Message Scheduling for the FlexRay Protocol: The Static Segment

Klaus Schmidt, Ece Guran Schmidt, *Member, IEEE*

***Abstract*—In the recent years, *time-triggered* communication protocols have been developed to support time-critical applications in in-vehicle communication. In this respect, the FlexRay protocol is likely to become the de-facto standard. In this paper, we investigate the scheduling problem of periodic signals in the *static segment* of FlexRay. We identify and solve two subproblems, and introduce associated performance metrics: (i) the signals have to be packed into equal size messages to obey the restrictions of the FlexRay protocol, while using as little bandwidth as possible. To this end, we formulate a nonlinear integer programming (NIP) problem in order to maximize the bandwidth *utilization*. Furthermore, we employ the restrictions of the FlexRay protocol to decompose the NIP and compute the *optimal message set* efficiently; (ii) a message schedule has to be determined such that the periodic messages are transmitted with minimum *jitter*. For this purpose, we propose an appropriate software architecture, and derive an integer linear programming (ILP) problem that both minimizes the jitter and the *bandwidth allocation*. A case study based on a benchmark signal set illustrates our results.**

***Index Terms*—Vehicular communication networks, FlexRay, real-time, scheduling, integer programming**

I. Introduction

In today's cars, a great variety of electronic devices including micro controllers, sensors, and actuators, are used to replace mechanical and hydraulic components. These *electronic control units* (ECU) require information exchange among each other to support the execution of their tasks. In today's luxury cars up to 70 ECUs exchange up to 2500 signals [1], [2].

Different in-vehicle communication networks for automotive systems have been developed. Currently, the most widely used network is the Controller Area Network (CAN) [2], [3]. It can provide bounded delay communication among ECUs at data rates between 125 kb/s to 1 Mb/s [4]. However, due to its *event triggered* nature and its relatively low data rate, it is not well suited for novel applications such as x-by-wire, which require periodic data exchange with low jitter. Several *time-triggered* technologies such as Time-triggered CAN (TTCAN, [5], [6]), Time-triggered Protocol (TTP, [7], [8]), and FlexRay ([9], [10]) have been designed to provide predictable medium access at a higher available bandwidth.

Time-triggered in-vehicle communication networks transmit signal data encapsulated in *messages* whose transmission instants are given by a pre-computed *message schedule*. In this paper, *periodic* real-time messages, i.e., messages that contain periodically generated signal data, are considered. In this case, the message schedule must enable the message transmission with low jitter, i.e., a low deviation from the periodicity.

Our study focuses on the FlexRay protocol, as it is expected to be the new de-facto standard for in-vehicle communication [2]. FlexRay has a high bandwidth of 10 Mbit/s, and consists of a *static segment* with Time Division Multiple Access (TDMA) operation, and a *dynamic segment* with flexible TDMA (FTDMA) operation. Accordingly, it combines the advantages of time-triggered and event-triggered communication.

In this work, we investigate the message schedule computation for the static segment of FlexRay that is designed to accommodate periodic real-time messages. Previous work on this topic focuses on the timing analysis of applications on a FlexRay bus [11], [12] or on heuristic strategies that aim at finding a feasible message schedule for a given message set [13]. In contrast, we address the problem of constructing feasible and efficient message schedules with low jitter starting from the signal data to be transmitted. We introduce a formal problem description to capture the properties of the FlexRay protocol. Furthermore, we define the bandwidth *utilization,* the number of *allocated frame identifiers (FIDs)* and the *jitter* as *performance metrics* that measure the efficiency of each message schedule. Then, linear integer programming is employed to find the schedule that optimizes the defined performance metrics in two steps. First, we determine how signal data have to be packed into message frames while maximizing the utilization. Second, the obtained messages are scheduled with minimum jitter in the FlexRay static segment while using a minimum number of FIDs.

The paper is organized as follows; in Section II, the FlexRay protocol is described, and notation and performance metrics are introduced in Section III. An optimization problem for packing signals into messages is developed in Section IV, and the message scheduling problem is elaborated in Section V. Section VI provides a case study based on a benchmark message set [14], and conclusions are given in Section VII.

II. The FlexRay Protocol

The FlexRay protocol [2], [9] is a *time-triggered* protocol. Its operation is based on a repeatedly executed *FlexRay cycle* (FC) with a fixed duration. Messages are transmitted in *FlexRay frames* that consist of message data as multiples of 2 byte-words and a *framing overhead*. If the message data comprise b words, then the frame size f in bit evaluates to

$$f = b \cdot 16\,\mathrm{bit} + (b \cdot 4\,\mathrm{bit} + O_{\mathrm{F}}) = b \cdot 20\ \mathrm{bit} + O_{\mathrm{F}}, \qquad (1)$$

where the framing overhead according to [9] is $b \cdot 4\ \mathrm{bit} + O_{\mathrm{F}}$.

Klaus Schmidt is with the Chair of Automatic Control, University of Erlangen-Nuremberg, Germany, e-mail: klaus.schmidt@rt.eei.uni-erlangen.de.

Ece Guran Schmidt is with the Department of Electrical and Electronics Engineering, Middle East Technical University, Ankara, Turkey, e-mail: eguran@metu.edu.tr.

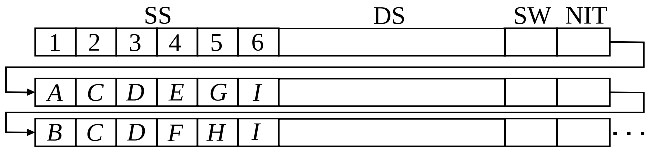

Fig. 1. FlexRay cycle description.

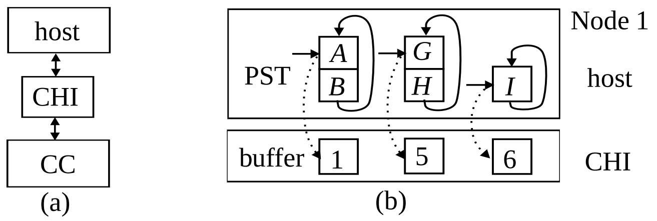

Fig. 2. (a) FlexRay node; (b) Software architecture.

A. Description of the FlexRay Cycle

The FlexRay cycle comprises a *static segment* (SS), a *dynamic segment* (DS), a *symbol window* (SW), and the *network idle time* (NIT). A generic FlexRay cycle is depicted in the upper part of Fig. 1.

Similar to the Time Triggered Protocol (TTP) in [8], the organization of the SS is based on a time-division multiple access (TDMA) scheme. It consists of a fixed number of equal size *static slots* (STS) that are incrementally counted by a *slot counter* in each FC starting from 1. The bus arbitration is performed by uniquely assigning *frame identifiers* (FIDs) to nodes such that in each STS, the node with the FID that is equal to the current value of the slot counter can send a message. Fig. 1 shows a SS with 6 STSs. The FIDs have been assigned such that for example the messages A and B are transmitted by the node with the FID 1.

The DS is similar to ByteFlight [15], and employs the flexible TDMA (FTDMA) approach. The investigation of the DS is not in the scope of this paper and can be studied as an independent scheduling problem. We refer the reader to the companion paper [16] for a detailed description. The SW and the NIT provide time for the transmission of internal control information and protocol-related computations.

B. Software Architecture

In this paper, the case where several network *nodes* are connected by a single FlexRay communication channel is addressed. According to the FlexRay specification [9], each node consists of a *host* and a *communication controller* (CC) that are connected by a *controller-host interface* (CHI) as depicted in Fig. 2 (a). Here, the CHI serves as a buffer between the host and the CC. The host processes incoming messages and generates outgoing messages, while the CC independently implements the FlexRay protocol services.

In order to support the periodic (jitter-free) transmission of *periodic messages* in the SS, we propose the following software architecture. In compliance with the protocol specification [9], each slot in the FC with its corresponding FID is uniquely assigned to a host, where multiple FIDs can be allocated to each host. In addition, we adopt the assignment of messages to FIDs in [11] such that each individual message cannot have more than one FID. With this prerequisite, we propose that each host holds a *periodic scheduling table* (PST) per allocated FID. In each FC, the PST determines a unique message to be transferred to the corresponding *transmit buffer* of the CHI among the periodic messages with the same FID.

Fig. 2 (b) shows the software architecture for a host that generates the periodic messages A, B (period 2), G, H (period 2), and I (period 1) with the respective FIDs $1, 5, 6$ (see Fig. 1). For each FID, there is a PST that holds the related messages. An arrow indicates the current message to be transferred to the respective transmit buffer in the CHI, i.e., in the first FC the periodic messages A, G and I are transmitted. The arrow moves one step ahead in each FC.

C. Static Segment Scheduling: Issues and Previous Work

The goal of this work is the formulation and solution of the message scheduling problem for the SS of the FlexRay protocol with the software architecture in Section II-B. To this end, we divide our investigation into two subproblems.

1) Signal Framing: In principle, the task of the SS in FlexRay-based communication systems is the exchange of periodic signal data among different nodes, whereby the organization of the SS and the periodic recurrence of signals has to be respected. Hence, on the one hand, the STS size has to be fixed, and on the other hand, an assignment of signals to message frames has to be determined. Here, it is desired that the framing overhead is minimized, and the resulting messages can be fit into FlexRay frames such that the most number of bits are used for messages in each STS. This problem is an *open dimension problem* (ODP) in the context of *bin packing* according to [17], where signals represent *small items* that have to be fit into equal *large objects* (messages), while the size of the large objects is variable. In Section IV, we employ an *integer linear programming* (ILP) formulation in order to compute an optimal message set from a given set of signals.

2) Message Scheduling: In the next step, it has to be noted that the message schedule can be computed independently for each node as FIDs are uniquely assigned to nodes. It also has to be observed that messages in the SS have to be scheduled with minimum jitter, i.e., minimum deviation from the periodicity. Furthermore, it is advantageous if the message schedule for each node requires a small number of FIDs as this guarantees the efficient use of the SS. Together, we want to provide a message schedule for each node that minimizes the jitter for periodic messages, and that requires the allocation of a minimum number of FIDs. In Section III-B, we give a formal description of our performance metrics, and in Section V-B, we state an ILP problem that results in the desired optimal message schedule.

The FlexRay SS and TTP have been studied in [11], [12], [13] using a similar software architecture. [11], [12] perform a timing analysis of message transmissions on FlexRay, while [13] provides heuristics to determine message schedules with small response times including an experimental evaluation.

However, none of the above approaches accounts for the potential jitter in the message transmission. Message scheduling without jitter is investigated for TTCAN in [18]. Although similar ideas can be used to capture the scheduling restrictions and performance metrics, the particular properties of the FlexRay protocol lead to a different approach in our paper.

III. Notation and Performance Metrics

A. Definitions

We formally describe the periodic FlexRay cycle (FC) according to Section II-A. It consists of the *static segment* (SS) and the *dynamic segment* (DS). The duration of the SS in ms is T_{SS}, and it comprises N_{STS} static slots (STS) with the duration T_{STS}, i.e., $T_{\mathrm{SS}} = N_{\mathrm{STS}} \cdot T_{\mathrm{STS}}$. Furthermore, T_{STS} has to be a multiple of the so-called *macrotick* with the fixed duration T_{MT} (according to [9], $1\mu s \leq T_{\mathrm{MT}} \leq 6\mu s$). Respecting (1), we define the STS duration T^b_{STS} required to transmit frames with b two byte-words as

$$T^b_{\mathrm{STS}} := \lceil \frac{b \cdot 20\mathrm{bit} + O_{\mathrm{F}}}{T_{\mathrm{MT}} \cdot C} \rceil \cdot T_{\mathrm{MT}}. \tag{2}$$

Here, $C = 10$ Mbit/s denotes the FlexRay bandwidth. Assuming that a set of N *nodes* $\mathcal{N} = \{1, \ldots, N\}$ communicates on the FlexRay bus, we define a map $n_{\mathrm{FID}} : \mathcal{N} \to \mathbb{N}_0$, such that $n_{\mathrm{FID}}(n)$ indicates the number of FIDs that are allocated to each node $n \in \mathcal{N}$. In this case, it is required that $\sum_{n=1}^{N} n_{\mathrm{FID}}(n) \leq N_{\mathrm{STS}}$. As we do not study scheduling for the DS in this paper, we just introduce the DS duration T_{DS}. Together, the duration of the FC including a possible NIT and a SW is $T_{\mathrm{c}} \geq T_{\mathrm{SS}} + T_{\mathrm{DS}}$.

Considering the software architecture for the SS in Section II-B, we define the *scheduling period* N^n_{SP} of each node n as the *least common multiple* (lcm) of its PST periods. Hence, the overall message schedule repeats after the least common multiple $N_{\mathrm{SP}} := \mathrm{lcm}(N^1_{\mathrm{SP}}, \ldots, N^N_{\mathrm{SP}})$ of $N^1_{\mathrm{SP}}, \ldots, N^N_{\mathrm{SP}}$.

For each node $n \in \mathcal{N}$, we denote $\mathcal{S}^n = \{S^n_1, \ldots, S^n_{F^n}\}$ the set of *signals* to be sent on the bus. Each signal $S^n_s \in \mathcal{S}^n$ has a *period* ps^n_s, a *deadline* ds^n_s, and the *signal data* bs^n_s. Observing that all signals have to be scheduled in multiples of the FC duration T_{c}, it is required to choose T_{c} as the *greatest common divisor* (gcd) of the signal periods or an integer divisor of that value. Hence, we express signal periods and deadlines in integer multiples of T_{c}. Signal data are represented in multiples of the bit time τ_{bit}, where bit and τ_{bit} are used interchangeably fitting to the context. As only periodic signals are considered in this work, it holds that $ps^n_s = ds^n_s$.

For transmission on the bus, signals can be compiled to form a set of *messages* $\mathcal{M}^n = \{M^n_1, \ldots, M^n_{G^n}\}$ for each node $n \in \mathcal{N}$. Hence, we associate a map $pack^n : \mathcal{M}^n \to 2^{\mathcal{S}^n}$ with each node $n \in \mathcal{N}$, where $pack^n(M^n_m)$ returns the signals in $\mathcal{S}^n$ that are packed into the message $M^n_m \in \mathcal{M}^n$. Here, we require that $ps^n_s = ps^n_t$ for all signals $S^n_s, S^n_t \in pack^n(M^n_m)$, i.e., only signals with the same period can be packed in a message. Then, $pm^n_m := ps^n_s$ denotes the *period*, $dm^n_m := ds^n_s$ denotes the *deadline* and $bm^n_m := \sum_{S^n_s \in pack^n(M^n_m)} bs^n_s$ is the *number of data bits* of M^n_m. If $b = \lceil bm^n_m / 16\,\mathrm{bit} \rceil$ denotes the number of two byte-words of a message M^n_m, then it must hold that $T_{\mathrm{STS}} \geq T^b_{\mathrm{STS}}$ such that M^n_m fits into a single STS.

We now formalize the message scheduling that is implemented in the host and the communication controller. Within a scheduling period N^n_{SP} of node $n \in \mathcal{N}$, we denote $w^n_{m,k}$, $k = 1, \ldots, W^n_m := \lfloor N^n_{\mathrm{SP}} / pm^n_m \rfloor$ the FCs modulo N^n_{SP} where $M^n_m \in \mathcal{M}^n$ is scheduled, i.e., M^n_m is scheduled in the FCs $z \cdot N^n_{\mathrm{SP}} + w^n_{m,k}$, $z \in \mathbb{N}_0$. Using this notation, different performance metrics can be introduced as follows.

B. Performance Metrics

For a signal $S^n_s \in \mathcal{S}^n$ of some node n, the fraction of the FlexRay bandwidth C that is demanded by S^n_s amounts to

$$D^n_s := \frac{bs^n_s}{ps^n_s \cdot T_{\mathrm{c}} \cdot C}. \tag{3}$$

Similarly, for a message $M^n_m \in \mathcal{M}^n$, the fraction of C that is allocated for M^n_m is

$$A^n_m := \frac{(T_{\mathrm{STS}} \cdot C)}{pm^n_m \cdot T_{\mathrm{c}} \cdot C} = \frac{T_{\mathrm{STS}}}{pm^n_m \cdot T_{\mathrm{c}}}. \tag{4}$$

Accordingly, the fraction of C demanded for signal data is

$$D := \sum_{n=1}^{N} \sum_{s=1}^{F_n} D^n_s, \tag{5}$$

and the fraction of C allocated for messages is

$$A := \sum_{n=1}^{N} \sum_{m=1}^{G_n} A^n_m, \tag{6}$$

Our performance metrics are based on (5) and (6).

1) Utilization (U) and FID Allocation (FA): The bandwidth *utilization* U captures how much of the allocated bandwidth is used for signal data transmission in the SS. In Section IV, our goal is to maximize U.

$$U := \frac{D}{A}. \tag{7}$$

The *FID allocation* FA denotes the number of FIDs that have to be allocated for message transmission. It is computed as the sum of the number $n_{\mathrm{FID}}(n)$ of FIDs that are allocated to each individual node $n \in \mathcal{N}$.

$$FA := \sum_{n=1}^{N} n_{\mathrm{FID}}(n). \tag{8}$$

Since $N_{\mathrm{STS}} \geq FA \geq N$, FA represents the minimum length of the SS that is bounded from below by the number of nodes N. In order to provide schedulability and system extensibility, our goal in Section V is to minimize FA.

2) Jitter (J): Periodic messages are to be delivered periodically to the receiving nodes. Hence, ideally, the transmission instant of each message should be scheduled such that it is transmitted without any deviation from the periodicity (jitter).

Consider a periodic message M^n_m. We define the *local jitter* $J^n_{m,k}$ for each sending instant $(z \cdot N^n_{\mathrm{SP}} + w^n_{m,k}) \cdot T_{\mathrm{c}}$, $z \in \mathbb{N}_0$, $k \in \{1, \ldots, W_m\}$ of M^n_m as the deviation of the inter-transmission

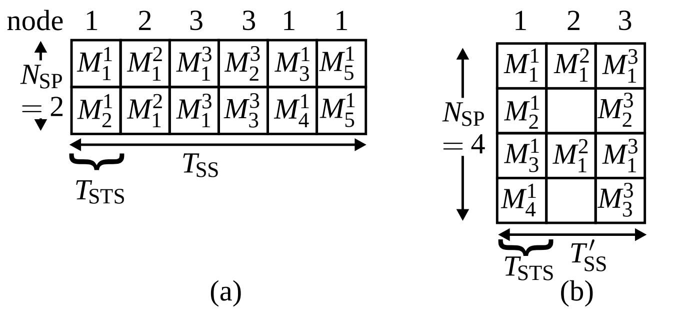

Fig. 3. (a) $T_{\mathrm{SS}} = 6T_{\mathrm{STS}}$; (b) $T'_{\mathrm{SS}} = 3T_{\mathrm{STS}}$.

time at $(z \cdot N_{\mathrm{SP}}^n + w_{m,k}^n) \cdot T_{\mathrm{c}}$ from the actual message period $pm_m^n \cdot T_{\mathrm{c}}$.

$$J_{m,k}^n := \begin{cases} \left|(w_{m,k}^n - w_{m,k-1}^n) - pm_m^n\right| \cdot T_{\mathrm{c}} & \text{for } k \neq 1 \\ \left|(w_{m,k}^n + N_{\mathrm{SP}}^n - w_{m,W_m^n}^n) - pm_m^n\right| \cdot T_{\mathrm{c}} & \text{otherwise.} \end{cases} \tag{9}$$

The average *jitter per message* M_m^n in one FC is then $J_m^n = (\sum_{k=1}^{W_m^n} J_{m,k}^n)/N_{\mathrm{SP}}^n$. The average *jitter per node* evaluates to $J^n = \sum_{m=1}^{G_m^n} J_m^n$, and the average *jitter for the SS* is

$$J := \sum_{n=1}^{N} J^n. \tag{10}$$

The goal of the FlexRay schedule construction for the SS is to transmit all periodic messages with a minimum FID allocation FA and minimum jitter J. Different from other communication protocols such as TTCAN, FIDs are periodically allocated to nodes due to the direct relation between FIDs and nodes (see Fig. 1). Thus, both the evaluation of the performance metrics and the construction of the message schedule can be performed for each individual FlexRay node without any impact on the other nodes. As a consequence, it is sufficient to schedule the periodic messages for each node n with minimum $n_{\mathrm{FID}}(n)$ and J^n as discussed in Section V.

C. Choice of the FlexRay Cycle Time

According to Section III-A, the FC duration T_{c} is chosen as the gcd of all signal periods, denoted as $T_{\mathrm{c},SS}$, or an integer divisor of $T_{\mathrm{c},SS}$. As an additional requirement, the scheduling approach for sporadic messages in the DS in our companion paper [16] requires that T_{c} is an integer divisor of a variable $T_{\mathrm{c},DS}$. The following argument shows that it is favorable to use the largest possible value of T_{c}, i.e., $T_{\mathrm{c}} = \gcd(T_{\mathrm{c},SS}, T_{\mathrm{c},DS})$.

Assume that the configuration in Fig. 3 (a) is chosen to schedule 9 messages from 3 nodes, where $T_{\mathrm{c}} > T_{\mathrm{SS}} = 6\,T_{\mathrm{STS}}$ and such that $T_{\mathrm{c}} = \gcd(T_{\mathrm{c},SS}, T_{\mathrm{c},DS})$. In this configuration, it is possible to associate STSs to nodes with a granularity of 2 (STSs per FID). In contrast, Figure 3 (b) depicts the case, where the integer divisor $T'_{\mathrm{c}} = T_{\mathrm{c}}/2$ of T_{c} is selected as the FC duration, while maintaining the fraction of time that is allocated to the SS, i.e., $T'_{\mathrm{SS}} = T_{\mathrm{SS}}/2$. Here, the granularity amounts to 4 (STSs per FID). As a result, the message M_5^1 of node 1 cannot be scheduled. Consequently, it is favorable to choose the largest possible T_{c} as claimed above.

TABLE I
SIGNAL SET FOR TWO FLEXRAY NODES

signal	$S_{3,1}^1$	$S_{3,2}^1$	$S_{3,3}^1$	$S_{3,4}^1$	$S_{3,5}^1$	$S_{2,1}^2$	$S_{2,2}^2$	$S_{2,3}^2$
data (bit)	65	50	30	40	35	20	25	10
signal	$S_{2,4}^2$	$S_{2,5}^2$	$S_{2,6}^2$	$S_{1,1}^2$	$S_{1,2}^2$	$S_{1,3}^2$	$S_{1,4}^2$	$S_{1,5}^2$
data (bit)	25	45	30	30	30	15	50	25

IV. Frame Packing of Periodic Signals

In order to formulate the optimization problem for the maximization of (7), we first observe that only signals from the same node and with the same period are packed into the same message.[1] For each $n \in \mathcal{N}$, we define $\mathcal{P}^n = \{p_1, \ldots, p_{P^n}\}$ as the set of different *signal periods* of node n, and for each period $p_j \in \mathcal{P}^n$, we introduce the set of signals $\mathcal{S}_{p_j}^n = \{S_{p_j,1}^n, \ldots, S_{p_j,R_{p_j}^n}\} \subseteq \mathcal{S}^n$ with period p_j. The signals in $\mathcal{S}_{p_j}^n$ have to be transmitted in at most $R_{p_j}^n$ different messages $M_{p_j,1}^n, \ldots, M_{p_j,R_{p_j}^n}^n$. For each such message $M_{p_j,k}^n$, $k = 1, \ldots, R_{p_j}^n$ and for each signal $S_{p_j,i}^n \in \mathcal{S}_{p_j}^n$, we introduce a binary variable $x_{p_j,i,k}^n$, where $x_{p_j,i,k}^n = 1$ means that the signal $S_{p_j,i}^n$ is packed into the message $M_{p_j,k}^n$ and otherwise $x_{p_j,i,k}^n = 0$. With the additional constraint that each signal has to be packed into exactly one message, it must hold that

$$0 \leq x_{p_j,i,k}^n \leq 1 \text{ for } i, k = 1, \ldots, R_{p_j}^n, \tag{11}$$

$$\sum_{k=1}^{R_{p_j}^n} x_{p_j,i,k}^n = 1 \text{ for } i = 1, \ldots, R_{p_j}^n. \tag{12}$$

Hence, for all nodes n and for all periods $p_j \in \mathcal{P}^n$, the number of data bits for the message $M_{p_j,k}^n$ is

$$bm_{p_j,k}^n = \sum_{i=1}^{R_{p_j}^n} x_{p_j,i,k}^n \cdot bs_{p_j,i}^n, \tag{13}$$

i.e., the sum of all signal data packed into the message.

For illustration, assume that the signal set in Table I with the respective amount of data bits is given. There are 2 nodes, where node 1 has 5 signals with period 3, and node 2 has 6 signals with period 2 and 5 signals with period 1. For node 1, we provide $R_3^1 = 5$ messages $M_{3,k}^1$, $k = 1, \ldots, 5$ with the respective variables $x_{3,i,k}^1$, $i, k = 1, \ldots, 5$. A possible evaluation of these variables that obeys (12) is $x_{3,1,2}^1 = x_{3,2,2}^1 = x_{3,3,1}^1 = x_{3,4,2}^1 = x_{3,5,1}^1 = 1$ and $x_{3,i,k}^1 = 0$ for all other combinations of i and k. Then, the signals $S_{3,3}^1$, $S_{3,5}^1$ are packed in message $M_{3,1}^1$, and the signals $S_{3,1}^1$, $S_{3,2}^1$, $S_{3,4}^1$ are packed in message $M_{3,2}^1$. All other messages are not used.

The FlexRay specification as described in Section II-A states that all messages have to fit into the STS duration T_{STS}, where the minimum and maximum value of T_{STS} are achieved when using 2 and 127 two byte-words of data, respectively.

[1]Note that the choice of T_{c} in Section III-C is not affected by this approach as the resulting message periods equal the signal periods.

This requirement is captured by the following equations.

$$T_{\rm STS} = k_{\rm STS} \cdot T_{\rm MT}, \tag{14}$$

$$T^2_{\rm STS} \le T_{\rm STS} \le T^{127}_{\rm STS} \tag{15}$$

$$20\,\text{bit} \cdot \lceil bm^n_{p_j,k}/16 \rceil \tau_{\rm bit} \le y^n_{p_j,k} \cdot (T_{\rm TST} - O_{\rm F} \cdot \tau_{\rm bit}), \tag{16}$$

$$0 \le y^n_{p_j,k} \le 1 \tag{17}$$

In (14), the new integer variable $k_{\rm STS}$ captures that the duration of the STS has to be a multiple of $T_{\rm TM}$, while (15) expresses the limits of $T_{\rm STS}$ as given by the FlexRay specification. Furthermore, (16) states that each message with size $bm^n_{p_j,k}$ has to fit into $T_{\rm STS}$. Here, the new binary variable $y^n_{p_j,k}$ is 1 if at least one signal is packed into the message $M^n_{p_j,k}$, i.e., the message is used. Otherwise both $y^n_{p_j,k}$ and $bm^n_{p_j,k}$ are 0.

For the example, we choose $\tau_{\rm bit} = 0.1\mu$s/bit and $T_{\rm MT} = 3\mu$s. Also assume that $O_{\rm F} = 90\,$bit. Considering node 1 as described above, it holds that $bm^1_{3,1} = 65\,\text{bit} > 0 \Rightarrow y^1_{3,1} = 1$, and $bm^1_{3,2} = 155\,\text{bit} > 0 \Rightarrow y^1_{3,2} = 1$. Hence, with (14) and (16), $T_{\rm STS} = k_{\rm STS} \cdot 3\mu\text{s} \ge 20.0\mu\text{s} + 9.0\mu\text{s} \Rightarrow k_{\rm STS} \ge 10$. The remaining variables $bm^1_{3,k}$ and $y^1_{3,k}$ are 0.

The objective of packing signals into frames is to maximize the utilization as defined in (7). We first note that this is equivalent to minimizing A in (6), since D in (5) is constant. It holds for each message $M^n_{p_j,k}$, that the value $A^n_{p_j,k}$ in (4) evaluates to $A^n_{p_j,k} = y^n_{p_j,k} \cdot T_{\rm STS}/(p_j \cdot T_{\rm c})$. Combining all variables in (11) to (17) in a vector X, the optimization problem can be written as

$$\min_X \sum_{n=1}^{N} \sum_{p_j \in \mathcal{P}^n} \sum_{k=1}^{R^n_{p_j}} \frac{y^n_{p_j,k} \cdot T_{\rm STS}}{p_j \cdot T_{\rm c}}. \tag{18}$$

subject to the constraints in (11) to (17).

The output of the minimization in (18) is (i) the optimal value for the STS time $T_{\rm STS}$, and (ii) the packing map $pack^n$ for each node n: for each $M^n_{p_j,k}$ with $y^n_{p_j,k} = 1$, it holds that

$$pack^n(M^n_{p_j,k}) = \{S^n_{p_j,i} \in \mathcal{S}^n_{p_j} | x^n_{p_j,i,k} = 1\}, \tag{19}$$

i.e., together we arrive at the *optimal STS time* and the *optimal message set* to be used for scheduling in Section V.

In our example node 1, it must hold that $y^1_{3,1} = y^1_{3,2} = 1$ such that two messages accommodate the signals with period 3. Accordingly, (19) implies that $pack^1(M^1_{3,1}) = \{S^1_{3,3}, S^1_{3,5}\}$, and $pack^1(M^1_{3,2}) = \{S^1_{3,1}, S^1_{3,2}, S^1_{3,4}\}$.

Unfortunately, combining (16) and (14), and (18) and (14), it turns out that the optimization problem in (18) is a *nonlinear integer programming* (NIP) problem. However, investigating the particular structure of our formulation, the NIP can be decomposed in an *integer linear programming* problem (ILP) and an enumeration over a finite number of values of $T_{\rm STS}$. To this end, instead of including (15) into the optimization, we perform a separate optimization for each possible value of $T_{\rm STS}$, i.e., $T_{\rm STS} = T^b_{\rm STS}$ for $b = 2, \dots, 127$. The remaining optimization problem for each value of b is thus

$$\min_X \sum_{n=1}^{N} \sum_{p_j \in \mathcal{P}^n} \sum_{k=1}^{R^n_{p_j}} \frac{y^n_{p_j,k} \cdot T^b_{\rm STS}}{p_j \cdot T_{\rm c}} \tag{20}$$

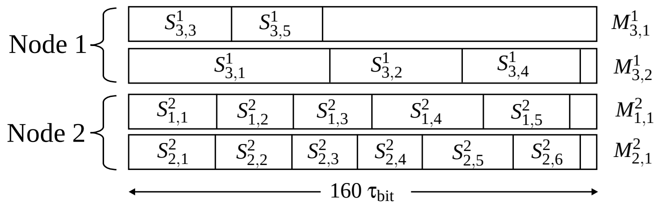

Fig. 4. Signal data packed into messages for two example nodes.

subject to (11), (12), (13), (17) and

$$20\,\text{bit} \cdot \lceil bm^n_{p_j,k}/16 \rceil \tau_{\rm bit} \le y^n_{p_j,k} \cdot (T^b_{\rm TST} - O_{\rm F} \cdot \tau_{\rm bit}). \tag{21}$$

Hence, an ILP has to be solved for each value of b. As a final step, the objective values obtained for $b = 1, \dots, 127$ are compared in order to determine the overall optimum. Note that although we carry out a decomposition, we still obtain the optimal result.

The decomposition has been applied to the two nodes in Table I and a FC duration of $T_{\rm c} = 1$ms. Fig. 4 shows the optimal message set that has been determined using the Gnu Linear Programming Kit (GLPK) [19]. It has 1 message with period 1 and 2, respectively, and 2 messages with period 3. The optimal STS duration is $T_{\rm STS} = T^{10}_{\rm STS} = 30.0\mu\text{s} \ge 20.0\mu\text{s} + O_{\rm F}\,\tau_{\rm bit}$ s.t. signals with 160 bits (i.e., 10 two byte-words) can be packed into each message. With (5) and (6), it holds that $D = 0.030$ and $A = 0.065$, respectively. Thus, the optimal utilization is $U = 0.463$.

V. Message Schedule for the Static Segment

In this section, we assume that an optimal message set has been found as described in Section IV. Hence, we can turn our attention to the message schedule construction for the case of scheduling without jitter (Section V-B) and with minimized jitter (Section V-C). Since the scheduling problem can be solved independently for different nodes as discussed at the end of Section III-B, we consider a generic node $n \in \mathcal{N}$.

A. Scheduling Restrictions without Jitter

If all messages in $\mathcal{M}^n$ have to be scheduled without jitter, then $N^n_{\rm SP} = \text{lcm}(pm^n_1, \dots, pm^n_{G^n})$ has to be chosen. Furthermore, there is a fundamental restriction on the message periods. Assume that $\mathcal{M}^n$ contains four messages with periods $pm^n_1 = pm^n_2 = 3$ and $pm^n_3 = pm^n_4 = 6$. Then, with $N^n_{\rm SP} = 6$, these messages can be scheduled without jitter in the same FID of the FlexRay schedule as depicted in the left part of Fig. 5 (a). The number of allocated FIDs is $n_{\rm FID}(n) = 1$. Now assume that $pm^n_4 = 7$. As can be seen in the right part of Fig. 5 (a), M^n_4 cannot be scheduled in the first FID as it would collide with the other messages. Hence, now $N^n_{\rm SP} = 42$, and although the new value of pm^n_4 is larger than the old value, the new number of allocated FIDs evaluates to $n_{\rm FID}(n) = 2$.

The notion of an *x-group* formalizes this issue.

Definition 5.1 (X-Group) Let $x, y \in \mathbb{N}_0$ be non-negative integers, and $0 \le y \le x - 1$. Then the STSs in the FCs $y + i \cdot x$, $i \in \mathbb{N}_0$ for an FID form an x-group for that FID. □

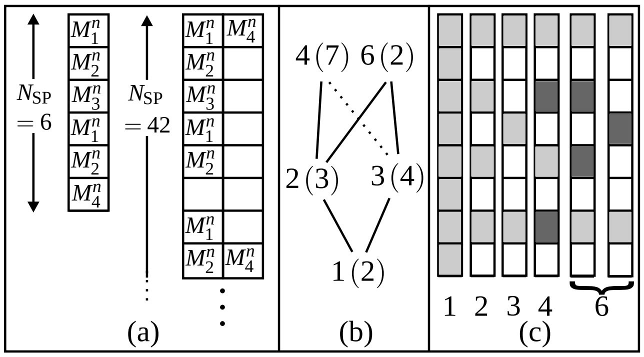

Fig. 5. (a) Occupied FIDs; (b) Partial order of message periods; (c) Illustration of x-groups.

This means that an x-group occupies $1/x$ of the STSs corresponding to an FID in the FlexRay schedule. For example, the message M_1^n in Fig. 5 (a) occupies a 3-group. The above discussion indicates that 3-groups can be scheduled with the same FID with 3-groups and 6-groups but not with 7-groups. The following result generalizes this observation.

Proposition 5.1 (Coprime Message Periods) Let M_m^n and M_l^n be messages with coprime periods, i.e., $\gcd(pm_m^n, pm_l^n) = 1$. Then M_m^n and M_l^n cannot be scheduled with the same FID without jitter. □

Proposition 5.1 relies on the following Lemma.

Lemma 5.1 (Coprime Integer Division) Let a, b be coprime with $a > b$, and $l := \mathrm{lcm}(a, b) = ab$. Also define $r_i := i \cdot a \mod b$ for $i = 1, \ldots, b$. Then it holds that $(r_1, \ldots, r_b)$ is a permutation of the set $\{0, \ldots, b-1\}$. □

Proof: It is sufficient to show that all r_i are distinct, i.e., $r_i \neq r_j \; \forall i \neq j$. Assume the contrary, i.e., $r_j = r_i$ for some $i > j$. Then it holds that $a \cdot i = k_i \cdot b + r_i$ and $a \cdot j = k_j \cdot b + r_j$ for some k_i, k_j. Thus, $a \cdot i - a \cdot j = k_i \cdot b - k_j \cdot b + r_i - r_j$. As $r_i = r_j$ this means that $a(i-j) = b(k_i - k_j)$. Observing that $i - j < b$, we have $a(i-j) < a \cdot b = \mathrm{lcm}(a, b)$. Hence, a and b are not coprime which leads to contradiction. ■

With this result, Proposition 5.1 can be proved.

Proof: W.l.o.g. let M_m^n and M_l^n be scheduled in the cycles $o_m^n + x \cdot pm_m^n$ and $x_l^n + y \cdot pm_l^n$, respectively, and assume that $pm_m^n > pm_l^n$. It has to be shown that there is a FC, where both messages have to be scheduled, i.e., there are values $x < pm_l^n$ and $y < pm_m^n$ that solve the equation

$$o_m^n + x \cdot pm_m^n = o_l^n + y \cdot pm_l^n.$$

Writing $x \cdot pm_m^n = y \cdot pm_l^n + (o_l^n - o_m^n)$, and considering $r_x := o_l^n - o_m^n$, this is equivalent to finding an x such that

$$x \cdot pm_m^n \mod pm_l^n = r_x$$

with $x \in 1, \ldots, pm_l^n$ and $0 \leq r_x \leq pm_l^n - 1$. As pm_m^n and pm_l^n are coprime by assumption, Lemma 5.1 ensures the existence of such x. ■

B. Message Schedule without Jitter

Based on the results in the previous section, we now construct message schedules without introducing jitter.

1) Ordering of Messages: In order to apply Proposition 5.1, we define a *partial order*, i.e., a *reflexive*, *anti-symmetric* and *transitive* order relation, "|" on the set of messages $\mathcal{M}^n$ s.t. for $M_m^n, M_l^n \in \mathcal{M}^n$, $M_m^n | M_l^n$ if pm_m^n divides pm_l^n.

In accordance with the result in Proposition 5.1, this partial order can be related to the respective schedules in Fig. 5 (a). Observing that 3 divides 6, i.e., $3|6$, it holds that $6/3 = 2$ messages with period 6 can be used to fill one 3-group. Hence, the messages in the left part of the figure fill exactly three 3-groups, which corresponds to all slots of one FID. On the other hand, it is clear that a message with period 7 cannot be used to fill any 3-group. Hence, in the right part of the figure, a new FID has to be allocated for such messages.

2) Message Schedule Optimization - Exemplary Study: Having introduced the partial order for messages, we now perform the schedule optimization. Fig. 5 (b) displays the partial order of messages for an example message set. Vertices are represented by combinations $a\,(b)$, where b denotes the number of messages with period a. Vertices are connected by solid lines if the respective periods divide each other.

Applying Proposition 5.1 to the messages in Fig. 5 (b), we point out how messages can be scheduled depending on the prime factorization of their periods. To support our considerations, Fig. 5 (c) illustrates the choices for scheduling messages with the periods in the example message set. Here, each column represents SSs in consecutive FCs for an FID, where light gray boxes indicate the SSs that are used for scheduling messages of the respective period.

On the one hand, there is only one *scheduling choice* (SC) for prime message periods such as 1, 2 and 3. They have to be scheduled in a 1-group, 2-group, and 3-group, respectively, as can be seen in the three leftmost columns of Fig. 5 (c). We denote such SC by (1), (2), and (3), and introduce the associated *SC counts* $n_{(1)}$, $n_{(2)}$ and $n_{(3)}$ that represent the number of messages with the SCs (1), (2), and (3). Hence, $n_{(1)} = 2$, $n_{(2)} = 3$ and $n_{(3)} = 4$. On the other hand, there are messages with non-prime periods such as 4 and 6. Messages with period 4 always occupy one half of a 2-group (i.e., a 2-group of a 2-group). Accordingly, we denote the associated SC as (2,2) with the SC count $n_{(2,2)} = 7$. In the fourth column in Fig. 5 (c), all gray boxes together represent a 2-group, while the light gray boxes describe a 4-group. Differently, messages with period 6 have multiple SCs. They can either fill one third of a 2-group (i.e., 3-group of a 2-group) or one half of a 3-group (i.e., 2-group of a 3-group), as shown in the fifth and sixth column of Fig. 5, respectively. The corresponding SCs are written as $(2, 3)$ and $(3, 2)$ with the respective SC counts $n_{(2,3)}$ and $n_{(3,2)}$. Here, the additional requirement that $n_{(2,3)} + n_{(3,2)} = 2$ has to be fulfilled as there are exactly 2 messages with period 6.

In order to achieve an efficient message schedule, the number of allocated FIDs $n_{\mathrm{FID}}(n)$ has to be minimized for each node $n \in \mathcal{N}$. It is the case, that at least one FID has to be allocated for each coprime period, and can be filled with messages whose periods are divided by the respective coprime period. Referring to Fig. 5 (b) and the coprime period 2, the $n_{(2)} = 3$ messages with period 2 are placed in 3 2-groups, the $n_{(2,2)} = 7$ messages with period 4 are placed in $\lceil 7/2 \rceil = 4$ 2-groups, and the 2 messages with period 6 are placed in $\lceil n_{(2,3)}/3 \rceil$ 2-groups. Together, these messages occupy $3 + \lceil \frac{7}{2} \rceil + \lceil \frac{n_{(2,3)}}{3} \rceil$ 2-groups, which amounts to $\lceil \frac{3}{2} + \frac{1}{2}(\lceil \frac{7}{2} \rceil + \lceil \frac{n_{(2,3)}}{3} \rceil) \rceil$ FIDs. Hence, the messages with periods $1, 2, 3, 4, 6$ occupy

$$\begin{aligned} n_{\mathrm{FID}}(n) \ = \ & n_{(1)} + \lceil \tfrac{3}{2} + \tfrac{1}{2}(\lceil \tfrac{7}{2} \rceil + \lceil \tfrac{n_{(2,3)}}{3} \rceil) \rceil + \\ & + \lceil \tfrac{4}{3} + \tfrac{1}{3} \lceil \tfrac{n_{(3,2)}}{2} \rceil \rceil \end{aligned} \tag{22}$$

FIDs (or 1-groups), where $n_{(2,3)} + n_{(3,2)} = 2$ must hold. Considering the restriction for $n_{(2,3)}$ and $n_{(3,2)}$, and defining $X = \left[n_{(2,3)}, n_{(3,2)} \right]$ as the vector of all unknown variables, the optimization problem for our example is

$$\min_X n_{\mathrm{FID}}(n) \tag{23}$$

subject to the constraint

$$n_{(2,3)} + n_{(3,2)} = 2. \tag{24}$$

Due to the ceiling operators with variable operands such as $\lceil n_{(2,3)}/3 \rceil$, the constraint optimization problem in (23) is a nonlinear integer programming problem (NIP). Fortunately, (23) can be linearized and then solved by integer linear programming (ILP). To this end, we outline a method to replace all ceiling operators by linear expressions. For example, evaluating the term $\lceil n_{(2,3)}/2 \rceil$ can be substituted by the linear expression $(n_{(2,3)} + k_{(2,3)})/2$ with the additional linear constraint $n_{(2,3)} + k_{(2,3)} = K_{(2,3)} \cdot 2$, and the new positive integer variables $k_{(2,3)}$ and $K_{(2,3)}$. In the latter expression, adding the smallest feasible value for $k_{(2,3)}$ is equivalent to carrying out the ceiling operator. Then, $K_{(2,3)} = \lceil n_{(2,3)}/2 \rceil$. Employing this technique for each ceiling operator with variable operands in (22) starting from the innermost ceiling operators, and augmenting X with the new variables, i.e., $X = [n_{(2,3)}, n_{(3,2)}, k_{(2,3)}, k_{(3,2)}, k_{(2)}, k_{(3)}, K_{(2,2)}, K_{(2,3)}, K_{(3,2)}, K_{(2)}, K_{(3)}]$, the linearized optimization problem is

$$\min_X n_{\mathrm{FID}}(n) = \min_X n_{(1)} + K_{(2)} + K_{(3)} \tag{25}$$

with the constraints

$$\begin{aligned} & n_{(2,2)} + k_{(2,2)} = 2 \cdot K_{(2,2)}, \\ & n_{(2,3)} + k_{(2,3)} = 3 \cdot K_{(2,3)}, \\ & n_{(3,2)} + k_{(3,2)} = 2 \cdot K_{(3,2)}, \\ & n_{(2)} + k_{(2)} + K_{(2,2)} + K_{(2,3)} = 2 \cdot K_{(2)}, \\ & n_{(3)} + k_{(3)} + K_{(3,2)} = 3 \cdot K_{(3)}, \\ & n_{(2,3)} + n_{(3,2)} = 2. \end{aligned} \tag{26}$$

Using GLPK, it could be verified that the optimization problem in (25) is solved for $X = [0, 2, 0, 0, 1, 1, 4, 0, 1, 4, 2]$ with a minimum number of $n_{\mathrm{FID}}(n) = 8$ FIDs, i.e., the two messages with period 6 are used to fill a 3-group.

3) Message Schedule Optimization - General Formulation: Based on the ideas in the previous section, we develop a general formulation of the optimization problem. Let $\mathcal{F}_{p_j} = \{f_{p_j,1}, \dots, f_{p_j,K}\}$ be the multiset of prime factors per period $p_j \in \mathcal{P}^n$. Then, each permutation $(f_{p_j,i_1}, \dots, f_{p_j,i_{K-1}}, f_{p_j,i_K})$ of $\mathcal{F}_{p_j}$ represents a SC of messages with period p_j in the sense that each message with the SC $(f_{p_j,i_1}, \dots, f_{p_j,i_{K-1}}, f_{p_j,i_K})$ is scheduled in a f_{p_j,i_K}-group of a $f_{p_j,i_{K-1}}$-group of $\cdots$ of a f_{p_j,i_1}-group. Let $\mathcal{C}_{p_j}$ be the set of all possible SCs for the period p_j, and for each SC $c_{p_j} \in \mathcal{C}_{p_j}$, denote $n_{c_{p_j}}$ the respective SC count in a specific FlexRay schedule. Then it must hold that $\sum_{c_{p_j} \in \mathcal{C}_{p_j}} n_{c_{p_j}} = N^n_{p_j}$, where $N^n_{p_j}$ is the number of messages of node n with period p_j.

We consider Fig. 5 (b) as an example. It holds that $\mathcal{P}^n = \{1, 2, 3, 4, 6\}$ and $\mathcal{F}_6 = \{2, 3\}$. The possible SCs for the period 6 are $(2, 3)$ and $(3, 2)$, which implies that $\mathcal{C}_6 = \{(2, 3), (3, 2)\}$. Hence, it must hold that $\sum_{c_6 \in \mathcal{C}_6} n_{c_6} = n_{(2,3)} + n_{(3,2)} = N^n_6 = 2$, which corresponds to the constraint in (24).

Furthermore, we introduce the map $F^n : \mathbb{N} \to 2^{\mathbb{N}}$ s.t. for $k \in \mathbb{N}$, $F^n(k) := \{f \in \mathbb{N} : (k \cdot f) | p \text{ for some } p \in \mathcal{P}^n \text{ and } f \text{ is prime}\}$. In this expression, $2^{\mathbb{N}}$ is the power set of $\mathbb{N}$, and F^n maps an integer k to all prime factors f in $\mathbb{N}$ s.t. $k \cdot f$ divides a period in $\mathcal{P}^n$. Choosing X as in Section V-B2, the optimization problem can be formalized as

$$\begin{aligned} \min_X n_{(1)} + & \sum_{f_1 \in F^n(1)} \lceil \tfrac{n_{(f_1)}}{f_1} + \tfrac{1}{f_1} \sum_{f_2 \in F^n(f_1)} \lceil \tfrac{n_{(f_1,f_2)}}{f_2} + \\ \tfrac{1}{f_2} \sum \cdots + & \tfrac{1}{f_{k-1}} \sum_{f_k \in F^n(f_1 \cdots f_{k-1})} \lceil \tfrac{n_{(f_1,\dots,f_k)}}{f_k} \rceil \cdots \rceil \rceil \end{aligned} \tag{27}$$

[2]subject to the constraint

$$\forall p_j \in \mathcal{P}^n : \sum_{c_{p_j} \in \mathcal{C}_{p_j}} n_{c_{p_j}} = N^n_{p_j}.$$

Analogous to the previous section, (27) can be transformed into an ILP by replacing each operation $\lceil n_{(f_1,\dots,f_i)}/f_i + \sum \cdots \rceil$ by $(n_{(f_1,\dots,f_i)} + k_{(f_1,\dots,f_i)} + \sum \cdots)/f_i$ with the additional constraint $n_{(f_1,\dots,f_i)} + k_{(f_1,\dots,f_i)} + \sum \cdots = K_{(f_1,\dots,f_i)} \cdot f_i$. Considering the problem formulation in this section, it is readily observed that the message schedule computation can be automatized. Given the number of messages with their respective periods, the linearized version of the optimization problem in (27) can be algorithmically formulated and then solved by an appropriate computational tool such as GLPK.

C. Message Schedule with Jitter Optimization

In the previous section, the message schedule was constructed such that all messages are scheduled without jitter. In this section, we investigate the case, where jitter is allowed for certain messages. To this end, we discuss the configuration in Fig. 5 (b), where jitter is allowed for a number of $N^n_{4,\mathrm{jitter}}$ messages with period 4. Then, such message can be placed into any free x-group with $x < 4$. Accordingly, we extend the graph in Fig. 5 (b) with the additional choices for messages with period 4 as indicated by the dotted line, i.e., 3-groups can be used. We define the variable $n_{(3),4}$ that represents the number of messages with period 4 that are scheduled with the SC (3). It must still hold that the number of messages scheduled with period 4 is equal to N^n_4, i.e.,

$$\sum_{c_4 \in \mathcal{C}_4} n_{c_4} + n_{(3),4} = N^n_4. \tag{28}$$

[2]The iterative summation stops when $f_1 \cdots f_k \in \mathcal{P}^n$.

However, as some messages can be placed with jitter, we obtain the additional constraint

$$n_{(3),4} \leq N^n_{4,\text{jitter}}. \tag{29}$$

In order to include the jitter in the objective function, we evaluate (9) for our special case. Assume that a message with period 4 is scheduled with period 3. It holds that 3 and 4 are coprime. Hence, because of Lemma 5.1, within $\text{lcm}(4,3) = 12$ FCs, the jitter for 3 occurrences of the message assumes all the values between $0\,T_\text{c}$ and $2\,T_\text{c}$. Thus, the accumulated jitter within N^n_SP FCs is $N^n_\text{SP} \cdot \text{gcd}(4,3)/\text{lcm}(4,3) \cdot \sum_{i=0}^{2} i \cdot T_\text{c} = 3T_\text{c}$. Summing up this expression for all messages that are scheduled with jitter yields the overall jitter J^n_4 as

$$J^n_4 = n_{(3),4} \cdot 3T_\text{c}. \tag{30}$$

Combining (22) and (30), the optimization problem for the case with jitter can be formulated, where J^n_4 is added to the sum with a weight ρ.

$$\begin{aligned}\min_X \quad & n_{(1)} + \lceil \tfrac{n_{(2)}}{2} + \tfrac{1}{2}(\lceil \tfrac{n_{(2,2)}}{2} \rceil + \lceil \tfrac{n_{(2,3)}}{3} \rceil) \rceil + \\ & + \lceil \tfrac{n_{(3)}+n_{(3),4}}{3} + \tfrac{1}{3} \lceil \tfrac{n_{(3,2)}}{2} \rceil \rceil + \tfrac{\rho}{T_\text{c} \cdot N^n_\text{SP}} \cdot J^n_4\end{aligned}$$

subject to the constraints in (24) as well as (28) and (29).

The term $1/(T_\text{c} \cdot N^n_\text{SP}) \cdot J^n_4$ reflects the number of FIDs that can be completely filled with the accumulated jitter. Hence, ρ specifies how much jitter is tolerated while using less FIDs.

For our example, the optimization problem has been linearized analogous to (25), and $\rho = 1$ has been chosen. Solving the optimization problem with GLPK yields a smaller number of 7 FIDs, whereby $n_{(3),4} = 1$, i.e., 1 message with period 4 is scheduled with period 3. The jitter for these messages evaluates to $J^n_4 = n_{(3),4} \cdot 3T_\text{c} = 3T_\text{c}$ per scheduling period.

Analogous to Section V-B3, a general formulation of the above considerations can be derived which is not in the scope of this paper. Thus, the computation of the optimal message schedule can be automatized given the additional information about the messages that tolerate jitter and the parameter ρ.

VI. Application to Benchmark Examples

We apply the frame packing and scheduling approaches presented in Section IV and V to the SAE benchmark signal set [14] in order to analyze general characteristics of FlexRay scheduling. The SAE set comprises 22 signals whose periods are integer multiples of $5\,$ms, and that are exchanged among 6 nodes (see Table II). In addition to scheduling the original SAE signal set, we investigate modifications of this set. In particular, the impact of varying the *number of signals* to be scheduled, the *signal periods*, and the *number of nodes* in the network is studied. All experimental results have been carried out using GLPK [19], and for each data point, 100 sample runs have been evaluated.

A. Basic SAE Signal Set

As all signals are represented in multiples of $5\,$ms, we choose $T_\text{c} = 5\,$ms as discussed in Section III-C. Hence, the set of signal periods is $\{1, 2, 20, 200\}$ and the scheduling period

TABLE II
Signals of the SAE benchmark

sender	1	1	2	3	4	4	4
# signals	2	1	1	2	1	1	1
period/size	1/8	20/8	1/8	1/8	1/8	20/8	200/2
sender	5	5	5	5	6	6	6
# signals	2	2	1	1	4	2	1
period/size	1/8	2/8	200/1	200/2	20/8	200/8	200/1

for the signal set is $N_\text{SP} := \text{lcm}(1, 2, 20, 200) = 200$. We assume that $T_\text{MT} = 3\,\mu\text{s} = 30\,\text{bit}\,\tau_{bit}$.

We first examine the case where each signal is scheduled in an individual frame. All of the signals can fit into the smallest frame with a payload of $32\,$bit. Then, with (1) and (16), $T_\text{STS} = 15\,\mu\text{s} = 150\,\text{bit}\,\tau_\text{bit}$. In this case, we compute the utilization as expressed in (7) as $U = 0.06$. If, in a naive approach, each message is also allocated an individual FID, then $FA = 22$ FIDs are allocated. When we schedule these messages without jitter as presented in Section V-B, the FID allocation is reduced to $FA = 15$ slots. Next, we apply our frame packing approach presented in Section IV to the same signal set followed by computing an optimal schedule without jitter. In this case, e.g., the 4 signals with period 20 of node 6 can be packed into one frame. Hence, the utilization increases to the optimal value of $U = 0.11$ and the FID allocation is further reduced to the optimal value of $FA = 9$ slots.

The same signal set was scheduled on the TTCAN bus with utilizations of $U = 0.095$ (without frame packing) and $U = 0.105$ (with frame packing) in [18]. On TTCAN, the utilization improvement (10.5%) is small compared to FlexRay (83.3%). This reflects the fact that FlexRay exhibits a large framing overhead especially for small signal sizes, and highlights the relevance of signal packing for FlexRay scheduling.

Due to the high FlexRay bandwidth of $C = 10$Mbit/s, the fraction of C that is demanded for signal data of the SAE signal set as defined in (5) is relatively low: $D = 0.0015$. Next, we extend the SAE signal set to point out the characteristics of FlexRay schedules derived by our optimal approach for higher values of D.

B. Extended SAE Signal Set: Benefits of Signal Packing

In today's cars, more than 2500 signals are exchanged over in-vehicle networks [1]. Accordingly, this study investigates the benefits of signal packing as introduced in Section IV for signal sets of relevant size. To this end, in each of the following experiments, we first construct an extended signal set by randomly choosing signals from the SAE signal set and randomly assigning each signal to one of the 6 nodes until a given value of D is reached. We increase D gradually up to 0.7 where the total number of signals is more than 9700. Then, we apply our optimal scheduling approach to this extended signal set both after optimally packing the signals to messages ("Packed") and scheduling each signal in an individual message ("Unpacked") as indicated in Fig. 6.

Fig. 6 (a) shows that the utilization increases up to $U = 0.7$ when frame packing is applied, whereas it is less than $U = 0.067$ without frame packing due to the large framing overhead. As our second characteristic, the number of required

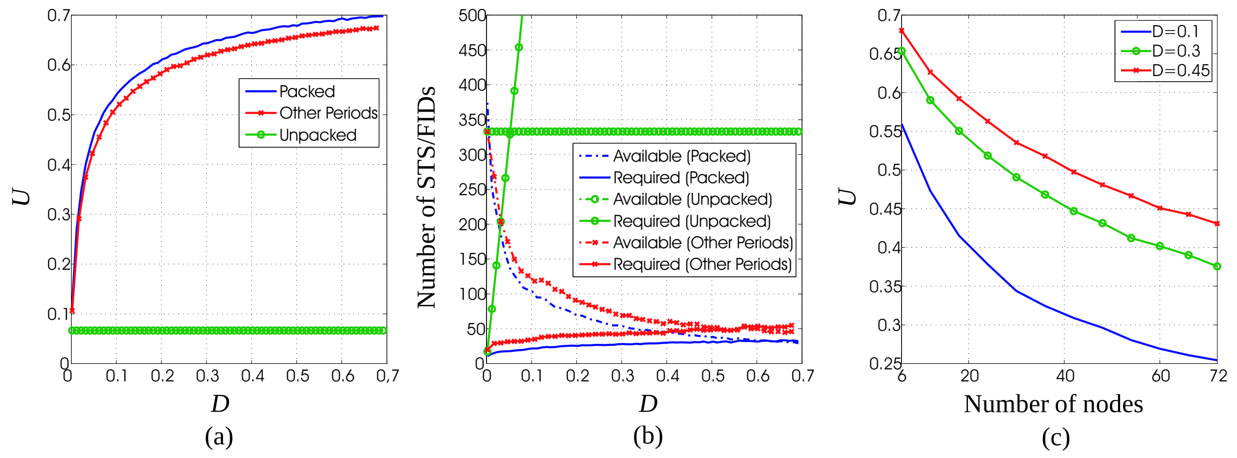

Fig. 6. (a) Utilization w.r.t. D; (b) Required FIDs versus available STS w.r.t. D; (c) Utilization for different numbers of nodes.

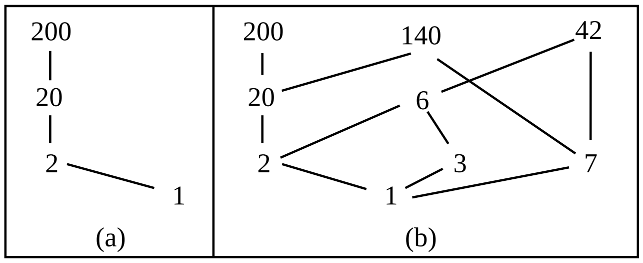

Fig. 7. Partial order of signal periods: (a) SAE set; (b) Extended SAE set.

FIDs is depicted in Fig. 6 (b) together with the respective number of available static slots per FC. For both message sets constructed with and without frame packing, there exists a value of D, where the number of required FIDs exceeds the number of available slots, i.e., not all of the required FIDs can be allocated to achieve schedulability. This value is slightly less than $D = 0.6$ (8400 signals) for the message set with packing, while it is around $D = 0.05$ (750 signals) for the message set without packing due to the very small utilization. Together, the experiments in this section suggest that it is essential to apply signal packing for FlexRay in order to both support an efficient bandwidth use and achieve schedulability even for small signal sets.

C. Extended SAE Signal Set: Impact of Signal Periods

The SAE benchmark set only contains signals with periods that are multiples of each other as shown in Fig. 7 (a). Next, we examine the impact of introducing signals with coprime periods. To this end, we extend the SAE signal set by adding 45 signals with periods from the set $\{3, 6, 7, 42, 140\}$ to the 22 signals of the original SAE set. The partial order of message periods is as illustrated in Fig. 7 (b). We then apply frame packing to these signals for increasing values of D followed by the optimal schedule computation. The results are depicted in Fig. 6 (a) and 6 (b) by the curves labeled “Other periods”.

The utilization achieved for this second extended set is slightly lower than for the first extended set (see Fig. 6 (a)). In this experiment, this is mostly due to the fact that less signals have the same period and can be packed in the same frame, leading to smaller frames with a larger framing overhead. As a consequence, there are more available slots as shown in Fig. 6 (b). However, also the number of required FIDs increases in this experiment due to the incompatibility of coprime message periods as stated in Proposition 5.1. Hence, the main observation is that although the optimal scheduling approach in Section V enables the message schedule construction for large signal sets, the schedulability is decreased if coprime periods are introduced.[3]

D. Extended SAE Signal Set: Increased Number of Nodes

In-vehicle communication in today’s cars involves up to 70 ECUs exchanging signal data [1]. In this experiment we study the impact of employing a larger number of FlexRay nodes while keeping the fraction of C that is demanded for signal data constant at values of $D = 0.1$, $D = 0.3$ and $D = 0.45$. That is, we randomly assign signals from the extended SAE signal set in Section VI-B to up to 72 FlexRay nodes until the respective value of D is reached, and then apply our optimal frame packing and scheduling approach.

Fig. 6 (c) shows that the utilization achieved by frame packing decreases with an increasing number of nodes for each value of D. This is due to the fact that only the signals from the same node can be packed together in the same frame, and the number of signals per node decreases with the increasing number of nodes. With the same argument, the optimal frame size decreases, and hence, the number of available slots increases on networks with more nodes (see Fig. 8). Conversely, owing to the decreased utilization, the number of required FIDs exceeds the number of available slots as the number of nodes increases for $D = 0.3$ and $D = 0.45$. In this context, it is interesting to note, that a larger value of D leads to a violation of schedulability for networks with fewer nodes, i.e., the more nodes are connected on a FlexRay bus, the less signal data can be scheduled.

[3]Note that, according to Proposition 5.1, this is not a limitation of our approach but a inherent property of scheduling periodic messages on FlexRay.

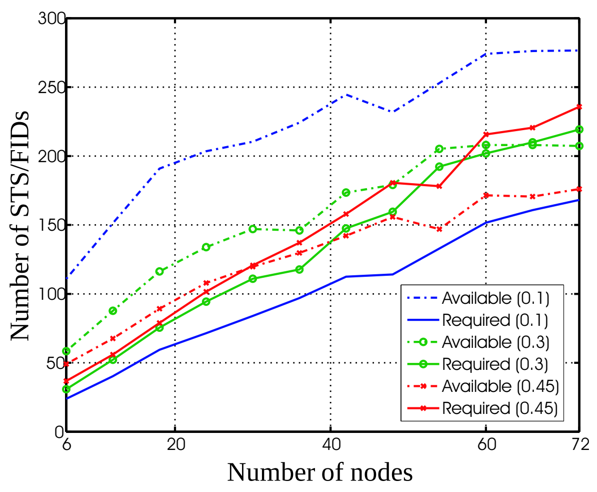

Fig. 8. Required FIDs versus available STSs for different numbers of nodes.

VII. Conclusion

In this paper, the message schedule construction for the static segment of the FlexRay protocol is investigated, i.e., the transmission of *periodic* signal data is considered. To this end, a formal description of the scheduling problem is established, and appropriate performance metrics such as the bandwidth *utilization* and the number of *allocated FIDs* are introduced. Furthermore, the message schedule construction is decomposed into two subproblems.

First, the given periodic signal data have to be assembled to periodic message *frames* that can be transmitted on the FlexRay bus. We formulate a nonlinear integer programming problem (NIP) to address this issue, where the bandwidth utilization of the bus is maximized. The NIP is then reduced to a linear integer program (LIP) by evaluating the properties of the FlexRay static segment.

Second, it is desired that the periodic messages obtained in the first step are scheduled periodically while obeying the FlexRay operation. To solve this problem, we propose an ILP that exploits the properties of the message periods in order to minimize the number of allocated FIDs. Here, both the cases without jitter (no deviation from the periodicity) and with minimum jitter are studied.

General characteristics of FlexRay scheduling have been assessed by an experimental study that applies our optimal frame packing and scheduling approach to a benchmark signal set. It can be concluded that frame packing is essential to achieve a satisfactory utilization, and that less signal data can be scheduled on a FlexRay bus with a larger number of nodes.

References

[1] A. Albert, "Comparison of event-triggered and time-triggered concepts with regard to distributed control systems," in *Embedded World*, 2004.
[2] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert, "Trends in automotive communication systems," *Proc. IEEE*, vol. 93, pp. 1204 – 1224, 2005.
[3] (2005, May) CAN in automation. [Online]. Available: http://www.can-cia.org/can/
[4] K. Tindell, A. Burns, and A. J. Wellings, "Calculating controller area network (CAN) message response times," *Control Engineering Practice*, vol. 3, pp. 1163–1169, Aug. 2000.
[5] (2005, May) TTCAN. [Online]. Available: http://www.can-cia.org/can/ttcan/
[6] G. Leen and D. Heffernan, "TTCAN: A new time-triggered controller area network," *Microprocessors and Microsystems*, vol. 26, pp. 77–94, 2002.
[7] H. Kopetz and G. Bauer, "The time-triggered architecture," *Proc. IEEE*, vol. 91, no. 1, pp. 112–126, 2003.
[8] (2003, Nov.) Time-triggered protocol TTP/C, high-level specification document, protocol version 1.1. [Online]. Available: http://www.tttech.com
[9] (2004, Jun.) FlexRay communication system, protocol specification, version 2.0. [Online]. Available: http://www.flexray.com
[10] 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.
[11] P. Pop, P. Eles, and Z. Peng, "Bus access optimization for distributed embedded systems based on schedulability analysis," in *Proceedings of the Conference on Design, Automation and Test in Europe*. New York, NY, USA: ACM, 2007, pp. 567–575.
[12] S. Ding, N. Murakami, H. Tomiyama, and H. Takada, "A GA-based scheduling method for flexray systems," in *Proceedings of EMSOFT*, 2005.
[13] P. Pop, P. Eles, and Z. Peng, "Schedulability-driven communication synthesis for time-triggered embedded systems," *Real-Time Systems Journal*, vol. 24, pp. 297–325, 2004.
[14] "Class C application requirement considerations," Society for Automotive Engineers, Tech. Rep. J2056/1, 1993.
[15] 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
[16] E. G. Schmidt and K. Schmidt, "Message scheduling for the FlexRay protocol: The dynamic segment," *accepted for publication in Vehicular Technology, IEEE Transactions on*, 2008.
[17] G. Wascher, H. Haussner, and H. Schumann, "An improved typology of cutting and packing problems," *European Journal of Operational Research*, vol. 127, no. 3, pp. 1109–1130, December 2007, available at http://ideas.repec.org/a/eee/ejores/v183y2007i3p1109-1130.html.
[18] K. Schmidt and E. G. Schmidt, "Systematic message schedule construction for time-triggered CAN," *Vehicular Technology, IEEE Transactions on*, vol. 56, no. 6, pp. 3431–3441, Nov. 2007.
[19] (2003) GNU linear programming kit. [Online]. Available: http://www.gnu.org/software/glpk/

Klaus Schmidt Klaus Schmidt received the Diploma and Ph.D. (with distinction) degrees in electrical, electronic, and communication engineering from the University of Erlangen-Nuremberg, Erlangen, Germany, in 2002 and 2005, respectively.

He is currently a Post-Doctoral Researcher with the Chair of Automatic Control, University of Erlangen-Nuremberg. His research interests include controller synthesis for discrete event systems, networked control systems, and vehicular communication networks.

Ece Guran Schmidt Ece G. Schmidt received the B.S. degree in electrical and electronics engineering from the Middle East Technical University, Ankara, Turkey, in 1997 and the M.S. and Ph.D. degrees in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 2001 and 2004, respectively.

She is currently a Faculty Member with the Department of Electrical and Electronics Engineering, Middle East Technical University. Her research interests include high-speed networks, networked control systems, and vehicular communication networks.