USBtingo EB – USB to CAN-FD Interface
Protocol description

1. USB Communication

The application firmware enumerates as a USB vendor device.

idVendor:	1fc9
idProduct:	8320
Product:	USBtingo
Manufacturer:	Fischl.de

The VID/PID pair is sublicenced from NXP to Fischl.de.

Interface 0 endpoint configuration:
EP0: Control/Setup - Command pipe
EP1 IN Interrupt – Status – 64 Byte
EP2 IN: Bulk 512 Bytes - Logic level bitstream
EP3 OUT: Bulk 512 Bytes - CAN messages from host
EP3 IN: Bulk 512 Bytes - CAN messages to host

The device accepts only High-Speed mode.

The CAN module is a Bosch M_CAN. If you need a more detailed description of the registers or flags used in message transfers, you can find additional details in the M_CAN User Manual:
https://www.bosch-semiconductors.com/media/ip_modules/pdf_2/m_can/mcan_users_manual_v331.pdf

2. Commands on EP0

Command	Off	wValue		wIndex		wLength		Data phase
	Index:	2	3	4	5	6	7	
0x01 CMD_ECHO		echoval	0	0	0	2	0	IN: echoval
0x02 CMD_START_BOOTLOADER		0x37	0x52	0	0	0	0	
0x03 CMD_GET_DEVICEINFO		0	0	0	0	12	0	OUT: Device information
0x04 CMD_SET_PROTOCOL	X	protocol	flags	0	0	0	0	
0x05 CMD_SET_BAUDRATE	X	nominal=0/data=1	0	0	0	4	0	OUT: Baudrate in Hz (uint32 = 4 bytes)
0x06 CMD_SET_MODE		mode	0	0	0	0	0	
0x07 CMD_CLEAR_ERRORFLAGS		clearmask	0	0	0	0	0	
0x08 CMD_GET_STATUSREPORT		0	0	0	0	0	0	IN: 64 byte statusreport (see „Statusreports“)
0x20 CMD_FILTER_DISABLE_ALL		0	0	0	0	0	0	
0x21 CMD_FILTER_SET_STD		filterid	0	0	0	12	0	OUT: enabled (uint32), filter (uint32), mask (uint32)
0x22 CMD_FILTER_SET_EXT		filterid	0	0	0	12	0	OUT: enabled (uint32), filter (uint32), mask (uint32)
0x30 CMD_MCAN_REG_READ		address		0	0	number of bytes to read		IN: values read starting at given MCAN offset
0x31 CMD_MCAN_REG_WRITE		address		0	0	number of bytes to write		OUT: values to write to given MCAN offset
0x40 CMD_LOGIC_SETCONFIG		transfermode=0	clockdivider	0	0	0	0	
0x41 CMD_LOGIC_GETTXERRORS		0	0	0	0	4	0	IN: number of TX errors (uint32 = 4 bytes)

Off: allowed only in Off mode.

Parameters:

protocol

0x00	Classic CAN 2.0	
0x01	CAN FD ISO	
0x02	CAN FD NON-ISO (Bosch CAN FD Specification V1.0)	

flags

Bit 0	BRSE	Bit rate switching for transmissions if CAN Fd is activated
Bit 1	TXP	Transmit pause
Bit 2	EFBI	Edge filtering during bus integration
Bit 3	PXHD	Protocol exception handling disable. (0 = exception handling enabled, 1 = disabled)
Bit 4	DAR	Disable automatic retransmission. (0 = automatic retransmission enabled, 1 = disabled)

mode

0x00	Off	
0x01	Active	
0x02	Listen-only	Bus monitoring mode (see ISO11898-1, 10.12 Bus monitoring).

Example using Python and libusb1:

```
# set baudrate:
usbhandle.controlWrite(usb1.TYPE_VENDOR, CMD_SET_BAUDRATE, 0, 0, struct.pack("<I", bitrate))
usbhandle.controlWrite(usb1.TYPE_VENDOR, CMD_SET_BAUDRATE, 1, 0, struct.pack("<I", data_bitrate))
```

Device information:

Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0	Application firmware version minor							
1	Application firmware version major							
2	Hardware model identifier							
3	Number of CAN channels = 1							
4...7	Device unique identifier (LSB first)							
8...11	CAN clock in Hz (LSB first)							

Filters:
There are two filter banks: 32 for standard and 32 for extended frames. After power up filterid 0 of each bank is enabled to accept all messages.
DisableAll: Disable all filters. With all filters disabled, no message is received! You have to set at least one filter.
Filterid 0..31 Filter identifier
Filter Filter value
Mask Filter mask. 0 = don‘t care, 1 = include in comparison
Example „Filter only ID 0x123“:
Filter = 0x123
Mask = 0x7ff

3. Statusreports on EP1 IN

Status reports are automatically sent to the host every second. Instead of using the interrupt transfer via EP1, the status report can also be retrieved as a control transfer using the „Get_Statusreport“ command. Note that the statistical values are reset after each retrieval.
All values including overflow flags are cleared on change of operating mode (except change to OFF). The overflow flags (RXOVF and TXEOVF) can be cleared with command „Clear Overflow Errorflags“.

Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0	0x80							
1	Operation mode							
2	0	0	0	0	0	0	TXEOVF	RXOVF

3	0	0	0	0	0	0	0	0
4	PROCTS<7:0>							
5	PROCTS<15:8>							
6	PROCTS<23:16>							
7	PROCTS<31:24>							
8	TEC<7:0>							
9	RP	REC<6:0>						
10	CEL<7:0>							
11	-	-	-	-	-	-	-	-
12	BO	EW	EP	ACT<1:0>		LEC<2:0>		
13	-	-	-	-	-	DLEC<2:0>		
14		TDCV<6:0>						
15	-	-	-	-	-	-	-	-
16...19	Stats: number of standard frames (LSB first)							
20...23	Stats: number of extended frames (LSB first)							
24...27	Stats: number of data bytes without bitrate switching (LSB first)							
28...31	Stats: number of data bytes with bitrate switching (LSB first)							

Operation mode: Current operation mode. See „Set mode“ command
PROCTS: Processing timestamp
RXOVF: Overflow while receiving CAN frames
TXEOVF: Overflow while receiving TX events
TEC: Transmit Error Counter
REC: Receive Error Counter
RP: Receive Error Passive. 1 = The Receive Error Counter has reached the error passive level of 128.
CEL: CAN Error Logging
LEC: Last Error Code
ACT: Activity
EP: Error Passive. 1 = The MCAN is in the Error_Passive state.
EW: Warning Status. 1 = At least one of error counter has reached the Error_Warning limit of 96.
BO: Bus_Off Status. 1 = The MCAN is in Bus_Off state.
DLEC: Data Phase Last Error Code
TDCV: Transmitter Delay Compensation Value

Statistics:
Number of frames and data bytes since the last transmission/request of status report. These values can be used to calculate the bus load:
Without bit stuffing:

#bits = 47 * #standard_frames + 65 * #extended_frames + 8 * #data_bytes

#bitsBRS = 8 * #data_bytesBRS

With estimated bit stuffing:

#bits = 50 * #standard_frames + 70 * #extended_frames + 9 * #data_bytes

#bitsBRS = 9 * #data_bytesBRS

Bus load:

bus load percentage =

(#bits (per second) / bitrate + #bitsBRS (per second) / bitrateBRS) * 100

4. Logic level bitstream on EP2 IN:

If the logic recording function is activated (see command „LOGIC_SETCONFIG“), USBtingo sends permanent logic levels via EP2. The packets always contain 512 bytes of bitstream. This bit stream is generated by sampling the CAN RX signal with the configured rate (up to 40Msps). Least significant bit first.

5. Messages on EP3 IN (CAN to host)

EP3 IN transfers CAN messages to host. There are two different message types. RX messages contain received CAN frames, TX event messages inform about sent CAN messages.

Each message is initiated by a header. A USB packet can contain several messages.

5.1. Header

Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0	MessageType							
1	MessageSize							
2	0							
3	0							

MessageType 0x80 Skip message
0x81 RX message
0x82 TX event message
MessageSize Size of message in words (32 bit). Counting starts at byte 4. E.g. RX Frame with DLC=8 has MessageSize=4, TX event message has MessageSize=3

5.2. Skip message (0x80)

Skip the next MessageSize words. This message is used to pad USB packages. This allows the data flow to be optimized.

5.3. RX message (0x81)

Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
4	PROCTS<7:0>							
5	PROCTS<15:8>							
6	PROCTS<23:16>							
7	PROCTS<31:24>							
8	ID<7:0>							
9	ID<15:8>							
10	ID<23:16>							
11	ESI	XTD	RTR	ID<28:24>				
12	RXTS<7:0>							
13	RXTS<15:8>							
14	-	-	FDF	BRS	DLC<3:0>			
15	ANMF	FIDX<6:0>						

16 … 84	Data byte 0..n

PROCTS Processing timestamp
ID Standard or extended CAN identifier depending on bit XTD. A standard identifier is stored into ID[28:18].
RTR 1 = Remote transmission request bit, 0 = normal message
XTD Extended Identifier. 0: 11-bit standard identifier, 1: 29-bit extended identifier.
ESI Error State Indicator. 0: Transmitting node is error active, 1: Transmitting node is error passive.
RXTS Rx Timestamp
DLC Data length code
BRS Bit Rate Switch
FDF FD Format
ANMF Accepted Non-matching Frame
FIDX Filter Index
Data Data bytes. Pad with zeros to be 32 Bit aligned!

RX Timestamping:
There are two different timestamps:

- RXTS – this is a 16bit counter stored by CAN module on SOF.
- PROCTS – this is a 32bit counter stored by firmware when processing a message (copy message from CAN fifo to internal fifo).

Both timestamps are derived from the same system clock. The RXTS is incremented with 100kHz, the PROCTS 100kHz/4096. Both timestamps are reset at command "SetMode".
Under the condition that the USB packets are taken from the host in time (processing is not delayed more than 650ms), both timestamps can be merged to a large 44bit value:

```
diff = (procts & 0xF) - (rxts >> 12)
if diff < 0: diff = diff + 0x10
ts = ((procts - d) << 12) | (rxts & 0xfff)
```

One timer step equals 10us. The 32bit and thus the generated 44bit timer overflows after approx. 5.5 years of continuous operation.
The accuracy depends on the system crystal (+/-10ppm).

5.4. TX event message (0x82)

Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
4	PROCTS<7:0>							
5	PROCTS<15:8>							
6	PROCTS<23:16>							
7	PROCTS<31:24>							
8	ID<7:0>							
9	ID<15:8>							
10	ID<23:16>							
11	ESI	XTD	RTR	ID<28:24>				

12	TXTS<7:0>			
13	TXTS<15:8>			
14	ET	FDF	BRS	DLC<3:0>
15	TXMM<7:0>			

Most values correspond to those from the TX message - see Section 4 "TX messages". Additional:

PROCTS	Processing timestamp
TXTS	Tx Timestamp
ET	0x1 = Tx event, 0x2 = Transmission in spite of cancellation
TXMM	Message marker. Copied from TX message for identification.

6. Messages on EP3 OUT (Host to CAN)

Several TX messages can be contained in one USB packet.

TX message

Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0	0x01 (MessageType = CAN_TX)							
1	MessageSize							
2	0							
3	0							
4	ID<7:0>							
5	ID<15:8>							
6	ID<23:16>							
7	ESI	XTD	RTR	ID<28:24>				
8	0							
9	0							
10	EFC	0	FDF	BRS	DLC<3:0>			
11	TXMM							
12 … 76	Data byte 0..n							

MessageType	1 = TX via Fifo
MessageSize	Size of message in words (32 bit). Counting starts at byte 4. E.g. Frame with DLC=0 has MessageSize=2.
ID	Standard or extended CAN identifier depending on bit XTD. A standard identifier is stored into ID[28:18].
RTR	1 = Remote transmission request bit, 0 = normal message
XTD	Extended Identifier. 0: 11-bit standard identifier, 1: 29-bit extended identifier.
ESI	Error State Indicator. 0: Transmitting node is error active, 1: Transmitting node is

	error passive.
DLC	Data length code
BRS	Bit Rate Switch
FDF	FD Format
EFC	Event fifo control. Set 1 to get TX event message.
TXMM	Message marker. Copied to TX event message for identification.
Data	Data bytes. Pad with zeros to be 32 Bit aligned!