IJICTR
International Journal of Information & Communication Technology Research

Volume 13- Number 4 – 2021 (8 -17)

Real-Timeness Improvement of CAN-based Industrial Networks Based on Criticality Level

Ismail Ghodsollahee
Dependable Distributed Embedded Systems (DDEmS) Laboratory,
Department of Computer Engineering,
Ferdowsi University of Mashhad,
Mashhad, Iran.
esmailelahee@gmail.com

Yasser Sedaghat*
Dependable Distributed Embedded Systems (DDEmS) Laboratory,
Department of Computer Engineering,
Ferdowsi University of Mashhad,
Mashhad, Iran.
y_sedaghat@um.ac.ir

Received: 12 August 2021 - Accepted: 25 October 2021

Abstract—**Although applying new Internet-based communication technologies on industrial physical processes made great improvements in factory automation, there are still many challenges to meet the response time and reliability requirements of industrial communications. These challenges resulted from strict real-time requirements of industrial control system communications which are performed in harsh environments. The controller area network (CAN) communication protocol is commonly employed to deal with these challenges. However, in this protocol, even message retransmission requests of a faulty node can lead to timing failures. In this paper, to control the behavior of nodes, message retransmission is performed based on the criticality level of message reception. The proposed method, called MRMC+, improves the real-time behavior of a CAN bus in terms of response time by an average of 36.32% and 18.02%, respectively, compared to the standard CAN and WCTER-based approaches.**

Keywords: Controller Area Network; Reliability; Real-Timeness; Criticality Level; Consumed Bandwidth Reduction.

Article type: Research Article

Publisher: ICT Research Institute

* Corresponding Author

[DOI: 10.52547/itrc.13.4.8]

I. INTRODUCTION

Following the collapse of financial systems, many countries have focused on recovering their industries and improving the efficiency of industrial processes employing information and communication technologies (ICT). China, for example, has introduced the "Made in China 2025" strategy to establish a high-level industrial internet dialogue between the Chinese and German governments to accelerate the industrial process between the two countries [1].

The concept of the industrial Internet was first presented in 2012 [2]. Industrial Internet or Industrial Internet of things (IIoT) focuses on the Internet technologies application in industrial networks [3]. Industrial networks are a combination of distributed computer and embedded system networks utilized in automation and control systems [4]. The authors of reference [5] describe industrial networks by an industrial pyramid. As shown in Fig. 1, industrial networks are classified into three levels including, management, cell, and field levels.

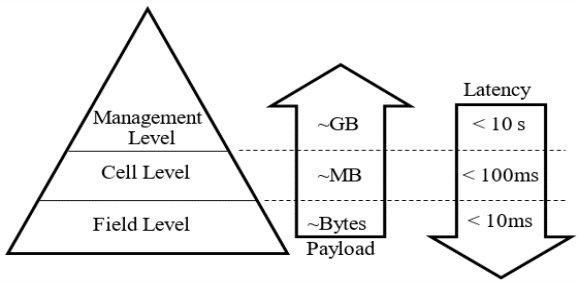


Fig. 1. Industrial Pyramid [5]

The management level consists of typical IT infrastructures, and the cell level consists of PLCs and industrial computers. In these two levels, a large amount of data is exchanged between network elements without strict real-time requirements. In contrast, the field-level includes closed-loop communications between controllers, actuators, and sensors of the physical process that require tight latency and high reliability. At this level, Fieldbus communication protocols are traditionally utilized. One of the Fieldbus communication protocols employed in industrial networks with a 20 % market share of the Fieldbus networks is the CAN communication protocol [6].

The CAN communication protocol was designed in the 1980s by Robert Bosch for intra-vehicular-networks. This communication protocol is widely employed in internal vehicular networks due to its low implementation cost and its fault-tolerant behavior against network errors [7]. Today, this communication protocol is employed in industrial fields and IIoT in addition to intra-vehicular-networks [8], [9].

As mentioned, IIoT systems are safety-critical [10] and have error handling and strict real-time requirements [11], [12]. Although the deterministic nature and fault-tolerant behavior of CAN communication protocol meet the industrial field-level requirements, it can increase the worst-case response time (WCRT) of the IIoT systems due to its error handling mechanisms. Since high WCRT violates the real-time requirements of the safety-critical systems [13], in this paper, the MRMC+ method is presented to improve the error handling mechanism of CAN.

The CAN communication protocol considers an error-handling mechanism to handle five types of errors, including Bit errors, Stuff errors, Cyclic Redundancy Check (CRC) errors, Form errors, and Acknowledgement errors. Among these errors, the CRC errors and Acknowledgement errors are dependent on the receiver node computations. CRC errors are diagnosed by detecting the inconsistency of the received and computed CRC in the receiver nodes. Any receiver node that does not diagnose this inconsistency, sends reception acknowledgment to the sender node through transmitting the dominant bit in the ACK slot that previously leaves recessive by the sender node. As shown in Fig. 2, the ACK field consists of the ACK slot and ACK diameter. If the ACK slot is not changed to dominant, incorrect message transmission is detected by sender node about all receiver nodes. In this case, the sender node detects acknowledgment error and retransmit the unacknowledged frame.

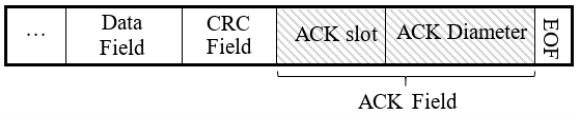


Fig. 2. CAN ACK field format

Although the error-handling mechanism in the CAN communication protocol notifies the sender node about the error that occurred during message transmission, the sender node will not be aware of which nodes received the message incorrectly. The sender node, in response to the occurred error, retransmits its message. Retransmissions are performed even if a non-critical node detects an error. For example, in Fig. 3, although node N1 wants to transmit its message, a faulty node prevents message propagation and propagates error flags. In this situation, node N3 with critical tasks couldn't receive its desired frame, and node N1 retransmits its message after error frame propagation. As can be seen, due to an erroneous node with non-critical tasks, node N3 violates its critical tasks deadlines.

In this paper, to deal with the destructive behavior of faulty nodes, the MRMC+ method is presented, which is an extension of our previous paper [14]. In this method, receiving nodes propagate the error based on the critical level of the message reception.

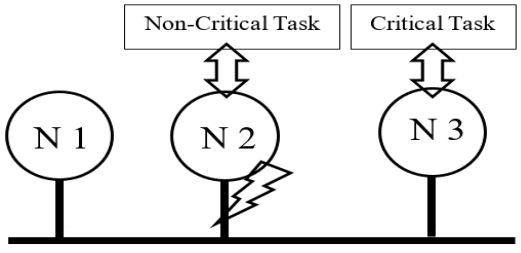


Fig. 3. Tasks with different criticality levels

[DOI: 10.52547/itrc.13.4.8]

II. Related Work

Industrial networks are safety-critical systems based on deterministic bus technology. Meeting the exact real-time requirements of these networks is a growing necessity [15]. As a field-level industrial communication protocol, many real-time improvement methods [16-31] have been proposed for the CAN communication protocol. These methods can be classified as shown in Fig. 4. According to this classification, these methods can be divided into three categories including, hardware-based, software-based, and hybrid.

Hardware-based methods include net guardian methods [16–18] and topology change methods [19–21]. Network Guardian (NG) is commonly employed to prevent babbling idiot failures caused by faulty nodes [16–18]. Boster and Burn [16] present a bus guardian (BG) for event-trigger networks. This BG utilizes the node's message transmission times to compute the node's future propagation windows. Employing these windows avoids faulty nodes from transmission on the communication bus. Similarly, in [17], a simple bus guardian for the FlexCAN network is introduced. The FlexCAN is a CAN protocol extension that focuses on hardware redundancy. This BG employs propagation intervals of nodes, diagnoses offending nodes, and utilizes an OR Gate to control the message propagation ability of the nodes. Although utilization of the NG prevents the high bandwidth overhead caused by faulty nodes, the level of babbling that NG prevents the node from sending a message to the network is the same for all frames. For this reason, an analysis of the Guardian-based approach is presented in [18]. In this approach, the message retransmission number is determined based on the criticality level of the messages.

In addition to NG methods that prevent high traffic consumption due to faulty nodes, other hardware-based methods [19-21] prevent fault propagation from one subnet to the others by changing the linear topology of the CAN network. In [19], by changing the topology of the CAN network, the RedCAN method is presented. The RedCAN method pairs several bus sections into a broadcast bus. In RedCAN, after detecting physical defects in one sector to prevent fault propagation, other nodes disconnect this sector and employ a redundant link. These sections enable RedCAN to prevent fault propagation after detecting physical defects in the broadcasting bus by disconnecting the connected node to the faulty sector and employing a redundant link.

Also, Barranco and Proenza [20] propose an active star topology, called CANcentrate. In CANcentrate, central hub prevents fault propagation. Although CANcentrate prevents communication network failure, its active star topology hub represents a single point of failure. As a result, they present replicated active star topology called ReCANcentrate, which is based on the hardware redundancy of the hub. In this method, each hub similar to the CANcentrate has mechanisms to detect faulty links and isolate them. Moreover, these hubs monitor each other through a dedicated link [21].

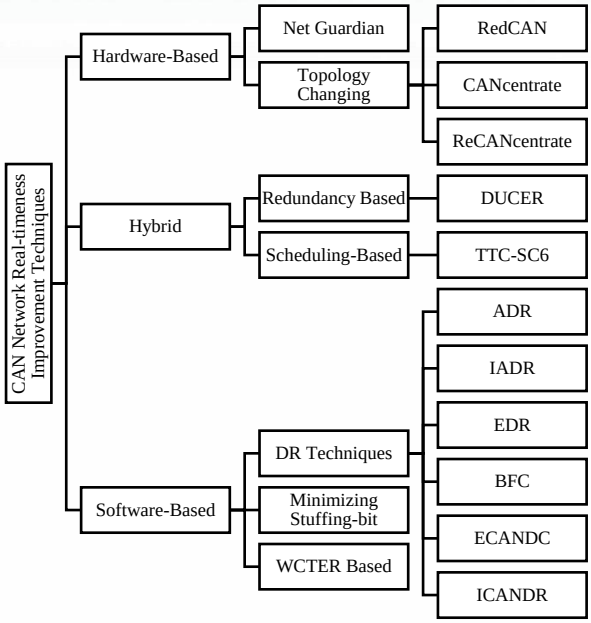


Fig. 4. Classification of Methods for Real-Timeness Improvement of CAN Network

In contrast to hardware-based methods, software-based real-timeness improvement methods [22-30] focus mainly on bit stuffing, data reduction (DR), and scheduling. In DR methods [22-29], the data is compressed through software algorithms before sending. These pre-transmission compression methods reduce bandwidth consumption and improve CAN network real-timeness. In DR methods, the similarity of the data of feature messages with specific IDs to their previous message is considered the main factor in the consumed bandwidth reduction. In [22], a simple DR method is presented, which is based on the detection of repeated bytes in consecutive frames. In this method, the compressed frame first byte is assigned to the data compression byte (DCB). Each bit of this byte indicates the replication of the corresponding byte in a consecutive frame.

Moreover, Ramtekand and Mahmud [23] improve compression ratio by presenting an adaptive DR (ADR) method. In the ADR method, since the DCB reduces the free bandwidth in the absence of replication between two consecutive frames, the sending of DCB is based on the possibility of reduction through the replication detection between two consecutive frames. Authors of reference [24], to improve the ADR method, introduce the Improved Adaptive DR (IADR) method. In this method, they consider three-level for each signal including, sending entire signals, sending signal differences, and not sending the signal. In this method, if the first bit of the compressed message is zero, it indicates that all signals are sent in their original form, and if this bit is one, each message signal is either sent in differential form or never sent.

In the IADR method, the first bit of compressed message is considered to dedicate compressed message from non-compressed ones, and also a byte to specify the compression status of different signals in the compressed message. Such overheads can increase the comprised message length over the original message. For this reason, the authors of references [25] present

[DOI: 10.52547/itrc.13.4.8]

the enhanced DR method. In this method, the compressed message length is measured before transmission. If the Compressed message length is longer than the original message itself, the original message is propagated.

In the Enhanced DR method, one-byte compression status indication increases compressed messages length, and compressed message length comparison before sanding decreases compression performance. In [26], the boundary of fifteen (BFC) method is presented to deal with these challenges. In this method, the compression overhead is reduced by placing the compression status bit before each signal and dividing signals into the boundary of fifteen and non-BFC, the compression performance increases. Two consecutive signals with a difference of fewer than 15 bits are in the BFC category. These signals are sent in compressed format because the compression overhead of these signals does not increase the length of compressed messages over original ones.

In the BFC method, signal compression is limited to the signal differences in Range +15 bits. Since this limitation affects the performance data reduction, Wu and Chung present the efficient CAN DR (ECANDR) method [27]. This method is based on compression area selection and signal rearrangement algorithms. In this method, each compressed signal is presented employing 9 bits. The most significant 8 bit is assigned to the difference value, and the remained bit represents the sign of the difference value. In this method, each '1' on the 8-bit header indicates that the corresponding signal is transmitted in differential form. Moreover, in reference [28], the improved CAN DR (ICANDR) method is proposed to improve the compression ratio of the ECANDR method. In the ICANDR method, instead of signal differences, the exclusive-or of two consecutive signals is transmitted. In addition, if all header bits are zero, they are not sent. In the ICANDR method, a rearrangement algorithm is employed that each combination of signal arrangement leads to different compression efficiency. Since in the ICANDR, this issue is not addressed, the authors of reference [29] present a multi-data arrangement method. In this method, the arrangement of signals is computed based on their magnitude and frequency.

In addition to DR methods, others improve the real-timeness by minimizing stuffing-bits. In CAN communication protocol, stuffing bits are employed to keep all nodes synchronized. Stuffing bits in the CAN are injected after five consecutive bits with the same polarity with opposite polarity. Although these stuffing bits synchronize all CAN nodes, in the worst case can generate 22% overhead [30]. For this reason, authors of reference [30] propose an XOR-based stuffing-bit minimization mechanism. In this mechanism first an XOR mask is initialized to "1010…", then one is assigned to the $1 + [log_2 m]$ most significant bits and zero is assigned to the $[log_2 m]$ bits of the XOR mask to prevent priority inversion.

Another category of real-time improvement methods is hybrid methods [7], [31]. In [7], the dual CRC error correction (DUCER) method is presented. In this method, in addition to using the hardware redundancy of the communication bus, the software error correction is prevented to prevent an erroneous message from retransmission. Error correction in DUCER is performed by comparing residual polynomials resulting from the division of the receiving message polynomials by the CRC generator polynomials. Moreover, in [31], a hybrid real-time improvement method named TTC-SC6 is proposed. In this method, in addition to hardware topology changing by applying star topology, a shared clock algorithm is presented that ensures faults on one link cannot affect the rest of the network. Furthermore, in our previous research paper [14], a hybrid method named MRMC was proposed that improve the real-timeness of the CAN Network through controlling the message retransmission based on the criticality of message reception. In this way, if the reception of an ongoing message is not critical for the faulty part of the network, the rest of the network accepts it as a correct one.

A comparison of the mentioned real-timeness improvement methods is shown in Table I. In this table, to evaluate real-timeness improvement compared to the standard CAN, parameters like response time, latency, transmission rate and bus load are considered. As can be seen in this table, data reduction methods [22-23, 25-27] improve the real-timeness of the CAN network in terms of latency. However, since fault occurrences are not considered in these methods, a faulty node can make these methods inefficient. Although hardware methods [17] and [19] prevent such destructive behavior, they only cover the destructive behavior of transmitter nodes. As a result, a faulty receiving node can violate the real-timeness improvement of these methods.

TABLE I. COMPARISON OF CAN REAL-TIMENESS IMPROVEMENT METHODS

Methods	Benchmark Tool	Evaluation Parameters	Improvement Percentage
FlexCAN [17]	SAE	Response Time	13.7
RedCAN [19]	RedCAN Simulation	Response Time	9.5
DR [22]	-	latency	5.7
ADR [23]	-	latency	7.5
EDR [25]	-	latency	8.25
ECANDR [27]	-	Transmittion rate	22
BFC [26]	-	Bus Load	25
MRMC [14]	NetCarBench	Response Time	20.82

III. PROPOSED METHOD

In our previous paper [14], MRMC-CAN method is presented, in which nodes transmit error flags based on the criticality of message reception. Although in this method non-critical message receptions can't interfere with critical message receptions, as in this method criticality level is not considered, low-critical tasks can lead to deadline violation of high-critical ones. To deal with these challenges in this paper MRMC+ method, an extended version of our previous research work [14], is presented. In the MRMC+ method, three different criticality levels including, non-critical, low-critical, and high-critical, are considered for message receptions.

Similar to MRMC-CAN, in MRMC+ receiver nodes should decide about error flag propagation. This

[DOI: 10.52547/itrc.13.4.8]

controlled error frame propagation makes it possible to retransmit a message just if it is critical and reduces the worst-case response time of the whole system. To give receiver nodes the control ability of their error frame propagation in each node, a list of high-critical and low-critical identifiers is defined. Receiver nodes based on these identifiers make decisions to transmit an error flag or not. This mentioned process is possible until no errors happen in the ID field of a transmitted frame. Therefore to make receiver nodes sure about the received ID, the ID field of the standard CAN communication protocol is changed as to Fig. 5. As seen in Fig. 5, the ID field of the MRMC+ method is broken into two sub-fields, including a reduced identifier (RID) and a CRC of RID (RID-CRC). The receiver node can use this newly defined ID field to verify the ID by comparing the received RID-CRC with the computed one.

To control the node's error frame transmission behavior, in addition to standard CAN modules three more modules named error frame transmission permission (EFTP), error flag counter (EFC), and criticality level detection (CLD) modules are considered. The EFTP Module is responsible for blocking error frames resulting from non-critical message receptions. The MRMC+ method control message retransmission by counting the number of consecutive error flags propagated for a specific ID and comparing it with a predefined error flag propagation threshold for this ID. The CLD module determines these thresholds by predefined criticality tables in each node. The proposed MRMC+ method block diagram is shown in Fig. 6. In MRMC+, the CLD module detects criticality levels of message receptions by comparing received message ID with a predefined critical ID (CID) list. An example of this list is shown in Table II. In this table, critical message reception IDs and their criticality level are defined. The response time improvement of the MRMC+ method over the MRMC-CAN is originated from considering the level of message reception criticality.

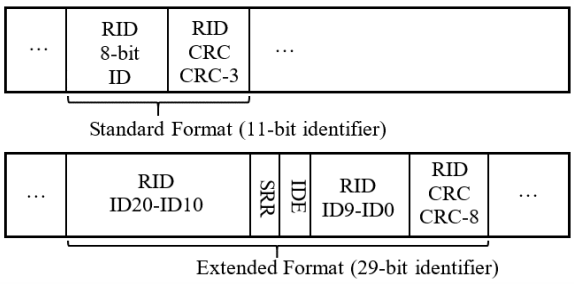


Fig. 5. MRMC+ ID Format

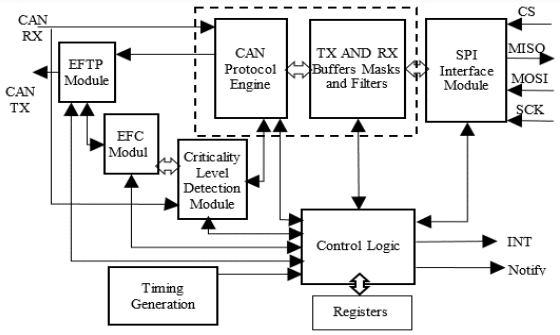


Fig. 6. Block Diagram of MRMC+

TABLE II. EXAMPLE OF THE CRITICAL ID LIST

Critical IDs	Levels of Criticality	Allowed Number of Error Flag Transmission (ANEFT)
738	High	-
248	Low	2
…	…	…

The CLD module first checks the CRC of the received message ID. Then, if the received CRC-RID is equal to the calculated one, it compares the received ID with the predefined critical ID list. If the received ID matches the CID list, the CLD module sets the criticality signal to high. Additionally, if this incoming message has a high critical reception level, this module sets the criticality level signal to high. The implementation of this module is shown in Fig. 7.

In the critical ID list of MRMC+, for low-priority messages, the allowed number of error frame transmission is defined. This feature enables the MRMC+ method to perform better behavior controlling of the receiving nodes. This feature in the block diagram of the proposed method, received by the EPC module. This module in each receiving node compares the allowed number of error frame transmissions for a received message ID and the current number of transmitted error flags. Then based on this comparison, it sets EF_THD signal to high if the current number of transmitted error flags exceeds the allowed number of error frames transmission.

After determining the critical level of the received message and detecting the exceedance of the current number of transmitted error flags from the allowed threshold, the transmission of the error flag is controlled by the ETP module. The implementation of this module is shown in Fig. 8. This module controls the error propagation of receiver nodes by disconnecting receiving nodes from the communication bus. In short, this node allows the receiver node to propagate the error frame in two ways:

1) If the received message has a high priority, it provides unlimited propagation possibility of the error frame for the receiving node;

2) If the received message has a low priority and the current number of transmitted error frames by the node exceeds the allowable limit or not, it allows the node to propagate the error frame.

Disconnecting an erroneous node that has received a non-critical message can be problematic. Because after 11 consecutive recessive bits, this node detects the

[DOI: 10.52547/itrc.13.4.8]

communication is idle and wants to start a new transmission. To deal with this challenge, in the MRMC+ method, a silent interval is considered in which this node should not send a message.

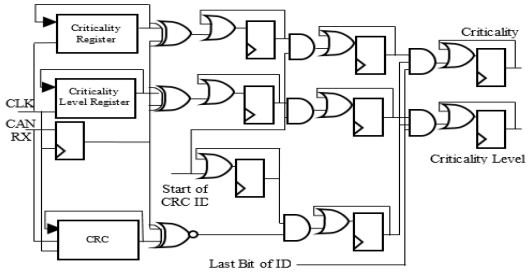


Fig. 7. Implementation of Criticality Level Detection

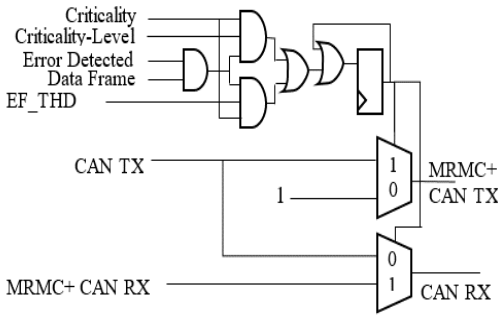


Fig. 8. Implementation of ETP Module

To further clarify the proposed method, consider the topology of Fig. 3 again assuming that, all nodes are equipped with the MRMC+ module. According to Fig. 9, if N1 sends a message on the network, N2 receives this message incorrectly due to its faulty communication link. Since the received message ID is not in the critical ID list of this node, MRMC+ modules prevent the node from sending an error frame. As a result, N3, for which receiving this message is critical, receives this message without error recovery and message retransmission latency. Moreover, as seen in this figure, a silence interval is considered for N2 that MRMC+ modules prevent this node from transmission until an ongoing message is successfully transmitted.

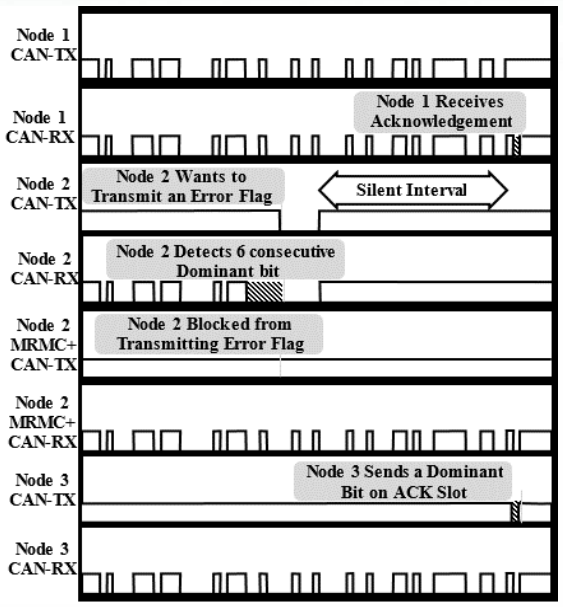


Fig. 9. Silence Interval

IV. Implementation and Evaluation

In this section, the implementation and evaluation of the proposed method are introduced separately. First, the fault injection is explained, then the benchmark method will be presented, and finally, three different case studies are evaluated.

A. Fault Injection

In this paper, the independent fault injection (IFI) method [32] is employed for performing fault injection. In IFI, faults are injected uniquely into each node using multiplexers. The multiplexer's utilization makes it possible to inject faults into the received and transmitted frames between CAN controllers and CAN transceivers of nodes. In our evaluations, since the proposed method seeks to control the behavior of the receiving nodes, the IFI method is modified like Fig. 10. As seen in this figure, the faults are injected only in the receiving path of nodes. The implementation of the modified IFI fault injection for the three nodes is shown in Fig. 10. As can be seen in this figure, 74HC153 and STM32F103C8T6 chips are employed to implement the IFI Controller and multiplexers.

B. Benchmark Messageset

Typically, response time evaluation is performed by utilizing a benchmark message set. In this paper, the NetCARBench tool [33] is employed to generate this kind of message set. This benchmark generates a set of benchmark messages based on user configuration of bandwidth, network load, and nodes number. The generated benchmark message set includes IDs, payloads, and periodicities of all nodes messages. Since the priority of receiving the message is not considered in NetCARBench, this tool is modified. A sample of the benchmark message generated for the three nodes is shown in Table III.

[DOI: 10.52547/itrc.13.4.8]

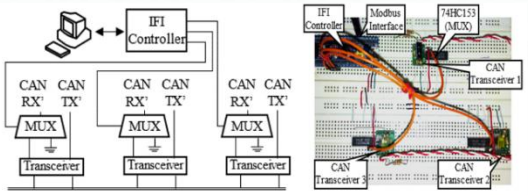


Fig. 10. Modified IFI Fault Injector

TABLE III. THE BENCHMARK MESSAGE SET

Node 1							
ID	payload	Period	Deadline	Criticality Level of Reception for nodes			
				2		3	
				Criticality	ANEFT	Criticality	ANEFT
251	8	20	20	High	∞	Non	0
538	4	100	5	Low	1	Non	0
242	4	20	5	High	∞	Non	0
Node 2							
ID	payload	Period	Deadline	Criticality Level of Reception for nodes			
				1		3	
				Criticality	ANEFT	Criticality	ANEFT
737	8	20	30	High	∞	Non	0
386	8	50	5	Low	2	Non	0
248	4	100	10	High	∞	Non	0
Node 3							
IDD	payload	Period	Deadline	Criticality Level of Reception for nodes			
				1		2	
				Criticality	ANEFT	Criticality	ANEFT
715	8	50	25	Non	0	Non	0

C. Testbed Setup

The schematic of the test board is shown in Fig. 11. As can be seen in this figure, for each node, the MRMC+ module, multiplexer, and transceiver are considered. Moreover, a logic analyzer is employed to monitor the behavior of the nodes. The implementation of the test board is shown in Fig. 12. The utilized components of this implementation are presented in Table IV.

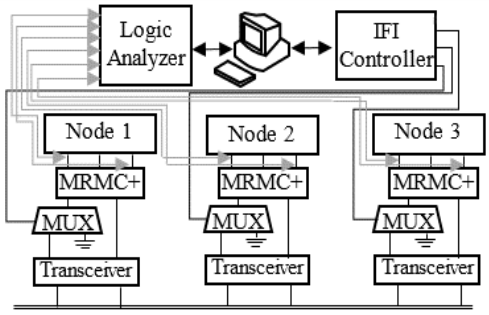


Fig. 11. Schematic of Testbed

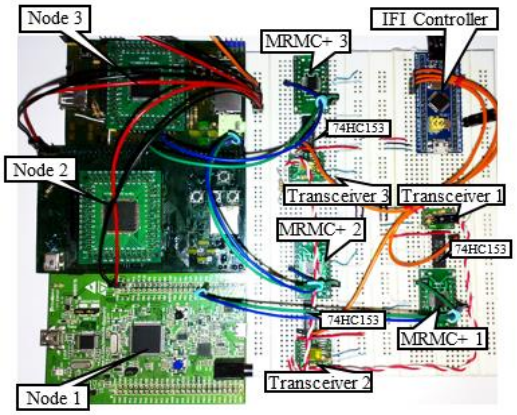


Fig. 12. Implementation of Testbed

TABLE IV. COMPONENT OF TESTBED

Component	Specification	Employed as
STM32F407	Arm Cortex M4	CAN Nodes
STM32F103	Arm Cortex M3	IFI Controller
STM32F030	Arm Cortex M0	MRMC+ Module
SN65HVD235	Transceiver	CAN Transceiver
74HC153	Multiplexer	Multiplexer

D. Case Studies

In this section, three case studies are analyzed to illustrate the real-time improvement of the MRMC+ method. In the first case study, IC faults are just injected into Node 3. As seen in Table III, Node 3 doesn't have any critical message reception. In this case, as shown in Fig. 13, MRMC+ and MRMC methods compared to standard CAN and WCTER-based approaches improve response time by an average of 38.60% and 20.85%, respectively.

In the second case study, faults are injected into node 2, which has mixed-criticality in the message reception. As shown in Fig. 14, the evaluation results of this case study show that the MRMC+ method improves WCRT by an average of 34.05%, 15.20%, and 10.39%, respectively, compared to the standard CAN, WCTER-based, and MRMC methods.

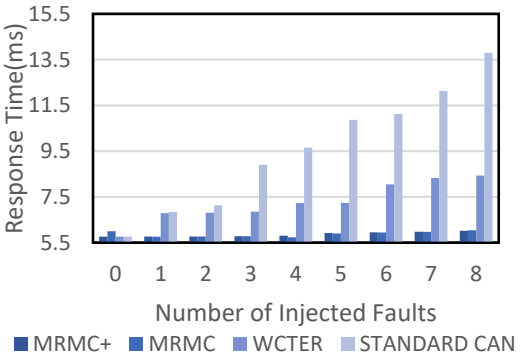


Fig. 13. Implementation of Testbed

[DOI: 10.52547/itrc.13.4.8]

Moreover, based on the evaluations made in these two case studies, the proposed method improves the real-timeness behaviors of a CAN bus in terms of response time by an average of 36.32% and 18.02%, respectively, compared to the standard CAN and WCTER-based approaches.

In the third case study, we evaluate the overhead of MRMC+ in the case of Device, ROM, and RAM usage. In the case of hardware implementation, the proposed method is implemented on Xilinx Spartan 6. The device utilization of hardware implementation is shown in Table V. Moreover, ROM and RAM usage of software-based MRMC+ is presented in Table VI.

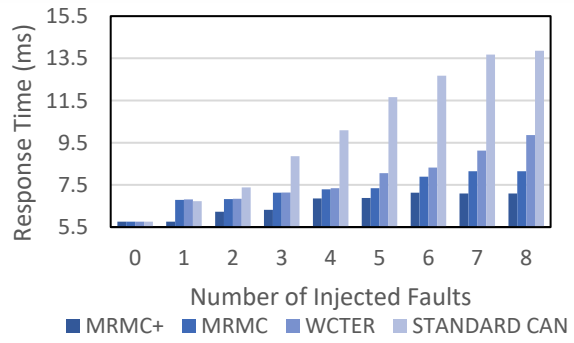


Fig. 14. Implementation of Testbed

TABLE V. AREA OVERHEAD OF MRMC-CAN MODULES FOR 6SLX9TQG144

	Number of Gates	Overheads (%)
Basic CAN Controller	20643	-
MRMC	111	0.5
MRMC+	184	0.8

TABLE VI. MRMC ROM, RAM USAGE FOR STM32F030F4

	MRMC	MRMC+
RAM Usage	0.7 %	2.3 %
ROM Usage	30.7 %	34.7 %

V. CONCLUSION AND FUTURE WORKS

Although many real-time improvement methods have been proposed for the CAN network so far, none of these methods consider the importance of receiving the message. As a result, in this paper, the MRMC + method is presented, which controls the error flag transmission based on the criticality of the message reception. However, the MRMC+ method improves the response time of the CAN network by an average of 36.32%, does not address the determination of the appropriate number of error flag transmissions for low criticality messages reception. In future research, we will further improve real-timeness by determining the appropriate number of error flags transmission for low-priority messages reception.

REFERENCES

[1] I. V. Jorge Posada, Carlos Toro, Inigo Barandiaran, David Oyarzun, Didier Stricker, Raffaele de Amicis, Eduardo B. Pinto, Peter Eisert, Jurgen Dollner, “Visual Computing as a Key Enabling Technology for Industrie 4.0 and Industrial Internet,” IEEE Comput. Graph. Appl., vol.vol.35, no. No02, pp. 26–40, 2015.

[2] P. C. Evans and M. Annunziata, “Industrial Internet: pushing the boundaries of minds and machines,” 2012[Online]. Available:www.ge.com/docs/chapters/Industrial_Internet.pdf

[3] R. Sousa, P. Pedreiras, and P. Goncalves, “Enabling IIoT IP backboneswith real-time guarantees,” inProc. IEEE 20th Conf. Emerg. Technol.Factory Autom., Luxembourg, Sep. 2015, pp. 1–6.

[4] M. Hankel and B. Rexroth, "The reference architectural model industrie 4.0 (rami 4.0)", ZVEI, vol. 2, no. 2, pp. 4, 2015.

[5] D. Cavalcanti, J. Perez-Ramirez, M. M. Rashid, J. Fang, M. Galeev and K. B. Stanton, "Extending Accurate Time Distribution and Timeliness Capabilities Over the Air to Enable Future Wireless Industrial Automation Systems," in Proceedings of the IEEE, vol. 107, no. 6, pp. 1132-1152, June 2019, doi: 10.1109/JPROC.2019.2903414.

[6] M. Vuković, D. Mazzei, S. Chessa and G. Fantoni, "Digital Twins in Industrial IoT: a survey of the state of the art and of relevant standards," 2021 IEEE International Conference on Communications Workshops (ICC Workshops), 2021, pp. 1-6, doi: 10.1109/ICCWorkshops50388.2021.9473889.

[7] L. Zhang, F. Yang, and Y. Lei, “Tree-based intermittent connection fault diagnosis for controller area network,” IEEE Trans. Veh. Technol., vol. 68, no. 9, pp. 9151–9161, Sep. 2019.

[8] H. Kimm and M. Jarrell, “Controller area network for fault tolerant small satellite system design,” in 2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE). IEEE, 2014, pp. 81–86.

[9] X. Jiang, M. Lora, and S. Chattopadhyay, "An Experimental Analysis of Security Vulnerabilities in Industrial IoT Devices," ACM Transactions on Internet Technology, 2020.

[10] J. Y. Guido Marchetto, Riccardo Sisto and A. Ksentini, “Formally verified latency-aware vnf placement in industrial internet of things,” in 14th IEEE International Workshop on Factory Communication Systems (WFCS), Imperia, Italy, 2018.

[11] S. Saadaoui, A. Khalil, M. Tabaa1, M. Chehaitly, F. Monteiro and A. Dandache, “Improved many to one architecture based on discrete wavelet packet transform for industrial IoT applications using channel coding,” Springer, Journal of Ambient Intelligence and Humanized Computing, vol. 11, no. 12, Dec. 2020.

[12] B. Chen and J. Wan, "Emerging trends of ml-based intelligent services for industrial internet of things (iiot)", In Proc. 2019 IEEE Computing, Communications and IoT Applications (ComComAp), 2019.

[13] H. Kong, J. Cheng, K. Narayanan and J. Hu, "DUCER: a Fast and Lightweight Error Correction Scheme for In-Vehicle Network Communication", 2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES), 2018.

[14] I. Ghodsollahee and Y. Sedaghat, "MRMC-CAN: A Method to Improve Real-Timeness and Response Time of CAN," 2021 5th International Conference on Internet of Things and Applications (IoT), 2021, pp. 1-8, doi: 10.1109/IoT52625.2021.9469716.

[15] R. Zhohov, D. Minovski, P. Johansson, and K. Andersson, “Real-time performance evaluation of LTE for IIoT,” inProc. IEEE 43rd Conf.Local Comput. Netw. (LCN), 2018, pp. 623–631.

[16] I. Broster and A. Burns, "An analysable bus-guardian for event-triggered communication", Proc. 24th IEEE Real-Time Systems Symp. (RTSS'03), pp. 410-419, 2003.

[DOI: 10.52547/itrc.13.4.8]

[17] G. Buja, J. R. Pimentel and A. Zuccollo, "Overcoming babbling-idiot failures in CAN networks: A simple and effective bus guardian solution for the FlexCAN architecture", IEEE Trans. Ind. Informat., vol. 3, no. 3, pp. 225-233, Aug. 2007.

[18] A. Burns and R.I. Davis, "Mixed criticality on controller area network", In Proc. Euromicro Conference on Real-Time Systems (ECRTS), pp. 125-134, 2013.

[19] H. Sivencrona, T. Olsson, R. Johansson and J. Torin, "RedCAN/sup TM/: simulations of two fault recovery algorithms for CAN," 10th IEEE Pacific Rim International Symposium on Dependable Computing, 2004. Proceedings., 2004, pp. 302-311

[20] M. Barranco, J. Proenza, G. Rodriguez-Navas and L. Almeida, "An active star topology for improving fault confinement in CAN networks", IEEE Trans. Ind. Electron., vol. 2, no. 2, pp. 78-85, May 2006.

[21] M. Barranco, L. Almeida and J. Proenza, "ReCANcentrate: A Replicated Star Topology for CAN Networks", Proc. 10th IEEE Int'l Conf. Emerging Technologies and Factory Automation (ETFA 05), pp. 469-476, 2005.

[22] S. Misbahuddin, S. M. Mahmud and N. Al-Holou, "Development and performance analysis of a data-reduction algorithm for automotive multiplexing", IEEE Trans. Veh. Technol., vol. 50, no. 1, pp. 162-169, Jan. 2001.

[23] P. R. Ramteke, S.M. Mahmud, "An Adaptive Data-Reduction Protocol for the future In-Vehicle Networks," Soc. Automotive Eng., SAE Paper 2005-01-1540, 2005.

[24] R. Miucic and S. M. Mahmud. “An improved adaptive data reduction protocol for in-vehicle networks”. In SAE, editor, In-Vehicle Software & Hardware Systems, number 2006-01-1327 in Transactions Journal of Passenger Cars: Electronic and Electrical Systems, pages pp. 650-658. SAE, April 2006. SAE 2006 World Congress & Exhibition.

[25] Radovan Miucic, S. M. Mahmud, Zeljko Popovic, "An Enhanced Data-Reduction Algorithm for Event-Triggered Networks," IEEE Transactions on vehicular Technology, Vol. 58, No.6, pp. 2663-2678, July, 2009.

[26] S. Kelkar and R. Kamal, "Boundary of Fifteen Compression algorithm for Controller Area Network based automotive applications," 2014 International Conference on Circuits, Systems, Communication and Information Technology Applications (CSCITA), 2014, pp. 162-167, doi: 10.1109/CSCITA.2014.6839253.

[27] Y. Wu and J. Chung, "Efficient controller area network data compression for automobile applications", Frontiers of Info. Technol. & Electro. Eng., vol. 16, no. 1, pp. 70-78, Jan. 2015.

[28] Y.-J. Wu and J.-G. Chung, "An improved controller area network data-reduction algorithm for in-vehicle networks", IEICE Trans. Fundamentals, vol. E100-A, no. 2, pp. 346-352, Feb 2017.

[29] Y.-J. Kim, Y Zou,Y.-E. Kim, and J.-G. Chung, " Multi-Level Data Arrangement Algorithm for CAN Data Compression", Springer, International Journal of Automotive Technology, vol. 21, no. 6, pp. 1527-1537, 2020.

[30] K. Park, M. Kang, and D. Shin, "Mechanism for Minimizing Stuffing-bit in CAN Messages," The 33rd Annual Conference of the IEEE Industrial Electronics Society (IECON'07), pp. 735-737, Nov. 2007.

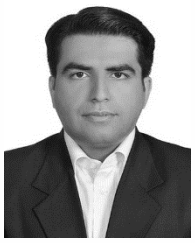
[31] A. Muhammad, D. Ayavoo and M. J. Pont, "A Novel Shared-Clock Scheduling Protocol for Fault-Confinement in CAN-based Distributed Systems", IEEE 5th International Conference on System of Systems Engineering, 2015.

[32] G. Rodriguez-Navas, J. Jimenez and J. Proenza, "An architecture for physical injection of complex fault scenarios in CAN networks", Proc. IEEE Emerging Technol. Factory Autom., vol. 2, pp. 125-127, 2003.

[33] C. Braun, L. Havet, and N. Navet, "NETCARBENCH: a benchmark for techniques and tools used in the design of automotive communication systems", in 7th IFAC International Conference on Fieldbuses and Networks in Industrial and Embedded Systems, 2007, pp. 321-328, Available at http://www.netcarbench.org.

[DOI: 10.52547/itrc.13.4.8]

Ismail Ghodsollahee received the M.Sc. degree in Computer Engineering from Ferdowsi University of Mashhad, Mashhad, Iran, in 2021. His main research area includes Real-Time Systems, Embedded Network, and Fault-Tolerant Design.



Yasser Sedaghat received the M.Sc. and Ph.D. degrees in Computer Engineering from Sharif University of Technology, Tehran, Iran, in 2006 and 2011, respectively. He is an Assistant Professor with the Department of Computer Engineering, Ferdowsi University of Mashhad (FUM), Mashhad, Iran. He has established and has been one of the chairs of the Dependable Distributed Embedded Systems (DDEmS) Laboratory, FUM, since 2012. His current research interests include Dependable Embedded Systems and Networks, Reliable Software Design, Embedded Operating Systems, and FPGA-based Designs.

[DOI: 10.52547/itrc.13.4.8]