16_t	Slot identifier
mHeaderCRC1	uint16_t	Header CRC FlexRay channel 1 (A)
mHeaderCRC2	uint16_t	Header CRC FlexRay channel 2 (B)
mByteCount	uint16_t	Payload length in bytes
mDataCount	uint16_t	Number of bytes of the payload stored in mDataBytes. If the CC-frame buffer was too small to receive the complete payload, then mDataCount is smaller than mByteCount.
mCycle	uint16_t	Cycle number
mTag	uint32_t	Type of communication controller, see 3.3.3

Parameter	Type	Description
mData	uint32_t	Controller specific frame state information, see 3.3.4
mFrameFlags	uint32_t	See description of flags, see 3.3.5
mAppParameter	uint32_t	Not used, reserved
mFrameCRC	uint32_t	Frame CRC
mFrameLengthNS	uint32_t	Length of frame in ns (only valid for frames received in asynchronous mode, bit 15 is set in the frame flags)
mFrameId1	uint16_t	For PDUs only: This is the slot ID of the frame which contains this PDU
mPDUOffset	uint16_t	For PDUs only: offset in bytes of PDU in an owner (raw) frame
mBlfLogMask	uint16_t	Only valid for frames. Every stands for one PDU. If set, the PDU must be extracted out of the frame. The bit order is the PDU order in the frame starting with the PDU with the smallest offset.
mReservedW	uint16_t	Reserved
mReserved[6]	uint32_t	Reserved
mDataBytes[254]	uin8_t	Payload

3.9 VBLFLEXRAYVFrStartCycle

Description: FlexRay StartCycle event transmitted by the FlexRay hardware.

3.9.1 Controller Specific Information

Field	Cyclone I	Cyclone II	VN-Interface
mData[0]	Rate correction of CC, read from RCVR register	Sync correction of CC, read from RCV register	Sync correction of CC, read from RCV register
mData[1]	Offset correction of CC, read from OCVR register	Offset correction of CC, read from OCV register	Offset correction of CC, read from OCV register
mData[2]		Cycles with no correction, read from CCEV register	Cycles with no correction, read from CCEV register
mData[3]		Cycles with correction in passive mode, read from CCEV register	Cycles with correction in passive mode, read from CCEV register
mData[4]		Sync Frame status, read from SFS register	Sync Frame status, read from SFS register

3.9.2 Descriptions of Parameters

Parameter	Type	Description
mHeader	VBLObjectHeader	Common header type
mChannel	uint16_t	Application channel
mVersion	uint16_t	Object version, for internal use
mChannelMask	uint16_t	See 3.3.2
mDir	uin8_t	See 3.3.1
mCycle	uin8_t	Cycle number

Parameter	Type	Description
mClientIndex	uint32_t	Client index of send node
mClusterNo	uint32_t	Number of cluster: channel number - 1
mNmSize	uint16_t	Length of NM-Vector in bytes
mDataBytes[12]	uin8_t	Array of databytes (NM vector max. length)
mTag	uint32_t	Type of communication controller, see 3.3.3
mData[5]	uint32_t	Driver flags for internal usage
mReserved	uint16_t	Reserved

3.10 VBLFLEXRAYVFrStatus

Description: The content of the FlexRay status event depends on the type of hardware interface. The event is generated in one of the following situations:

- A symbol is received
- The POC state or wakeup state of the CC has changed
- The status of the symbol window has changed

3.10.1 Controller Specific Information

CC-Type: Cylone I

Field	Description
mData[0]	Content of Protocol state register (PSR)
mData[1]	Content of Module config register (MCR0)

CC-Type: BUSDOCTOR

Field	Description
LOW-uint16_t of mData[0]	Symbol length
HI-uint16_t of mData[0]	Flags: 1 = possible CAS
mData[1]	Reserved

CC-Type: VN-Interface

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>mData[0]</td><td>POC state of E-Ray register CCSV. Only valid
for Vector interfaces
if wakeup state is 0
POC State in the operation control phase:
<table>
<tr><td>Mask</td><td>Description</td></tr>
<tr><td>0x00</td><td>DEFAULT_CONFIG</td></tr>
<tr><td>0x01</td><td>READY</td></tr>
<tr><td>0x02</td><td>NORMAL_ACTIVE</td></tr>
<tr><td>0x03</td><td>NORMAL_PASSIVE</td></tr>
<tr><td>0x04</td><td>HALT</td></tr>
<tr><td>0x05</td><td>MONITOR_MODE</td></tr>
<tr><td>0x0F</td><td>CONFIG</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td></td><td>POC State in the wake-up phase:
<table>
<tr><td>Mask</td><td>Description</td></tr>
<tr><td>0x10</td><td>WAKEUP_STANDBY</td></tr>
<tr><td>0x11</td><td>WAKEUP_LISTEN</td></tr>
<tr><td>0x12</td><td>WAKEUP_SEND</td></tr>
<tr><td>0x13</td><td>WAKEUP_DETECT</td></tr>
</table>
POC State in the start-up phase:
<table>
<tr><td>Mask</td><td>Description</td></tr>
<tr><td>0x20</td><td>STARTUP_PREPARE</td></tr>
<tr><td>0x21</td><td>COLDSTART_LISTEN</td></tr>
<tr><td>0x22</td><td>COLDSTART_COLLISION_RESOLUTION</td></tr>
<tr><td>0x23</td><td>COLDSTART_CONSISTENCY_CHECK</td></tr>
<tr><td>0x24</td><td>COLDSTART_GAP</td></tr>
<tr><td>0x25</td><td>COLDSTART_JOIN</td></tr>
<tr><td>0x26</td><td>INTEGRATION_COLDSTART_CHECK</td></tr>
<tr><td>0x27</td><td>INTEGRATION_LISTEN</td></tr>
<tr><td>0x28</td><td>INTEGRATION_CONSISTENCY_CHECK</td></tr>
<tr><td>0x29</td><td>INITIALIZE_SCHEDULE</td></tr>
<tr><td>0x30</td><td>ABORT_STARTUP</td></tr>
<tr><td>0x31</td><td>STARTUP_SUCCESS</td></tr>
</table>
All other values are reserved.</td></tr>
<tr><td>LOW-uint16_t of mData[1]</td><td>Bit field indicating the symbol window status of the controller and the event source.
<table>
<tr><td>Value</td><td>Meaning</td></tr>
<tr><td>1</td><td>SESA (Syntax error in symbol window channel A)</td></tr>
<tr><td>2</td><td>SBSA (Slot boundary violation in symbol window channel A)</td></tr>
<tr><td>4</td><td>TCSA (Transmission conflict in symbol window channel A)</td></tr>
<tr><td>8</td><td>SESB (Syntax error in symbol window channel B)</td></tr>
<tr><td>16</td><td>SBSB (Slot boundary violation in symbol window channel B)</td></tr>
<tr><td>32</td><td>TCSB (Transmission conflict in symbol window channel B)</td></tr>
<tr><td>64</td><td>The event was generated from a controller-independent protocol interpreter (Spy).</td></tr>
<tr><td>128</td><td>Cold-start helper POC indicator, if set, event contains the POC state of the cold-start helper</td></tr>
</table>
All other bits are reserved. CANoe/CANalyzer may set some of these bits to 1. Other applications must set them to 0.</td></tr>
<tr><td>HI-uint16_t of mData[1]</td><td>Symbol length in bit times. Only valid for symbol type 4 and if the value is not zero.</td></tr>
</table>

3.10.2 Attributes

<table>
<tr><th>Parameter</th><th>Type</th><th>Description</th></tr>
<tr><td>mHeader</td><td>VBLObjectHeader</td><td>Common header type</td></tr>
<tr><td>mChannel</td><td>uint16_t</td><td>Application channel</td></tr>
<tr><td>mVersion</td><td>uint16_t</td><td>Object version, for internal use</td></tr>
<tr><td>mChannelMask</td><td>uint16_t</td><td>See 3.3.2</td></tr>
<tr><td>mCycle</td><td>uin8_t</td><td>Cycle number</td></tr>
<tr><td>mClientIndex</td><td>uint32_t</td><td>Client index of send node. Must be set to 0 if file is written from other applications</td></tr>
<tr><td>mClusterNo</td><td>uint32_t</td><td>Number of cluster: channel number – 1</td></tr>
<tr><td>mWus</td><td>uint32_t</td><td>WakeUp state. Only valid
for Vector interfaces and for Cyclone II,
if symbol is void (mReserved[0] = 0)
<table>
<tr><td>Value</td><td>Meaning (see E-Ray specification for a detailed description)</td></tr>
<tr><td>0</td><td>UNDEFINED</td></tr>
<tr><td>1</td><td>RECEIVED_HEADER</td></tr>
<tr><td>2</td><td>RECEIVED_WUP</td></tr>
<tr><td>3</td><td>COLLISION_HEADER</td></tr>
<tr><td>4</td><td>COLLISION_WUP</td></tr>
<tr><td>5</td><td>COLLISION_UNKNOWN</td></tr>
<tr><td>6</td><td>TRANSMITTED</td></tr>
<tr><td>7</td><td>EXTERNAL_WAKEUP</td></tr>
<tr><td>8</td><td>WUP_RECEIVED_WITHOUT_WUS_TX</td></tr>
</table></td></tr>
<tr><td>mCcSyncState</td><td>uint32_t</td><td>Sync-State, only valid
for Cyclone 1
for Cyclone II if the wakup state value is 0.
0 = Not synced passive
1 = Synced active
2 = Not synced</td></tr>
<tr><td>mTag</td><td>uint32_t</td><td>Type of communication controller, see 3.3.3</td></tr>
<tr><td>mData[2]</td><td>uint32_t</td><td>Driver flags for internal usage</td></tr>
<tr><td>mReserved[0]</td><td>uint16_t</td><td>If this value is not zero, then the event contains the information about a symbol.
0 = Void
1 = CAS
2 = MTS
3 = WUS
4 = Network interface doesn’t provide a symbol interpretation, e.g. if spy-mode is used or the BUSDOCTOR interface. In spy mode, the symbol length is stored in the HI-uint16_t of mData[1].</td></tr>
<tr><td>mReserved[15]</td><td>uint16_t</td><td>Reserved</td></tr>
</table>

3.11 VBLFLEXRAYVFrError

Description: FlexRay Error event transmitted by the FlexRay hardware.

3.11.1 Controller Specific Information

CC-Type: Cylone I

Field	Description
mData[0]	Error flags from driver API

CC-Type: Cylone II

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>mData[0]</td><td>Error packet flag:
0 = No error
1 = FlexCard overflow
2 = PCO error mode changed
3 = Sync frames below minimum
4 = Sync frame overflow
5 = Clock correction failure
6 = Parity error
7 = Receive FIFO overrun
8 = Empty FIFO access
9 = Illegal input buffer access
10 = Illegal output buffer access
11 = Syntax error
12 = Content error
13 = Slot boundary violation
14 = Transmission across boundary
15 = Latest transmit violation</td></tr>
<tr><td>mData[1]</td><td>uint32_t layout depends on the error packet value (see previous row)
<table>
<tr><td>Error packet</td><td>Description</td></tr>
<tr><td>2</td><td>0 = Unknown state
1 = FlexRay protocol spec. > CONFIG
2 = FlexRay protocol spec. > NORMAL_ACTIVE
3 = FlexRay protocol spec. > NORMAL_PASSIVE
4 = FlexRay protocol spec. > HALT
5 = FlexRay protocol spec. > READY
6 = FlexRay protocol spec. > STARTUP
7 = FlexRay protocol spec. > WAKEUP</td></tr>
<tr><td>3 or 4</td><td>Bits 0..3 > Sync frames even on channel A
Bits 4..7 > Sync frames even on channel B
Bits 8..11 > Sync frames odd on channel A
Bits 12..15 > Sync frames odd on channel B</td></tr>
<tr><td>5</td><td>Bit 0 > Missing rate correction
Bit 1 > Rate correction limit reached
Bit 2 > Offset correction limit reached
Bit 3 > Missing offset correction
Bit 4..7 > Sync frames even on channel A
Bits 8..11 > Sync frames even on channel B</td></tr>
</table>
</td></tr>
</table>

Field	Description	
		Bits 12..15 > Sync frames odd on channel A Bits 16..19 > Sync frames odd on channel B
	11.. 15	LOW-uint16_t of mData[1] > Channel HI-uint16_t of mData[1] > Slot count

CC-Type: BUSDOCTOR

Field	Description
mData[0]	Error flags from driver API

CC-Type: VN-Interface

Field	Description		
mData[0]	Error tag: 0 = FR_ERROR_POC_MODE 1 = FR_ERROR_SYNC_FRAMES_BELOWMIN 2 = FR_ERROR_SYNC_FRAMES_OVERLOAD 3 = FR_ERROR_CLOCK_CORR_FAILURE 4 = FR_ERROR_NIT_FAILURE 5 = FR_ERROR_CC_ERROR 6 = FR_ERROR_OVERFLOW		
mData[1] and mData[2]	uint32_t layout depends on the error tag value (see previous row):		
	Error tag	Value or Bit-Range	Description
	0	0	FR_ERROR_POC_ACTIVE
		1	FR_ERROR_POC_PASSIVE
		2	FR_ERROR_POC_COMM_HALT
	1 or 2	Bits 0..3	Sync frames even on channel A
		Bits 4..7	Sync frames even on channel B
		Bits 8..11	Sync frames odd on channel A
		Bits 12..15	Sync frames odd on channel B
	3	Bit 0	Missing rate correction
		Bit 1	Missing rate correction limit reached
		Bit 2	Offset correction limit reached
		Bit 3	Missing offset correction
		Bits 4..19	Clock correction failed counter
		Bit 20..23	Sync frames even on channel A
		Bit 24..27	Sync frames even on channel B
		Bit 28..31	Sync frames odd on channel A
		Bit 32..35	Sync frames odd on channel B
	4	1	FR_ERROR_NIT_SENA
		2	FR_ERROR_NIT_SBNA
		4	FR_ERROR_NIT_SENB
		8	FR_ERROR_NIT_SBNB
	5	0x00000001	POC Error Mode Changed

Field	Description		
		0x00000004	Sync Frames Below Minimum
		0x00000008	Sync Frame Overflow
		0x00000010	Clock Correction Failure
		0x00000040	Parity Error, data from MHDS (internal ERay error)
		0x00000200	Illegal Input Buffer Access (internal ERay error)
		0x00000400	Illegal Output Buffer Access (internal ERay error)
		0x00000800	Message Handler Constraints Flag data from MHDF (internal ERay error)
		0x00010000	Error Detection on channel A, data from ACS
		0x00020000	Latest Transmit Violation on channel A
		0x00040000	Transmit Across Boundary on Channel A
		0x01000000	Error Detection on channel B, data from ACS
		0x02000000	Latest Transmit Violation on channel B
		0x04000000	Transmit Across Boundary on Channel B

3.11.2 Attributes

Parameter	Type	Description
mHeader	VBLObjectHeader	Common header type
mChannel	uint16_t	Application channel
mVersion	uint16_t	Object version, for internal use
mChannelMask	uint16_t	See 3.3.2
mCycle	uin8_t	Cycle number
mClientIndex	uint32_t	Client index of send node. Must be set to 0 if file is written from other applications
mClusterNo	uint32_t	Number of cluster: channel number - 1
mTag	uint32_t	Type of communication controller, see 3.3.3
mData[4]	uint32_t	Driver flags for internal usage
mReserved	uint16_t	Reserved