Advantages of CAN FD Error detection mechanisms compared to Classical CAN

Dr. Arthur Mutter, Robert Bosch GmbH
Florian Hartwich, Robert Bosch GmbH

CAN FD offers, at similar costs, a net data rate that is several times higher than in Classical CAN [1], [2]. Beside this, CAN FD also provides improved error detection mechanisms.

Error detection is a crucial functionality provided by communication protocols. A receiving node has to be able to judge with a high probability whether a frame was received with or without bit errors.

This paper introduces the four main improvements in CAN FD regarding error detection: *(1)* use of CRC polynomials of higher order, *(2)* inclusion of dynamic stuff bits into CRC calculation, *(3)* inclusion of the number of inserted dynamic stuff bits into the frame, and *(4)* the use of fixed stuff bits in the CRC field of the frame. Three of these improvements were already present in the original Bosch proposal for CAN FD. Improvement *(3)* was introduced during the ISO standardization process, which provides an opportunity for participating experts to propose and review improvements or additional requirements.

The experts of the ISO standardization project team assumed two fault types: fault type A (bit flip) and fault type B (bit drop and bit insertion). This paper describes these fault types. It also shows, that in a practical setup, even a single bit drop is hard to achieve and a single bit insertion is nearly impossible to achieve.

This paper also classifies the potential error cases at a receiving node into three classes to analyze the interaction of fault types and error detection mechanisms.

1. Error Detection Capabilities of CAN

In CAN communication, all nodes in a network check the validity of each frame, including the transmitter of the current frame. The checks are based on a combination of several protocol mechanisms for error detection. They are described in the following.

1.1 Bit Monitoring

Each CAN node sends either a recessive or a dominant bit value, so the CAN physical layer may be in one of the two states, recessive or dominant. It is in dominant state if at least one node sends the dominant bit value. It is in recessive state if all nodes send the recessive bit value. All nodes monitor each bit on the CAN bus and compare it with the bit value they sent. A node detects a bit error when it samples a recessive bit while it sends a dominant bit. The inverse case, a node samples a dominant bit while it sends a recessive bit, is generally not regarded as a bit error, except for the transmitter of a frame outside arbitration and acknowledge. In the data phase of a CAN FD frame, the transmitter has the option to delay the monitoring of transmitted bits to a secondary sample point after the end of the actual bit, to compensate for the signal delay time that may be longer than a data phase bit time.

1.2 Frame Format Check

Most parts of a CAN frame (identifier, control, or data bits) are variable or are calculated from the variable bits (CRC sequence), but some bits (delimiters, end of frame) have a fixed format (see Figure 1). A receiver detects a form error when it samples a fixed-format bit with the wrong value. A special case is the reserved bit following the FDF bit in CAN FD frames. The reserved bit is expected to be dominant. In current applications, a form error is detected when this bit is sampled as recessive. For future applications, this bit may be used to distinguish between the CAN FD frame format and another – not yet defined – new frame format. When this alternative is selected (by software configuration) and if then this bit is sampled as recessive, the receiver enters a protocol exception state until the bus is idle again. This allows the introduction of future new frame formats that are tolerated by existing CAN FD implementations.

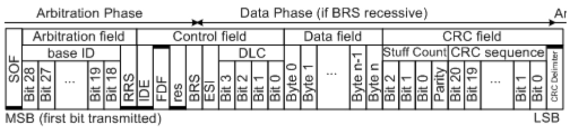

Figure 1: FD Base Frame Format, 20 to 64 data bytes [2]

1.3 Cyclic Redundancy Check

The transmitter of a frame calculates the CRC sequence from the variable parts of the frame and transmits it in the CRC field. The receivers also calculate the CRC sequence based on the bits they sample. A receiving node detects a CRC error, when its calculated CRC sequence and the received one are unequal. The CRC sequence is calculated differently in Classical CAN frames and in CAN FD frames. In Classical CAN, a 15-bit CRC polynomial is used and the dynamic stuff bits are not included in the CRC calculation. In CAN FD frames, the dynamic stuff bits before the CRC field are included in the CRC calculation. Further, a 17-bit polynomial is used for CAN FD frames with up to 16 data bytes and a 21-bit polynomial for CAN FD frames with up to 64 data bytes.

All three CRC polynomials are BCH codes, they are adapted to the three different message lengths in order to keep a Hamming distance (HD) of 6. HD=6 means that up to 5 bit flips are detected in a code word (frame) of fixed length. Since the chosen generator polynomials contain the factor (x+1), they are also able to detect any odd number of bit flips [3].

1.4 Acknowledgement

Transmitters expect to get an active acknowledgement for their frames, which is a dominant bit in the ACK slot. When a transmitter does not sample a dominant bit during ACK slot, it regards this as an ACK error. The transmitter considers a frame that does not get an acknowledgement as invalid and retransmits it (if retransmission is not intentionally disabled).

1.5 Stuff Rule Check

The bits of a CAN frame between Start-of-Frame and CRC Delimiter are coded by the method of bit stuffing. That means, the transmitter inserts, after each sequence of five consecutive equal bits, one bit of inverse value, called a (dynamic) stuff bit. The purpose of stuff bits is to ensure that there are enough edges in the bit stream for resynchronization of the receivers. This coding also ensures that, when an active error flag (consisting of six consecutive dominant bits) overwrites parts of a frame, this is detected by all nodes latest at the sixth bit of the error flag.

Receivers check the stuff rule and detect a stuff error at the bit time of the sixth consecutive equal bit level.

A different stuffing method is used in the CRC field of CAN FD frames. Here the stuff bits are inserted at fixed positions, once at the start of the CRC field and then after each fourth bit of the field. The value of such a fixed stuff bit is the inverse value of the preceding bit. This also ensures that there are no sequences of six consecutive equal bits, so receivers check the stuff rule also in CAN FD frames.

There is an additional stuff rule check for the fixed stuff bits in the CRC field of a CAN FD frame. Here the receivers check that each group of four bits is followed by a fixed stuff bit of inverse value. When a receiver samples a fixed stuff bit that has not the expected value, the receiver regards this as a format error.

1.6 *Stuff Count Check*

The Stuff Count check is a new error detection mechanism, introduced for the CAN FD frame format. In Classical CAN frames and in CAN FD frames before the CRC field, a transmitter inserts stuff bits dynamically, after five consecutive bits of equal value.

The Stuff Count in CAN FD frames is the number of dynamic stuff bits (modulo 8), counted by the transmitter of the frame. It is transmitted Gray-coded, with a parity bit, at the beginning of the CRC field. The invariable fixed stuff bits of the CRC field are not counted. All receivers compare the received Stuff Count value with their own Stuff Count. If the two values do not match, the receiver treats this like a CRC error and it regards the frame as invalid. The Stuff Count was introduced during ISO standardization by the experts of the project team to safeguard CAN FD frames against errors that convert dynamic stuff bits into data bits and vice versa.

1.7 *Interaction between Error Detection and Error Signaling*

Error signaling disturbs the current frame and thereby converts local errors into global errors in order to ensure data consistency in the network.

All nodes in the CAN network check the validity of all frames. If at least one node detects an error, it signals this error to the other nodes by sending an error frame. In most cases, a node starts sending the error frame immediately after the bit where it detected the error. In case of a CRC error, a receiver delays the error frame until after the ACK delimiter, so that a transmitter cannot mistake the start of the error flag as ACK.

The six consecutive dominant bits of the active error flag invalidate the current frame for all nodes, latest when the sixth bit causes a stuff error. The transmitter, monitoring the bit stream (cf. chapter 1.1), will detect a bit error at a transmitted recessive (stuff) bit and will abort (and later restart) the transmission.

There is another interaction between the CRC error and the acknowledgement error. Receivers that detect a CRC error (or a Stuff Count mismatch in CAN FD frames) will not immediately respond with an error frame, only after the ACK delimiter. Meanwhile they do not give the active acknowledgement expected by the transmitter. If no receiver gives the active acknowledgement, the transmitter detects the acknowledgement error. Even if some receivers send an ACK, the other receivers that detected a CRC error will start an error flag after ACK delimiter, causing all nodes to see a format error.

Nodes that see a high number of errors may enter error passive state where they are no longer able to transmit active (dominant) error flags. Under the assumption that, when a node sees more local errors than other nodes, there is also a local cause for the errors (e.g. bad contact to bus line), the error counting rules will cause such a node to enter error passive state earlier than the other nodes. The other nodes are then able to communicate without being disturbed by the faulty node. A further increase of its error count causes the faulty node to enter bus-off state, which effectively disconnects the node from the bus.

2. Improved Error Detection in CAN FD

CAN FD has four main improvements regarding error detection.

The Classical CAN CRC polynomial was chosen for frames with up to 8 data bytes. The longer CAN FD frames are protected by *(1)* longer CRC polynomials to keep the Hamming Distance equal to that in Classical CAN.

In Classical CAN, some rare cases are known [4] where two bit flips that generate and eliminate stuff conditions are not detected by the CRC. This problem (conversion of data bits into stuff bits or vice versa) is addressed in CAN FD by two new mechanisms, the *(2)* inclusion of the dynamic stuff bits into the CRC calculation and the *(3)* introduction of the Stuff Count.

Having *(4)* fixed stuff bits in the CRC field is also a new mechanism introduced in CAN FD. The fixed stuff bits increase the capacity of the format check. Further, *(1)*, *(3)*, and *(4)* reduce the probability of a random bit sequence to match with an expected CRC field by several orders of magnitude compared to Classical CAN. Chapter 4 describes this case.

3. Fault Types

In the context of this paper we define:

- **Fault**: An erroneous frame modification seen by a CAN node.
- **Error**: A fault detected by the protocol's error detection mechanisms.

A frame modification occurs in two steps. Firstly, a disturbance (e.g. EM radiation) changes either the CAN Bus signal (recessive/ dominant) or the digital signals CAN RX/TX between controller and transceiver. Secondly, this falsification causes a receiving node to interpret the transmitted frame differently (i.e. modified) and a transmitting node may detect a bit error.

During ISO standardization of CAN FD, the experts of the project team assumed two fault types [5]: fault type A and fault type B. The following two sections introduce these fault types.

3.1 Fault Type A: Bit Flip

"Fault type A" means that a CAN node samples a bit with the inverse (flipped) value compared to the transmitted bit value. This fault type can occur in a receiving node and in a transmitting node. Figure 2 shows an example for such a bit flip at bit 3.

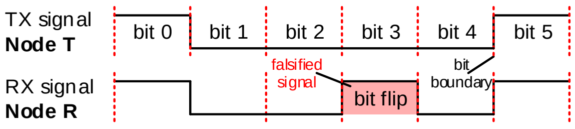

Figure 2: Fault Type A: Bit flip example

3.2 Fault Type B: Bit Drop or Bit Insertion

"Fault type B" means that a receiving node drops a bit from or inserts a bit into the bit sequence. Fault type B cannot occur in a transmitting node as this generates the bit sequence.

In order to cause a fault of type B, the following needs to happen. A disturbance (e.g. EM radiation) influences the CAN physical layer. As consequence, additional or shifted recessive to dominant (rec-dom) edges appear in the bit sequence. Transmitting and/or receiving node resynchronize based on these faulty edges. This resynchronization may increase the phase error ([6], [2]) between transmitting and receiving node. A hard synchronization may increase the phase error even more. When the absolute value of the phase error is above a critical level, the receiving node drops a bit from or inserts a bit into the bit sequence.

General properties of fault type B are:

- Fault type B may appear in the arbitration phase and in the data phase of a CAN FD frame.
- Fault type B can also happen in connection with stuff conditions. This means the receiving node may interpret dynamic stuff bits as data bits and vice versa.
- Fault type B can theoretically happen several times per frame.
- The relation between the transmitting node's and the receiving node's clock rates (due to the oscillators' frequency tolerance) determines if a bit drop or insertion is more likely to happen. The clock rates' relation determines the naturally introduced phase error between the nodes.
- The longer a bit sequence without a resynchronization to a correct rec-dom edge, the higher can be the natural phase error between transmitting and receiving node. A high natural phase error makes fault type B more likely.
- A transmitting and/or receiving node can stimulate fault type B by one or more resynchronizations on faulty rec-dom edges. The actual fault type B (bit

drop or insertion) occurs only in the receiving node.

- If several faulty rec-dom edges are necessary to stimulate a fault type B, these have to be **consecutive**. A correct rec-dom edge between two faulty edges reduces the phase error between transmitting and receiving node.
- The CAN bit timing configuration in the receiving node defines the sample point (SP) position and the synchronization jump width (SJW) [7], [8]. These two parameters mainly influence if a bit drop or insertion is more likely.

3.3 Fault Type B: Bit Drop Example

Figure 3 shows an example for a bit drop. The receiving node has a slower clock (= longer bits) than the transmitting node, due to the oscillator tolerance. This introduces an initial phase error. The receiving node resynchronizes into the wrong direction due to the shifted rec-dom edge. The phase error at bit 5 of the receiving node is so large that the sample point of bit 5 is already inside the following transmitted stuff bit. Now the receiving node samples the bit sequence “100000i” as “100001”.

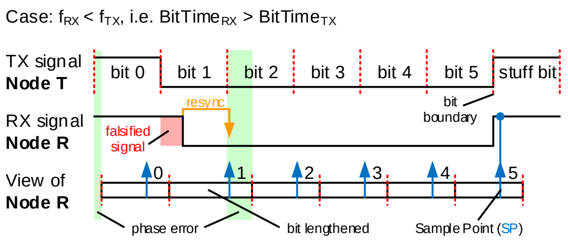

Figure 3: Bit drop example

3.4 Fault Type B: Bit Insertion Example

Figure 4 shows an example for a bit insertion. The receiving node has a faster clock (= shorter bits) than the transmitting node, due to the oscillator tolerance. This introduces an initial phase error. The receiving node resynchronizes into the wrong direction due to the shifted rec-dom edge. The phase error at bit 5 of the receiving node is so large that the sample point of bit 5 is still inside transmitted bit 4. Now the receiving node samples the bit sequence “100001” as “100000i”. To achieve a bit insertion by resynchronization on a single faulty rec-dom edge, a suboptimal SP position of 30% needed to be used in this example.

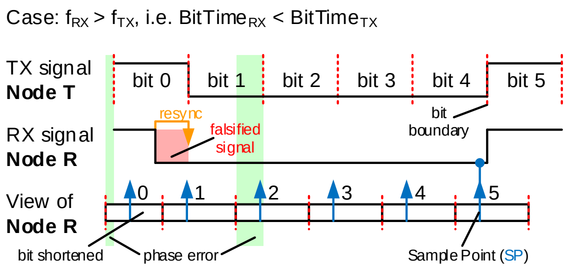

Figure 4: Bit insertion example

3.5 Fault Type B in a Practical Setup

To demonstrate how realistic the occurrence fault type B is, we use a (from today’s point of view) typical setup. Herein we focus only on the data phase of the CAN FD frame. We assume a data phase bit timing with following properties: SP = 70% and SJW = 20% of the data bit time. In this setup the receiving node requires a phase error of > 30% of the bit time for a bit drop and a phase error of > 70% for a bit insertion. Figure 5 visualizes this bit timing configuration.

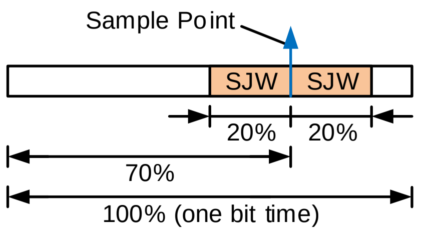

Figure 5: Exemplary data phase bit timing configuration

Bit drop: Three preconditions have to be met simultaneously for a bit drop to be caused by a **single** disturbance.

- In context of the oscillator tolerance, the receiving node’s clock needs to be sufficiently slower than the transmitting node’s clock (e.g. f_{RX} = 39.85 MHz, f_{TX} = 40.15 MHz),
- A sufficiently long bit sequence without rec-dom edge is necessary to accumulate a sufficiently large phase error in the receiving node, e.g. “00001111”.

- An disturbance has to shift the rec-dom edge by a sufficient amount (cf. Figure 3). The receiving node synchronizes on this faulty edge, increases its phase error, and drops one of the following bits.

Bit insertion: In this setup, it is not possible to insert a bit with a **single** disturbance. The receiving node can increase its phase error with a synchronization to a faulty edge only by SWJ (20%). If receiving and transmitting node's oscillators were ideal with respect to their oscillator tolerance ($f_{RX} = f_{TX}$), then four consecutive resynchronizations on faulty rec-dom edges are necessary to gain phase error of > 70%. Even if the receiving node has, within the allowed oscillator tolerance, a much faster clock (see Figure 4), then still at least two coordinated disturbances are necessary to insert a bit.

Bit drop and insertion: In this setup a bit drop and a bit insertion in the same CAN FD frame are hard to achieve. Example 1: If the nodes' oscillators have nearly no tolerance, 6 coordinated disturbances are needed: 4 for bit insertion and 2 for bit drop. Example 2: With an oscillator frequency relation that privileges bit insertion, still about 4 coordinated disturbances are necessary: 2 for insertion and 2 for bit drop. Consequently, we consider both a bit drop and a bit insertion occurring in the same frame as practically impossible.

We can summarize that fault type B is complex and requires many preconditions to be met. A single disturbance can cause one bit drop, but only under certain constraints. Several coordinated disturbances are necessary for one bit insertion. Consequently, a bit insertion in a typical setup should be considered as a "multi bit error". Finally, we consider a bit drop and a bit insertion occurring in the same frame as practically impossible.

4. Classification of Error Cases

All error detection mechanisms have their limits. In CAN, a transmitting node monitors if its bits appear correctly on the bus and aborts the frame when it detects a bit flip (fault type A). To remain potentially undetected, bit flips have to be local errors at one or more receiving nodes. By definition, a bit drop or insertion (fault type B) can only occur at a receiving node (see chapter 3.3). Consequently, only a receiving node may fail to detect a fault.

In [4] the authors classify the error cases that can occur in a receiving node. In this paper, we extend these classes to fit CAN FD.

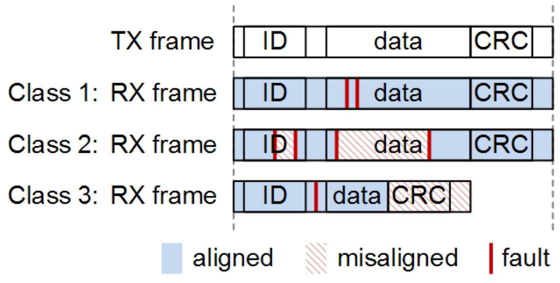

Figure 6: Classes of error cases for a receiving node – one example per class

4.1 Class 1: Normal Bit Errors

The receiving node samples the whole frame while it is **aligned** to the transmitted frame, i.e. it correctly recognizes the frame type, frame length, CRC field, etc. (see Figure 6). **Faults of type A** (bit flip) may be present in the received frame.

For this error class the Cyclic Redundancy Check (CRC) is a valid method to detect the erroneous frames. Both, CAN FD and Classical CAN use CRC polynomials that guarantee a Hamming Distance of 6. See chapter 1.3 for details.

4.2 Class 2: Encoding Related Errors

The receiving node samples the frame while it is **aligned** to the transmitted frame, i.e. it correctly recognizes the frame type, frame length, CRC field, etc. The receiving node may be **temporarily misaligned** to the transmitted frame during the sampling of the frame identifier (ID) or the data field.

Figure 6 shows an example. **Faults of type A** (bit flip) **and B** (bit drop and bit insertion) may be present in the received frame. They may cause the receiving node to interpret dynamic stuff bits as data bits and vice versa.

As long as only faults of type A (bit flip) are present in the frame, the CRC is a valid method to detect erroneous frames with up to 5 bit flips. When faults of type B are present in the frame, the CRC is corrupted and is therefore not sufficient to decide with absolute certainty, if the frame was received correctly or not. The Stuff Count in the CRC field of the frame contains the number of dynamic stuff bits in the transmitted frame (see chapter 1). This enables the receiving node to detect conversions between dynamic stuff bits and data bits.

4.3 Class 3: Message Length Modifying Errors

The receiving node samples at least the CRC field and EOF of the frame **misaligned** to the transmitted frame. The parts of the frame prior to the CRC field may be sampled aligned but also misaligned. Figure 6 shows an example. The misalignment of the receiving node starts earliest with the first fault. The misalignment of the receiving node may vary from a single bit to many bytes. The receiving node may sample key-bits (DLC, IDE, FDF, BRS) as transmitted or erroneously. **Faults of type A** (bit flip) **and B** (bit drop and bit insertion) may be present in the received frame.

The receiving node accepts the frame as valid if the following three constraints are all fulfilled: (i) the CRC field (sampled misaligned) accidentally matches the expected value, (ii) the frame format (reserved bits, delimiters, EOF, etc.) and dynamic stuffing are correct, and (iii) the ACK bit does not overwrite a recessive bit of the transmitted frame.

Constraints (ii) and (iii) are nearly identical in Classical CAN and CAN FD. Constraint (i) – the CRC field that has to fit accidentally – is much longer in CAN FD. A comparison of the CRC field lengths visualizes this:

• Classical CAN:	16 bit
• CAN FD (CRC17):	28 bit
• CAN FD (CRC21):	33 bit

This means in this error class more bits have to match accidentally for a CAN FD frame compared to a Classical CAN frame: With CRC17 28–16=12 more bits and with CRC21 33–16=17 more bits.

Remark: In Classical CAN, only the assumed CRC sequence bits have to match accidentally. The dynamic stuff bits (if present) automatically match, because the transmitting node sends a correctly stuffed data field. In CAN FD frames, not only the assumed CRC sequence bits, but also the Stuff Count and the fixed stuff bits have to match accidentally. The fixed stuff bits need to be considered, as the receiving node expects in the CRC field after each 4 data bits a fixed stuff bit, but the transmitting node typically adds after 5 equal data bits a dynamic stuff bit. Consequently, the stuff bits do not automatically match like in Classical CAN[1].

5. Summary and Conclusion

CAN provides several error detection mechanisms (e.g. Frame Format Check, Cyclic Redundancy Check, etc.). This paper gave an overview on these mechanisms and their interaction.

It also outlined the four main improvements in CAN FD regarding error detection: *(1)* use of CRC polynomials of higher order, *(2)* inclusion of the dynamic stuff bits into the CRC calculation, *(3)* introduction of the Stuff Count, and *(4)* introduction of fixed stuff bits in the CRC field.

Improvements *(2)* and *(3)* enable to detect faults where data bits are converted into

[1] It might happen that in the assumed CRC field of a received CAN FD frame all fixed stuff bits match with a dynamic stuff bit in the transmitted frame. As there are very few bit sequences where this is the case, we neglect this.

stuff bits and vice versa. The fixed stuff bits *(4)* extend the capability of the frame format checks. Improvements *(1)*, *(3),* and *(4)* reduce the probability of a random bit sequence to match with an expected CRC field by several orders of magnitude.

This paper introduced the two fault types that the experts of the project team assumed during the ISO standardization of CAN FD: Fault type A (bit flip) and fault type B (bit drop and insertion).

A bit flip has to be local to a receiving node to have the potential to stay undetected. A transmitting node would immediately detect the bit flip. A bit drop may be caused in a practical setup by a single disturbance (e.g. EM radiation) – but only if several conditions are met simultaneously. For one bit insertion several coordinated disturbances are necessary in a practical setup. Consequently, a bit insertion should be considered as a “multi bit error” in a practical setup. We consider a bit drop and a bit insertion occurring in the same frame as practically impossible.

The paper classified the potential error cases at a receiving node into three classes: Class 1: Normal bit errors, Class 2: Encoding related errors, and Class 3: Message length modifying errors. In class 3 error cases, the receiving node samples a random bit sequence as CRC field. To accept the frame this random bit sequence needs to accidentally match the expected CRC field. Compared to Classical CAN, in CAN FD frames 12 more bits (CRC17) or 17 more bits (CRC21) need to match in the expected CRC field.

6. Acknowledgment

We want to thank our colleague Dr. Jan Scheuing for the valuable discussions and his support.

Author	Dr. Arthur Mutter
Company	Robert Bosch GmbH
E-mail:	arthur.mutter@de.bosch.com
Author	Florian Hartwich
Company	Robert Bosch GmbH
E-mail:	florian.hartwich@de.bosch.com

References

[1] F. Hartwich, "CAN with Flexible Data-Rate," in Proceedings of the 13th international CAN Conference, Hambach Castle, Germany, 2012.

[2] ISO/DIS 11898-1:2015, Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signaling, 2015.

[3] P. Koopmann und T. Chakravarty, „Cyclic Redundancy Code (CRC) Polynomial Selection For Embedded Networks,“ in The International Conference on Dependable Systems and Networks (DSN), 2004.

[4] J. Unruh, H. J. Mathony und K. H. Kaiser, „Error Detection Analysis of Automotive Communication Protocols,“ in SAE Int. Congress, No. 900699, Detroit, 1990.

[5] A. Mutter, Summary of the discussion regarding the CAN FD CRC issue, Germany: CAN in Automation, 21.12.2014.

[6] A. Mutter, "Robustness of a CAN FD Bus System – About Oscillator Tolerance and Edge Deviations," in Proceedings of the 14th international CAN Conference, Paris, France, 2013.

[7] F. Hartwich, "The Configuration of the CAN Bit Timing," in Proceedings of the 6th international CAN Conference, Turin, Italy, 1999.

[8] F. Hartwich, "The Configuration of the CAN FD Bit Timing," in Proceedings of the 14th international CAN Conference, Paris, France, 2013.