Message Scheduling for the FlexRay Protocol: The Dynamic Segment

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

***Abstract*—The FlexRay communication protocol is expected to be the de-facto standard for high-speed in-vehicle communication. In this paper, we formally investigate the scheduling problem for the *dynamic segment* of FlexRay. We take the bounds on the generation times and the timing requirements of the signals into consideration to propose a reservation based scheduling approach that preserves the flexible medium access of the dynamic segment. In order to obtain efficient schedules, we formulate a nonlinear integer programming problem (NIP) that minimizes the required duration of the dynamic segment. This NIP is then decomposed into two linear binary integer programming problems to facilitate the computation of feasible message schedules. An experimental study illustrates our message scheduling approach for the dynamic segment of FlexRay.**

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

I. Introduction

MECHANICAL and hydraulic components in vehicles have been replaced by electronic components since the 1970's. These in-vehicle electronic systems employ *electronic control units* (ECU) which are embedded systems with, e.g., a micro controller, sensors and actuators. Communication networks enable the information exchange among ECUs to support most of their tasks. Today more than 70 ECUs exchange around 2500 signals in luxury cars [1], [2].

The communication networks in vehicles transmit signal data encapsulated in *messages*. Most of these messages are real-time messages, i.e., their timely delivery must be guaranteed. Technically, pre-computed *message schedules* have to be supplied to meet such timing requirements. In addition, considering the fast growth in the number of ECUs and signals in automotive electronics, the communication must be efficient to provide system extensibility.

One of the first in-vehicle communication networks for automotive systems is the Controller Area Network (CAN) [2], [3]. It can provide bounded delay communication among ECUs at data rates between 125 kbit/s to 1 Mbit/s, and is currently the most widely used in-vehicle network. However, it is not suitable for novel applications such as electronic components of power train or x-by-wire applications, which are hard real-time in nature, and require high-speed, robust, and predictable communication. The first attempts to meet these demands are Time-triggered CAN (TTCAN, [4]), Time-triggered Protocol (TTP, [5], [6]) and ByteFlight [7]. TTCAN and TTP are *time-triggered* technologies with predictable medium access, whereas ByteFlight is based on flexible Time Division Multiple Access (FTDMA), which aims at an efficient bandwidth use.

FlexRay in-vehicle communication networks was founded as an industry consortium by BMW, Daimler-Chrysler, Philips and Freescale in the year 2000 [8], [9]. Currently there are more than 150 members in the consortium, and the first series production car with FlexRay was on the road in 2006. FlexRay has a static segment with TDMA operation and a dynamic segment with FTDMA operation. It is expected to be the new de-facto standard combining the advantages of time-triggered and FTDMA communication [2]. It provides two channels with a bandwidth of 10 Mbit/s each, enabling applications such as x-by-wire which were not realizable with CAN.

The dynamic segment of FlexRay is designed to accommodate *sporadic* real-time messages which are generated by event occurrences and have to be transmitted before their *deadline*. To this end, it is required to find feasible message schedules that meet the timing requirements. Previous work on the FlexRay dynamic segment mostly provides methods to test if a given schedule is feasible [10], [11]. In addition, one study [12] analyzes and evaluates deadline monotonic scheduling.

Different from the previous work, we propose a method to synthesize efficient and feasible message schedules. Based on a formal problem description, our approach determines the required system parameters such that the sporadic messages are delivered on time. Adopting ideas from our work in [13], we consider the bounds on the message generation times and the timing requirements for message delivery of the sporadic messages to reserve bandwidth for each message while maintaining the benefits of the FTDMA operation of the FlexRay dynamic segment. In our framework, we define appropriate performance metrics to measure the efficiency of each schedule. Then, Integer Programming is applied to select the most efficient feasible schedule.

The structure of the paper is as follows. In Section II, we describe the operation of FlexRay, and introduce our notation for the system parameters. Section III addresses different issues related to message scheduling for the dynamic segment. Our idea of message grouping in order to reduce the bandwidth reservation is first discussed in Section IV, and then employed in Section V to find optimal message schedules. Section VI presents experimental results, and conclusions are given in Section VII.

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

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

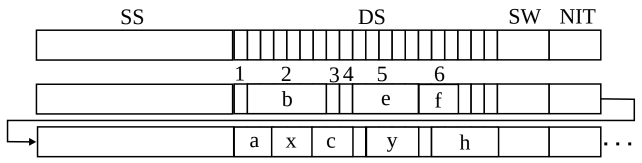


Fig. 1. FlexRay cycle description.

II. The FlexRay Protocol

The FlexRay protocol defines two channels that operate at a bandwidth of $C = 10\,$Mbit/s each leading to a bit time of $\tau_{\text{bit}} = 0.1\,\mu$s. In this work, we consider message transmissions on one FlexRay channel.

A. Description of the FlexRay Cycle

The operation of each FlexRay channel is based on a fixed-duration, repeatedly executed *FlexRay cycle* (FC) that is time-slotted [8]. The FC comprises a *static segment* (SS), a *dynamic segment* (DS), a *symbol window* (SW), and the *network idle time* (NIT). The SW and the NIT provide time for the transmission of internal control information and protocol-related computations. The duration of the FC, SS and DS, which are fixed during the configuration of a given system, are measured in ms and denoted as T_{c}, T_{SS} and T_{DS}, respectively. A generic FC is depicted in the upper part of Figure 1.

The SS is similar to TTP [6] and employs the TDMA approach. The investigation of the SS 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 [14] for a detailed description. The DS is similar to ByteFlight [7] and employs the flexible TDMA (FTDMA) approach. The smallest time unit in the DS is the *minislot* (MS) with a duration of T_{MS} (in ms), and the DS contains a fixed number of N_{DS} MS, where $N_{\text{DS}} \leq N_{\text{DS,max}} = 7994$. The DS consists of consecutive *dynamic slots* (DYS) that are superimposed on MS. If a message is transmitted in a DYS, then the length of the DYS is equal to the number of MS needed for message transmission. Otherwise, the length of the DYS is one MS.

Each node maintains a *slot counter* to follow the progress of the DS. It is initialized to 1 at the beginning of each FC, and is incremented in every DYS. The arbitration procedure ensures that only frames with a Frame Id (FID) that equals the current value of the slot counter can be transmitted [8]. Therefore, we interchangeably use the notion FID to express the Frame ID and the value of the slot counter in the remainder of the paper. The DS in Figure 1 consists of 20 MS. In the first FlexRay cycle, messages are transmitted in the second, fifth, and sixth DYS, whereas the length of, e.g., the second DYS is 6 MS.

B. Messages

We consider a communication system that consists of N *nodes* (ECUs) which are connected by FlexRay, where *the set of nodes* is $\mathcal{N} = \{1, \ldots, N\}$. The nodes exchange *periodic* and *sporadic* real-time messages which are transmitted in *FlexRay Frames*. We assume that all periodic messages are scheduled in the SS as studied in [14]. In this paper, we investigate the transmission of sporadic messages in the DS.

Our representation of the timing properties of sporadic messages follows the lines of related work in [12], [13], [15]–[17]. For each sporadic message, there is a *deadline* which is the largest tolerable time interval between the generation and the transmission of the message. In our work, the deadline includes the message transmission time as well as the maximum jitter of the message as defined in [10]. In addition, the recurrence of a sporadic message is described by its minimum inter-arrival time denoted as *period*, which characterizes the minimum time interval between two consecutive message generations.

The sporadic messages of a node $n \in \mathcal{N}$ constitute a set $\mathcal{M}_{\text{S}}^n = \{M_1^n, \ldots, M_{S_n}^n\}$, and the entire set of sporadic messages is denoted as $\mathcal{M}_{\text{S}} := \bigcup_{n=1}^{N} \mathcal{M}_{\text{S}}^n$. Each sporadic message $M_m^n \in \mathcal{M}_{\text{S}}^n$ has a period pm_m^n and deadline dm_m^n, where $dm_m^n \leq pm_m^n$. The *length* lm_m^n (in MS) of M_m^n can be computed as in [8], including the signal data s_m^n in multiples of two Byte words, the FlexRay framing overhead $s_m^n \cdot 4\text{ bit} + O_{\text{F}}$ and the communication-free DYS idle phase.

$$l_m^n = \lceil (s_m^n \cdot 16\text{ bit} + s_m^n \cdot 4\text{ bit} + O_{\text{F}})\tau_{\text{bit}}/T_{\text{MS}} \rceil, \quad (1)$$

C. Dynamic Segment Scheduling: Issues and Previous Work

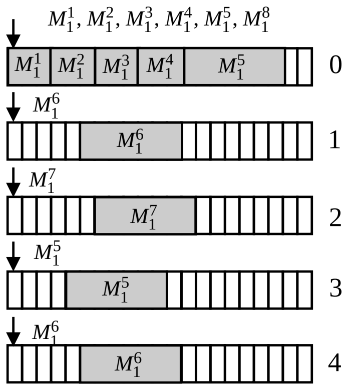
The construction of the FC requires the offline computation of several system parameters. The FC duration T_{c} has to be chosen considering the bandwidth and delay requirements of the messages that are scheduled in both the FlexRay SS and DS, and the duration T_{SS} of the SS and T_{DS} of the DS have to fulfill $T_{\text{SS}} + T_{\text{DS}} \leq T_{\text{c}}$. Hence, it is desirable to keep T_{DS} as small as possible in order to accommodate the bounds given by the SS and the FC time which depend on the message properties in the SS and the DS. Furthermore, a unique assignment of FIDs to nodes has to guarantee that the messages are transmitted before their deadlines.

Since FlexRay is fairly new compared to the legacy bus standards such as CAN, the literature on the DS is limited. In particular, most approaches assume that the above parameters have already been determined and suggest a schedulability *analysis* to verify if all message deadlines are met. [18] performs a basic analysis of FTDMA (based on ByteFlight) and provides guidelines to find the maximum FID such that each message is scheduled on the bus cycle where it arrives. [10] conducts a schedulability analysis for the DS given the above parameters. As an extension of that work, [11] models the DS using service curves and investigates service bounds for messages. The *synthesis* of feasible schedules is first addressed in [12], where a Deadline-monotonic (DM) approach for assigning FIDs to messages is proposed to compute T_{DS}. In this study, the *response times* of the sporadic messages are minimized, since feasible schedules are achieved if all response times are smaller than the respective message deadlines. Our approach directly synthesizes feasible schedules that minimize T_{DS} by employing the knowledge about the message deadlines.

We provide a simple example to demonstrate drawbacks related to the DM assignment of FIDs to messages.

M_1^n	lm_1^n [MS]	dm_1^n [ms]	FID
M_1^1	2	15	1
M_1^2	2	15	2
M_1^3	2	15	3
M_1^4	2	15	4
M_1^5	40	15	5
M_1^6	40	15	6
M_1^7	40	15	7
M_1^8	112	25	8

(a)


$M_1^1, M_1^2, M_1^3, M_1^4, M_1^5, M_1^8$
M_1^1 M_1^2 M_1^3 M_1^4 M_1^5
0
M_1^6
M_1^6
1
M_1^7
M_1^7
2
M_1^5
M_1^5
3
M_1^6
M_1^6
4

(b)

Fig. 2. (a) Message set; (b) DM scheduling.

Example 1 Let the nodes in $\mathcal{N} = \{1, \ldots, 8\}$ communicate using FlexRay with $T_\text{c} = 5$ ms, and let the message set $\mathcal{M}_S = \{M_1^1, \ldots, M_1^8\}$ in Figure 2 (a) be given. Here, we require that $pm_1^n = dm_1^n$ for $n = 1, \ldots, 8$. Furthermore, it is given that the length of the DS is limited to $N_\text{DS} = 120$ MS.

Assume that all messages arrive at the beginning of the DS. Consider the arrival of messages in Figure 2 (b). Although M_1^8 arrives in FC 0, it cannot be transmitted before FC 4, and hence misses its deadline. This is due to the fact that because of the repeated arrival of M_1^5, M_1^6 and M_1^7, there is always a message with a smaller FID to be transmitted, while the remaining DS is shorter than lm_1^8. Note that this arrival scenario can be extended such that M_1^8 is not scheduled indefinitely.

The drawback of DM scheduling is that each message is transmitted in the first DYS where it fits: when M_1^6 arrives (FC 4), there are 2 more FCs until its deadline (FC 6). However, M_1^6 is scheduled in FC 4 and M_1^8 misses its deadline. $\square$

In this paper we propose a *scheduling policy* which tackles this problem. We *reserve* bandwidth in order to provide DYS with certain lengths and periodic recurrence. Then, messages can be assigned to these DYS such that they are only transmitted in their reserved DYS. Based on this reservation idea, our scheduling approach determines *feasible schedules* such that a guaranteed opportunity exists for each message to be transmitted before its deadline while the duration T_DS of the DS is minimized and the duration T_c of the FC is chosen respecting the joint requirements of the SS and the DS as described in Section III-C.

It is important to note that the FTDMA structure of the DS is preserved. If the messages to be transmitted in a reserved DYS are not ready at the time of the DYS, then the DYS is only 1 T_MS long and the next DYS can start immediately.

D. Software Architecture

As specified in [8], the components of each node are a host and a communication controller (CC) that are connected by a controller-host interface (CHI) as shown in Fig. 3 (a). In our framework, the host provides the scheduling functionality for the *sporadic messages* in the DS, while the CC independently implements the FlexRay protocol services. Hence, the FIDs allocated to the nodes do not directly indicate the frames to be sent but indicate the nodes to transmit in a given DYS.

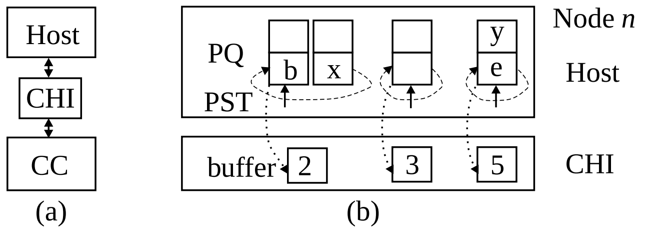


Fig. 3. FlexRay software architecture.

To support reservations as described above, the CHI of each node contains a buffer for each related FID (see Fig. 3 (b)). The host implements a *periodic scheduling table* (PST) per allocated FID that indicates the FCs in which the FID is to be used for message transmission. For each used entry in such a PST, the host maintains a corresponding priority queue (PQ) that holds the messages to be sent with the respective FID and FC sorted by increasing deadline. Then, in each reserved DYS, the highest-priority sporadic message, i.e, the message with the smallest deadline in the PQ associated to the current FC is assigned to the buffer of the respective FID in the CHI.[1]

Figure 3 shows the software architecture for a FlexRay node that is configured to schedule sporadic messages with FIDs $2, 3, 5$ (compare Figure 1). In the PQ, the priority increases toward the lower part of the figure. The PST for FID 2 chooses messages from two PQs in alternate FCs, while the PSTs for the FIDs 3 and 5 provide access for messages in one PQ in each FC. In the first FC, the sporadic messages b and e are assigned to the DYS 2 and 5 as indicated by the solid arrows, while no message is present for the DYS 3. The message y with a lower priority than e has to wait until e is transmitted, while x can be transmitted with FID 2 in the next FC according to its location in the PST.

III. Message Schedule for the Dynamic Segment

A. Scheduling Issues for the Dynamic Segment

In our scheduling policy, each message M_m^n is mapped to a specific DYS which has at least lm_m^n reserved MS at least every time period of dm_m^n. Considering that on the one hand, M_m^n can be generated during the dm_m^n interval only once ($dm_m^n \leq pm_m^n$), and on the other hand, the DYS reoccurs before the deadline of M_m^n, it is sufficient to transmit M_m^n during the reserved DYS such that M_m^n meets the deadline. In the following, we formalize this idea.

A *reservation* R for a node n is a 4-tuple (n, rp, w, l) with the *reservation period* $rp \in \mathbb{N}$, the *offset* $w \in \{0, \ldots, rp-1\}$ and the *reservation length* $l \in \mathbb{N}$. In our scheduling framework, R stands for l MS that are reserved at all FCs $(z \cdot rp + w)$, $z \in \mathbb{N}_0$, while 1 MS is reserved in the remaining FCs. Hence, the *bandwidth reservation* per FC for a given R is $B_R = l/rp$ MS. Two example reservations for a node n are depicted in Figure 4 (a), where $R_1 = (n, 2, 0, 5)$ and

[1]Here, we assume that ties among messages with the same deadline are resolved according to a pre-determined rule.

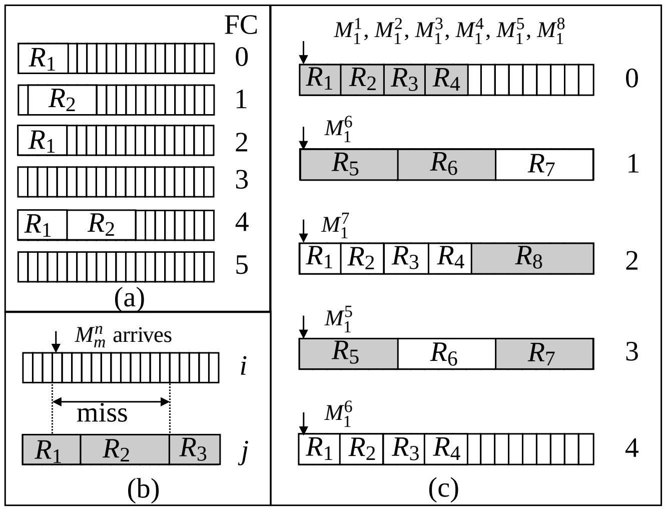


Fig. 4. (a) DYS reservation; (b) Worst case deadline miss; (c) Example 1.

$R_2 = (n, 3, 1, 7)$. The respective bandwidth reservation per FC evaluates to $B_{R_1} = 5/2\,\mathrm{MS}$ and $B_{R_2} = 7/3\,\mathrm{MS}$.

Each reservation for a node n provides a recurring DYS for at least one message in $\mathcal{M}_\mathrm{S}^n$. Denoting the set of all reservations for n as $\mathcal{R}^n$ and the overall set of reservations as $\mathcal{R} := \bigcup_{n=1}^{N} \mathcal{R}^n$, this assignment is expressed by the map $r : \mathcal{M}_\mathrm{S} \to \mathcal{R}$, i.e., we require that there is only one reservation for each message in $\mathcal{M}_\mathrm{S}$. Furthermore, if $r(M_m^n) = (n, rp, w, l)$ for $M_m^n \in \mathcal{M}_\mathrm{S}^n$, then $rp \cdot T_\mathrm{c} \le dm_m^n$ and $l \ge lm_m^n$ must be satisfied such that the corresponding DYS can accommodate the length of M_m^n and M_m^n meets its deadline. Together, our goal is to determine $\mathcal{R}$ and r while optimizing the performance metrics defined in Section III-D.

B. FID Assignment

In order to relate the reservation idea to the software architecture in Section II-D, we note that each reservation has to be associated to an FID and entries in the corresponding PST with their respective PQs. In Section V, we present an optimization approach which guarantees that every FID assignment that obeys the following rules can be used to schedule the sporadic messages.

Let $\mathcal{F}^n$ be the set of FIDs *assigned* to each node n. The map $f^n : \mathcal{R}^n \to \mathcal{F}^n$ that relates each reservation to an FID has to fulfill the following conditions:

(i) FIDs are uniquely assigned to nodes as stated in [8].
(ii) FID 1 is assigned,
(iii) FID assignments are consecutive,
(iv) The total number of FIDs that are assigned to N nodes on a given FlexRay DS is smaller than $N_{\mathrm{DS,max}}$

As long as f^n satisfies (i) - (iv), FID assignments are arbitrary for our scheduling policy.

C. Choice of the FlexRay Cycle Time

There are constraints for choosing T_c such that feasible schedules can be constructed for *any* given message set. If a given message M_m^n is restricted to be transmitted in the DS as discussed in Section II-B and if T_c is chosen larger than dm_m^n, transmitting M_m^n multiple times in the same FlexRay cycle does not guarantee that M_m^n meets the deadline. The interval between the last transmission of M_m^n in the DS of the previous FC and the first transmission of M_m^n in the DS of the current FC can be longer than dm_m^n. Hence, it must hold that $T_\mathrm{c} \le dm_{min}$ where dm_{min} is the minimum deadline among all sporadic message deadlines.

An additional constraint is illustrated in Figure 4 (b), where shaded areas indicate the transmitted messages. Assume that $r(M_m^n) = R_3$ and $rp_3 = dm_m^n$. Let $R_1, R_2, R_3 \in \mathcal{R}$ with $f^n(R_1) = 1$, $f^n(R_2) = 2$ and $f^n(R_3) = 3$. In the worst case, M_m^n arrives at MS 3 of FC i and misses its reserved DYS by 1 MS. M_m^n is then transmitted in the next FC j with the reserved DYS for M_m^n, i.e., $j = i + rp_3$. Suppose R_1 and R_2 take up the indicated number of MS in FC j as shown in Figure 4 (b). As $f^n(R_1) < f^n(R_3)$ and $f^n(R_2) < f^n(R_3)$, the scheduling of M_m^n is delayed in FC j, and M_m^n misses the deadline. This deadline miss is less than T_c, as it is due to the relative MS positions between arrival and transmission of M_m^n in FC i and j, respectively. This can be prevented independent of the FID assignments if $(rp_3+1)\cdot T_\mathrm{c} \le dm_m^n$. Following this concept, we denote the required *message reservation period* for a message M_m^n by $rpm_m^n := dm_m^n / T_\mathrm{c} - 1$.

Having determined the message reservation period for each message in $\mathcal{M}_\mathrm{S}$, a good choice for the FC time T_c is the *greatest common divisor* (gcd) of all message reservation periods, since this choice enables reservations with the maximum allowable scheduling periods. We denote this parameter as $T_{\mathrm{c,DS}}$. Furthermore, our result for the SS in [14] indicates that T_c must be an integer divisor of a parameter $T_{\mathrm{c,SS}}$. Hence, we propose to use the FC time $T_\mathrm{c} = \gcd(T_{\mathrm{c,SS}}, T_{\mathrm{c,DS}})$.

For notational convenience, we express dm_m^n, pm_m^n, rpm_m^n and rp_i in the units of T_c for the rest of the paper. In particular, we now have $rpm_m^n = dm_m^n - 1$, i.e., it must be satisfied for a feasible schedule that if $r(M_m^n) = R$, then $rp \le rpm_m^n$.

We revisit Example 1 in Figure 4 (c) where our reservation based scheduling is applied. Here, each message has $r(M_1^i) = R_i$ and $f(M_1^i) = i - 1$ with $rp_i = dm_1^i - 1$ and $i = 1, \ldots, 8$. The shaded areas show the reservations that are used to transmit the messages with the arrival pattern analogous to Example 1. In our approach, M_1^8 has a reserved DYS to be transmitted every $rp_8 = 5 - 1 = 4$ FCs. Although M_1^7 with a smaller FID arrives in FC 2, it waits for its next reserved DYS in FC 3. Thus, M_1^8 is transmitted before its deadline in the reserved DYS in FC 2, and schedulability is achieved with $N_{\mathrm{DS}} = 120\,\mathrm{MS}$.

D. Performance Metrics

We introduce the *cycle load* L_j of a FlexRay cycle j as the first performance metric. It denotes the maximum number of MS that is reserved for message transmission in FC j for an arbitrary assignment of FIDs, considering that at most one FID can be assigned per message. L_j includes both the case where no message is transmitted for an FID (duration of 1 MS) and the case where a message is transmitted. Let $\mathcal{R}_j \subseteq \mathcal{R}$ be the set of all reserved DYS for message transmission on FC j,

TABLE I
SPORADIC MESSAGES FOR EXAMPLE 2

M_m^n	lm_m^n [MS]	rpm_m^n [T_c]	pm_m^n [T_c]	M_m^n	lm_m^n [MS]	rpm_m^n [T_c]	pm_m^n [T_c]
M_1^n	40	2	4	M_4^n	36	18	20
M_2^n	100	4	6	M_5^n	20	20	40
M_3^n	48	8	8				

i.e., $\forall R \in \mathcal{R}_j$, $\exists z \in \mathbb{N}_0$ s.t. $j = (z \cdot rp + w)$. Then, L_j is defined as follows:[2]

$$L_j = \sum_{R \in \mathcal{R}_j} l + (|\mathcal{M}_S| - \sum_{R \in \mathcal{R}_j} 1) = \sum_{R \in \mathcal{R}_j} (l-1) + |\mathcal{M}_S| \quad (2)$$

As the reservations are periodic, the reservation pattern of the DS repeats every RP FCs, where the *DS reservation period* RP is the *least common multiple* (lcm) of all reservation periods. Hence, the FC with the maximum cycle load occurs within RP FCs. We define the *maximum cycle load* as $L_{\max} = \max_{j \in \{1,\ldots,RP\}}(L_j)$. Then, we choose $N_{DS} = L_{\max}$, and minimize $L_{\max}$ in Section V-A to determine a feasible schedule with an as short as possible DS.

The *bandwidth reservation* $B^n = \sum_{R \in \mathcal{R}^n} (l/rp)$ and $B = \sum_{n=1}^{N} B^n$ indicate the number of MS reserved per FC for each node $n \in \mathcal{N}$, and for all of the nodes, respectively.

The reserved MS are dedicated to transmitting specific messages, and they cannot be used to transmit new messages. Hence, a low B for a given message set indicates the efficiency of the bandwidth reservation and the extensibility of the system. We propose to minimize B in Section V-B.

IV. MESSAGE GROUPING

A. Message Grouping: Example

In our setting, the relevant timing properties of each message M_m^n are given as the deadline dm_m^n and the period pm_m^n, where usually $dm_m^n < pm_m^n$. This means that not necessarily all reservations for M_m^n are utilized to transmit a message M_m^n. Similar to our work in [4], we propose to assign multiple messages to the same reservation while preserving schedulability. As a result, more reservations are utilized such that the bandwidth reservation B is minimized.

The following example message set for a node $n \in \mathcal{N}$ demonstrates this approach. The properties of the sporadic message set $\mathcal{M}_S^n = \{M_1^n, \ldots, M_5^n\}$ are listed in Table I.

Example 2 Assume that the messages M_1^n and M_3^n in Table I have the reservations $R_1 = r(M_1^n) = (n, 2, w_1, 40)$ and $R_3 = r(M_3^n) = (n, 8, w_3, 48)$. In this case, a number of $(40/2) + (48/8) = 26$ MS is reserved for R_1 and R_3 per FC. Furthermore, depending on the choice of the offsets w_1 and w_3, it holds that either $L_{\max} = 48$ MS or $L_{\max} = 88$ MS. Here, at least $\lfloor 8/rpm_1^n \rfloor = 4$ DYS *have to be reserved* for M_1^n within 8 FCs to guarantee its timely transmission. However, at most $\lceil 8/pm_1^n \rceil = 2$ messages *can be generated*. Hence, at least 2 reserved DYS for M_1^n remain unused.

[2]Here, $|\mathcal{M}_S|$ denotes the number of messages in $\mathcal{M}_S$.

Alternatively, we can assign M_1^n and M_3^n to the same reservation R_1 such that $r(M_3^n) = r(M_1^n) = R_1 = (n, \min(rpm_1^n, rpm_3^n), w_1, \max(lm_1^n, lm_3^n)) = (n, 2, w_1, 48)$. For every occurrence of R_1, only one message is transmitted. If M_1^n and M_3^n are ready at the same time, we always give M_1^n the higher priority as $rpm_1^n < rpm_3^n$. Thus, M_1^n is transmitted according to the software architecture in Section II-D. Nevertheless, the transmission of M_3^n within $rpm_3^n = 8$ FCs after it is generated is guaranteed because M_1^n can only be generated twice during 8 FCs, leaving out at least two reserved but unused DYS for the transmission of M_3^n.

The benefits regarding the performance metrics defined in Section III-D are as follows. When assigning M_1^n and M_3^n to the same reservation R_1, their contribution to the bandwidth reservation B^n decreases to $\max(lm_1^n, lm_3^n)/\min(rpm_1^n, rpm_3^n) = 24$ MS/T_c. The maximum cycle load for R_1 now is $L_{\max} = \max(lm_1^n, lm_3^n) = 48$ MS. Consequently, such *grouping* can indeed be used to improve the performance metrics as defined in Section III-D. □

It can readily be observed that there is more than one assignment of reservations that includes M_1^n. For example, it is possible to assign M_2^n and M_1^n to the same reservation as 1 DYS for M_1^n remains unused in $rpm_2^n = 4$ FCs. However, not all of such multiple assignments improve our performance metrics. Using separate reservations for M_1^n and M_2^n, the bandwidth reservation is $(lm_1^n/rpm_1^n) + (lm_2^n/rpm_2^n) = 45$ MS/T_c, whereas when assigning them to the same reservation, the total bandwidth reservation becomes $\max(lm_1^n, lm_2^n)/min(rpm_1^n, rpm_2^n) = 50$ MS/T_c which increases B^n. Accordingly, such groups will be excluded in the optimization in the next section.

B. Message Grouping: General Formulation

In this section, the construction of *message groups*, i.e., assignments of (multiple) messages to a reservation for a node $n \in \mathcal{N}$ is generalized. A message group $\mathcal{G}_q^n \subseteq \mathcal{M}_S^n$, $q \in \mathbb{N}$ corresponds to the reservation $g^n(\mathcal{G}_q^n) = (n, rp, w, l)$, where $g^n(\mathcal{G}_q^n) = r(M_m^n)$ for all $M_m^n \in \mathcal{G}_q^n$. Here, we determine $rp = \min_{M_m^n \in \mathcal{G}_q^n}(rpm_m^n)$ and $l_q^n = \max_{M_m^n \in \mathcal{G}_q^n}(lm_m^n)$ as discussed in Section III-C and III-D.

The construction process of a message group $\mathcal{G}_q^n$ starts with an empty group, i.e., any message M_m^n can be added to $\mathcal{G}_q^n$. Then, $\mathcal{G}_q^n = \{M_m^n\}$, $g^n(\mathcal{G}_q^n) = (n, rpm_m^n, w, lm_m^n)$. For the further discussion, we define $RM_{q,l}^n$ and $P_{q,l}^n$ as metrics to check if a new message M_l^n can be added to a non-empty $\mathcal{G}_q^n$. Let $g^n(\mathcal{G}_q^n) = (n, rp, w, l)$.

$RM_{q,l}^n$ denotes the least number of *remaining* DYS that can be used for M_l^n within a time period of rpm_l^n, if all messages with smaller scheduling periods in G_q^n are generated as frequently as possible and scheduled before M_l^n.

$$RM_{q,l}^n = \lfloor rpm_l^n / rp \rfloor - \sum_{M_m^n \in \mathcal{G}_q^n, rpm_m^n \le rpm_l^n} \lceil rpm_l^n / pm_m^n \rceil \quad (3)$$

If $RM_{q,l}^n \geq 1$, then M_l^n can be scheduled in $\mathcal{G}_q^n$ together with the already present higher priority messages.

$P^n_{q,l}$ is the *profit* in B^n when adding M^n_l to $\mathcal{G}^n_q$ compared to the case where M^n_l is scheduled separately.

$$P^n_{q,l} = (l/rp + lm^n_l/rpm^n_l) - \max(lm^n_l, l)/rp \qquad (4)$$

If $P^n_{q,l} \geq 0$, then adding M^n_l to $\mathcal{G}^n_q$ decreases B^n. Considering (3) and (4), M^n_m fits into $\mathcal{G}^n_q$ if $RM^n_{q,l} \geq 1$ and $P^n_{q,l} \geq 0$.

There are already two groups $\{M^n_1\}$ and $\{M^n_1, M^n_3\}$ which include M^n_1 in Example 2. Next, we construct additional groups with M^n_1. Consider $\mathcal{G}^n_1 = \{M^n_1\}$ with $g^n(\mathcal{G}^n_1) = (n, 2, w_1, 40)$. We extend the group to $\mathcal{G}^n_2 = \{M^n_1, M^n_3\}$ with $g^n(\mathcal{G}^n_2) = (n, 2, w_2, 48)$ as discussed above. If we consider M^n_4, we see that $RM^n_{2,4} = 1$ and $P^n_{2,4} = 2$. A new group is $\mathcal{G}^n_3 = \{M^n_1, M^n_3, M^n_4\}$ with $g^n(\mathcal{G}^n_3) = (n, 2, w_3, 48)$. No more new groups can be formed by extending $\mathcal{G}^n_3$ as M^n_4 uses the last available DYS. Another possible message to extend $\mathcal{G}^n_2$ is M^n_5 with $RM^n_{2,5} = 2$ and $P^n_{2,5} = 2$. Then, $\mathcal{G}^n_4 = \{M^n_1, M^n_3, M^n_5\}$ with $g^n(\mathcal{G}^n_4) = (n, 2, w_4, 48)$.

Similarly, the set of all possible message groups can be determined algorithmically. Algorithm 4.1 checks if M^n_l can be added to a given group $\mathcal{G}^n_q$ while all existing messages in $\mathcal{G}^n_q$ are still transmitted within their scheduling periods. There are three possible results. If the result is *no_fit*, then M^n_l does not fit into $\mathcal{G}^n_q$. If the result is *last_fit*, M^n_l is added to $\mathcal{G}^n_q$ but no other messages can be added. If the result is *more_fit*, more messages can be added after M^n_l. If M^n_l can be added to $\mathcal{G}^n_q$ then Algorithm 4.1 generates the update $\mathcal{G}^n_q = \mathcal{G}^n_q \cup \{M^n_l\}$ and adds the new $\mathcal{G}^n_q$ to the set $\mathcal{G}^n$ of all message groups for node n.

Algorithm 4.1 (Check and Add) Input: M^n_l, $\mathcal{G}^n_q$, $\mathcal{G}^n$.

```
Init: result = more_fit
if ($P^n_{q,l} < 0$ or $RM^n_{q,l} < 1$)
    result = no_fit
else
    $\mathcal{G}^n_q := \mathcal{G}^n_q \cup \{M^n_l\}$ and $\mathcal{G}^n = \mathcal{G}^n \cup \mathcal{G}^n_q$
    if($RM^n_{q,l} = 1$)
        result = last_fit
return result                                              □
```

The following Algorithm 4.2 uses Algorithm 4.1 to enumerate all possible groups $\mathcal{G}^n_q \subseteq \mathcal{M}^n_{\mathrm{S}}$. Here, the ordered *list* LM^n_{S} of all messages sorted by increasing deadline is used. Two operators **next**(M^n_l) and **last**(LM^n_{S}) return the message following M^n_l and the last message in LM^n_{S}, respectively. The comparison $M^n_k < M^n_l$ returns $true$ if M^n_k is located before M^n_l in LM^n_{S}. Before the Algorithm 4.2 is run for $M^n_m \in \mathcal{M}^n_{\mathrm{S}}$, $\mathcal{G}^n_q$, $\mathcal{G}^n$ and M^n_{c} are initialized to $\{M^n_m\}$, $\{\{M^n_m\}\}$, and M^n_m, respectively.

Algorithm 4.2 (Group) Input: LM^n_{S}, M^n_{c}, $\mathcal{G}^n_q$, $\mathcal{G}^n$

```
(while $M^n_{\mathrm{c}} <$ last($LM^n_{\mathrm{S}}$))
    $M^n_{\mathrm{c}} =$ next($M^n_{\mathrm{c}}$))
    $temp\mathcal{G}^n_q = \mathcal{G}^n_q$
    result =Check and Add($M^n_{\mathrm{c}}$, $temp\mathcal{G}^n_q$, $\mathcal{G}^n$)
    if (result = more_fit and $M^n_{\mathrm{c}} \neq$ last($LM^n_{\mathrm{S}}$))
        $tempM^n_{\mathrm{c}} = M^n_{\mathrm{c}}$
        Group ($LM^n_{\mathrm{S}}$, $tempM^n_{\mathrm{c}}$, $temp\mathcal{G}^n_q$, $\mathcal{G}^n$)        □
```

The messages in LM^n_{S} are checked to fit in $\mathcal{G}^n_q$. Any remaining capacity is indicated when **Check and Add** returns a *more_fit* value. In this case, a new group is formed which extends $\mathcal{G}^n_q$ with the remaining of the list of messages by running **Group** recursively.

TABLE II
MESSAGE GROUP CONSTRUCTION FOR M^n_1 IN EXAMPLE 2

Init: $LM^n_{\mathrm{S}} = M^n_1, M^n_2, M^n_3, M^n_4, M^n_5$
Call 0: **Group:** $M^n_{\mathrm{c}} = M^n_1$, $\mathcal{G}^n_q = \{M^n_1\}$, $\mathcal{G}^n = \{\{M^n_1\}\}$
$M^n_{\mathrm{c}} = M^n_2$, $temp\mathcal{G}^n_q = \{M^n_1\}$ $P^n_{q,2} = (20 + 25) - 50 = -5$ $\Rightarrow$ result = *no_fit*
$M^n_{\mathrm{c}} = M^n_3$, $temp\mathcal{G}^n_q = \{M^n_1\}$ $P^n_{q,3} = (20 + 6) - 24 = 2$, $RM^n_{q,3} = \lfloor 8/2 \rfloor - \lceil 8/4 \rceil = 2$ $\Rightarrow$ result = *more_fit* $temp\mathcal{G}^n_q = \{M^n_1, M^n_3\}$, $tempM^n_{\mathrm{c}} = M^n_3$
Call 1: **Group:** $M^n_{\mathrm{c}} = M^n_3$, $\mathcal{G}^n_q = \{M^n_1, M^n_3\}$, $\mathcal{G}^n = \{\{M^n_1\}, \{M^n_1, M^n_3\}\}$
$M^n_{\mathrm{c}} = M^n_4$, $temp\mathcal{G}^n_q = \{M^n_1, M^n_3\}$ $P^n_{q,4} = (24 + 2) - 24 = 2$ $RM^n_{q,4} = \lfloor 18/2 \rfloor - (\lceil 18/4 \rceil + \lceil 18/8 \rceil) = 1$ $\Rightarrow$ result = *last_fit* $temp\mathcal{G}^n_q = \{M^n_1, M^n_3, M^n_4\}$ $\mathcal{G}^n = \{\{M^n_1\}, \{M^n_1, M^n_3\}, \{M^n_1, M^n_3, M^n_4\}\}$
$M^n_{\mathrm{c}} = M^n_5$, $temp\mathcal{G}^n_q = \{M^n_1, M^n_3\}$ $P^n_{q,5} = (24 + 1) - 24 = 1$ $RM^n_{q,5} = \lfloor 20/2 \rfloor - (\lceil 20/4 \rceil + \lceil 20/8 \rceil) = 2$ $\Rightarrow$ result = *more_fit* $temp\mathcal{G}^n_q = \{M^n_1, M^n_3, M^n_5\}$, $M^n_{\mathrm{c}} =$ **last**(LM^n_{S})) $\mathcal{G}^n = \{\{M^n_1\}, \{M^n_1, M^n_3\}, \{M^n_1, M^n_3, M^n_4\}, \{M^n_1, M^n_3, M^n_5\}\}$
Return 1: **Group**
$M^n_{\mathrm{c}} = M^n_4$, $temp\mathcal{G}^n_q = \{M^n_1\}$ $P^n_{q,4} = (20 + 2) - 20 = 2$ $RM^n_{q,4} = \lfloor 18/2 \rfloor - \lceil 18/4 \rceil = 4$ $\Rightarrow$ result = *more_fit* $temp\mathcal{G}^n_q = \{M^n_1, M^n_4\}$, $tempM^n_{\mathrm{c}} = M^n_4$
Call 2: **Group:** $M^n_c = M^n_4$, $\mathcal{G}^n_q = \{M^n_1, M^n_4\}$ $\mathcal{G}^n = \{\{M^n_1\}, \{M^n_1, M^n_3\}, \{M^n_1, M^n_3, M^n_4\},$ $\{M^n_1, M^n_3, M^n_5\}, \{M^n_1, M^n_4\}\}$
$M^n_{\mathrm{c}} = M^n_5$, $temp\mathcal{G}^n_q = \{M^n_1, M^n_4\}$ $P^n_{q,5} = (24 + 1) - 24 = 1$ $RM^n_{q,5} = \lfloor 20/2 \rfloor - (\lceil 20/4 \rceil + \lceil 20/20 \rceil) = 4$ $\Rightarrow$ result = *more_fit* $temp\mathcal{G}^n_q = \{M^n_1, M^n_4, M^n_5\}$, $M^n_{\mathrm{c}} =$ **last**(LM^n_{S})) $\mathcal{G}^n = \{\{M^n_1\}, \{M^n_1, M^n_3\}, \{M^n_1, M^n_3, M^n_4\},$ $\{M^n_1, M^n_3, M^n_5\}, \{M^n_1, M^n_4\}, \{M^n_1, M^n_4, M^n_5\}\}$
Return 2: **Group**
$M^n_{\mathrm{c}} = M^n_5$ $temp\mathcal{G}^n_q = \{M^n_1\}$ $P^n_{q,5} = (20 + 1) - 20 = 1$ $RM^n_{q,5} = \lfloor 20/2 \rfloor - \lceil 20/4 \rceil = 5$ $\Rightarrow$ = *more_fit* $temp\mathcal{G}^n_q = \{M^n_1, M^n_5\}$, $M^n_{\mathrm{c}} =$ **last**(LM^n_{S})) $\mathcal{G}^n = \{\{M^n_1\}, \{M^n_1, M^n_3\}, \{M^n_1, M^n_3, M^n_4\}, \{M^n_1, M^n_3, M^n_5\},$ $\{M^n_1, M^n_4\}, \{M^n_1, M^n_4, M^n_5\}, \{M^n_1, M^n_5\}\}$
Return 0: **Group**

We apply Algorithm 4.2 to our example message set in Table I, and construct the message groups for M^n_1 with the scheduling period $rpm^n_1 = 2$. The ordered list evaluates to $LM^n_{\mathrm{S}} = M^n_1, M^n_2, M^n_3, M^n_4, M^n_5$. The step-by-step evaluation of the algorithm is depicted in Table II.

The entire set $\mathcal{G}^n$ obtained by applying Algorithm 4.2 for the rest of the messages in $\mathcal{M}^n_S$ is:

$$\begin{aligned}\mathcal{G}^n \;\; = \;\; & \{\{M^n_1\}, \{M^n_1, M^n_3\}, \{M^n_1, M^n_3, M^n_4\}, \{M^n_1, M^n_3,\\ & M^n_5\}, \{M^n_1, M^n_4\}, \{M^n_1, M^n_4, M^n_5\}, \{M^n_1, M^n_5\},\\ & \{M^n_2\}, \{M^n_2, M^n_4\}, \{M^n_2, M^n_5\}, \{M^n_3\}, \{M^n_4\}, \{M^n_5\}\}\end{aligned}$$

The set $\mathcal{G}^n$ has the following characteristics that need to be addressed. First, there are multiple groups that for example contain the message M^n_1. However, in the final schedule, exactly one reservation for M^n_1 is required. Thus, one out of these groups has to be selected for transmitting M^n_1. Second,

TABLE III
MESSAGE GROUPS FOR EXAMPLE 3

$\mathcal{G}_1 = \{M_1^1\}$, $pm_1^1 = 3$, $dm_1^1 = 5$ $R_1 = (1, 2, w_1, 20)$	$\mathcal{G}_2 = \{M_1^1, M_2^1\}$, $R_2 = (1, 2, w_2, 30)$
$\mathcal{G}_3 = \{M_2^1\}$, $pm_2^1 = 5$, $dm_2^1 = 7$ $R_3 = (1, 4, w_3, 30)$	$\mathcal{G}_4 = \{M_3^1\}$, $pm_3^1 = 4, dm_3^1 = 6$ $R_4 = (1, 3, w_4, 10)$
$\mathcal{G}_5 = \{M_1^2\}$, $pm_1^2 = 3$, $dm_1^2 = 7$ $R_5 = (2, 2, w_5, 22)$	$\mathcal{G}_6 = \{M_1^2, M_2^2\}$ $R_6 = (2, 2, w_6, 48)$
$\mathcal{G}_7 = \{M_1^2, M_2^2, M_3^2\}$ $R_7 = (2, 2, w_7, 48)$	$\mathcal{G}_8 = \{M_1^2, M_3^2\}$ $R_8 = (2, 2, w_8, 30)$
$\mathcal{G}_9 = \{M_1^2, M_4^2\}$ $R_9 = (2, 2, w_9, 42)$	$\mathcal{G}_{10} = \{M_2^2\}, pm_2^2 = 7, dm_2^2 = 9$ $R_{10} = (2, 6, w_{10}, 48)$
$\mathcal{G}_{11} = \{M_3^2\}, pm_3^2 = 7, dm_3^2 = 9$ $R_{11} = (2, 6, w_{11}, 30)$	$\mathcal{G}_{12} = \{M_4^2\}, pm_4^2 = 5, dm_4^2 = 5$ $R_{12} = (2, 4, w_{12}, 42)$

for each group $G_q^n \in \mathcal{G}^n$, the offset of the corresponding reservation $g^n(\mathcal{G}_q^n)$ has not been determined, yet. In the next section, we employ the enumeration of *all* possible message groups as derived above in order to find the choice of groups and reservation offsets that optimizes the performance metrics in Section III-D.

V. OPTIMAL SCHEDULING OF MESSAGES

The *schedule* for $\mathcal{M}^S$ establishes the number of reserved DYS for each $M_m^n \in \mathcal{M}^S$. As we discussed in Section III-A, FIDs can be assigned to reservations arbitrarily. Hence, the parameters left to be determined are the selection of message groups to be used and the offsets for corresponding reservations while minimizing the cycle load.

A message M_m^n can be included in multiple groups in $\mathcal{G}^n$. Among these groups, one $\mathcal{G}_q^n \in \mathcal{G}^n$ with $M_m^n \in \mathcal{G}_q^n$ must be *selected* such that one and only one reservation for M_m^n is included in the schedule. Let $\mathcal{G} := \bigcup_{n=1}^{N} \mathcal{G}^n$ denote the set of all groups, and $\mathcal{G}_{\mathrm{S}} \subseteq \mathcal{G}$ denote the set of *selected groups*. Once $\mathcal{G}_q^n$ is selected, the reservation $(n, rp, w, l) := g^n(\mathcal{G}_q^n)$ is allocated to node n to transmit messages in $\mathcal{G}_q^n$ with a contribution of l/rp to the bandwidth reservation B^n. Furthermore, depending on the offset w, the cycle load of certain FCs is increased by l.

We propose an integer programming approach to determine $\mathcal{G}_{\mathrm{S}}$ and w for the reservation of each $\mathcal{G}_q^n \in \mathcal{G}_{\mathrm{S}}$ such that B and L_{max} are minimized as discussed in Section III-D.

We illustrate our optimal message scheduling approach with the following example.

Example 3 Let $\mathcal{N} = \{1, 2\}$. We assume that the groups in $\mathcal{G}$ have been computed as listed in Table III by running Algorithm 4.2 on a given set of messages $\mathcal{M}_{\mathrm{S}}$.

Our goal is now to determine $\mathcal{G}_{\mathrm{S}}$ and the offsets w_i of the corresponding reservations such that $L_{\max}$ (and thus the required duration T_{DS} of the DS) is minimized. $\square$

A. Exact Formulation

We formulate integer programming problems with two components to find the optimal message schedule. The first component addresses the selection of the message groups and the corresponding reservations. The binary decision variable $g_i \in \{0, 1\}$ takes the value of 1 if $\mathcal{G}_i \in \mathcal{G}_{\mathrm{S}}$ and is 0 otherwise. The second component is determining the reservation offsets. A reservation R_i can have an offset $w_i \in \{0 \ldots rp_i - 1\}$. The binary decision variable $x_{i,k} \in \{0, 1\}$ takes the value of 1 if $w_i = k$ and is 0 otherwise, where $k = 0 \ldots rp_i - 1$. Furthermore, it can readily be observed that the reservation pattern repeats after G_{RP} FCs, where $G_{\mathrm{RP}} = \mathrm{lcm}_{\mathcal{G}_i \in \mathcal{G}}\, rp_i$ is the least common multiple of the reservation periods rp_i corresponding to each group $\mathcal{G}_i \in \mathcal{G}$. Hence, only the FCs $0, \ldots, G_{\mathrm{RP}} - 1$ need to be taken into account.

Consider a reservation R_i which corresponds to $\mathcal{G}_i \in \mathcal{G}$. The contribution of R_i to L_j, $j = 1, \ldots, G_{\mathrm{RP}} - 1$ is as follows:

1) $g_i = 0$: Then, $\mathcal{G}_i \notin \mathcal{G}^S$ and R_i does not add to L_j.
2) $g_i = 1$ and $x_{i,k} = 1$ for $k = j \mod rp_i$: Then, $w_i = k$ and l_i MS are reserved for R_i in L_j.
3) $g_i = 1$ and $x_{i,k} = 0$ for $k = j \mod rp_i$: Then, $w_i \neq k$ and 1 MS is reserved for R_i in L_j.

Accordingly, we can express the cycle load L_j as follows:

$$L_j = \sum_{\mathcal{G}_i \in \mathcal{G}} g_i \cdot (x_{i,k} \cdot l_i + (1 - x_{i,k}) \cdot 1), \qquad (5)$$

where $k = j$ mod rp_i. Any FC $j \in \{0, \ldots, G_{\mathrm{RP}} - 1\}$ can have the maximum cycle load. Assuming without loss of generality that $L_{max} = L_0$, the expression to be minimized is

$$\min_X L_0 = \min_X \sum_{\mathcal{G}_i \in \mathcal{G}} g_i \cdot x_{i,0} \cdot l_i + g_i(1 - x_{i,0}) \cdot 1, \qquad (6)$$

where X is a vector with all variables g_i and $x_{i,k}$. The requirement that only one reservation is selected for each M_m^n and exactly one offset is determined for each used reservation is formulated by the constraints in (7) and (8), respectively. Since $L_{max} = L_0$, there are no FCs $j \in \{1, \ldots, G_{\mathrm{RP}} - 1\}$ with $L_j > L_0$. This constraint is stated in (9).

$$\forall M_m^n, \sum_{i, M_m^n \in \mathcal{G}_i} g_i = 1 \qquad (7)$$

$$\text{for } i = 1 \ldots |\mathcal{G}|, \sum_{k=0}^{rp_i - 1} x_{i,k} = g_i \qquad (8)$$

$$\text{for } j = 1 \ldots G_{\mathrm{RP}} - 1, L_j \leq L_0 \qquad (9)$$

The optimal message schedule is the solution of the optimization problem with the objective function in (6) and the constraints in (7) to (9). Note that it is a nonlinear integer programming problem (NIP), as the computations in (5) and (9) contain products of the decision variables g_i and $x_{i,k}$.

A feasible schedule for Example 3 has been computed using the TOMLAB optimization environment [19]. As a result, $g_2 = g_4 = g_9 = g_{10} = g_{11} = 1$ and $g_i = 0$ for the remaining values of i have been found. Hence, $\mathcal{G}_{\mathrm{S}} = \{\{M_1^1, M_2^1\}, \{M_3^1\}, \{M_1^2, M_4^2\}, \{M_2^2\}, \{M_3^2\}\}$. Furthermore, the corresponding offsets are $w_2 = 0$, $w_4 = 2$, $w_9 = 1$, $w_{10} = 0$, $w_{11} = 1$, and the worst-case maximum cycle load is $L_{\max} = 81$ MS.

Respecting the conditions in Section III-B, the FIDs can now be assigned to the reservations of the selected message groups arbitrarily. However, a further analysis of the NIP solution can reduce the number of required FIDs, and hence

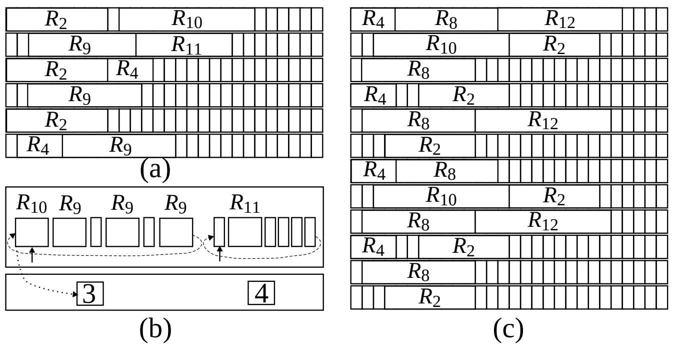


Fig. 5. Optimal reservations for Example 3: (a) NIP; (b) Software Architecture; (c) Decomposed BIP.

lead to a shorter DS, by assigning multiple reservations of one node that never appear in the same FC to the same FID.[3]

An efficient FID assignment for Example 3 is $f^1(R_2) = 1$, $f^1(R_4) = 2$, $f^2(R_9) = 3$, $f^2(R_{11}) = 4$ and $f^2(R_{10}) = 3$ as shown in Fig. 5 (a). Here, the reservations R_9 and R_{10} of node 2 can be assigned to the same FID, instead of choosing a separate FID such as $f^2(R_{10}) = 5$, as they never occupy the same FC. Hence, the 2 unused MS that would appear in the longest FC (with R_2 and R_{10}) in the latter case, can be eliminated. The resulting cycle load is reduced from 81 MS to 79 MS. Fig. 5 (b) depicts the corresponding software architecture according to Section II-D.

B. Two-step Formulation

As solving the offline NIP is a hard problem even for small message sets, we propose to decompose the formulation in (6) to (9) into two linear binary integer programming problems (BIP) in order to enable the problem solution also for large message sets. In the first step, we suggest to select the groups to be scheduled such that our performance metric B is minimized. In the second step, the offsets for these selected groups are computed to minimize L_{max}.

The following objective function is used to minimize B:[4]

$$\min_X B = \min_X \sum_{\mathcal{G}_i \in \mathcal{G}} g_i \cdot (l_i / rp_i) \tag{10}$$

subject to the constraints:

$$\forall M_m^n \in \mathcal{M}_\mathrm{S}, \sum_{i, M_m^n \in \mathcal{G}_i} g_i = 1. \tag{11}$$

Completing the first step yields $\mathcal{G}_\mathrm{S} \subseteq \mathcal{G}$ where $\mathcal{G}_i \in \mathcal{G}_\mathrm{S} \Leftrightarrow g_i = 1$. In the following step, L_{max} is minimized. Here, we denote the number of groups in $\mathcal{G}_\mathrm{S}$ as G_S and the lcm of their reservation periods as $G_\mathrm{S,RP}$.

$$L_\mathrm{max} = \min_X L_0 = \min_X \sum_{\mathcal{G}_i \in \mathcal{G}_\mathrm{S}} x_{i,0} \cdot l_i + (1 - x_{i,0}) \cdot 1 \tag{12}$$

subject to the constraints:

$$\text{for } j = 1 \ldots G_\mathrm{S,RP} - 1 : L_j \leq L_0 \tag{13}$$

$$\text{for } i = 1 \ldots G_\mathrm{S} : \sum_{k=0}^{rp_i - 1} x_{i,k} = 1 \tag{14}$$

The BIP in (10) and (11) has been solved for Example 3 using Tomlab [19]. As a result, $g_2 = g_4 = g_8 = g_{10} = g_{12} = 1$ and $g_i = 0$ for the remaining values of i. Hence, $\mathcal{G}_\mathrm{S} = \{\{M_1^1, M_2^1\}, \{M_3^1\}, \{M_1^2, M_3^2\}, \{M_2^2\}, \{M_4^2\}\}$. Note that, different from the NIP solution, M_3^2 is grouped with M_1^2 in $\mathcal{G}_8$, since the resulting bandwidth reservation $B = 55.1$ MS is smaller than for the NIP solution with $B = 55.7$ MS.

In the next step, we solve the BIP in (12) and (13) for Example 3. $x_{2,1}$, $x_{4,0}$, $x_{8,0}$, $x_{10,1}$ and $x_{12,0}$ are found to be 1, while the rest of the $x_{i,k}$ is 0. The worst-case maximum cycle load is $L_\mathrm{max} = 84$, and can be reduced to 82 MS by an efficient FID assignment as depicted in Fig. 5 (c). Although the resulting DS is slightly larger compared to the NIP solution, now only two BIPs have to be solved.

Together, it can be stated that there is a possible trade-off between the bandwidth reservation B and the maximum cycle load L_max. Furthermore, the decomposed optimization can both provide an upper bound for the minimum L_max and a good initial feasible solution for running the NIP.

VI. Experimental Study

In this section, we present a comparison of the NIP solution in Section V-A and the BIP solution in Section V-B. Furthermore, we study the schedule construction for large message sets. In all our experiments, we used the CPLEX solver of Tomlab [19] to obtain the integer programming solutions, and for each data point, 10 sample runs have been evaluated on a PC with a Dual Core Pentium 4 3.4 GHz processor and 1 GB of RAM.

A. SAE Benchmark Set

The SAE benchmark set in [15] comprises 31 sporadic messages with data sizes smaller than 8 bit whose deadlines and periods are integer multiples of 5 ms, and that are transmitted by 5 nodes (see Table IV). With the choice of $T_\mathrm{MS} = 6.0\,\mu$s and $O_\mathrm{F} = 90$ bit (see [8]), each message fits into a frame with $\lceil (2 \cdot 20 + 90) \cdot 0.1 / 6.0 \rceil$ MS $= 3$ MS. For the SAE benchmark set, the NIP and BIP formulation in combination with the efficient FID assignment yield the same result of $T_\mathrm{DS} = 26 T_\mathrm{MS} = 156\,\mu$s.

TABLE IV
SPORADIC MESSAGES OF THE SAE BENCHMARK

sender	1	2	3	5	6	6
# signals	1	8	6	11	4	1
deadline $[T_\mathrm{c}]$/period $[T_\mathrm{c}]$	4/4	4/10	4/10	4/10	4/10	1/10

[3]An algorithmic approach to tackle such FID assignment has been developed but is not in the scope of this paper.

[4]Note that the fractional coefficients of the objective function can be converted into integers by multiplying B with G_RP.

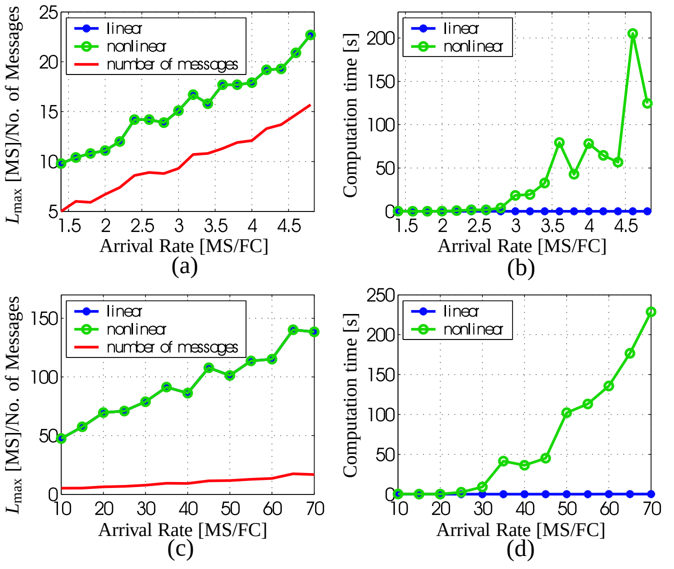


Fig. 6. Cycle load: (a) SAE message set; (b) constructed message set; optimization time: (c) SAE message set; (d) constructed message set.

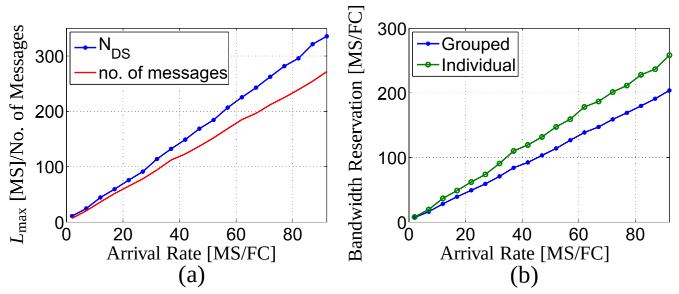


Fig. 7. Large message sets: (a) DS length; (b) Effect of grouping.

B. Comparison between the NIP and the BIP Solution

Our comparison between the solutions of the NIP formulation in Section V-A and the BIP formulation in Section V-B is based on the SAE message set in Table IV that represents a practical message set and the example message set in Table III that was constructed to illustrate the potential different solutions of the NIP and the BIP. As in the SAE message set, we employ 5 nodes communicating on the FlexRay bus. Since the number of messages in both sets is not sufficient to generate considerable traffic on FlexRay, we extend these sets by randomly choosing messages from the respective set and assigning them to one of five FlexRay nodes until a given *arrival rate* is reached. Here, for a given message set $\mathcal{M}_{\mathrm{S}}$, we denote the arrival rate as $\sum_{M_m^n \in \mathcal{M}_{\mathrm{S}}} l_m^n / dm_m^n$. For both message sets, the NIP can be solved for up to 16 messages in the DS, while the solver fails to find an optimal solution for larger message sets. Fig. 6 (a) (SAE messages) and (c) (constructed message set) plot the maximum cycle load $L_{\max}$ as computed in (12) against the arrival rate. It can be seen that the NIP and the BIP formulation yield the same optimization results in all our experiments. This suggests that although cases exist where the BIP does not give an optimal solution (compare Example 3), BIP is suitable in practical examples. Moreover, solving the BIP is much less computationally expensive as illustrated in Fig. 6 (b) and (d).

C. Message Scheduling for Large Message Sets

Further experiments were carried out to evaluate the BIP approach for larger message sets. Fig. 7 (a) shows that more than 270 messages of the SAE message set can be scheduled in a DS with $T_{\mathrm{DS}} \leq 336 \cdot T_{\mathrm{MS}} = 2.0\,\mathrm{ms}$, while computation times of less than $1\,\mathrm{h}$ are required. The benefit of the grouping idea is presented in Fig. 7 (b) by comparing the bandwidth reservation B (see (10)) needed for scheduling individual messages ("individual") to the bandwidth reservation with grouping ("grouped"). In all cases, the bandwidth required for scheduling the given message set is reduced by about 20%.

D. Discussion

It has to be noted that allowing arbitrary FID assignments in the optimization according to Section III-B potentially leads to suboptimal bandwidth reservations. However, it is readily observed that determining a globally optimal FID assignment is computationally intractable since its computational complexity is even higher than the NIP formulated in Section V-A which can only be solved for small message sets as shown in Section VI-B. In this respect, the decomposition of the NIP into two BIP enables the schedule construction for large messages sets as described in Section VI-C, while the experiments in Section VI-B indicate that for practical message sets, the NIP formulation and the BIP formulation lead to identical results. Hence, our BIP approach generates feasible schedules for large message sets, while ensuring a minimal bandwidth reservation and an as short as possible duration T_{DS} of the DS.

VII. Conclusion

This paper addresses the message schedule construction for *sporadic* real-time messages that are to be transmitted in the dynamic segment of the FlexRay protocol. Our approach proposes to reserve bandwidth such that each sporadic message can meet its *deadline*. Based on a formal description of the scheduling problem, we determine a nonlinear integer programming problem (NIP) in order to compute an optimal message schedule. Here, the bandwidth *reservation* and the *cycle load* are employed as appropriate *performance metrics* that have to be minimized.

To facilitate the problem solution, we suggest a decomposition of the NIP into two binary integer programming problems (BIP) to approximate the optimal result. First, we find a set of reservations that minimizes the bandwidth reservation, and then we schedule the obtained reservations such that the cycle load is minimized.

The performance of the proposed approach was evaluated in an experimental study. It is verified that the NIP and the BIP formulations yield identical results for practical message sets. Furthermore, it was possible to construct feasible schedules for large message sets. Together, our approach enables the algorithmic computation of an optimal schedule for the sporadic messages in the FlexRay dynamic segment.

APPENDIX A NOTATION TABLE

TABLE V
NOTATIONS

Notation	Explanation
T_c	FlexRay cycle duration
T_SS,T_DS	Static segment, dynamic segment duration)
τ_bit, T_MS	bit time, minislot duration
N_DS	Number of minislots in the dynamic segment
$\mathcal{N}$	Set of nodes on the FlexRay network
$\mathcal{M}_\mathrm{S}^n$	Sporadic messages of node $n \in \mathcal{N}$
$\mathcal{M}_\mathrm{S}$	All sporadic messages
M_m^n	Message m of node n
pm_m^n, dm_m^n	Period and deadline of M_m^n
lm_m^n	Length of M_m^n
$R = (n, rp, w, l)$	Reservation for node n
rp, w, l	Reservation period, offset, length
$\mathcal{R}^n$, $\mathcal{R}$	Set of reservations for node n, all reservations
$r : \mathcal{M}_\mathrm{S} \to \mathcal{R}$	Map of messages to reservations
L_j	Cycle load of a FlexRay cycle j
$\mathcal{R}_j$	All reserved DYS on FC j,
RP	lcm of all reservation periods
$f^n : \mathcal{R}^n \to \mathcal{F}^n$	Map of reservations to FIDs
$B^n = \sum\limits_{R \in \mathcal{R}^n} (l/rp)$	Bandwidth reservation for node n per FC
$B = \sum\limits_{n=1}^{N} B^n$	Bandwidth reservation for all nodes
$\mathcal{G}_q^n \subseteq \mathcal{M}_\mathrm{S}^n$	Message group q for node n
$g^n(\mathcal{G}_q^n)$	Reservation for $\mathcal{G}_q^n$
$\mathcal{G} = \bigcup_{n=1}^{N} \mathcal{G}^n$	All message groups
$\mathcal{G}_\mathrm{S} \subseteq \mathcal{G}$	Selected groups

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] (2005, May) TTCAN. [Online]. Available: http://www.can-cia.org/can/ttcan/

[5] (2003, Nov.) Time-triggered protocol TTP/C, high-level specification document, protocol version 1.1. [Online]. Available: http://www.tttech.com

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

[7] 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

[8] (2004, Jun.) FlexRay communication system, protocol specification, version 2.0. [Online]. Available: http://www.flexray.com

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

[10] T. Pop, P. Pop, P. Eles, Z. Peng, and A. Andrei, "Timing analysis of the FlexRay communication protocol," in *18th Euromicro Conference on Real-Time Systems*, 2006.

[11] A. Hagiescu, U. D. Bordoloi, S. Chakraborty, P. Sampath, P. V. V. Ganesan, and S. Ramesh, "Performance analysis of FlexRay-based ECU networks," *Design Automation Conference*, pp. 284–289, 2007.

[12] 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.

[13] 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, 2007.

[14] ——, "Message scheduling for the FlexRay protocol: The static segment," *accepted for publication in Vehicular Technology, IEEE Transactions on*, 2008.

[15] "Class C application requirement considerations," Society for Automotive Engineers, Tech. Rep. J2056/1, 1993.

[16] 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 1995.

[17] J. Gamiz, J. Samitier, J. Fuertes, and O. Rubies, "Practical evaluation of messages latencies in CAN," *Emerging Technologies and Factory Automation, 2003. IEEE Conference*, vol. 1, pp. 185–192 vol.1, Sept. 2003.

[18] G. Cena and A. Valenzano, "On the properties of the flexible time division multiple access technique," *Industrial Informatics, IEEE Transactions on*, vol. 2, no. 2, pp. 86–94, 2006.

[19] (2008, July) Tomlab. [Online]. Available: http://tomopt.com/tomlab/

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.

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.