Implementing and Evaluating Various Response-Time Analyses for Mixed Messages in CAN using MPS-CAN Analyzer

Saad Mubeen*†, Jukka Mäki-Turja*† and Mikael Sjödin*
* Mälardalen Real-Time Research Centre, Mälardalen University, Västerås, Sweden
† Arcticus Systems, Järfälla, Sweden
{saad.mubeen, jukka.maki-turja, mikael.sjodin}@mdh.se

***Abstract*—We integrate the Response Time Analysis (RTA) with offsets for mixed messages in Controller Area Network (CAN), where the CAN controllers implement abortable transmit buffers, with the MPS-CAN analyzer. Mixed messages are partly periodic and partly sporadic. They are implemented by several higher-level protocols for CAN that are used in the automotive industry. MPS-CAN analyzer is a free tool that supports several other existing RTA for periodic, sporadic and mixed messages in CAN. We perform extensive evaluation of the newly integrated analysis profile. Using the analyzer, we also perform a detailed comparative evaluation of various RTA for CAN.**

I. Introduction

Controller Area Network (CAN) [1] is a multi-master, event-triggered, serial communication bus protocol supporting bus speeds of up to 1 Mbit/s. It has been standardized as ISO 11898-1 [2]. It is a widely used protocol in the automotive domain. There are several higher-level protocols for CAN that are developed for various industrial applications such as CAN Application Layer, CANopen, J1939, Hägglunds Controller Area Network (HCAN) and MilCAN. Often, CAN finds its applications in hard real-time systems that must ensure that their deadlines are met. For this purpose, *a priori* analysis techniques, such as schedulability analysis [3], [4], [5], have been developed. Response-Time Analysis (RTA) [3], [4], [5], [6] is a powerful, mature and well established schedulability analysis technique to calculate upper bounds on response times of tasks or messages in a real-time system or a network respectively. Tindell et al. [7] developed RTA for CAN which is later revised by Davis et. al [8].

A. Previous work and paper contribution

In our previous work [9] we presented first implementation of MPS-CAN Analyzer. It is the first and only freely-available tool that supports RTA of periodic, sporadic as well as *mixed* messages in CAN[1]. Mixed messages are partly periodic and partly sporadic. They are implemented by several higher-level protocols used in the industry. In [9], we discussed the implementation of basic RTA for mixed messages in CAN [11], whereas the implementation of other analyses was an ongoing work. Moreover, [9] did not discuss comparative evaluation of the extended analyses for mixed messages in CAN. In [12], we discuss the implementation of several other extensions of RTA for periodic, sporadic and mixed messages in CAN. These extensions support response-time calculations for messages scheduled with or without offsets; messages having arbitrary jitter and deadlines; CAN controllers implementing different queueing policies, e.g., priority and FIFO; and controllers implementing abortable or non-abortable transmit buffers. However, the implementation in [12] does not support analysis of mixed messages that are scheduled with offsets in the network where controllers implement abortable or non-abortable transmit buffers [13]. In this paper we implement RTA for CAN in the system where periodic and mixed messages can be scheduled with offsets while the CAN controllers implement abortable or non-abortable transmit buffers. We also improve the graphical layout of the tool to support better usability. Furthermore, we perform extensive evaluation of newly added analysis. We also perform a detailed comparative evaluation of various RTA for CAN and provide recommendations.

[1]A commercial tool implements basic analysis for mixed messages [10].

II. Mixed transmission patterns supported by higher-level protocols

There are several higher-level protocols and commercial extensions of CAN that support mixed transmission. In this transmission, the task that queues messages can be invoked periodically as well as sporadically. If a message can be queued for transmission periodically as well as sporadically, it is said to be mixed. In other words, a mixed message is simultaneously time- and event-triggered. We identify three different implementations of mixed messages by higher-level protocols for CAN used in the industry namely CANopen [14], AUTOSAR [15] and HCAN [16]. The transmission pattern of a mixed message in these protocols is shown in Fig. 1(a), 1(b) and 1(c) respectively. The down-pointing arrows symbolize queueing of messages while the upward lines (labeled with alphabetic characters) represent arrival of events.

The CANopen protocol supports mixed transmission that corresponds to the Asynchronous Transmission Mode coupled with the Event Timer. A mixed message can be queued for transmission at the arrival of an event provided the *Inhibit Time* has expired. The Inhibit Time is the minimum time that must be allowed to elapse between queueing of two consecutive messages. A mixed message can also be queued periodically at the expiry of the Event Timer. The Event Timer is reset every time the message is queued. Once a mixed message is queued, any additional queueing of it will not take place during the Inhibit Time [14].

AUTOSAR can be viewed as a higher-level protocol if it uses CAN for network communication. Mixed transmission mode in AUTOSAR is widely used in practice. In AUTOSAR, a mixed message can be queued for transmission repeatedly with a time period. The mixed message can also be queued at the arrival of an event provided the Minimum Delay Time (MDT) has expired. However, each transmission of a mixed message, regardless of being periodic or sporadic, is limited by the MDT. This means that both periodic and sporadic transmissions are delayed until the MDT expires.

A mixed message in the HCAN protocol contains signals out of which some are periodic and some are sporadic. A mixed message is queued for transmission not only periodically, but also as soon as an event occurs that changes the value of one or more event signals, provided the Minimum Update Time (MUT) between the queueing of two successive sporadic instances of the mixed message has elapsed. Hence, the transmission of a mixed message due to arrival of events is constrained by the MUT.

In CANopen, the Event Timer is reset with every mixed transmission. The implementation of a mixed message in AUTOSAR is similar to CANopen to some extent. The main difference is that the periodic transmission can be delayed until the expiry of the MDT in AUTOSAR as indicated in Fig. 1(b). Whereas in CANopen, the periodic transmission is not delayed, in fact, the Event Timer is restarted with every sporadic transmission as shown in Fig. 1(a). The MDT timer is started with every periodic or sporadic transmission of a mixed message. Hence, the worst-case periodicity of a mixed message in CANopen and AUTOSAR can never be higher than the Inhibit Timer and MDT respectively. As a result, the mixed message can be treated as a special case of sporadic transmission. Therefore, all existing RTA are still applicable.

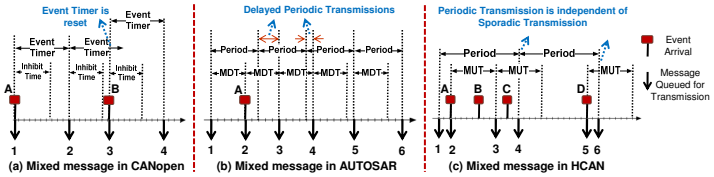

Fig. 1. Mixed transmission pattern in higher-level protocols for CAN

However, the periodic transmission is independent of the sporadic transmission in the HCAN protocol. The periodic timer is not reset with every sporadic transmission. A mixed message can be queued for transmission even if the MUT is not expired, e.g., see the transmission of instances 4 and 6 of the mixed message in Fig. 1(c). This indicates that the periodic transmission of a mixed message cannot be interfered by its sporadic transmission which is unlike in CANopen and AUTOSAR. The worst-case periodicity of a mixed message is neither bounded by the period nor by the MUT. Therefore, the existing analyses cannot be applied in this case. To the best of our knowledge, there is no free tool except for the MPS-CAN analyzer that analyzes this type of mixed messages.

III. Buffer Limitations and Queueing Policies

The different types of queueing polices implemented by CAN device drivers and communications stacks, internal organization, and hardware limitations in CAN controllers can have significant impact on the timing behavior of CAN messages. If an Electronic Control Unit (ECU) transmits more messages compared to the number of transmit buffers, the messages may be subjected to extra delay and jitter due to priority inversion.

A. Abortable transmit buffers

Let us consider the case in which the CAN controllers support transmission abort requests, e.g., Atmel AT89C51CC03/AT90CAN32/64 and Microchip MPC2515 [17]. In order to demonstrate an additional delay due to priority inversion in this case, consider the example of a message set shown in Fig. 2(a). Assume there are three nodes CC_c, CC_j and CC_k in the system and each node has three transmit buffers. m_1 is the highest priority message in the node CC_c as well as in the system. When m_1 becomes ready for transmission in the message queue, a lower priority message m_6 belonging to node CC_k is already under transmission. m_6 cannot be preempted because CAN uses fixed priority non-preemptive scheduling. This represents the blocking delay for m_1. At this time, all transmit buffers in CC_c are occupied by lower priority messages (say m_3, m_4 and m_5). The device drivers signal an abort request for the lowest priority message in the transmit buffers of CC_c) that is not under transmission. Hence, m_5 is aborted and copied from the transmit buffer to the message queue, whereas m_1 is moved to the vacated transmit buffer. The time needed to do the swapping is identified as *swapping time* in Fig. 2(a). A series of events may occur during the swapping : m_6 finishes its transmission, new arbitration round starts, message m_2 belonging to node CC_j and having priority lower than m_1 wins the arbitration and starts its transmission. Thus m_1 has to wait in the transmit buffer until m_2 finishes its transmission. This results in the priority inversion for m_1 and adds an extra delay to its response time. In [18], Khan et al. pointed out that this extra delay of the higher priority message appears as its additional jitter to the lower priority messages, e.g., m_5 in Fig. 2(a).

1) Discussion on message copy time and delay: If the message copy time is smaller than or equal to the inter-frame space (i.e., time to transmit 3 bits on CAN bus or $3{*}\tau_{bit}$ time), a lower priority message in the transmit buffer (that is not under transmission) can be swapped with a higher priority message in the message queue before transmission of the next frame [1]. Hence, there will be no priority inversion. This means that the message copy time must be, at least, $4{*}\tau_{bit}$ for the priority inversion to occur. In Legacy systems, there may be slow controllers, i.e., the speed of the controllers can be slower than the maximum operating speed of the CAN bus (1 Mbit/s). Since the amount of data transmitted in a CAN message ranges from 0 to 8 bytes, the transmission time of a message also varies accordingly. According to [8], the transmission time of a CAN message with standard frame format ranges from $55{*}\tau_{bit}$ to $135{*}\tau_{bit}$ for the amount of data contained in the message that ranges from 0 to 8 bytes respectively. Intuitively, the message copy time of $4{*}\tau_{bit}$ can range from 7.3% to 3% of transmission time of a message with 0 to 8 bytes of data respectively. Due to slow controllers in legacy systems, the message copy time can be greater than $4{*}\tau_{bit}$, hence, higher than 7.3% of its transmission time.

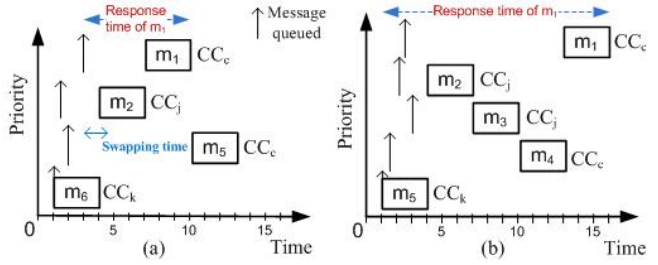

Fig. 2. Demonstration of priority inversion in the case of (a) abortable transmit buffers, (b) non-abortable transmit buffers

B. Non-abortable transmit buffers

Now we consider the case in which the CAN controllers implement non-abortable transmit buffers, e.g., Philips 82C200 [19], [20], [7]. Consider an example of three controllers CC_c, CC_j, CC_k connected to a single CAN network in Fig. 2 (b). Let m_1, belonging to CC_c, be the highest priority message in the system. Assume that when m_1 is ready to be queued, all transmit buffers in CC_c are occupied by lower priority messages which cannot be aborted because the controllers implement non-abortable transmit buffers. In addition, m_1 can be blocked by any lower priority message because the lower priority message already started its transmission. In this example m_1 is blocked by m_5 that belongs to node CC_k. Since all transmit buffers in CC_c are full, m_1 has to wait in the message queue until one of the messages in the transmit buffers of node CC_c is transmitted.

Let m_4 be the highest priority message in the transmit buffers of node CC_c. m_4 can be interfered by higher priority messages (m_2 and m_3) belonging to other nodes. Hence, it can be seen that priority inversion for m_1 takes place

because m_1 cannot start its transmission before m_4 finishes its transmission, while m_4 has to wait until messages m_2 and m_3 are transmitted. This adds an additional delay to the worst-case response time of m_1. In this example, this additional delay is the sum of the worst-case transmission times of m_2, m_3 and m_4. This additional delay appears as additional jitter of m_1 as seen by the lower priority messages.

C. Priority and FIFO queues

The most natural queuing policy suited to CAN nodes is priority-based queuing. However, due to simplicity of FIFO policy some CAN controllers implement FIFO queues, e.g., Microchip PIC32MX, Infineon XC161CS, Renesas R32C/160 and XILINX LogiCORE IP AXI Controller [17], [18]. In case of nodes implementing priority queues, each node selects the highest priority message from its transmit buffers while entering into the bus arbitrations. The highest priority message among them wins the bus arbitration. On the other hand, when the nodes implement FIFO queues, the oldest message in the transmit queue of each node competes for the bus. However, the bus arbitration among these messages is done on priority basis. Consider an example of three nodes that are connected to a single CAN network as shown in Fig. 3. Assume that Node A sends the messages m_1, m_3 and m_5; Node B sends the messages m_2, m_4 and m_9; and Node C sends the messages m_6, m_7 and m_8. The priority of a message is indicated by its subscript (smaller the subscript, the higher the priority).

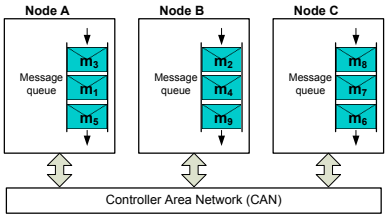

Fig. 3. Example to demonstrate different queueing policies

Let the nodes implement priority queues. In the first round, Nodes A, B, and C pick messages m_1, m_2 and m_6 respectively. m_1 wins the arbitration because of higher priority and is transmitted over the network as shown in Fig. 4. In the second round, Nodes A, B, and C pick messages m_3, m_2 and m_6 respectively. m_2 wins the arbitration and is transmitted over the network. Similar priority-based selection and arbitration occur during the rest of the rounds as shown in Fig. 4.

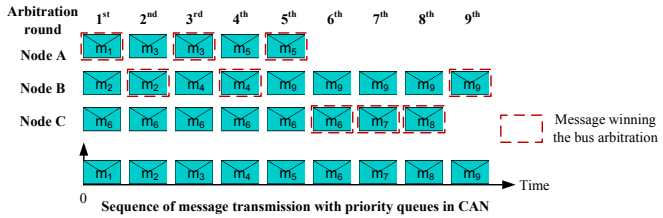

Fig. 4. priority-based queues and CAN arbitration

Now we assume that the nodes implement FIFO queues. In the first round, Nodes A, B, and C pick the oldest messages m_5, m_9 and m_6 respectively. m_5 wins the bus arbitration due to its higher priority and is transmitted as shown in Fig. 5. In the second round, Nodes A, B, and C pick messages m_1, m_9 and m_6 respectively. This time, m_1 wins the bus arbitration and is transmitted over the network. Similar FIFO selection and priority-based arbitration occur during the rest of the rounds as shown in Fig. 5. It can be seen that the priorities of messages are sometimes not respected in the FIFO queue within a node, e.g., a lower priority message m_5 is transmitted before the higher priority message m_1 as shown in Fig. 5. This results in priority inversions due to which higher priority messages may have very large response times, e.g., different response time of m_2 in the systems with priority and FIFO queues in Fig. 4 and Fig. 5 respectively.

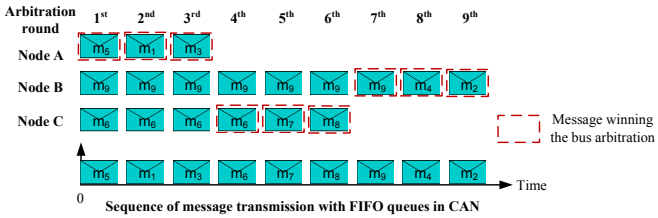

Fig. 5. FIFO-based queues and CAN arbitration

IV. Related Work and Implemented Analysis

A. Related work

In [21], Davis et al. extend the analysis of [7], [8] which is now applicable to the CAN network where some nodes implement priority queues and some implement FIFO queues. The message deadlines in [21] are assumed to be smaller than or equal to the corresponding periods. This assumption is lifted in [22] by by supporting the analysis of messages with arbitrary deadlines. Moreover, they extend their work to support RTA of CAN for FIFO and work-conserving queues. The analysis in [7], [8] assumes that the CAN controllers have very large transmit buffers. However, most CAN controllers have small number of transmit buffers [23], [22]. If all buffers in the controller are occupied by lower priority messages, a higher priority message released in the same controller may suffer from priority inversion [7], [18], [20], [24]. The analysis in [7], [8] has been extended in [18] and [23] to support the analysis of network that contain abortable and non-abortable transmit buffers in the controllers respectively. Most of the CAN enabled ECUs support transmit abort requests [18].

All these analyses assume that the messages are queued for transmission periodically or sporadically. Mubeen et al. [11] extend the existing analysis [7], [8] to support mixed messages in CAN where nodes implement priority queues. Mubeen et al. [25] further extend their analysis to support mixed messages in the network where some nodes implement priority queues while others implement FIFO queues. RTA for mixed messages in CAN [11] has been extended to support the analysis of network that contain abortable and non-abortable transmit buffers in the controllers in [26] and [27] respectively. But, none of the analyses discussed above supports messages that are scheduled with offsets i.e., using externally imposed delays between the times when the messages can be queued. In order to avoid deadlines violations due to high transient loads, current automotive embedded systems are often scheduled with offsets [28]. The worst-case response-times of lower priority messages in CAN can be reduced if the messages are scheduled with offsets [29], [30]. A method for the assignment of offsets to improve the overall bandwidth utilization is proposed in [30]. RTA with offsets for CAN has been developed by several researchers [31], [32], [29], [33], [28].

None of the above analyses supports mixed messages that are scheduled with offsets. Offset-based analysis [31] is extended in [34] to support response-time calculations for mixed messages in CAN. However, this analysis is restricted due to limitations regarding message jitter and deadlines. The source of these limitations comes from the base analysis [31]. In [35], Mubeen et al. removed these limitations and extended the analysis for mixed messages [11] with offsets [28]. Mubeen et al. further extend the analysis for mixed messages with offsets in CAN supporting abortable transmit buffers [13].

B. Related tools

VNA [36] is a communication design tool that supports RTA for CAN. It implements RTA of CAN developed by Tindell et al. [7]. Vector [37] is a tools provider for the development of networked electronic systems. CANalyzer [38] supports the simulation, analysis and data logging for the systems that use CAN. CANoe [39] is a tool for simulation of functional and extra-functional (e.g., timing) behavior of ECU networks. Network Designer CAN is another tool by Vector that is able to perform timing analysis of CAN. SymTA/S [40] is a tool for model-based timing analysis and optimization. Among other analyses, it supports statistical, worst- and best-case timing analyses for CAN. RTaW-Sim [41] is a tool for the simulation and performance evaluation of the CAN network. The Rubus-ICE is a commercial tool suite developed by Arcticus Systems [42] in close collaboration with Mälardalen University Sweden. Among other analyses, it supports RTA of CAN [7], [8] and RTA of CAN for mixed messages[11], [43]. To the best of our knowledge, there is no freely-available tool that implements RTA of CAN for mixed messages. The main purpose of MPS-CAN Analyzer is to support RTA of periodic, sporadic and mixed messages in CAN. The analyses implemented in MPS-CAN analyzer are shown in Fig. 6.

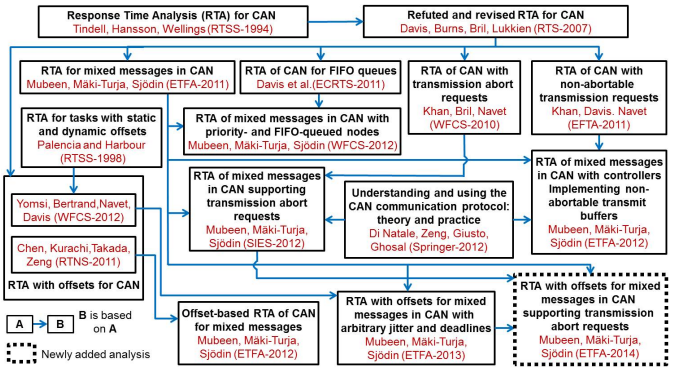

Fig. 6. Graphical representation of Response Time Analysis (RTA) and its extensions implemented in MPS-CAN Analyzer

C. Implementation and distribution

The tool is implemented in C language. Each analysis profile supported by the tool is implemented as a separate C file. The Layout of the tool is shown in Fig. 7. It has a scope for further extensions in the future. The link to the tool can be found at https://github.com/saadmubeen/MPS-CAN.

V. Evaluation of various RTA for CAN

A. Experimental setup

The system consists of six ECUs that are connected to the CAN network. The speed of the network is set to 250 Kbit/s. There are 60 messages in the system. The message set is generated from the NETCARBENCH tool [44] which is a benchmark used in the design of automotive embedded systems. It should be noted that NETCARBENCH cannot generate mixed messages. We randomly assign mixed, periodic, and sporadic transmission types to 40%, 30%, and 30% generated messages respectively. This means, there are 24 mixed, 18 periodic and 18 sporadic messages in the system. The messages are equally distributed among the ECUs, i.e., each ECU sends 4 mixed, 3 periodic and 3 sporadic messages over the network. All the attributes of these messages are tabulated in the Fig. 8. The attributes of a message m_m are identified as follows. The priority, sender node ID, transmission type, number of data bytes in the message, offset, jitter, period, minimum update time and deadline are represented by P_m, CC_m, ξ_m, s_m, O_m, J_m, T_m, MUT_m and D_m respectively. All timing values in the table are expressed in milliseconds. We perform a number of tests on the message set. The network bandwidth utilization calculated by MPS-CAN analyzer for this message set in each test is equal to 59.203793%.

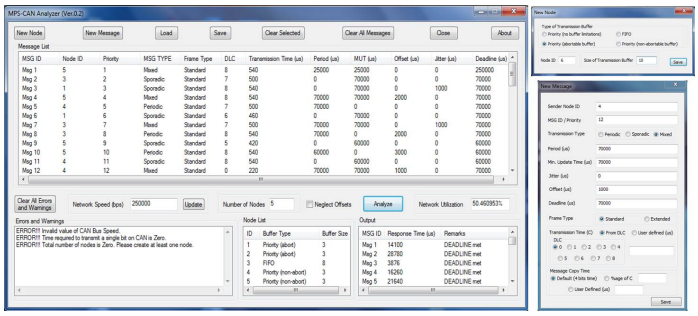

Fig. 7. MPS-CAN Analyzer layout, inputs and outputs

P_m	CC_m	ξ_m	s_m	O_m	J_m	T_m	MUT_m	D_m	P_m	CC_m	ξ_m	s_m	O_m	J_m	T_m	MUT_m	D_m	P_m	CC_m	ξ_m	s_m	O_m	J_m	T_m	MUT_m	D_m
1	5	M	8	0	0	25	25	25	21	4	M	8	3	0	70	70	70	41	2	M	1	7	1	70	70	70
2	3	S	7	0	0	0	70	70	22	1	M	0	4	1	60	60	60	42	1	P	1	6	0	70	0	70
3	1	S	8	0	1	0	70	70	23	2	S	0	0	1	0	70	70	43	4	S	8	0	2	0	80	80
4	5	M	8	2	0	70	70	70	24	3	S	6	0	0	0	70	70	44	5	S	8	0	2	0	70	70
5	4	P	7	0	0	70	0	70	25	3	M	8	5	1	70	70	70	45	6	S	8	0	2	0	70	70
6	1	S	6	0	0	0	70	70	26	2	P	6	3	0	70	0	70	46	3	M	2	8	1	80	80	80
7	3	M	7	0	1	70	70	70	27	5	M	2	7	1	60	60	60	47	3	S	4	0	2	0	70	70
8	3	P	8	2	0	70	0	70	28	4	P	1	5	0	80	0	80	48	6	M	8	7	2	70	70	70
9	5	S	5	0	0	0	60	60	29	3	M	6	5	0	70	70	70	49	1	M	8	8	1	70	70	70
10	5	P	8	3	0	60	0	60	30	1	P	1	5	0	70	0	70	50	6	M	7	8	1	70	70	70
11	4	S	8	0	0	0	60	60	31	2	M	7	4	1	70	70	70	51	6	P	8	0	2	70	0	70
12	4	M	0	1	0	70	70	70	32	1	S	8	0	0	0	70	70	52	6	P	6	2	1	70	0	70
13	1	M	6	2	0	60	60	60	33	2	S	8	0	0	0	70	70	53	6	S	1	0	1	0	70	70
14	3	P	8	3	0	50	0	50	34	2	P	8	5	2	80	0	80	54	6	P	2	3	0	70	0	70
15	5	M	8	4	0	70	70	70	35	2	P	5	5	0	60	0	60	55	6	S	1	0	1	0	70	70
16	4	M	5	4	0	50	50	50	36	4	S	8	0	1	0	70	70	56	6	M	2	4	1	70	70	70
17	2	S	8	0	1	0	80	80	37	1	P	5	6	1	70	0	70	57	5	S	8	0	2	0	20	80
18	2	M	8	1	0	70	70	70	38	4	P	1	6	1	80	0	80	58	1	M	8	7	2	70	70	70
19	5	P	8	4	0	70	0	70	39	3	P	8	7	1	80	0	80	59	6	M	8	8	1	70	70	70
20	5	P	7	5	1	70	0	70	40	4	M	0	7	1	70	70	70	60	2	M	7	8	1	70	70	70

Fig. 8. Attributes of the message set under analysis

B. Comparison of various RTA for CAN

In this subsection, we perform five different tests as follows.

1) All ECUs implement priority queuing policy while the number of transmit buffers are large enough to avoid aborting transmissions. The message set is analyzed using the RTA for mixed messages in CAN with no buffer limitations [11].
2) All ECUs implement priority queuing policy and abortable transmit buffers. The message set is analyzed using the RTA for mixed messages in CAN with abortable transmit buffers [26], [13].
3) All ECUs implement priority queuing policy and non-abortable transmit buffers. The message set is analyzed using the RTA for mixed messages in CAN with non-abortable transmit buffers [27].
4) All ECUs implement FIFO queues. The message set is analyzed using the RTA for mixed messages in CAN with FIFO queues [25].
5) Heterogeneous system: two ECUs implement priority queuing policy and abortable transmit buffers; two ECUs implement priority queuing policy and non-abortable transmit buffers; and two ECUs implement FIFO queues. The MPS-CAN analyzes each ECU differently using the corresponding analysis profile from the above three tests.

Response times of the messages calculated in all five tests are plotted in Fig. 9. It can be seen that the response times are lowest (best) in the first test because we have considered ideal conditions (no buffer limitations in the CAN controllers). The second test results in the second best response times. However, they are higher compared to the first test due to extra delay from priority inversion due to transmission abort requests. The

third test results in overall third best response times (with some exceptions). It can be seen that the extra delay due to priority inversion in non-abortable transmit buffers is higher compared to abortable transmit buffers. The fourth test yields the highest response times because of high buffering time and delays due to priority inversion in FIFO queues. Furthermore, the response times of messages are significantly high compared to the rest of the tests. The response times in the heterogeneous system are higher compared to first three tests but significantly lower than the fourth test where FIFO queues are used. From the results, one can infer that the ECUs that implement FIFO queues in the CAN controllers should be avoided. In order to calculate correct (not optimistic) response times, the RTA for CAN should correctly match the queueing policy and practical limitations in the CAN controllers.

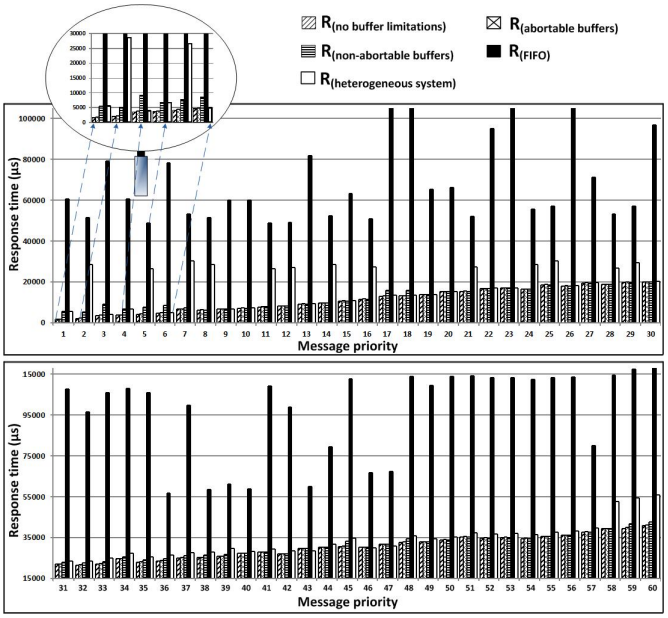

Fig. 9. Analysis results using various RTA for CAN

C. Effect of message copy time on schedulability

In this subsection, we compare the effect of message copy times on their response times. We perform four tests where all ECUs implement abortable transmit buffers. However, the message copy times are different in these tests. In the first test, the message copy time for each message is equal to $4*\tau_{bit}$ time (see Subsection III-A1). In the rest of the tests, it is 10, 20 and 30 percent of corresponding transmission times of messages respectively. The calculated response times of the messages in all test are plotted in Fig. 10. The results indicate that the increase in the response times of messages is directly proportional to the increase in the amount of message copy times. If the message copy time is less than the inter-frame space (time required to transmit 3-bits of data on CAN), the response times of messages in the system with abortable transmit buffers becomes equals to the response times of same messages in the system with no buffer limitations.

D. Effect of offsets on schedulability

Finally, we perform four more tests on the message set to explore the effect of offsets on the schedulability of the message set. In the first two tests, the message set is analyzed using RTA for mixed messages in CAN with no buffer limitations [11] and with abortable transmit buffers [26]. However the offsets of all messages are assumed to be zero. There is a newly added check box "Neglect offsets" in the MPS-CAN analyzer as shown in Fig. 7 that actually neglects the offsets when analyzing the messages. In the next two tests, the first two tests are repeated while considering message offsets. Also, the messages are analyzed using the newly implemented RTA with offsets for mixed messages in CAN supporting transmission abort requests [13]. The response times calculated in the four tests are plotted in Fig. 11. The results indicate that the response times can be reduced when messages are scheduled with offsets. We observe 2.462% improvement in the schedulability of the system when the messages are scheduled with offsets. As discussed earlier, NETCARBENCH cannot generate mixed messages, hence, the offsets assigned to the mixed messages are not optimal. The schedulability can be further improved if an optimal offset assignment algorithm for mixed messages is used. The percentage improvement in schedulability is calculated as follows.

$$\left[\left(\sum_{\forall m_m \in \aleph}\left[\frac{R_m^{\{no-offset\}} - R_m^{\{offset\}}}{D_m}\right]\right) \Big/ (sizeof(\aleph))\right] * 100$$

Where, m_m, R_m, D_m and $\aleph$ represent a message, response time, deadline and the set of all messages in the system respectively.

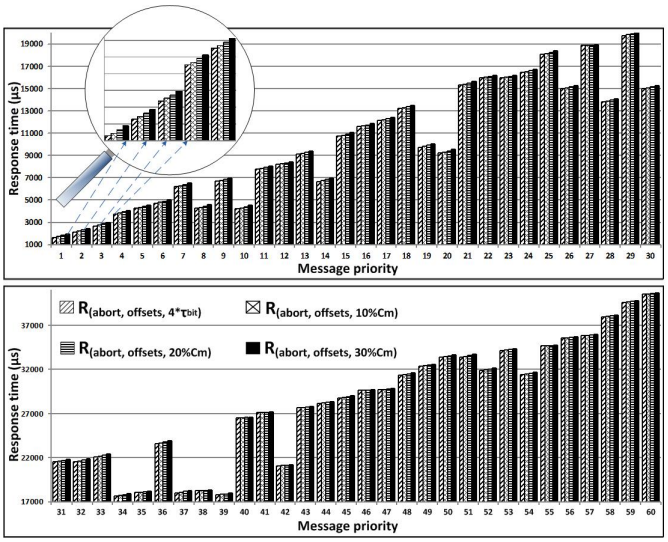

Fig. 10. Analysis results: effect of message copy time on schedulability

VI. Conclusion

We implemented a new RTA for CAN in a free tool MPS-CAN analyzer. The implemented RTA supports periodic, sporadic as well as mixed messages in the system where transmission abort requests in CAN controllers all allowed. Mixed messages are partly periodic and partly sporadic and are implemented by several higher-level protocols for CAN that are used in the automotive industry today. We conducted a number of tests to perform detailed evaluation of all RTA profiles available in MPS-CAN analyzer. These analyses consider various aspects and practical limitations such as mixed messages; messages scheduled with offsets; messages with arbitrary jitter and deadlines; priority or FIFO queueing policies; limitations of transmit buffers in CAN controllers such abortable or non-abortable; and heterogeneous systems that consist of different types of ECU's. We can make several recommendations based on the analyses results and their evaluation. The ECUs that implement FIFO queues should be avoided because of high buffering time and delays due to priority inversion in FIFO queues. Due to lower response times, the controllers that implement abortable transmit buffers should be preferred over those that implement non-abortable

transmit buffers. The schedulability of the system can be improved if messages are scheduled with offsets. We observed 2.462% improvement in schedulability in one of the tests when the messages are scheduled with offsets. Finally, it can be concluded that if RTA for CAN does not correctly account for transmission patterns by higher-level protocols, queueing policies and practical limitations in the CAN controllers, the calculated response times of messages can be optimistic.

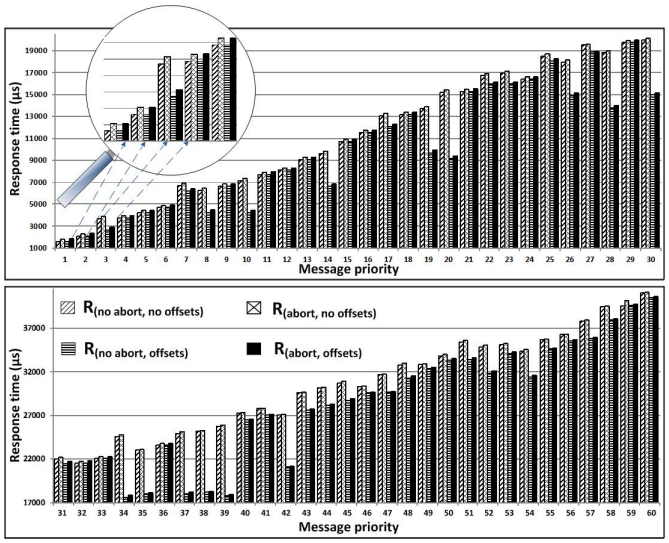

Fig. 11. Analysis results: effect of offsets on schedulability

ACKNOWLEDGEMENT

This work is supported by the Swedish Research Council within the project SynthSoft. The authors thank the industrial partners Arcticus Systems, BAE Systems Hägglunds and Volvo CE, Sweden.

REFERENCES

[1] Robert Bosch GmbH, "CAN Specification Version 2.0," postfach 30 02 40, D-70442 Stuttgart, 1991.
[2] ISO 11898-1, "Road Vehicles interchange of digital information controller area network (CAN) for high-speed communication, ISO Standard-11898, Nov. 1993."
[3] N. Audsley, A. Burns, M. Richardson, K. Tindell, and A. J. Wellings, "Applying new scheduling theory to static priority pre-emptive scheduling," *Software Engineering Journal*, vol. 8, no. 5, pp. 284–292, 1993.
[4] N. Audsley, A. Burns, R. Davis, K. Tindell, and A. Wellings, "Fixed priority pre-emptive scheduling: an historic perspective," *Real-Time Systems*, vol. 8, no. 2/3, pp. 173–198, 1995.
[5] L. Sha, T. Abdelzaher, K.-E. A. rzén, A. Cervin, T. P. Baker, A. Burns, G. Buttazzo, M. Caccamo, J. P. Lehoczky, and A. K. Mok, "Real Time Scheduling Theory: A Historical Perspective," *Real-Time Systems*, vol. 28, no. 2/3, pp. 101–155, 2004.
[6] M. Joseph and P. Pandya, "Finding response times in a real-time system," *The Computer Journal*, vol. 29, no. 5, pp. 390–395, 1986.
[7] K. Tindell, H. Hansson, and A. Wellings, "Analysing real-time communications: controller area network (CAN)," in *Real-Time Systems Symposium (RTSS) 1994*, pp. 259 –263.
[8] R. Davis, A. Burns, R. Bril, and J. Lukkien, "Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised," *Real-Time Systems*, vol. 35, pp. 239–272, 2007.
[9] S. Mubeen, J. Mäki-Turja, and M. Sjödin, "Many-in-one Response-Time Analyzer for Controller Area Network," in *In WATERS workshop*, 2013.
[10] "Rubus-ICE," http://www.arcticus-systems.com.
[11] S. Mubeen, J. Mäki-Turja, and M. Sjödin, "Extending schedulability analysis of controller area network (CAN) for mixed (periodic/sporadic) messages," in *16th IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, sept. 2011.
[12] S. Mubeen, J. Mäki-Turja and M. Sjödin, "MPS-CAN Analyzer: Integrated Implementation of Response-Time Analyses for Controller Area Network," *Journal of Systems Architecture*, 2014.
[13] S. Mubeen, J. Mäki-Turja, and M. Sjödin, "Response Time Analysis with Offsets for Mixed Messages in CAN Supporting Transmission Abort Requests," in *19th IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, sept. 2014.
[14] "CANopen Application Layer and Communication Profile. CiA Draft Standard 301. Version 4.02. February 13, 2002," http://www.can-cia.org/index.php?id=440.
[15] "AUTOSAR Techincal Overview, Version 2.2.2., Release 3.1, The AUTOSAR Consortium, Aug., 2008," http://autosar.org.
[16] "Hägglunds Controller Area Network (HCAN), Network Implementation Specification," *BAE Systems Hägglunds, Sweden (internal document)*, April 2009.
[17] R. Davis, S. Kollmann, V. Pollex, and F. Slomka, "Schedulability analysis for controller area network (can) with fifo queues priority queues and gateways," *Real-Time Systems*, vol. 49, no. 1, pp. 73–116, 2013.
[18] D. Khan, R. Bril, and N. Navet, "Integrating hardware limitations in can schedulability analysis," in *8th IEEE International Workshop on Factory Communication Systems (WFCS)*, may 2010, pp. 207 –210.
[19] D. Khan, R. Davis, and N. Navet, "Schedulability analysis of CAN with non-abortable transmission requests," in *16th IEEE Conference on Emerging Technologies Factory Automation (ETFA)*, sept. 2011, pp. 1–8.
[20] M. D. Natale, "Evaluating message transmission times in controller area networks without buffer preemption," in *8th Brazilian Workshop on Real-Time Systems*, 2006.
[21] R. I. Davis, S. Kollmann, V. Pollex, and F. Slomka, "Controller Area Network (CAN) Schedulability Analysis with FIFO queues," in *23rd Euromicro Conference on Real-Time Systems*, July 2011.
[22] R. Davis and N. Navet, "Controller Area Network (CAN) Schedulability Analysis for Messages with Arbitrary Deadlines in FIFO and Work-Conserving Queues," in *9th IEEE International Workshop on Factory Communication Systems (WFCS)*, may 2012, pp. 33 –42.
[23] D. Khan, R. Davis, and N. Navet, "Schedulability analysis of CAN with non-abortable transmission requests," in *16th IEEE Conference on Emerging Technologies Factory Automation (ETFA)*, sept. 2011.
[24] Marco Di Natale, Haibo Zeng, Paolo Giusto, Arkadeb Ghosal, *Understanding and Using the Controller Area Network Communication Protocol*. Springer, 2012.
[25] S. Mubeen, J. Mäki-Turja and M. Sjödin, "Response-Time Analysis of Mixed Messages in Controller Area Network with Priority- and FIFO-Queued Nodes," in *9th IEEE International Workshop on Factory Communication Systems (WFCS)*, May 2012.
[26] S. Mubeen, J. Mäki-Turja, and M. Sjödin, "Response Time Analysis for Mixed Messages in CAN Supporting Transmission Abort Requests," in *7th IEEE International Symposium on Industrial Embedded Systems (SIES)*, June 2012.
[27] S. Mubeen, J. Mäki-Turja, and M. Sjödin, "Extending response-time analysis of mixed messages in can with controllers implementing non-abortable transmit buffers," in *17th IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, sept. 2012.
[28] P. Yomsi, D. Bertrand, N. Navet, and R. Davis, "Controller Area Network (CAN): Response Time Analysis with Offsets," in *9th IEEE International Workshop on Factory Communication Systems*, May 2012.
[29] A. Szakaly, "Response Time Analysis with Offsets for CAN," Master's thesis, Department of Computer Engineering, Chalmers University of Technology, Nov. 2003.
[30] M. Grenier, L. Havet, and N. Navet, "Pushing the limits of can-scheduling frames with offsets provides a major performance boost," in *4th European Congress on Embedded Real Time Software*, 2008.
[31] Y. Chen, R. Kurachi, H. Takada, and G. Zeng, "Schedulability comparison for can message with offset: Priority queue versus fifo queue," in *19th International Conference on Real-Time and Network Systems (RTNS)*, Sep. 2011, pp. 181–192.
[32] L. Du and G. Xu, "Worst case response time analysis for can messages with offsets," in *IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, nov. 2009, pp. 41 –45.
[33] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, and R. Ernst, "System level performance analysis - the symta/s approach," *Computers and Digital Techniques*, vol. 152, no. 2, pp. 148–166, March 2005.
[34] S. Mubeen, J. Mäki-Turja, and M. Sjödin, "Worst-case response-time analysis for mixed messages with offsets in controller area network," in *17th IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, sept. 2012.
[35] S. Mubeen, J. Mäki-Turja and M. Sjödin, "Extending offset-based response-time analysis for mixed messages in controller area network," in *18th IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, sept. 2013.
[36] "Volcano Network Architect (VNA). Mentor Graphics," http://www.mentor.com/products/vnd/communication-management/vna.
[37] "Vector. http://www.vector.com."
[38] "CANalyzer. http://www.vector.com/vi_canalyzer_en.html."
[39] "CANoe. www.vector.com/portal/medien/cmc/info/canoe_productinformation_en.pdf."
[40] A. Hamann, R. Henia, R. Racu, M. Jersak, K. Richter, and R. Ernst, "Symta/s - symbolic timing analysis for systems," 2004.
[41] "RTaW-Sim. http://www.realtimeatwork.com/software/rtaw-sim."
[42] "Arcticus Systems AB," http://www.arcticus-systems.com.
[43] S. Mubeen, J. Mäki-Turja, and M. Sjödin, "Support for end-to-end response-time and delay analysis in the industrial tool suite: Issues, experiences and a case study," *Computer Science and Information Systems, ISSN: 1361-1384*, vol. 10, no. 1, 2013.
[44] 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 & Networks in Industrial & Embedded Systems*, Nov. 2007.