CAN FD: Improved residual error-rate

Classical CAN provides several error-detection mechanisms. They determine the residual error rate. CAN FD uses the same mechanisms and an additional one that reduces the probability of undetected errors further.

Related articles

- Florian Hartwich, Robert Bosch GmbH
  CAN with flexible data-rat
  CAN Newsletter (print), June 2012
- Magnus-Maria Hell, Infineon Technologies
  The physical layer in the CAN FD world
  CAN Newsletter (print), March 2014
- Bernd Elend, NXP
  CAN FD: Impact on system design
  CAN Newsletter (print), June 2014

References

[1] P. Koopman and T. Chakravarty: CRC polynomial for embedded networks, International Conference on Dependable Systems and Networks (DSN-2004)

[2] F. Yang: Residual error rate of CAN FD (unpublished paper), June 2014

One of the most powerful error-detection mechanisms is the CRC (Cyclic Redundancy Code) embedded in each CAN data/remote frame. The 15-bit polynomial used in Classical CAN provides a Hamming Distance (HD) of six, meaning it can detect all randomly distributed 5-bit failures in a single frame. It can also detect any 15-bit burst errors.

CRCs are a first line of defense against data corruption. The achievable HD depends on the length of the data to be protected. The chosen 15-bit CRC is capable of detecting 5-bit errors when the protected data has 112 bit or less [1]. However, CRCs protect data only if the bit string has the very same size (the same number of bits) on the transmitting and the receiving sides. In Classical CAN you find cases in which two bit-flips (generating/eliminating stuff conditions) can lead to a valid frame from the view of the CRC. The reason for this is that the dynamic stuff-bits are not considered in the CRC calculation.

To overcome this "weakness", the CRCs that are used in the CAN FD protocol consider the dynamic stuff-bits. Additionally, in the CRC field fixed stuff-bits are used. Unfortunately, considering the dynamic stuff-bits in the CRC calculation allows a situation in which a single error is not detectable. This happens for example, when a local glitch leads to a mis-synchronisation of a receiving node while the CRC generator registers are at "0…0". If such a glitch coincides with a stuff condition, it may happen that the receiving node reads the bit sequence "00000i" (i = stuff-bit) as "00001". In other words, this is a shortening of the frame by skipping a bit. Of course, this scenario is not likely. Nevertheless, this has a negative impact on the residual error-rate. Engineers working with Renesas found such scenarios: They showed that a corruption of the Start-of-Frame bit is not detectable by means of the CRC mechanism. Subsequently, experts from Bosch showed that this may also happen at other positions of the frame.

"To solve this weakness of the CAN FD protocol, we proposed to introduce a stuff-bit counter (SBC)," said Dr. Arthur Mutter from Bosch. "The receiving node needs to know the total number of transmitted bits for each frame. From the protocol specification and the DLC (data length code) the receiver knows the length except for the number of dynamic stuff-bits. It is sufficient to transmit the stuff-bit count modulo 8, because an HD of "just" six is required. The three SBC bits are able to detect up to seven lengthening or shortening errors, which otherwise could remain undetected, if they coincided with stuff conditions."

The SBC bits belong to the CRC field where fixed stuff-bits are used. They are transmitted before the CRC bits. The SBC bits are not part of the Classical CAN stuff-bit rule, because a stuff-bit in the SBC cannot be included in the counting. The SBC bits are protected by the CRC calculation.

Safeguarding of the SBC

When a stuff-bit is dropped or inserted by synchronization failure, the CRC is corrupted. If in the same frame a bit-flip falsified the stuff-bit count, the receiver may not be able to detect this error. This is why the SBC needs to be safeguarded. There are two safeguards implemented now:

- Adding an even parity–bit
- Gray coding the stuff-bit count

▷

Table 1: The stuff-bit counter and its parity bit is located in the CRC field in front of the CRC polynomial, which starts with an fixed stuff-bit

Stuff-bit count (modulo 8)	First bits of the CRC field		
	SBC value	SBC parity bit	Fixed stuff-bit
0	000	0	1
1	001	1	0
2	011	0	1
3	010	1	0
4	110	0	1
5	111	1	0
6	101	0	1
7	100	1	0

The table shows the SBC bits, the parity-bit, and the following fixed stuff-bit. “The parity check and the fixed stuff-bit (always with the inverted value of the preceding bit) detect any single-bit error of these bits,” explained Dr. Mutter. “The same holds for two bit-flips, if at least one of the bit-errors occurs in the parity-bit or the following fixed stuff-bit. If two bits in the Gray-coded SBC are corrupted, this results in a stuff-bit count value with a difference of at least 2. This is detected through a comparison with the internally counted stuff-bit value. The minimum number of bit-errors that could remain undetected is four. This happens only if two bit-flips in the Gray-coded SBC coincide with two stuff-bits dropped or inserted.”

The receiver checks the received stuff-bit count (modulo 8) with its internal count and also performs a parity check. A mismatch during the SBC comparison is treated the same way as a detected CRC error. This means that the related Error flag is transmitted after the ACK field.

Detection of all single-bit errors

Other 2-bit errors can cause undetected faulty messages too. If the IDE (identifier extension), the FDF (FD frame), or one of the DLC bits are corrupted and in the data field one of the stuff-bits is evaluated as recessive by mistake, it is possible that a “shorter” valid frame is accepted by the receiving node. This has been described in detail by the Chinese researcher Fuyu Yang [2]. Of course, the additional bits would cause an error condition. But the “faulty” message in front has already been accepted and eventually processed (depending on the acceptance filtering settings). Even if these scenarios are very unlikely, they need to be considered when calculating the residual error-rate. In this case, the receiver checks the CRC while the transmitter sends data bits. There is a probability that the perceived CRC field matches with random data bits depending on its length. The residual error-rate of CAN FD is expected to be much lower than in Classical CAN because the CRC field is much longer in CAN FD. In Classical CAN this critical bit sequence is 15 bit long while in CAN FD it is 27 bit in a frame with CRC-17 and 32 bit in a frame with CRC-21.

In order to improve the CRC checking, the initialization vector for the CRC-17 and CRC-21 has been changed from (0..0) to (100..0), where the “1” is at the most significant bit position followed by “0”. All these improvements will be introduced in the next version of ISO 11898-1 CAN data link layer standard, which is currently under review.

Holger Zeltwanger

Specialist for Custom Made Industrial Computer Systems

Janz Tec AG, has been one of the leading manufacturers of electronic assemblies and complete industrial computer systems since 1982. The company describes itself as *Industrial Computing Architects* and develops embedded PCs, industrial computers and industrial communication according to customer requirements.

With 30 years of experience in the market, Janz Tec AG is capable of meeting individual customer requirements at any time – this starts as early as the concept phase, continues through the development and production of the components up to series delivery.

Janz Tec AG
Im Doerener Feld 8
33100 Paderborn
Fon: +49.52 51.15 50-0
www.janztec.com

janztec
Industrial Computing Architects