Scheduling for Fault-Tolerant Communication on the Static Segment of FlexRay

Bogdan Tanasa, Unmesh D. Bordoloi, Petru Eles, Zebo Peng
Linköpings Universitet, Sweden
E-mail: {g-bogta, unmbo, petel, zebpe}@ida.liu.se

***Abstract*— FlexRay has been widely accepted as the next generation bus protocol for automotive networks. This has led to tremendous research interest in techniques for scheduling messages on the FlexRay bus, in order to meet the hard real-time deadlines of the automotive applications. However, these techniques do not generate reliable schedules in the sense that they do not provide any performance guarantees in the presence of faults. In this work, we will present a framework for generating fault-tolerant message schedules on the time-triggered (static) segment of the FlexRay bus. We provide formal guarantees that the generated fault-tolerant schedules achieve the reliability goal even in the presence of transient and intermittent faults. Moreover, our technique minimizes the required number of re-transmissions of the messages in order to achieve such fault tolerant schedules, thereby, optimizing the bandwidth utilization. Towards this, we formulate the optimization problem in Constraint Logic Programming (CLP), which returns optimal results. However, this procedure is computationally intensive and hence, we also propose an efficient heuristic. The heuristic guarantees the reliability of the constructed schedules but might be sub-optimal with respect to bandwidth utilization. Extensive experiments run on synthetic test cases and real-life case studies illustrate that the heuristic performs extremely well. The experiments also establish that our heuristic scales significantly better than the CLP formulation.**

I. Introduction

Transient faults are very frequent in modern day electronics and are caused by electromagnetic interference, radiation, temperature variations, etc. [10]. Such faults appear for a very short duration, cause miscalculations in the logic, data corruption, and then disappear without permanent or physical damage to the circuit. These type of faults are in contrast to permanent faults (e.g., those caused by physical damage) which cause long term malfunctioning. Transient-to-permanent fault ratios can reach upto 100:1 [10]. With a proliferation of electronic devices in cars — modern high-end cars have around 70 ECUs (Electronic Control Units) exchanging upto 2500 signals between them over a field us [1] — it has been observed that automotive electronics are also affected by transient faults [4], [21] and that proper fault tolerant techniques are needed against transient faults.

However, *X-by-wire* automotive applications are safety-critical and must maintain high levels of integrity even in the presence of transient faults. This motivates the need for a reliable communication mechanism for *X-by-wire* systems, which is fault tolerant and has predictable timing behavior. Reliability can be enhanced by re-transmitting faulty messages over the communication bus, which leads to extra load on the bus bandwidth. At the same time, however, communication bandwidth is becoming an increasingly scarce resource with more and more *X-by-wire* features being added to newer vehicles. Thus, reliable communication must be achieved at the expense of minimal extra load on the bandwidth. In this work, we propose a framework for synthesizing message schedules on the automotive communication bus, FlexRay, such that the constructed schedules satisfy the real-time constraints even in the presence of transient or intermittent faults. We provide formal guarantees that the generated schedules achieve the desired reliability goal, while minimizing the bandwidth utilization.

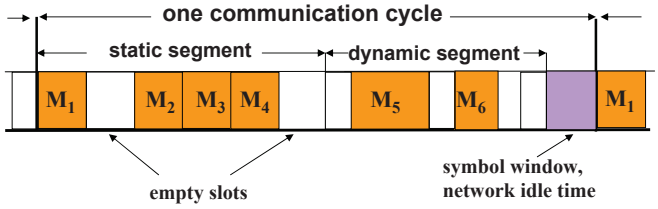

Fig. 1. One communication cycle in FlexRay. In the static segment, a colored slot imply that the slot has been assigned to a message. In the dynamic segment, the colored portions imply a message has been given access based on its priority.

Motivation for focusing on FlexRay: We focus on FlexRay because it is backed by a wide consortium [7] of automotive manufacturers and suppliers and is poised to become the de-facto standard for automotive communication systems very soon. The firs car with a pilot application on FlexRay technology hit the road in 2006, and very recently, BMW has rolled out its 7 series with a FlexRay equipped brake application [5]. As the cost associated with FlexRay deployment is expected to go down over the next few years, more and more *X-by-wire* applications are expected to communicate over the FlexRay bus. This popularity of FlexRay has led to a lot of recent interest in scheduling techniques and tool-support targeting FlexRay-based designs.

The popularity of FlexRay is driven by the fact that it allows both *time-triggered* and *event-triggered* communication. In FlexRay, the time-triggered component is known as the static (ST) segment and the event-triggered component is known as the dynamic segment (see Figure 1). Communication activities in the event-triggered dynamic segment are triggered by the occurrence of specifi events and the protocol define a policy for resolving the contention for the shared bus when messages from multiple ECUs or tasks are ready at the same time. This is similar in spirit to the CAN [3] bus protocol, where data

is segmented into frames and each frame is labeled with a priority which is used to resolve bus contention. In the time-triggered static segment, on the other hand, communication activities or frame transfers occur at predetermined points in time, which are commonly referred to as *slots*. The sequence of slots and their lengths are statically define and the resulting schedule repeats itself infinitel . The static segment of FlexRay is thus similar to the case of TTP [15] which is an example of purely time-triggered paradigm.

Event-triggered protocols are more efficien in terms of communication bandwidth usage and allow incremental system design (i.e., new ECUs or tasks can be added without redesigning the system from scratch). However, they are difficul to analyze and hence, verifying timing properties is inherently difficult On the other hand, time-triggered protocols are highly predictable in terms of their temporal behavior, but suffer from poor bandwidth utilization and are infl xible. With its hybrid nature, FlexRay attempts to combine the advantages of both time-triggered and event-triggered paradigms.

A. Our Contributions and Related Work

Inspite of its growing popularity, the utility of FlexRay as an in-vehicle communication network for futuristic high speed safety-critical *X-by-wire* systems is hindered. This is because it lacks an application layer scheme to prevent loss of faulty messages and hence provides no guarantees on reliability. The FlexRay standard does not provide acknowledgment or retransmission mechanism at the application level. In this paper, we shall propose a technique to implement message retransmissions on the FlexRay static segment to provide guarantees on reliability while optimizing the number of slots used, i.e., bandwidth utilization.

As noted before, recent literature abounds with efforts towards the timing and scheduling analysis of the FlexRay communication protocol. For instance, techniques have been proposed to generate schedules for the static segment [22], [17], [18], while analysis mechanisms to predict the timing behavior of messages transmitted over the dynamic segment were proposed in [19], [11]. Moreover, study of sensitivity and robustness of FlexRay design parameters were reported in [14], [9]. Unfortunately, all of the above lines of research assumed a fault free environment.

The only known attempt to enhance the reliability of the messages transmitted over the FlexRay bus in the presence of transient faults was proposed in [16]. Li et. al. [16] formulated the scheduling problem as a mixed integer linear programming algorithm, where the objective was to retransmit as many faulty messages as possible. However, this approach is limited in several ways, the most important being the fact that the proposed mechanism offers no guarantees on the reliability that can be achieved because the messages are chosen in an adhoc manner for retransmission. In contrast to [16], our proposed technique offers formal guarantees that the generated schedules will achieve the desired reliability levels. This is achieved by a systematic probability analysis based on the probability of failure of each message. Secondly, our scheme achieves the reliability goal with minimal extra load on the bandwidth while the approach in [16] attempts

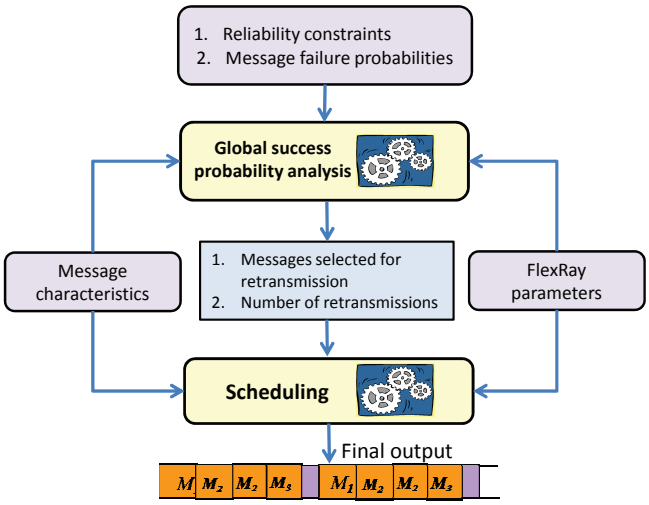

Fig. 2. Overview of our proposed scheme. In the optimal CLP formulation, the various components are intertwined, while, in the heuristic they are decoupled into different stages.

to utilize as much available bandwidth as possible in order to enhance reliability without any formal analysis. Finally, while in [16], the retransmission scheme was built on top of existing FlexRay schedules, our scheme builds the schedule from scratch assuming no existing schedules.

It should be mentioned here that there have been some efforts to design fault tolerant schedules for the TTP protocol [8]. As noted before, TTP is similar in spirit to the FlexRay ST segment because both are time-triggered paradigms and in that sense, these lines of work are related to our work. However, Gaujal et. al. [8] did not consider the hard real-time deadlines of the messages while generating the schedules. This is a major difference compared to our method where we generate schedules that are reliable in the presence of tight timing constraints.

B. Overview of our Scheme

A high level overview of our proposed scheme is illustrated in Figure 2. The scheme consists of two major components: (i) global probability analysis and (ii) scheduling of the messages on the FlexRay static segment. Based on the probability of failure of the messages, characteristics like period and the FlexRay communication cycle length, we conduct a systematic **global probability analysis** and utilize it to fin the messages that must be retransmitted and the number of times they have to be retransmitted in order to achieve the specifie reliability goal. The second component involves constructing a feasible **schedule**, i.e., assigning messages to slots such that the deadlines are satisfied Note that both components would have to be intertwined if we are interested in find ing the optimal solution (where our objective is minimizing the bandwidth utilization). A Constraint Logic Programming (CLP) [2] based formulation would be presented in Section IV towards this. CLP allows users to write constraints in logic programming and solves the problem using branch and bound search based on constraint programming. However, this is computationally expensive and hence, in Section V we present an efficien heuristic. This heuristic based method decouples the two major components mentioned above. The heuristic

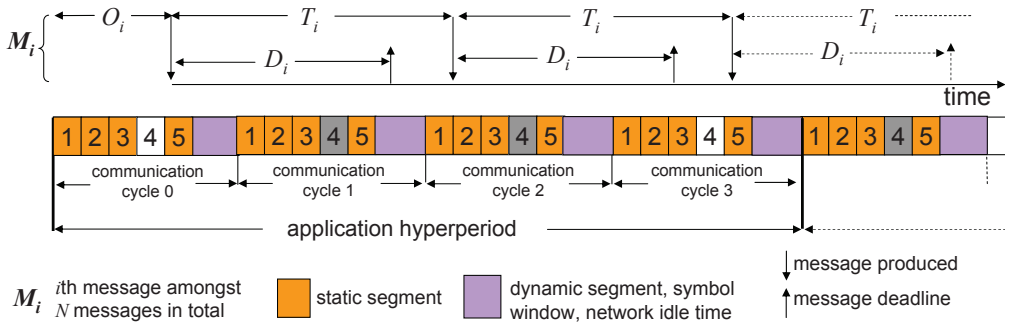

Fig. 3. Illustration of the characteristics of one message M_i. The message is assigned to slot 4 of the FlexRay static segment. In this example, 4 FlexRay cycles constitute one application hyperperiod. The white slots show that the slot goes empty, while the black slots show that the message is being transmitted.

firs searches for the minimum number of retransmissions by which reliability can be achieved in order to minimize the bandwidth utilization. This is followed by scheduling of the messages and their retransmissions on the bus. We would like to mention here that our heuristic formally guarantees that the constructed schedules satisfy the reliability constraints. Also, the heuristic is safe, i.e., if a feasible schedule does not exist, the heuristic can never construct one. Extensive experiments run on synthetic test cases illustrate that the heuristic returns the same result as the optimal CLP for over 93% of the test cases. The experiments also establish that our heuristic scales significantl better than the CLP formulation. In the next section, we introduce our system model followed by the formal problem statement in Section III.

II. System Model

Our system model consists of 3 components — (i) message characteristics (ii) the FlexRay bus and (iii) the fault model.

Messages: Our system model consists of a set of messages $\{M_1, M_2, \ldots, M_N\}$ generated by application level tasks running on a distributed automotive architecture. Each message M_i is characterized by the following parameters (the characteristics of a message M_i are illustrated in Figure 3).

- **Period:** The period T_i, denotes the rate at which M_i is produced. Typically, a message inherits the period of the task that generates it.
- **Offset:** The offset, O_i, is the latest time after which the firs instance of the message M_i is produced. The offset is expressed with regard to the start of the firs FlexRay communication cycle, when time $t = 0$. Thus, the subsequent instances of M_i are produced latest by $O_i + k \times T_i$ time instants.
- **Deadline:** The deadline D_i, of a message M_i is the relative time since the production of M_i until the time by which the transmission of M_i must end. We assume that no message instance can be overwritten in a transmission buffer if it has not been transmitted. This implies that each instance of a message must be transmitted before the next instance is ready, i.e., $D_i \leq T_i$.
- **Size:** W_i denotes the size of the message in bits.

FlexRay Static Segment: We assume that the messages $\{M_1, M_2, \ldots, M_N\}$ are transmitted over the FlexRay static segment. Let us consider that the FlexRay static segment length is ST and the length of one communication cycle in FlexRay is FC. As mentioned in the previous section, the static segment is partitioned into a fi ed number of equal-length slots. Let us denote this number as NS. Each message is allowed to send a message only during a slot that is allocated to it, and this allocation is determined statically. If a message is not ready, then its slot goes empty (i.e., other messages are not allowed to use it). This is illustrated in Figure 3 where slot 4 of the FlexRay cycle is allocated to M_i. In communication cycles 0 and 3 of the figure the message M_i is not ready and hence, the slot 4 remains empty. In the rest of the communication cycles, M_i is transmitted in slot 4.

The least common multiple of the periods of the messages and the FlexRay communication cycle is called the hyperperiod H, i.e., $H = lcm(T_1, T_2, \cdots, T_N, FC)$. Figure 3 illustrates an example with one message where the hyperperiod consists of 4 communication cycles. Note that if all messages meet their deadlines in the hyperperiod, then the system is schedulable.

Fault Model: The automotive industry currently refers to the international standard (IEC61508 [12]) for functional safety of electronic safety-related systems. The standard identifie various levels of integrity or system reliability. For each level, the standard constrains the permissible probability of system-level failure in **time unit**, τ, which is typically one hour. Following this, we assume that the maximum probability of a system failure due to faults on the FlexRay communication bus in a time unit, τ is constrained by γ. Given γ, we defin $\rho = 1 - \gamma$ as the **reliability goal**. It represents the quantifie performance level with respect to transient faults which has to be met by the FlexRay communication sub-system.

We also assume that the **probability of failure** p_i of each message M_i is known to us. Such probabilities can be computed from the sizes of the messages (W_i) and the Bit Error Rates (BER). The BER values for the FlexRay bus can be computed by utilizing commercial tools like [6], [20] which provide specifi modules to inject faults. Given BER, p_i can be computed as $p_i = 1 - (1 - BER)^{W_i}$.

III. Problem Statement

Our problem can be stated as follows: given the setup described in Section II as an input — (i) fin the number of required retransmissions of each message M_i in order to meet the reliability goal ρ, and (ii) construct a feasible schedule

(i.e., assign slots to the messages $\{M_1, M_2, \ldots, M_N\}$ such that the deadlines of the messages are satisfied — subject to the optimization goal of conserving the bandwidth, i.e., minimizing the number of slots utilized. We illustrate the problem and highlight the challenges with the help of two motivational examples.

Example 1: Consider that two messages M_1 and M_2 with periods $T_1 = 2ms$ (milliseconds) and $T_2 = 1ms$, respectively have to be scheduled on the FlexRay bus with a communication cycle length $FC = 2ms$. Assume that reliability goal is $\rho = 0.12$ over a time unit $\tau = 2ms$ and that the failure probabilities for each message are given as $p_1 = 0.5$ and $p_2 = 0.6$ respectively. In this example, τ and the hyperperiod are of same length, i.e., $2ms$. Thus, within the time unit of functionality τ, which has to be considered for the reliability computation, M_1 occurs once and M_2 occurs twice. Hence, probability of success for M_1 to be transmitted without faults is $1 - p_1 = 0.5$. The probability of success for each instance of M_2 is $1 - p_2$. It follows that the probability of successful transmission for both instances of M_2 is $(1 - p_2)^2 = 0.16$. Thus, considering all instances of both messages, the global probability of success is $0.5 \times 0.16 = 0.08$, which falls short of the system reliability, $\rho = 0.12$. In order to enhance reliability, one might choose to retransmit M_2 because M_2 has a higher probability of failure. By retransmitting M_2 once, the probability of successful retransmission of each instance of M_2 is now $(1 - p_2^2)$. Thus, global probability of successful transmission is now computed as $(1 - p_1) \times (1 - p_2^2)^2 = 0.2048$, which is enough to meet the reliability requirement. However, as illustrated in Figure 4(a), such a scheme involving retransmission of M_2 cannot generate a feasible schedule.

On the other hand, if we retransmit the message M_1, the global probability of successful transmission is $(1 - p_1^2) \times (1 - p_2)^2 = 0.12$ which also meets the reliability goal $\rho = 0.12$. Moreover, a feasible schedule can be now constructed as shown in Figure 4(b).

Example 2: Consider that two messages M_1 and M_2 with periods $T_1 = 3ms$ and $T_2 = 2ms$, respectively have to be scheduled on the FlexRay bus with a communication cycle length $6ms$. Thus, the application hyperperiod is $lcm(T_1, T_2, FC) = 6ms$. Assume that the reliability goal is $\rho = 0.09$ over a time unit $\tau = 6ms$ and that the failure probabilities for each message are given as $p_1 = 0.6$ and $p_2 = 0.4$ respectively. Like the previous example, τ and the hyperperiod are of same length, i.e., $6ms$. Thus, M_1 occurs 2 times and M_2 occurs 3 times within the time unit of functionality, τ. As the probability of successfully transmitting one instance of M_1 is $1 - p_1$, it follows that the probability of successfully transmitting both instances is $(1 - p_1)^2 = 0.16$. Similarly, the probability transmitting all 3 instances of M_2 is $(1 - p_2)^3 = 0.216$. Thus, considering all instances of both messages, the global success probability is $(1 - p_1)^2 \times (1 - p_2)^3 = 0.16 \times 0.216 = 0.03456$, which falls short of the system reliability, $\rho = 0.09$. In order to enhance reliability, if we choose to retransmit M_1 once, overall probability of successful transmission is computed as $(1 - p_1^2)^2 \times (1 - p_2)^3 = 0.08847$, which is still not enough

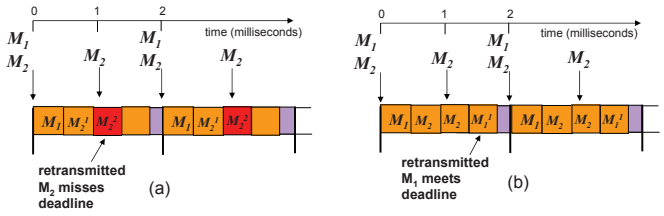

Fig. 4. **Example1:** (a) Retransmitting M_2 leads to deadline miss. (b) Choosing to retransmit M_1 leads to a schedulable solution. Retransmitted instances of the messages are denoted with a superscript, e.g., M_2^1 shows fir t retransmission of M_2.

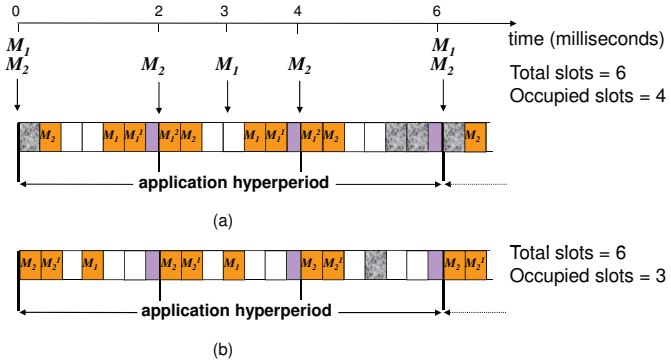

Fig. 5. **Example2:** Although M_1 occurs less frequently, (a) retransmitting M_1 in order to achieve reliability leads to poor bandwidth utilization compared to (b) retransmitting M_2. Slots not assigned to any message are white.

to meet the reliability requirement. With similar probabilistic analysis as above, it can be verifie that (a) retransmitting M_1 twice or (b) M_2 once, the reliability of $\rho = 0.09$ is achieved. As shown in Figure 5, feasible schedule can be constructed for both these cases. However, retransmitting M_1 twice leads to utilizing 4 of the 6 slots (see Figure 5(a)) while retransmitting M_2 once, only 3 of the slots are utilized (see Figure 5(b)). It is important to observe that M_1 occurs less frequently, but choosing to retransmit M_1 ultimately leads to a poor bandwidth utilization.

From the above examples, we observe the following. Firstly, there are combinatorially large number of choices in which messages might be chosen to be retransmitted in order to achieve the reliability goal ρ. Secondly, amongst the set of choices which lead to reliable transmissions, some choices might lead to unfeasible schedule or lead to poor bandwidth. In the following section, we present a CLP formulation which find the optimal number of retransmissions to conserve the bandwidth. However, this formulation is computationally expensive, and in Section V we propose an efficien heuristic.

IV. CLP-based Optimal Approach

In this section, we firs propose an analysis to compute the probability of successful transmissions for the overall communication system. In Section IV-B, we discuss how this global probability analysis is utilized by the optimal CLP-based optimization approach in conjunction with scheduling constraints in order to generate FlexRay schedules.

A. Global Probability Analysis

We are interested in computing the global success probability, GP, i.e., the probability that all instances of all messages

can be successfully transmitted. Note that the system reliability requirement ρ is specifie over a time unit τ, and thus, GP has to be computed over time τ as well. If $GP \geq \rho$, this would imply that the system is reliable. In the following, we give an expression to compute GP and provide its derivation.

Lemma IV.1. *The probability that **all** instances of **all** the messages within a time unit τ can be successfully transmitted at least once, where each message M_i has a probability of failure p_i and is retransmitted k_i times is given by* $GP = \prod_{i=1}^{N} \left(1 - p_i^{k_i+1}\right)^{\frac{\tau}{T_i}}$.

Proof. The probability of **one** instance of a message M_i to encounter faults in each of its transmissions (including the initial transmission and the following k_i retransmissions) is:

$$PF_i(k_i) = p_i^{k_i+1} \tag{1}$$

Following Eq. 1, the probability of **one** instance of the message M_i to have **at least one** transmission without faults is:

$$PF_i'(k_i) = 1 - p_i^{k_i+1} \tag{2}$$

The above calculation considers only one instance of the message M_i. However, as discussed in Section II, the system reliability is define for a time unit τ. During the time interval τ, the message M_i occurs with a period T_i for $\frac{\tau}{T_i}$ times. Extending Eq. 2 to consider **all** instances of the message M_i, the probability to have **at least one** transmission without fault for each instance over a period of time τ is:

$$PS_i(k_i) = \left(1 - p_i^{k_i+1}\right)^{\frac{\tau}{T_i}} \tag{3}$$

We will call $PS_i(k_i)$ as the success probability of message M_i. Finally, considering **all** messages and **all** instances of the messages within τ, Eq. 3 can be extended to obtain the global success probability GP as:

$$GP = \prod_{i=1}^{N} PS_i(k_i) = \prod_{i=1}^{N} \left(1 - p_i^{k_i+1}\right)^{\frac{\tau}{T_i}} \tag{4}$$

□

B. CLP Formulation

Above, we assumed that the number of retransmissions k_i for each message M_i is known when we computed the global probability of successful transmission. In this section, we will compute the number of retransmissions k_i, needed for each message M_i such that (i) $GP \geq \rho$ (i.e., reliability is achieved) and (ii) the schedule is feasible (i.e., the deadlines are satisfied) We formulate this as an optimization problem in CLP where the optimization objective is to conserve the number of utilized slots. As an output of the CLP, we obtain the individual number of retransmissions - k_i for each message. We also obtain the slot assignment for each message M_i.

In the following we describe the constraints of the CLP. The firs set of constraints are related to the reliability requirements, and the second set of constraints are based on the scheduling constraints like FlexRay static segment protocol and deadlines.

Reliability constraint: The firs constraint is that the global success probability (see Lemma IV.1) must be greater than the reliability goal, i.e.,

$$GP = \prod_{i=1}^{N} \left(1 - p_i^{k_i+1}\right)^{\frac{\tau}{T_i}} \geq \rho \tag{5}$$

Bounding retransmissions: We shall now bound the minimum number of retransmissions required for each message that are **must** in order to achieve the goal ρ. Essentially, we fin a lower bound k_i^L on the variable k_i which allows us to write the following constraint:

$$k_i \geq k_i^L \tag{6}$$

Here, k_i^L is define as the smallest value of k_i which satisfie $PS_i(k_i) \geq \rho$, and $PS_i(k_i)$ is as define in Equation 3. Any value of k_i less than k_i^L for a message M_i will imply that the reliability goal can not be satisfied and this can be proven as follows.

Theorem IV.2. *For a message M_j, if k_j is less than k_j^L, the condition $GP \geq \rho$ cannot be satisfied, irrespective of how the k_i values are chosen for the rest of the messages M_i, $i \neq j$ and where k_j^L is defined as the smallest value of k_j which satisfies $PS_j(k_j) = \left(1 - p_j^{k_j+1}\right)^{\frac{\tau}{T_j}} \geq \rho$.*

Proof. In this theorem, we want to prove that if $k_j < k_j^L$, then $GP < \rho$. The theorem says that the k_i values for the rest of the messages M_i, where $i \neq j$, can have any value. This leads to two cases. In the firs case, consider that the k_i values for the rest of the messages M_i are such that:

$$\prod_{i=1, i\neq j}^{N} PS_i(k_i) < \rho \tag{7}$$

Based on our initial assumption $k_j < k_j^L$, we know that the following condition holds true.

$$PS_j(k_j) < \rho \tag{8}$$

We obtain $GP = \prod_{i=1}^{N} PS_i(k_i)$ by multiplying $PS_j(k_j)$ to Eq. 7. It is easy to see that the theorem holds because both $PS_j(k_j)$ and $\prod_{i=1, i\neq j}^{N} PS_i(k_i)$ are sub-unitary numbers and thus, $GP < \rho$. In the second case, that k_i values are such that all the other messages M_i, $i \neq j$ are such that they can satisfy the reliability goal. Thus, we have:

$$\prod_{i=1, i\neq j}^{N} PS_i(k_i) \geq \rho \tag{9}$$

We note that the above product is a product of sub-unitary numbers. Thus,

$$\rho \leq \prod_{i=1, i\neq j}^{N} PS_i(k_i) < 1 \tag{10}$$

Multiplying the Eq. 10 with $PS_j(k_j)$ and comparing with the inequality in Eq. 8, we obtain the following inequality:

$$\rho \times PS_j(k_j) \leq \prod_{i=1}^{N} PS_i(k_i) < PS_j(k_j) < \rho \qquad (11)$$

From Eq. 4, we know that $GP = \prod_{i=1}^{N} PS_i(k_i)$, and thus, Eq. 11 implies $GP < \rho$, hence proving the theorem. □

Above, we bounded the minimum number of retransmissions required in Eq. 6 by computing k_i^L. This bound was derived based on the reliability requirements. Below, we give another constraint which provides an *upper bound* for the total number of retransmissions based on the number of slots available in the FlexRay cycle.

$$\sum_{i=1}^{N}(k_i + 1) \leq NS \qquad (12)$$

Scheduling constraints: We will now introduce the constraints related to scheduling. In order to succinctly represent the constraints, we defin a function called $slot$. The function $slot$ takes a message as a parameter and returns the slot assigned to it. The domain (represented by the set D) for this function is thus, the set of messages including the n_i instances of the message M_i in the hyperperiod H and their retransmissions. This is because all instances and the retransmissions in H must individually meet the deadlines and this must be captured by the constraints. The constraints for the domain D and the function $slot$ can be formally written as follows.

$$\begin{cases} D_i = \{M_i^{j,1}, M_i^{j,2}, \cdots, M_i^{j,k_i+1}\}, \forall 1 \leq j \leq n_i \\ D = D_1 \cup D_2 \cup \cdots \cup D_N \end{cases} \qquad (13)$$

$$\begin{cases} slot : D \rightarrow \{1, 2, \cdots, NC\} \times \{1, 2, \cdots, NS\} \\ slot(M_i^{j,l}) = (c_i^{j,l}, s_i^{j,l}) \end{cases} \qquad (14)$$

In the above equations, NC represents the number of cycles in the hyperperiod H, i.e., $NC = \frac{H}{FC}$). The tuple $(c_i^{j,l}, s_i^{j,l})$ denotes that the message $M_i^{j,l}$ would be transmitted in the slot $s_i^{j,l}$ in the FlexRay cycle, and in the cycle $c_i^{j,k}$ within the hyperperiod H. The superscript j stands for an instance of the message, while $l \in (1, 2, \cdots, k_i + 1)$ represents the lth transmission of the jth instance of M_i.

In order to write the constraints regarding deadlines, we firs introduce a function called t. This function takes a slot as a parameter and returns the absolute time of the slot within the hyperperiod H.

$$\begin{cases} t : \{1, 2, \cdots, NC\} \times \{1, 2, \cdots, NS\} \rightarrow \mathcal{R}^+ \\ t((c, s)_i^{j,l}) = c_i^{j,l} \times FC + s_i^{j,l} \times \frac{ST}{NS} \end{cases} \qquad (15)$$

Let us denote the time instant of production of a particular instance j of a message M_i as a_i^j. From Section II we obtain:

$$a_i^j = O_i + (j-1) \times T_i \qquad (16)$$

The time instants by which messages must finis can be computed as follows:

$$f_i^j = O_i + (j-1) \times T_i + D_i \qquad (17)$$

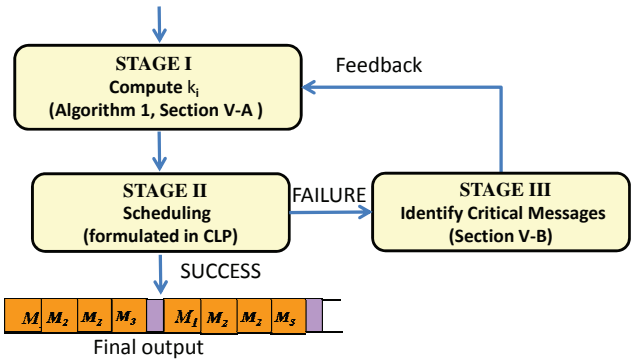

Fig. 6. The overall heuristic algorithm consists of three major stages.

Given the above, the following constraint captures the fact that all the retransmitted messages of an instance must also meet the deadline.

$$a_i^j \leq t(slot(M_i^{j,0}) < t(slot(M_i^{j,1}) < \cdots < t(slot(M_i^{j,k_i}) \leq f_i^j \qquad (18)$$

A slot can be occupied only by **one** retransmission of a particular instance of a particular message M_i, i.e., instance a of message M_i **cannot** share the same slot with instance b of message M_j or with another instance of message M_i.

$$slot(M_{i_1}^{j_1,k_1}) \neq slot(M_{i_2}^{j_2,k_2}) \Leftrightarrow s_{i_1}^{j_1,k_1} \neq s_{i_2}^{j_2,k_2}, \forall i_1 \neq i_2 \qquad (19)$$

According to the FlexRay protocol, different instances of the same message are assigned to the same slot.

$$slot(M_i^{j_1,k}) = slot(M_i^{j_2,k}) \Leftrightarrow s_i^{j_1,k} = s_i^{j_2,k} \qquad (20)$$

Optimization objective: The optimization objective is to minimize the number of used slots in the FlexRay cycle FC. Note that the number of used slots is same as the total number of required transmissions because each message occupies one slot and hence, we have the following objective:

$$minimize : \sum_{i=1}^{N}(k_i + 1) \qquad (21)$$

V. Efficient Heuristic Approach

The CLP formulation described in the section above will return optimal solutions but is computationally expensive and cannot scale to large designs. In this section, we propose an efficien heuristic for the optimization problem. Our proposed heuristic consists of three main stages (see Figure 6). In the firs stage, described in Section V-A the heuristic computes the required number of retransmissions, i.e., k_i values such that the required reliability goal ρ is satisfied This computation is based on global probability analysis that was described in Section IV. Essentially, the heuristic minimizes the required number of k_is in order to achieve ρ. The second stage involves scheduling, i.e., the assignment of the slots to messages. This stage is implemented by invoking the CLP with the scheduling constraints (Eq. 13 to Eq. 20) and the CLP returns a solution which satisfie the constraints, i.e., it generates a feasible schedule. If the CLP returns a negative answer and a feasible schedule cannot be built, our heuristic enters the third stage. This is described in Section V-B where heuristic identifie

Algorithm 1 Procedure "Grouping" to compute k_i for each message M_i

Input: Messages $\{M_1, M_2, \ldots, M_N\}$, with probability of failures p_i, and reliability goal ρ

1: **for** $i \in \{1, 2, \ldots, N\}$ **do**
2: compute k_i^L (see Theorem IV.2)
3: compute $PS_i(k_i^L) \leftarrow \left(1 - p_i^{k_i^L+1}\right)^{\frac{\tau}{T_i}}$ (see Eq.3)
4: **end for**
5: sort $PS_i(k_i^L)$ values in descending order to obtain $PS_1(k_1^L) > PS_2(k_2^L) > \cdots > PS_N(k_N^L)$
6: $U \leftarrow N$
7: **for** $j \leftarrow 1$ to $N-1$ **do**
8: **if** $(\prod_{i=1}^{j} PS_i(k_i^L) \geq \rho) \& (\prod_{i=1}^{j+1} PS_i(k_i^L) < \rho)$ **then**
9: $U \leftarrow j$
10: break;
11: **end if**
12: **end for**
13: **for** $i \in \{1, 2, \ldots, U\}$ **do**
14: $k_i \leftarrow k_i^L$
15: **end for**
16: **if** $U < N$ **then**
17: $\rho' \leftarrow \frac{\rho}{\prod_{i=1}^{U} PS_i(k_i^L)}$
18: recursively call "Grouping" with the remaining messages $\{M_{U+1}, M_{U+2}, \cdots, M_N\}$ and ρ'
19: **else**
20: return from recursion
21: **end if**

bottlenecks, i.e., critical messages which contribute to the deadline violation. The heuristic then enters the firs stage once again, where it computes a different set of k_i values such that a feasible schedule maybe found.

As noted above, our proposed heuristic utilizes CLP for the scheduling. We call this approach *H-CLP*. This approach will scale considerably better than the *Optmal-CLP* approach described in Section IV. However, *H-CLP* will still have exponential complexity and hence, we also implemented another version of *H-CLP* called *H-H*. In *H-H*, we invoke the CLP with a feature called "limited discrepancy search" [2] which allows a fast heuristic search.

A. Computing the Number of Required Retransmissions

The basic idea of the heuristic is to utilize as less retransmissions as possible to achieve reliability. In Section IV-B we computed the lower bounds k_i^L, which are the smallest possible values for each k_i. Essentially, our heuristic attempts to assign these lower bounds to as many messages as possible. For the messages to which the heuristic fails to assign k_i^L, it attempts to assign a value as close to k_i^L as possible.

Algortihm Description: Our procedure to compute k_i for each message M_i is listed in Alg. 1. It is a recursive process and is explained below. The algorithm starts by computing the values of k_i^L and $PS_i(k_i^L)$ for all messages (lines 2 and 3 of Alg. 1). Based on these values there are two scenarios as follows.

Case 1: The firs case is when lower bounds on the retransmissions as specifie by the k_i^L are enough to satisfy the reliability goal ρ, i.e.,

$$\prod_{i=1}^{N} \left(1 - p_i^{k_i^L+1}\right)^{\frac{\tau}{T_i}} \geq \rho \tag{22}$$

In this case, k_i values are set to k_i^L which are the minimum possible values (see Theorem IV.2). If the scheduler fails to build a feasible schedule in this case, this implies that a solution does not exist. If it returns a schedule, we obtain the optimum results.

Case 2: The other case is when the lower bounds on the k_i^L are not enough to meet the reliability constraints, i.e.,

$$\prod_{i=1}^{N} \left(1 - p_i^{k_i^L+1}\right)^{\frac{\tau}{T_i}} < \rho \tag{23}$$

In this case some messages will have to be assigned k_i which are higher than k_i^L so that the reliability goal can be achieved. Here, the heuristic takes a greedy approach to assign k_i^L to as many k_is as possible. Towards this, the computed values of $PS_i(k_i^L)$ in line 3 are sorted and new order obtained is $PS_1(k_1^L) > PS_2(k_2^L) > \cdots > PS_N(k_N^L) \geq \rho$ (line 5). Then the algorithm (lines 7 to 12) find the largest set of messages $\{M_1, M_2, \ldots, M_U\}$ which satisfy the reliability goal. (Theorem I.1 proves that lines 7 to 12 in the algorithm computes the largest group of messages which satisfy the reliability, and thus, the lowest value of k_i^L is assigned to as many k_is as possible.) For these messages, k_i is assigned the values k_i^L. Once this group is built the remaining messages will have to satisfy a new reliability goal as shown in line 17. Thus, the algorithm is called again recursively with messages $\{M_{U+1}, M_{U+2}, \ldots, M_N\}$ and ρ' (line 18).

B. Identifying Critical Messages

Once the values of k_is are computed according to Alg. 1, the scheduler is invoked. In case the scheduler fails to build a schedule, our heuristic attempts to identify messages which might have created bottlenecks. Towards this, we formulate a new CLP problem called CLP-B, which has the scheduling constraints Eq. 13 to Eq. 20. We also introduce new variables $xk_i \in [k_i^L..k_i]$, where k_i stands for the computed value of k using the "Grouping" procedure (Alg. 1). Then we introduce a set of boolean variables y_i as follows.

$$\begin{cases} y_i = 1, & xk_i \neq k_i \\ y_i = 0, & xk_i = k_i \end{cases} \tag{24}$$

Finally, the objective of the CLP-B formulation as follows.

$$minimize : \sum_{i=1}^{N} y_i \tag{25}$$

The intuition is that we invoke the CLP without any reliability constraints, and we let the CLP change the values of k_i such that the scheduling constraints are satisfied However, we keep

Message	Offset (ms)	Period (ms)	Deadline (ms)	Size (bits)
1	0	5	1	32
2	0	15	1	32
3	0	20	1	32
4	0	8	5	32
5	1	16	6	32
6	2	32	5	32
7	2	18	5	32
8	2	36	5	32

TABLE I

EXAMPLE TO SHOW THE FLOW OF THE HEURISTIC.

the number of such changes to the minimum with the above objective function. In other words the CLP-B formulation will try to minimize the number of changes in the previous computed k_i values such that a schedule can be constructed. Then, for the messages $(M_1, M_2, \ldots, M_B)$s whose k_i values were changed by the CLP (i.e., $y_i = 1$) are the ones which created bottleneck. Hence, for these messages, we will fi the k_is to k_i^L. Then, Alg. 1 is invoked again with the set of non-critical messages and ρ' which have to be satisfie by the non-critical messages.

We illustrate the overall fl w of the heuristic with the help of an example. The example consists of 8 message whose characteristics are shown in Table V-B. The FlexRay cycle $FC = 5$ ms has a static segment length $ST = 3$ ms with $NS = 21$ slots. The BER value is assumed to be $BER = 10^{-7}$. The time unit of functionality is $\tau = 1$ hour and assumed to be $\rho = 1 - 10^{-5}$.

Applying Alg. 1, the lower bounds $k_1^L, k_2^L, \ldots, k_8^L$ were computed as $(1, 1, 1, 1, 1, 1, 1, 1)$. The "Grouping" procedure (Alg 1) computes $k_1, k_2, \ldots, k_8$ as $(2, 2, 1, 2, 1, 1, 1, 1)$. It can been seen that messages M_3, M_5, M_6, M_7 and M_8 are assigned the lower bounds. With this set of k_is, however, the scheduling component fails to build a schedule. Then we invoke CLP-B, which find that M_2 is the critical message. At this point, we enter our overall heuristic for the second iteration. Thus, Alg. 1 invoked again with the lower bound $k_2 = 1$ fi ed for message M_2. The algorithm obtains new k_i values as $(2, 1, 1, 2, 2, 1, 1, 1)$. In this case messages M_2, M_3, M_6, M_7 and M_8 are assigned the lower bounds and with these values the scheduling component builds a schedule.

VI. RESULTS

In this section, we discuss experimental results obtained by comparing the 3 methods — *H-H, H-CLP*, and the *Optimal-CLP* on a large set of synthetic test cases and 2 real-life scenarios. The entire framework has been implemented in Constraint Logic Programming using *Prolog*[2]. All the experiments were conducted on a Windows 7 machine running a 4-core Xeon(R) 2.67 GHz processor. In what follows we will firs describe the setup for conducting the synthetic experiments.

A. Experimental Setup

The synthetic test cases were generated by randomly varying the message parameters like periods and deadlines in order to cover a wide range of possible scenarios. The periods were varied between $5ms$ and $40ms$ while the deadlines were varied between $1ms$ and $30ms$. Note that deadlines were generated under the assumption that $D_i \leq T_i$ in order to avoid buffer overfl ws as discussed in Section II. We assume that the

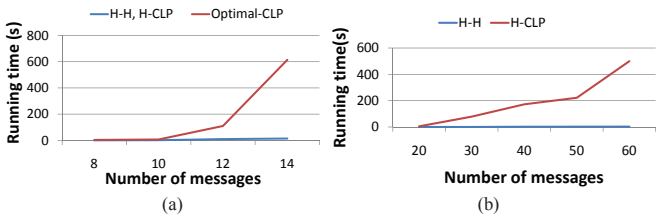

Fig. 7. Running times for test cases with (a) small number of messages (b) large number of messages.

FlexRay communication cycle period is $5ms$ and the static cycle length is $3ms$, following the usual design practice in the industry [17], [18]. We conducted two broad categories of experiments with the synthetic test cases as follows.

1) We generated test cases with 8, 10, 12, and 14 messages and for each set 20 examples were studied. In these experiments we compared all the three proposed methods — *H-H*, *H-CLP* and *Optimal-CLP*.
2) To show the scalability of the heuristics, we also conducted a second category of experiments where test cases with a larger number (from 20 to 60) of messages were generated. Again, each set contained 20 randomly generated test cases.

We assume $BER = 10^{-7}$, the time unit τ, is considered to be an hour over which the reliability goal is define as $\rho = 1 - 10^{-5}$ [13].

B. Results on Synthetic Test Cases

We firs discuss the results obtained on the smaller test cases and then the results obtained on the larger test cases.

Small Test Cases: In total $20 \times 4 = 80$ experiments were conducted for messages $8, 10, 12$ and 14. The average running times of the conducted experiments with a small number of messages is presented in Figure 7(a). As seen in the figure the time required by the optimization problem *Optimal-CLP* to fin the optimal solution grows exponentially with the number of messages while both the heuristics *H-H* and *H-CLP* ran to completion in a significantl smaller amount of time.

Apart from the running times, we would also like to note that every time the *Optimal-CLP* reported that a test case has no solution, the heuristics (*H-H, H-CLP*) also reported the same. This buttresses our claim that the heuristic will never give unsafe answers. On the other hand, a solution might exist but the heuristic may fail to fin a solution. Out of the 80 test cases, the heuristics *H-H* and *H-CLP* failed only in 5 cases, i.e., reported there is no solution, while the *Optimal-CLP* found a solution. Thus, heuristic and *Optimal-CLP* returned same results for over 93% of the test cases. We note that in all these cases where the heuristic was successful, it also obtained the same optimization cost (i.e., bandwidth utilization) as the *Optimal-CLP*.

Large Test Cases: In the above, we compared the results of the heuristic with the *Optimal-CLP* for test cases containing upto 14 messages. This is because beyond 14, the *Optimal-CLP* did not scale at all and could not provide a solution in a reasonable amount of time (less then 1 hour). To compare

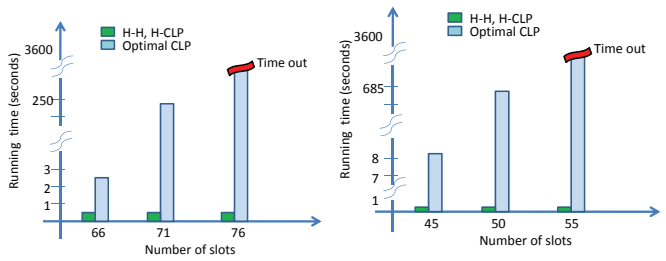

Fig. 8. Running times for brake-by-wire messages with BER values (a) 10^{-7} and (b) 10^{-9}.

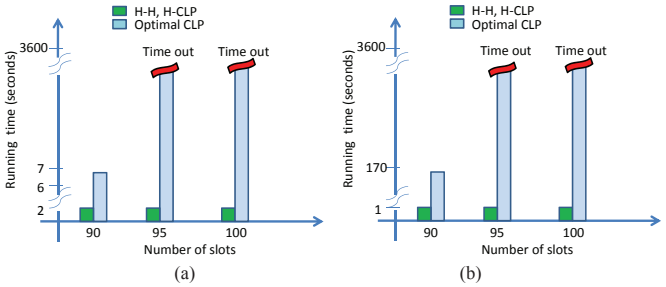

Fig. 9. Running times for ACC messages with BER values (a) 10^{-7} and (b) 10^{-9}.

the scalability of *H-H*, and *H-CLP* we ran them on test cases with upto 100 messages. As seen from Fig. 7(b), the *H-CLP* failed to scale beyond 60 messages.

C. Real-life Examples

We considered two real-life scenarios. First, we considered a brake-by-wire application with 20 messages. The message characteristics are shown in Table II. The FlexRay parameters are $FC = 1$ ms, $ST = 0.75$ ms, and $\tau = 1$ hour of functionality and $\rho = 1 - 10^{-5}$. We conducted two sets of experiments with different BER values — 10^{-7} and 10^{-9}. For each set, we varied the number of slots in the static segment at intervals of 5. The running times are plotted in Fig. 8. We also studied another scenario, where the messages of the brake-by-wire application are sharing the bus with the messages of an Adaptive Cruise Controller (ACC). We conducted similar experiments as in the firs scenario and the running times are plotted in Fig. 9. From both the plots (Fig. 8 and Fig. 9), it may noticed that the *Optimal-CLP* does not scale for real-life example, while the heuristics returns a feasible schedule within few seconds. Out of the 12 experiments in total, the *Optimal-CLP* could not return an answer in 6 cases within one hour. These are reported as time outs in the plots. It should be mentioned here that the 6 cases where the results between the heuristic and the *Optimal-CLP* could be compared, the heuristic returned a feasible schedule if the *Optimal-CLP* returned a feasible schedule. Moreover, the heuristic always obtained the same optimization cost (i.e., bandwidth utilization) as the *Optimal-CLP*.

VII. Generalization of the Proposed Scheme

Our proposed scheme assumed that all the messages together must achieve a global reliability. However, our technique is quite general and it is not restricted to this formulation. For instance, certain messages might be more

Message	Offset (ms)	Period (ms)	Deadline (ms)	Size (bits)
1	0.764	8	8	1024
2	0.765	8	8	1024
3	0.105	1	1	1280
4	0.530	1	1	896
5	0.120	1	1	1280
6	0.565	1	1	896
7	0.160	1	1	1536
8	0.200	1	1	1536
9	0.450	1	1	384
10	0.180	8	8	256
11	0.580	8	8	1280
12	0.199	8	8	256
13	0.599	8	8	1280
14	0.201	8	8	256
15	0.601	8	8	1536
16	0.204	8	8	256
17	0.604	8	8	1280
18	0.426	8	8	896
19	0.426	8	8	1536
20	0.349	8	8	960

TABLE II

Brake-by-wire Message Parameters

Message	Offset (ms)	Period (ms)	Deadline (ms)	Size (bits)
1	0.426	16	16	1024
2	0.349	16	16	1024
3	0.426	24	24	1024
4	0.349	24	24	1024
5	0.580	32	32	1280
6	0.199	32	32	256
7	0.599	32	32	1280
8	0.201	32	32	256

TABLE III

Adaptive Cruise Controller (ACC) Message Parameters

safety-critical than others, and designers might specify that such a subset of messages must achieve a higher reliability. Formally, it implies that messages are divided into R sets $A_1, A_2, \ldots, A_R$, where messages in each set must meet reliability goal $\rho_1, \rho_2, \ldots, \rho_R$, respectively. The *Optimal-CLP* formulation in Section IV can be extended by introducing the following constraint for each of these sets:

$$GP_r = \prod_{i=1}^{|A_r|} \left(1 - p_i^{k_i+1}\right)^{\frac{\tau}{T_i}} \geq \rho_r, \forall r \in 1, 2, \ldots, R \quad (26)$$

Note that our heuristic can also be seamlessly extended to handle this formulation. Towards this, the overall heuristic approach as shown in Fig. 6 remains same. The only modificatio is in stage I, where the "Grouping" procedure must now be invoked individually for each subset $A_1, A_2, \ldots, A_R$. Thereafter, the second and third stages of the heuristic remain the same. With the help of this discussion, we emphasize that the proposed method can be adapted with ease to many different problem scenarios, one of which was explained here.

VIII. Conclusions and Future Work

In this paper, we proposed techniques towards synthesizing reliable schedules on the FlexRay bus. Towards this, we introduced an analysis scheme which computes the maximum number of required retransmissions, such that the overall reliability is guaranteed. Since our focus was on the static segment, for each required retransmission a slot is reserved in the static segment. It will be interesting to see if similar formal probability analysis techniques as proposed in this paper can

be developed for the dynamic segment of FlexRay, where unlike the static segment, messages can be scheduled to be retransmitted when an acknowledgment is received.

The FlexRay specificatio [7] provides what is called "scalable" fault-tolerance. This feature allows dual channel communication where messages can be replicated over the second channel for fault-tolerance. However, using the second channel is optional and it also incurs an additional cost. Computing the reliability that can be achieved by using the dual channel, and evaluating the associated cost trade-offs is another interesting open question.

APPENDIX I
PROOF RELATED TO ALG. 1 IN SECTION V

In Alg. 1, line 7 to 12 compute a group $M_1, M_2, \ldots, M_U$ such that the product of their success probabilities is greater than the reliability goal. For these messages, the algorithm assigns the lower bounds, k_i^L to k_i. Here, we will show that the selection of the group $M_1, M_2, \ldots, M_U$ is not done in an adhoc manner by proving that the Alg. 1 computes the largest group $M_1, M_2, \ldots, M_U$ that satisfie ρ thereby allowing us to assign the lower bound k_i^L to as many messages as possible in each step. In essence, this implies that although the algorithm makes a greedy choice at each step, this choice is an optimum at this step. Formally,

Theorem I.1. *If*

$$\begin{cases} 1 : \prod_{i=1}^{U} PS_i(k_i^L) \geq \rho \\ 2 : \prod_{i=1}^{U+1} PS_i(k_i^L) < \rho \end{cases} \tag{27}$$

then the set $A = \{PS_1, PS_2, \cdots, PS_U\}$ is the largest subset of success probabilities that can satisfy the reliability goal ρ where the lower bounds k_i^L are assigned to k_i for each message in the set A.

Proof. Let q be number of arbitrary items that are removed from A and let v be the number of arbitrary items ($v > q$), from the rest of the PS values, that will be added to the A set. We will prove that the set A is the largest subset by showing that adding any $v - q$ new elements, the ρ cannot be satisfied Note that it is sufficien to prove that the items $\{PS_1, PS_2, \cdots, PS_q\}$ are the items to be replaced because they have the highest probabilities. We also note that the items that will be added $\{PS_{a_1}, PS_{a_2}, \cdots, PS_{a_v}\}$ will have the property $PS_U \geq PS_{a_1} \geq PS_{a_2} \geq \cdots \geq PS_{a_v}$. It follows from the assumptions in the theorem that,

$$PS_1 \times PS_2 \times \cdots \times PS_U \times PS_{a_1} \leq \rho \tag{28}$$
$$PS_1 \times PS_2 \times \cdots \times PS_U \times PS_{a_2} \leq \rho \tag{29}$$
$$\cdots$$
$$PS_1 \times PS_2 \times \cdots \times PS_U \times PS_{a_v} \leq \rho$$

In Eq.28, if PS_1 is replaced by PS_{a_2},

$$\begin{aligned} &PS_{a_2} \times PS_2 \cdots \times PS_U \times PS_{a_1} \\ &\leq PS_1 \times PS_2 \cdots \times PS_U \times PS_{a_1} \\ &\leq \rho \end{aligned} \tag{30}$$

In Eq.30, if PS_2 is replaced by PS_{a_3},

$$\begin{aligned} &PS_{a_2} \times PS_{a_3} \cdots \times PS_U \times PS_{a_1} \\ &\leq PS_{a_2} \times PS_2 \cdots \times PS_U \times PS_{a_1} \\ &\leq \rho \end{aligned} \tag{31}$$

$\ldots$ PS_q is replaced by $PS_{a_{q+1}}, \cdots$

$$\begin{aligned} &PS_{a_2} \times PS_{a_3} \cdots \times PS_{a_{q+1}} \times PS_{q+1} \cdots \times PS_U \times PS_{a_1} \\ &\leq PS_{a_2} \times PS_{a_3} \cdots \times PS_q \times PS_{q+1} \cdots \times PS_U \times PS_{a_1} \\ &\leq \rho \end{aligned} \tag{32}$$

From Eq. 32, by adding the remaining (v - q) PS values: $Q \times PS_{a_{q+2}} \times PS_{a_{q+3}} \times \cdots \times PS_{a_v} \leq \rho \Rightarrow S_{U+v-q} = \{PS_{q+1}, PS_{q+2}, \cdots, PS_U, PS_{a_1}, PS_{a_2}, \cdots, PS_{a_v}\}$ does not satisfy ρ. Thus, it follows that A is the largest subset of PS values that satisfy the reliability goal ρ. □

There might be many groups of size U which form the largest group, but the above construction will fin the group which has the highest success probability. Let us say there are two groups A_1 and A_2 with U elements and each set having GP_1 and GP_2 as global success probabilities. Formally,

$$\begin{cases} A_1 = \{PS_{a_1}, PS_{a_2}, \cdots, PS_{a_U}\}, GP_1 = \prod_{i=1}^{U} PS_{a_i} \\ A_2 = \{PS_{b_1}, PS_{b_2}, \cdots, PS_{b_U}\}, GP_2 = \prod_{i=1}^{U} PS_{b_i} \end{cases} \tag{33}$$

If A_1 is constructed by Alg. 1, then $GP_1 > GP_2$. This is a corrollary of the above theorem and the proof is omitted. This fact implies that by choosing A_1 amongst all possible groups, the Alg. 1 imposes the least possible burden on the rest of the messages for which lower bounds could not be choosen.

REFERENCES

[1] A. Albert. Comparison of event-triggered and time-triggered concepts with regard to distributed control systems. In *Embedded World*, Nürnberg, Germany, 2004. www.semiconductors.bosch.de/pdf/embedded_world_04_albert.pdf.

[2] K. R. Apt and M. G. Thiran. *Constraint Logic Programming using ECL^iPS^e*. Cambridge University Press, 2007.

[3] CAN Specification Ver 2.0, Robert Bosch GmbH. www.semiconductors.bosch.de/pdf/can2spec.pdf, 1991.

[4] F. Corno, M. Sonza Reorda, S. Tosato, and F. Esposito. Evaluating the effects of transient faults on vehicle dynamic performance in automotive systems. In *International Test Conference*, 2004.

[5] BMW brake system relies on FlexRay. http://www.automotivedesignline.com/news/218501196, July 2009.

[6] Elektrobit Austria GmbH. www.elektrobit.com.

[7] The FlexRay Communications System Specification , Ver. 2.1. www.flexray.com.

[8] B. Gaujal and N. Navet. Maximizing the robustness of TDMA networks with applications to TTP/C. *Real-Time Systems*, 31(1-3):5–31, 2005.

[9] A. Ghosal, H. Zeng, M. Di Natale, and Y. Ben-Haim. Configurin the communication on FlexRay: the case of the static segment. In *DATE*, 2010.

[10] P. Peti H. Kopetz, R. Obermaisser and N. Suri. From a federated to an integrated architecture for dependable embedded systems. Technical Report 22, Technische Universität Wien, 2004.

[11] A. Hagiescu, U. D. Bordoloi, S. Chakraborty, P Sampath, P. V. V. Ganesan, and S. Ramesh. Performance analysis of FlexRay-based ECU networks. In *DAC*, 2007.

[12] Functional safety of electrical/electronic/ programmable electronic safety-related systems, IEC61508. http://www.iec.ch/.

[13] V. Izosimov. *Scheduling and Optimization of Fault-Tolerant Distributed Embedded Systems*. PhD thesis, Linkping University, 2009.

[14] S Kanajan and J. Abell. Sensitivity analysis on FlexRay dynamic segment design parameters. In *International Conference on Systems and Networks Communications*, 2009.

[15] H. Kopetz and G. Bauer. The time-triggered architecture. *Proceedings of the IEEE*, 91(1):112–126, 2003.

[16] W. Li, M. D. Natale, W. Zheng, P. Giusto, A. L. Sangiovanni-Vincentelli, and S. A. Seshia. Optimizations of an application-level protocol for enhanced dependability in FlexRay. In *DATE*, 2009.

[17] M. Lukasiewycz, M. Glaß, J. Teich, and P. Milbredt. FlexRay schedule optimization of the static segment. In *CODES+ISSS '09*, 2009.

[18] G. Mathieu, H. Lionel, and N. Nicolas. Configurin the communication on FlexRay: the case of the static segment. In *European Congress Embedded Real Time Software*, 2008.

[19] T. Pop, P. Pop, P. Eles, Z. Peng, and A. Andrei. Timing analysis of the FlexRay communication protocol. In *ECRTS*, 2006.

[20] Vector. www.vector.com.

[21] E. Zanoni and P. Pavan. Improving the reliability and safety of automotive electronics. *IEEE Micro*, 13(1), 1993.

[22] H. Zeng, W. Zheng, A. Ghosal, P. Giusto, A. Sangiovanni-Vincentelli, and M. Di Natale. Scheduling and mapping in an incremental design methodology for distributed real-time embedded systems. In *DAC*, 2009.