Journal of Embedded Systems, 2018, Vol. 5, No. 1, 1-6
Available online at http://pubs.sciepub.com/jes/5/1/1

DOI:10.12691/jes-5-1-1

Impact of Reducing Bit Stuffing Jitter on the Control Performance of a CAN-Based Distributed Furnace System

Mouaaz Nahas*

Department of Electrical Engineering, College of Engineering and Islamic Architecture,
Umm Al-Qura University, Makkah, Saudi Arabia
*Corresponding author: mmnahas@uqu.edu.sa

Abstract The Controller Area Network (CAN) protocol is widely used in distributed real-time, resource-constrained embedded systems. CAN uses "Non Return to Zero" (NRZ) coding and employs a bit-stuffing mechanism for clock synchronization. Such a mechanism causes a variation in the CAN frame length which may have a detrimental impact on the control behaviour of safety-critical systems employing this protocol. To address this issue, two techniques known as "byte-based XOR masking" and "software bit stuffing" were developed and achieved a jitter reduction of up to 20% and 40%, respectively, when employed in practical designs. This paper investigates the effectiveness of such techniques in a real-time control application; that is a simple furnace system case study based on a "hardware-in-the-loop" (HIL) testbed facility. The results show that reducing bit stuffing jitter has the potential to improve the control performance of distributed real-time systems employing CAN protocol.

Keywords: *jitter, bit stuffing, scheduler, time-triggered, shared-clock, furnace system, hardware-in-the-loop*

Cite This Article: Mouaaz Nahas, "Impact of Reducing Bit Stuffing Jitter on the Control Performance of a CAN-Based Distributed Furnace System." *Journal of Embedded Systems*, vol. 5, no. 1 (2018): 1-6. doi: 10.12691/jes-5-1-1.

1. Introduction

The Controller Area Network (CAN) protocol is widely used in distributed embedded control systems [1,2]. CAN uses "Non Return to Zero" (NRZ) coding for bit representation, under which drift in the receiver's clock may occur when a long sequence of identical bits has been transmitted. This drift might, in turn, result in message corruption. To avoid the possibility of this scenario, the CAN communication protocol (in its physical layer) employs a bit-stuffing mechanism which operates as follows. After five consecutive identical bits have been transmitted in a given frame, the sending node adds an additional bit of the opposite polarity. All receiving nodes must remove the 'stuffed' bits to recover the original data [3,4]. Figure 1 provides a schematic illustration of the bit stuffing process in CAN hardware.

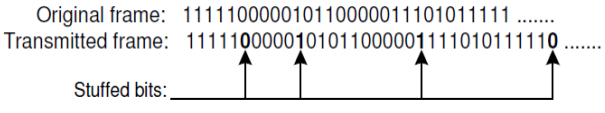

Figure 1. The CAN bit stuffing mechanism.

While providing an effective mechanism for clock synchronization between the communicating processors, the bit-stuffing mechanism in CAN makes it difficult to predict the transmission time of messages, a fact which may have an adverse impact on a wide-range of time- as well as safety-critical embedded applications. In more detail, the bit-stuffing mechanism causes the frame length to become (in part) a complex function of the data contents. In [5], the level of message variation that this process may induce was discussed and it was clearly demonstrated that at the maximum CAN baud rate (i.e. 1 Mbit/sec), the possible variation in message lengths is up to 24 µs. Of course, at lowers data rates that is very common in practice, it is expected that this variation would increase linearly with a potential impact on the performance of real-time systems employing this protocol. One key impact of bit stuffing is that there can be a relative jitter in the timing of tasks which are due to execute simultaneously on different nodes in a CAN network. The presence of such jitter may have a detrimental impact on the performance of many distributed embedded systems including control applications [6-14].

To begin to address this issue, Nolte et al [15,16] proposed a technique in which the data section of each CAN frame is XOR-ed with a bit-pattern that includes alternating ones and zeros (i.e. 101010…). At the receiver, the same bit operation must be applied to extract the original data. This study was based on the analysis of 25,000 CAN frames gathered from a real automotive system where it was found that probability of having bit value of "1" (or "0") in the data section was not 50% as assumed with pseudorandom data. In a more general case, where the data transmitted do not have the same

characteristics as those observed by Nolte and his colleagues, we would not expect to see a significant reduction in the level of bit-stuffing if the Nolte's (XOR) transform is applied (this was discussed and verified in [17]). Therefore, two techniques were proposed to deal with such high levels of message-length variations (i.e. jitter) when pseudorandom data is exchanged in the CAN network. The first technique was based on the investigation of each byte and only if the byte is subject to CAN bit stuffing is the whole byte XOR-ed with the bit-pattern 10101010; such a technique was described as "selective byte-based XOR masking" and referred to as "Nolte C" [5,17]. Please note that in [5], "Nolte A" refers to the direct application of Nolte's method, and "Nolte B" refers to a "frame-based XOR masking" which was also found ineffective with pseudorandom data. The results of implementing "Nolte C" technique in practical systems – employing time-triggered, shared-clock scheduling protocols [18] – demonstrated a 20% reduction of transmission jitter. To achieve a better improvement in the overall temporal behaviour of the system, the "software bit stuffing" (SBS) [5] and "eight-to-eleven modulation" (EEM) [19] techniques were developed and evaluated. Both techniques were found to be effective in reducing the jitter – in CAN frames – by up to 40% (which is twice that achieved by "Nolte C" approach). However, as detailed in [19], the main advantage of EEM over SBS is its flexibility, manifested in the wide range of possible implementation options that can be adopted. In contrast, the SBS – which can only be implemented using online approach – has the potential to reduce memory requirements in practical hardware designs, making it superior to EEM in many practical cases. Note that the abovementioned jitter-reduction techniques have been taken further by other researchers, and a set of modified versions as well as alternative techniques were presented to deal with jitter and clock synchronisation issues in CAN-based systems [20-26]. It is worth noting that CAN protocol still works well for many systems due to its profound roots in automotive industry besides its simplicity, low implementation costs and widespread availability [27]. Moreover, experience gained with CAN over the past decades allows the creation of extremely reliable systems using this protocol [28].

The impact of a preliminary version of software bit stuffing algorithm on a control performance of a distributed CAN-based adaptive cruise control system was investigated in [11] and a general improvement had been achieved. In this paper, we seek to explore the impact of applying "Nolte C" and SBS on another realistic case study; that is the distributed control of a CarboNitriding heat treatment furnace system built using "hardware-in-the-loop" (HIL) testing method [29,30]. HIL has been extensively used in embedded systems development and refinement processes, particularly for automotive applications (see, for example, [31-37]).

The main aim of this study is to show how much improvement in the control performance of such a furnace system can be achieved by reducing transmission jitter in the CAN network when applying "Nolte C" and SBS techniques. The system considered here has two nodes connected via CAN (hardware) protocol and the "shared-clock" (S-C) scheduling (software) protocol that is employed to achieve time-triggered message communication (see [38] for various implementation schemes of the S-C scheduler). Moreover, each individual node executes its tasks using a "time-triggered co-operative" (TTC) scheduler (see [39,40] for further details).

The remainder of the paper is organised as follows. Section 2 describes the CarboNitriding heat treatment furnace system. In Section 3 the experimental methodology used in this study is outlined. Section 4 presents the empirical results obtained in this study which includes both jitter and control performance measures. Finally, the overall paper conclusions is drawn in Section 5.

2. Furnace System

CarboNitriding heat treatment furnace system is used here as a real-time case study which can be implemented using HIL testbed. The furnace system designed in this study will be first described.

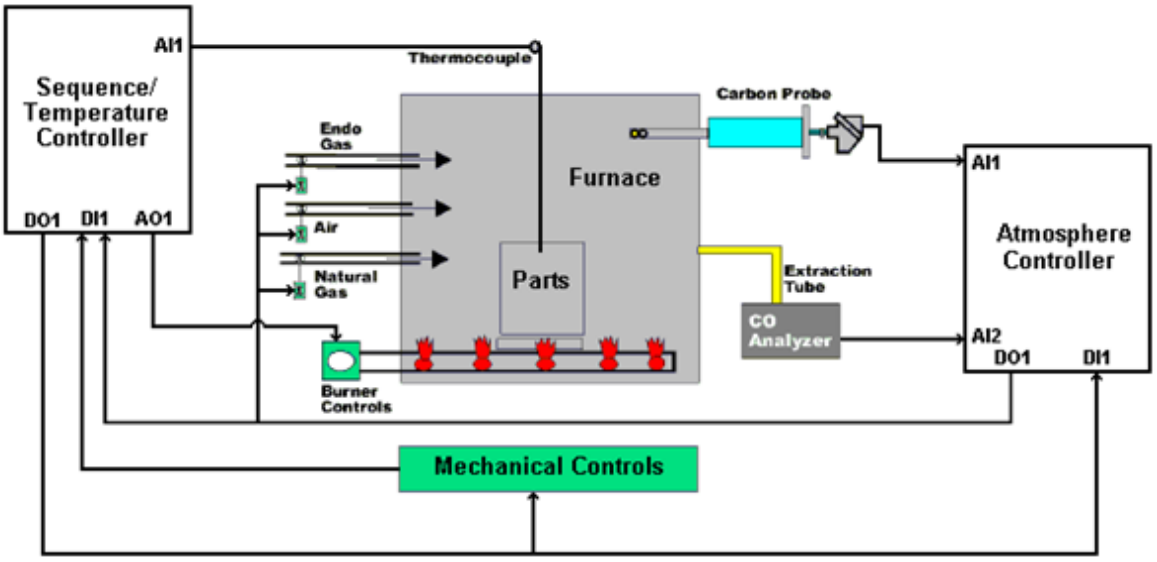

Figure 2. CarboNitriding Furnace Schematic Embedded control system.

2.1. CarboNitriding Heat-Treatment Furnace System

Furnace system based on CarboNitriding heat treatment process [41] has been developed to harden the surface of steel parts to enable them to resist surface wear. Since most steel components need to be produced soft to allow them to be machined, the case hardening technique is normally applied after the part has been formed, and before final machining takes place. The parts to be hardened are heated to a critical temperature in a furnace with a carefully controlled, artificially created atmosphere that is rich in nitrogen (from endogas – ammonia) and carbon (from methane) mixed with a suitable carrier gas (in this case air). The parts are left at this critical temperature for a required amount of time before being quenched in hot oil to achieve the full case hardening. A fully representative HIL simulation of such a process has been developed to evaluate different approaches to the development of a suitable control system [42]. Figure 2 shows a general schematic of the process.

2.2. Embedded Control System

The furnace control system consists of three main elements:

- The sequence controller
- The temperature controller
- The atmosphere controller

The sequence controller mainly controls the sequence of opening, loading and starting the furnace for a particular batch of parts. The temperature controller reads the signal from a processed thermocouple, and controls the state of a gas burner to regulate the temperature to a given set point. The atmosphere controller directly controls the composition of the furnace atmosphere, by controlling the gas inlet valves based on readings from one or more CO analysis devices. In the present study, the performance of the temperature controller is examined, both before and after the data is masked (processed) for reducing jitter. Hereby, the temperature controller is distributed over two nodes: Master and Slave. The Master node samples and filters the temperature reading. This information is then transmitted to the Slave, along with the elapsed time, which allows the Slave to calculate the desired set-point, apply negative feedback and calculate the required burner setting. The Slave then applies this signal to the proportional gas valve.

2.3. Temperature Controller

Due to the fact that the furnace chamber is relatively small and the furnace is directly gas fired, the dynamics can be considered to be quite quick and responsive. The burners are able to heat the furnace to a maximum temperature of 1250°C when fully ignited. The burner proportional valves are to be controlled by a 5V signal from the control system. The process can be represented by a "first order plus time delay" (FOPTD) model, which is given by:

$$G_p(s) = \frac{250\, e^{-2s}}{1+6s} \tag{1}$$

From this process model, the "integral of time-multiplied absolute error" (ITAE) set-point criterion was used to select gains for a parallel PID controller [42]. The open-loop gain crossover frequency of the compensated plant is 0.1156 Hz. To discretize the controller, a limit was calculated for the sampling frequency so as to introduce no more than approximately 1° extra phase shift due to the sampling process; that is $f_s \geq 180 \times 0.1156 = 20.8$ Hz.

In the Master node, a digital anti-aliasing filter was implemented with a cut-off frequency of 62.8 rad/s, using a sampling frequency of 100 Hz, to reduce the effects of noise and high-frequency components in the input signal. The representative, realistic hardware implementation of the temperature control system is shown in Figure 3. Thus, from the figure, it can be seen that any jitter in the Slave tasks may detriment the quality of the temperature control. The following section describes the metrics that were used to assess this quality measure.

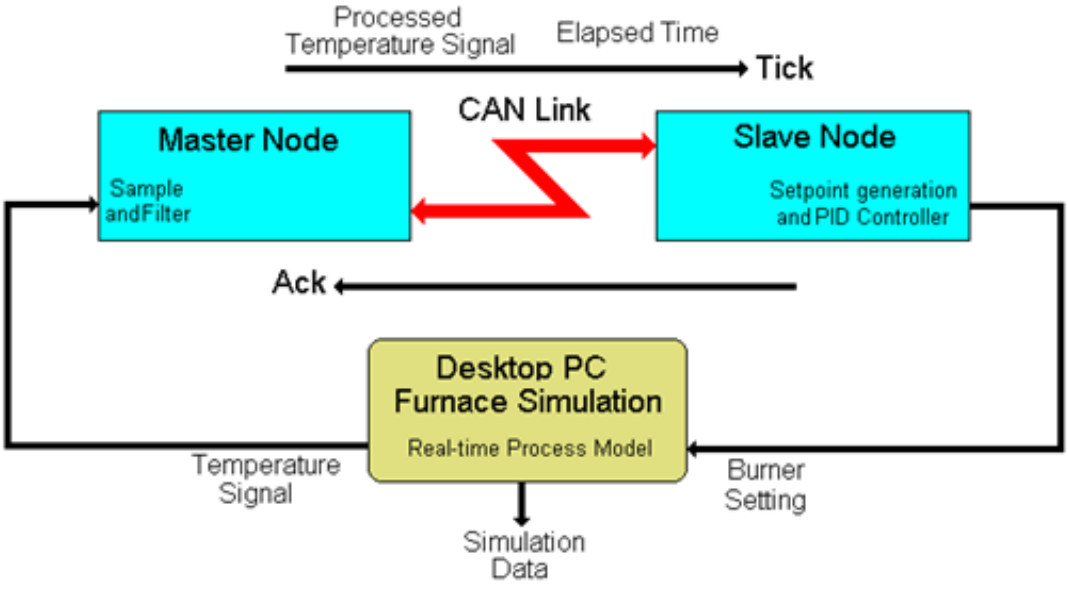

Figure 3. Temperature control system implementation.

3. Experimental Methodology

The system setup in this study has one Master node and one Slave node, both are based on Infineon C167-CS boards (running at 1 Mbit/s baudrate). The Master processor performs two tasks: timing task and sampling task. In the sampling task, the controller reads the signal from a processed thermocouple. In the "interrupt service routine" (ISR) [43] and through the tick message, Master sends the elapsed time and the sampled temperature over the CAN bus. A 5 ms tick interval is used; the sampling task is therefore executed every 2 ticks, and the actuation task is executed every 10 ticks. After calculating the control output, the Slave applies this as a 10-bit input into the simulation model. In all models considered here, six bytes are used for 'real' data (the first byte contained the Slave ID; see [18] for further details). Note that one additional bit is required for "Nolte C" coding and two additional bits were required for SBS coding (this is explained in detail in [5]).

To measure the performance of both controller implementations, ITAE of the system is measured in response to a 60°C change of set-point. The ITAE is defined mathematically in (2). The error signal $e(t)$ represents the difference between the measured temperature and the reference at each time instant t, over the period of the test duration T, which is equal to 300 seconds:

$$ITAE = \int_0^T |e(t)| \cdot t \, dt \qquad (2)$$

In addition, to gauge the disturbance rejection performance of the system, the "integral of absolute error" (IAE) of the system is measured after injecting a 10°C disturbance in the process output at t = 20 seconds, after allowing the controller to settle into a steady-state at a given set-point.

The jitter for both the compensated and uncompensated systems (i.e. before and after "Nolte C" and SBS are applied) is also measured. To obtain data regarding real-time stability, the latency between Master and Slave clock ticks is recorded for a period of 10,000 samples for each system. To make these measurements, a pin on the Master node is set high (for a short period) at the start of the Master ISR function. Another pin on the Slave (initially high) is set low at the start of Slave ISR function. The signals from these two pins are then AND-ed (using a 74LS08N chip from Texas Instruments), to give a pulse stream with widths that represent the transmission delays. These widths are measured using a National Instruments data acquisition card 'NI PCI-6035E' [44], used in conjunction with appropriate software LabVIEW 7 [45].

Note that the jitter values presented here reflect the impact of CAN bit-stuffing only (since the scheduler software is designed to have no jitter [46]). Two values are recorded for jitter: the *average jitter* and the *difference jitter*. The average jitter is taken as the standard deviation of the total latency of the entire sample range. As the standard deviation increases the jitter level increases. The difference jitter is taken as the difference between the worst-case and the best-case latency values of the entire sample range (this jitter is sometimes referred to as *absolute jitter*: see [47]).

4. Results from the Furnace System

Before the timing and performance experiments were carried out, a small experiment was done to select the optimal values of the K_P, K_I and K_D gains for the PID controller (implemented on the Slave node). The experiment was based on running the system with all possible combinations of some suggested gains. These gains were:

$$K_P = 2.4552 \pm 10\%$$
$$K_I = 0.3057 \pm 10\%$$
$$K_D = 1.6404 \pm 10\%.$$

Note that the original values were calculated based on the ITAE tuning criteria for an FOPDT process model, adapted from [42]. The best system performance was obtained when the following gains were used: K_P = 2.20968, K_I = 0.33627 and K_D = 1.80444. Table 1 summarizes the basic timing results obtained from this furnace system.

Table 1. Basic timing results from the furnace study (all in µs)

Test	Without Masking	With "Nolte C"	With SBS
Min transmission time	164.5	171	177.2
Max transmission time	173	177.8	182
Average transmission time	167.2	173.6	179.5
Difference Jitter	8.5	6.8	4.8
Average Jitter	1.2	1.1	0.9

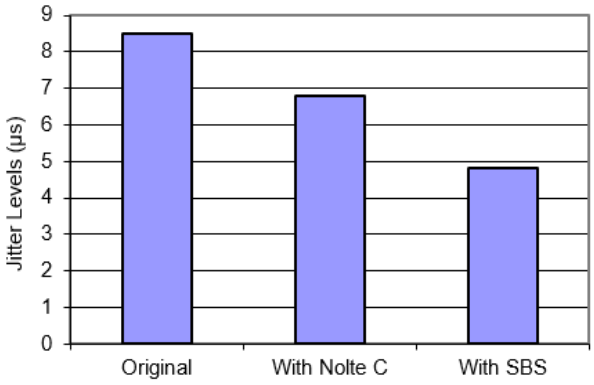

Figure 4. Difference jitter levels before and after encoding the furnace system.

The results presented in the table show that jitter levels have been improved as an effect of applying "Nolte C" and SBS on the transmitted data between the sampling (Master) node and the actuating (Slave) node in the furnace system considered. For comparison purposes, Figure 4 summarizes the difference jitter levels with and without masking. If the difference jitter is to be looked at, almost the same rate of jitter reduction has been found here as compared to that obtained with random data using the same processor hardware [5]. However, average jitter improvement level has been somewhat reduced. The

reason for this is that there are no large changes in the transmitted information (e.g. sampled temperature) after the first few seconds of the system run, which means that there is less opportunity for average jitter reduction. This simply implies that for each application, results would entirely depend on the characteristics of the data exchanged between the Master and Slave nodes.

After observing the jitter enhancement, it would only make sense to consider such techniques in real applications if we see a significant improvement in their control performance. The performance measurements of the furnace system are presented in Table 2 and Table 3.

Table 2. Impact of "Nolte C" and SBS on the observed behaviour in the furnace study (without disturbance)

Version	ITAE	Improvement % ITAE	IAE	Improvement % IAE
Original	2616.2983	0	458.4764	0
"Nolte C"	2616.7705	-0.0180	458.4697	0.0015
SBS	2578.8867	1.4299	454.6107	0.8432

Table 3. Impact of "Nolte C" and SBS on the observed behaviour in the furnace study (with disturbance)

Version	ITAE	Improvement % ITAE	IAE	Improvement % IAE
Original	239.1617	0	51.6693	0
"Nolte C"	238.9135	0.1038	51.6055	0.1235
SBS	159.8445	33.1647	43.6295	15.5601

The table shows that there is no measurable improvement in the system performance as a direct application of "Nolte C" jitter-reduction technique. When SBS was applied, the improvements in the ITAE and IAE measures, without a disturbance, were about 1.4% and 0.8%, respectively. The situation was seen better when considering a 10°C disturbance. In this case, around 33% and 16% improvements were achieved for the ITAE and IAE measures, respectively. This is best explained by observing that a control system will respond differently to set-point changes than it will for disturbances, due to the topology of the feedback scheme and the source of the disturbing signal.

5. Conclusions

The main aim of this paper was to explore the impact of two data coding algorithms developed previously for jitter reduction purposes on the performance of real-time applications. Therefore, both "Nolte C" (i.e. selective byte-based XOR masking) and SBS techniques were applied here using an HIL testbed facility. The case study considered was the CarboNitriding heat-treatment furnace system consisting of two communicating nodes where the nodes were connected physically via CAN hardware network protocol.

In addition to the timing measurements, the impact of "Nolte C" and SBS methods on the performance of the furnace system was studied using quantitative measures which are directly related to the control behaviour of the system. Overall, from the results obtained, the SBS technique had higher potential to improve the system performance by significantly reducing the impact of bit-stuffing in CAN hardware, especially with the case of disturbance rejection that is required in many control applications. This implies that despite that the SBS method is based on a simple idea and that it does not require high computational and memory requirements [19], its impact on real application can still be seen significant especially when it is considered for time- and / or safety-critical embedded systems.

It would then be recommended to test the effectiveness of such data coding techniques on a wider range of control applications to verify their usefulness in this area of studies. Moreover, alternative, more recent data coding techniques (such as those referred to in the literature review) can also be tested to demonstrate their effectiveness in real-time resource-constrained embedded control applications.

Acknowledgements

Author would like to thank Dr Michael Pont (SafeTTy Systems Ltd, UK) for providing useful comments on the work carried out in this paper. The HIL simulation model used here was developed by Dr Michael Short (Teesside University, UK) to whom the author is grateful.

References

[1] M. Farsi and M. B. M. Barbosa, *CANopen implementation: applications to industrial networks*. Baldock, Hertfordshire, England; Philadelphia, PA: Research Studies Press, 1999.

[2] M. D. Natale, H. Zeng, P. Giusto, and A. Ghosal, *Understanding and Using the Controller Area Network Communication Protocol: Theory and Practice*. Springer Science & Business Media, 2012.

[3] Bosch, *CAN Specification Version 2.0*. Bosch, 1991.

[4] R. Obermaisser, *Time-Triggered Communication*. CRC Press, 2011.

[5] M. Nahas, M. J. Pont, and M. Short, "Reducing message-length variations in resource-constrained embedded systems implemented using the Controller Area Network (CAN) protocol," *Journal of Systems Architecture*, May 2009, vol. 55, no. 5-6, pp. 344-354.

[6] F. Abugchem, M. Short, and D. Xu, "An experimental HIL study on the jitter sensitivity of an adaptive control system," in *2013 IEEE 18th Conference on Emerging Technologies Factory Automation (ETFA)*, 2013, pp. 1-8.

[7] F. Cottet and L. David, "A Solution to the Time Jitter Removal in Deadline Based Scheduling of Real-time Applications," presented at the 5th IEEE Real-Time Technology and Applications Symposium - WIP, Vancouver, Canada, 1999, pp. 33-38.

[8] Q. Huynh-Thu and M. Ghanbari, "Impact of jitter and jerkiness on perceived video quality," in *Proc. Workshop on Video Processing and Quality Metrics*, 2006.

[9] A. J. Jerri, "The Shannon sampling theorem #8212; Its various extensions and applications: A tutorial review," *Proceedings of the IEEE*, Nov. 1977, vol. 65, no. 11, pp. 1565-1596.

[10] P. Marti, J. M. Fuertes, G. Fohler, and K. Ramamritham, "Jitter compensation for real-time control systems," in *22nd IEEE Real-Time Systems Symposium, 2001. (RTSS 2001). Proceedings*, 2001, pp. 39-48.

[11] M. Nahas, M. Short, and M. J. Pont, "The impact of bit stuffing on the real-time performance of a distributed control system," presented at the Proceeding of the 10th International CAN conference iCC, Rome, Italy, 2005, pp. 10-1-10-7.

[12] F. M. Proctor and W. P. Shackleford, "Real-time operating system timing jitter and its impact on motor control," in *Intelligent Systems and Advanced Manufacturing*, 2001, pp. 10016.

[13] F. Smirnov, M. Gla\s s, F. Reimann, and J. Teich, "Formal reliability analysis of switched Ethernet automotive networks

under transient transmission errors," in *Design Automation Conference (DAC), 2016 53nd ACM/EDAC/IEEE*, 2016, pp. 1-6.

[14] F.-Y. Wu and Y.-M. Chen, "Impact of PWM Duty Cycle Jitter on Switching-Mode Power Converter Efficiency," *IEEE Transactions on Power Electronics*, 2017.

[15] T. Nolte, H. Hansson, and C. Norstrom, "Minimizing CAN response-time jitter by message manipulation," in *Eighth IEEE Real-Time and Embedded Technology and Applications Symposium, 2002. Proceedings*, 2002, pp. 197-206.

[16] T. Nolte, H. Hansson, C. Norström, and S. Punnekkat, "Using bit-stuffing distributions in CAN analysis," presented at the IEEE Real-Time Embedded Systems Workshop, London, 2001.

[17] M. Nahas and M. J. Pont, "Using XOR operations to reduce variations in the transmission time of CAN messages: A pilot study," in *Proceedings of the Second UK Embedded Forum*, Birmingham, UK, 2005, pp. 4-17.

[18] M. J. Pont, *Patterns for time-triggered embedded systems: building reliable applications with the 8051 family of microcontrollers*. Harlow: Addison-Wesley, 2001.

[19] M. Nahas, "Applying Eight-to-Eleven Modulation to reduce message-length variations in distributed embedded systems using the Controller Area Network (CAN) protocol," *Canadian Journal on Electrical and Electronics Engineering*, vol. 2, no. 7, pp. 282-293, 2011.

[20] G. Cena, I. C. Bertolotti, T. Hu, and A. Valenzano, "A mechanism to prevent stuff bits in CAN for achieving jitterless communication," *IEEE Transactions on Industrial Informatics*, 2015, vol. 11, no. 1, pp. 83-93.

[21] M. M. Hassan, "Third Bit Complement (TBC) Mechanism to Reduce Bit Stuffing Jitter in Controller Area Network (CAN)," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering (An ISO 3297: 2007 Certified Organization)*, vol. 4, no. 5, 2015.

[22] T. R. Jena, A. K. Swain, and K. Mahapatra, "A novel bit stuffing technique for Controller Area Network (CAN) protocol," in *Advances in Energy Conversion Technologies (ICAECT), 2014 International Conference on*, 2014, pp. 113-117.

[23] S. K. Kabilesh and B. V. Kumar, "Design and simulation of modified selective XOR algorithm for payload attrition in CAN," in *2016 3rd International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2016, vol. 1, pp. 1-6.

[24] H. J. Lad and V. G. Joshi, "Hybrid message conversion technique to reduce jitter in CAN based distributed embedded system," in *Emerging Technology Trends in Electronics, Communication and Networking (ET2ECN), 2014 2nd International Conference on*, 2014, pp. 1-6.

[25] K. R. Priyanga, K. Venkatesan, and others, "A fixed length payload encoding for can," *International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)*, 2014, vol. 3, no. 12.

[26] G. Cena, I. C. Bertolotti, T. Hu, and A. Valenzano, "Fixed-length payload encoding for low-jitter controller area network communication," *IEEE Transactions on Industrial Informatics*, 2013, vol. 9, no. 4, pp. 2155-2164.

[27] D. Ayavoo, "The development of reliable x-by-wire systems: assessingt he effectiveness of a'simulation first'approach," PhD Thesis, University of Leicester, 2006.

[28] M. Short and M. J. Pont, "Fault-Tolerant Time-Triggered Communication Using CAN," *IEEE Transactions on Industrial Informatics*, May 2007, vol. 3, no. 2, pp. 131-142.

[29] M. Bacic, "On hardware-in-the-loop simulation," in *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, 2005, pp. 3194-3198.

[30] M. Schlager, *Hardware-in-the-Loop Simulation*. Omniscriptum Gmbh & Company Kg., 2008.

[31] I. Briggs, M. Murtagh, R. Kee, G. McCulloug, and R. Douglas, "Sustainable non-automotive vehicles: The simulation challenges," *Renewable and Sustainable Energy Reviews*, 2017, vol. 68, pp. 840-851.

[32] K. Enisz, D. Fodor, I. Szalay, and L. Kovacs, "Reconfigurable real-time hardware-in-the-loop environment for automotive electronic control unit testing and verification," *IEEE Instrumentation & Measurement Magazine*, vol. 17, no. 4, pp. 31-36, 2014.

[33] V. Josef, G. Robert, K. Petr, L. František, and M. Karel, "Hardware-In-the-Loop simulation for automotive parking assistant control units," in *Mechatronics-Mechatronika (ME), 2014 16th International Conference on*, 2014, pp. 325-330.

[34] M. Kloc, R. Weigel, and A. Koelpin, "Making real-time hardware-in-the-loop testing of automotive electronic control units wireless," in *Advanced Technologies for Communications (ATC), 2016 International Conference on*, 2016, pp. 407-412.

[35] J. Schroeder, C. Berger, and T. Herpel, "Challenges from integration testing using interconnected hardware-in-the-loop test rigs at an automotive oem: An industrial experience report," in *Proceedings of the First International Workshop on Automotive Software Architecture*, 2015, pp. 39-42.

[36] M. Short and M. J. Pont, "Assessment of high-integrity embedded automotive control systems using hardware in the loop simulation," *Journal of Systems and Software*, Jul. 2008, vol. 81, no. 7, pp. 1163-1183.

[37] Z.-G. Zhao, L.-J. Zhou, J.-T. Zhang, Q. Zhu, and J.-K. Hedrick, "Distributed and self-adaptive vehicle speed estimation in the composite braking case for four-wheel drive hybrid electric car," *Vehicle System Dynamics*, pp. 1-24, 2017.

[38] M. Nahas, "Developing a Novel Shared-Clock Scheduling Protocol for Highly-Predictable Distributed Real-Time Embedded Systems," *American Journal of Intelligent Systems*, Dec. 2012, vol. 2, no. 5, pp. 118-128.

[39] M. Nahas, "Implementation of highly-predictable time-triggered cooperative scheduler using simple super loop architecture," *International Journal of Electrical & Computer Sciences*, 2011, vol. 11, pp. 33-38.

[40] M. Nahas, "Employing Two 'Sandwich Delay' Mechanisms to Enhance Predictability of Embedded Systems Which Use Time-Triggered Co-Operative Architectures," *Journal of Software Engineering and Applications*, 2011, vol. 4, no. 7, pp. 417-425.

[41] W. Joseph M, "Control arrangement for controlled atmosphere furnace," US3237928 A, 01-Mar-1966.

[42] D. E. Seborg, D. A. Mellichamp, T. F. Edgar, and F. J. Doyle III, *Process dynamics and control*. John Wiley & Sons, 2010.

[43] M. Nahas and A. M. Nahhas, *Ways for implementing highly-predictable embedded systems using time-triggered co-operative (TTC) architectures*. INTECH Open Access Publisher, 2012.

[44] "Datasheet PDF Template - 4daqsc202-204_ETC_212-213.pdf." [Online]. Available: http://www.ni.com/pdf/products/us/4daqsc202-204_ETC_212-213.pdf. [Accessed: 30-Aug-2017].

[45] "LabVIEW System Design Software - National Instruments." [Online]. Available: http://www.ni.com/labview/. [Accessed: 30-Aug -2017].

[46] M. Nahas, M. J. Pont, and A. Jain, "Reducing task jitter in shared-clock embedded systems using CAN," in *Proceedings of the UK Embedded Forum 2004*, Birmingham, UK, 2004, pp. 184-194.

[47] G. C. Buttazzo, *Hard real-time computing systems: predictable scheduling algorithms and applications*. New York: Springer, 2005.