ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 5, May 2015

Third Bit Complement (TBC) Mechanism to Reduce Bit Stuffing Jitter in Controller Area Network (CAN)

Maha Medhat Hassan

Demonstrator, Dept. of Systems and Computers, Engineering College, Al-Azhar University, Cairo, Egypt

ABSTRACT: A In spite of wide usage of controller area network (CAN) in the embedded networked control system, nondeterministic response time have restricted the wider use of CAN in safety-critical real time systems, since the CAN uses Non Return to Zero (NRZ) coding and includes a bit-stuffing mechanism, which causes the CAN frame length to become a complex function of the data contents. To minimize CAN response-time jitter and make the transmission time fixed value according to the number of bytes in the data field, many techniques have been suggested as XOR masking , Software Bit Stuffing (SBS), inversion bit stuffing mechanism (IBSM) and Eight-to-Eleven Modulation (EEM), in this paper a novel alternative method known as Third Bit Complement (TCB) is used for "bit stuffing" to prevent hard-ware bit stuffing in the data field part completely, the new technique was compared with previous available techniques showing that six data bytes can be inserted in the data part compared with five bytes in SBS and EEM techniques.

KEYWORDS: Controller area network, bit stuffing, third bit complement, Non Return to Zero, CAN response-time jitter.

I.INTRODUCTION

Controller area networks (CANs) [1] have steadily gained popularity and are nowadays adopted in a variety of embedded networked control systems; it is also a popular solution to support real-time communication in a number of different contexts, which range from the automotive scenario to factory automation environments. Due to the low cost and interesting performance, it is used in many (networked) embedded control systems as well.

While there is no doubt that CAN will be the solution of choice for many years to come in all those cases where inexpensive implementation is a key requirement, many research activities have been carried out recently to demonstrate its suitability for special-purpose applications with tight timing requirements too. As CAN uses Non Return to Zero (NRZ) bit encoding [2], long sequence of same polarity bit will cause the losses of synchronization among the CAN controller. CAN provide an effective mechanism for clock synchronization known as "bit-stuffing", in which no more than five consecutive bits (with the same polarity) are transmitted on the bus .Transmitted bits are checked before transmission; if a long sequence of same polarity bit comes, it add an opposite polarity bit with a regular interval. The added bit will give an extra transition for the synchronization of receiver and transmitter. Fig.1 shows the basic operation of the bit-stuffing mechanism carried out in the sending end of CAN controller [3].

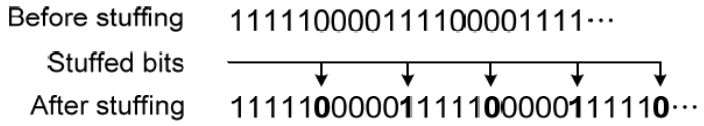

Fig 1 the basic bit stuffing operation

Fig. 1 shows the basic operation of bit-stuffing in the sending CAN controller, where any five consecutive identical bits(11111 or 00000) are considered for transition, then an additional bit of the opposite polarity should be added before transmission to keep synchronization.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 5, May 2015

The bit stuffing (BS) mechanism of CAN causes the message transmission time to become (in part) a complex function of the contents of the data fields. This variation in transmission times makes it difficult to predict the precise behaviour of real-time systems, the variation in message response time is referred to as "Jitter", which may worsen the overall accuracy of the control system [4].
Impacts of jitter on various applications are given in [5, 6]. Further possible sources of jitter and proposed solutions to limit this jitter in CAN networks are presented in [7] – [8]. The proposed contribution on this paper only deals with jitter caused by bit stuffing.

In order to overcome this particular issue, a number of solutions have appeared in the past few years that rely on specific encoding schemes. The first proposed approach was XOR masking [9, 10], which was subsequently extended with selective XOR approaches [11]. Higher performance was achieved by means of software bit stuffing [12] and eight-to-eleven modulation [4], and recently Inversion Bit Stuffing Mechanism (IBSM) [14] has been introduced to reduce the frequently stuffed bits, where the data bytes (2-5) have only one stuffing bit (in average).
The goal of this paper is to reduce the timing variation due to variable message length causes by bit-stuffing in CAN protocol.
The rest of this paper comprises the following. A brief background discussion of the previous different "bit-stuffing" techniques are discussed in section II, the new "third bit complement (TCB) technique" is proposed and the implementation algorithm is presented in the section III, different techniques comparison is illustrated in section IV, Finally a conclusion is drawn in section V.

II. PREVIOUS WORKS

A. Selective XOR masking

XOR masking was first proposed by Nolte et al. [9, 10], by analysing 25,000 CAN frames from an automotive system, Nolte found that the probability of having bit value of ''1" (or ''0") in the data segment is not 50%. To remove these sequences of ones and zeros that can be highly found in a real CAN traffic, Nolte presented a simple coding scheme based on XOR-bit-operation, in this technique transmitted bytes are XOR-ed with the masking bit "101010…" Fig. 2, the receiver must apply the same bit operation to extract the original data.

Original frame:	000000111110011000000111 …
XOR with bit-mask:	101010101010101010101010 …
Transmitted frame:	101010010100110010101101 …

Fig. 2 XOR encoding process for Nolte method

In fig. 2, the original data frame is xor_ed with the bit mask (10101010.....) and the result is final frame to be transmitted on the CAN bus.
This technique reduces CAN message length variations (i.e. jitter) to low levels and also reduces large computational or memory overheads [9] and [10].
In a more general case, the data transmitted may not have the same characteristics as those observed by Nolte. For example, if a general CAN message is modelled using random data, then use of the Nolte (XOR) transform would not be expected to produce a significant reduction in the level of bit-stuffing [11], In this case the data bytes are tested and the Nolte XOR masking is applied to the selected data byte.

B. Inversion Bit Stuffing Mechanism (IBSM)

Inversion Bit Stuffing Mechanism (IBSM) [14] is introduced to reduce the number of stuffed bits in the data field. In this technique the data field of the CAN frame was checked byte by byte, if five consecutive identical bits have been found in any byte (11111 or 00000), the fifth identical bit is inverted (11110 or 00001), at the receiving node this bit is inverted again to avoid changing of the data, so a flag bit is required to indicate the inversion process. one byte is reserved for flags, and seven bytes are used to data in the CAN frame instead of eight, The Inversion Bit Stuffing Mechanism (IBSM) has a good effect on the reduction of the number of frequently stuffed bits, where the data bytes (2-5) have only one stuffing bit (in average).

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 5, May 2015

C. Software Bit Stuffing (SBS)

In selective XOR masking [11] and Inversion Bit Stuffing Mechanism (IBSM) [14] the levels of message length variation was reduces significantly, but few conditions remain unsolved like the boundaries of the adjacent bytes are still processed individually, So the continuous 0 or 1 at the end of one frame and the same continuous 0 or 1 at the start of the next frame give a long sequence of 0 or 1. To completely eliminate the need for hardware bit stuffing in the CAN data segment, an alternative technique which is referred to 'software bit stuffing' has been proposed [12].
In this technique a given frame is checked before transmitting, if a sequence of four consecutive identical bits is detected, the algorithm adds an additional bit, of opposite polarity after-wards.

D. Eight-TO-Eleven modulation (EEM)

In SBS data encoding / decoding are performing at run-time (while the system is operating) using a function call approach [12]. Due to run time processing the CPU overheads of the processor increased.
To achieve high processor utilization as well as to reduce the effect of bit stuffing EEM was proposed.
EEM is a fixed-length 8-to-11 modulation scheme [4], which encodes every byte in the original payload as an 11-bit pattern, as shown in fig. 3

Bit 1	SB 1	Bit 2	Bit 3	Bit 4	SB 2	Bit 5	Bit 6	Bit 7	SB 3	Bit 8

Fig 3: Encoding data byte in EEM method (SB stands for "Stuffed Bit")

In Fig. 3 SB are the stuff bit, bit 1 to 8 are the original 8 bit input data. Stuff bits values are the negation of the previous bit.
In this encoding format the worst case data frame will not exceed (4) consecutive same polarities. For example let the input sequence will be "11110001". After encoding the result will be "10111000011" that satisfies the "bit-stuffing" condition and al-so solve the boundary condition problem.

III. THIRD BIT COMPLEMENT TECHNIQUE (TBC)

The data field in CAN frames—and, typically, the message payload as well—are encoded on an integral number of bytes. The encoding efficiency is defined as the ratio between the size of the original payload and the DLC value in the encoded frame [13].

Only XOR does not cause any overhead. On the contrary, part of the data field is unavoidably wasted in all of the other approaches selective (XOR, SBS, EEM and IBSM).
In this paper a new techniques that prevent hard- ware bit stuffing completely is proposed.
In SBS[12] and EEM[4] techniques a part of the data field is unavoidably wasted, and 5 byte at maximum can be encoded at the payload, because of encoding process in EEM[4] every 8 bit is encoded to 11 bit, while the SBS[12] a padding is required at the end of the encoded bit sequence to make its length fixed [13].

To achieve high processor utilization as well as to prevent hard-ware bit stuffing completely (prevent the bit stream 11111 or 00000), and to allow 6 byte payload to be encoded as maximum, the proposed new technique (TBC)is achieved by the follows steps:

-The complement of the fourth bit (B4) is initially inserted after the fourth bit.
-Then after every three bits, the complement of the third bits is then inserted.
- In this mechanism data of six bytes require 48 bit and 15 inserted bits (total of 63 bits) which can be fit in 8 bytes.
-Since six byte is the maximum payload size, the bit stream from (B1) to (B48) is encoded as in fig.4

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 5, May 2015

B_1	B_2	B_3	B_4	$\sim B_4$	B_5	B_6	B_7	$\sim B_7$	B_8
B_9	B_{10}	$\sim B_{10}$	B_{11}	B_{12}	B_{13}	$\sim B_{13}$	B_{14}	B_{15}	B_{16}
$\sim B_{16}$	B_{17}	B_{18}	B_{19}	$\sim B_{19}$	B_{20}	B_{21}	B_{22}	$\sim B_{22}$	B_{23}
B_{24}	B_{25}	$\sim B_{25}$	B_{26}	B_{27}	B_{28}	$\sim B_{28}$	B_{29}	B_{30}	B_{31}
$\sim B_{31}$	B_{32}	B_{33}	B_{34}	$\sim B_{34}$	B_{35}	B_{36}	B_{37}	$\sim B_{37}$	B_{38}
B_{39}	B_{40}	$\sim B_{40}$	B_{41}	B_{42}	B_{43}	$\sim B_{43}$	B_{44}	B_{45}	B_{46}
$\sim B_{46}$	B_{47}	B_{48}							

Fig 4: Data field encoding according to third bit complement (TBC) technique, the inverted shadowed bits are the inserted bits

As shown in fig. 4, the bit pattern of (11111 or 00000) can never be found because of the inverted inserted bits, but additional bits have been added. The required DLC bytes in the CAN frame are shown in table 1.

Table 1 the required bytes for data bytes (0-6) after insertion of inverted bits.

Payload size Sizes (bytes)	NO. of bits	NO. of inserted bits (TBC)	Total number of bits	DLC (bytes)
0	0	0	0	0
1	8	2	10	2
2	16	4	20	3
3	24	7	31	4
4	32	10	42	6
5	40	12	52	7
6	48	15	63	8

Table 1 shows the required DLC bytes after the insertion of the complemented bits for the data bytes (1-6) bytes. (Example: two additional bits ~B4 & ~B7 have been added to the first byte (B1-B8)).

IV. ANALYSIS AND COMPARISON OF BIT STUFFING TECHNIQUES

In third bit complement (TBC) technique the hardware bit stuffing can be completely avoided, and more data bytes can be inserted in the data field payload compared with EEM and SBS techniques [13], the comparison is shown in table 2.

Since in XOR[9] and selective XOR[11] techniques bit stuffing can be reduced but not prevented, only SBS[12] and EEM[4] are compared.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 5, May 2015

Table 2 The ratio between the size of the original payload and the DLC value in the encoded frame for SBS, EEM and TBC techniques

Payload size Size s (bytes)	Data field size DLC (bytes)		
	EEM[4]	SBS[12]	TBC(proposed)
0	0	0	0
1	2	2	2
2	3	3	3
3	5	5	4
4	6	6	6
5	7	7	7
6	-	-	8
7	-	-	-
8	-	-	-

As shown in table 2, six bytes can be inserted in the CAN data frame if the TBC technique is applied to prevent bit stuffing, compared with five bytes in both SBS[12] and EEM[4].

V. CONCLUSION

In order to reduce the impact of variation in transmission times due to Bit stuffing technique in controller area network (CAN) which is referred as “Jitter”, a new technique "third bit complement (TCB)" has been proposed to prevent the bit stuffing at all and make the transmission time fixed value according to the number of bytes in the data field.

"EEM"[4] and "SBS"[12] techniques have been proposed before to overcome the "jitter" problem, but the encoding scheme in this two techniques allows only five byte data as a maximum to be inserted in the data field, "third bit complement (TCB)" allows six byte to be inserted with very simple encoding process to avoid the bit pattern (11111 or 00000) which need bit stuffing.

REFERENCES

[1] International Organization for Standardization. ISO 11898-1 – Road vehicles – Controller area network (CAN) – Part 1: Data link layer and physical signaling, 2003.
[2] Mannisto, Daniel, and Mark Dawson. "An Overview of Controller Area Network (CAN) Technology." Raport instytutowy, Machine Bus Corporation, November (2003).
[3] M. Farsi, and M.Barbosa, CAN open Implementation, applications to industrial networks, Research Studies Press Ltd, England, 2000.
[4] Jena, Tapas Ranjan, Ayas Kanta Swain, and Kamalakanta Mahapatra. "A novel bit stuffing technique for Controller Area Network (CAN) protocol." Advances in Energy Conversion Technologies (ICAECT), 2014 International Conference on. IEEE, 2014.
[5] A. Stothert, and I. MacLeod, “Effect of Timing Jitter on Distributed Computer Control System Performance”. Proceedings of DCCS’98 –15th IFAC Workshop on Distributed Computer Control Systems ,September 1998.
[6]M. Nahas, M. Short, and M. J. Pont. The impact of bit stuffing on the real-time performance of a distributed control system. In Proc. 10th International CAN conference, pages 10.1–10.7, 2005.
[7] L. Rodrigues, M. Guimares, and J. Rufino, “Fault-Tolerant Clock Synchronization in CAN”, Proceedings of the 19th IEEE Real- Time Systems Symposium, Madrid, Spain, December 2-4, 1998.
[8] J. Barreiros, E. Costa, J. Fonseca, and F. Coutinho, “Jitter reduction in areal-time message transmission system using genetic algorithms”, Proceedings of the CEC 2000 – Conference of Evolutionary Computation, USA, July 2000.
[9] T. Nolte, H. Hansson, C. Norstr○m, and S. Punnekkat, “Using Bit stuffing Distributions in CAN Analysis”, IEEE/IEE Real- Time Embedded Systems Workshop (Satellite of the IEEE Real-Time Systems Symposium) Lon-don, 2001.
[10] Nolte, Thomas, Hans Hansson, and Christer Norstrom. "Minimizing CAN response-time jitter by message manipulation." Real-Time and Embedded Technology and Applications Symposium, 2002. Proceedings. Eighth IEEE. IEEE, 2002.
[11] Nahas, Mouaaz, and Michael J. Pont. "Using XOR operations to reduce variations in the transmission time of CAN messages: A pilot study." Proceedings of the Second UK Embedded Forum. 2005..
[12] Nahas, Mouaaz, Michael J. Pont, and Michael Short. "Reducing message-length variations in resource-constrained embedded systems implemented using the Controller Area Network (CAN) protocol." Journal of Systems Architecture 55.5 (2009): 344-354.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 5, May 2015

[13] Cena, Gianluca, et al. "Fixed-length payload encoding for low-jitter controller area network communication." IEEE Transactions on Industrial Informatics 9.4 (2013): 2155-2164.

[14] Maha Medhat Hassan, "Bit stuffing techniques Analysis and a Novel bit stuffing algorithm for Controller Area Network (CAN)", In International Journal of Computer Systems, pages 80-87, Volume 02- Issue 03, March, 2015.