Session 2 - Old Problems revisited

Chair: Luis Almeida

Rapporteur: Björn Andersson

Session Summary: Old Problems Revisited

Chair : Luis Almeida
DET/IEETA, Universidade de Aveiro
Aveiro, Portugal
lda@det.ua.pt

Rapporteur: Björn Andersson
DEI-ISEP, Polytechnic Institute of Porto
Porto, Portugal
bandersson@dei.isep.ipp.pt

1. Luis Report

This was a short session with one paper dedicated to response-time analysis issues. The paper, entitled *Message response time analysis for ideal controller area network (CAN) refuted* was presented by R. Bril and co-authored by J. Lukkien, both from TU Eindhoven and R. Davis and A. Burns from the University of York. The paper basically shows that the well known analysis to deduce the worst-case response time of messages in CAN, initially presented by Ken Tindell in 1994, is optimistic in some cases. In fact, for such cases, the worst-case response time of a message does not occur when it is released synchronously with all higher priority ones. The cause seems to be the blocking that a previous instance of a given message can cause to higher priority messages leading to higher interference on the next instance of the same message. Curiously, this effect is known for many years in the context of non-preemptive task scheduling and appropriate analysis was proposed, which is based on the fact that the worst-case response time still occurs in the synchronous busy interval.

Thus, because of the large impact that Tindell's work had on the real-time analysis developed for CAN in the past 12 years, this paper was awaited with some anxiety. A lively discussion took place after the presentation trying to understand the problem, its probability of occurrence and conditions that can lead to its occurrence. It was acknowledged that the situation indicated is relatively rare, which is also confirmed by the time that it took to find it. Also, it was acknowledged that such situation is not necessarily associated with very high utilization levels. The discussion ended considering whether Tindell's analysis could be adapted, with some non-optimal parameter, e.g. extra blocking or release jitter, to cope with the found situation but, as R. Bril indicated, it does not seem likely.

.

2. Björn Report

The paper claims that the schedulability analysis published (by Ken Tindell) on the CAN bus is not a sufficient schedulability test. None of the workshop participants disagreed on that. Figure 1 in the paper shows that the highest priority task τ_1 can cause more than C_1 interference on task τ_3. A question was brought whether it is only the highest priority task that causes more interference than the previously published CAN analysis expresses and the author gave the answer that there are task sets where the two highest priority tasks cause more interference than the previously proposed analysis. It was discussed if the previous analysis is correct for certain restricted task sets; in particular one of the workshop participants asked if the CAN analysis is incorrect for low utilization; say less than 50%. For the system model used in the paper; the workshop did not give an answer. For systems with non-zero jitter, the author claimed that there are task sets with a utilization close to 0% where the CAN analysis (by Ken Tindell) is not sufficient. It was discussed whether this analysis carry over to another scheduling problems that are non-preemptive-like, for example PCP. No clear answer was given by the author or the workshop attendees but the general intuition of the workshop attendees was that the analysis of PCP remains valid.

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

Robert 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 [17], 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 [6].

1 Introduction

Controller Area Network (CAN) is a serial, broadcast, bus for sending and receiving short real-time control messages, consisting of between 0 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 [1, 7]. Currently, it is not only a widely used vehicular network, with more than 100 million CAN nodes sold in 2000 [10], but it is also used in numerous industrial applications.

Analysis of worst-case message response times for CAN has been pioneered in [17], 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. The analysis is well-known and has been used widely in the academic literature and in industrial practice. The analysis presented in [15, 16] is similar to the analysis of [17].

In this paper, we show that worst-case response time analysis for FPNS with arbitrary phasing and deadlines within periods, as presented in [17], is optimistic. As a result, the worst-case message response time analysis for ideal CAN is also optimistic. The response time of a message can therefore be larger than the worst-case message response time as determined by the analysis presented in [17], and an unschedulable set of messages can therefore incorrectly be considered schedulable. Assuming discrete scheduling, the problem can be resolved by applying worst-case response time analysis for FPNS as described in [6].

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 [17]. Whereas the first example is primarily meant for illustration purposes, the second example is based on realistic worst-case transmission times for CAN. Section 5 recapitulates the worst-case response time analysis for FPNS under discrete scheduling as described in [6], and presents the results of that analysis for the examples of Section 4. The paper is concluded in Section 6.

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 [12].

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 time $d_{ik} = a_{ik} + D_i$.

The set of phasings φ_i is termed the phasing φ of the task set $\mathcal{T}$. 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 [8, 9, 12].

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.

$$WR_i = \sup_{\varphi,k} r_{ik}. \tag{1}$$

A *critical instant* of a task is defined as an (hypothetical) instant that leads to the worst-case response time for that task.

As well as arbitrary phasing, we also assume other standard basic assumptions [12], i.e. tasks are ready to run at the start of each period and do not 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 $\mathcal{T}$ of 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 the highest priority and task τ_n has the 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 ideal 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 [12].

From this notion of critical instants, Joseph and Pandya [8] 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 [2]. 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 [9, 11, 14] for techniques to derive worst-case response times for tasks with 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 [8] given in [17] to take account of tasks being non-preemptive. Next, we recapitulate how the updated analysis can be applied to CAN as described in [17]. 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 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.

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

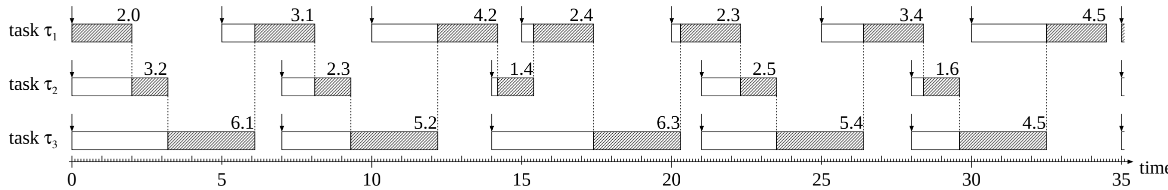

Figure 1. Timeline for $\mathcal{T}_1$ under FPNS with a simultaneous release at time zero. The numbers at the top right corner of the boxes denote the response times of the respective releases. Note that response time is counted from the moment of release up to the corresponding completion.

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 for 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 for FPNS, like 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. 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 Equation (6). With a 1Mbit/sec bus, τ_{bit} is equal to $1\mu s$. In this paper, we express the message characteristics T_i, C_i and D_i as multiples of τ_{bit}. Based on Version 2.0 A, standard format [1], we use for C_i

$$C_i = 47 + 8b_i + \left\lfloor \frac{34 + 8b_i - 1}{4} \right\rfloor = 55 + 10b_i \tag{7}$$

where b_i is the number of data bytes in the message (i.e. $b_i \in \{0, 1, \ldots, 8\}$), 47 is the number of control bits in a CAN frame, and 34 is the number of control bits that are subject to bit-stuffing. Bit-stuffing is required, because six consecutive bits of the same polarity (i.e. 111111 or 000000) are used for error signaling in CAN. A bit of opposite polarity is therefore inserted after five consecutive bits of the same polarity, giving rise to the floor-function and the numbers 1 and 4 in the equation.

The worst-case message response time can now be derived using Equations (4), (5), and (6). In the next section, we will show that analysis based on these equations can be optimistic.

4 Counterexamples

In this section, we give two examples that refute the existing analysis in [17]. Whereas the first example is primarily meant for illustration purposes, the second example is based on realistic worst-case transmission times for CAN.

4.1 Analysis for FPNS is optimistic

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 [17] and [6]. Note that the (*processor*) *utilization factor* U of the

task	$T = D$	C	$\widetilde{WR}^{\mathrm{N}}$ ([17])	WR^{N} ([6])
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.

task set $\mathcal{T}_1$ is given by $U = \frac{2}{5} + \frac{1.2}{7} + \frac{2.9}{7} \approx 0.986$.

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

Based on Equations (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 Equation (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 *hy-*

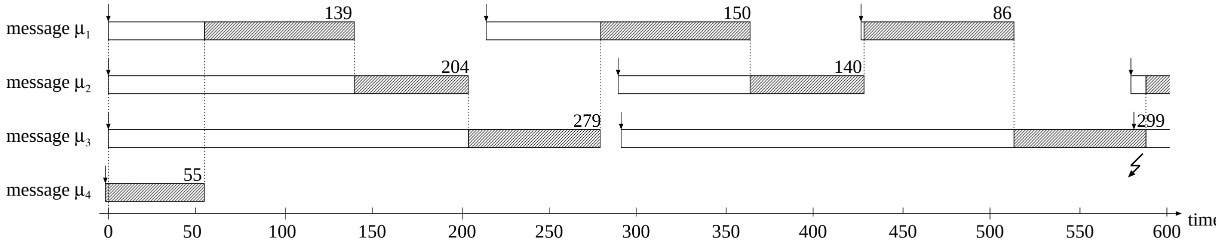

Figure 2. Timeline for $\mathcal{M}_2$ with a transmission at time 0 for μ_1, μ_2, and μ_3, and at time -1 for μ_4.

perperiod 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 to 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.

4.2 Existing analysis for CAN is optimistic

Table 2 presents message characteristics of a message set $\mathcal{M}_2$ with realistic worst-case transmission times for CAN, including the worst-case message response times for ideal CAN. Messages μ_1 to μ_4 contain 3, 1, 2, and 0 data bytes, respectively; see also Equation (7). Note that $\mathcal{M}_2$ has a utilization $U = \frac{85}{214} + \frac{65}{289} + \frac{75}{290} + \frac{55}{3000} \approx 0.90$.

message	$T = D$	C	$\widetilde{WR}^{\mathrm{N}}$ ([17])	WR^{N} ([6])
μ_1	214	85	160	159
μ_2	289	65	225	224
μ_3	290	75	280	299
μ_4	3000	55	590	590

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

We will now show that the worst-case response time of message μ_3 as determined by Equations (4), (5) and (6) is also optimistic.

Based on Equations (6) and (4), and using $\tau_{res} = \tau_{bit} = 1$, we derive

$$\begin{aligned}
w_3^{(0)} &= B_3 + C_1 + C_2 = 55 + 85 + 65 = 205 \\
w_3^{(1)} &= B_3 + \sum_{j<3} \left\lceil \frac{w_3^{(0)} + \tau_{bit}}{T_j} \right\rceil C_j \\
&= 55 + \left\lceil \frac{205+1}{214} \right\rceil \cdot 85 + \left\lceil \frac{205+1}{289} \right\rceil \cdot 65 \\
&= 205,
\end{aligned}$$

and we find $w_3 = 205$. Using Equation (5), we now get $\widetilde{WR}_3^{\mathrm{N}} = 205 + 75 = 280$. Similarly, we find $\widetilde{WR}_1^{\mathrm{N}} = 160$, $\widetilde{WR}_2^{\mathrm{N}} = 225$, and $\widetilde{WR}_4^{\mathrm{N}} = 590$. Hence, according to the existing analysis the set of messages is schedulable on a CAN bus.

Figure 2 shows a timeline with a transmission at time $t = 0$ for messages μ_1, μ_2, and μ_3, and at time $t = -1$ for message μ_4. As illustrated in Figure 2, the 2nd transmission of message μ_3 has a response time of 299. This value is not only larger than the derived value for $\widetilde{WR}_3^{\mathrm{N}} = 280$, but also larger than the deadline $D_3 = 290$. Hence, although the set of messages is deemed schedulable according to the existing analysis, it is actually unschedulable. The existing analysis is therefore also optimistic for the example given in Table 2.

4.3 Cause of optimism in existing analysis

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. This is illustrated in Figure 1, amongst others. The 1st job of task τ_3 experiences an interference of 3.2, corresponding to the sum of the computation times of tasks τ_1 and τ_2. The 3rd job of τ_3 experiences an additional interference of 0.2 because the 3rd job of τ_1 is deferred by the 2nd job of τ_3.

We observe that the origin of the problem is basically the same as described in [4] 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. A similar issue with work on preemption thresholds [18] was first identified and corrected by Regehr [13] in 2002.

5 CAN analysis based on discrete scheduling

In [6], worst-case response time analysis is presented for FPNS with *arbitrary* deadlines, arbitrary phasing, and

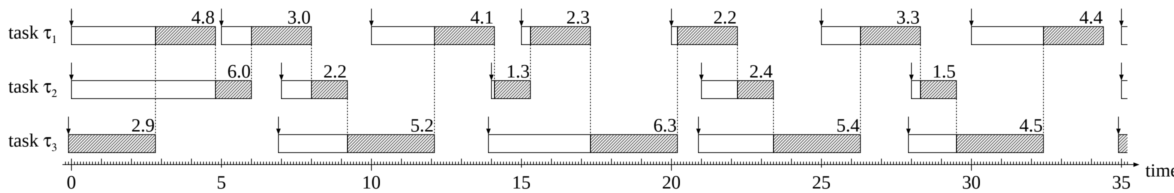

Figure 3. Timeline for $\mathcal{T}_1$ under FPNS with a release at time 0 for τ_1 and τ_2, and at time -0.1 for τ_3.

discrete (rather than continuous) scheduling [3]. For discrete scheduling, all task parameters are restricted to integers, and tasks are scheduled at integer times. Assuming discrete scheduling for CAN, the problem with the existing analysis can be resolved by applying the analysis for FPNS as described in [6]. In this section, we first recapitulate the analysis from [6]. Next, we present the results of applying the analysis to the counterexamples given in Section 4. We conclude this section with a remark about the differences between the values for $\widetilde{WR}^{\mathrm{N}}$ and WR^{N}.

5.1 Analysis for FPNS for discrete scheduling

To recapitulate the worst-case response time analysis as presented for FPNS in [6], Lemma 6 and Theorem 15 of that report 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 $\tau_1, \ldots,$ τ_n, the worst-case response time of any task τ_i is given by*

$$WR_i^{\mathrm{N}} = \max_{q=0,\ldots,Q} \{w_{i,q} + C_i - qT_i\}, \tag{8}$$

where

$$w_{i,q} = qC_i + \sum_{j<i} \left(1 + \left\lfloor \frac{w_{i,q}}{T_j} \right\rfloor \right) C_j + \max_{k>i} \{C_k - 1\}, \tag{9}$$

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

$$l = \max_{j>i} \{C_j - 1\} + \sum_{j \leq i} \left\lceil \frac{l}{T_j} \right\rceil C_j. \tag{10}$$

We observe that equation $Q = \left\lfloor \frac{L_i}{T_i} \right\rfloor$ in Theorem 1 yields a value that is one too large when the length L_i of the longest level-i busy period is an integer multiple of the period T_i. This can be easily resolved by using the equation $Q = \left\lceil \frac{L_i}{T_i} \right\rceil - 1$ instead. Although the existing equation does not give rise to problems, i.e. Equation (9) is just evaluated one extra, we prefer this more efficient formulation.

5.2 Counterexamples revisited

The worst-case response times WR^{N} of the tasks of $\mathcal{T}_1$ as determined by the analysis of [6] 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 3, and of task τ_3 in Figure 1.

Similarly, Table 2 includes the worst-case message response times WR^{N} of the messages of $\mathcal{M}_2$. Based on Lemma 1, we conclude that the worst-case message response time WR_3^{N} of message μ_3 is illustrated in Figure 2.

5.3 Concluding remarks

Considering Tables 1 and 2, it is remarkable that the values for $\widetilde{WR}^{\mathrm{N}}$ and WR^{N} are different for all but the lowest priority message μ_4. The optimism in $\widetilde{WR}_3^{\mathrm{N}}$ for task τ_3 in Table 1 and message μ_3 in Table 2 has already been explained in Section 4.3. This section deals with the differences in the other values.

We observe that the characteristics of the tasks and messages of both our counterexamples are integral multiples of a value $\delta \geq \tau_{res}$. As a consequence, reducing τ_{res} to an arbitrary small positive value does not change the values for $\widetilde{WR}^{\mathrm{N}}$ in either Table 1 or Table 2. Moreover, using τ_{res} and a ceiling function in Equation (6) therefore also has the same effect for our counterexamples as using a floor function and an addition term $+1$ in Equation (9). Hence, the differences are not caused by the usage of τ_{res}. Instead, the cause of the differences is found in the values used for the maximum blocking, i.e. Equation (9) includes an additional term -1 when compared to Equation (4). Note that $\widetilde{WR}_4^{\mathrm{N}} = WR_4^{\mathrm{N}}$ in Table 2 because the maximum blocking is, in both cases, equal to zero for the lowest priority message.

6 Conclusion

In this document, we revisited basic worst-case message response times for ideal controller area network (CAN). We showed by means of two examples with a high load (i.e. of $\approx$ 99% and $\approx$ 90%) that the analysis as presented in [17] is optimistic. Assuming discrete scheduling, the problem can be resolved by applying the analysis for FPNS presented in [6].

We are currently investigating how the optimism scales, i.e. whether or not the existing analysis can result in optimistic results for *any* task (or message) given an arbitrary number of tasks (or messages). We are also investigating whether or not optimistic results can occur for task(or message) sets with low utilization. Worst-case response time analysis under FPNS for continuous scheduling is a topic of future work.

Acknowledgements

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 [5], a document on which this paper is based.

References

[1] *CAN specification.* Technical Report Version 2.0, Bosch, Postfach 50, D-700 Stuttgart 1, September 1991.

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

[3] 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.

[4] 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, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven (TU/e), The Netherlands, February 2006.

[5] R. Bril, J. Lukkien, R. Davis, and A. Burns. *Message response time analysis for ideal controller area network (CAN) refuted.* CS-Report 06-19, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven (TU/e), The Netherlands, May 2006.

[6] 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.

[7] ISO. *ISO 11898 road vehicles - Interchange of Digital Information - Controller Area Network (CAN) for high speed communication.* Technical report, International Standards Organization, November 1993.

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

[9] 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.

[10] G. Lee and D. Heffernan. Expanding automotive electronic systems. *IEEE Computer*, 35(1):88–93, January 2002.

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

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

[13] J. Regehr. Scheduling tasks with mixed preemption relations for robustness to timing faults. In *Proc.* 23^{rd} *IEEE Real-Time Systems Symposium (RTSS)*, pp. 315–326, December 2002.

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

[15] K. Tindell and A. Burns. *Guaranteeing message latencies for distributed safity-critical hard real-time control networks.* Report YCS 229, Department of Computer Science, University of York, June 1994.

[16] K. Tindell, A. Burns, and A. Wellings. Calculating controller area network (CAN) message response times. *Control Engineering Practice*, 3(8):1163–1169, August 1995.

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

[18] Y. Wand and M. Saksena. Scheduling fixed-priority tasks with preemption threshold. In *Proc.* 6^{th} *International Conference on Real-Time Computing Systems and Applications (RTCSA)*, pp. 328–335, December 1999.