Message response time analysis for ideal controller area network (CAN) refuted

Reinder J. Bril and Johan J. Lukkien
Technische Universiteit Eindhoven (TU/e),
Den Dolech 2, 5600 AZ Eindhoven, The Netherlands
$\{$*r.j.bril, j.j.lukkien*$\}$*@tue.nl*

Rob I. Davis and Alan Burns
University of York,
York, Y01 5DD, England
$\{$*rob.davis, burns*$\}$*@cs.york.ac.uk*

Abstract

This paper revisits basic message response time analysis of controller area network (CAN). We show that existing message response time analysis, as presented in [10], is optimistic. Assuming discrete scheduling, the problem can be resolved by applying worst-case response time analysis for fixed-priority non-preemptive scheduling (FPNS) as described in [4].

1 Introduction

Controller Area Network (CAN) is a serial, broadcast, bus for sending and receiving short real-time control messages, consisting of between 1 and 8 bytes, and has been designed to operate at speeds of up to 1 Mbit/sec. CAN was originally developed for the automotive industry, and is now used in numerous industrial applications.

Analysis of worst-case message response times for CAN has been pioneered in [10], based on the observation that scheduling messages on a CAN bus is analogous to scheduling tasks by fixed priorities. Because CAN messages are non-preemptive, the existing worst-case response time analysis for fixed-priority preemptive scheduling (FPPS) has been updated to take account of tasks being non-preemptive, i.e. resulting in worst-case response time analysis for fixed-priority non-preemptive scheduling (FPNS). The result has subsequently been applied to CAN.

In this paper, we show that worst-case response time analysis for FPNS with arbitrary phasing and deadlines within periods, as presented in [10], is optimistic. As a result, the worst-case message response time analysis for CAN is also optimistic. Assuming discrete scheduling, the problem can be resolved by applying worst-case response time analysis for FPNS as described in [4].

This paper is organized as follows. Section 2 briefly describes a real-time scheduling model for FPNS. Response time analysis for FPNS is recapitulated in Section 3. In Section 4, we present two examples that refute the analysis in [10]. Whereas the first example is primarily meant for illustration purposes, the second example is based on realistic worst-case transmission times for CAN. The section includes an analysis based on results for FPNS as presented in [4]. The paper is concluded in Section 5.

2 Real-time scheduling models

This section describes a basic scheduling model for FPPS and a refined model for FPNS. Most of the definitions and assumptions of these models originate from [8].

2.1 Basic model for FPPS

We assume a single processor and a set $\mathcal{T}$ of n periodically released, independent tasks $\tau_1, \tau_2, \ldots, \tau_n$. At any moment in time, the processor is used to execute the highest priority task that has work pending.

Each task τ_i is characterized by a (*release*) *period* $T_i \in \mathbb{R}^+$, a *computation time* $C_i \in \mathbb{R}^+$, a (*relative*) *deadline* $D_i \in \mathbb{R}^+$, where $C_i \leq \min(D_i, T_i)$, and a *phasing* $\varphi_i \in \mathbb{R}$. An *activation* (or *release*) *time* is a time at which a task τ_i becomes ready for execution. A release of a task is also termed a *job*. The job of task τ_i with release time φ_i serves as a reference activation, and is referred to as job zero. The release of job k of τ_i therefore takes place at time $a_{ik} = \varphi_i + kT_i$, $k \in \mathbb{Z}$. The deadline of job k of τ_i takes place at $d_{ik} = a_{ik} + D_i$. The set of phasings φ_i is termed the phasing φ of the task set $\mathcal{T}$.

The *response interval* of job k of τ_i is defined as the time span between the activation time of that job and its completion time c_{ik}, i.e. $[a_{ik}, c_{ik})$. The *response time* r_{ik} of job k of τ_i is defined as the length of its response interval, i.e. $r_{ik} = c_{ik} - a_{ik}$. The *worst-case response time* WR_i of a task τ_i is the largest response time of any of its jobs, i.e.

$$WR_i = \sup_{\varphi, k} r_{ik}. \qquad (1)$$

A *critical instant* of a task is defined as an (hypothetical)

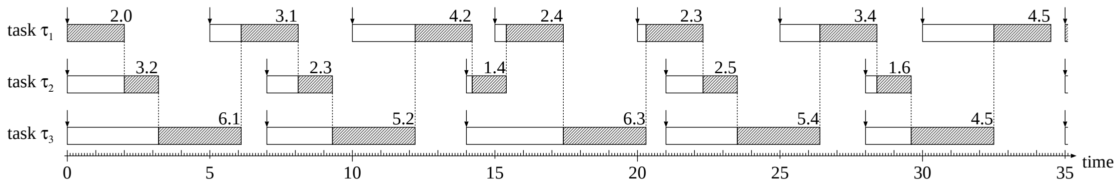

Figure 1. Timeline for $\mathcal{T}_1$ under FPNS with a simultaneous release at time zero. The numbers at the top right corner of the boxes denote the response times of the respective releases.

instant that leads to the worst-case response time for that task.

We assume that we do not have control over the phasing φ, for instance since the tasks are released by external events, so we assume that any arbitrary phasing may occur. This assumption is common in real-time scheduling literature [5, 6, 8]. We also assume other standard basic assumptions [8], i.e. tasks are ready to run at the start of each period and do no suspend themselves, tasks will be preempted instantaneously when a higher priority task becomes ready to run, a job of a task does not start before its previous job is completed, and the overhead of context switching and task scheduling is ignored. Finally, we assume that the deadlines are hard, i.e. each job of a task must be completed before its deadline. Hence, a set $\mathcal{T}$ on n periodic tasks can be scheduled if and only if

$$WR_i \leq D_i \tag{2}$$

for all $i = 1, \ldots, n$.

For notational convenience, we assume that the tasks are given in order of decreasing priority, i.e. task τ_1 has highest priority and task τ_n has lowest priority.

2.2 Refined model for FPNS

For FPNS, we need to refine our basic model of Section 2.1. Unlike FPPS, tasks are no longer instantaneously preempted when a higher priority task becomes ready to run, but are allowed to complete their execution. As a result, the processor need not execute the highest priority task that has work pending at a particular moment in time.

3 Recapitulation of existing analysis

In this section, we recapitulate worst-case response time analysis for FPPS and worst-case message response time analysis for CAN. The latter is based on worst-case response time analysis for FPNS. Because we discuss response times under both FPPS and FPNS, we will use subscripts P and N to denote FPPS and FPNS, respectively.

3.1 Worst-case response time analysis for FPPS

To determine worst-case response times under arbitrary phasing, it suffices to consider only critical instants. For FPPS, critical instants are given by time points at which all tasks have a simultaneous release [8].

From this notion of critical instants, Joseph and Pandya [5] have derived that for deadlines within periods (i.e. $D_i \leq T_i$) the worst-case response time WR_i^{P} of a task τ_i is given by the smallest $x \in \mathbb{R}^+$ that satisfies

$$x = C_i + \sum_{j<i} \left\lceil \frac{x}{T_j} \right\rceil C_j. \tag{3}$$

To calculate worst-case response times, we can use an iterative procedure based on recurrence relationships [1]. The procedure starts with a lower bound.

$$\begin{aligned} wr_i^{(0)} &= \sum_{j\leq i} C_j \\ wr_i^{(k+1)} &= C_i + \sum_{j<i} \left\lceil \frac{wr_i^{(k)}}{T_j} \right\rceil C_j \end{aligned}$$

The procedure is stopped when the same value is found for two successive iterations of k or when the deadline D_i is exceeded. In the former case, it yields the smallest solution of the recursive equation, i.e. the worst-case response time of τ_i. In the latter case the task is not schedulable. Termination of the procedure is ensured by the fact that the sequence $wr_i^{(k)}$ is bounded (from below by C_i, and from above by D_i) and non-decreasing, and that different values for successive iterations differ by at least $\min_{j<i} C_j$.

The interested reader is referred to [6, 7, 9] for techniques to derive worst-case response times for arbitrary deadlines. The main difference with deadlines within periods is that for arbitrary deadlines the worst-case response time of a task is not necessarily assumed for the first job that is released at the critical instant.

3.2 Message response time analysis for CAN

In this section, we recapitulate basic message response time analysis for ideal CAN. To this end, we first present the update of [5] given in [10] to take account of tasks being non-preemptive. Next, we recapitulate how the updated analysis can be applied to CAN as described in [10]. The analysis assumes deadlines within periods, i.e. $D_i \leq T_i$.

The non-preemptive nature of tasks may cause blocking of a task by at most one lower priority task. The maximum

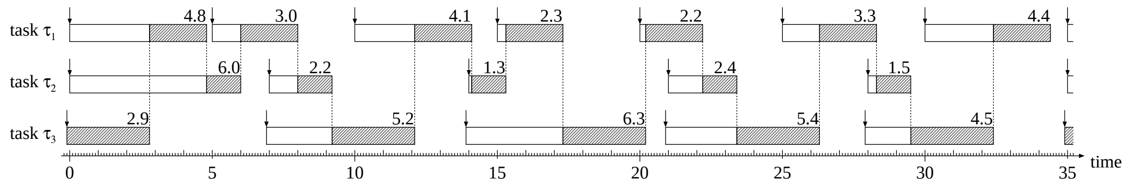

Figure 2. Timeline for $\mathcal{T}_1$ under FPNS with a release at time 0 for τ_1 and τ_2, and at time -0.1 for τ_3.

blocking B_i of task τ_i by a lower priority task is equal to the longest computation time of a task with a priority lower than task τ_i, i.e.

$$B_i = \max_{j>i} C_j. \tag{4}$$

The worst-case response time $\widetilde{WR}_i^{\mathrm{N}}$ is given by

$$\widetilde{WR}_i^{\mathrm{N}} = w_i + C_i, \tag{5}$$

where w_i is the smallest $x \in \mathbb{R}^+$ that satisfies

$$x = B_i + \sum_{j<i} \left\lceil \frac{x+\tau_{res}}{T_j} \right\rceil C_j. \tag{6}$$

In this latter equation, τ_{res} is the resolution with which time is measured. To calculate w_i, an iterative procedure based on recurrence relationships can be used. An appropriate initial value of this procedure is $w_i^{(0)} = B_i + \sum_{j<i} C_j$.

Because scheduling messages on a CAN bus is analogous to scheduling tasks by fixed priorities, the analysis given above can be used to determine the worst-case message response time for CAN. A message μ_i has a *period* T_i, a *worst-case transmission time* C_i, and a (*relative*) *deadline* D_i, where C_i is a function of the number of message bytes b_i of μ_i. On a CAN bus, one deals with time units as multiples of the bit-time, which is denoted as τ_{bit}, i.e. $\tau_{res} = \tau_{bit}$ in (6). With a 1Mbit/sec bus, τ_{bit} is equal to $1\mu s$. The worst-case message response time can now be derived using equations (4), (5), and (6).

4 Counterexamples

The task characteristics of our first counterexample are given in Table 1. The table includes the worst-case response times of the example as determined by means of [10] and [4]. Note that the (*processor*) *utilization factor U* of the task

task	T	C	$\widetilde{WR}^{\mathrm{N}}$ ([10])	WR^{N} ([4])
τ_1	5	2	4.9	4.8
τ_2	7	1.2	6.1	6.0
τ_3	7	2.9	6.1	6.3

Table 1. Task characteristics of $\mathcal{T}_1$ and worst-case response times under FPNS.

set $\mathcal{T}_1$ is given by $U = \frac{2}{5} + \frac{1.2}{7} + \frac{2.9}{7} \approx 0.986$. This example will be used for illustration purposes.

Table 2 presents message characteristics with realistic worst-case transmission times for CAN (Version 2.0 A, standard format), including the worst-case message response times for ideal CAN. Note that $\mathcal{M}_2$ has a utilization

message	T	C	$\widetilde{WR}^{\mathrm{N}}$ ([10])	WR^{N} ([4])
μ_1	221	85	220	219
μ_2	286	65	285	284
μ_3	348	135	285	341

Table 2. Message characteristics (as multiples of τ_{bit}) of $\mathcal{M}_2$ and worst-case message response times for ideal CAN.

$U = \frac{85}{221} + \frac{65}{286} + \frac{135}{348} \approx 0.982$.

4.1 Existing analysis for CAN is optimistic

We will now show that the worst-case response time of task τ_3 as determined by (4), (5) and (6) is optimistic.

Based on (6) and (4), and using $\tau_{res} = 0.1$, we derive

$$\begin{aligned} w_3^{(0)} &= B_3 + C_1 + C_2 = 0 + 2.0 + 1.2 = 3.2 \\ w_3^{(1)} &= B_3 + \sum_{j<3} \left\lceil \frac{w_3^{(0)} + \tau_{res}}{T_j} \right\rceil C_j \\ &= 0 + \left\lceil \frac{3.2+0.1}{5} \right\rceil \cdot 2.0 + \left\lceil \frac{3.2+0.1}{7.0} \right\rceil \cdot 1.2 \\ &= 3.2, \end{aligned}$$

and we find $w_3 = 3.2$. Using (5), we now get $\widetilde{WR}_3^{\mathrm{N}} = 3.2 + 2.9 = 6.1$. Similarly, we find $\widetilde{WR}_1^{\mathrm{N}} = 4.9$ and $\widetilde{WR}_2^{\mathrm{N}} = 6.1$.

Figure 1 shows a timeline with the executions of the three tasks of $\mathcal{T}_1$ in an interval of length 35, i.e. equal to the *hyperperiod H* of the tasks, which is equal to the least common multiple (lcm) of the periods. The schedule in $[0,35)$ is repeated in the intervals $[hH,(h+1)H)$ with $h \in \mathbb{Z}$, i.e. the schedule is periodic with period H. As illustrated in Figure 1, the derived value for $\widetilde{WR}_3^{\mathrm{N}}$ corresponds with the response time of the 1st job of task τ_3 upon a simultaneous release with tasks τ_1 and τ_2. However, the response time of the 3rd job of task τ_3 is equal to 6.3 in that figure, illustrating that the existing analysis is optimistic.

We merely mention that the existing analysis is also optimistic for the example given in Table 2.

4.2 Discussion

Above, we have shown that even when deadlines are within periods, we cannot restrict ourselves to the response time of a single job of a task when determining the worst-case response time of that task under FPNS. The reason for this is that a job of task τ_i can defer the execution of higher priority tasks, which can potentially give rise to higher interference for subsequent jobs of task τ_i. We observe that the origin of the problem is basically the same as described in [3] for the problem with existing analysis for worst-case response times for fixed-priority scheduling with deferred preemption (FPDS) with arbitrary phasing and deadlines within periods.

In [4], worst-case response time analysis is presented for FPNS with *arbitrary* deadlines, arbitrary phasing, and *discrete* (rather than continuous) scheduling [2]. For discrete scheduling, all task parameters are restricted to integers, and tasks are scheduled at integer times. For completeness, Lemma 6 and Theorem 15 of [4] are given below, with minor modifications to match our terminology and scheduling model. The lemma describes a critical instant for task τ_i.

Lemma 1 *The worst-case response time of τ_i is found in a level-i busy period by releasing all tasks τ_j with $j \leq i$ simultaneously at time $t = 0$, and by releasing the longest task τ_k with $k > i$, if any, at time $t = -1$.*

Theorem 1 *Given a task set $\mathcal{T}$ consisting of n tasks $\tau_1, \ldots, \tau_n$, the worst-case response time of any task τ_i is given by*

$$WR_i^{\mathrm{N}} = \max_{q=0,\ldots,Q} \{w_{i,q} + C_i - qT_i\}, \tag{7}$$

where

$$w_{i,q} = qC_i + \sum_{j<i} \left(1 + \left\lfloor \frac{w_{i,q}}{T_j} \right\rfloor\right) C_j + \max_{k>i}\{C_k - 1\}, \tag{8}$$

and $Q = \left\lfloor \frac{L_i}{T_i} \right\rfloor$, where L_i is the length of the longest level-i busy period in non-preemptive context, which is given by the smallest positive integer l satisfying the following equation

$$l = \max_{j>i}\{C_j - 1\} + \sum_{j \leq i} \left\lceil \frac{l}{T_j} \right\rceil C_j. \tag{9}$$

The worst-case response times of the tasks of $\mathcal{T}_1$ as determined by this analysis are also included in Table 1. In order to make the analysis applicable, we first multiplied all task parameters with 10, subsequently performed the analysis, and finally divided the resulting worst-case response times by 10. Based on Lemma 1, we conclude that the worst-case response times of tasks τ_1 and τ_2 are illustrated in Figure 2, and of task τ_3 in Figure 1.

5 Conclusion

In this document, we revisited basic worst-case message response times for ideal controller area network (CAN). We showed by means of examples with a high load ($\approx 98\%$) that the analysis as presented in [10] is optimistic. Assuming discrete scheduling, the problem can be resolved by applying the analysis for FPNS presented in [4].

Worst-case response time analysis under FPNS with arbitrary phasing for continuous scheduling is a topic of future work.

Acknowledgement

We thank M.A. (Alina) Weffers-Albu and P.H.F.M. (Richard) Verhoeven from the TU/e for their comments on an earlier version of this document.

References

[1] N. Audsley, A. Burns, M. Richardson, and A. Wellings. Hard real-time scheduling: The deadline monotonic approach. In *Proc.* 8th *IEEE Workshop on Real-Time Operating Systems and Software (RTOSS)*, pp. 133–137, May 1991.

[2] S. Baruah, L. Rosier, and R. Howell. Algorithms and complexity concerning the preemptive scheduling of periodic, real-time tasks on one processor. *Real-Time Systems*, 2:301–324, 1990.

[3] R. Bril. Existing worst-case response time analysis of real-time tasks under fixed-priority scheduling with deferred preemption is too optimistic. CS-Report 06-05, Technische Universiteit Eindhoven (TU/e), February 2006.

[4] L. George, N. Rivierre, and M. Spuri. Preemptive and non-preemptive real-time uni-processor scheduling. Technical Report 2966, Institut National de Recherche et Informatique et en Automatique (INRIA), France, September 1996.

[5] M. Joseph and P. Pandya. Finding response times in a real-time system. *The Computer Journal*, 29(5):390–395, 1986.

[6] M. Klein, T. Ralya, B. Pollak, R. Obenza, and M. González-Harbour. *A Practitioner's Handbook for Real-Time Analysis: Guide to Rate Monotonic Analysis for Real-Time Systems*. Kluwer Academic Publishers, 1993.

[7] J. Lehoczky. Fixed priority scheduling of periodic task sets with arbitrary deadlines. In *Proc.* 11th *IEEE Real-Time Systems Symposium (RTSS)*, pp. 201–209, December 1990.

[8] C. Liu and J. Layland. Scheduling algorithms for multiprogramming in a real-time environment. *Journal of the ACM*, 20(1):46–61, 1973.

[9] K. Tindell. An extendible approach for analysing fixed priority hard real-time tasks. Report YCS 189, Dep. of Computer Science, University of York, December 1992.

[10] K. Tindell, H. Hansson, and A. Wellings. Analysing real-time communications: Controller area network (CAN). In *Proc.* 15th *IEEE Real-Time Systems Symposium (RTSS)*, pp. 259–263, December 1994.