Extending Worst-Case Response-Time Analysis for Mixed Messages in Controller Area Network with Priority and FIFO Queues

Saad Mubeen, Student Member, IEEE, Jukka Mäki-Turja, Member, IEEE, and Mikael Sjödin, Member, IEEE

Abstract—The existing worst-case response-time analysis for Controller Area Network (CAN) with nodes implementing priority and First In First Out (FIFO) queues does not support mixed messages. It assumes that a message is queued for transmission either periodically or sporadically. However, a message can also be queued both periodically and sporadically using mixed transmission mode implemented by several higher-level protocols for CAN that are used in the automotive industry. We extend the existing analysis for CAN to support any higher-level protocol for CAN that uses periodic, sporadic, and mixed transmission of messages in the systems where some nodes implement priority queues while others implement FIFO queues. In order to provide a proof of concept, we implement the extended analysis in a free tool, conduct an automotive-application case study, and perform comparative evaluation of the extended analysis with the existing analysis.

Index Terms—Controller area network, CAN protocol, real-time network, response-time analysis, distributed embedded systems, schedulability analysis, FIFO queues, mixed messages.

I. INTRODUCTION

The Controller Area Network (CAN) [1] is a widely used real-time network protocol in the automotive domain. In 2003, the International Organization for Standardization (ISO) standardized CAN in ISO 11898-1 [2]. It is a multi-master, event-triggered, serial communication bus protocol supporting bus speeds of up to 1 Mbit/s. CAN with Flexible Data-rate (CAN FD) [3] is a new protocol based on CAN that can achieve bus speed of more than 1 Mbit/s. According to CAN in Automation (CiA) [4], the estimated number of CAN enabled controllers sold in 2011 are about 850 million. In total, more than two billion CAN controllers have been sold until today. Out of this huge number, approximately 80% CAN controllers have been used in the automotive applications. For example, there can be as many as 20 CAN networks\(^1\) used in a modern heavy truck, while the number of CAN messages transmitted over these networks can be over 6000 [5]. These facts and figures indicate the popularity of CAN in the automotive domain. It is also used in other domains such as industrial control, medical equipments, maritime electronics, production machinery, and many others. There are a number of higher-level protocols for CAN that are developed for many industrial applications such as CAN Application Layer (CAL) [6], CANopen [7], Hägglunds Controller Area Network (HCAN) [8], and CAN for Military Land Systems domain (MiCAN) [9].

CAN finds its applications in the systems that have real-time requirements. This means that the time for response to some stimulus is as crucial as logical correctness of the response. In other words, logically correct but late response may be considered as bad as logically incorrect response. Hence, the providers of these systems are required to ensure that the actions by the systems will be taken at times that are appropriate to their environment. In order to provide evidence that each action by the system will be provided in a timely manner, a priori analysis techniques, such as schedulability analysis [10], [11], [12], have been developed by the research community. Response-Time Analysis (RTA) [13], [10], [11], [12] is a powerful, mature and well established schedulability analysis technique. It is a method to calculate upper bounds on the response times of tasks or messages in a real-time system or a real-time network respectively. RTA applies to systems (or networks) where tasks (or messages) are scheduled with respect to their priorities and which is the predominant scheduling technique used in real-time operating systems (or real-time network protocols, e.g., CAN) today [14].

A. Extended version

This paper extends our previous work that was presented in the 9th IEEE International Workshop on Factory Communication Systems (WFCS 2012) [15]. The workshop paper presents the response-time analysis for mixed messages in CAN with FIFO queues. However, it lacks the calculations for maximum buffering time in the FIFO queues which is an important factor in the response-time calculations. Moreover, it does not evaluate and compare the extended analysis with the other related analyses. In the extended version of the paper, we generalize the analysis, by complementing it with the algorithm to calculate maximum buffering time in the FIFO queues. Moreover, we implement the extended analysis in a freely-available tool. We also show the applicability of the extended analysis by conducting an automotive-application case study. We also perform extensive evaluation of the extended analysis.

B. Related works

Tindell et al. [16] developed the schedulability analysis for CAN. It has been implemented in the automotive industrial
tools such as Volcano Network Architect (VNA) [17], Davis et al. [18] found the analysis to be flawed in some cases. Accordingly, they revisited and revised the original analysis. The revised analysis is also implemented in the existing industrial tool suite Rubus-ICE [19], [20] which is used by several international companies.

The scheduling model used in [16], [18] assumes that the messages are queued for transmission either periodically or sporadically. These analyses do not support the response time calculations for mixed messages in CAN, i.e., the messages that are simultaneously time (periodic) and event triggered. Mixed messages are implemented by several higher-level protocols based on CAN that are used in the automotive industry. Mubeen et al. [21] extended the seminal analysis [16], [18] to support the worst-case response-time calculations for mixed messages in CAN.

However, the analyses in [16], [18], [21] assume that the device drivers in the CAN controllers implement priority-based queues. This means that the highest priority message at each node\textsuperscript{2} enters into the bus arbitration. This assumption may become invalid when some controllers in the network implement FIFO queues. Some examples of the CAN controllers implementing FIFO queues are Infineon XC161CS, Microchip PIC32MX, Renesas R32C/160 and XILINX LogiCORE IP AXI Controller [22], [23], [24]. Davis et al. [25], [22] extended the analysis for CAN where some nodes implement priority queues while others implement FIFO queues.

In the works in [25], [22], the message deadlines are assumed to be smaller than or equal to the corresponding periods. In [26], Davis et al. lifted this assumption by supporting the analysis for CAN messages with arbitrary deadlines. Furthermore, they extended their previous works to support RTA for CAN with FIFO and work-conserving queues. However, the analyses for CAN with FIFO queues do not support mixed messages.

C. Paper contributions and motivation

We identified that the existing RTA for CAN with FIFO queues [25], [22], [26] does not support the analysis of common message transmission patterns, i.e., mixed messages. These type of messages are implemented by some higher-level protocols for CAN that are used in the automotive industry. Further, the existing analysis for mixed messages in CAN [21] does not support the analysis of the systems containing nodes that implement FIFO queues. We extend the existing analysis for CAN with FIFO queues [25], [22], [26] by integrating it with the analysis for mixed messages in CAN with priority queues [21]. Moreover, we generalize the extended analysis for CAN with FIFO queues by presenting the algorithm for the calculations of maximum buffering time in the FIFO queues. The relationship between the existing and extended analyses is shown in Fig. 1.

The extended analysis does not put any restrictions on the message deadlines, i.e., the deadline of a message can be lower, equal, or higher than its transmission period. The extended analysis is able to calculate the worst-case response times of periodic, sporadic and mixed CAN messages in networks where some nodes implement priority queues while others implement FIFO queues. We also implement the extended analysis in a freely-available tool [27]. Furthermore, we show the applicability of the extended analysis by conducting the automotive-application case study. We also perform extensive evaluation of the extended analysis.

The motivation for this work comes from the industrial requirements and the activity of implementing the holistic response-time analysis [28] in the existing industrial tool suite, Rubus-ICE [20]. This tool provides a model- and component-based development environment for resource-constrained automotive distributed real-time systems while supporting several higher-level protocols based on CAN.

D. Paper layout

The rest of the paper is organized as follows. In Section II, we discuss mixed transmission patterns supported by several higher-level protocols for CAN. In Section III, we discuss some common queueing policies in the transmit buffers of the CAN controllers. In Section IV, we describe the scheduling model. In Section V, we extend the existing analysis. Section VI presents the case study and evaluation of the extended analysis. Finally, Section VII summarizes and concludes the paper.

II. MIXED TRANSMISSION PATTERNS SUPPORTED BY THE HIGHER-LEVEL PROTOCOLS FOR CAN

In order to be consistent throughout the paper, we use the terms message and frame interchangeably. This is because we only consider messages that fit into one frame, i.e., the maximum size of a message can be 8 bytes. If a message is queued for transmission at periodic intervals, we use the term “Period” to refer to its periodicity. On the other hand, a sporadic message is queued for transmission as soon as a sporadic event occurs that changes the value of one or more signals contained in the message provided the Minimum Update Time (MUT\textsuperscript{3}) between the queuing of two successive sporadic messages has elapsed. The seminal RTA for CAN [16] and most of its extensions assume that the tasks queuing CAN messages are invoked either periodically or sporadically. However, there are some higher-level protocols and commercial extensions of CAN in which the tasks that queue the messages can be invoked periodically as well as sporadically. If a message is queued for transmission periodically as well as sporadically, the transmission type of a message is called mixed. That is, a mixed message is simultaneously time- and event-triggered. We identify three different types of implementations of the mixed messages by the higher-level protocols for CAN that are used in the automotive industry.

\textsuperscript{2}It should be noted that a node or ECU contains a CAN controller. We overload the terms node, processor, Electronic Control Unit (ECU), and CAN controller throughout the paper.

\textsuperscript{3}We overload the term MUT to refer to the Inhibit Time in the CANopen protocol [7] and the Minimum Delay Time (MDT) in the AUTOSAR communication [29].
A. Implementation of mixed message in the CANopen protocol

The CANopen protocol [7] supports mixed transmission that corresponds to the Asynchronous Transmission Mode coupled with Event Timer. The Event Timer is used for cyclic transmission of an asynchronous message. The mixed message in this protocol can be queued for transmission at the arrival of a sporadic event provided the Inhibit Time has expired. The Inhibit Time is the minimum time that must be allowed to elapse between the queuing of two consecutive messages. The mixed message can also be queued periodically when the Event Timer expires. The Event Timer is reset every time the message is queued. Once the mixed message is queued, any additional queuing of this message will not take place during the Inhibit Time [7]. The transmission pattern of the mixed message in the CANopen protocol is illustrated in Fig. 2(a). The down-pointing arrows show queuing of the message while the numbers below them represent the instance number of the queued message. The upward lines labeled with alphabetic characters represent the arrival of events. Instance 1 of the mixed message is queued as soon as the event A arrives. Both the Event Timer and Inhibit Time are reset. As soon as the Event Timer expires, instance 2 is queued due to periodicity and both the Event Timer and Inhibit Time are reset again. Instance 3 of the mixed message is immediately queued upon arrival of the event B because the Inhibit Time has already expired. Note that the Event Timer is also reset at the same time when instance 3 is queued as shown in Fig. 2(a). The instance 4 of the mixed message is queued because of the expiry of the Event Timer. There exists a dependency relationship between the Inhibit Time and the Event Timer, i.e., the Event Timer is reset with every sporadic transmission.

B. Implementation of mixed message in the AUTOSAR communications

AUTOSAR (AUTomotive Open System ARchitecture) [30] can be viewed as a higher-level protocol if it uses CAN for network communication. Mixed transmission in AUTOSAR is widely used in practice. In this protocol, a mixed message can be queued for transmission periodically with the mixed transmission mode time period. The mixed message can also be queued at the arrival of an event provided the Minimum Delay Time (MDT) has been expired. However, each transmission of the mixed message, regardless of being periodic or sporadic, is limited by the MDT timer. This means that both periodic and sporadic transmissions will always be delayed until the expiry of the MDT timer. Fig. 2(b) shows the transmission pattern of the mixed message implemented by AUTOSAR. The MDT timer is started as soon as the first instance of the mixed message is queued due to partly periodic nature of the mixed message. Its second instance is queued immediately upon arrival of the event A because the MDT timer has already expired. The next periodic transmission is scheduled 2 time units after the transmission of instance 2. However, the next two periodic transmissions corresponding to instances 3 and 4 are delayed because the MDT timer is still running.
transmissions that are delayed due to non-expiry of the $MDT$ timer are identified in Fig. 2(b). The periodic transmissions corresponding to instances 5 and 6 take place at the scheduled times because the $MDT$ timer is already expired in both cases.

C. Implementation of mixed message in the HCAN protocol

The mixed message in the HCAN protocol [8] contains signals out of which some are periodic and some are sporadic. The 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 $MUT$ between the queuing of two successive sporadic instances of the mixed message has elapsed. Hence, the transmission of the mixed message due to arrival of events is constrained by the $MUT$. The transmission pattern of mixed message in the HCAN protocol is illustrated in Fig. 2(c). Instance 1 of the mixed message is queued because of periodicity. As soon as event $A$ arrives, instance 2 is queued. When event $B$ arrives, the next instance of the mixed message is not queued immediately because the $MUT$ is not expired yet. As soon as the $MUT$ expires, the third instance is queued. The third instance contains the signal changes that correspond to event $B$. Similarly, the next instance of the mixed message is not immediately queued when the event $C$ arrives because the $MUT$ is not expired. Instance 4 of the mixed message is queued because of periodicity. Although, the $MUT$ was not expired, the event signal corresponding to event $C$ was packed in instance 4 and queued as part of the periodic message. Hence, there is no need to queue an additional sporadic instance of the mixed message when the $MUT$ expires. This indicates that the periodic transmission of a mixed message cannot be interfered by its sporadic transmission. This is a unique property of the HCAN protocol. When the event $D$ arrives, a sporadic instance of the mixed message is immediately queued as message 5 because the $MUT$ has already expired. Instance 6 is queued due to partly periodic nature of the mixed message.

D. Comparison of the three implementations of mixed message

In the first implementation method, the Event Timer is reset every time the mixed message is queued for transmission. The implementation of the mixed message in method 2 is similar to method 1 to some extent. The main difference is that the periodic transmission can be delayed until the expiry of the $MDT$ in method 2. Whereas in method 1, the periodic transmission is not delayed, in fact, the Event Timer is restarted with every sporadic transmission. The $MDT$ timer is started with every periodic or sporadic transmission of the mixed message. Hence, the worst-case periodicity of the mixed message in methods 1 and 2 can never be higher than the Inhibit Timer and the $MDT$ respectively. Therefore, the existing analyses for CAN with FIFO queues [25], [22], [26] hold intact. However, the periodic transmission is independent of the sporadic transmission in the third implementation method. The periodic timer is not reset with every sporadic transmission. The mixed message can be queued for transmission even if the $MUT$ is not expired. The worst-case periodicity of the mixed message is neither bounded by the period nor by the $MUT$. Therefore, the existing analyses for CAN with FIFO queues [25], [22], [26] cannot be applied to the mixed messages in the third implementation method.

III. Common queueing policies used in the CAN controllers

The timing behavior of CAN messages is influenced by many factors including the type of queueing policies implemented by the CAN device drivers and communication stack. The most common queueing policies in the nodes connected to the CAN network are priority- and FIFO-ordered policies.

A. Priority-ordered queues

The CAN protocol implements priority-based arbitration for the transmission of messages on the network. This means, each node selects the highest priority message from its transmit buffers while entering into the bus arbitration. The highest priority message among the messages selected from each node wins the arbitration, i.e., the right to transmit over the network. Intuitively, the most natural queueing policy suited to CAN controllers is priority-ordered queueing.

Let us consider an example to demonstrate the priority-based queueing policy as shown in Fig. 3. Let there be three nodes namely Node A, Node B and Node C that are connected to a single CAN network. Each node sends three messages over the network. Node A sends the messages $m_1$, $m_3$ and $m_5$; Node B sends the messages $m_2$, $m_4$ and $m_9$; whereas, Node C sends the messages $m_6$, $m_7$ and $m_8$. The subscript in the name of a message represents its priority. We assume that the smaller the value of the subscript, the higher the priority.
of the message. Intuitively, \( m_1 \) is the highest priority message, whereas, \( m_9 \) is the lowest priority message in the system.

In order to simplify the example, assume that the transmission periods of all messages are very high compared to their transmission times. Assume that all messages in each node are queued for transmission. We also assume that there cannot be multiple instances of a message queued for transmission at the same time.

Let the nodes implement priority queues. Each node selects the highest priority message from its queue to enter into bus arbitration. In the first round, Nodes A, B, and C select messages \( m_1, m_2 \) and \( m_3 \) respectively. Message \( m_1 \) wins the arbitration 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 \( m_4 \) respectively. This time, message \( m_2 \) wins the arbitration and is transmitted over the network. Similar priority-based selection and arbitration continue during the rest of the rounds as shown in Fig. 4.

C. Effect of queueing policy on the response times of messages

When FIFO queues are used, the priorities of messages are often not respected in the transmit queue within a node, e.g., the lower priority message \( m_5 \) is transmitted before the highest priority message \( m_1 \) as shown in Fig. 5. As a result, priority inversion can occur due to which higher priority messages may have very large response times. This becomes evident by comparing the response time of message \( m_2 \) in the systems with priority and FIFO queues as shown in Fig. 4 and Fig. 5 respectively.

IV. SYSTEM MODEL

The system scheduling model is based on the seminal model in [16] and its extensions for FIFO queues [25] and mixed messages [21]. The system consists of a number of nodes connected to a single CAN network. A node may implement a priority queue or a FIFO queue. In the former case, the node is designated as a PQ-node and it enters the highest priority message from its transmit queue in the bus arbitration. Whereas, in the later case the node is identified as an FQ-node and it enters the oldest message from its transmit queue in the bus arbitration.

Each CAN message \( m_n \) has a unique identifier and a priority denoted by \( ID_m \) and \( P_m \) respectively. The priority of a message is assumed to be equal to its ID. The priority of the message \( m_n \) is considered higher than the priority of another message \( m_k \) if \( P_{m_k} < P_{m_k} \). Let the sets \( hp(m_n) \), \( lp(m_n) \), and \( hop(m_n) \) contain the messages with priorities higher, lower, and equal and higher than \( m_n \) respectively. Although the priorities of CAN messages are unique, the set \( hop(m_n) \) is used in the case of mixed messages.

Associated to each message is a \( FRAME\_TYPE \) that specifies whether the frame is a standard or an extended CAN frame. The difference between the two frame types is that the standard CAN frame uses an 11-bit identifier whereas the extended CAN frame uses a 29-bit identifier. In order to keep the notations simple and consistent, we define a function \( \xi_m \) that denotes the transmission type of a message. \( \xi_m \) specifies
whether $m_m$ is periodic ($P$), sporadic ($S$) or mixed ($M$). Formally, the domain of $\xi_m$ can be defined as follows.

$$\xi_m \in \{P, S, M\}$$

Each message $m_m$ has a transmission time $C_m$ and queueing jitter $J_m$, which is inherited from the task that queues $m_m$, i.e., the sending task. We assume that $J_m$ can be smaller, equal or greater than $T_m$ or $MUT_m$. Each message can carry a data payload that ranges from 0 to 8 bytes. This integer value is specified in the header field of the frame called Data Length Code and is denoted by $s_m$. In the case of periodic transmission, $m_m$ has a transmission period which is denoted by $T_m$. Whereas, in the case of sporadic transmission, $m_m$ has the $MUT_m$ time. $B_m$ denotes the blocking time of $m_m$ which refers to the largest amount of time $m_m$ has to wait for the transmission of a lower priority message.

If an FQ-node transmits the message $m_m$ then the set of all messages transmitted by this node is defined by $M(m_m)$. The Lowest priority message in $M(m_m)$ is denoted by $L_m$. The sum of the transmission time of all the messages in $M(m_m)$ is identified by $C_{\text{SUM}}$. The transmission time of the shortest and longest messages in $M(m_m)$ are denoted by $C_{\text{MIN}}$ and $C_{\text{MAX}}$ respectively. $f_m$ denotes the maximum buffering time between the instant the message $m_m$ enters the FIFO queue and the instant it becomes the oldest message in the queue. It is equal to zero for a message belonging to a node that implements a priority queue [25].

We duplicate a message when its transmission type is mixed. Hence, each mixed message $m_m$ is treated as two separate messages, i.e., one periodic and the other sporadic. The duplicates share all the attributes except for $T_m$ and $MUT_m$. The periodic copy inherits $T_m$ while the sporadic copy inherits the $MUT_m$. Each message has a worst-case response time, denoted by $R_m$, and defined as the longest time between the queuing of the message (on the sending node) and the delivery of the message to the destination buffer (on the destination node). $m_m$ is deemed schedulable if its $R_m$ is less than or equal to its deadline $D_m$. The system is considered schedulable if all of its messages are schedulable.

We consider the deadlines to be arbitrary which means that they can be greater than the periods or $MUT$s of corresponding messages. We assume that the CAN controllers are capable of buffering more than one instance of a message. The instances of a message are assumed to be transmitted in the same order in which they are queued (i.e., we assume FIFO policy among the instances of the same message). For better readability, all the notations used in this paper are tabulated at the end of the paper.

V. EXTENDED ANALYSIS

We extend the existing analysis of CAN with both PQ-nodes and FQ-nodes [25] by adapting the RTA of CAN for mixed messages [21]. Let the message under analysis be denoted by $m_m$. The extended analysis treats a message differently based on its transmission type. Here we consider two different cases. In the first case, $m_m$ is assumed to be a periodic or a sporadic message. Whereas, $m_m$ is considered to be a mixed message in the second case.

A. Case 1: when $m$ is a periodic or a sporadic message

Consider $m_m$ to be a periodic or a sporadic message. We calculate the worst-case response time of a message differently depending upon the type of the queueing policy implemented in the sending node. That is, we treat the message under analysis differently for the PQ-and FQ-nodes. Therefore, once again, we consider two cases: (a) the first case assumes that $m$ belongs to a node that implements priority queue, (b) the second case considers that $m$ belongs to a node that implements FIFO queue.

1) Case 1 (a): when $m$ belongs to a priority-queued node:

Since we consider arbitrary deadlines for messages, there can be more than one instance of $m_m$ that may become ready for transmission before the end of priority level-$m$ maximum busy period. The maximum busy period is the longest contiguous interval of time during which $m_m$ is unable to complete its transmission due to two reasons. First, the network is occupied by the higher priority messages. In other words, at least one message of priority level-$m$ or higher has not completed its transmission. Second, a lower priority message already started its transmission when $m_m$ is queued for transmission. The maximum busy period starts at the so-called critical instant. In a system where messages are scheduled without offsets, the critical instant corresponds to the point in time when all higher priority messages in the system are queued simultaneously with $m_m$ while their subsequent instances are queued after the shortest possible interval of time [18].

There can be another reason to check if more than one instance of $m_m$ is queued for transmission in the priority level-$m$ maximum busy period. Since, the message transmission in CAN is non-preemptive, the transmission of previous instance of $m_m$ could delay the current instance of a higher priority message that may add to the interference received by the current instance of $m_m$. This phenomenon was identified by Davis et al. [18] and termed as “push-through interference”. Because of this interference, a higher priority message may be waiting for its transmission before the transmission of the current instance of $m_m$ finishes. Hence, the length of busy period may extend beyond $T_m$ or $MUT_m$.

Intuitively, the response time of each instance of $m_m$ within priority level-$m$ maximum busy period should be calculated. The largest value among the calculated response times of all instances of $m_m$ is considered as the worst-case response-time of $m_m$. Let $q_m$ be the index variable to denote instances of $m_m$. The worst-case response time of $m_m$ is given by:

$$R_m = \max\{R_m(q_m)\}$$

Constituents of the worst-case response time. According to the existing analysis [16], [18], the worst-case response-time of any instance of $m_m$ consists of three parts as follows.

1) The queueing jitter denoted by $J_m$. It is inherited from the sending task, i.e., the task that queues $m_m$ for transmission. Basically, it represents the maximum variation in time between the release of the sending task and queuing of the message in the transmit queue (buffers). It
is calculated by taking the difference between the worst- and best-case response time of the sending task.

2) The worst-case transmission time denoted by $C_m$. It represents the longest time it takes for $m_m$ to be transmitted over the network.

3) The queueing delay denoted by $\omega_m$. It is equal to the longest time that elapses between the instant $m_m$ is queued by the sending task in the transmit queue and the instant when $m_m$ is about to start its successful transmission. In other words, $\omega_m$ is the interference caused by other messages to $m_m$.

Thus, the worst-case response time of any instance $q_m$ of a periodic or sporadic message $m_m$ is given by the following set of equations.

$$R_m(q_m) = \begin{cases} J_m + \omega_m(q_m) - q_m T_m + C_m, & \text{if } \xi_m = P \\ J_m + \omega_m(q_m) - q_m MUT_m + C_m, & \text{if } \xi_m = S \end{cases}$$

(2)

The terms $q_m T_m$ and $q_m MUT_m$ in (2) are used to support the response-time calculations for multiple instances of $m_m$. If the transmission type of $m_m$ is periodic then the message period is taken into account. However, if the transmission type of $m_m$ is sporadic, minimum update time is used in the above equation.

Calculations for the worst-case transmission time $C_m$. The worst-case transmission time of $m_m$ can be calculated using the method derived in [16] and later adapted in [18]. For the standard CAN identifier format, $C_m$ is calculated as follows.

$$C_m = \left( 47 + 8s_m + \left\lfloor \frac{34 + 8s_m - 1}{4} \right\rfloor \right) \tau_{bit}$$

(3)

Where $\tau_{bit}$ represents the time required to transmit a single bit of data on the CAN network. Its value depends upon the speed of the network. In (3), $47$ is the number of bits due to protocol overhead. It is composed of start of frame bit (1-bit), arbitration field (12-bits), control field (6-bits), Cyclic Redundancy Check (CRC) field (16-bits), acknowledgement (ACK) field (2-bits), End of Frame (EoF) field (7-bits), and inter-frame space (3-bits). The number of bits due to protocol overhead in the case of extended CAN frame format is equal to 67.

In [31], Broster identified that the analysis in [16], [18] uses 47-bits instead of 44-bits as the protocol overhead for a standard CAN identifier frame format. This is because the analysis in [16], [18] accounts only for 3-bit inter-frame space as part of the CAN frame. The 3-bit inter-frame space must be considered when calculating the interferences or blocking from other messages. However, Broster argued that this adds slight amount of pessimism to the response time of the message under analysis if the 3-bit inter-frame space is also considered in its transmission time. This is because the destination node can access the message before the inter-frame space. In order to avoid this pessimism, we subtract 3-bit time from the response time of the instance of the message under analysis.

The term $\left\lfloor \frac{34 + 8s_m - 1}{4} \right\rfloor$ in (3) is added to compensate for the extra time due to 'bit stuffing'. It should be noted that the bit sequences 0000000 and 111111 are used for error signals in CAN. In order to be unambiguous in non-erroneous transmission, a stuff bit of opposite polarity is added whenever there are five bits of the same polarity in the sequence of bits to be transmitted [18]. The value $\frac{34}{4}$ indicates that only 34-bits out of 47-bits protocol overhead are subject to bit stuffing. The term $\left\lfloor \frac{x}{4} \right\rfloor$ is the notation for floor function. It returns the largest integer that is less than or equal to $\frac{x}{4}$.

For the message with extended CAN identifier format, $C_m$ is calculated as follows.

$$C_m = \left( 67 + 8s_m + \left\lfloor \frac{54 + 8s_m - 1}{4} \right\rfloor \right) \tau_{bit}$$

(4)

The calculations for $C_m$ in (3) can be simplified as follows.

$$C_m = (55 + 10s_m) \tau_{bit}$$

(5)

Similarly, the calculations for $C_m$ in (4) can be simplified as follows.

$$C_m = (80 + 10s_m) \tau_{bit}$$

(6)

Calculations for the worst-case queueing delay $\omega_m$. The calculations for $\omega_m$ should include the interference caused by all the other periodic, sporadic and mixed messages. The existing analyses for CAN with FIFO queues [25], [22], [26] have a limitation that they consider the effect of interference from only periodic and sporadic messages.

It is important to mention that CAN uses fixed-priority non-preemptive scheduling, therefore, a message cannot be interfered by higher priority messages during its transmission on the bus. Whenever we use the term interference, it refers to the amount of time $m_m$ has to wait in the transmit queue because the higher priority messages win the arbitration, i.e., the right to transmit before $m_m$. For a message queued at a PQ-node, $\omega_m$ is calculated by the following fixed-point iteration.

$$\omega_m^{n+1}(q_m) = B_m + q_mC_m + \sum_{\forall m_k \in hp(m_m)} I_kC_k$$

(7)

The last term in (7) represents the interference from the higher priority messages. In order to solve this iterative equation, initial value of $\omega_m^n$ can be taken as follows.

$$\omega_m^0(q_m) = B_m + q_mC_m$$

(8)

The iterations in (7) stop either when the queueing delays in the previous and current iterations are equal or when the response time exceeds the deadline. Since, CAN uses fixed priority non-preemptive scheduling, any message can be blocked by only one message in the set of lower priority messages. Hence, the message under analysis can only be blocked by either the periodic copy or the sporadic copy of any lower priority mixed message. It should be noted that both the copies of a mixed message have the same transmission time, $C_m$. Hence, $B_m$ is equal to the largest transmission time among all periodic, sporadic and mixed messages in the set of lower priority messages with respect to $m_m$ and is given by the following equation.
\[
B_m = \max_{\forall m_k \in lp(m_m)} (C_k) \quad (9)
\]

A higher priority message \( m_k \) contributes an extra delay, equal to \( f_k \), to the worst-case queuing delay of \( m_m \) if \( m_k \) belongs to the FQ-node. \( f_k \) represents the delay after which the higher priority message \( m_k \) belonging to the FQ-node becomes the oldest message in the queue and can take part in the priority-based arbitration [25]. The existing analysis for mixed messages in CAN [21] does not take this additional delay into account. \( f_k \) is zero if \( m_k \) belongs to a PQ-node. We will come back to the calculations for \( f_k \) in Section V-C.

In (7), \( I_k \) is calculated differently for different values of \( \xi_k \) (\( k \) is the index of any higher priority message) as shown below. The interference by a higher priority mixed message contains the contribution from both the duplicates.

\[
I_k = \begin{cases} \frac{\omega_m(q_m) + q_m + f_k + \tau_{bit}}{T_k}, & \text{if } \xi_k = P \\ \frac{\omega_m(q_m) + J_m + f_k + \tau_{bit}}{MUT_k}, & \text{if } \xi_k = S \\ \frac{\omega_m(q_m) + J_m + f_k + \tau_{bit}}{T_k}, & \text{if } \xi_k = M \end{cases} \quad (10)
\]

**Length of the maximum busy period.** The length of priority level-\( m \) maximum busy period, denoted by \( t_m \), is given by the following equation. The effect of extra delay from the messages belonging to the FQ-nodes is also taken into account. \( t_m \) can be calculated by the following iterative equation.

\[
t_m^{n+1} = B_m + \sum_{\forall m_k \in hep(m_m)} I_k C_k \quad (11)
\]

\( I_k \) is given by the following relation. Note that the contribution of both the duplicates of a mixed message \( m_k \) in the set \( hep(m_m) \) is taken into account.

\[
I_k' = \begin{cases} \frac{t_m^{n+1} + f_k + f_k}{T_k}, & \text{if } \xi_k = P \\ \frac{t_m^{n+1} + f_k + f_k}{MUT_k}, & \text{if } \xi_k = S \\ \frac{t_m^{n+1} + f_k + f_k}{T_k} + \frac{t_m^{n+1} + f_k + f_k}{MUT_k}, & \text{if } \xi_k = M \end{cases} \quad (12)
\]

In order to solve the iterative equation (11), \( C_m \) can be used as the initial value of \( t_m^0 \) as shown below.

\[
t_m^0 = C_m \quad (13)
\]

The right hand side of (11) is a monotonic non-decreasing function of \( t_m \). Equation (11) is guaranteed to converge if the bus utilization for messages of priority level-\( m \) and higher, denoted by \( U_m \), is less than 1. That is,

\[
U_m < 1 \quad (14)
\]

where \( U_m \) is calculated by the following equation:

\[
U_m = \sum_{\forall m_k \in hep(m_m)} C_k I_k'' \quad (15)
\]

where \( I_k'' \) is given by the following relation:

\[
I_k'' = \begin{cases} \frac{1}{T_k}, & \text{if } \xi_k = P \\ \frac{1}{MUT_k}, & \text{if } \xi_k = S \\ \frac{1}{T_k} + \frac{1}{MUT_k}, & \text{if } \xi_k = M \end{cases} \quad (16)
\]

In the above equation, the contribution by both the copies of all mixed messages belonging to the set \( hep(m_m) \) is taken into account while calculating the bus utilization.

The number of instances of \( m_m \), denoted by \( Q_m \), that becomes ready for transmission before the busy period ends is given by the following equation (similar to the existing analysis for mixed messages).

\[
Q_m = \begin{cases} \frac{t_m + J_m}{T_m}, & \text{if } \xi_m = P \\ \frac{t_m + J_m}{MUT_m}, & \text{if } \xi_m = S \end{cases} \quad (17)
\]

The index of each message instance is identified by \( q_m \) and its range is given as follows.

\[
0 \leq q_m \leq (Q_m - 1) \quad (18)
\]

2) **Case 1 (b): when \( m \) belongs to a FIFO-queued node:** Similar to the existing RTA for CAN with FIFO queues [25], the extended analysis is FIFO-symmetric. This means that all the messages belonging to FQ-node will have same upper bound for their worst-case response times. In order to derive the worst-case response time of a periodic or sporadic message belonging to the FQ-node, we consider the worst-case conditions. Hence, we assume that the message under analysis is the lowest priority message, i.e., \( L_m \) in the group \( M(m_m) \) with the largest transmission time \( C_m^{MAX} \) (to maximize the interference from the messages in \( M(m_m) \) as well as from the messages belonging to other nodes). The response time of a particular instance \( q_m \) of a periodic or sporadic message \( m_m \) that is queued at the FQ-node is given by the following equation.

\[
R_m(q_m) = \begin{cases} J_m + \omega_m(q_m) - q_m T_m + C_m^{MAX}, & \text{if } \xi_m = P \\ J_m + \omega_m(q_m) - q_m MUT_m + C_m^{MAX}, & \text{if } \xi_m = S \end{cases} \quad (19)
\]

In [25], message deadlines are assumed to be equal to or less than the corresponding periods. Hence, for any message \( m_m \) belonging to \( M(m_m) \) in the FQ-node, there could be only one instance of every other message queued ahead of \( m_m \). In the existing analysis, the maximum amount of interference received by \( m_m \) before it becomes the oldest message in the FIFO queue and ready to take part in the priority-based arbitration is bounded by \( (C_m^{SUM} - C_m^{MIN}) \). This interference bound may not be applicable in our case because we assume
that the messages have arbitrary deadlines which means that they can be greater than the periods or minimum update times of the corresponding messages. Therefore, it is possible to have more than one instance of any higher priority message queued ahead of \( m_m \) in the FIFO queue. This is the reason we select the transmission time of \( m_m \) in FIFO-queued nodes to be equal to \( C_{m_m}^{MAX} \) instead of \( C_{m_m}^{MIN} \).

**Interference received by \( m_m \) from the messages in \( M(m_m) \).**

Now, we derive an upper bound for the number of instances of each message in the group \( M(m_m) \) that can be queued ahead of \( m_m \). Consider a simple but intuitive example as shown in Fig. 6. Let the message under analysis be \( m_m \) (lowest priority message in \( M(m_m) \)). Also consider an arbitrary message \( m_i \) belonging to the group \( M(m_m) \). Assume both \( m_i \) and \( m_m \) are periodic and have same transmission times. We consider four different cases with respect to the relationship between message periods as shown in Fig. 6. In case (a), \( T_i \) is smaller than \( T_m \). In case (b), \( T_i \) is equal to \( T_m \). In case (c), \( T_i \) is greater than \( T_m \). In case (d), \( T_i \) is smaller than \( T_m \) and at the same time \( T_m \) is an integer multiple of \( T_i \). These cases essentially cover all the cases required to derive the upper bound on the maximum number of instances of \( m_i \) queued ahead of any instance of \( m_m \).

![Fig. 6. Demonstration of maximum interference on \( m_m \) from the messages in the group \( M(m_m) \).](image)

The periods of \( m_i \) and \( m_m \) in each case are shown in Fig. 6. The left hand side of Fig. 6 shows the time line during which each instance of \( m_i \) and \( m_m \) is queued in the FIFO queue. Whereas, the right hand side of Fig. 6 depicts the corresponding FIFO queue as if none of the messages was transmitted. The maximum number of instances of \( m_i \) that are queued ahead of any instance of \( m_m \) in the FIFOs are 3, 1, 1 and 2 in the case (a), (b), (c) and (d) respectively. Let \( Q_i \) denote the maximum number of instances of \( m_i \) in the group \( M(m_m) \) that can be queued ahead of any instance of \( m_m \) in the FIFO queue. We can generalize \( Q_i \) for all the cases as follows.

\[
Q_i = \left\lfloor \frac{T_m + J_i}{T_i} \right\rfloor \tag{20}
\]

Let us consider the effect of jitter of \( m_i \), denoted by \( J_i \), on the interference of \( m_m \). Because of \( J_i \), additional instances of \( m_i \) can be queued ahead of \( m_m \). Thus, taking the effect of jitter into account, (20) can be written as:

\[
Q_i = \left\lfloor \frac{T_m + J_i}{T_i} \right\rfloor \tag{21}
\]

Since, \( m_i \) can be periodic, sporadic or mixed, we can generalize (21) as follows.

\[
Q_i = \begin{cases} 
\left\lfloor \frac{T_m + J_i}{T_i} \right\rfloor, & \text{if } \xi_i = P \\
\left\lfloor \frac{T_m + J_i}{MUT_i} \right\rfloor, & \text{if } \xi_i = S \\
\left\lfloor \frac{T_m + J_i}{MUT_i} \right\rfloor + \left\lfloor \frac{T_m}{MUT_i} \right\rfloor, & \text{if } \xi_i = M
\end{cases} \tag{22}
\]

**Calculations for the worst-case queueing delay.** The worst-case queueing delay, \( \omega_m \), in (19) can be calculated in a similar fashion as in (7) with the addition of extra delay shown in (22).

\[
\omega^{n+1}_m(q_m) = B_{L_m} + \sum_{\forall m_i \in M(m_m) \land i \neq m} Q_i C_i + q_m C_{m_m}^{MAX} + \sum_{\forall m_k \in L(L_m) \land m_k \notin M(m_m)} I_k C_k \tag{23}
\]

Where \( m_k \) is any message that has priority greater than the lowest priority message in the FQ-node in which \( m_m \) is queued. Moreover, \( m_k \) does not belong to the FQ-node in which \( m_m \) is queued. \( m_k \) is any message, other than \( m_m \), in the group \( M(m_m) \). \( B_{L_m} \) is the blocking time of \( L_m \) which refers to the maximum transmission time of a message in the set of messages with lower priority than \( L_m \) that are sent by the other nodes. Since, the interference contributed to \( m_m \) by higher priority messages from other nodes (both PQ and FQ) is independent of \( m_m \) belonging to a PQ-node or FQ-node, \( I_k \) can be calculated using (10). The initial value of \( \omega^0_m \) to solve the iterative equation (23) can be selected as follows.

\[
\omega^0_m = B_{L_m} + \sum_{\forall m_i \in M(m_m) \land i \neq m} Q_i C_i + q_m C_{m_m}^{MAX} \tag{24}
\]

**Length of the maximum busy period.** The length of priority level-\( m \) maximum busy period, denoted by \( t_m \), can be calculated in a similar fashion as in (11) and by following the intuition from (23). The effect of extra delay from the messages belonging to the FQ-nodes is also taken into account. \( t_m \) can be calculated by the following iterative equation.

\[
t^{n+1}_m = B_{L_m} + \sum_{\forall m_i \in M(m_m) \land i \neq m} Q_i C_i + \sum_{\forall m_k \in L(L_m) \land m_k \notin M(m_m)} I_k^C C_k \tag{25}
\]

The initial value for \( t^0_m \) can be selected using (13). Since, the interference to \( m_m \) by higher priority messages from other nodes (both PQ and FQ) is independent of \( m_m \) belonging to a PQ-node or FQ-node, \( I_k \) can be calculated using (12). Similarly, the total number of instances of \( m_m \) that becomes ready for transmission before the busy period ends can be calculated using (17). The worst-case response time of \( m_m \) is the largest value of response time among all its instances as shown in (1).
B. Case 2: when \( m \) is a mixed message

When the message under analysis is mixed, we treat it as two separate message streams, i.e., each mixed message is duplicated as the periodic and sporadic messages. The response times of both the duplicates are calculated separately. For simplicity, we denote the periodic and sporadic copies of a mixed message \( m_m \) by \( m_{mp} \) and \( m_{ms} \) respectively. Let the worst-case response time of \( m_{mp} \) and \( m_{ms} \) be denoted by \( R_{mp} \) and \( R_{ms} \) respectively. The worst-case response time of \( m_m \) is equal to the largest value between \( R_{mp} \) and \( R_{ms} \) as given by the following equation.

\[
R_m = \max(R_{mp}, R_{ms})
\]  

(26)

1) Case 2 (a): when \( m \) belongs to a priority-queued node:

For a priority-queued mixed message, the response times of each instance of \( m_{mp} \) and \( m_{ms} \) are calculated separately by adapting the existing analysis for mixed messages in CAN [21]. Let us denote the total number of instances of \( m_{mp} \) and \( m_{ms} \), occurring in the priority level-\( m \) maximum busy period, by \( Q_{mp} \) and \( Q_{ms} \) respectively. Assume that the index variable for message instances of \( m_{mp} \) and \( m_{ms} \) is denoted by \( q_{mp} \) and \( q_{ms} \) respectively. Their ranges are given by the following equations.

\[
0 \leq q_{mp} \leq (Q_{mp} - 1)
\]  

(27)

\[
0 \leq q_{ms} \leq (Q_{ms} - 1)
\]  

(28)

The worst-case response time of \( m_{mp} \) is equal to the largest value among the response times of all of its instances occurring in the busy period as shown by the following equation.

\[
R_{mp} = \max(R_{mp}(q_{mp}))
\]  

(29)

Similarly, the worst-case response time of \( m_{ms} \) is equal to the largest value among the response times of all of its instances occurring in the busy period. It is given by the following equation.

\[
R_{ms} = \max(R_{ms}(q_{ms}))
\]  

(30)

The worst-case response time of each instance of \( m_{mp} \) and \( m_{ms} \) can be derived by adapting the equations for the calculation of worst-case response time of periodic and sporadic messages respectively (derived in the first case) as given by the following two equations.

\[
R_{mp}(q_{mp}) = J_m + \omega_{mp}(q_{mp}) - q_{mp}T_m + C_m
\]  

(31)

\[
R_{ms}(q_{ms}) = J_m + \omega_{ms}(q_{ms}) - q_{ms}MUT_m + C_m
\]  

(32)

The queueing jitter, \( J_m \), is the same (equal) in both the equations (31) and (32). The transmission time, \( C_m \), is also the same in these equations and is calculated using (5) or (6) depending upon the type of CAN frame identifier. Although, both the duplicates of \( m_m \) inherit same \( J_m \) and \( C_m \) from it, they experience different amount of worst-case queueing delays caused by other messages.

Calculations for the worst-case queueing delay. The worst-case queueing delay experienced by \( m_{mp} \) and \( m_{ms} \) is denoted by \( \omega_{mp} \) and \( \omega_{ms} \) in (31) and (32) respectively. \( \omega_{mp} \) and \( \omega_{ms} \) can be calculated by adapting the equation for the calculations of worst-case queueing delay in (7). However, in this equation we need to add the effect of self interference in a mixed message. By self interference we mean that the periodic copy of a mixed message can be interfered by the sporadic copy and vice versa. Since, both \( m_{mp} \) and \( m_{ms} \) have equal priorities, any number of instances of \( m_{mp} \) queued ahead of \( m_{ms} \) contribute an extra delay to the worst-case queueing delay experienced by \( m_{ms} \) and vice versa. We adapt the calculations for self interference in a mixed message that we derived in [21]. The worst-case queueing delay for \( m_{mp} \) and \( m_{ms} \) can be calculated using the following equations.

\[
\omega_{mp}^{n+1}(q_{mp}) = B_m + q_{mp}C_m + \sum_{\forall m \in hp(m_m)} I_{k_p}C_k + Q_{mp}^P C_m
\]  

(33)

\[
\omega_{ms}^{n+1}(q_{ms}) = B_m + q_{ms}C_m + \sum_{\forall m \in hp(m_m)} I_{k_s}C_k + Q_{ms}^S C_m
\]  

(34)

The effect of self interference can be seen in the last terms of (33) and (34). \( Q_{mp}^P \) denotes the total number of instances of \( m_{mp} \) that are queued ahead of \( q_{mp} \) instance of \( m_{mp} \). Similarly, \( Q_{ms}^S \) denotes the total number of instances of \( m_{mp} \), that are queued ahead of \( q_{ms} \) instance of \( m_{ms} \). The values of \( Q_{mp}^P \) and \( Q_{ms}^S \) are calculated as follows.

\[
Q_{mp}^P = \begin{cases} 
\left[ \frac{q_{mp}T_m + J_m + \tau_{kmp}}{MUT_m} \right] , & \text{if } (q_{mp} = 0) \& \& (J_m = 0) \\
\left[ \frac{q_{mp}T_m + J_m}{MUT_m} \right] , & \text{otherwise}
\end{cases}
\]  

(35)

\[
Q_{ms}^S = \begin{cases} 
\left[ \frac{q_{ms}MUT_m + J_m + \tau_{kms}}{T_m} \right] , & \text{if } (q_{ms} = 0) \& \& (J_m = 0) \\
\left[ \frac{q_{ms}MUT_m + J_m}{T_m} \right] , & \text{otherwise}
\end{cases}
\]  

(36)

In order to solve the iterative equations (33) and (34), the initial values of \( \omega_{mp}^0(q_{mp}) \) and \( \omega_{ms}^0(q_{ms}) \) can be selected according to (8) in a similar fashion. \( I_{k_p} \) and \( I_{k_s} \) are calculated using the following equations.
\[
I_{k_P} = \begin{cases} 
\frac{\omega_{m_p}(q_{m_P}) + J_k + f_k + \tau_{ist}}{T_k}, & \text{if } \xi_k = P \\
\frac{\omega_{m_p}(q_{m_P}) + J_k + f_k + \tau_{ist}}{MUT_k}, & \text{if } \xi_k = S \\
\frac{\omega_{m_p}(q_{m_P}) + J_k + f_k + \tau_{ist}}{MUT_k} + f_k, & \text{if } \xi_k = M
\end{cases}
\]

\[
I_{k_S} = \begin{cases} 
\frac{\omega_{m_S}(q_{m_S}) + J_k + f_k + \tau_{ist}}{T_k}, & \text{if } \xi_k = P \\
\frac{\omega_{m_S}(q_{m_S}) + J_k + f_k + \tau_{ist}}{MUT_k}, & \text{if } \xi_k = S \\
\frac{\omega_{m_S}(q_{m_S}) + J_k + f_k + \tau_{ist}}{MUT_k} + f_k, & \text{if } \xi_k = M
\end{cases}
\]

The values of \(I_{k_P}\) and \(I_{k_S}\) in (37) and (38) differ from those calculated in [21] in a way that we consider an extra jitter, i.e., \(f_k\) from every message that belongs to the FQ-node.

**Calculations for the length of the maximum busy period.**

The length of priority level-\(m\) maximum busy period, denoted by \(t_m\), can be calculated using (11) that is developed for periodic and sporadic messages in a PQ-node. This is because (11) takes into account the effect of queueing delay from all the higher and equal priority messages. Since, the duplicates of a mixed message inherit the same priority from it, the contribution of queueing delay from the duplicate is also covered in this equation. Therefore, there is no need to calculate \(t_m\) for \(m_p\) and \(m_s\) separately. It should be calculated only once for a mixed message.

Although \(t_m\) is the same for \(m_p\) and \(m_s\), the number of instances of both the messages that become ready for transmission just before the end of the maximum busy period, i.e., \(Q_{m_P}\) and \(Q_{m_s}\) respectively, may be different. The reason is that the calculations for \(Q_{m_P}\) and \(Q_{m_s}\) require \(T_m\) and \(MUT_m\) respectively and which may have different values. \(Q_{m_P}\) and \(Q_{m_s}\) can be calculated by adapting (17) that is derived for the calculation of the number of instances of periodic and sporadic messages that become ready for transmission before the end of the busy period. \(Q_{m_P}\) and \(Q_{m_s}\) are given by the following equations.

\[
Q_{m_P} = \left[ \frac{t_m + J_m}{T_m} \right]
\]

\[
Q_{m_s} = \left[ \frac{t_m + J_m}{MUT_m} \right]
\]

2) Case 2 (b): when \(m\) belongs to a FIFO-queued node:

The worst-case response times of each instance of \(m_p\) and \(m_s\) queued at the FQ-node are calculated similar to the case of FIFO-queued messages that are periodic or sporadic.

\[
R_{m_P}(q_{m_P}) = J_m + \omega_{m_p}(q_{m_P}) - q_{m_P}T_m + C_{m_P}^{MAX}
\]

\[
R_{m_S}(q_{m_S}) = J_m + \omega_{m_S}(q_{m_S}) - q_{m_S}MUT_m + C_{m_S}^{MAX}
\]

**Calculations for the worst-case queueing delay.**

The worst-case queueing delays for \(m_p\) and \(m_s\) are calculated by adapting the calculations in the equations (23), (33) and (34) as follows.

\[
\omega^{n+1}_{m_P}(q_{m_P}) = B_{L_m} + \sum_{\forall m_s \in (M(m_m) \cup M(m_s))} Q_{m_P}C_{m_P}^{MAX} + \sum_{\forall m_k \in hp(L_m)} I_{k_P}C_k + Q_{m_s}^{P}C_{m_s}^{MAX}
\]

\[
\omega^{n+1}_{m_S}(q_{m_S}) = B_{L_m} + \sum_{\forall m_s \in (M(m_m) \cup M(m_s))} Q_{m_S}C_{m_S}^{MAX} + \sum_{\forall m_k \in hp(L_m)} I_{k_S}C_k + Q_{m_s}^{S}C_{m_s}^{MAX}
\]

Since, the interference caused by higher priority messages from other PQ- and FQ-nodes is independent of the mixed message \(m\) belonging to a PQ-node or FQ-node, \(I_{k_P}\) and \(I_{k_S}\) can be calculated using (37) and (38). The initial values of \(\omega_{m_P}\) and \(\omega_{m_S}\) can be selected according to (24) while considering the respective index of each instance of \(m_p\) and \(m_s\). The value of \(Q_s\) is calculated using (22) similar to the case of FIFO queued periodic or sporadic messages. The values of \(Q_{m_p}^{P}\) and \(Q_{m_s}^{P}\) are calculated using (35) and (36) that are derived for mixed message in a PQ-node. \(Q_{m_s}^{FQ}\) denotes the total number of instances of \(m_s\) that are queued ahead of \(q_{m_p}^{th}\) instance of \(m_p\). Therefore, we consider only queuing jitter in (35) and do not take into account any additional delay that may occur after queueing of \(m_p\) such as \(f_m\). Similar arguments hold for \(Q_{m_s}^{S}\).

**Calculations for the length of the maximum busy period.**

The length of priority level-\(m\) maximum busy period, denoted by \(t_m\), can be calculated by using (25) that is developed for periodic and sporadic messages in a FQ-node. This is because (25) takes into account the effect of queueing delay from all the higher and equal priority messages. Since, the duplicates of a mixed message inherit the same priority from it, the contribution of the queueing delay from the duplicate is also covered in (25). Therefore, there is no need to calculate \(t_m\) for \(m_p\) and \(m_s\) separately. It should be calculated only once for a mixed message.

Although the length of the busy period is the same, the number of instances of \(m_p\) and \(m_s\) that become ready for transmission just before the end of the maximum busy period, i.e., \(Q_{m_P}\) and \(Q_{m_s}\) respectively, may be different. \(Q_{m_P}\) and \(Q_{m_s}\) can be calculated by following the same reasoning and using the equations that we derived for a mixed message in the PQ-node in (39) and (40) respectively.
C. Algorithm for the calculations of maximum buffering time in FIFO queues

The algorithm for the calculations of maximum buffering time in FIFO queues is adapted from [25] to support mixed messages in CAN with FIFO queues. The buffering time for any priority-queued message is equal to zero. It should be noted that the calculations for the response times in equations (2), (19), (31), (32), (41) and (42) are dependent upon the corresponding iterative calculations for the queueing delays in (7), (23), (33), (34), (43) and (44) respectively. Whereas, the calculations for queueing delay depends upon the maximum buffering time. Therefore, the response times and maximum buffering times should be calculated iteratively and simultaneously as shown in Algorithm 1.

Algorithm 1 Algorithm for the calculations of maximum buffering times and message response times simultaneously.

1: begin  
2: for all Messages in the system do  
3: \( f_m \leftarrow 0 \) // Initialize buffering time for all messages.  
4: end for  
5: Repeat \( \leftarrow \) TRUE  
6: while Repeat = TRUE do  
7: \( \text{Repeat} \leftarrow \text{FALSE} \)  
8: for Every message \( m_m \) in the system do  
9: if \( m_m \in \text{ECU with FIFO queue} \) then  
10: if Message type of \( m_m \) == PERIODIC or SPORADIC then  
11: \( \text{CALCULATE } R_m \text{ USING EQUATION (19)} \)  
12: else if Message type of \( m_m \) == MIXED then  
13: \( \text{CALCULATE } R_m \text{ USING EQUATIONS (41)} \)  
14: end if  
15: if \( R_m > D_m \) then  
16: \( m_m \) IS UNSCHEDULABLE  
17: end if  
18: if \( f_m < \omega_m \) then  
19: \( f_m \leftarrow \omega_m \)  
20: \( \text{Repeat} \leftarrow \text{TRUE} \)  
21: end if  
22: else if \( m_m \in \text{ECU with priority queue} \) then  
23: \( f_m \leftarrow 0 \) // buffering time for a priority queued message is always zero.  
24: if Message type of \( m_m \) == PERIODIC or SPORADIC then  
25: \( \text{CALCULATE } R_m \text{ USING EQUATION (2)} \)  
26: else if Message type of \( m_m \) == MIXED then  
27: \( \text{CALCULATE } R_m \text{ USING EQUATIONS (31)} \)  
28: end if  
29: if \( R_m > D_m \) then  
30: \( m_m \) IS UNSCHEDULABLE  
31: end if  
32: end if  
33: end for  
34: end while  
35: end

VI. CASE STUDY AND EVALUATION

In this section, we conduct an automotive-application case study. Basically, we adapt the case study of the experimental vehicle that is analyzed for only periodic messages in [32]. We implemented\(^4\) the extended analysis in a freely-available tool MPS-CAN Analyzer [27]. Using this tool, we compare and evaluate the response times of periodic, sporadic and mixed messages in the experimental vehicle using the extended analysis for mixed messages in CAN with FIFO queues and the existing analysis for CAN with priority queues.

A. Experimental setup

There are six ECUs in the experimental vehicle that are connected to a single CAN network. The selected speed for CAN is 500 Kbit/s. There are 81 CAN messages in the system; out of which 27 are periodic, 27 are sporadic, while the remaining 27 are mixed. All the attributes of these messages are shown in the table depicted in Fig. 7. The attributes of each message are identified as follows. The priority, transmission type, number of data bytes in the message, transmission period, and minimum update time are represented by \( P_m, \xi_m, s_m, T_m, \text{and } \text{MUT}_m \) respectively. We assume, the smaller the value of the \( P_m \) parameter of a message, the higher its priority. Accordingly, the message with priority 1 is the highest priority message, whereas the message with priority 81 is the lowest priority message in the system. All timing parameters are in microseconds. We perform two sets of experiments. In the first set, all ECUs in the system implement priority queues. In the second set of experiments, all ECUs implement FIFO queues. In both sets of experiments, each ECU has 32 buffers in the transmit queue.

B. Comparison of various response-time analyses

In the first set of experiments, the response times of all messages are calculated using the existing response-time analysis for mixed, periodic and sporadic messages in CAN with priority queues [21]. The calculated response times are denoted by \( R_m[Prio] \) in the table in Fig. 7. On the other hand, in the second set of experiments, the response times of all messages are calculated using the extended analysis presented in the previous section. The calculated response times in this case are denoted by \( R_m[FIFO] \) in the table in Fig. 7. The maximum network bandwidth utilization calculated in both cases is equal to 33.776970%.

The response times of all messages in these two cases are shown by the bar graphs in Fig. 8. The first bar (solid black bar) in each set of the two bars represents the response time of a message in the system where all ECU’s implement priority queues. Whereas, the second bar (pattern bar) in each set of the two bars represents the response time of a message in the system where all ECUs implement FIFO queues.

The response-time graphs show that the message response times are the best (smallest) in the case when all the ECUs use priority-based queueing policy. On the other hand, the response

\(^4\)The discussion about the implementation in the tool is not in the scope of this paper.
times of the messages are the worst (largest) in the system where the ECUs implement FIFO queues. In fact, the response times are significantly large in the case of FIFO queues. This is because of the priority inversion in FIFO queues (discussed in Section III-C). Moreover, the worst-case buffering time in the FIFO queues can be significantly large that adds to the worst-case response times of the messages.

C. Discussion and recommendations

In order to get short response times of CAN messages, those ECUs should be selected that implement priority-based queueing policy. Although FIFO policy is simple and easy to implement, configure, and use as compared to the priority queueing policy, the messages can have very large worst-case response times in the case of ECUs implementing FIFO queues as shown in Fig. 8. The ECUs which implement priority-based queueing policy should be preferred over the ECUs which implement FIFO queues especially in the systems that have high network utilization and short stimulus-to-response requirements. Moreover, it is important to use the right RTA that correctly matches the queueing policies in the ECUs; and transmission type of messages used in the higher-level protocols. If these constraints are not rightly considered in the RTA, the calculated response times can be optimistic.

VII. SUMMARY AND CONCLUSION

The existing worst-case response-time analysis for messages in Controller Area Network (CAN) with priority- and FIFO-queued nodes does not support the analysis of mixed messages. A mixed message can be queued both periodically and sporadically, i.e., it may not have a periodic activation pattern. Mixed
messages are implemented by several high-level protocols based on CAN that are used in the automotive industry. We identified three different implementations of mixed messages in higher-level protocols for CAN. For some implementations, the existing analysis still provides safe upper bounds for worst-case response times. Whereas for the others, the existing analysis calculates optimistic worst-case response times.

We extended the existing analysis for CAN with FIFO queues to provide safe upper bounds on the worst-case response times of mixed messages. The extended analysis is generally applicable to any higher-level protocol for CAN that supports periodic, sporadic, and mixed transmission of messages in a system comprising of priority- and FIFO-queued nodes. We conducted a case study and performed comparative evaluation of the extended analysis with the existing analysis for mixed, periodic and sporadic messages in CAN with priority queues.

The FIFO queues are already used in practical CAN controllers. Although, they are easy to implement and use, they can result in higher response times of messages. Therefore, the CAN controllers which implement priority queues should be preferred over the CAN controllers that implement FIFO queues. Moreover, it is important to use the response-time analysis that correctly matches the queueing policies in the ECUs; and transmission types of messages used in the higher-level protocols. If these constraints are not rightly considered in the response-time analysis, the calculated response times can be optimistic.

ACKNOWLEDGEMENT

This work is supported by the Swedish Knowledge Foundation (KKS) within the projects FEMMVA and EEMDEF, the Swedish Research Council (VR) within projects SynthSoft and TiPCEs, and the Strategic Research Foundation (SSF) with the centre PROGRESS. The authors would like to thank the industrial partners Arcticus Systems AB, BAE Systems Hägglunds and Volvo Construction Equipment (VCE), Sweden.

REFERENCES


### APPENDIX A

<table>
<thead>
<tr>
<th>Notation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m_n$</td>
<td>Any message $n$</td>
</tr>
<tr>
<td>$ID_n$</td>
<td>Unique identifier of $m_n$</td>
</tr>
<tr>
<td>$P_n$</td>
<td>Priority of $m_n$</td>
</tr>
<tr>
<td>$\xi_n$</td>
<td>Transmission type of $m_n$. It specifies whether $m_n$ is periodic ($P$), sporadic ($S$) or mixed ($M$)</td>
</tr>
<tr>
<td>$C_n$</td>
<td>Worst-case transmission time of $m_n$</td>
</tr>
<tr>
<td>$J_n$</td>
<td>Queueing jitter of $m_n$</td>
</tr>
<tr>
<td>$s_n$</td>
<td>Size of data payload in $m_n$</td>
</tr>
<tr>
<td>$T_n$</td>
<td>Transmission period of $m_n$</td>
</tr>
<tr>
<td>$MUT_n$</td>
<td>Minimum Update Time of $m_n$. It is the minimum time that should elapse between the transmission of any two sporadic messages</td>
</tr>
<tr>
<td>$B_n$</td>
<td>Blocking time of $m_n$</td>
</tr>
<tr>
<td>$R_n$</td>
<td>Worst-case response time of $m_n$</td>
</tr>
<tr>
<td>$D_n$</td>
<td>Deadline of $m_n$</td>
</tr>
<tr>
<td>$hp(m_n)$</td>
<td>Set of higher priority messages than $m_n$</td>
</tr>
<tr>
<td>$lp(m_n)$</td>
<td>Set of lower priority messages than $m_n$</td>
</tr>
<tr>
<td>$hep(m_n)$</td>
<td>Set of higher and equal priority messages than $m_n$</td>
</tr>
<tr>
<td>$lep(m_n)$</td>
<td>Set of lower and equal priority messages than $m_n$</td>
</tr>
<tr>
<td>$\omega_n$</td>
<td>Queueing delay for $m_n$</td>
</tr>
<tr>
<td>$f_n$</td>
<td>Maximum buffering time for $m_n$</td>
</tr>
<tr>
<td>$\tau_{bt}$</td>
<td>Time required to transmit a single bit of data over CAN</td>
</tr>
<tr>
<td>$t_n$</td>
<td>Length of the priority level-$n$ busy period</td>
</tr>
<tr>
<td>$q_n$</td>
<td>Index variable to denote multiple instances of $m_n$</td>
</tr>
<tr>
<td>$U_n$</td>
<td>Bus utilization for priority level-$n$</td>
</tr>
<tr>
<td>$Q_n$</td>
<td>Total Number of instances of $m_n$ that are queued in priority level-$n$ busy period</td>
</tr>
<tr>
<td>$M(m_n)$</td>
<td>The set of FIFO-queued messages in the sender ECU of $m_n$</td>
</tr>
<tr>
<td>$L_n$</td>
<td>The lowest priority message in the set $M(m_n)$</td>
</tr>
<tr>
<td>$B_{L_n}$</td>
<td>Blocking time due to $L_n$</td>
</tr>
<tr>
<td>$C_{MAX}$</td>
<td>Maximum transmission time of a message in the set $M(m_n)$</td>
</tr>
<tr>
<td>$C_{MIN}$</td>
<td>Minimum transmission time of a message in the set $M(m_n)$</td>
</tr>
<tr>
<td>$m_{nP}$</td>
<td>Periodic part of a mixed message $m_n$</td>
</tr>
<tr>
<td>$m_{nS}$</td>
<td>Sporadic part of a mixed message $m_n$</td>
</tr>
<tr>
<td>$R_{nP}$</td>
<td>Response time of $m_{nP}$</td>
</tr>
<tr>
<td>$R_{nS}$</td>
<td>Response time of $m_{nS}$</td>
</tr>
</tbody>
</table>

**TABLE I:** Notations and terminology.