Overview

The Controller Area Network (CAN) is a highly reliable serial bus protocol defined in the Bosch CAN specifications for standard CAN 2.0B and CAN FD, as well as ISO 11898-1:2024.
The TES CAN Flexible Data-Rate Controller IP core is a Hardware IP core written in VHDL. The core is intended for use in a System-on-Chip (SoC) environment. It can be integrated into a wide range of applications and target technologies. The code is synthesizable for Application Specific Integrated Circuits (ASIC) and Field Programmable Gate Arrays (FPGA).

Features

- CAN 2.0B protocol compatible (ISO 11898-1).
- CAN Flexible Data-Rate (FD) protocol compatible (ISO 11898-1:2024).
- Supported Frame Formats:
  - CAN Base: 11-bit Identifier and constant bit rate.
  - CAN Extended: 29-bit Identifier and constant bit rate.
  - CAN FD Base: CAN Base format with dual bit rate.
  - CAN FD Extended: CAN Extend with dual bit rate.
- Data rates:
  - Up to 1 Mbit/s for CAN 2.0B.
  - Up to 8 Mbit/s for CAN FD.
- Payload sizes:
  - CAN Base/Extended: 0...8 bytes.
  - CAN FD Base/Extended: 0…64 bytes.
- Clock Prescaler provides a wide frequency range.
- Flexible interface configurations.
- Receive frame filters.
- Bosch CAN Reference model certified for CAN 2.0B functionality.
- CAN 2.0B functionality certified for aviation-related applications.
- CAN FD functionality working in FPGA application.
- CAN FD in Silicon Q1’25.

Function

MAC Device

The TES CAN FD Controller IP core is a device positioned in the OSI model in the Medium Access Control (MAC) sublayer section of the Data Link Layer (Layer 2).

The main tasks of an MAC layer device are:

- Message framing.
- Transmission and reception of frames.
- Arbitration.
- Error detection and signaling.
- Fault confinement.
- Bit timing and synchronization.

Logical Link Control (LLC) Support

To achieve a high application flexibility for this core, the acceptance filtering and buffering of received messages is provided for Logical Link Control (LLC, part of layer 2 of the OSI model) sublayer application support, which may be implemented using hardware, software, or a combination of both.

RX Filter

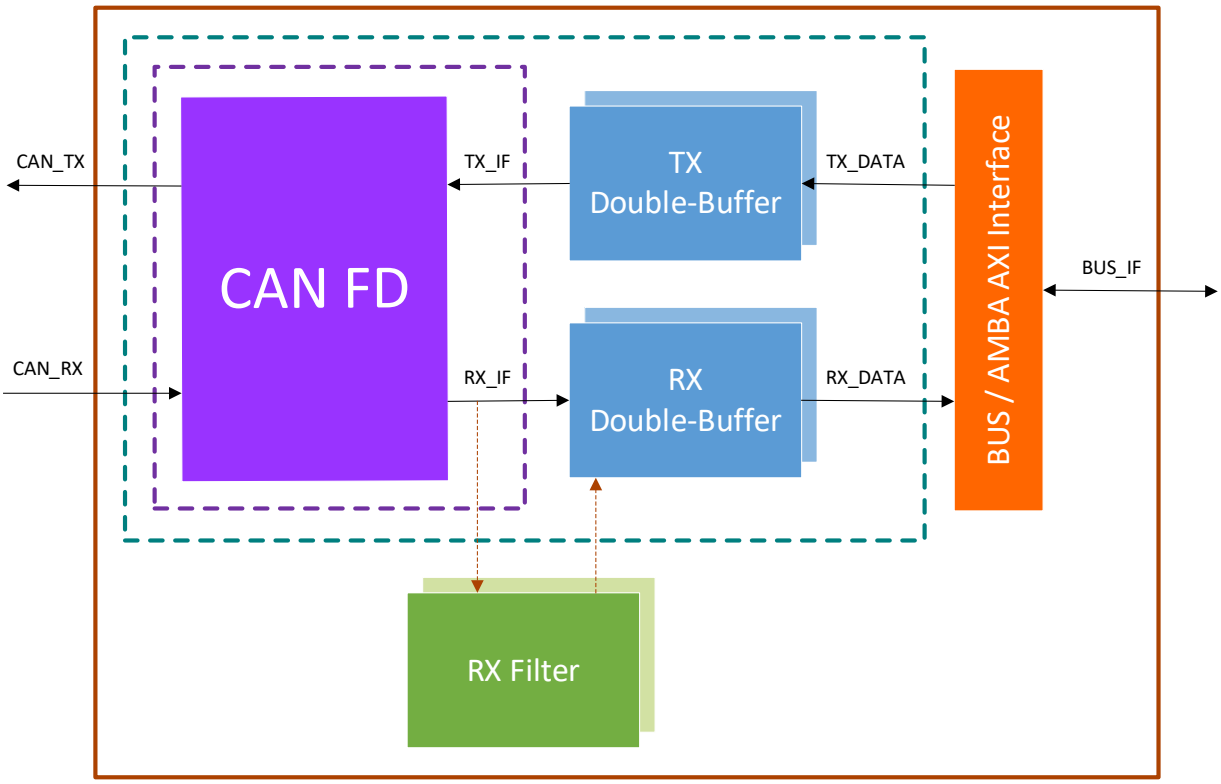
Multiple RX frame filter can be added. The filter can be masked for standard and extended identifier and the frame format. Each filter can be enabled and disabled separately. Each filter has a separate interrupt enable and interrupt flag.

Interface Configurations

The TES CAN FD Controller IP core can be integrated into SoC device or other applications using one of the following configurations:

- Core interface with a memory interface.
- Core interface double buffers for transmit and receive. Data interface to the double buffers.
- BUS / AMBA AXI interface for access via internal SoC bus.

The RX Filter can be added in each variant.



Core Interface with a memory interface

The CAN FD IP core interface is defined in following table. There is full flexibility to configure and control the CAN FD to transmit and receive frames.

Signal name	Type	Description
clk	In	System clock
reset_n	In	Reset input
Synchronization Nominal		
tseg1_i	In[5:0]	Time segment 1 nominal
tseg2_i	In[4:0]	Time segment 2 nominal

sjw_i	In[3:0]	Sample jump width nominal
prescale_i	In[4:0]	Prescaling factor nominal
Synchronization Data		
tseg1_dat_i	In[5:0]	Time segment 1 data
tseg2_dat _i	In[4:0]	Time segment 2 data
sjw_dat _i	In[3:0]	Sample jump width data
prescale_dat _i	In[4:0]	Prescaling factor data
Delay Compensation		
delay_comp_en_i	In	Delay compensation enable
delay_comp_off_i	In[6:0]	Delay compensation offset
delay_comp_pos_o	Out[7:0]	Delay compensation position
delay_comp_en_vld_o	Out	Delay compensation enable valid (internal proofe)
RX Buffer IF		
ram_rx_ena_o	Out	RX Buffer enable
ram_rx_wr_o	Out	RX Buffer write enable
ram_rx_addr_o	Out[9:0]	RX Buffer address
ram_rx_data_i	In[31:0]	RX Buffer data input
ram_rx_data_o	Out[31:0]	RX Buffer data output
ram_rx_rdy_i	In	RX Buffer ready input
TX Buffer IF		
ram_tx_ena_o	Out	TX Buffer enable
ram_tx_wr_o	Out	TX Buffer write enable
ram_tx_addr_o	Out[9:0]	TX Buffer address
ram_tx_data_i	In[31:0]	TX Buffer data input
ram_tx_data_o	Out[31:0]	TX Buffer data output
ram_tx_rdy_i	In	TX Buffer ready input
Control Interface		
rx_receive_done_o	Out	RX received message (without any error)
rx_pending_o	Out	RX pending (during receiving a frame)
tx_start_i	In	TX start request to send a frame
tx_stop_i	In	TX stop request
tx_pending_o	Out	TX pending (during sending a frame)
tx_retransmit_max_i	In[2:0]	TX retransitions (7 -> infinite)
CAN Interface		
rx	In	CAN RX (of serial interface)
tx	Out	CAN TX (of serial interface)
Error Counters and States		
tec_o	Out[8:0]	TX error counter
tec_warning_o	Out	TX warning flag
rec_o	Out[7:0]	RX error counter
rec_warning_o	Out	RX warning flag
bus_off_o	Out	CAN bus off
error_passive_o	Out	CAN bus passive
events_o	Out[6:0]	Error/warning event flags
Testmode		
testmode_n	In	Testmode (loopback, internally tx->rx)

Core interface double buffers for transmit and receive

The CAN FD IP interface with double buffers is defined in following table. There is already the double buffer mechanism implemented. The RAM Blocks are single port RAMs, this means that during the

transfer of a frame from the first buffer, only the second buffer can be written and read from the outside.

Signal name	Type	Description
clk	In	System clock
reset_n	In	Reset input
Synchronization Nominal		
tseg1_i	In[5:0]	Time segment 1 nominal
tseg2_i	In[4:0]	Time segment 2 nominal
sjw_i	In[3:0]	Sample jump width nominal
prescale_i	In[4:0]	Prescaling factor nominal
Synchronization Data		
tseg1_dat_i	In[5:0]	Time segment 1 data
tseg2_dat _i	In[4:0]	Time segment 2 data
sjw_dat _i	In[3:0]	Sample jump width data
prescale_dat _i	In[4:0]	Prescaling factor data
Delay Compensation		
delay_comp_en_i	In	Delay compensation enable
delay_comp_off_i	In[6:0]	Delay compensation offset
delay_comp_pos_o	Out[7:0]	Delay compensation position
delay_comp_en_vld_o	Out	Delay compensation enable valid (internal proofe)
RX Buffer IF		
ram_rx_ena_i	In	RX Buffer enable
ram_rx_wr_i	In	RX Buffer write enable
ram_rx_addr_i	In[9:0]	RX Buffer address
ram_rx_data_i	In[31:0]	RX Buffer data input
ram_rx_data_o	Out[31:0]	RX Buffer data output
ram_rx_rdy_o	Out	RX Buffer ready input
TX Buffer IF		
ram_tx_ena_i	In	TX Buffer enable
ram_tx_wr_i	In	TX Buffer write enable
ram_tx_addr_i	In[9:0]	TX Buffer address
ram_tx_data_i	In[31:0]	TX Buffer data input
ram_tx_data_o	Out[31:0]	TX Buffer data output
ram_tx_rdy_o	Out	TX Buffer ready input
Control Interface		
rx_receive_done_o	Out	RX received message (without any error)
rx_pending_o	Out	RX pending (during receiving a frame)
tx_start_i	In	TX start request to send a frame
tx_stop_i	In	TX stop request
tx_pending_o	Out	TX pending (during sending a frame)
tx_retransmit_max_i	In[2:0]	TX retransitions (7 -> infinite)
CAN Interface		
rx	In	CAN RX (of serial interface)
tx	Out	CAN TX (of serial interface)
Error Counters and States		
tec_o	Out[8:0]	TX error counter
tec_warning_o	Out	TX warning flag
rec_o	Out[7:0]	RX error counter
rec_warning_o	Out	RX warning flag
bus_off_o	Out	CAN bus off

error_passive_o	Out	CAN bus passive
events_o	Out[6:0]	Error/warning event flags
Testmode		
testmode_n	In	Testmode (loopback, internally tx->rx)

BUS / AMBA AXI interface for access via internal SoC bus

The CAN FD IP bus interface is defined in following table. Each configuration and control are reachable through the bus interface. In the next table is shown a proprietary bus interface, but there is also an AXI bus interface available.

Signal name	Type	Description
clk	In	System clock
reset_n	In	Reset input
BUS Interface		
bus_ena_i	In	RX Buffer enable
bus_wr_i	In	RX Buffer write enable
bus_addr_i	In[9:0]	RX Buffer address
bus_data_i	In[31:0]	RX Buffer data input
bus_data_o	Out[31:0]	RX Buffer data output
bus_rdy_o	Out	RX Buffer ready input
CAN Interface		
rx	In	CAN RX (of serial interface)
tx	Out	CAN TX (of serial interface)

The following register definition is implemented. The purple cells are the registers and the white cells are the fields of the register above.

Register name	Addr	Description
- Field name	Bits	Description
CAN_TX_SID	**0**	**CAN TX Standard ID Register**
- SID	[28:0]	Standard Identifier + Extended ID
CAN_TX_DLC	**1**	**CAN TX DLC Register**
- DLC	[3:0]	DLC (Data Length; 0 -> 0; 1 -> 1; ... 8 -> 8; 9 -> 12; 10 -> 16; 11 -> 20; 12 -> 24; 13 -> 32; 14 -> 48; 15 -> 64)
- FORMAT	[10:8]	Frame format (000b .. CBFF; 100b .. CEFF; 010b .. FBFF; 110b .. FEFF; XLFF not supported)
- FTYP	[11]	RTR ('1' .. RF; '0' .. DF; remote frame selector)
- BRS	[12]	Bit rate switch (only FBFF and FEFF)
- ESI	[13]	Error state indicator (only FBFF and FEFF)
CAN_TX_BUFFER	**[2:17]**	**Transmit Buffer**
- DATA	[31:0]	Payload data
CAN_RX_SID	**18**	**CAN RX Standard ID Register**
- SID	[28:0]	Standard Identifier + Extended ID
CAN_TX_DLC	**19**	**CAN TX DLC Register**
- DLC	[3:0]	DLC (Data Length; 0 -> 0; 1 -> 1; ... 8 -> 8; 9 -> 12; 10 -> 16; 11 -> 20; 12 -> 24; 13 -> 32; 14 -> 48; 15 -> 64)
- FORMAT	[10:8]	Frame format (000b .. CBFF; 100b .. CEFF; 010b .. FBFF; 110b .. FEFF; XLFF not supported)
- FTYP	[11]	RTR ('1' .. RF; '0' .. DF; remote frame selector)

- BRS	[12]	Bit rate switch (only FBFF and FEFF)
- ESI	[13]	Error state indicator (only FBFF and FEFF)
CAN_TX_BUFFER	**[20:35]**	**Transmit Buffer**
- DATA	[31:0]	Payload data
CAN_CONTROL	**36**	**CAN Control Register**
- TX_SEND_REQ	[0]	TX send request
- TX_ABORD	[1]	TX abord
- TX_BUSY	[2]	TX busy / pending
- TX_DELAY_COMPENSATION_EN	[3]	TX delay compensation (will do delay compensation only if CAN_CONFIG_DATA.BRP <= 1)
- RX_BUSY	[4]	RX busy
- RX_RECEIVED	[5]	RX received (should be cleared to receive next message! To clear, write '0'; if still pending '1' a seconded message already received)
- RX_RECEIVEDIE	[6]	RX received interrupt enable
- RX_RECEIVEDIF	[7]	RX received interrupt flag (clear by writing '0')
- RETRANSMIT_MAX	[10:8]	Max Number of Retransmission (0 .. no retransmission; 7 .. unlimited retransmission)
- RX_OVERRUN	[30]	RX received frame overrun (Buffer full), will be cleared together with RX_RECEIVED
- LOOPBACK	[31]	Enable loopback mode (RX = TX, without tranceiver)
CAN_CONFIG_NOMINAL	**37**	**Nominal Bit Time Configuration Register**
- SJW	[3:0]	Synchronization Jump Width bits
- TSEG1	[13:8]	Time Segment 1 bits (Propagation Segment + Phase Segment 1)
- TSEG2	[20:16]	Time Segment 2 bits (Phase Segment 2)
- BRP	[28:24]	Baud Rate Prescaler bits
CAN_CONFIG_DATA	**38**	**Data Bit Time Configuration Register**
- SJW	[3:0]	Synchronization Jump Width bits
- TSEG1	[13:8]	Time Segment 1 bits (Propagation Segment + Phase Segment 1)
- TSEG2	[20:16]	Time Segment 2 bits (Phase Segment 2)
- BRP	[28:24]	Baud Rate Prescaler bits
CAN_DELAY_COMP	**39**	**CAN Delay Compensation (for FD Frames)**
- OFFSET	[6:0]	Delay compensation offset (suggestion to use TSEG1 of data rate)
- POSITION	[15:8]	Delay compensation position (= measured delay + OFFSET)
CAN_REC	**40**	**CAN Transmit/Receive Error Count Register**
- REC	[7:0]	Receive Error Counter bits
- TEC	[16:8]	Transmit Error Counter bits
- EWARN	[17]	Transmitter or Receiver is in Error Warning State bit
- RXWARN	[18]	Receiver in Error Warning State bit (128>REC>95)
- TXWARN	[19]	Transmitter in Error Warning State bit (128>REC>95)
- RXBP	[20]	Receiver in Error Passive State bit (REC>127)
- TXBP	[21]	Transmitter in Error Passive State bit (TEC>127)
- TXBO	[22]	Transmitter in Bus Off State bit (TEC>255)
- EVENTS	[31:24]	Indicating internal conditions and events (Bit 0: Acknowledgement Error; Bit 1: Bit Error; Bit 2: Stuff Error; Bit 3: Form Error; Bit 4: CRC Error; Bit 5: Overload Condition; Bit 6: Lost Arbitration)
CAN_FILTER	**[41:x]**	**CAN Filter Register**

- SID	[28:0]	Standard Identifier + Extended ID
- FORMAT	[31:29]	Frame format
CAN_FILTER_MASK	**[x:y]**	**CAN Filter Mask Register**
- SID	[28:0]	Standard Identifier + Extended ID
- FORMAT	[31:29]	Frame format
CAN_FILTER_CTRL	**[y:z]**	**CAN Filter Control Register**
- ENABLE	[0]	Filter enable
- CANFLT_IE	[1]	Filter interrupt enable
- CANFLT_IF	[2]	Interrupt flag (clear by writing '0')

Testing

The testcases are defined to the ISO 11898-1 requirements.

Deliveries

- VHDL RTL code or netlist for each technology
- Testbench

Sales & Marketing Contact

TES Electronic Solutions GmbH
Nikolaus-Otto-Straße 25
70771 Leinfelden-Echterdingen
Germany
sales@tes-dst.com
www.tes-dst.com