CAN/CAN FD IP Overview

Synective

27 August 2015

Contents

1. Introduction

CAN (Controller Area Network) is a multicast multi-master serial bus commonly used in automotive and industrial applications. The CAN bus is a versatile fault-tolerant bus architecture. Our IP conforms to the CAN FD (Flexible Data-Rate) version 1.0 standard and supports standard CAN bus speeds between 1 kbit/s to 1Mbit/s and CAN FD data phase bit rates at 3 clock cycles per bit reaching 13.333... Mbit/s with a 40 MHz clock. The IP is available for Xilinx, Altera, and other FPGA vendors.

During the CAN-FD plug-fest arranged by CiA 2015-03-24, this IP was one of few that successfully communicated with the version of CAN-FD expected to become ISO 11898-1. The IP will be verified towards Bosch and ISO 16845-1 reference tests when the tests are available to fully comply with the standard.

The IP is targeted at high-end systems and allows high efficiency with FIFO buffers at transmitter and receiver end. The transmit packet acknowledge data stream with tag information allows efficient transmission buffer management. The dual-destination buffer DMA functionality is designed for minimal added latency and an interrupt frequency that scales down as needed at higher system loads.

Delivery of the IP is made as a bus interfaceable core bundled with demo software to allow for easy integration.The bus type varies between platforms.It is also possible to directly interface the BSP (Bit Stream Processor) block. This is the core module of the IP, stripped of buffers and many features, thus making it very small.

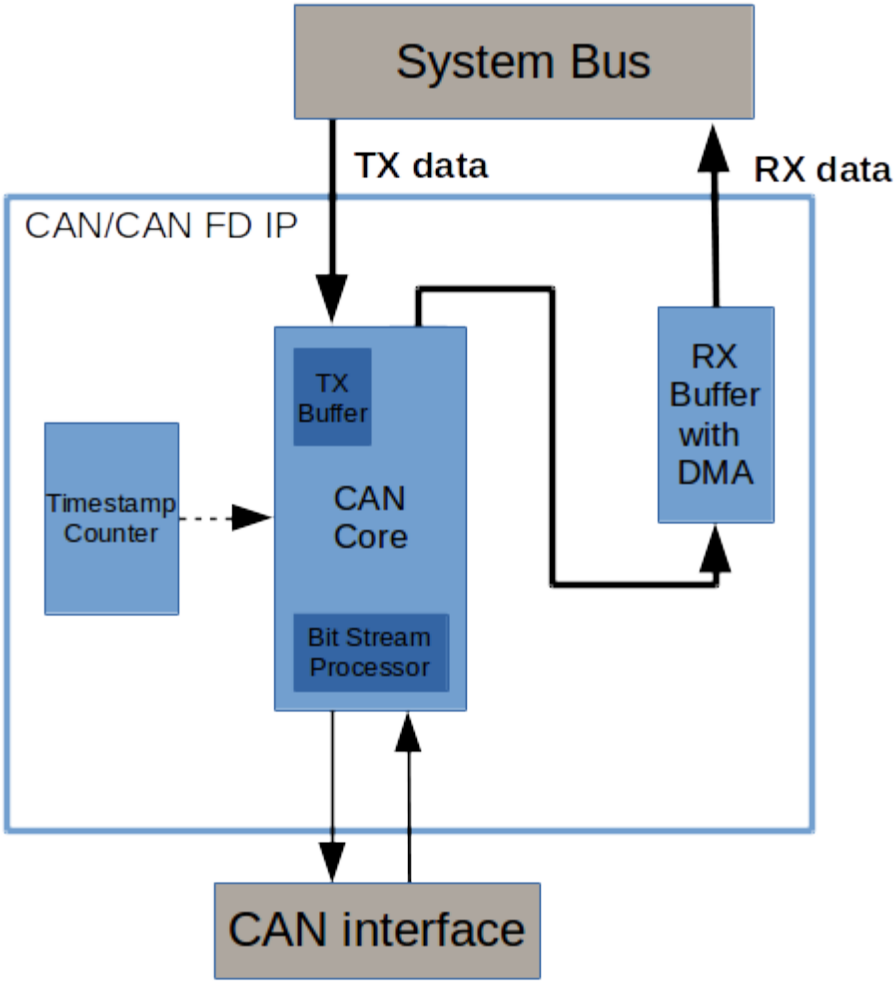


Figure 1: An example instantiation of the CAN/CAN FD IP. The timestamp unit is optional, and the transmit and receive buffer sizes parameterizeable. The BSP contains the core logic for bus communication. It is possible to interface this block directly without any wrapper logic.

2. Features

The following features are available for Synective's CAN controller:

Multiple Channels

Low-Latency DMA with Interrupt Rate Adaption

Small Size

Multiple CAN Formats

Option to Use Only Core Logic

Timestamps

Bus Load Calculation

Status Updates in Data Stream

Separate System Bus and Core Clocks

Configurable Hardware Buffer Size

Transmit Rate Limit

Listen Only-mode

Interrupt Logic

Available for Multiple Vendors

2.1 Multiple Channels

It is possible to configure a bus system to use multiple CAN cores. These use a shared RX buffer and timestamp counter. Because data only need to be read from one buffer, this is very well suited for a system where the data is to be processed in software

A tag is added to each packet written to the RX buffer, so that packets may be separated. Figure 2 displays a system with two CAN channels (cores).

2.2 Low-Latency DMA with Interrupt Rate Adaption

An optional low-latency DMA engine is available in the CAN controller. This engine uses two output buffers for improved performance.

Figure 3 describes the working principles of the DMA engine. By generating an interrupt and switching to a new buffer as soon as it's been emptied, the package latency is kept at a minimum. If the host cannot process the buffer quickly enough, more packets are instead put in the same buffer, thereby decreasing the load on the host at the cost of increased latency.

If DMA is not used, the same buffer is instead read as a slave.

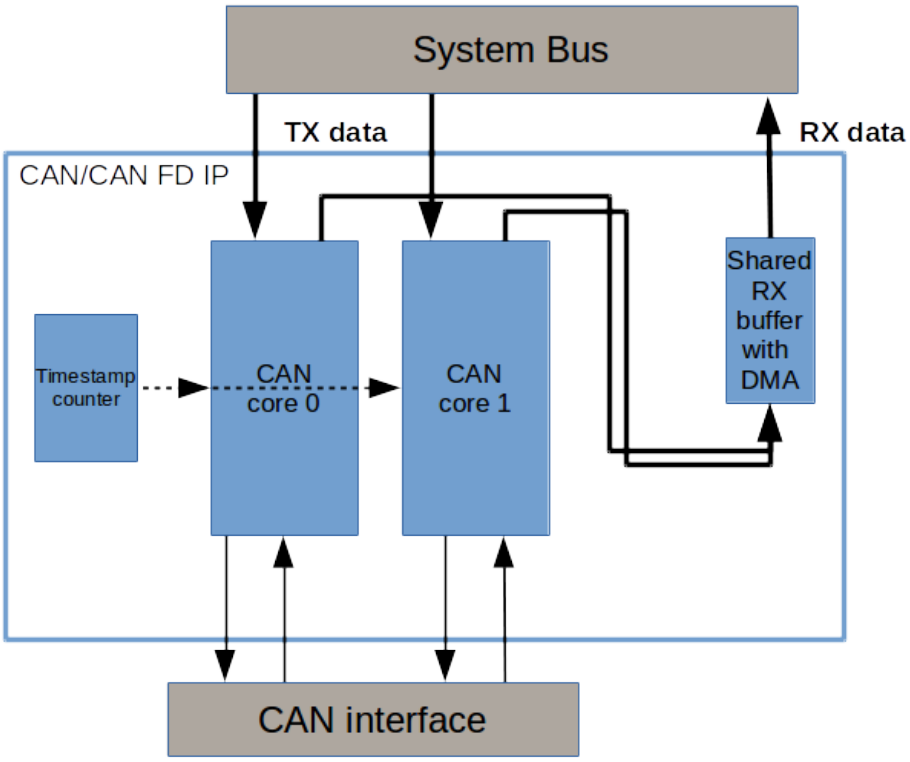


Figure 2: An instantiation of the CAN/CAN FD IP with two CAN cores (channels). These share the timestamp unit and receive buffer.

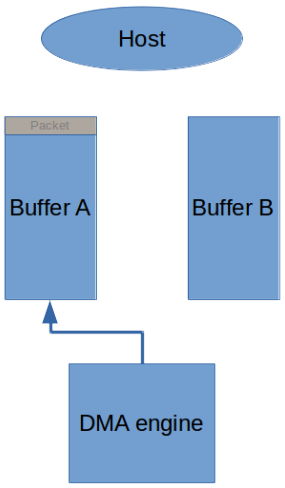
2.3 Small Size

Considerable effort has been put into optimizing the area requirements of the IP. Table 1 shows approximate resource usage resulting from different build properties.

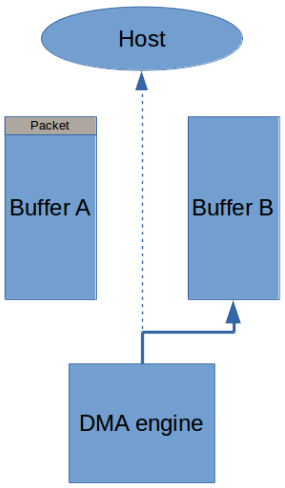
Setup	4-Input LUTs	Registers
CAN core, RX buffer	2,400	1,200
CAN core, RX buffer with DMA	2,500	1,300
CAN core, RX buffer with DMA, Timestamps	2,700	1,600
CAN core, RX buffer with DMA, Timestamps, Bus synchronization	2,900	1,800

Table 1: Approximate resource requirements for different setups of one CAN core and receive buffer.

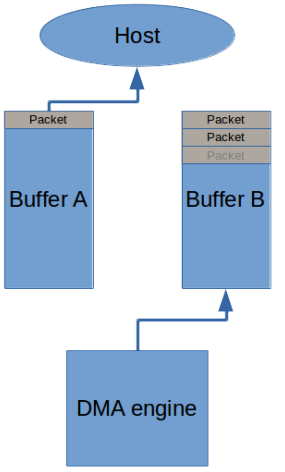
The possibility of sharing a receive buffer and timestamp counter will further reduce the size of the IP when multiple channels are to be used. The buffer size selected will have a minor impact on the amount of distributed logic used, but will have a direct relation to the amount of block-RAM used.



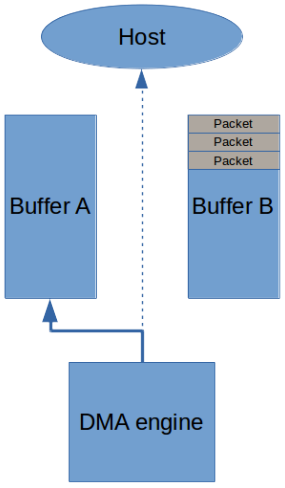
(a) The DMA engine writes the very first package to Buffer A.



(b) When the first package finishes, an interrupt is generated and the DMA engine switches buffer.



(c) The DMA engine will write any new packages to buffer B until the host has finished reading buffer A.



(d) When buffer A is empty, the DMA engine will switch buffer again as soon as it is ready with its current packet. A new interrupt is generated.

Figure 3: DMA engine buffer handling. If the host services the interrupts quickly, a short latency will be achieved. If it does not, the number of interrupts is reduced, resulting in lower load on the host.

2.4 Multiple CAN Formats

Synective's CAN controller supports both CAN 2.0A and 2.0B, as well as the non-ISO and ISO versions of CAN FD. A parameter is set during runtime to select either non-ISO or ISO CAN FD mode.

In case only standard CAN is needed, this can be specified as a build option in order to reduce the size of the IP.

2.5 Option to Use Only Core Logic

The IP comes with the option of only using the BSP (Bit Stream Processor) block. This is the core module of the IP, and handles the CAN bus communication. The BSP has no buffers and no special features, making it very small. Approximate resource usage is 1,100 4-input LUTs and 330 registers.

2.6 Timestamps

A timestamp can be included for each packet written to the RX buffer. The timestamp value may be configured to use 32 or 64 bits, or be omitted completely. The speed at which the value is incremented is configurable by specifying the number of system clock cycles to count before incrementing the value.

2.7 Bus Load Calculation

If enabled, the bus load is periodically calculated and reported.

2.8 Status Updates in Data Stream

If enabled, transmit acknowledges can be reported inside the data stream as special packets tagged with a channel ID from its core. This eases transmit buffer management. Core status, bus errors and bus load calculations may be reported in this way as well.

2.9 Separate System Bus and Core Clocks

The system bus and the CAN system may use different clocks. The type of system bus used depends on the platform. If the same clock is used, or if the clocks have a known relationship, bus synchronization logic may be excluded through a build parameter.

Please note: Although the CAN FD standard does not strictly specify which clock frequencies that may be used, there is a recommendation to always use a 20 MHz or a 40 MHz clock, or one that can be divided to one of these frequencies. This puts some limitation on which clocks may be used for the CAN core.

2.10 Configurable Hardware Buffer Size

The sizes of both the RX buffer and the TX buffers may be configured.

2.11 Transmission Rate Limit

If needed, the transmission rate of each CAN core may be limited. This can be used to guarantee that the unit does not clog the bus.

2.12 Listen Only-Mode

It is possible to put the core in a *Listen Only*-mode, in which it will be completely silent on the bus. This also means that no packets may be transmitted, and that no acknowledge or error bits are sent.

2.13 Interrupt Logic

A number of interrupt sources may be used to keep track of the system. It is possible to individually enable and disable these interrupt sources. The interrupt sources range from user and bus error detection to status updates.

2.14 Available for Multiple Vendors

The IP is available for Altera and Xilinx, as well as other FPGA vendors.

3. Build Parameters

By disabling features that are not to be used, the size may be reduced.

3.1 Usage

The build parameters are set during synthesis of the IP, and are all accessed from the top hierarchy module. Please note: some of these parameters will enable the use of a function, but the function may still have to be activated using a register access.

Because these parameters are used at compile time, they cannot be changed if the IP was delivered as a netlist.

3.2 Available Parameters

3.2.1 N_CHANNELS

Selects the number of channels of the IP. At minimum one channel must be included.

Default value: 2

3.2.2 USE_CAN_FD

Set to 1 to include CAN FD and 0 to remove.

Default value: 1

3.2.3 USE_DMA

Set to 1 to include the DMA engine and 0 to remove. If disabled, the data is instead read with the receive buffer appearing as a slave on the bus.

Default value: 0

3.2.4 DMA_BASE_ADDR

Sets the address of the first buffer the DMA will use. The second buffer is assumed to be allocated directly after the first, and both buffers are 4 kiB.

Default value: 0

3.2.5 ASYNC_CLOCK_REGIONS

Set to 1 to include synchronization of bus logic and 0 to remove. If the bus and CAN core clocks are the same, clock domain crossing logic may me removed.

Default value: 1

3.2.6 TX_BUFFER_MULT

Selects the TX buffer size of each channel. Set to maximum number of packets in buffer divided by 3 (e.g. set to 5 for 15 packets in buffer.)

Default value: 5

3.2.7 RX_BUFFER_MULT

Selects the RX buffer size. Set to maximum number of packets in buffer divided by 3 (e.g. set to 32 for 96 packets in buffer.)

Default value: 32

3.2.8 CAN_FREQUENCY

Informs the IP of the system frequency, given in Hz. This parameter is used internally to determine bit widths during synthesis. Thus, setting a larger value than needed may result in extra resource usage, but will not have a negative effect on the function.

Default value: 80000000

3.2.9 USE_TX_RATE_LIMIT

Set to 1 to include transmission rate limitation, and 0 to remove. When enabled, transmission rate limitation may be configured via the register bank. This can be used to set a limit on how much the controller will use the CAN bus.

Default value: 1

3.2.10 BUS_LOAD_CALC

Set to 1 to include CAN bus load calculation and 0 to remove. The function is controlled via registers.

Default value: 1

3.2.11 USE_CLASSIC_MODE

Set to 1 to make it possible to configure the IP to work in classic mode. In this mode, the IP assumes that all messages on the CAN bus are standard CAN (i.e. not CAN FD).

Default value: 1

3.2.12 EN_NONDATA_PACKETS

Set to 1 to enable non-data packets in the data stream. This includes status updates and acknowledges.

Default value: 0

3.2.13 TSEG1_N_RESET

Default value of nominal rate Timing Segment 1

Default value: 5

3.2.14 TSEG2_N_RESET

Default value of nominal rate Timing Segment 2

Default value: 2

3.2.15 SJW_N_RESET

Default value of nominal rate Sync Jump Width

Default value: 1

3.2.16 BRP_N_RESET

Default value of nominal rate Bit Rate Prescaler

Default value: 10

3.2.17 TSEG1_D_RESET

Default value of data phase rate Timing Segment 1

Default value: 2

3.2.18 TSEG2_D_RESET

Default value of data phase rate Timing Segment 2

Default value: 1

3.2.19 SJW_D_RESET

Default value of data phase rate Sync Jump Width

Default value: 1

3.2.20 BRP_D_RESET

Default value of data phase rate Bit Rate Prescaler

Default value: 1

3.2.21 CHANNEL_ID_FIRST

Sets the ID of the first CAN channel in the IP. Following channels (if any) receive ID:s that are increments of the first. The channel ID:s are used to differentiate between packets coming from each channel.

Default value: 0

3.2.22 TIMESTAMP_SIZE

The timestamp may be built to use 32 or 64 bits, or may not be included at all. This is controlled directly by this build parameter.

Default value: 64

3.2.23 USE_INTERNAL_TIME_EVENT

Set to 1 to enable the timestamp counter to increment after a configurable number of system clock cycles. Set to 0 to remove this function.

Default value: 1

3.2.24 INITIAL_INTERVAL

Sets the initial number of system clock cycles between timestamp updates.

Default value: 400000

3.2.25 USE_EXTERNAL_TIME_EVENT

Set to 1 to enable the timestamp counter to increment at the rising edge of an external signal. Set to 0 to remove this function.

Default value: 1