Extending Response-Time Analysis for Mixed Messages with Offsets in Controller Area Network

Saad Mubeen*, Jukka Mäki-Turja*† and Mikael Sjödin*

*Mälardalen Real-Time Research Centre (MRTC), Mälardalen University, Sweden
†Arcticus Systems, Järfälla, Sweden
{saad.mubeen, jukka.maki-turja, mikael.sjodin}@mdh.se

Abstract. The existing offset-aware response-time analysis of Controller Area Network (CAN) for mixed messages has certain practical limitations. It is based on the assumption that the jitter and deadline of a message are smaller or equal to the transmission period. However, practical systems may contain messages with release jitter greater than the period. Consequently, the deadlines specified for such messages are also greater than their periods. In this paper, we extend the existing response-time analysis for mixed messages in CAN that are scheduled with offsets and have arbitrary jitter and deadlines. Mixed messages are implemented by several higher-level protocols based on CAN that are used in the automotive industry. The extended analysis is applicable to any higher-level protocol for CAN that uses periodic, sporadic and mixed transmission modes.

1 Introduction

The Controller Area Network (CAN) [1] is a multi-master, event-triggered, serial communication bus protocol supporting bus speeds of up to 1 mega bits per second. It is a widely used protocol in the automotive domain. It has been standardized as ISO 11898-1 [2]. According to CAN in Automation (CiA) [3], the estimated number of CAN enabled controllers sold in 2011 are about 850 million. CAN also finds its applications in other domains, e.g., industrial control, medical equipments, maritime electronics, and production machinery. There are several higher-level protocols for CAN that are developed for many industrial applications such as CAN Application Layer (CAL), CANopen, J1939, Hägglunds Controller Area Network (HCAN), CAN for Military Land Systems domain (MilCAN).

In order to provide evidence that each action by the system will be provided in a timely manner, i.e., each action will be taken at a time that is appropriate to the environment of the system, *a priori* analysis techniques such as schedulability analysis have been developed by the research community. Response-Time Analysis (RTA) [4, 5] 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 network respectively. RTA is used to perform a schedulability test which means it checks whether or not tasks (or messages) in the system (or network) will satisfy their deadlines. 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) [6].

1.1 Motivation and related work

Tindell et al. [7] developed the schedulability analysis for CAN. This analysis has been implemented in several tools that are used in the automotive industry [8–11]. Davis et al. [12] refuted, revisited and revised the analysis by Tindell et al. In [13], Davis et al. extended the analysis in [7, 12] which is applicable to the CAN network where some nodes implement priority queues and some implement FIFO queues. All these analyses assume that the messages are queued for transmission periodically or sporadically. They do not support mixed messages in CAN, i.e., the messages that are simultaneously time (periodic) and event (sporadic) triggered. Mubeen et al. [14] extended the existing analysis to support mixed messages in CAN where nodes implement priority-based queues. Mubeen et al. [15, 16] further extended their analysis to support mixed messages in the network where some nodes implement priority queues while others implement FIFO queues.

But, none of the analysis 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 [17]. Furthermore, the worst-case response times of messages (especially with lower priority) in CAN increase with the increase in the network load. However, the worst-case response-times of lower priority messages in CAN can be reduced if the messages are scheduled with offsets [18–20]. A method for the assignment of offsets to improve the overall bandwidth utilization is proposed in [19, 20]. The worst-case response-time analysis for CAN messages with offsets has been developed by several researchers [21, 22, 18, 23, 17].

None of these analyses support mixed messages that are scheduled with offsets. In [24], we extended the existing offset-based analysis[21] to support worst-case 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 [21]. In this paper, we remove these limitations. Basically, we extend the analysis for mixed messages [14] by building it upon the analysis for CAN messages with offsets [17]. Figure 1 depicts the relation between the existing and extended analyses.

1.2 Paper contribution

We extend the response-time analysis of CAN for mixed messages that are scheduled with offsets. The existing analysis for mixed messages with offsets [24] places restrictions on message deadline and jitter, i.e., each of them should be less than or equal to message period. Message jitter may be higher than its period, e.g., for the messages scheduled at the gateway node [17]. The existing offset-aware analysis does not support mixed messages whose jitter and deadlines are higher than their transmission periods. In this paper, we lift these restrictions by assuming deadline and jitter to be arbitrary, i.e., each one of them can be higher than message period. Intuitively, there can be several instances of same message that are queued for transmission. Hence, our extended analysis considers the response times of all these instances while calculating the worst-case response time. Mixed message are implemented by several higher-level protocols

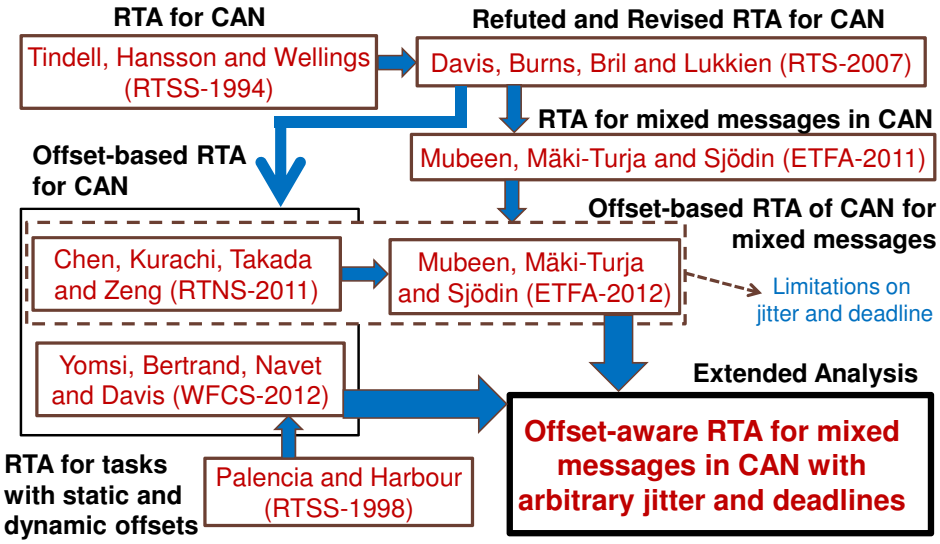

Fig. 1. Relation between the existing and extended Response Time Analysis (RTA)

used in the industry. The analysis is applicable to any higher-level protocol for CAN that uses periodic, sporadic and mixed transmission of messages that are scheduled with offsets. We also show the applicability of the extended analysis by conducting the automotive-application case study.

1.3 Paper layout

The remainder of the paper is organized as follows. In Section 2, we discuss mixed transmission patterns supported by higher-level protocols for CAN. Section 3 describes the scheduling model. Section 4 presents the extended analysis. In Section 5, we present a case study. Section 6 concludes the paper and discusses the future work.

2 Mixed transmission patterns supported by higher-level protocols

When CAN is employed for network communication in a distributed real-time system, each node (processor) or Electronic Control Unit (ECU) is equipped with a CAN interface that connects the node to the bus [25]. Application tasks in each node, that require remote transmission, are assumed to queue messages for transmission over the CAN network. The messages are transmitted according to the protocol specification of the CAN protocol. Traditionally, it is assumed that the tasks queueing CAN messages are invoked either by periodic events with a period or sporadic events with a minimum inter-arrival time. However, there are some higher-level protocols and commercial extensions of CAN in which the task that queues the messages can be invoked periodically as well as sporadically. If a message can be queued for transmission periodically as well

as at the arrival of a sporadic event then the transmission type of the message is said to be mixed. In other words, a mixed message is simultaneously time (periodic) and event triggered (sporadic). We identified three types of implementations of mixed messages used in the industry.

Consistent terminology. To stay consistent, we use the terms message and frame interchangeably because we only consider messages that will fit into one frame (maximum 8 bytes). For the purpose of using simple notation, we call a CAN message as periodic, sporadic or mixed if it is queued by an application task that is invoked periodically, sporadically or both (periodically and sporadically) respectively. If a message is queued for transmission at periodic intervals, we use the term "Period" to refer to its periodicity. A sporadic message is queued for transmission as soon as an event occurs that changes the value of one or more signals contained in the message provided a Minimum Update Time (MUT) between the queueing of two successive sporadic messages has elapsed. Hence, the transmission of a sporadic frame is constrained by MUT. We overload the term "MUT" to refer to "Inhibit Time" in CANopen protocol [26] and "Minimum Delay Time (MDT)" in AUTOSAR communication [27].

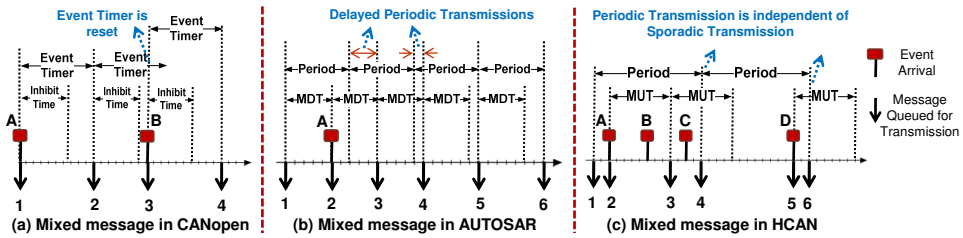

Fig. 2. Mixed transmission pattern in higher-level protocols for CAN

2.1 Method 1: Implementation in CANopen

The CANopen protocol [26] supports mixed transmission that corresponds to the Asynchronous Transmission Mode coupled with the Event Timer. The Event Timer is used to transmit an asynchronous message cyclically. 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 the 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 [26].

The transmission pattern of a mixed message in CANopen is illustrated in Figure 2(a). The down-pointing arrows symbolize the queueing of messages while the upward lines (labeled with alphabets) represent arrival of the events. Message 1 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, message 2 is queued due to periodicity and both the Event

Timer and Inhibit Time are reset again. When the event B arrives, message 3 is immediately queued because the Inhibit Time has already expired. Note that the Event Timer is also reset at the same time when message 3 is queued as shown in Figure 2(a). Message 4 is queued because of the expiry of the Event Timer. There exists a dependency relationship between the Inhibit Time and the Event Timer.

2.2 Method 2: Implementation in AUTOSAR

AUTOSAR (AUTomotive Open System ARchitecture) [28] 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 period equal to 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 a mixed message, regardless of being periodic or sporadic, is limited by MDT. This means that both periodic and sporadic transmissions are delayed until MDT expires. The transmission pattern of a mixed message implemented by AUTOSAR is illustrated in Figure 2(b). Message 1 is queued (MDT is started) because of partly periodic nature of a mixed message. When the event A arrives, message 2 is queued immediately because MDT has already expired. The next periodic transmission is scheduled 2 time units after the transmission of message 2. However, next two periodic transmissions corresponding to messages 3 and 4 are delayed because MDT is not expired. This is indicated by "Delayed Periodic Transmissions" in Figure 2(b). The periodic transmissions corresponding to messages 5 and 6 take place at the scheduled times because MDT is already expired in both cases.

2.3 Method 3: Implementation in HCAN

A mixed message in HCAN protocol [29] 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 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 MUT. The transmission pattern of a mixed message is illustrated in Figure 2(c). Message 1 is queued because of periodicity. As soon as event A arrives, message 2 is queued. When event B arrives it is not queued immediately because MUT is not expired yet. As soon as MUT expires, message 3 is queued. Message 3 contains the signal changes that correspond to event B. Similarly, a message is not immediately queued when the event C arrives because MUT is not expired. Message 4 is queued because of the periodicity. Although, MUT was not expired, the event signal corresponding to event C was packed in message 4 and queued as part of the periodic message. Hence, there is no need to queue an additional sporadic message when MUT expires. This indicates that the periodic transmission of a mixed message cannot be interfered by its sporadic transmission (a unique property of HCAN protocol). When the event D arrives, a sporadic instance of the mixed message

is immediately queued as message 5 because MUT has already expired. Message 6 is queued due to periodicity.

2.4 Discussion

In the first method, the Event Timer is reset every time a mixed message is queued for transmission. The implementation of a mixed message in method 2 is similar to method 1 to some extent. The main difference is that in method 2, the periodic transmission can be delayed until the expiry of MDT. 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 a mixed message. Hence, the worst-case periodicity of a mixed message in methods 1 and 2 can never be higher than Inhibit Timer and MDT respectively. Therefore, the existing analyses for CAN messages with offsets [21, 22, 18, 23, 17] can be used for analyzing mixed messages in the first and second implementation methods.

However, the periodic transmission is independent of the sporadic transmission in the third method. The periodic timer is not reset with every sporadic transmission. A mixed message can be queued for transmission even if MUT is not expired. Hence, the worst-case periodicity of a mixed message is neither bounded by period nor by MUT. Therefore, the analyses in [21, 22, 18, 23, 17] cannot be used for analyzing mixed messages in the third implementation method.

3 System model

The system, S, consists of a number of CAN controllers that are connected to a single CAN network. The nodes implement priority-ordered queues, i.e., the highest priority message in a node enters into the bus arbitration. Each CAN message m_m has a unique identifier and 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 m_m is considered higher than the priority of m_n if $P_m < P_n$.

Let the sets $hp(m_m)$, $lp(m_m)$, and $hep(m_m)$ contain the messages with priorities higher, lower, and equal and higher than m_m respectively. Although the priorities of CAN messages are unique, the set $hep(m_m)$ will be 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 ξ_m that denotes the transmission type of a message. ξ_m specifies whether m is periodic (P), sporadic (S) or mixed (M). Formally the domain of ξ_m can be defined as:

$$\xi_m \in [P, \quad S, \quad M]$$

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 of m_m. Each message can carry a data payload that ranges from 0 to 8 bytes. This number is specified in the header field of 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 a MUT_m (Minimum Update Time) that refers to the minimum time that should elapse between the transmission of any two sporadic messages. B_m denotes the blocking time of m_m which refers to the largest amount of time m_m can be blocked by any lower priority message.

We duplicate a message when its transmission type is mixed. Hence, each mixed message m_m is treated as two separate messages, i.e., periodic and sporadic. The duplicates share all the attributes except T_m and MUT_m. The periodic copy inherits T_m while the sporadic copy inherits MUT_m. Each message has a worst-case response time, denoted by R_m, and defined as the longest time between the queueing 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 minimum update times of the corresponding messages. We assume that the CAN controllers are capable of buffering more than one instance of a message.

Let O_m denotes the offset of m_m. We assume that the offset of m_m is always smaller than its period. The first arrival time of m_m is equal to its offset while the subsequent arrivals occur periodically with respect to the first arrival. We assume that the smallest offset in a node is zero. It should be noted that each node has its own local time and there is no global synchronization among the nodes. We assume that the offset relations exist only among periodic messages and periodic copies of mixed messages within a node. We further assume that there are no offset relations:

1. among sporadic messages,
2. between a periodic message and a sporadic message,
3. between a periodic copy of a mixed message and a sporadic message,
4. between the duplicates of a mixed message,
5. between any two messages belonging to different nodes.

All periodic messages and periodic copies of mixed messages in a node are gathered into a single transaction denoted by Γ_i. Each transaction Γ_i belongs to Γ which is a set of all transactions in the system. This transaction model is adapted from [30]. It should be noted that the offset relations exist only within a transaction, and there are no offset relations among transactions. In the context of a transaction, we denote a message m_j belonging to transaction i by m_i^j. Each transaction has a period denoted by T_{Γ_i} and defined as the Least Common Multiple (LCM) of the periods of all messages present in the transaction. Each sporadic message or sporadic copy of a mixed message is modeled as a transaction of its own.

An example of two messages transmitted by a node is depicted in Figure 3. m_1 is a mixed message with high priority while m_2 is a periodic message with low priority. Transaction Γ_1 contains both m_2 and periodic copy of m_1. The period of Γ_1 denoted by T_{Γ_i} is the LCM of T_1 and T_2. Transaction Γ_2 consists of only sporadic copy of m_1.

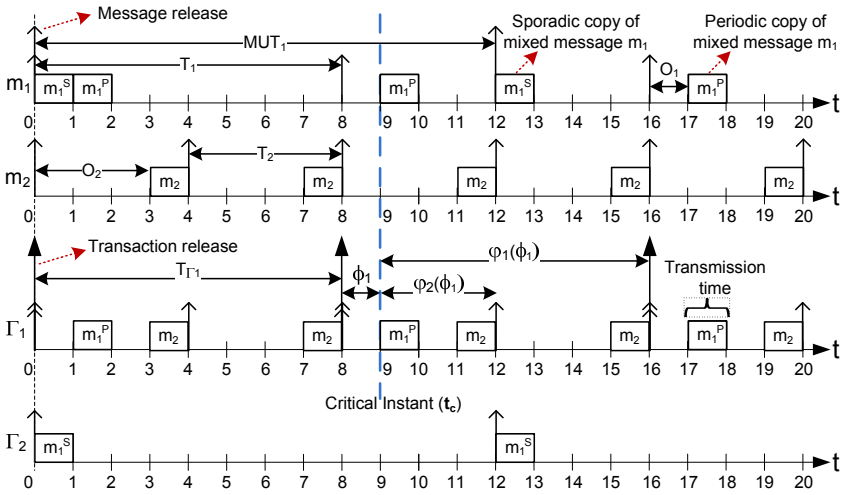

Fig. 3. Example demonstrating messages and transactions

4 Worst-case response-time analysis

Let m_m be the message under analysis. We analyze m_m differently based on its transmission type. Intuitively, we consider three different cases namely periodic, sporadic and mixed. We first discuss few terms that are used in the analysis.

In order to calculate the worst-case response time of m_m, the maximum busy period [7, 12] for priority level-m should be known first.

Maximum Busy Period. It is the longest contiguous interval of time during which m_m is unable to complete its transmission due to two reasons. First, the bus is occupied by the higher priority messages. Second, a lower priority message already started its transmission when m_m is queued for transmission. The maximum busy period starts at the critical instant.

Critical Instant. For a system where messages are scheduled without offsets, the critical instant corresponds to the point in time when all higher priority messages are assumed to be queued simultaneously with m_m while their subsequent instances are assumed to be queued after the shortest possible interval of time [12]. However, this assumption does not hold in the system where messages are scheduled with offsets.

We redefine the critical instant for priority level-m busy period as the instant when (1) m_m or any other higher priority message belonging to the same node as that of m_m is queued for transmission, (2) At least one message with priority higher than m_m is queued for transmission from every node, (3) all sporadic messages and sporadic copies of mixed messages belonging to the set $hp(m_m)$ from every node are simultaneously queued for transmission at the respective nodes, and (4) a lower priority message just started its transmission when m_m is queued. The critical instant for priority level-2 busy period is identified at t_c in Figure 3. According to condition (3) stated above, the arrival of Γ_2 should coincide with the critical instant.

Worst-Case Candidates. The main issue regarding condition (2) is to determine which message in the set $hp(m_m)$ is the candidate to start the critical instant, i.e., contributing to the worst-case response-time of m_m. The solution is that any message in the set $hp(m_m)$ can be the worst-case candidate. Therefore, each message has to be tested in the busy period as the potential worst-case candidate. The response time of m_m should be computed from every worst-case candidate and the maximum among all should be considered as the worst-case response time of m_m.

We present the response-time analysis with respect to any worst-case candidate in the following subsections.

4.1 Case: When m_m is a periodic message

Let m_m belongs to transaction Γ_i. The worst-case response time of m_m is equal to the maximum value among the response times of all of its instances. We calculate the response times of all instances of m_m within the priority level-m busy period. Let q_m be the index variable to denote instances of m_m. Let q_m^L and q_m^H denote lowest- and highest-numbered instances respectively. The worst-case response time of m_m is given by:

$$R_m = max\{R_m(q_m)\}, \qquad \forall\, q_m^L \leq q_m \leq q_m^H \tag{1}$$

It should be noted that q_m will be equal to 1 if the message instance is queued for transmission between the critical instant and T_m. Further, q_m will be equal to 2 if the message instance is queued for transmission between T_m and $2.T_m$. Similarly, q_m will be equal to 0 if the message instance is queued for transmission between the critical instant and $-T_m$. Since the jitter of a message can be greater than its transmission period, it is possible that the previous instances of the message may also be delayed due to jitter and enter in the maximum busy period. The calculations for the response time of instance q_m are adapted from [17] as follows.

$$R_m(q_m) = ST_m + C_m - (\varphi_m(\phi_i) + (q_m - 1).T_m) \tag{2}$$

ϕ_i in (2) denotes the time interval between latest arrival of Γ_i (prior to the critical instant) and the critical instant. Consider the example message set in Figure 3. ϕ_i is equal to 1 time unit and is identified as ϕ_1 on the third time line from the top. $\varphi_m(\phi_i)$ in (2) represents the length of the time interval between the critical instant and first release of m_m that occurs at or after the critical instant. Consider again the example message set in Figure 3. $\varphi_m(\phi_i)$ for messages m_1^P and m_2 are identified by $\varphi_1(\phi_1)$ and $\varphi_2(\phi_1)$ respectively. The calculations for $\varphi_m(\phi_i)$ are adapted from [30] as follows.

$$\varphi_m(\phi_i) = (T_m - (\phi_i - O_m) \mod T_m) \mod T_m \tag{3}$$

ST_m in (2) denotes the Start Time (ST) when the priority level-m busy period ends and $m_m(q_m)$ can start its transmission. Basically, it sums up the interferences due to higher priority messages, previous instances of the same message and blocking factor. It can be calculated by solving the following equation.

$$ST_m^{n+1} = B_m + (q_m - q_m^L).C_m + \sum_{\forall \Gamma_k \in \Gamma} W_m(\Gamma_k, \phi_k, ST_m^n) \tag{4}$$

Where the term $(q_m - q_m^L).C_m$ represents the interference due to previous instances of m_m that are queued ahead of the instance under analysis. B_m is the blocking delay for m_m. It is defined as the amount of time equal to the largest transmission time in the set of lower priority messages. B_m is calculated as follows.

$$B_m = \max_{\forall m_j \in lp(m_m)} \{C_j\} \tag{5}$$

(4) is an iterative equation. It is solved iteratively until two consecutive solutions become equal. The starting value for ST_m^n in (4) can be selected equal to $B_m + (q_m - q_m^L).C_m$. W_m in (4) represents the amount of interference due to the messages in the set $hp(m_m)$ that are queued for transmission since the beginning of the busy period. It is important to mention that CAN uses fixed-priority non-preemptive scheduling, and therefore, a message cannot be interfered by higher priority messages during its transmission. Whenever we use the term interference, it refers to the amount of time m_m has to wait in the send queue because the higher priority messages win the arbitration, i.e., the right to transmit before m_m. W_m can be calculated as follows.

$$W_m(\Gamma_k, \phi_k, ST_m^n) = \sum_{\forall m_j \in hp_k(m_m)} \Upsilon_k^j(ST_m^n).C_j \tag{6}$$

Where $hp_k(m_m)$ represents the set of all those messages that belong to Γ_k and have priority higher than m_m. $\Upsilon_k^j(ST_m^n)$ in (6) is calculated differently based on the transmission type ξ_j of the higher priority message m_j. The calculations for $\Upsilon_k^j(ST_m^n)$ are adapted from [17] and [14] as follows. It should be noted that the existing analysis considers only higher priority periodic messages while calculating $\Upsilon_k^j(ST_m^n)$. On the other hand, we consider all higher priority periodic, sporadic and mixed messages while calculating $\Upsilon_k^j(ST_m^n)$.

$$\Upsilon_k^j(ST_m^n) = \begin{cases} \left\lfloor \frac{J_j + \varphi_j(\phi_k)}{T_j} \right\rfloor + \left\lfloor \frac{ST_m^n - \varphi_j(\phi_k)}{T_j} \right\rfloor + 1, & \text{if } \xi_j = \text{P} \\ \left\lceil \frac{ST_m^n + J_j + \tau_{bit}}{MUT_j} \right\rceil, & \text{if } \xi_j = \text{S} \\ \left\lfloor \frac{J_j + \varphi_j(\phi_k)}{T_j} \right\rfloor + \left\lfloor \frac{ST_m^n - \varphi_j(\phi_k)}{T_j} \right\rfloor + 1 & \\ \quad + \left\lceil \frac{ST_m^n + J_j + \tau_{bit}}{MUT_j} \right\rceil, & \text{if } \xi_j = \text{M} \end{cases} \tag{7}$$

Where $\varphi_j(\phi_k)$ is calculated by replacing the indices $_m$ and $_i$ with $_j$ and $_k$ in (3) respectively. In (7), the periodic case is adapted from [17]. $\left\lfloor \frac{J_j + \varphi_j(\phi_k)}{T_j} \right\rfloor$ represents the maximum number of instances of the higher priority periodic message or periodic copy of mixed message m_j that may accumulate at the critical instant. Whereas

$\left\lfloor \frac{ST_m^n - \varphi_j(\phi_k)}{T_j} \right\rfloor + 1$ represents the maximum number of instances of m_j that are queued for transmission in the interval that starts with the critical instance and ends at Υ_m^n.

There are no offset relations of m_m with any sporadic message. Moreover, all sporadic messages are assumed to be queued for transmission at the critical instant. Therefore, the sporadic and mixed cases in (7) are adapted from [14]. $\left\lceil \frac{ST_m^n + J_j + \tau_{bit}}{MUT_j} \right\rceil$ represent the maximum number of instances of higher priority sporadic message or sporadic copy of mixed message m_j that are queued for transmission in the interval that starts with the critical instance and ends at Υ_m^n. This also includes the number of instances of m_j that may accumulate at the critical instant due to jitter.

It is evident from (7) that interference from both periodic and sporadic copies of higher priority mixed message is taken into account. τ_{bit} denotes the time required to transmit a single bit on the CAN bus. Its value depends upon the speed of the bus. The lowest- and highest-numbered instances of m_m denoted by q_m^L and q_m^H are calculated as follows.

$$q_m^L = -\left\lfloor \frac{J_m + \varphi_m(\phi_i)}{T_m} \right\rfloor + 1 \tag{8}$$

$$q_m^H = \left\lceil \frac{L_m - \varphi_m(\phi_i)}{T_m} \right\rceil \tag{9}$$

Where L_m represents the length of priority level-m busy period. The existing analysis [17] considers only higher priority periodic messages while calculating L_m. We adapt the existing analysis to consider all higher priority periodic, sporadic and mixed messages while calculating L_m. Similar to (4), L_m can be calculated using fixed-point method as follows.

$$L_m^{n+1} = \left[\left\lfloor \frac{J_m + \varphi_m(\phi_i)}{T_m} \right\rfloor + \left\lceil \frac{L_m^n - \varphi_m(\phi_i)}{T_m} \right\rceil \right].C_m + B_m + \sum_{\forall \Gamma_k \in \Gamma, m_j \in hp_k(m_m)} \mathsf{M}_k^j(L_m^n).C_j \tag{10}$$

Where

$$\mathsf{M}_k^j(L_m^n) = \begin{cases} \left\lfloor \frac{J_j + \varphi_j(\phi_k)}{T_j} \right\rfloor + \left\lceil \frac{L_m^n - \varphi_j(\phi_k)}{T_j} \right\rceil, & \text{if } \xi_j = \text{P} \\ \left\lceil \frac{L_m^n + J_j + \tau_{bit}}{MUT_j} \right\rceil, & \text{if } \xi_j = \text{S} \\ \left\lfloor \frac{J_j + \varphi_j(\phi_k)}{T_j} \right\rfloor + \left\lceil \frac{L_m^n - \varphi_j(\phi_k)}{T_j} \right\rceil & \\ + \left\lceil \frac{L_m^n + J_j + \tau_{bit}}{MUT_j} \right\rceil, & \text{if } \xi_j = \text{M} \end{cases} \tag{11}$$

4.2 Case: When m_m is a sporadic message

Let again the message under analysis be m_m that belongs to Γ_i, i.e, transaction of its own. The worst-case response time of m_m can be calculated similar to the periodic case with one exception. That is, sporadic message does not have any offset relations with any other message in the system. Moreover, all sporadic messages including m_m are assumed to be queued for transmission at the critical instant. Intuitively, ϕ_i will be equal to MUT_m, i.e., the latest arrival of m_m prior to critical instant will be MUT_m time units before the critical instant. Let us use O_m equal to zero, and MUT_m in place of both T_m and ϕ_i in (3).

$$\varphi_m(\phi_i) = 0 \tag{12}$$

In this case, (1), (4), (5), (6), (7) and (11) hold intact. However, we need to replace the new value of $\varphi_m(\phi_i)$ from (12) in the calculations for (2), (8), (9) and (10) as follows.

$$R_m(q_m) = ST_m + C_m - (q_m - 1).MUT_m \tag{13}$$

$$q_m^L = -\left\lfloor \frac{J_m}{MUT_m} \right\rfloor + 1 \tag{14}$$

$$q_m^H = \left\lceil \frac{L_m}{MUT_m} \right\rceil \tag{15}$$

$$L_m^{n+1} = \left\lceil \frac{L_m^n + J_m + \tau_{bit}}{MUT_m} \right\rceil .C_m + B_m + \sum_{\forall \Gamma_k \in \Gamma, m_j \in hp_k(m_m)} \mathsf{M}_k^j(L_m^n).C_j \tag{16}$$

4.3 Case: When m_m is a mixed message

Let again m_m be the message under analysis. Since a mixed message is duplicated as two separate messages, the extended analysis treats them separately. Let the periodic and sporadic copies of m_m be denoted by m_{m_P} and m_{m_S} respectively. We denote the worst-case response times of m_{m_P} and m_{m_S} by R_{m_P} and R_{m_S} respectively. The worst-case response time of m_m is the maximum between R_{m_P} and R_{m_S} as follows.

$$R_m = max\{R_{m_P}, \ \ R_{m_E}\} \tag{17}$$

Where R_{m_P} and R_{m_S} are equal to maximum among the response times of their respective instances. Let q_{m_P} be the index variable to denote the instances of m_{m_P}. Let $q_{m_P}^L$ and $q_{m_P}^H$ denote the lowest- and highest-numbered instances of m_{m_P} respectively. Let q_{m_S}, $q_{m_S}^L$ and $q_{m_S}^H$ denote the index variable for instances, and lowest- and highest-numbered instances of m_{m_S} respectively. Then R_{m_P} and R_{m_S} are given by the following equations.

$$R_{m_P} = max\{R_{m_P}(q_{m_P})\}, \forall\ q^L_{m_P} \leq q_{m_P} \leq q^H_{m_P} \tag{18}$$

$$R_{m_S} = max\{R_{m_S}(q_{m_S})\}, \forall\ q^L_{m_S} \leq q_{m_S} \leq q^H_{m_S} \tag{19}$$

The calculations for the worst-case response time of each instance of m_{m_P} and m_{m_S} are adapted from (2) and (13) as follows.

$$R_{m_P}(q_{m_P}) = ST_{m_P} + C_m - (\varphi_{m_P}(\phi_i) + (q_{m_P} - 1).T_m) \tag{20}$$

$$R_{m_S}(q_{m_S}) = ST_{m_S} + C_m - (q_{m_S} - 1).MUT_m \tag{21}$$

Where $\varphi_{m_P}(\phi_i)$ is calculated using (3). The calculations for ST_{m_P} and ST_{m_S} are adapted from (2) and (13) after some augmentation.

$$\begin{aligned} ST^{n+1}_{m_P} &= B_m + \left\lceil \frac{q_{m_P}.T_m + J_m + O_m}{MUT_m} \right\rceil .C_m \\ &+(q_{m_P} - q^L_{m_P}).C_m + \sum_{\forall \Gamma_k \in \Gamma} W_{m_P}(\Gamma_k, \phi_k, ST^n_{m_P}) \end{aligned} \tag{22}$$

$$\begin{aligned} ST^{n+1}_{m_S} &= B_m + \left\lceil \frac{q_{m_S}.MUT_m + J_m}{T_m} \right\rceil .C_m \\ &+(q_{m_S} - q^L_{m_S}).C_m + \sum_{\forall \Gamma_k \in \Gamma} W_{m_S}(\Gamma_k, \phi_k, ST^n_{m_S}) \end{aligned} \tag{23}$$

$\left\lceil \frac{q_{m_P}.T_m + J_m + O_m}{MUT_m} \right\rceil .C_m$ and $\left\lceil \frac{q_{m_S}.MUT_m + J_m}{T_m} \right\rceil .C_m$ in (22) and (23) respectively represent 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_{m_P} and m_{m_S} have equal priorities, any instance of m_{m_S} queued ahead of m_{m_P} will contribute an extra delay to the worst-case queueing delay experienced by m_{m_P}. A similar argument holds in the case of m_{m_S}. We reuse the effect of self interference in a mixed message that we derived in [14].

The calculations for W_{m_P}, $q^L_{m_P}$, $q^H_{m_P}$ and L_{m_P} are carried out using (6), (8), (9) and (10) by replacing the index $_m$ with $_{m_P}$ respectively. Similarly, W_{m_S}, $q^L_{m_S}$, $q^H_{m_S}$ and L_{m_P} are calculated using (6), (14), (15) and (16) by replacing the index $_m$ with $_{m_S}$ respectively. Further, the calculations in (5), (7) and (11) hold intact with proper replacement of the index variable for both m_{m_P} and m_{m_S}.

5 Automotive-application case study

In this section, we conduct the automotive case study, i.e., Steer-By-Wire (SBW) system. The SBW system is an automotive feature that substitutes most of the mechanical and hydraulic components with the electronic components in the steering system of the vehicle. We adapt the SBW system from [31]. A part of the SBW system is shown in

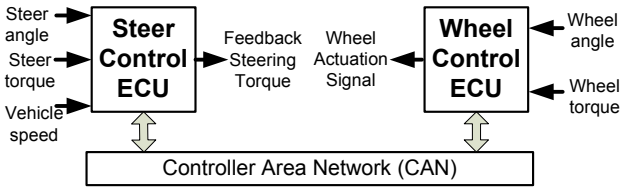

Fig. 4. Partial architecture of the steer-by-wire system

ECU	m	P_m	ξ_m	T_m	MUT_m	C_m	O_m	J_m	D_m	R_m
ECU_S	m_1	1	P	10	-	1	0	1	10	3
ECU_S	m_2	2	M	10	10	1	1	11	20	15
ECU_S	m_3	3	S	-	10	1	0	0	10	7
ECU_W	m_4	4	P	10	-	1	0	0	10	8
ECU_W	m_5	5	P	10	-	1	2	12	20	18

Table 1. Attributes and response times of periodic, sporadic and mixed messages in the steer-by-wire system

Figure 4. There are two ECUs (rest of the ECUs are not shown for simplicity) that are connected to the CAN network.

The Steering Control ECU denoted by ECU_S receives input from three sensors that correspond to the steering angle, steering torque (applied by the driver) and vehicle speed signals. It sends three messages m_1, m_2 and m_3 to the network. These messages carry information regarding steering angle, torque and feedback signals. m_1 is a periodic, m_2 is a mixed and m_3 is a sporadic message. ECU_S receives the periodic message m_4 that contains information about wheel torque that is sent by the Wheel Control ECU. Based on these inputs, it calculates the feedback steering torque and sends it to the feedback actuator. The feedback torque actuator is responsible for producing the turning effect of the steering which in turn produces the feeling of turning the wheels for the driver.

Similarly, the Wheel Control ECU denoted by ECU_W acquires signals from wheel angle and wheel torque sensors. Depending upon these signals and the signals received in the CAN message, it calculates the wheel torque, and produces actuation signals for the wheel control actuators. Apart from m_4, it also sends a periodic message m_5 to the network. The timing attributes of all messages are shown in Table 1. We analyzed this message set with the extended analysis. The response times calculated using the extended analysis are also listed in Table 1. Since, the jitter of messages m_2 and m_5 is higher than the corresponding periods, there are several instances of these messages present in the corresponding busy periods. For example, there are four instances of m_2 that are present in the priority level-2 busy period. The extended analysis calculates the response times of all these instances and considers maximum among them as the worst-case response time of m_2. It should be noted that correct analysis of this message set would not have been possible with the existing analysis because it contains a mixed message whose jitter and deadline are greater than corresponding period.

6 Conclusion

The existing response-time analysis of CAN does not support the analysis of mixed messages that are scheduled with offsets and have jitter and deadlines higher than their transmission periods. Message jitter can be higher than its period in practical systems. We extended the existing offset-aware analysis for CAN by lifting the restrictions on message jitter and deadline. The extended analysis provides safe upper bounds on the response times of mixed messages that are scheduled with offsets. Mixed messages are implemented by several higher-level protocols for CAN that are used in the automotive industry today. The extended analysis is applicable to any higher-level protocol for CAN that uses periodic, sporadic, and mixed transmission of messages that are scheduled with offsets. We also conducted the automotive-application case study to demonstrate the applicability of our analysis.

In the future, we plan to develop an optimized offset assignment method for the systems that contain periodic as well as mixed messages. We also plan to implement the extended analysis as a plug-in for the existing industrial tool suite the Rubus-ICE [11].

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. "Automotive networks. CAN in Automation (CiA)," http://www.can-cia.org/index.php?id=416.
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. Nolin, J. Mäki-Turja, and K. Hänninen, "Achieving Industrial Strength Timing Predictions of Embedded System Behavior," in *ESA*, 2008, pp. 173–178.
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. "Volcano Network Architect (VNA). Mentor Graphics," http://www.mentor.com/products/vnd/communication-management/vna.
9. S. Mubeen, J. Mäki-Turja, and M. Sjödin, "Support for Holistic Response-time Analysis in an Industrial Tool Suite: Implementation Issues, Experiences and a Case Study," in *19th IEEE Conference on Engineering of Computer Based Systems (ECBS)*, April 2012, pp. 210 –221.
10. 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.
11. "Arcticus Systems," http://www.arcticus-systems.com.
12. 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.

13. 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.
14. 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.
15. S. Mubeen, J. Mäki-Turja and M. Sjödin, "Extending response-time analysis of controller area network (CAN) with FIFO queues for mixed messages," in *16th IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, sept. 2011, pp. 1–4.
16. 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.
17. 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 (WFCS)*, May 2012.
18. A. Szakaly, "Response Time Analysis with Offsets for CAN," Master's thesis, Department of Computer Engineering, Chalmers University of Technology, Nov. 2003.
19. 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 (ERTS)*, 2008.
20. Grenier, Mathieu and Havet, Lionel and Navet, Nicolas, "Scheduling messages with offsets on controller area network - a major performance boost," in *Automotive Embedded Systems Handbook. CRC Press, 2009*, 2008.
21. 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.
22. 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.
23. 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.
24. 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.
25. K. Tindell and A. Burns, "Guaranteeing Message Latencies on Controller Area Network (CAN)," in *1st International CAN Conference, 1994*, pp. 1 –11.
26. "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.
27. "Requirements on Communication, Release 3.0, Rev 7, Ver. 2.2.0. The AUTOSAR Consortium, Sep., 2010," www.autosar.org/download/R3.0/AUTOSAR_SRS_COM.pdf.
28. "AUTOSAR Techincal Overview, Version 2.2.2., Release 3.1, The AUTOSAR Consortium, Aug., 2008," http://autosar.org.
29. "Hägglunds Controller Area Network (HCAN), Network Implementation Specification," *BAE Systems Hägglunds, Sweden (internal document)*, April 2009.
30. J. Palencia and M. G. Harbour, "Schedulability Analysis for Tasks with Static and Dynamic Offsets," *IEEE International Symposium on Real-Time Systems (RTSS)*, 1998.
31. K. Chaaban, P. Leserf, and S. Saudrais, "Steer-by-wire system development using AUTOSAR methodology," in *14th IEEE Conference on Emerging Technologies and Factory Automation*, 2009.