An Experimental Study of the FlexRay Dynamic Segment

Klaus Schmidt [*] **Ece G. Schmidt** [**] **Ali Demirci** [**]
Emrah Yürüklü [***] **Utku Karakaya** [***]

[*] *Department of Electronic and Communication Engineering, Çankaya University, Ankara (e-mail: schmidt@cankaya.edu.tr)*
[**] *Department of Electrical and Electronics Engineering, Middle East Technical University, Ankara (e-mail: {eguran,e119723}@metu.edu.tr)*
[***] *TOFAŞ Türk Otomobil Fabrikası A.Ş., BURSA (e-mail:{emrah.yuruklu,utku.karakaya}@tofas.com.tr)*

Abstract: It is expected that the time-triggered FlexRay bus will replace the event-triggered Controller Area Network (CAN) for the high-speed in-vehicle communication in future automobiles. To this end, FlexRay provides a *static segment* for the transmission of periodic messages and a *dynamic segment* that is suitable for exchanging event-based (sporadic) messages. In this paper, we experimentally evaluate the operation of the FlexRay dynamic segment. In particular, we study how the maximum and average message delays are affected if the *length of the dynamic segment*, the message *payload*, the *utilization* of the dynamic segment and the priority *assignment* changes. Our experiments are carried out on a FlexRay network with 6 nodes.

Keywords: FlexRay, in-vehicle networks, sporadic communication, experimental evaluation.

1. INTRODUCTION

In today's cars, a great variety of electronic devices including microcontrollers, sensors, and actuators, are used to replace mechanical and hydraulic components or to enhance the driving safety and comfort. A large number of up to 100 such electronic control units (ECUs) are implemented in modern cars (Albert, 2004; Navet et al., 2005; Lukasiewycz et al., 2009). These ECUs exchange messages among each other over a communication network to support the execution of their tasks.

The messages exchanged via in-vehicle networks are either periodic, i.e., they contain periodically generated signal data, or sporadic messages, i.e., they are generated based on asynchronous event occurrences. The recently developed FlexRay protocol (FLE, 2004) supports both message types and is expected to become the de-facto standard for high-speed in-vehicle communication in the future (Navet et al., 2005; Makowitz and Temple, 2006). FlexRay has a high bandwidth of 10 Mbit/s and is suitable for time-critical applications such as x-by-wire and advanced vehicle dynamics control systems. Its cyclic operation combines the advantages of time-triggered and event-triggered communication. In each FlexRay cycle, the static segment (SS) employs the idea of time division multiple access (TDMA) to enable the transmission of periodic messages in unique static slots, while sporadic messages can be sent in dynamic slots in the dynamic segment (DS) that is based on flexible TDMA (FTDMA).

In this paper, we study basic properties of the event-triggerd communication in the FlexRay dynamic segment. Based on a message set that is constructed according to a benchmark set of the *Society for Automotive Engineers* (SAE) (SAE, 1993), we experimentally investigate the effect of varying the length of the dynamic segment, the utilization, the message payload and the message priority assignment. All experiments are conducted on a network with 6 FlexRay nodes that implement the generation and transmission of their sporadic messages, while all message traffic is recorded by a bus analysis tool. It has to be noted that experiments on the FlexRay dynamic segment also have been carried out in (Nielsen et al., 2007; Jung et al., 2008). While (Nielsen et al., 2007) performs measurements in order to validate a Markov chain model of the FlexRay dynamic segment, (Jung et al., 2008) determines message delays for a particular message priority assignment. Hence, both papers consider specific topics that are different from our work that focuses on basic properties of the dynamic segment operation.

The paper is organized as follows. Section 2 gives a detailed description of the FlexRay protocol with its relevant configuration parameters. Our experimental setup including the investigated performance metrics is outlined in Section 3, while the experimental results are presented in Section 4. Section 5 summarizes and discusses the main findings of the paper.

2. THE FLEXRAY PROTOCOL

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

2.1 Description of the FlexRay Cycle

The operation of each FlexRay channel is based on a fixed-duration, repeatedly executed *FlexRay cycle* (FC)

(FLE, 2004). As depicted in Fig. 1, the FC comprises the *static segment* (SS), the *dynamic segment* (DS), the *symbol window* (SW), and the *network idle time* (NIT). The SS is designed for the periodic transmission of real-time data, while the DS supports the transmission of low-priority data and event-triggered (sporadic) real-time data. The optional SW allows to send certain symbols and the NIT provides time for protocol-related computations such as clock correction. The successive FCs are counted by the protocol internal variable `vCycleCounter` that ranges from 0 to 63 and is equal among all nodes on a FlexRay network. The basic time unit of the protocol operation is the *macrotick* (MT) with a duration of `gdMacrotick` that can be configured between 1 μs and 6 μs. Accordingly, the fixed duration of each FC can be expressed as `gdCycle` = `gMacroPerCycle` · `gdMacrotick`, where `gMacroPerCycle` denotes the number of MTs per FC (between 10 and 16000).

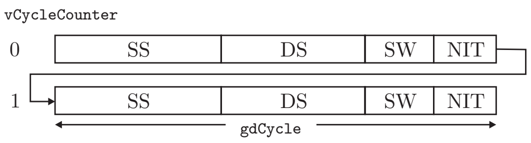

Fig. 1. FlexRay cycle description.

The operation of the FlexRay SS is similar to the time-triggered protocol (TTP) (TTP, 2003; Kopetz and Bauer, 2003) and employs the TDMA approach. In each FC, the SS provides `gNumberOfStaticSlots` *static slots*, where each static slot is uniquely assigned to one of the nodes on the FlexRay network as illustrated in Fig. 2. All static slots have a fixed duration of `gdStaticSlot` (between 4 and 661 MT) such that the duration of the SS can be expressed as `gNumberOfStaticSlots` · `gdStaticSloc` · `gdMacrotick`. Since the investigation of the SS is not in the scope of this paper, we refrain from a more detailed description.

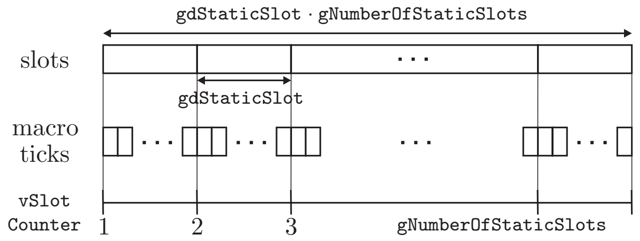

Fig. 2. FlexRay static segment (SS).

The DS is similar to ByteFlight (Berwanger et al., 1999) and employs the flexible TDMA (FTDMA) approach. Its basic operation is shown in Fig. 3. The smallest time unit in the DS is the *minislot* (MS) with a duration of `gdMinislot` (between 2 and 63 MT). The DS comprises a maximum number of `gNumberOfMinislots` (between 0 and 7986) MSs. Frames are transmitted within *dynamic slots* that are superimposed on the minislots. For each channel, the variable `vSlotCounter` captures the ID of the current dynamic slot starting from the value `vSlotCounter` = `gNumberOfStaticSlots`+1 =: x. In each dynamic slot, a frame with the corresponding ID is transmitted if present. In that case, the duration of the dynamic slot is determined by the length of the transmitted frame. Otherwise, the duration of the dynamic slot is one MS. The message ransmission in the DS is completed if either the internal minislot counter `zMinislot` reaches the latest transmission instant `pLatestTx` or `vSlotCounter` exceeds the maximum number of transmission slots `cSlotIDMax`. Note that the transmission on both channels is independent in the DS.

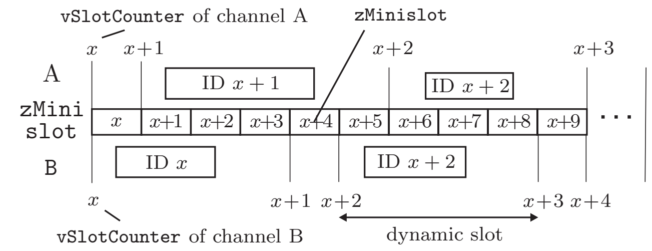

Fig. 3. FlexRay dynamic segment (DS).

Precisely, the actual frame transmission in each dynamic slot happens between the *action points* of two MSs as depicted in Fig. 4 within the *transmission phase* of the dynamic slot. Such action points occur a pre-configured number of `gdMinislotActionPointOffset` MT (between 0 and 31) after the start of the MS. In addition, the dynamic slot includes an *idle phase* with a duration of `gdDynamicSlotIdlePhase` minislots (between 0 and 2).

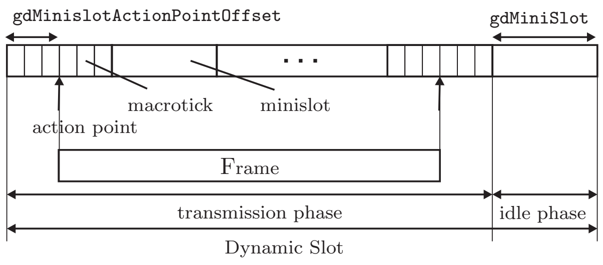

Fig. 4. FlexRay dynamic slot.

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

2.2 Messages

FlexRay frames consist of three main segments, the *header*, the *payload* and the *trailer*. Besides certain flags, the FlexRay header comprises the frame ID that specifies the ID of the dynamic slot to be used, the length of the payload carried by the frame, a CRC for the header and the current value of `vCycleCounter`. The payload amounts to 0−254 B and contains the actual data transmitted in the frame, while the last 3 B of the frame are reserved for the CRC.

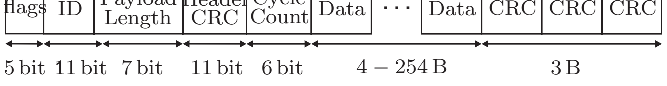

Fig. 5. FlexRay frame format.

Together, the frame duration in `gdBit` including protocol specific frame start and end durations evaluates as

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

where $\texttt{PayloadLength}$ is given in multiples of 2 B words (FLE, 2004). In addition, the number of MSs needed for the transmission of each frame is

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

With the characterization of the message length in (1), it is now possible to compute the minimum length of the dynamic segment that potentially gives each message in a set of n messages the chance to be transmitted. Note that the last dynamic slot that can be used for message transmission is determined by

$$\texttt{pLatestTx} = \texttt{gNumberOfMinislots} - \texttt{aMinislotPerDynamicFrame}, \quad (3)$$

where $\texttt{aMinislotPerDynamicFrame}$ denotes the number of MSs needed for the longest frame (FLE, 2004). Hence, the requirement that $\texttt{pLatestTx} \geq n$ leads to

$$\texttt{gNumberOfMinislots} \geq n + \texttt{aMinislotPerDynamicFrame}. \quad (4)$$

The objective of this paper is the study of event-based real-time message frames that are transmitted in the Flexray DS. In addition to the protocol specific description, we characterize the timing properties of sporadic messages following the lines of related work in (SAE, 1993; Tindell et al., 2000; Pop et al., 2008; Schmidt and Schmidt, 2007, 2009). For each message, there is a *deadline* which is the largest tolerable time interval between the generation and the completed 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 (Pop et al., 2008). In addition, the recurrence of a message is described by its minimum inter-arrival time denoted as *period*, which characterizes the minimum time interval between two consecutive message generations.

2.3 Software Architecture

Considering the software architecture, each FlexRay node consists of a host and a communication controller (CC) that are connected by a controller-host interface (CHI), as depicted in Fig. 6. Here, the CHI serves as a buffer between the host and the CC. The host processes incoming messages and generates outgoing messages, while the CC independently implements the FlexRay protocol services.

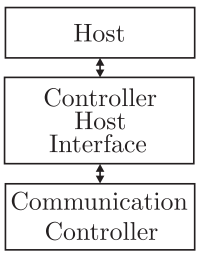

Fig. 6. FlexRay software architecture.

3. EXPERIMENTAL SETUP

3.1 Hardware

Our experiments are performed using 6 FlexRay nodes that are connected to a bus analysis tool in order to record relevant information about the messages transmitted on the network. The experimental setup is illustrated in Fig. 7. Here, the FlexRay nodes are realized by the evaluation boards SK-91465X-100MPC (Fuj, 2009). Amongst others, these evaluation boards comprise a 32-bit Flash micro-controller unit (MCU) MB91F465XA that supports the FlexRay protocol operations. In particular, the CC for the channels A and B is implemented by two Bosch E-Ray type IP-modules (Bos, 2009) and the physical layer of the FlexRay bus is realized by AMS8221B transceivers. As the bus analysis tool, the Flexcard Cyclone II SE (Fle, 2009) is used. In our setup, it receives all messages that are transmitted on the FlexRay bus and records information about the message transmissions including accurate time stamping. Note that the results obtained in the subsequent experiments apply to any hardware that complies with the FlexRay standard (FLE, 2004).

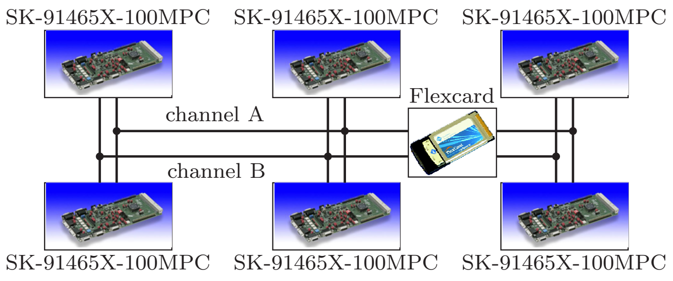

Fig. 7. Experimental setup with 6 FlexRay nodes.

3.2 Configuration Parameters

In order to perform experiments on the FlexRay bus, the configuration parameters introduced in Section 2.1 have to be set. Throughout our study, we keep the values of the parameters in Table 1 fixed. They are all realistic in the sense that they are all similar to the parameters of preliminary implementations of the FlexRay protocol such as (Schedl, 2007). Furthermore, it has to be noted that we focus on a single channel for message transmissions since the operation of the two FlexRay channels is independent.

Configuration parameter	Value
$\texttt{gdMacrotick}[\mu\text{s}]$	1
$\texttt{gdMacroPerCycle}$	5 000
$\texttt{gdCycle}[\mu\text{s}]$	5 000
$\texttt{gdStaticSlot}[\text{MT}]$	31
$\texttt{gNumberOfStaticSlots}[\text{MT}]$	96
$\texttt{gdMinislot}[\text{MT}]$	6
$\texttt{gdMinislotActionPointOffset}[\text{MT}]$	2
$\texttt{gdDynamicSlotIdlePhase}[\text{MS}]$	1
$\texttt{gdSymbolWindow}[\text{MT}]$	0

Table 1. Fixed configuration parameters.

In our experiments, we vary $\texttt{gNumberOfMiniSlots}$ in order to adjust the length of the dynamic segment. In this respect, we compensate a modification of the dynamic segment length by also changing the parameter $\texttt{gdNIT}$ so as to keep the FC duration of 5 000 μs constant.

3.3 Message Generation

The *timing properties* of the sporadic messages used in our experiments are taken from a message set that is published by the *Society for Automotive Engineers* (SAE, 1993). Hence, we consider sporadic messages with a period (minimum inter-arrival time) and a deadline of 50 ms. We use a number of 36 sporadic messages with the IDs 97 to 132 respecting the number of 96 static slots in the SS. Accordingly, we set the maximum number of transmission slots to `cSlotIDMax` = 132. The message parameter that varies in our experiments is the message payload. The particular message sets are listed with the respective experiment in Section 4.

The sporadic messages are distributed equally to the 6 FlexRay nodes. They are generated by an interrupt-based routine that runs as the host process of each node. To this end, the 16-bit *reload timers* of the nodes' MCU with a clock frequency of 125 kHz are used to count down the time until the next generation of each message. In order to achieve the event-based message generation, the next generation time for each message is computed using the *rand*() function of the C standard library. Here, the lower bound for the equally distributed generation time is determined by the message period of 50 ms while the upper bound is set to the maximum possible value of 2^{16}/125kHz= 524.29 ms if not specified otherwise.

3.4 Performance Metrics

In this work, we investigate the dependency among different performance metrics that are defined based on the properties and transmission characteristics of the respective messages. The *dynamic segment ratio dynRatio* denotes the fraction of the FC that is occupied by the dynamic segment.

$$dynRatio = \frac{\texttt{gdMinislot} \cdot \texttt{gNumberOfMinislots}}{\texttt{gMacroPerCycle}} \quad (5)$$

and the *dynamic segment utilization dynUtil* describes the fraction of the dynamic seqment that is actually used for sporadic message transmissions.

$$dynUtil = \frac{messageTime}{expTime} \cdot \frac{1}{dynRatio}. \quad (6)$$

Here, *expTime* is the overall time of each experiment, while *messageTime* is the sum of the sporadic message transmission times observed during each experiment. Moreover, we study the *deadline miss ration DMR*

$$DMR = \frac{\text{\# messages with missed deadlines}}{\text{\# messages transmitted}} \quad (7)$$

and the *delay* for each message, i.e., the difference between the generation time and the time when the message is fully transmitted. In this context, it has to be noted that the message delay strongly depends on the cyclic operation of FlexRay as illustrated in Fig. 8. In the favorable case, a message is generated before its dynamic slot and there is no higher-priority message that is transmitted before (see Fig. 8 (a)). However, it is also possible that a message is generated after its dynamic slot while there are multiple higher-priority messages that block the transmission for several FCs as outlined in Fig. 8 (b). Here, the frame with the ID 6 is delayed by more than 4 FCs due to the transmission of higher-priority messages.

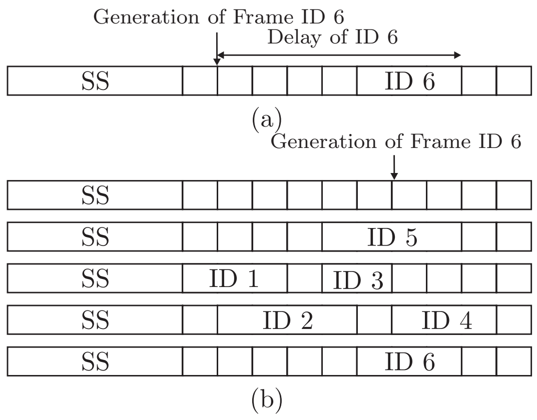

Fig. 8. Message delay.

4. EXPERIMENTAL RESULTS

The goal of this section is the quantitative analysis of basic properties of the FleyRay dynamic segment. In particular, we investigate the effect of varying the length and the utilization of the dynamic segment, using messages with different payloads and assigning priorities to messages with different payloads. All our experiments are carried out using the setup described in Section 3. Measurements are taken over an interval of 120 s and each experiment is carried out 3 times with different seeds for the random message generation. Then, the maximum and average message delays are evaluated over the whole set of measurements.

4.1 Varying the Length of Dynamic Segment

In this experiment, we use messages with a payload of 64 B which corresponds to 14 MS according to (1). Noting that the minimum length of the dynamic segment is determined by `gdNumberOfMinislots` $= 36 + 14 = 50$, we choose dynamic segments with 50, 100 and 200 MSs for our evaluation. The values for the corresponding performance metrics according to (5) to (7) are listed in Table 2 and the maximum delay measurements are shown in Fig. 9.

`gdNumberOfMinislots`	*dynRatio*	*dynUtil*	*DMR*
50	0.06	0.18	0.0021
100	0.12	0.09	0.0
200	0.24	0.045	0.0

Table 2. Performance metrics for varying DS length.

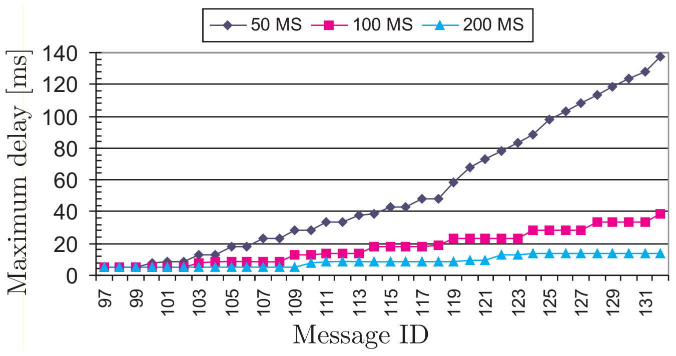

Fig. 9. Maximum delay for varying DS length.

The measurements show that the message delays increase monotonically with a decreasing message priority. Here, it is interesting to note that the differences of the maximum delays of different messages are in multiples of 5 ms, which reflects the cyclic operation of FlexRay. It can further be observed that although the utilization of the dynamic segment is small, a short dynamic segment can lead to deadline misses (delay measurements larger than 50 ms). The reason for this effect is that only a small number of messages fit in the DS in each FC. Hence, in the worst case, a small number of higher-priority messages are sufficient to block each FC for lower-priority messages whose transmission is delayed. Finally, it is clear that a larger dynamic segment leads to smaller maximum delays.

In addition, Fig. 10 shows the average delay values observed in this experiment. If the dynamic segment is long enough (100 MS and 200 MS), most of the messages can be sent either in the FC where they are generated or in the subsequent FC resulting in an average delay of about 2.5 ms. Only lower-priority messages can be blocked for multiple FCs if the dynamic segment is too short (50 MS).

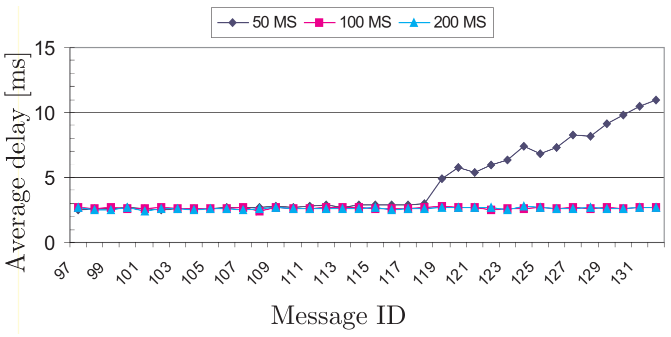

Fig. 10. Average delay for varying DS length.

4.2 Varying the Utilization

The utilization in the previous section corresponds to the sporadic message generation with the lower bound 50 ms and the upper bound 524.29 ms as described in Section 3.3. In order to validate our result, we now perform the same experiment with the worst-case utilization, i.e., the inter-arrival time for each sporadic message always equals its period of 50 ms (see Table 3).

gdNumberOfMinislots	*dynRatio*	*dynUtil*	*DMR*
50	0.06	1.0	0.39
100	0.12	0.5	0.0
200	0.24	0.25	0.0

Table 3. Performance metrics for varying utilization.

It is interesting to note that the delay measurements for the case of 100 MS and 200 MS do not change compared to the previous experiment although the utilization is considerably increased. This can be explained by noting that the maximum delay for each message occurs in case of a worst-case arrival of the higher-priority messages, whereby such worst-case arrival does not depend on the utilization. If the utilization is higher, it is only the case that the worst-case arrival happens more frequently. The latter statement can be verified by the measurement for

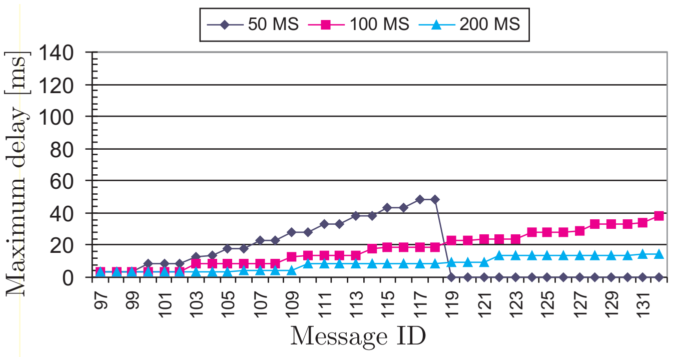

Fig. 11. Maximum delay for varying utilization.

50 MS in the DS. Here, the worst-case arrival always happens such that no messages with IDs larger than 118 can be transmitted. Moreover, it turns out that the average delays observed for all messages in this experiment are almost identical to the maximum delays in Fig. 11.

4.3 Varying the Message Payload Length

In this experiment, we increase the message payload from 64 B (14 MS) to 128 B (24 MS) and 254 B (45 MS) while using 100 MS in the DS and keeping the utilization fixed to $dynUtil = 0.09$.

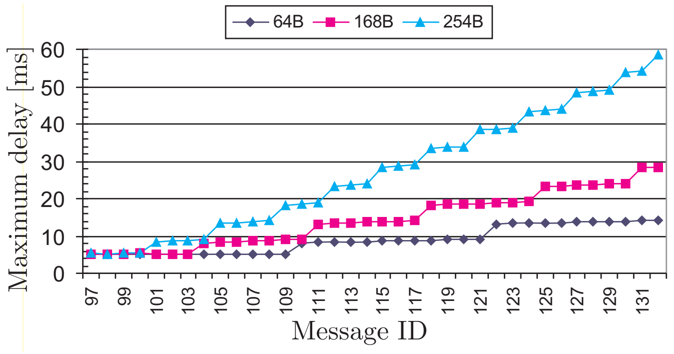

Fig. 12. Maximum delay for varying message payload.

The measurements in Fig. 12 indicate that larger messages potentially lead to larger maximum delays. This observation is due to the fact that, in the worst case, a certain number of larger messages can block more FCs for lower-priority messages than the same number of smaller messages. Hence, the delay of lower-priority messages is increased. In the experiment, this increase even leads to deadline misses for the messages with 254 B payload although the utilization of the dynamic segment is very low.

4.4 Priority Assignment with respect to Payload Length

In each of the previous experiments, messages with identical payload sizes are used. We now address the case where messages have different payloads and study different possible priority assignments. In particular, we choose 9 messages with payloads of 16 B, 64 B, 128 B and 254 B respectively. In the first case, we assign the priorities according to an increasing payload such that the messages with a payload of 16 B have the highest priority. In the second case, we study the reverse assignment such that the highest-priority messages have a payload of 254 B. The length of the dynamic segment is chosen as 200 MS and the resulting utilization is $dynUtil = 0.071$.

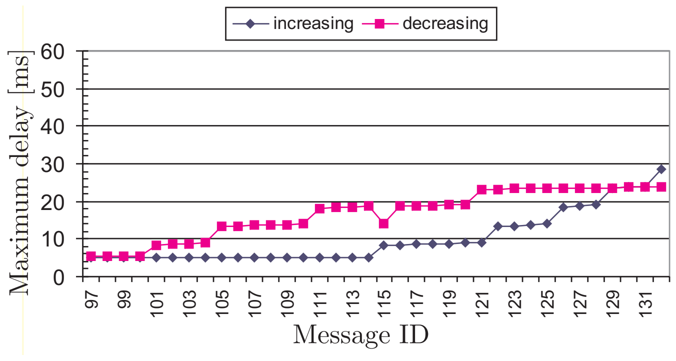

Fig. 13. Maximum delay for scheduling with increasing/decreasing payload.

Fig. 13 shows our delay measurements. It turns out that smaller maximum delays are achieved for most of the messages if the messages are scheduled with an increasing payload. Again, it is the case that the smaller messages with higher priorities cannot block as many FCs for lower-priority messages than the larger messages.

5. DISCUSSION AND CONCLUSION

The main contribution of this paper is the experimental evaluation of basic properties of the FlexRay dynamic segment that is conducted on a FlexRay network with 6 nodes and with a set of 36 sporadic messages. An investigation of the maximum message delay shows that these delays increase with a decreasing message priority as is expected from the protocol specification. Here, it is interesting to note that this increase in the message delay mostly happens in multiples of the FlexRay cycle time as a consequence of the cyclic operation of the FlexRay protocol. It is further verified that message deadlines can be violated if a message is delayed for too many FlexRay cycles, whereby it has to be highlighted that such situation can occur for very low utilizations of the dynamic segment: in the worst case, if the dynamic segment is too short, a small number of higher-priority messages is sufficient to block the transmission of a lower-priority message for multiple cycles. Finally, it has to be pointed out that, for the same utilization, transmitting messages with a larger payload leads to larger delays. Accordingly, it is verified that assigning higher priorities to messages with a smaller payload reduces the maximum delays observed on the FlexRay network. Since it is critical that the message delays in practical automotive applications remain small, the observations in this paper help to find a proper configuration of the FlexRay dynamic segment. Our future work includes a method for the analytical schedulability verification for the dynamic segment accompanied by detailed experiments.

ACKNOWLEDGEMENTS

The authors would like to thank TOFAŞ Türk Otomobil Fabrikaları A.Ş. and the Chair of Automatic Control, University of Erlangen-Nuremberg for the support of the project.

REFERENCES

(1993). Class C application requirement considerations. Technical Report J2056/1, Society for Automotive Engineers.

(2003). Time-triggered protocol TTP/C, high-level specification document, protocol version 1.1. URL `http://www.tttech.com`.

(2004). FlexRay communication system, protocol specification, version 2.0. URL `http://www.flexray.com`.

(2009). Bosch E-Ray FlexRay IP-Module user's manual. URL `http://www.semiconductors.bosch.de/pdf/ERay_Users_Manual_1_2_6.pdf`.

(2009). Flexcard Cyclone II SE. URL `http://www.eberspaecher.com/servlet/PB/menu/1053178_l2/index.html`.

(2009). Fujitsu FlexRay Evaluation Board: SK-91465X-100PMC. URL `http://mcu.emea.fujitsu.com/mcu_tool/detail/SK-91465X-100PMC.htm`.

Albert, A. (2004). Comparison of event-triggered and time-triggered concepts with regard to distributed control systems. In *Embedded World*.

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

Jung, K.H., Song, M.G., Lee, D.I., and Jin, S.H. (2008). Priority-based scheduling of dynamic segment in FlexRay network. In *International Conference on Control, Automation and Systems*.

Kopetz, H. and Bauer, G. (2003). The time-triggered architecture. *Proc. IEEE*, 91(1), 112–126.

Lukasiewycz, M., Glaß, M., Teich, J., and Milbredt, P. (2009). FlexRay schedule optimization of the static segment. In *7th IEEE/ACM International Conference on Hardware/software codesign and system synthesis*, 363–372.

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

Navet, N., Song, Y., Simonot-Lion, F., and Wilwert, C. (2005). Trends in automotive communication systems. *Proceedings of the IEEE*, 93(6), 1204–1223.

Nielsen, J.J., Schwefel, H.P., and Hamdan, A. (2007). Markov chain-based performance evaluation of flexray dynamic segment. In *Workshop on Real Time Networks*.

Pop, T., Pop, P., Eles, P., Peng, Z., and Andrei, A. (2008). Timing analysis of the flexray communication protocol. *Real-Time Syst.*, 39(1-3), 205–235.

Schedl, A. (2007). Goals and architecture of FlexRay at BMW. In *Vector FlexRay Symposium*.

Schmidt, E.G. and Schmidt, K. (2009). Message scheduling for the FlexRay protocol: The dynamic segment. *Vehicular Technology, IEEE Transactions on*, 58(5), 2170–2179.

Schmidt, K. and Schmidt, E.G. (2007). Systematic message schedule construction for time-triggered CAN. *Vehicular Technology, IEEE Transactions on*, 56(6), 3431–3441.

Tindell, K., Burns, A., and Wellings, A.J. (2000). Calculating controller area network (CAN) message response times. *Control Engineering Practice*, 3, 1163–1169.