Schedulability Analysis and Message Schedule Computation for the Dynamic Segment of FlexRay

Klaus Schmidt
Department of Electronic and Communication Engineering
Cankaya University, Ankara, Turkey
Email: schmidt@cankaya.edu.tr

Ece G. Schmidt
Department of Electrical and Electronics Engineering
Middle East Technical University, Ankara, Turkey
Email: eguran@metu.edu.tr

***Abstract*—In this paper, we perform the schedulability analysis and schedule computation for sporadic real-time messages in the dynamic segment of the FlexRay protocol. We first formulate a linear integer programming problem that allows to determine if a given message schedule is feasible, i.e., the *worst-case delay* of each message is smaller than its *deadline*. Then, we develop a heuristic algorithm that enables the efficient computation of feasible schedules. Our results are illustrated by an experimental setup with three FlexRay nodes.**

I. Introduction

The functionality of contemporary vehicles is supported by electronic control units (ECUs), which are embedded systems with, e.g., a microcontroller, sensors, and actuators. These ECUs exchange signals that are encapsulated in *messages* over an in-vehicle communication network. The messages can be *periodic* or *sporadic*, requiring time-triggered and event-triggered communication, respectively. In addition, most of the messages require real-time guarantees such as timely delivery.

The FlexRay in-vehicle communication network was founded by an industrial initiative in 2000 [2], [9], [8] to meet the demands of new in-vehicle applications such as x-by-wire. FlexRay is a high-bandwidth, reliable technology with two 10 Mbit/s channels and has a cyclic operation. Each FlexRay cycle consists of a static segment (SS) and a dynamic segment (DS). Periodic messages are transmitted in the SS in unique *static slots* according to time division multiple access (TDMA). Sporadic messages are sent in *dynamic slots* in the DS. In both cases, the timely message delivery depends on the *message schedule* which is statically configured before the network starts to operate. The schedule computation involves assigning the static slots for the periodic messages as well as the priority based dynamic slot assignment to the sporadic messages. A message set is *schedulable* if a given message schedule meets timing requirements such as deadlines.

In this paper, we study the schedulability analysis and schedule construction problem for the DS of FlexRay. We first present a new linear integer programming (LIP) formulation to analyze if all messages in a given schedule meet their deadlines. Then we integrate this schedulability analysis in a scheduling algorithm so as to determine a schedulable priority assignment for the messages in the DS. We illustrate our schedulability analysis and features of our scheduling algorithm by an experimental study performed with FlexRay hardware.

In the previous literature, [10] develops a first schedulability test and provides a schedule computation for the DS. In comparison, our paper allows a more rigorous and efficient problem solution. Furthermore, our previous work in [11] presents an optimal schedule computation for the DS while requiring an additional layer on top of the FlexRay protocol.

The organization of the paper is as follows. In Section II, we describe the functionality of the FlexRay protocol. Section III and IV explain our schedulability analysis and schedule computation, respectively, including an experimental study in Section IV. Conclusions are given in Section V.

II. The FlexRay Protocol

The FlexRay protocol defines two channels A and B that operate at a bandwidth of 10Mbit/s each leading to a bit time of $\texttt{gdBit} = 0.1\,\mu$s.

A. General Overview

The operation of each FlexRay channel is based on a fixed-duration, repeatedly executed *FlexRay cycle* (FC) with a duration `gdCycle` [2]. As depicted in Fig. 1, the FC comprises the *static segment* (SS), the *dynamic segment* (DS), the *symbol window* (SW), and the *network idle time* (NIT). The SS is designed for the periodic transmission of real-time data, while the DS supports the transmission of low-priority data and event-triggered (sporadic) real-time data. The optional SW allows to send certain symbols and the NIT provides time for protocol-related computations such as clock correction. The basic time unit of the protocol operation is the *macrotick* (MT) with a duration of `gdMacrotick` that can be configured between $1\,\mu$s and $6\,\mu$s.

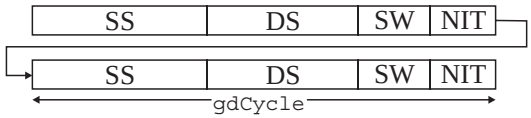

Fig. 1. FlexRay cycle description.

The operation of the FlexRay SS is similar to the time-triggered protocol (TTP) [1], [7] and employs the TDMA approach. Since this paper is concerned with the FlexRay DS, we only introduce the *SS duration* T_{SS}.

The DS is similar to ByteFlight [5] and employs the flexible TDMA (FTDMA) approach. Its operation is shown in Fig. 2. The smallest time unit in the DS is the *minislot* (MS) with a duration of `gdMinislot` (between 2 and 63 MT). The DS comprises a maximum number of `gNumberOfMinislots` (between 0 and 7986) MSs. Frames are transmitted within *dynamic slots* that are superimposed on the minislots. For each channel, the variable `vSlotCounter` captures the ID of the current dynamic slot starting from a pre-configured initial value x. In each dynamic slot, a frame with the corresponding ID is transmitted if present. In that case, the duration of the dynamic slot is determined by the length of the transmitted frame. Otherwise, the duration of the dynamic slot is one MS. The internal variable `zMinislot` counts the MSs in the DS.

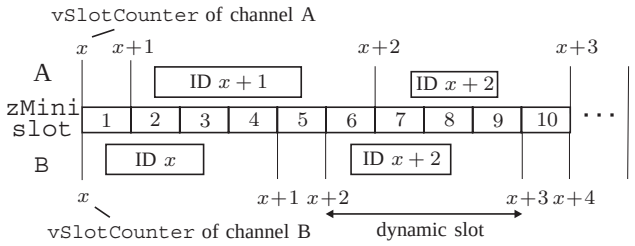

Fig. 2. FlexRay dynamic segment (DS).

According to the protocol specification, the length of each FlexRay frame in `gdBit` evaluates as

$$\texttt{FrameLength}[\texttt{gdBit}] = \texttt{PayloadLength} \cdot 20 + 94, \quad (1)$$

where `PayloadLength` is given in multiples of 2 B words [2]. The actual frame transmission in each dynamic slot happens between the *action points* of two MSs as depicted in Fig. 3 within the *transmission phase* of the dynamic slot. Such action points occur a pre-configured number of `gdMinislotActionPointOffset` MT (between 0 and 31) after the start of the MS. In addition, the dynamic slot includes an *idle phase* with a duration of `gdDynamicSlotIdlePhase` minislots (between 0 and 2).

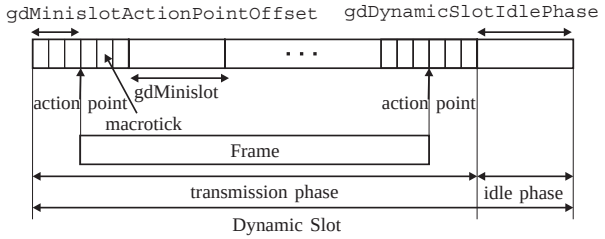

Fig. 3. FlexRay dynamic slot.

Together, the number of MSs needed for the transmission of each frame is

$$\texttt{MinislotPerDynamicFrame}[\text{MS}] = 1 + \left\lceil \frac{0.1003 \cdot \texttt{gdBit} \cdot (\texttt{FrameLength}+1)}{\texttt{gdMacrotick} \cdot \texttt{gdMinislot}} \right\rceil + \texttt{gdDynamicSlotIdlePhase}. \quad (2)$$

It has to be noted that no message can be transmitted in the DS after `zMinislot` reaches the latest transmission instant $\texttt{pLatestTx} = \texttt{gNumberOfMinislots} - \texttt{aMinislotPerDynamicFrame} + 1$, where `aMinislotPerDynamicFrame` is the number of MSs needed for the longest message.

Finally, we describe the duration of the SW and the NIT by the variables `gdSymbolWindow` (between 0 and 142 MT) and `gdNIT` (between 2 and 805 MT), respectively.

For convenience, we use the notational convention in Table I to replace the lengthy variable names from the protocol specification in [2].

`gdMinislot`	T_{MS}	`gdSymbolWindow`	T_{SW}
`gNumberOfMinislots`	N_{MS}	`gdNIT`	T_{NIT}
`gdCycle`	T_{c}	`pLatestTx`	N_{latest}

TABLE I
NOTATIONAL CONVENTION.

B. Scheduling Problem for the Dynamic Segment

The objective of this paper is the scheduling of event-based real-time message frames that are transmitted in the FlexRay DS. We assume that a set $\mathcal{D}$ of such sporadic messages has to be scheduled in the FlexRay DS. Following [6], [10], [11], each message $D \in \mathcal{D}$ is characterized by its *deadline* d_D, its *minimum interarrival time* p_D between two consecutive message generations and its *payload* b_D in two-byte words. Then, we compute the number of MSs for the transmission of D using (2) and denote it as $N_{D,\text{MS}}$.

Based on this message characterization, finding a message schedule for the DS is equivalent to assigning an appropriate MS ID to each message in D such that no message deadline is violated even for the worst-case message generation. Formally, we want to find a map $a_{\text{DS}} : \mathcal{D} \rightarrow \{1, \ldots, \texttt{gNumberOfMinislots}\}$ that determines the MS ID $a_{\text{DS}}(D)$ for each message $D \in \mathcal{D}$.

Note that such scheduling problem is also studied in [10], [11]. However, both papers employ an additional software layer on top of the standard FlexRay protocol to support the efficient use of the dynamic slots in the DS. In contrast, this paper investigates scheduling for the FlexRay DS without any additional software layer. In Section III, we develop a novel method for the schedulability analysis of a given MS assignment for the DS. In Section IV, we propose a heuristic scheduling algorithm for the DS that is based on an iterative application of the developed schedulability analysis.

III. SCHEDULABILITY ANALYSIS FOR THE DYNAMIC SEGMENT

The schedulability analysis for the DS emanates from the assumption that the map a_{DS} has already been determined for a given message set $\mathcal{D}$. Then, it is desired to find out if the worst-case delay experienced by each of the messages in $\mathcal{D}$ is smaller than its deadline.

A. Linear Integer Programming Formulation

We divide the worst-case delay for a generic message $D \in \mathcal{D}$ in two components. In the FC where D is generated, it experiences the maximum delay if it arrives right after the start of the MS given by $a_{\mathrm{DS}}(D)$. Hence this *initial delay* is

$$t_{D,\mathrm{init}} = (N_{\mathrm{MS}} - a_{\mathrm{DS}}(D) + 1)T_{\mathrm{MS}} + T_{\mathrm{SW}} + T_{\mathrm{NIT}}. \quad (3)$$

Next, we formulate a linear integer programming problem (LIP) that tries to fill the DS of the following f FCs with messages that have a smaller FID than D. If all such FCs can be filled, then the worst-case response time of D is larger than $t_{D,\mathrm{init}} + f \cdot T_{\mathrm{c}}$ and the same analysis has to be carried out for $f+1$ FCs. Otherwise, the worst-case response time of D can be computed from the longest delay D experiences within f FCs. Concretely, we introduce the boolean variables $x_{D',1}, \ldots, x_{D',f}$ for each message $D' \in \mathcal{D}$ with $a_{\mathrm{DS}}(D') < a_{\mathrm{DS}}(D)$, where $x_{D',i} = 1$ means that the message D' is transmitted in FC i and $x_{D',i} = 0$ otherwise for $1 \le i \le f$. In this respect, it has to be taken into account that the number of instances of D' that can occur within j consecutive FCs is limited to $\lceil j \cdot T_{\mathrm{c}}/p_{D'} \rceil$. Hence, for each j with $1 \le j \le f$, we introduce the following $f - j + 1$ constraints for each message $D' \in \mathcal{D}$.

$$\begin{array}{ccc} x_{D',1} + \cdots + x_{D',j} & \le & \lceil j \cdot T_{\mathrm{c}}/p_{D'} \rceil \\ \vdots & \vdots & \vdots \\ x_{D',f-j+1} + \cdots + x_{D',f} & \le & \lceil j \cdot T_{\mathrm{c}}/p_{D'} \rceil. \end{array} \quad (4)$$

Furthermore, in order to ensure that D can earliest be scheduled in the FC f, it must hold that D does not fit in the previous $f-1$ FCs. This requirement is captured by the following constraint for each i with $1 \le i < f$.

$$(N_{\mathrm{latest}} - 1) \cdot T_{\mathrm{MS}} < \sum_{\substack{D' \\ a_{\mathrm{DS}}(D') < a_{\mathrm{DS}}(D)}} x_{D',i} \cdot N_{D',\mathrm{MS}} \cdot T_{\mathrm{MS}} + (1 - x_{D',i}) \cdot T_{\mathrm{MS}} \le T_{\mathrm{DS}}. \quad (5)$$

Here, $N_{\mathrm{latest}} = N_{\mathrm{DS}} - \max_{D \in \mathcal{D}}(N_{D,\mathrm{MS}}) + 1$ as described in Section II-A. The term $x_{D',i} \cdot N_{D',\mathrm{MS}} \cdot T_{\mathrm{MS}}$ accounts for the messages with smaller FID that are transmitted in FC i, while the term $(1 - x_{D',i}) \cdot T_{\mathrm{MS}}$ captures the MSs that pass if a message with a smaller FID is not transmitted. In the last FC f it must hold that the sum of the durations of the messages transmitted in that FC has to be smaller than the DS duration.

$$\sum_{\substack{D' \\ a_{\mathrm{DS}}(D') < a_{\mathrm{DS}}(D)}} x_{D',f} \cdot N_{D',\mathrm{MS}} \cdot T_{\mathrm{MS}} + (1 - x_{D',f}) \cdot T_{\mathrm{MS}} \le T_{\mathrm{DS}}. \quad (6)$$

Finally, we formulate an objective function that describes the transmission time consumed by messages with a lower FID than D in the last (f-th) FC.

$$J_{D,f} = \sum_{\substack{D' \\ a_{\mathrm{DS}}(D') < a_{\mathrm{DS}}(D)}} x_{D',f} \cdot N_{D',\mathrm{MS}} \cdot T_{\mathrm{MS}} + (1 - x_{D',f}) \cdot T_{\mathrm{MS}}. \quad (7)$$

Our goal is to maximize $J_{D,f}$ in order to find the worst-case delay for D. Hence, we solve the optimization problem

$$J^*_{D,f} = \max_X J_{D,f} \quad (8)$$

subject to the constraints in (4), (5) and (6), and with the optimization vector X that contains all entries $x_{D',i}$ with $1 \le i \le f$ and $a_{\mathrm{DS}}(D') < a_{\mathrm{DS}}(D)$. If $J^*_{D,f} \le (N_{\mathrm{latest}} - 1) \cdot T_{\mathrm{MS}}$, then D can be transmitted in the f-th FC with a worst-case response time of

$$w_D = t_{D,\mathrm{init}} + (f-1) \cdot T_{\mathrm{c}} + T_{\mathrm{SS}} + J^*_{D,f} + N_{D,\mathrm{MS}} \cdot T_{\mathrm{MS}}. \quad (9)$$

Otherwise, D cannot be transmitted within f FCs, and the next larger number of $f+1$ FCs has to be tried. Together, we perform the following algorithm for schedulability analysis in the DS. It returns **true** if schedulability can be confirmed for all messages in $\mathcal{D}$ and **false** otherwise.

Algorithm 1: Input: message set $\mathcal{D}$, number of minislots in the DS N_{DS}, MS assignment a_{DS} and MS duration T_{MS}

```
for each message D ∈ 𝒟                                          1
    schedulable = false                                        2
    for each f ≤ ⌈d_D/T_c⌉                                      3
        if J*_{D,f} ≤ N_latest · T_MS and w_D ≤ d_D             4
            schedulable = true                                 5
            break                                              6
    if schedulable = false                                     7
        return false                                           8
return true                                                    9
```

Remark 1: Note that it is most favorable to choose the messages for the schedulability analysis in line 1 in the order of the message priority.

The schedulability analysis for the FlexRay DS based on a linear integer program is also studied in [10]. However, that work considers a fixed worst-case time interval for the analysis that has to be chosen properly. In contrast, our algorithm iteratively increases the time interval under investigation by one FC until a deadline violation occurs (line 8) or schedulability can be verified (line 6). Hence, [10] generally requires a larger number of variables leading to a more complex LIP. Moreover, the approach in [10] employs a weaker constraint than in (4) and does not enforce that *consecutive* FCs have to be filled with higher-priority message occurrences in order to find the worst-case response time for a given message. Hence, the analysis results in [10] can be conservative.

B. Schedulability Analysis Example

Message D	D_1	D_2	D_3	D_4	D_5
p_D/ms	10	10	20	20	25
d_D/ms	5	10	15	15	18
b_D	10	7	5	7	2
$N_{\mathrm{D,MS}}$	8	7	6	7	5

TABLE II
MESSAGE SET FOR THE SCHEDULABILITY ANALYSIS IN THE DS.

We perform the proposed schedulability analysis for the message set in Table II with a priority assignment according to the message index. Here, we first assume that $T_{\mathrm{SS}} = 3.01\,\mathrm{ms}$, $N_{\mathrm{MS}} = 18$, $T_{\mathrm{MS}} = 5\,\mu\mathrm{s}$, $T_{\mathrm{SW}} = 100\,\mu\mathrm{s}$ and $T_{\mathrm{NIT}} = 800\,\mu\mathrm{s}$

such that $T_\mathrm{c} = T_\mathrm{SS} + N_\mathrm{MS} \cdot T_\mathrm{MS} + T_\mathrm{SW} + T_\mathrm{NIT} = 4\,\mathrm{ms}$. In this example, N_latest evaluates to $N_\mathrm{latest} = N_\mathrm{MS} - 8 + 1 = 11$. We now consider the LIP for the message D_5 and the case of $f = \lceil d_{D_5}/T_\mathrm{c} \rceil = 5$ FCs. There are 5 variables $x_{D_i,1}, \ldots, x_{D_i,5}$ for each higher-priority message $D_i \in \{D_1, D_2, D_3, D_4\}$. Then, the constraint in (4) for each message $D_i \in \{D_1, D_2, D_3, D_4\}$ evaluates to

$$\lceil \tfrac{2T_\mathrm{c}}{p_{D_i}} \rceil \geq x_{D_i,k} + x_{D_i,k+1} \text{ for } k = 1,2,3,4$$
$$\lceil \tfrac{3T_\mathrm{c}}{p_{D_i}} \rceil \geq x_{D_i,k} + x_{D_i,k+1} + x_{D_i,k+2} \text{ for } k = 1,2,3$$
$$\lceil \tfrac{4T_\mathrm{c}}{p_{D_i}} \rceil \geq x_{D_i,k} + x_{D_i,k+1} + x_{D_i,k+2} + x_{D_i,k+3} \text{ for } k = 1,2$$
$$\lceil \tfrac{5T_\mathrm{c}}{p_{D_i}} \rceil \geq x_{D_i,1} + x_{D_i,2} + x_{D_i,3} + x_{D_i,4} + x_{D_i,5}$$

Furthermore in the FCs $j \in \{1,2,3,4\}$, the constraint in (5) applies such that

$$\begin{aligned}(N_\mathrm{latest} - 1) \cdot T_\mathrm{MS} &= \\ &= 50\,\mu\mathrm{s} < \big(x_{D_1,j} \cdot 7 + x_{D_2,j} \cdot 6 + x_{D_3,j} \cdot 5 + \\ &\qquad x_{D_4,j} \cdot 6 + 4\big) \cdot 5\,\mu\mathrm{s} \leq 90\,\mu\mathrm{s}.\end{aligned}$$

The constraint in (6) for $f = 5$ is

$$\begin{aligned}&\big(x_{D_1,5} \cdot (8-1) + x_{D_2,5} \cdot (7-1) + \\ &x_{D_3,5} \cdot (6-1) + x_{D_4,5} \cdot (7-1) + 4\big) \cdot 5\,\mu\mathrm{s} \leq 60\,\mu\mathrm{s}.\end{aligned}$$

Finally, the objective function is

$$\begin{aligned}J_{D_5,5} = &\big(x_{D_1,5} \cdot (8-1) + x_{D_2,5} \cdot (7-1) + \\ &x_{D_3,5} \cdot (6-1) + x_{D_4,5} \cdot (7-1) + 4\big) \cdot 5\,\mu\mathrm{s}.\end{aligned}$$

Maximizing the objective function w.r.t. the given constraints leads to $x_{D_1,1} = x_{D_1,3} = x_{D_2,2} = x_{D_2,4} = x_{D_3,2} = x_{D_4,4} = 1$, while all other variables are zero. This situation can also be seen in Fig. 4 (a) that illustrates the corresponding message occurrences. It is readily observed that, in the worst-case, the occurrence of D_5 is blocked by the higher-priority messages for at least 4 FCs, leading to the deadline violation of D_5. Note that MSs that pass without a message being sent are shaded in gray. In contrast, if the parameters $N_\mathrm{MS} = 20$ and $T_\mathrm{SS} = 2.95\,\mathrm{ms}$ are chosen (the DS is extended by shortening the SS), all messages are schedulable. For example, the worst-case response time for D_5 is illustrated by the situation in Fig. 4 (b). It holds that $w_{D_5} = T_{\mathrm{init},D_5} + 3 \cdot T_\mathrm{c} + T_\mathrm{SS} + J^*_{D_5,5} + N_{D_5,\mathrm{MS}} \cdot T_\mathrm{MS} = 0.98\,\mathrm{ms} + 12\,\mathrm{ms} + 3.0\,\mathrm{ms} + 20\,\mu\mathrm{s} + 5 \cdot 5\mu\mathrm{s} = 16.025\,\mathrm{ms} < 18\,\mathrm{ms} = d_{D_5}$.

IV. Schedule Computation for the Dynamic Segment

A. Scheduling Algorithm

In this section, we investigate the problem of algorithmically finding a schedulable MS assignment a_DS for the sporadic messages to be transmitted in the DS. In principle, there are two issues to be resolved. The number N_MS of MSs in the DS has to be chosen appropriately and the MS assignment a_DS has to be fixed such that all sporadic messages meet their deadline. We first note that this scheduling problem is computationally hard:

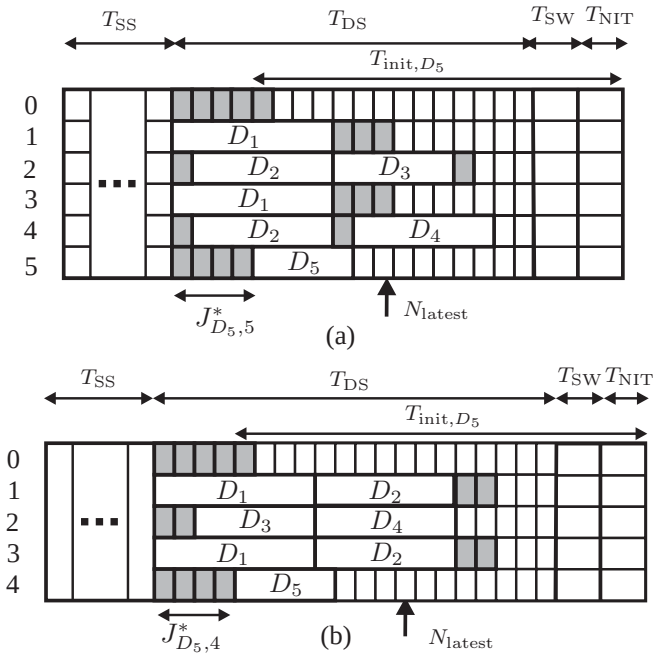

Fig. 4. Schedulability analysis for the DS.

1) assuming that there are N_MS MSs in the DS, there are $\binom{N_\mathrm{MS}}{|\mathcal{D}|} \cdot |\mathcal{D}|!$ possible MS assignments for the $|\mathcal{D}|$ sporadic messages. Even if it is enforced that only the first $|\mathcal{D}|$ MSs are assigned as dynamic slots, the number of possible MS assignments $|\mathcal{D}|!$ is factorial in $|\mathcal{D}|$.
2) the above number of possible MS assignments has to be evaluated for every potential number N_MS of MSs in the DS.

That is, all possible MS assignments a_DS and numbers of MSs N_MS in the DS have to be analyzed until a schedulable assignment has been found or all assignments are unschedulable. In the following, we propose the heuristic Algorithm 2 that tries to find a schedulable assignment without enumerating the overall search space. To this end, we suggest to iteratively perform the schedulability analysis introduced in the previous section for a fixed value N_MS. Each iteration (line 4) corresponds to an FID that is assigned to the unscheduled message in $\mathcal{D}'$ that has the smallest difference between its deadline and worst-case response time $d_D - w_D$ (line 15). The algorithm terminates successfully if all messages are schedulable (line 18). If the schedulability fails (line 12), we increment N_MS and restart the schedule computation (line 13). Although the heuristic approach can bring about a conservative result (i.e., the DS might be longer than necessary or a schedulable assignment might not be found), the algorithm provides practical results in many cases and terminates with at most $N_{\mathrm{MS},\max} \cdot |\mathcal{D}| \cdot (|\mathcal{D}| - 1)/2$ schedulability verifications, where $N_{\mathrm{MS},\max}$ denotes the maximum allowable number of MSs in the DS. The minimum length of the DS is determined by the longest sporadic message as $\max_{D \in \mathcal{D}} N_{D,\mathrm{MS}}$.

Algorithm 2: Input: message set $\mathcal{D}$, $N_{\mathrm{MS}} = N_{\mathrm{MS,max}}$, $N_{\mathrm{MS}} = \max_{D\in\mathcal{D}} N_{D,\mathrm{MS}}$, a_{DS} is empty.

```
while(N_MS ≤ N_MS,max)                                              1
  i = x, D' = D                                                     2
  while D' ≠ ∅                                                      3
    i = i + 1                                                       4
    schedulable = true                                              5
    for each message D ∈ D'                                         6
      set a_DS(D) = i                                               7
      if D is unschedulable according to Algorithm 1                8
        schedulable = false                                         9
        break                                                      10
      else                                                         11
        record d_D − w_D                                           12
    if schedulable = false                                         13
      N_MS = N_MS + 1                                              14
      break                                                        15
    set a_DS(D) = i for D ∈ D with minimal d_D − w_D                16
    D' = D' − {D}                                                  17
    if D' = ∅                                                      18
      return a_DS                                                  19
return false                                                       20
```

Note that a schedule computation is also carried out in [10]. However, that work does not incrementally assign FIDs to the sporadic messages but tries to alter an initial message schedule in order to achieve schedulability based on a delay estimate.

B. Schedule Computation Example

The scheduling algorithm is applied to the message set in Table II, whereby the parameters $T_{\mathrm{SW}} = 100\,\mu$s, $T_{\mathrm{NIT}} = 800\,\mu$s, $T_{\mathrm{MS}} = 5\,\mu$s and $T_{\mathrm{c}} = 4$ ms are kept constant. Hence, a possible enlargement of the DS is compensated by a reduction of the SS. Starting with $N_{\mathrm{MS}} = \max_{D\in\mathcal{D}} N_{D,\mathrm{MS}} = 8$, the algorithm determines that the message set $\mathcal{D}$ is schedulable for $N_{\mathrm{MS}} = 19$ and the priority assignment in Table III.

Message M	D_1	D_2	D_3	D_4	D_5
$a_{\mathrm{DS}}(M)$	1	2	4	3	5

TABLE III
PRIORITIES ASSIGNED BY THE SCHEDULING ALGORITHM.

The corresponding duration of the SS is $T_{\mathrm{SS}} = 3.005$ ms. Furthermore, the priority assignment of D_3 and D_4 with identical period and deadline values illustrates the features of Algorithm 2: the longer message D_4 obtains the higher-priority FID 3, due to its smaller margin to the deadline.

C. Simulation Study

We finally validate the results in Section III-B and IV-B by a hardware experiment. The 5 sporadic messages are distributed to 3 FlexRay nodes that are realized by the evaluation boards SK-91465X-100MPC [4] and the message delays are recorded by the bus analyzer Flexcard Cyclone II SE [3]. Table IV shows our measurement results that are obtained from ten 2 min sample runs, where the sporadic messages are generated according to Table II and the priority assignments and configuration parameters are given as in Section III-B and IV-B for the respective value of N_{MS}.

	$N_{\mathrm{MS}} = 18$	$N_{\mathrm{MS}} = 19$	$N_{\mathrm{MS}} = 20$
w_{D_1}/ms	4.034 (4.04)	4.034 (4.04)	4.034 (4.04)
w_{D_2}/ms	4.067 (4.07)	4.067 (4.07)	4.067 (4.07)
w_{D_3}/ms	8.022 (8.035)	8.053 (8.065)	8.02 (8.035)
w_{D_4}/ms	8.059 (8.065)	8.034 (8.035)	8.061 (8.065)
w_{D_5}/ms	19.804 (−)	15.943 (16.025)	14.819 (16.025)

TABLE IV
EXPERIMENTAL RESULTS FOR THE FLEXRAY DS.

Our results show that the worst-case delays determined by the schedulability analysis (in parenthesis) can indeed be verified by the experiment. In particular, if $N_{\mathrm{MS}} = 18$ MSs is chosen, the potential deadline violation of D_5 is confirmed by the experiment.

V. CONCLUSION

In this paper, we studied the message scheduling for sporadic real-time messages in the dynamic segment of FlexRay. Considering the analysis of a given message schedule, we formulated a linear integer programming problem whose solution yields the worst-case message delays. Hence, it is possible to decide if all message deadlines are met. In addition, we developed an algorithm that efficiently computes feasible schedules based on our schedulability analysis. The applicability of our method was illustrated by the schedule computation for an example message set and validated by message delay measurements on a FlexRay network with 3 nodes.

ACKNOWLEDGMENT

The authors would like to thank TOFAŞ Türk Otomobil Fabrikaları A.Ş. for supporting our hardware experiments.

REFERENCES

[1] (2003, Nov.) Time-triggered protocol TTP/C, high-level specification document, protocol version 1.1. [Online]. Available: http://www.tttech.com

[2] (2004, June) FlexRay communication system, protocol specification, version 2.0. [Online]. Available: http://www.flexray.com

[3] (2009) Flexcard Cyclone II SE. [Online]. Available: http://www.eberspaecher.com/servlet/PB/menu/1053178_l2/index.html

[4] (2009) Fujitsu FlexRay Evaluation Board: SK-91465X-100PMC. [Online]. Available: http://mcu.emea.fujitsu.com/mcu_tool/detail/SK-91465X-100PMC.htm

[5] J. Berwanger, M. Peller, and R. Griegbach. (1999) A new high-performance data bus system for safety related applications. [Online]. Available: http://www.byteflight.com/specification

[6] H. Kopetz, "A solution to an automotive control system benchmark," in *IEEE Real-time systems symposium*, 1994.

[7] H. Kopetz and G. Bauer, "The time-triggered architecture," *Proc. IEEE*, vol. 91, no. 1, pp. 112–126, 2003.

[8] R. Makowitz and C. Temple, "FlexRay - a communication network for automotive control systems," *Factory Communication Systems, IEEE International Workshop on*, pp. 207–212, June 27, 2006.

[9] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert, "Trends in automotive communication systems," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1204–1223, June 2005.

[10] T. Pop, P. Pop, P. Eles, Z. Peng, and A. Andrei, "Timing analysis of the FlexRay communication protocol," *Real-Time Syst.*, vol. 39, no. 1-3, pp. 205–235, 2008.

[11] E. G. Schmidt and K. Schmidt, "Message scheduling for the FlexRay protocol: The dynamic segment," *Vehicular Technology, IEEE Transactions on*, vol. 58, no. 5, pp. 2170–2179, 2009.