Optimal Message Scheduling for the Static 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 study the scheduling of periodic messages in the static segment of the FlexRay protocol. Our approach is based on two *performance metrics*. Similar to previous work, we seek to allocate a minimum number of frame IDs (FIDs) in the static segment. In addition, different from existing work, we want to minimize the message *jitter*, i.e., the deviation of the message transmission from the required periodicity. To this end, we first derive analytical expressions that quantify the *FID allocation* and the *jitter*, and then formulate a linear integer programming problem whose solution is the desired message schedule. An example illustrates our schedule optimization.**

I. Introduction

Modern automobiles comprise a multitude of microcontroller-based electronic devices that support the driving safety and comfort. These *electronic control units* (ECU) generally perform communication in order to support the execution of their tasks. In particular, recent applications such as x-by-wire necessitate the timely and reliable periodic and sporadic information exchange among ECUs via an appropriate automotive communication network.

The FlexRay protocol was developed recently to address these new requirements for in-vehicle communication [3], [9], [8]. In comparison to existing protocols such as CAN (Controller Area Network) [13], FlexRay offers two channels with a high bandwidth of 10 Mbit/s and its cyclic operation supports both time-triggered and event-triggered communication. In each FlexRay cycle, the static segment (SS) employs the idea of time division multiple access (TDMA) to enable the transmission of periodic messages in unique *static slots*, while sporadic messages can be sent in *dynamic slots* in the dynamic segment (DS). In this context, the proper utilization of FlexRay depends on the configuration of the message transmission by an appropriate *message schedule*.

In this paper, we study the message scheduling problem for the FlexRay static segment. It requires to assign a *frame ID* (FID), a *scheduling repetition* and a *scheduling offset* to each periodic *message* so as to configure its transmitting static slots [11]. The FIDs are limited and they are exclusively allocated to the nodes. Hence, for a given message set, decreasing the number of allocated FIDs per node is required for the extendability of the network. An important fraction of the messages in automotive networks are generated periodically, however it is possible that the transmission of these messages deviate from this periodicity with some *jitter* because of the message schedule. The control applications generally rely on the periodic transmission of these messages. Therefore, it is beneficial to achieve message schedules with minimum jitter.

According to these requirements, we define the *FID allocation* and the *jitter* as performance metrics to be optimized in the message schedule computation. We determine an analytic description of both performance metrics in terms of the scheduling parameters (FID, repetition, offset), which allows us to formulate a linear integer programming problem for the joint optimization of the FID allocation and the jitter. We illustrate our results by means of a practical message set.

[5] and [7] also provide scheduling algorithms for the static segment of the FlexRay protocol. [7] addresses the packing of signals in frames in order to minimize the FID allocation, while [5] proposes heuristics that minimize the number of slots needed to meet all signal deadlines. However, in none of these approaches, the repetition serves as a free parameter to capture the trade-off between the FID allocation and the jitter as important performance metrics for in-vehicle communication. Although our previous work in [11] addresses this trade-off, the message schedule computation is carried out assuming an additional software architecture on top of FlexRay.

The organization of the paper is as follows. After providing a description of the FlexRay protocol in Section II, we study message scheduling for the static segment in Section III. This includes the definition of our performance metrics and the formulation of an appropriate linear integer programming problem. Conclusions are given in Section IV.

II. The FlexRay Protocol

A. General Overview

The FlexRay operation is determined by a *FlexRay cycle* (FC) with a fixed duration that is repeatedly executed. It consists of the *static segment* (SS), the *dynamic segment* (DS), the *symbol window* (SW), and the *network idle time* (NIT) as shown in Fig. 1. 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. The fixed duration of each FC can be expressed as $\texttt{gdCycle} = \texttt{gMacroPerCycle} \cdot \texttt{gdMacrotick}$, where `gMacroPerCycle` denotes the number of MTs per FC (between 10 and 16000). The successive FCs are counted by the protocol internal variable `vCycleCounter` that ranges

from 0 to 63 and is equal among all nodes on a FlexRay network.

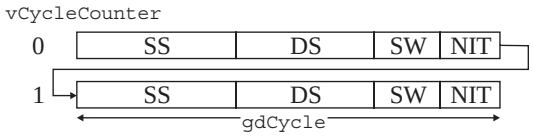

Fig. 1. FlexRay cycle description.

The SS is designed for the periodic transmission of real-time data. In each FC, the SS provides `gNumberOfStaticSlots` *static slots* as illustrated in Fig. 2, where all static slots have the same fixed duration of `gdStaticSlot` (between 4 and 661 MT). The number of the current static slot is captured by the internal variable `vSlotCounter` that is reset at the beginning of each FC and incremented in each static slot. Together, the duration of the SS can be expressed as `gNumberOfStaticSlots` · `gdStaticSloc`·`gdMacrotick`. The bus arbitration in the SS is similar to the time-triggered protocol (TTP) [2], [6] and employs the TDMA approach. To this end, frame identifiers (FIDs) are uniquely assigned to nodes such that in each static slot, the node with the FID that is equal to the current value of `vSlotCounter` can send a message.

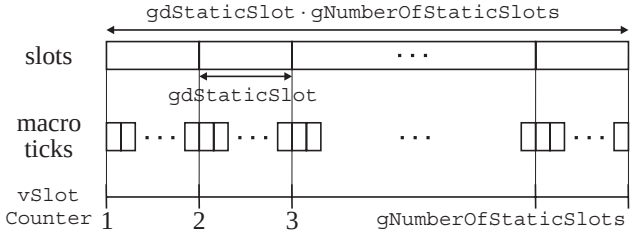

Fig. 2. FlexRay static segment (SS).

In addition, FlexRay allows *slot multiplexing* in the sense that the same FID can be used for different messages in different FCs. In this case, messages are not scheduled in every FC but only in pre-determined FCs that are identified based on the value of the `vCycleCounter` as specified in [3]. Noting that `vCycleCounter` periodically counts from 0 to 63, only assignments that repeat every $1, 2, 4, \ldots, 64$ FCs can be realized. Hence, we describe cycle multiplexing using this *repetition* of the respective FC assignment and the *offset* which denotes the first FC among the possible 64 FCs where the respective assignment starts. In any case, it must be true that the offset is smaller than the repetition. For example, the message A in Fig. 3 is scheduled with a repetition of 1 and B, C have a repetition of 2. The offset of A and B is 0, while C has an offset of 1.

The DS is designed for the transmission of sporadic messages. Its operation is similar to ByteFlight [4] and employs the flexible TDMA (FTDMA) approach. Finally, the SW and the NIT provide time for the transmission of internal control information and for protocol-related computations. Since we focus on the SS in this paper, we only assume that the durations of the respective components of the FC add up to the FC duration `gdCycle`.

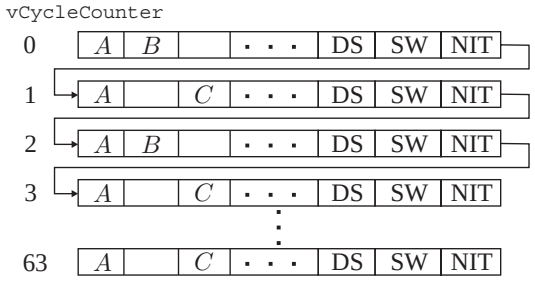

Fig. 3. Cycle multiplexing in the FlexRay SS.

Note that the FlexRay standard incorporates accurate synchronization of the nodes, such that each node can access the medium at unique pre-defined time instants. In particular, no prioritization scheme as for example in CAN is needed. Moreover, since the durations of the SS, DS, SW and NIT are fixed during system operation, there is no interference among the transmitted messages in different segments.

III. Schedule Computation for the Static Segment

A. Scheduling Problem

We consider the situation where n nodes communicate on a FlexRay network. Each node i transmits a set of periodic messages $\mathcal{M}^i$, where each message $M \in \mathcal{M}^i$ is characterized by its period p_M that is measured in multiples of the FC duration `gdCycle` as in [11], [7]. As is common for periodic messages, we assume that the message *deadline*, i.e., the maximum time that is allowed between message generation and reception is equal to the period. Furthermore, we assume that `gdStaticSlot` is chosen such that each message fits into the static slot size.

Then, the scheduling goal is to find an appropriate assignment of the free configuration parameters of each message. That is, for each message M, an FID f_M, a repetition r_M and an offset o_M has to be chosen. Following the description in the previous section, $r_M \in \{2^q | 0 \leq q \leq 6\}$, $0 \leq o_M < r_M$ and $0 \leq f_M \leq$ `gNumberOfStaticSlots`. Moreover, it is required that the repetition is not larger than the message period, i.e., $r_m \leq p_M$, such that the message can meet its deadline. Furthermore, the protocol specification [3] demands that each message is associated to a unique triple (f_M, r_M, o_M) and that each FID is assigned to a unique node. Note that the latter requirement implies that the configuration parameters for messages from different nodes are independent of each other. Hence, the message scheduling problem on FlexRay can be separated into n scheduling problems for the different nodes.

In addition to the scheduling restrictions that are defined by the protocol specification, our work is based on two *performance metrics* that describe practical requirements. On the one hand, it is desired to allocate a minimum number of FIDs per node in order to keep the used fraction of the SS small. This measure supports the extensibility of the FlexRay

configuration for new applications. We define the *FID allocation* FA^i that describes the number of FIDs that is allocated per node i to capture this metric. On the other hand, it has to be considered that, depending on the message properties, not all messages can be scheduled with their assigned period. In that case, the resulting deviation from the periodicity, i.e., the *jitter* J_M for each message M is used as a performance metric that has to be minimized. In the following, we propose a joint minimization of the FID allocation and the jitter while meeting the constraints of the protocol specification.

Note that, in practice, ECUs generate *signal data* that are packed into messages for transmission on FlexRay. In this paper, we assume that this signal packing has already been performed by an appropriate method such as [10], [11].

B. FID Allocation

We consider the messages $\mathcal{M}^i$ of a node i. For a single message $M \in \mathcal{M}^i$ that is scheduled with a repetition r_M, the fraction of the slots of one FID that is used by M is

$$A_M = \frac{1}{r_M}. \tag{1}$$

Based on this computation, the required number of allocated FIDs for node i can be directly evaluated as stated in the following proposition.[1]

Proposition 1 (FID Allocation): Consider the node i with its message set $\mathcal{M}^i$, and assume that r_M is given for each $M \in \mathcal{M}^i$. Then, the minimum FID allocation for the selected r_M for the messages of node i is given by

$$FA^i := \lceil \sum_{M \in \mathcal{M}^i} A_M \rceil = \lceil \sum_{M \in \mathcal{M}^i} \frac{1}{r_M} \rceil. \tag{2}$$

Proof: We prove the statement by algorithmically constructing a possible configuration for each $M \in \mathcal{M}^i$ such that only FA^i FIDs are allocated. Let L^i be a list of the messages in $\mathcal{M}^i$ that is ordered by ascending repetitions and f_{init} the initial FID. We further introduce the set of available offsets $\mathcal{O}$, the current FID f_{c} and the utilization of the current FID $util$ as variables. Then, we invoke the following algorithm.

Algorithm 1: Input: L_i, f_{init}; Variable: $\mathcal{O} = \{0, \ldots, 63\}$, $f_{\mathrm{c}} := f_{\mathrm{init}}$, $util := 0$,

while L_i is not empty	1
if $util = 1$	2
$util := 0$; $\mathcal{O} := \{0, \ldots, 63\}$, $f_{\mathrm{c}} := f_{\mathrm{c}} + 1$	3
remove the first element M from L_i	4
$util := util + 1/r_M$	5
assign $f_M := f_{\mathrm{c}}$	6
assign smallest element o in $\mathcal{O}$ to o_M: $o_M := o$	7
remove all elements $o + k \cdot r_M$ from $\mathcal{O}$ for $k \in \mathbb{N}_0$	8
return triple (r_M, o_M, f_M) for each $M \in \mathcal{M}^i$	9

The algorithm loops until all messages in L^i are configured. In each loop, a new FID is allocated if the current FID is fully utilized (line 1). In any case, the message with the currently smallest repetition is added to the current FID (line 4, 5 and 6). This message always fits due to the order of L^i and the restriction that r_M is a power of 2. Then, the next free offset is assigned to M and all potential offsets that are invalidated by adding M to the FID are removed from $\mathcal{O}$ (line 7 and 8). Together, it is readily observed that all FIDs that are allocated to node i except for the last one are completely filled resulting in the FID allocation in (2). ∎

Applying Proposition 1 to all n nodes, the overall FID allocation FA is given by

$$FA := \sum_{1 \le i \le n} FA^i. \tag{3}$$

As a further result, the proof of Proposition 1 shows that the minimum FID allocation is determined only by the message repetitions. Appropriate values for the FID and offset of each message can be found by a straightforward application of Algorithm 1.

C. Message Jitter Computation

The message *jitter* for a periodic message $M \in \mathcal{M}^i$ is defined as the deviation of the inter-transmission times from the periodicity [11]. In order to quantify the jitter, we consider a generic message $M \in \mathcal{M}^i$ of a node i with the period p_M that is scheduled with the repetition $r_M \le p_M$. Then, it holds that the period of M can be written as

$$p_M = k \cdot r_M + b, \quad k \in \mathbb{N}, \quad b = p_M \bmod r_M, \quad 0 \le b < r_M. \tag{4}$$

Unless $b = 0$, it is the case that M cannot be sent with its generation period p_M, i.e., the inter-transmission times between the sending of two message instances of M varies. From the representation, in (4) it can be deduced that the possible inter-transmission times are either $k \cdot r_M$ or $(k+1) \cdot r_M$.

Lemma 1: Assume the relation in (4) holds for a message $M \in \mathcal{M}^i$ with $b > 0$. Then, the possible values of the inter-transmission time for M are $k \cdot r_M$ and $(k+1) \cdot r_M$. Moreover, if $b = 0$, the inter-transmission time is $k \cdot r_M = p_M$.

Proof: $b > 0$: Assume the contrary, i.e., the inter-transmission time assumes the value $l \cdot r_M$ with 1) $l < k$ or 2) $l > k + 1$. We consider a pair of message instances where the inter-transmission time $l \cdot r_M$ occurs and assume that the first message instance is scheduled at some time t, while the second message instance is sent at $t + l \cdot r_M$.

Case 1): The first instance is generated at $t_1 > t - r_M$ and the second instance is generated at $t_2 \le t + l \cdot r_M$. Then, the difference is $t_2 - t_1 < t + l \cdot r_M - t + r_M = (l+1) \cdot r_M \le k \cdot r_M \le p_M$ according to (4). This contradicts the fact that M is generated with period p_M.

Case 2): The first instance is generated at time $t_1 \le t$ and the second instance is generated at $t_2 > t + (k+1) \cdot r_M$. Then, the difference is $t_2 - t_1 > t + (k+1) \cdot r_M - t = (k+1) \cdot r_M > p_M$. This contradicts the fact that M is generated with period p_M.

$b = 0$: This statement directly follows from (4). ∎

Respecting Lemma 1, it is clear that the possible jitter values for M are $|k \cdot r_M - p_M| = |k \cdot r_M - k \cdot r_M - b| = b$ or

[1] An analogous result is stated for time-triggered CAN (TTCAN) in [12].

$|(k+1)\cdot r_M - k\cdot r_M - b| = r_M - b$. Using this information, we now determine the *relative jitter per FC* J_M in order to quantify the jitter experienced by each message.

Theorem 1: Assume a message $M \in \mathcal{M}^i$ of node i with the period p_M that is scheduled with a repetition r_M. Then, using the representation in (4), the relative jitter per FC is

$$J_M := \frac{2\cdot(r_M - b)\cdot b}{p_M \cdot r_M}. \tag{5}$$

Proof: First, we consider the case $b = 0$. Then, M can be scheduled with its exact period and hence $J_M = 0$, which fulfills (5).

Now let $b > 0$, i.e., the possible inter-transmission times are $k \cdot r_M$ and $(k+1)\cdot r_M$ according to Lemma 1. We evaluate the accumulated jitter within $lcm(r_M, p_M)$ FCs, i.e., until the same situation repeats. It holds that the inter-transmission times within $lcm(r_M, p_M)$ FCs must add up to $lcm(r_M, p_M)$:

$$z_1 \cdot k \cdot r_M + z_2 \cdot (k+1)\cdot r_M = lcm(r_M, p_M). \tag{6}$$

Here, z_1 and z_2 denote the number of occurrences of the respective inter-transmission times. Furthermore, it must hold that exactly $lcm(r_M, p_M)/p_M$ message transmissions happen in $lcm(r_M, p_M)$ FlexRay cycles. Hence, it follows that

$$z_1 + z_2 = lcm(r_M, p_M)/p_M. \tag{7}$$

Together, (6) and (7) constitute two linear equations for the two unknowns z_1 and z_2 which can be solved as follows.

$$\begin{bmatrix} k & k+1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} lcm(r_M, p_M)/r_M \\ lcm(r_M, p_M)/p_M \end{bmatrix}$$
$$\Rightarrow \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \frac{lcm(r_M, p_M)}{p_M \cdot r_M} \cdot \begin{bmatrix} r_M - b \\ b \end{bmatrix}$$

Hence, the relative jitter per FC evaluates to

$$J_M = \frac{z_1 \cdot b + z_2 \cdot (r_M - b)}{lcm(p_M, r_M)} = \frac{2\cdot(r_M - b)\cdot b}{p_M \cdot r_M}.$$

■

Example 1: We choose a message M with the period $p_M = 5$ that is scheduled with a repetition of $r_M = 2^2 = 4$ and an offset of $o_M = 0$. Fig. 4 depicts the situation where M is first generated and transmitted in the initial FC. Here, the arrows on top of the figure indicate message generations, the arrows at the bottom of the figure represent message transmissions and the shaded boxes show the periodically recurring FCs that provide a static slot for the transmission of M. Then, the subsequent occurrences of M happen with inter-transmission times of $k\cdot r_M = 1\cdot 4$ or $(k+1)\cdot r_M = 8$ according to Lemma 1. In this case, the inter-transmission time 8 occurs once, while the inter-transmission time 4 occurs 3 times, leading to a jitter of $(8-5) + 3\cdot(5-4) = 6$ within $\mathrm{lcm}(4,5) = 20$ FCs. Hence, the relative jitter per FC is $J_M = 6/20 = 0.3$, which equals the computation according to Theorem 1 with $J_M = 2\cdot(4-1)\cdot 1/(4\cdot 5) = 0.3$.

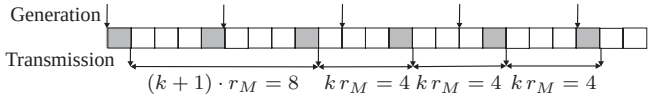

Fig. 4. Illustration of the jitter computation.

D. Optimal Message Schedule

The discussion in the previous sections suggests that both the FID allocation FA^i and the relative jitter $J^i := \sum_{M\in\mathcal{M}^i} J_M$ of each node i depend on the choice of the repetition r_M for each message $M \in \mathcal{M}^i$. In this section, we formulate a linear integer programming problem (LIP) that assigns an appropriate value for r_M while achieving the joint minimization of FA^i and J^i. To this end, we introduce the set $\mathcal{R}_M$ of potential repetitions for M as $\mathcal{R}_M = \{2^q | 2^q \le p_M\}$, i.e., the repetitions that are not larger than the message period. For each $r \in \mathcal{R}_M$, we introduce a boolean variable $x_{M,r}$ that assumes the value 1 if $r_M = r$ is chosen and $x_{M,r} = 0$ otherwise. Since only one value of the repetition period can be chosen for each message M, the following constraint applies.

$$\sum_{r\in\mathcal{R}_M} x_{M,r} = 1. \tag{8}$$

Applying Theorem 1, the jitter J_M evaluates to

$$J_M = \sum_{r\in\mathcal{R}_M} x_{P,r} \cdot \frac{2\cdot(r - (p_M \bmod r))\cdot(p_M \bmod r))}{p_M \cdot r}. \tag{9}$$

Furthermore, the FID allocation for node i follows from (2).

$$FA^i = \lceil \sum_{M\in\mathcal{M}^i} \sum_{r\in\mathcal{R}_M} \frac{x_{M,r}}{r} \rceil = \lceil \big(64\cdot \sum_{M\in\mathcal{M}^i} \sum_{r\in\mathcal{R}_M} \frac{x_{M,r}}{r}\big)/64 \rceil.$$

In the latter expression, the evaluation of the ceiling operator is equivalent to adding the smallest integer $c \in \mathbb{N}_0$ to the numerator such that it becomes a multiple of 64. Hence, FA^i is captured by the integer constraint

$$FA^i \cdot 64 = 64 \cdot \big(\sum_{M\in\mathcal{M}^i} \sum_{r\in\mathcal{R}_M} \frac{x_{M,r}}{r}\big) + c. \tag{10}$$

Using the results in (10) and (9), we formulate our optimization problem that jointly minimizes the FID allocation and the jitter using the optimization vector X whose entries are the variables $x_{M,r}$ for all messages $M \in \mathcal{M}^i$ and repetition periods $r \in \mathcal{R}_M$ as well as the variable c.

$$\min_X \gamma_{FA} FA^i + \gamma_J \cdot \sum_{M\in\mathcal{M}^i} J_M. \tag{11}$$

subject to the constraint in (8). By choosing γ_{FA} and γ_J appropriately, more weight can be given to the FID allocation and the jitter minimization, respectively. The overall message set is schedulable (all messages meet their deadlines) if the sum of the FID allocations per node remains below the number of static slots in the SS, i.e., $\sum_{i=1}^{n} FA^i \le$ `gNumberOfStaticSlots`. The scheduling algorithm for the FlexRay SS has been implemented in the form of a C++ library. Here, the required optimization of the repetitions is carried out by the open source library GLPK for linear integer programming [1], and the assignment of FIDs and offsets for the messages is performed according to Algorithm 1.

TABLE I
FLEXRAY MESSAGE SET.

message	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}
node	2	2	2	2	1	2	1	1	2	2	2	2	2	2
period	2	1	4	2	2	2	2	2	2	2	2	4	2	4
Exp. 1	2	1	4	2	2	2	2	2	2	2	2	4	2	4
Exp. 2	2	1	4	2	2	2	2	2	2	2	2	4	2	4
message	M_{15}	M_{16}	M_{17}	M_{18}	M_{19}	M_{20}	M_{21}	M_{22}	M_{23}	M_{24}	M_{25}	M_{26}	M_{27}	M_{28}
node	2	1	1	1	1	1	2	2	2	3	3	1	2	1
period	2	2	2	2	20	10	20	20	20	50	100	50	2	20
Exp. 1	2	2	2	2	4	2	4	4	4	2	4	2	2	4
Exp. 2	2	2	2	2	16	8	16	16	16	32	64	32	2	16
message	M_{29}	M_{30}	M_{31}	M_{32}	M_{33}	M_{34}	M_{35}	M_{36}	M_{37}	M_{38}	M_{39}	M_{40}	M_{41}	
node	1	1	1	1	1	1	3	1	1	2	2	3	1	
period	20	20	400	400	200	200	4	400	400	400	400	400	20	
Exp. 1	4	4	16	16	8	8	4	16	16	16	16	16	4	
Exp. 2	16	16	64	64	64	64	4	64	64	64	64	64	16	

TABLE II
JITTER VALUES FOR SCHEDULING WITH MINIMUM FID ALLOCATION.

p_M	1	2	4	10	20	50	100	200	400
J_M	0	0	0	0.3	0.3	0.32	0.32	0.07	0.06

E. Scheduling Example

We now perform the described message schedule computation for different values of γ_{FA} and γ_J using the set of 41 periodic messages in Table I that is adopted from an automotive application. Note that, different from the message sets in [5], [7], this realistic message set contains messages whose periods are not powers of two of the FC. The table shows the message periods and assignment to one of three FlexRay nodes. We perform two different experiments. In the first experiment, we determine a message schedule such that no message experiences jitter. To this end, we use the optimization in (11) with the parameters $\gamma_{FA} = 0.1$ and $\gamma_J = 10$. As a result, we obtain a schedule that assigns the repetitions in Table I for Exp. 1. It can be observed from the table that each message is scheduled with the largest possible repetition period that is a divisor of the message period. Hence, according to the optimization goal, the smallest possible FID count without introducing jitter is achieved. It amounts to 6 FIDs assigned to node 1, 8 FIDs assigned to node 2 and 2 FIDs assigned to node 3, i.e., a SS with at least 16 static slots is required.

In contrast, a schedule computation that enforces a minimum number of allocated FIDs is achieved with the parameters $\gamma_{FA} = 10$ and $\gamma_J = 0.1$. Here, only 4 FIDs are assigned to node 1, 7 FIDs are assigned to node 2 and 1 FID is assigned to node 3 such that a static segment with 12 static slots is required. However, in this case, the messages with periods 10, 20, 50, 100, 200 and 400 experience jitter. The respective relative jitter values computed using (9) are listed in Table II.

IV. CONCLUSION

The main contribution of this paper is the algorithmic solution of the scheduling problem of periodic messages in the static segment of the FlexRay protocol. Our approach is based on the *FID allocation* and the message *jitter* as performance metrics. We show that these performance metrics can be analytically expressed in terms of the message *repetition* as a free scheduling parameter and the fixed message *period* for all messages to be scheduled. Using this expression, we formulate a linear integer programming problem that jointly optimizes the performance metrics. An example with a practical message set demonstrates the features of our optimization approach.

REFERENCES

[1] (2003) GNU linear programming kit. [Online]. Available: http://www.gnu.org/software/glpk/

[2] (2003, Nov.) Time-triggered protocol TTP/C, high-level specification document, protocol version 1.1. [Online]. Available: http://www.tttech.com

[3] (2004, June) FlexRay communication system, protocol specification, version 2.0. [Online]. Available: http://www.flexray.com

[4] 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

[5] M. Grenier, L. Havet, and N. Navet, "Configuring the communication on flexray: the case of the static segment," in *European Congress on Embedded Real Time Software*, 2008.

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

[7] M. Lukasiewycz, M. Glaß, J. Teich, and P. Milbredt, "Flexray schedule optimization of the static segment," in *CODES+ISSS '09: Proceedings of the 7th IEEE/ACM international conference on Hardware/software codesign and system synthesis*. New York, NY, USA: ACM, 2009, pp. 363–372.

[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] R. Saket and N. Navet, "Frame packing algorithms for automotive applications," *Journal of Embedded Computing*, vol. 2, pp. 93–102, 2006.

[11] K. Schmidt and E. G. Schmidt, "Message scheduling for the FlexRay protocol: The static segment," *Vehicular Technology, IEEE Transactions on*, vol. 58, no. 5, pp. 2160–2169, 2009.

[12] ——, "Systematic message schedule construction for time-triggered CAN," *Vehicular Technology, IEEE Transactions on*, vol. 56, no. 6, pp. 3431–3441, Nov. 2007.

[13] I. Standard-11898, "Road vehicles-interchange of digital information – Controller Area Network (CAN) for high-speed communication," *International Standards Organisation (ISO)*, 1993.