Transitioning applications from CAN 2.0 to CAN FD

Orlando Esparza, Wilhelm Leichtfried, Fernando González, Microchip Technology Inc.

The CAN bus protocol is used in a wide variety of applications, including industrial, automotive, and medical. Approximately 1.5B CAN nodes are used each year. Designers of these applications benefit from the many advantages CAN offers, such as reliability, cost effectiveness, engineering expertise and the availability of tools and components. CAN FD builds on the existing benefits of CAN 2.0 technology, allowing designers to leverage CAN 2.0 expertise, tools, hardware and software while also taking advantage of CAN FD's increased data rate and data field length.
This paper will explore some of the considerations associated with CAN system design and how designers can transition their applications from CAN 2.0 to CAN FD. These considerations relate to physical layer, controller and overall system topics. Application designers must begin with hardware that conforms to both physical layer and controller requirements. Solutions for CAN FD controllers will be discussed, highlighting external CAN FD controllers as an alternative to integrated CAN FD controllers. These external controllers allow designers more flexibility when choosing an MCU that best fits the application and can reduce the migration effort from CAN 2.0 to CAN FD.

Introduction

Automotive manufacturers and suppliers are facing some challenges with today's CAN 2.0 networks. First, automotive manufacturers and suppliers are dealing with an increase in end-of-line (EOL) programming costs. This is due to an increase in Electronic Control Unit (ECU) memory requirements. Second, the use of automotive electronics continues to expand, requiring more ECUs to support the demands of these new electronic applications. This either decreases the available bandwidth on existing CAN 2.0 bus networks or it forces designers to introduce a new CAN 2.0 network into the system architecture. Last, as demand for cyber security continues to grow, ECUs will require more memory and bus utilization will increase drastically. CAN FD addresses some of these challenges by offering two significant enhancements over CAN 2.0. CAN FD increases the data rate capabilities in normal mode from 500 kb/s (typical) to 2 Mb/s, and in programming mode up to 5 Mb/s. In addition, it increases the data field from 8 to 64 data bytes. While these benefits can offer the designer faster EOL programming and free up network bandwidth, there are some development challenges associated with supporting new CAN FD applications. The following discusses and outlines some of the major design changes required and other considerations for designers who are transitioning their applications from CAN 2.0 to CAN FD.

Changes to CAN network architecture

Automotive system architectures utilize many different network technologies to support a wide range of safety, body and convenience, infotainment, and ADAS electronics within the automobile. Starting with the system gateway, CAN plays a major role in supporting many of these applications in today's architectures.
CAN FD will continue to play a major role within future architectures. The key factor to supporting these architectures is enabling faster throughput at the gateway and branching it out into the sub-networks. Current CAN 2.0 gateways achieve ~37 s/MB transfer time based on a 500 kb/s (typical) data rate and an 8 byte data payload. Future CAN FD gateways are targeted to achieve ~1.9 s/MB based on a 5 Mb/s data rate and a 64 byte data payload.
Today's system architectures support up to five or more CAN 2.0 networks. CAN 2.0 networks typically run at 500 kb/s and not 1 Mb/s.

The bandwidth on a CAN bus is limited by the propagation delay and by the bus topology. Future CAN FD architectures will utilize two types of networks: dedicated CAN FD networks and mixed CAN 2.0 and CAN FD networks.
In a dedicated CAN FD network, all CAN nodes on the network will be CAN FD capable. The advantage of this configuration is that the CAN FD protocol can always be used, and there will be minimal effect on the physical layer transceivers (i.e. no need for Partial Networking-like transceivers). The disadvantage of this approach is that the entire network will have to support CAN FD, making the change to CAN FD very significant and costly.
Some automotive manufacturers mix CAN 2.0 and CAN FD nodes in the same network. This is possible because CAN FD controllers support both CAN 2.0 and CAN FD protocols. One advantage of this configuration is that networks can be migrated to CAN FD node by node without requiring an entire network change. The disadvantage of this method is that physical layer transceivers will have to support a CAN FD filtering method on CAN 2.0 nodes to ensure they don't create any error frames during CAN FD communication. This adds cost and complexity to the system.

CAN FD and ISO standard

The CAN protocol is specified by the ISO 11898 standard. The ISO 11898-1 specifies the Data Link Layer. In 2014, an initiative to include the CAN FD requirements in this specification began. This year, the International Standards Organization has approved the ISO 11898-1 as a Draft International Standard (DIS) without any votes against it. The final ISO 11898-1 standard is expected to be published in April, 2016.
The ISO 11898-2 originally specified the requirements of the CAN 2.0 Physical Layer up to 1 Mb/s. The ISO 11898-5 is an extension of the ISO 11898-2 accommodating new low-power requirements during CAN 2.0 bus idle conditions. The ISO 11898-6 is an extension of the ISO 11898-2 and ISO 11898-5 specifying the Selective Wake-up (Partial Networking) functionality.
In 2014, an initiative to add CAN FD to the ISO 11898-2 and to combine it with ISO 11898-5, and ISO 11898-6 was also started. This year, the ISO 11898-2 successfully passed the Committee Draft Ballot. The Draft International Standard (DIS) version is currently under development, and submission is expected soon. The final ISO 11898-2 standard is expected to be published in July, 2017.

OSI reference model – ISO 11898-1/2

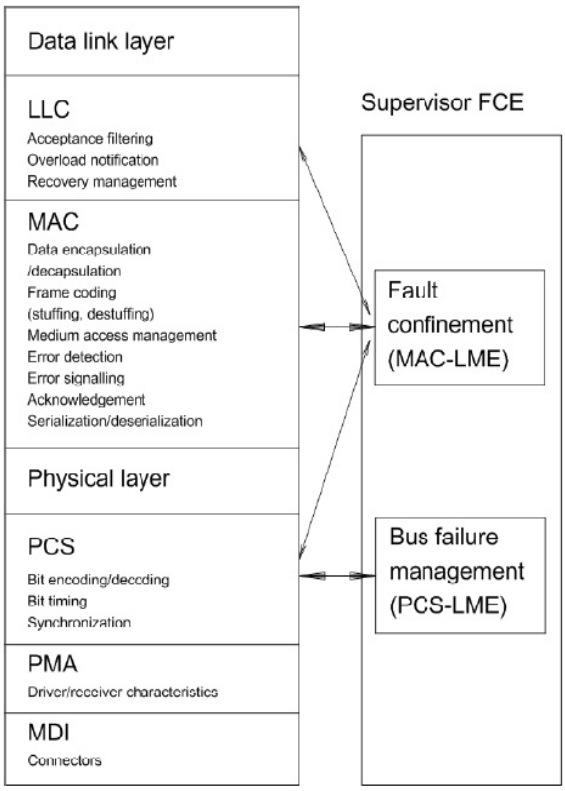

Figure 1: OSI reference model [1]

The Layered Architecture according to the OSI Reference Model specified in the ISO 11898-1 is the same for both CAN 2.0 and CAN FD (shown in Figure 1). The differences within the OSI Reference Model between CAN 2.0 and CAN FD are in the Logical Link Control (LLC) and the Medium Access Control (MAC) Sublayers of the Data Link Layer, and the Physical Coding Sublayer (PCS) and the Physical Medium Attachment (PMA) of the Physical Layer. The Medium Dependent Interface (MDI) of the Physical Layer is the same for CAN 2.0 and CAN FD. Table 1 illustrates the difference in requirements between CAN 2.0 and CAN FD.

Table 1: Differences between CAN 2.0 and CAN FD requirements

Requirement	CAN 2.0	CAN FD	Source
Data Rate	1Mb/s	2Mb/s (normal operation) 5Mb/s (point to point)	[2]
TXD to RXD Loop Prop Delay Symmetry	N/A	2Mb/s -20%/+10% 5Mb/s -40%/+10%	[2]
Bytes in Data Field	8	64	[1]

Data link layer (DLL) – ISO 11898-1

One of the primary differences between CAN 2.0 and CAN FD is in the MAC of the DLL, where the payload can be increased from 8 data bytes up to 64 data bytes in the data field of the CAN FD (see Figure 2). This increase in payload makes the CAN FD communication more efficient by reducing the protocol overhead. Messages that had to be split due to the 8 byte payload limit can be combined into one message. Additionally, security can be enhanced via the encryption of CAN FD messages as a result of the higher data rate and increased payload.

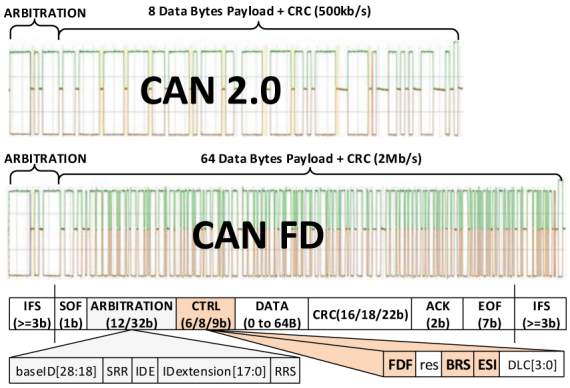

Figure 2: Message frame differences: CAN 2.0 vs. CAN FD

CAN FD switches the data rate during the data and CRC field. The Control field of the CAN FD frame contains three new bits. The FDF bit is used to distinguish between CAN 2.0 and CAN FD frames. Bit rate switching is initiating by setting the BRS bit. The error state of the transmitter is indicated by the ESI bit.

Physical layer – ISO 11898-2

The other main difference between CAN 2.0 and CAN FD is in the PCS of the Physical Layer, where the CAN 2.0 data rate was increased from typically 500 kb/s to 2 Mb/s for nominal vehicle operating conditions and up to 5 Mb/s for diagnostics or EOL programming.

Figure 3 illustrates the main circuit blocks of a CAN FD transceiver. The CAN FD transceiver interfaces with the CAN FD controller via the TXD and RXD digital signals. When in Normal mode (STBY low), the bit stream from the CAN FD controller on TXD gets encoded to differential output voltages on the physical CAN bus signals (CAN_H and CAN_L). The RXD output pin of the CAN FD transceiver reflects the differential voltages on the CAN bus.

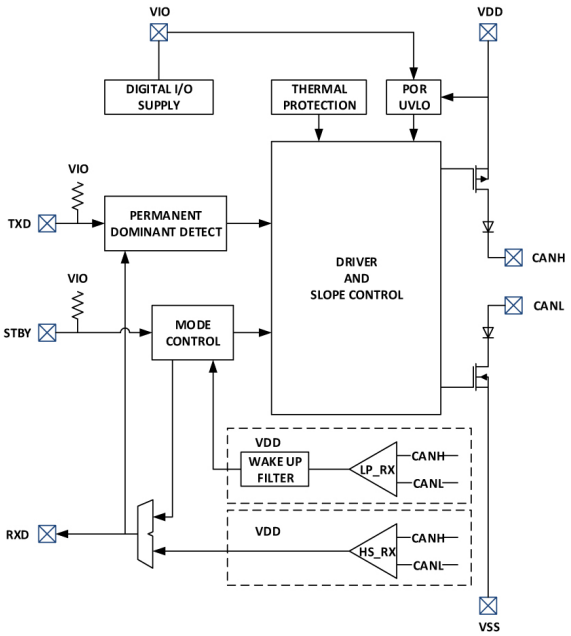

Figure 3: Block diagram of a typical CAN FD transceiver.

The TXD to RXD propagation delay of a CAN FD transceiver must not exceed 255ns for both dominant and recessive transitions. Because the CAN FD transceiver is not a push-pull driver, there is some asymmetry between recessive and dominant TXD to RXD propagation delay. As a result, the recessive bit time on RXD tends to shorten. Figure 4 describes how the loop delay symmetry parameters are measured.

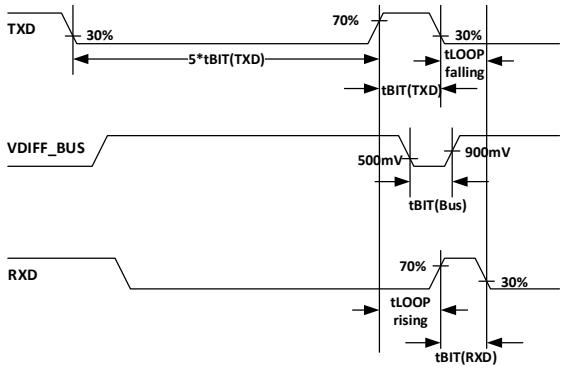

Figure 4: Measurement of loop delay symmetry

Mixed networks and physical layer

CAN FD transceivers are backwards compatible with CAN 2.0 transceivers. The Data Link Layer of CAN FD is not compatible with CAN 2.0. To implement mixed operation of CAN 2.0 and CAN FD nodes on the same bus, the CAN 2.0 nodes need to be ideal passive (invisible to the network) during CAN FD communication or error frames will be generated. At least three options are available to make CAN 2.0 nodes tolerant to CAN FD: Partial Networking (PN), CAN FD Shield, and CAN FD Filter. Currently, only PN transceivers are available on the market. PN allows the CAN 2.0 controller to be disconnected from the bus during CAN FD communication. The PN transceiver will ignore all CAN FD messages by decoding the incoming CAN frames. The PN transceiver waits for a valid CAN 2.0 wake-up message with a specific ID before it restarts routing CAN 2.0 messages to the CAN 2.0 controller.

CAN FD controller

Figure 5 illustrates the main blocks of a CAN FD controller. The CAN FD controller interfaces to the CAN FD transceiver using digital transmit and receive pins. The Bit Stream Processor (BSP) implements the CAN FD protocol. It transmits and receives CAN FD frames. The Transmit Handler prioritizes messages that are queued for transmission. The Receive Handler manages received messages. Only those messages that match the Acceptance Filters are stored in RX message objects or FIFOs. The Memory Interface controls the access to the RAM. Message Objects are stored in RAM. The message RAM can be located in the system RAM of a microcontroller; it doesn't have to be dedicated to the CAN FD controller. Optionally, the acceptance filter configuration can be stored in RAM. The microcontroller uses a Register Interface to access the Special Function Registers (SFR) of the CAN FD controller. The SFR are used to configure and control the CAN FD controller. Interrupts notify the microcontroller about successfully transmitted or received messages. Received messages are time stamped. Transmitted messages are optionally time stamped and their IDs can be stored in a Transmit Event FIFO (TEF).

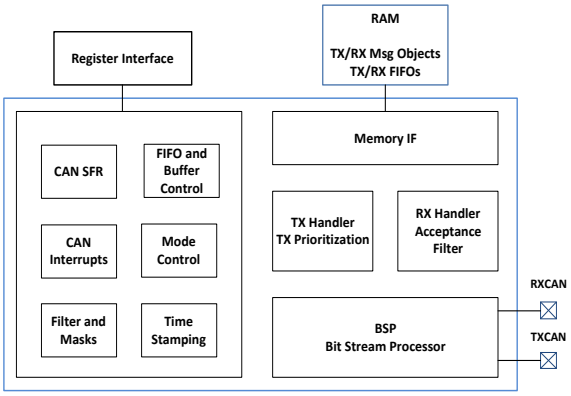

Figure 5: CAN FD controller block diagram

CAN FD controller requirements

Table 2 illustrates the difference in requirements between a CAN 2.0 and a CAN FD controller. The following section will discuss the major changes in more detail.

Higher bandwidth

During the arbitration phase, the data rate is limited by the CAN network propagation delay. In the data phase, only one transmitter remains, therefore, the bandwidth can be increased by switching the bit rate during the data phase. The transmitter always compares the intended transmitted bits with the actual bits on the bus. The propagation delay in the data phase can be longer than the bit time. Therefore, the bits are sampled at a Secondary Sample Point (SSP). Data Bit Time and SSP configuration require additional configuration registers.

In normal operation, real-world CAN 2.0 networks achieve a bandwidth of 17 bytes/ms when using a bit rate of 500 kb/s, 8 bytes of payload, and 50% bus utilization. During End Of Line (EOL) programming, 100% of the bus can be utilized, resulting in a bandwidth of 29 bytes/ms.

CAN FD improves the bandwidth up to a factor of four during normal operation. This can be achieved by increasing the data bit rate to 2 Mb/s and by increasing the payload to 32 bytes. Increasing the payload to 64 bytes and switching the data bit rate to 5 Mb/s results in ten times the bandwidth gain during EOL programming.

Increased bandwidth requires the CAN FD controller and the microcontroller to process the messages faster. This requires higher microcontroller and CAN FD controller clock speeds and FIFOs to buffer messages.

Bit time configuration and clock

CAN 2.0 controllers often use an 8 MHz clock, while CAN FD controllers require a faster clock. The selection of the sample point within a CAN FD network is critical. It is recommended that all CAN FD nodes use the same sample point setting; clock frequencies of 20, 40, or 80 MHz are recommended. This allows shorter time quanta and, therefore, higher resolution for setting the sample point. Correctly switching the bit time is a technical challenge. Using the same time quanta resolution during Nominal and Data bit phase is also recommended. This requires more time quanta per bit during the Nominal bit phase as compared to CAN 2.0.

Table 2: Differences between CAN 2.0 and CAN FD controller requirements

Requirement	CAN 2.0	CAN FD	Source
Conformance	CAN 2.0	CAN FD	[1]
Clock (MHz)	8	20, 40, 80	Bosch/CiA recommendation
Nominal bit time quanta	25	385 typical 81/181 min.	- [1]
Data bit time quanta	N/A	49 typical 25 min.	- [1]
SSP	N/A	Yes	[1]
Nominal BR (Mb/s)	0.125 to 1	0.25 to 1	OEM recommendation
Data BR (Mb/s)	N/A	1 to 8	OEM recommendation
Band width (Bytes/ms)	17 (500k, 8 bytes)	70 (500k/2M, 32 bytes)	OEM use case
Band width (Bytes/ms)	29 (500k, 8 bytes)	170 (500k/2M, 64 bytes)	OEM use case
Band width (Bytes/ms)	29 (500k, 8 bytes)	340 (500k/5M, 64 bytes)	OEM use case
Max payload bytes	8	64	[1]
AUTOSAR	4.0.3	4.2.2	[3]
Message FIFOs	N/A	TX, RX	Microchip Technology Inc. recommendation
TEF	N/A	Yes	Microchip Technology Inc. recommendation
Time stamping (bits)	16	16 to 32	OEM recommendation
Minimum RAM	640B	2432B	Microchip Technology Inc. recommendation
ISO CRC	N/A	Yes	[1]
Protocol Exception Disable	N/A	Yes	OEM recommendation
TX attempts configurable	Optional	Yes	[1]
Object Payload configuration	N/A	Yes	Microchip Technology Inc. recommendation
PDU Mapping	Static	Static, Dynamic Multi	OEM requirement

Bigger RAM

Increasing the payload of CAN FD messages requires more RAM for message storage. Storing 32 message objects with ID and a payload of 8 bytes requires 640 bytes of RAM. Increasing the payload to 64 bytes requires 2432 bytes of RAM.

Static PDU frames

In a CAN network, signals are mapped into meaningful Protocol Data Units (PDU), as

shown in Figure 6. Usually one PDU is mapped into one CAN frame. The signals inside a PDU and the length of a PDU don't change; they are static. The frames and signals are described in a message database.

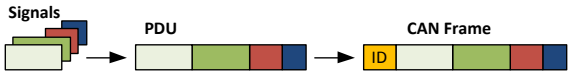

Figure 6: Mapping of signals into PDUs and into CAN frames

Only one ECU can transmit a certain PDU (see Figure 7). Multiple ECUs can receive a PDU. Acceptance filtering is used to accept PDUs that are of interest to the ECU. Acceptance filtering is done in hardware and reduces the required message processing in the microcontroller. Filters can be set up to filter on the ID of the CAN frame and optionally on the first two data bytes of the CAN frame. Filters can point to different message objects inside the CAN controller. The concept of static PDU frames can also be applied to CAN FD. In order to make CAN FD most efficient, a payload of 64 bytes should be used as much as possible, but the signal mapping gets even more complex for PDUs with 64 bytes.

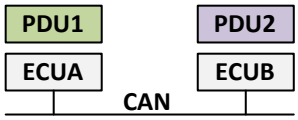

Figure 7: Static PDU frames on network

Dynamic multi-PDU frames

Dynamic Multi-PDU Frames (M-PDU) try to maximize the efficiency of CAN FD by dynamically combining multiple PDUs into one frame (see Figure 8). PDUs are only transmitted when the data changes.

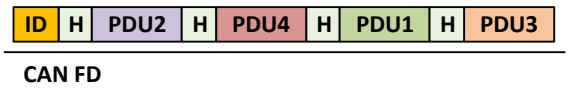

Figure 8: Dynamic M-PDU frame

The message database of static PDUs can be re-used. Each PDU contains a header to distinguish between the different PDUs inside a frame. The header consists of the PDU ID and the byte length. M-PDUs are especially useful for CAN FD gateways. The gateway can collect multiple PDUs and send them out in one frame (see Figure 9). The system designer defines the rules for combining PDUs and for delaying an M-PDU before it gets transmitted.

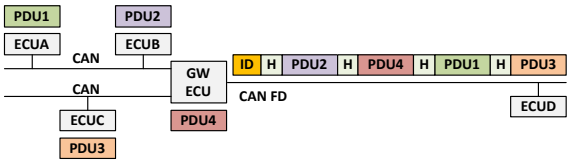

Figure 9: CAN FD gateway and dynamic M-PDU frames

Since multiple PDUs are combined dynamically into one CAN FD frame, the classic message filtering concept can't be used. All frames must be received and the PDUs have to be filtered by software, not by hardware. This will increase the demand for faster message processing. FIFOs with payloads up to 64 bytes will be required to buffer received messages and give the microcontroller time to process the messages. This will require even more RAM to store the received data.

Availability of CAN FD microcontrollers

A major challenge of CAN FD adoption and transition is the limited number of CAN FD controllers available today and in the near future. Some of the causes associated with this challenge are the following:

- Updating the ISO 11898-1 specification is a long process
- The ISO CRC fix delayed silicon component availability
- Silicon suppliers are limiting the number of developments due to risk of ISO spec changing
- Initial MCUs targeted for release are supporting high-end CAN FD applications

The causes described above result in the following challenges:

- Limited number of MCUs with CAN FD available on the market
- Lack of available components puts pressure on silicon supplier, tier supplier and OEM to meet aggressive timelines
- First MCUs with CAN FD released will be high performance and feature rich, leaving an MCU gap in many mid- to lower-end CAN FD applications

Replacing or updating the MCU is a major system change and will require the automotive supplier and manufacturer to redesign, revalidate and requalify the ECU. This is a significant amount of time, resources and investment. In many cases, a replacement MCU with CAN FD may not be available. As a result, customers will benefit from using an external CAN FD controller. This allows ECU designers to enable CAN FD by adding only one external component while they continue to utilize the majority of their design.

External CAN FD controller

External controllers that support CAN 2.0 applications are currently available, including the SJA1000 from NXP and the MCP2515 from Microchip. Both devices serve very well as external CAN 2.0 controllers and are widely used in automotive and industrial applications. These types of devices are typically used to add CAN 2.0 capability to an MCU or to add an additional CAN 2.0 controller to an existing MCU.

Many of the same considerations for using a CAN 2.0 controller apply to using an external CAN FD controller. It interfaces directly to the physical layer transceiver transmit and receive pins. The controller acts as a CAN FD engine, processing the CAN FD messages and relaying any relevant messages to the MCU. An external controller can interface to the MCU through a serial or parallel port. Figure 10 shows a typical application of an external CAN FD controller using an SPI interface.

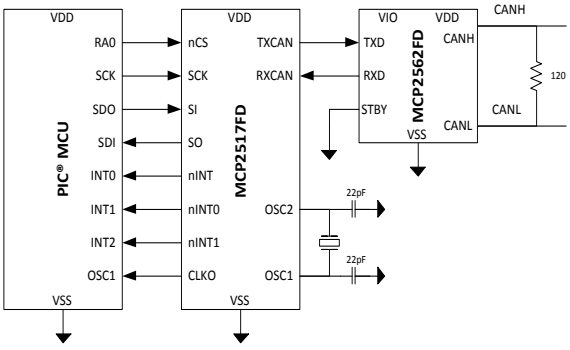

Figure 10: Typical CAN FD application using external CAN FD controller

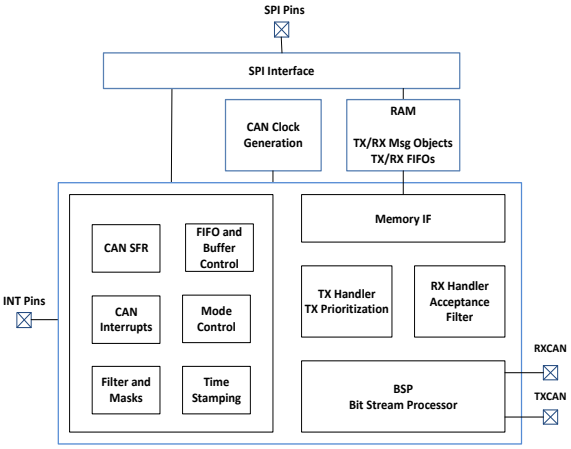

Figure 11: External CAN FD controller block diagram

Figure 11 illustrates the block diagram of an external CAN FD controller with an SPI interface. Using an SPI interface decreases the number of pins required as compared to a parallel interface. The CAN FD controller contains the same blocks as the integrated CAN FD controller. When integrated into an MCU, the CAN FD peripheral can share the system RAM. The RAM inside an external CAN FD controller is dedicated. The SPI interface accesses the SFR to control the CAN FD engine, to configure the CAN FD bit times and to set up the receive filters. The SPI interface also accesses the RAM to load transmit messages or to read received messages. The external CAN FD controller transmits and receives messages autonomously and interrupts the MCU only when a message is successfully transmitted or received. The SFRs are efficiently arranged to reduce the number of SPI transfers and to keep up with the higher CAN FD bandwidth. This allows the MCU to use a DMA to access larger SFR and RAM blocks via SPI. The external CAN FD controller integrates a clock generator to supply the CAN FD clock. Optionally, the clock can be provided to the MCU.

An external CAN FD controller has to meet the same requirements as an integrated CAN FD controller. The bandwidth requirements can be met by using an SPI with DMA and by increasing the SPI frequency. Calculations show that an external CAN FD controller utilizing an SPI frequency of 10 to 16 MHz can keep up with a 100% loaded CAN FD bus at data bit rates up to 8 Mb/s.

Summary

Applications transitioning from CAN 2.0 to CAN FD benefit from many enhancements, such as an increase in data rate and payload. However, application changes are required to take advantage of these enhancements. The transition affects all levels of the development process, from the tool supplier up to the end automotive manufacturer. Designers have already started to transition CAN FD into their automotive and industrial applications.
Automotive manufacturers in the US, Europe and Asia are planning to implement CAN FD as early as model year 2018 with a wider adoption expected in 2020. Automotive suppliers have started development to prepare for upcoming CAN FD programs and are facing aggressive timelines and design challenges. In many cases, cost-effective MCUs with CAN FD that are well-suited for their applications may not be available. For these situations, using an external CAN FD controller can be a viable alternative. Using an external CAN FD controller will help to minimize development timelines and be more cost effective than using a high-end MCU with CAN FD.

Orlando Esparza
Microchip Technology Inc.
2355 W. Chandler Blvd.
US-85244 Chandler, AZ

Tel. +1- 480-792-7363
orlando.esparza@microchip.com
www.microchip.com

Wilhelm Leichtfried
Microchip Technology Inc.
2355 W. Chandler Blvd.
US-85244 Chandler, AZ

Tel. +1- 480-792-7572
wilhelm.leichtfried@microchip.com
www.microchip.com

Fernando González
Microchip Technology Inc.
2355 W. Chandler Blvd.
US-85244 Chandler, AZ

Tel. +1- 480-792-4578
fernando.gonzalez@microchip.com
www.microchip.com

References

[1] ISO/DIS 11898-1:2015(E), Road vehicles - Controller area network (CAN)
Part 1: Data link layer and physical signaling
[2] ISO/CD 11898-2:2015-06-08, Road vehicles - Controller area network (CAN)
Part 2: High-speed medium access unit
[3] AUTOSAR 4.x Specification