Heuristic scheduling concepts for TTCAN networks

A. Albert, R.Hugel

Time-triggered CAN (TTCAN) combines the advantages of event- and time-triggered communication in order to fulfil the requirements of distributed real-time systems. Of crucial importance is thereby the generation of the communication schedule which should consider the demands of the time-triggered system on the one hand, while maintaining a good real-time performance for the event-triggered part of the system on the other. This paper deals with heuristic scheduling concepts for TTCAN networks and carries out a comparison by means of the above mentioned criteria. The suitability of the concepts is evaluated by laboratory measurements. The results enable the derivation of important clues in order to schedule TTCAN networks.

1 Introduction

It is quite evident that networks with time-triggered operation modes will play a major role in future automotive systems. One of such protocols is TTCAN, which combines the advantages of event- and time-triggered networks [1].
Generally, the main advantages of time-triggered networks are their quasi-deterministic behavior and the possibility to easily realize fault-tolerant systems. One of the drawbacks is the restrictive design process, since the communication structure has to be defined in advance. The procedure which determines the time slots for the respective messages is called scheduling. The main targets of the scheduling are to consider the demands of the time-triggered system on the one hand, while maintaining a good real-time performance for the event-triggered part of the system on the other.
This paper presents some heuristic scheduling concepts for TTCAN networks and carries out a comparison by means of the above mentioned criteria.

The paper is organized as follows:
Section 2 introduces into the scheduling problem and summarizes the typical requirements of distributed real-time systems. Furthermore, the section highlights the connection between these requirements and the restrictions due to the protocol as well as implementation specific limitations.
Section 3 compares some heuristic scheduling concepts for TTCAN networks on the basis of a meaningful example close to a real world application. The demands of the time-triggered part of the communication is essential and thus fulfilled by all approaches. Therefore the different approaches are evaluated by the ability of the system to react to asynchronous events.
Section 4 describes the test scenario and the measurement procedure for the evaluation of the system's real-time response. The used method yields the average latency response time and the jitter when the system is reacting to an asynchronous external event.
Afterwards, section 5 presents results for the different scheduling concepts. Further, the results are discussed. The paper ends with a summary in section 6.

2 TTCAN scheduling problem

Within a time-triggered framework the communication structure is defined in advance and generally not modified during operation. The corresponding procedure which determines the time slots for the corresponding messages is called scheduling. At start-up every communication controller is initialized with its own schedule. During operation merely the data of the time-triggered messages may be modified.
In general the scheduling problem requires the solution of an optimization problem, whereby the quality of the solution will mainly depend on an appropriate choice of the performance criterion and the considered conditions. Elementary requirements of the scheduling are concerned with the period times of

messages and relations between messages, like precedence and exclusion constraints as well as release times and deadlines. Some more sophisticated criteria will consider, for instance, the impacts of latencies on the functional software, the possibility to easily modify the schedule without side effects or the efficient use of resources.
It is not within the scope of this paper to intensively study the scheduling problem. Thus for simplicity reason we will take here into account only the required period times of the messages. Further we will not present any formal methods for the scheduling, but rather show the impacts of heuristic scheduling approaches onto the real-time performance.

Lastly, it should be mentioned that the possibilities of the scheduling depend also on constraints caused by

- protocol,
- implementation specifics, and
- physical limitations.

These are for instance (example values are taken from the Bosch TTCAN evaluation chip specification):

- number of basic cycles (in power of two, max 64)
- number of network time units per basic cycle (2^16)
- repetition period of messages related to multiples in powers of two
- time marks defined for the first basic cycle can not be changed in subsequent cycles
- number of triggers (32 per node)
- number of message buffers
- oscillator tolerance

3 Heuristic scheduling

Scheduling concepts with special emphasis on TTCAN networks can be found, for instance, in [2,3]. The work [2] solves the scheduling problem on the basis of a genetic algorithm. Starting from an initial system matrix, the proposed procedure determines out of a set of solutions the best one by means of a cost function which represents the deviations of the real to the desired transmission instants. The work [3] gives a quite comprehensive overview on heuristic scheduling concepts. Two strategies are presented. The one minimizes the number of basic cycles in order to produce a comprehensible schedule, whereas the other minimizes the length of the basic cycle in order to minimize the number of necessary triggers. Further, the work [3] distinguishes between a dense and a sparse allocation of the time-triggered slots and sketches theoretically the impacts on the time behavior of the remaining part of the system.
This is also one of the main focus of the present work. In addition, we will present measurements of the average delay and the jitter for the envisaged scheduling strategies and show the relationship to the TTCAN implementation and its limitations.

It is the intention of a heuristic approach to bring the requirements of the application (min. and max sample rate) in line with the possibilities of the communication system in a fairly simple manner.
First of all, the communication objects are sorted according to their

- repetition rate (period)
- message length
- periodic / spontaneous messages
- response time of spontaneous messages on events
- temporal dependencies between messages

With this input, a basic attempt for a rate monotonic schedule will be checked under the consideration of the before mentioned constraints.
The length of a basic cycle is chosen according to the shortest period and the number of basic cycles according to the longest period. If the number of basic cycles exceeds the maximum of 64, then the messages with longer periods shall be defined as spontaneous and their response time must be investigated later on.
Messages with periods not matching the power of two requirement will be scheduled for the next lower suitable period.
To get an overview of the system matrix, all messages with low periods are filled in the first time windows, followed by the messages with longer periods in ascending order until the basic cycle is full.

If there are more windows needed, then the larger periodic messages can be distributed to subsequent basic cycles. For simplicity, all time windows are of the same maximum length for eight byte messages.
The unoccupied windows can be used for spontaneous messages and defined as "Arbitrating Windows" or dedicated to messages with a suitable required response time.
With this basic approach, the performance of requirements demanded by the application and the constraints shall be checked.
This paper focuses on the influence of protocol and implementation specific constraints like number of basic cycles, triggers and response time and assumes that physical requirements like clock tolerances are fulfilled.
To explain the basic principle, the remaining part of this paper shows an example with the following characteristics:

- All messages of length 8 byte
- Baud rate 500 kbit/s
- period times of messages : 3 messages with 5ms (IDs 51_H, 52_H, 53_H), 5*10ms (IDs 101_H to 105_H), 4*20ms (IDs 201_H to 204_H), 4*40ms (IDs 401_H to 405_H) and some 100ms as well as 200ms messages, the total band width usage is approx. 40%

0.0ms 5.0ms ….	0.3ms 5.3ms ….	0.6ms 5.6ms …	0.9ms 5.9ms …	1.2ms 6.2ms ….	1.5ms 6.5ms ….	1.8ms 6.8ms ….	2.1ms 7.1ms ….	2.4ms 7.4ms ….	2.7ms 7.7ms ….	3.0ms 8.0ms ….	3.3ms 8.3ms ….	3.6ms 8.6ms ….	3.9ms 8.9ms ….	4.2ms 9.2ms …	4.5ms 9.5ms ….
51	52	53	101	102	103	104	105	201	202	203	204	401	402	403	404
51	52	53													
51	52	53	101	102	103	104	105								
51	52	53													
51	52	53	101	102	103	104	105	201	202	203	204				
51	52	53													
51	52	53	101	102	103	104	105								
51	52	53													

Figure 1: Rate monotonic schedule; every line represents 5ms along the time axis

window	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
time [ms]	0	0,16	0,45	0,75	1,05	1,35	1,65	1,95	2,25	2,55	2,85	3,15	3,45	3,75	4,05	4,35	4,65	5
BitTime[bit]	0	78	223	373	523	673	823	973	1123	1273	1423	1573	1723	1873	2023	2173	2323	2500
0	F0	51	52	53	101	102	103	104	105	201	202	203	204	401	402	403	404	
1	F0	51	52	53	1FF, 1	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 3	
2	F0	51	52	53	101	102	103	104	105	2FF, 1	2FF, 2	2FF, 2	2FF, 2	2FF, 2	2FF, 2	2FF, 2	2FF, 3	
3	F0	51	52	53	1FF, 1	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 3	
4	F0	51	52	53	101	102	103	104	105	201	202	203	204	4FF, 1	4FF, 2	4FF, 2	4FF, 3	
5	F0	51	52	53	1FF, 1	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 3	
6	F0	51	52	53	101	102	103	104	105	2FF, 1	2FF, 2	2FF, 2	2FF, 2	2FF, 2	2FF, 2	2FF, 2	2FF, 3	
7	F0	51	52	53	1FF, 1	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 2	1FF, 3	

Figure 2: Realization '8bc_version1' with 8 basic cycles and 3 merged arbitrating windows (time scale is distorted, spaces in between the messages appear smaller than they are)

window	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
time [ms]	0	0,16	0,45	0,75	1,05	1,35	1,65	1,95	2,25	2,55	2,85	3,15	3,45	3,75	4,05	4,35	4,65	5
BitTime[bit]	0	78	223	373	523	673	823	973	1123	1273	1423	1573	1723	1873	2023	2173	2323	2500
0	F0	51	52	1FF, 1	1FF, 2	1FF, 3	53	101	2FF, 1	2FF, 2	2FF, 3	102	103	4FF, 1	4FF, 2	4FF, 3	104	
1	F0	51	52	1FF, 1	1FF, 2	1FF, 3	53	105	2FF, 1	2FF, 2	2FF, 3	201	202	4FF, 1	4FF, 2	4FF, 3	203	
2	F0	51	52	1FF, 1	1FF, 2	1FF, 3	53	101	2FF, 1	2FF, 2	2FF, 3	102	103	4FF, 1	4FF, 2	4FF, 3	104	
3	F0	51	52	1FF, 1	1FF, 2	1FF, 3	53	105	2FF, 1	2FF, 2	2FF, 3	204	401	4FF, 1	4FF, 2	4FF, 3	402	
4	F0	51	52	1FF, 1	1FF, 2	1FF, 3	53	101	2FF, 1	2FF, 2	2FF, 3	102	103	4FF, 1	4FF, 2	4FF, 3	104	
5	F0	51	52	1FF, 1	1FF, 2	1FF, 3	53	105	2FF, 1	2FF, 2	2FF, 3	201	202	4FF, 1	4FF, 2	4FF, 3	203	
6	F0	51	52	1FF, 1	1FF, 2	1FF, 3	53	101	2FF, 1	2FF, 2	2FF, 3	102	103	4FF, 1	4FF, 2	4FF, 3	104	
7	F0	51	52	1FF, 1	1FF, 2	1FF, 3	53	105	2FF, 1	2FF, 2	2FF, 3	204	403	4FF, 1	4FF, 2	4FF, 3	404	

Figure 3: Realization '8bc_version2' with 8 basic cycles and 3 distributed merged arbitrating windows

window	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
time [ms]	0	0,16	0,45	0,75	1,05	1,35	1,65	1,95	2,25	2,55	2,85	3,15	3,45	3,75	4,05	4,35	4,65	5
BitTime[bit]	0	78	223	373	523	673	823	973	1123	1273	1423	1573	1723	1873	2023	2173	2323	2500
0	F0	1FF, 0	51	1FE, 0	52	1FD, 0	53	1FC, 0	101	1FB, 0	102	1FA, 0	103	1F9, 0	104	1F8, 1	1F8, 3	
1	F0	1FF, 0	51	1FE, 0	52	1FD, 0	53	1FC, 0	105	1FB, 0	201	1FA, 0	202	1F9, 0	203	1F8, 1	1F8, 3	
2	F0	1FF, 0	51	1FE, 0	52	1FD, 0	53	1FC, 0	101	1FB, 0	102	1FA, 0	103	1F9, 0	104	1F8, 1	1F8, 3	
3	F0	1FF, 0	51	1FE, 0	52	1FD, 0	53	1FC, 0	105	1FB, 0	204	1FA, 0	401	1F9, 0	402	1F8, 1	1F8, 3	
4	F0	1FF, 0	51	1FE, 0	52	1FD, 0	53	1FC, 0	101	1FB, 0	102	1FA, 0	103	1F9, 0	104	1F8, 1	1F8, 3	
5	F0	1FF, 0	51	1FE, 0	52	1FD, 0	53	1FC, 0	105	1FB, 0	201	1FA, 0	202	1F9, 0	203	1F8, 1	1F8, 3	
6	F0	1FF, 0	51	1FE, 0	52	1FD, 0	53	1FC, 0	101	1FB, 0	102	1FA, 0	103	1F9, 0	104	1F8, 1	1F8, 3	
7	F0	1FF, 0	51	1FE, 0	52	1FD, 0	53	1FC, 0	105	1FB, 0	204	1FA, 0	403	1F9, 0	404	1F8, 1	1F8, 3	

Figure 4: Realization '8bc_version3' with 8 basic cycles and widely spread distributed arbitrating windows

As already mentioned, we neglect any precedence and exclusion constraints and consider merely the required period times of the messages. Thus, it is straight forward to generate a first schedule in a rate monotonic fashion.
Figure 1 qualitatively shows the result (reference message is neglected here). The time windows are chosen as 0.3ms which is sufficiently wide for eight byte messages on a 500kbit/s system. Only the messages up to the period time 40ms are considered as periodic messages; messages with longer period times are treated here as asynchronous messages and will be sent within arbitrating windows. Thus the longest period is 40ms and since all other periods are integer factors of this value, the cycle time after which the communication structure is repeated is 40ms as well.
Figure 2 demonstrates an almost one to one realization of the rate monotonic result. It shows a realizable TTCAN schedule with eight basic cycles and a basic cycle length of 5ms. The reference message is denoted with the ID F0$_H$. The free spaces (the blank windows in figure 1) have been filled with arbitrating windows. Since consecutive arbitrating windows may be (if some certain requirements are met) combined to so called merged arbitrating windows, for the envisaged case three merged arbitrating windows with the dummy IDs 1FF$_H$, 2FF$_H$ and 4FF$_H$ are introduced. The number behind the dummy ID indicates the position of the current window within the merged arbitrating window. For instance, the notation 1FF$_H$,1 indicates that it is the first window and 1FF$_H$,3 that it is the last window within a merged arbitrating window, respectively. Merely three message objects have to be defined here, but there is a high demand for triggers since every time window in the merged arbitration window needs a dedicated trigger.
In general the number of triggers will depend on the structure of the communication matrix. The following table shows the influence of the number of the basic cycles on the number of triggers for the considered exemplary message set. It is assumed that the considered node participates in all messages which is a less likely occurrence but represents the worst case. Additionally, the values consider the reference message.

# basic cycles	# triggers
1 (d = 40ms)	57
2 (d = 20ms)	31
4 (d = 10ms)	20
8 (d = 5ms)	17
16 (d= 2.5ms)	17
32 (d= 1.25ms)	17

The number of triggers decreases with the number of basic cycles. A realization with one basic cycle is not possible, since the number of necessary triggers exceeds the value of 32 (Bosch TTCAN evaluation chip). For the two basic cycle solution there will be for instance merely one trigger be left for the definition of an additional arbitrating window. Starting from eight basic cycles, the worst case number of triggers for the periodic messages equals 17.

As one can imagine (it will be verified in section 5) the solution in figure 2 is not very appropriate from the real-time performance point of view. For instance, the system is not able to transmit any additional non periodic message in basic cycle 0, meaning that the system will not be able to react to any asynchronous event within a time range of more than 5ms. The realization in figure 3 shows a sparser allocation of the time-triggered messages and provides a better real-time performance. By the way the number of triggers is significantly decreased, since the messages within the merged arbitrating windows can be defined as messages with high repetition rates. Merely 9 triggers are required in contrast to 25 triggers for the realization in figure 2 (see also subsequent table). The number of message objects for the arbitrating windows is unchanged and equals three. As figure 4 shows, it is possible to further distribute the time-triggered messages and to achieve an even better real-time performance; merely the number of message objects is increased.
The following table summarizes the necessary number of arbitration message

objects and the corresponding number of triggers for the different realizations.

	# arbitration mess. objects	# triggers for arbit. Mess. objects
vers. 1	3	25
vers. 2	3	9
vers. 3	8	9

One has to take care that for every node the sum of number of triggers required for the periodic messages and the arbitrating messages as well as the number of message objects must not exceed the value 32.

4 Reaction to asynchronous external events

As presented in [4,5] it is possible to evaluate the ability of a communication systems to react to an asynchronous event by the so called 'Distinctness of Reaction' (DoR) [6]. The method is based on an orthogonal Walsh correlation and yields a reliability measure given by the average latency response time and the jitter when reacting to asynchronous external events.
In order to measure the DoR, the communication system is excited at node A by a square wave signal *i(t)* with an adjustable frequency as is shown in figure 5. Every rising edge of *i(t)* simulates the occurrence of a critical situation. The system reacts in a predefined manner with a so called A→B→A cycle. Node A informs another node B about the occurrence of the critical situation and waits for its response. After the A→B→A cycle the system reacts in a predefined manner with its response *x(t)*. The signals *i(t)* and *x(t)* are processed by a FPGA board which quantifies the DoR. The result is delivered via a serial interface. The DoR is a measure for the jitter in the system's response and takes on values from 100% (no jitter) to 0% (at least sporadic loss of excitations).
Not only the determination of a sole value is carried out (constant frequency of the excitation *i(t)*), but the recording of an entire frequency response, which consists of an amplitude response and a phase response. The DoR determines the amplitude response. The phase response is determined by the average response time τ of the system's response. In order to achieve a standard of comparison a normalization is carried out and the 'average skew' $s = -\tau/T$ is introduced. The skew *s* is scaled downwards from 0% to -100%. This choice provides the advantage that it allows to evaluate the system's quality from the plot by the simple rule 'the higher, the better' which holds true in the same manner also for the comparison by means of the DoR plots.
The figures 6 and 7 show the best and worst case for the schedule in figure 2. Figure 6 shows the situation where the excitation of the system occurs within an merged arbitrating window. The system is able to perform the A→B→A cycle within two consecutive windows of the merged arbitrating window, leading to the minimum response time. In the situation of figure 7 the worst case is illustrated. The system is excited at the end of basic cycle 7 such that it is not possible to perform the entire A→B→A cycle within that cycle. In fact, the reply message is transmitted in basic cycle 1 (since there is no possibility for a transmission in basic cycle 0), resulting in a system's response time of more than 6ms.

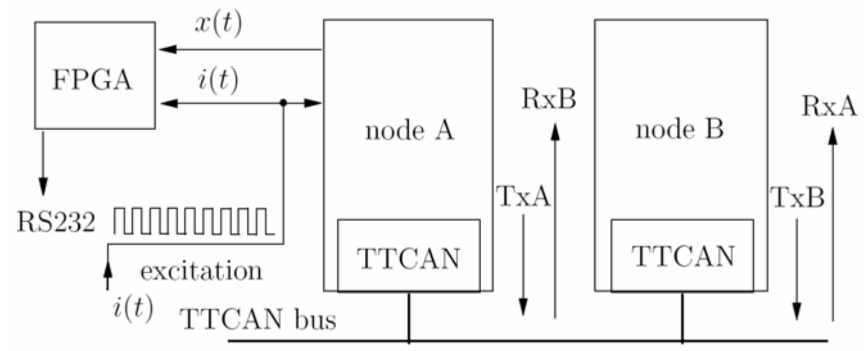

Figure 5: Relevant part of the system's configuration in order to evaluate the ability of the system to react to asynchronous events

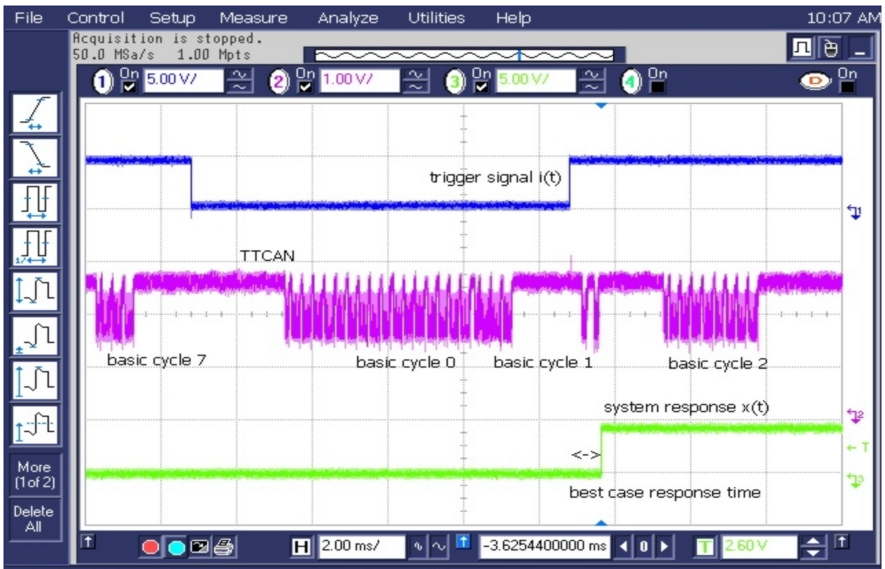

Figure 6: Best case response time for the schedule in figure 2

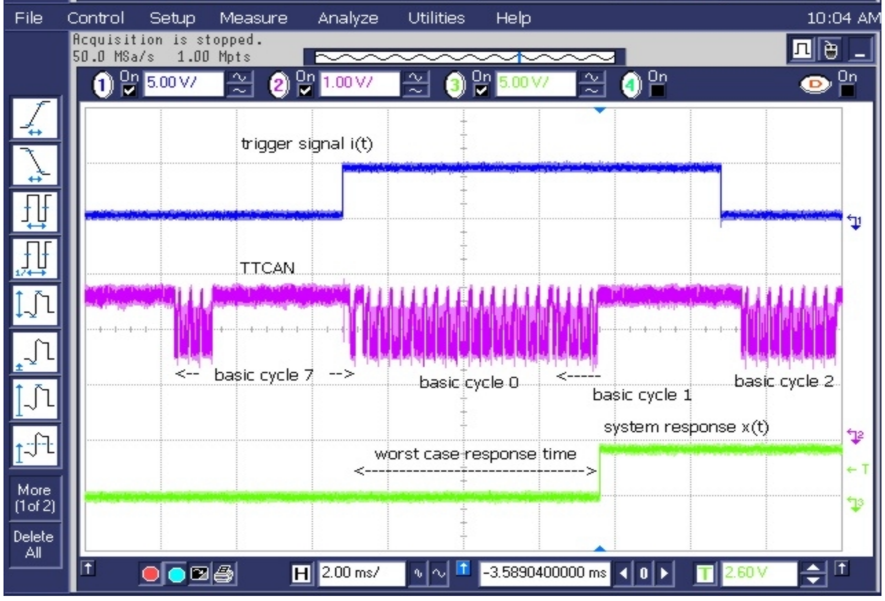

Figure 7: Worst case response time for the schedule in figure 2

5 Results

Figure 8 illustrates the measured 'frequency response' (DoR and skew) for the system realization in figure 2.
The curve ends for the excitation frequency 143Hz. I.e., for higher excitation frequencies there are losses of excitations. The value of 143Hz can be derived from the worst case scenario in figure 7.
Both curves, the DoR and the skew, decrease with increasing excitation frequencies. I.e., the higher the frequency, the higher the jitter in the system's response and the higher the delay in relation to the period of the excitation signal.
The skew shows an almost linear characteristic which means a quite regular behavior, since a linear skew means a constant average delay of the system.
Some interesting peculiarities occur for excitation frequencies which correspond to multiples of the basic cycle period of 5ms. For instance, we get resonances for 100Hz and 50Hz, since then the system is always excited in the same situation and thus behaves any time in the same way. As a result there is no jitter and the DoR equals approximately 1. For this cases the value of the skew is of no meaning and depends on the relative phase relation between both time bases, the bus and the excitation.
Figure 9 compares the different realizations with eight basic cycles, corresponding to the matrices in figures 2, 3, and 4. The real-time performance is significantly increased by the distribution of the arbitrating windows. The realization

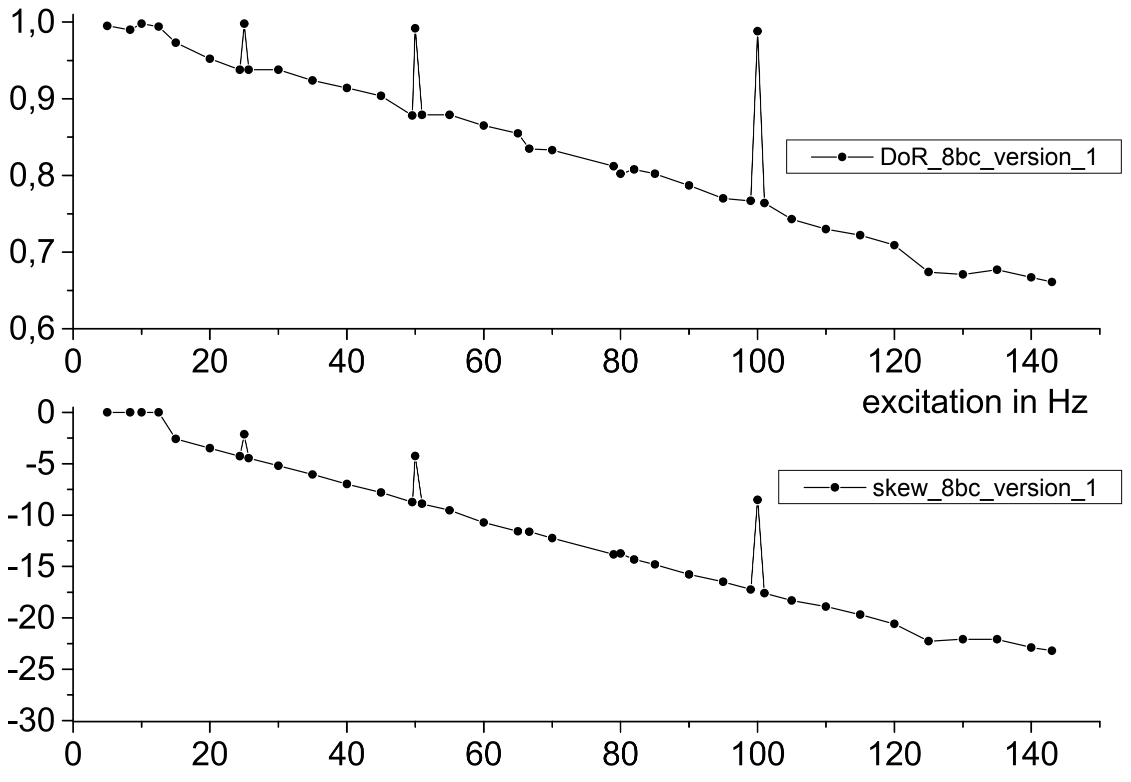

Figure 8: DoR and skew for the realization with 8 basic cycles and 3 merged arbitrating windows (8bc_version_1)

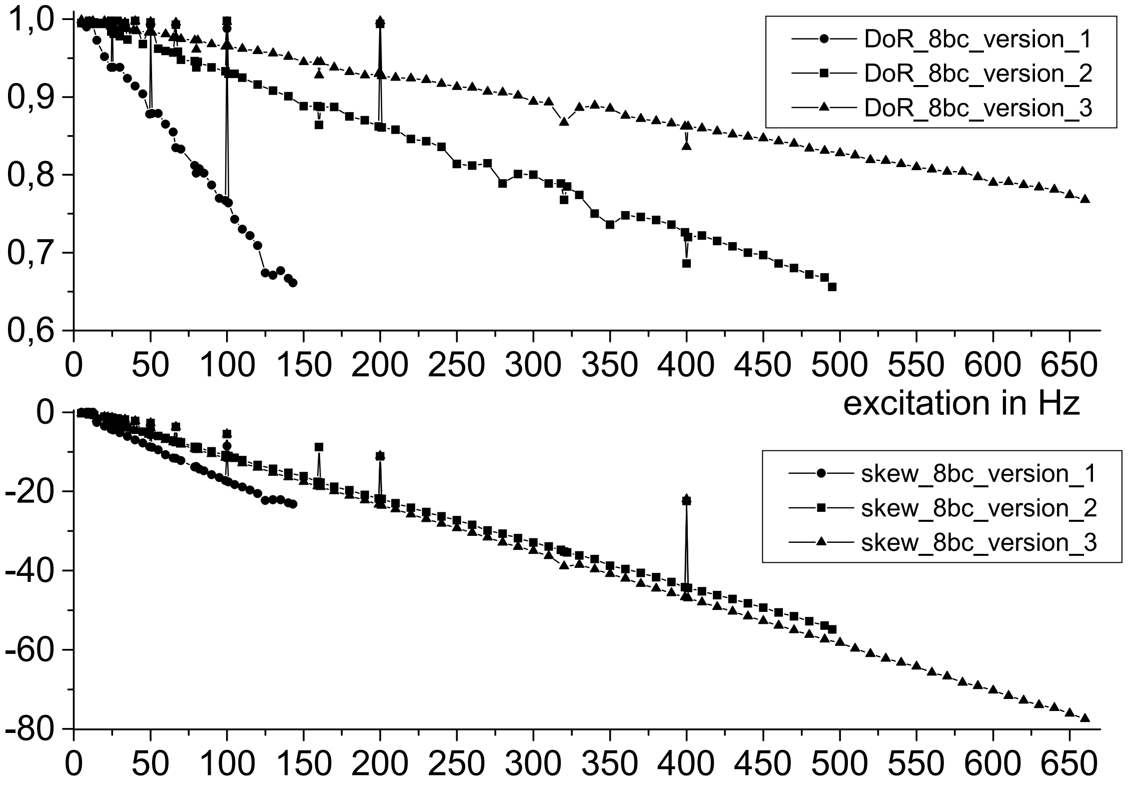

Figure 9: DoR and skew for the different realizations with 8 basic cycles (see figures 2,3,4)

'8bc_version2' already permits the occurrence of asynchronous events with approximately 500Hz, whereas the realization '8bc_version3' even enables up to 670Hz. For these realization there is less jitter as well as less average delay.

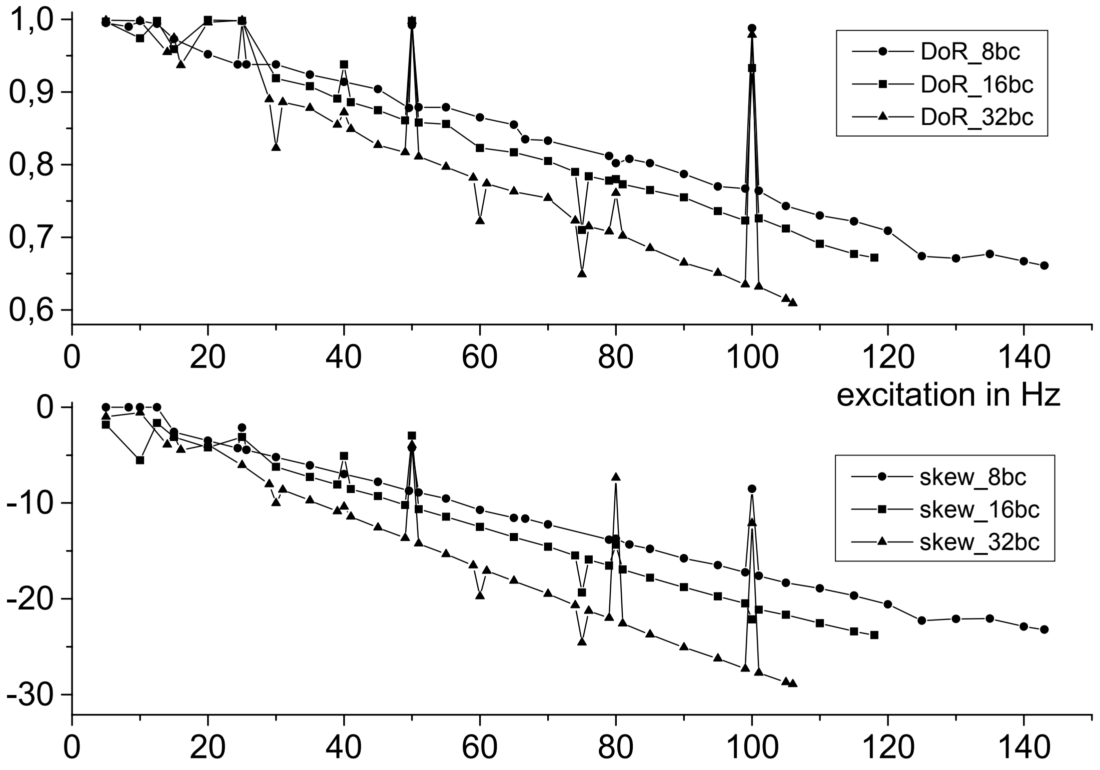

Figure 10: DoR and skew for the rate monotonic schedule realized with 8, 16 and 32 basic cycles

In order to investigate the influence of the number of basic cycles we have further compared different realizations with 8, 16 and 32 basic cycles. All communication matrices have been constructed in correspondence to figure 2. Figure10 shows the results. The more basic cycles are used, the worse is the real-time performance. I.e., there is more jitter and more average latency. This result is due to the fact that there is a worse usage of the potential band width with more basic cycles. On the one hand the reference message is sent more frequently; on the other hand the relative waste of time between the last realizable time window within the basic cycle and the end of the basic cycle may be increased.

6 Summary

This paper compared some heuristic scheduling concepts for TTCAN networks. For that purpose we have selected an exemplary set of messages close to real-world automotive applications. Within this paper we neglected any precedence and exclusion constraints and considered merely the required period times for the messages. Since all considered heuristic approaches fulfilled the corresponding (time-triggered) demands they differed only in their real-time performance. For the comparison we have presented measurements which yielded the average delay and the jitter when the system is reacting to asynchronous events.

The results with respect to the here considered conditions are summarized as follows:

- It is advantageous to use long basic cycles, since then the decrease in band width usage due to the reference message and bad fitness of the last

time window into the basic cycle length is lower.
A limiting factor might be the number of necessary triggers which is increased for basic cycles with many time windows. An appropriate compromise seems to be to set the basic cycle length equal to the lowest message period.

- A very significant increase in the real-time performance may be achieved by a well distribution of the arbitrating windows within the communication matrix. For the presented example we have gained an improvement by a factor of approximately 4. A limiting factor for the distribution might be the number of available triggers and/or the number of message objects.

References

[1] *International Standardization Organization*. ISO 11898-1 (Controller Area Network, Data Link Layer), ISO 11898-2 (High-Speed Transceiver), ISO 11898-3 (Fault-Tolerant Low-Speed Transceiver), ISO 11898-4 (Time-Triggered Communication).

[2] *J.A. Fonseca, F. Coutinho, and J. Barreiros*. Scheduling for a TTCAN Network with a Stochastic Optimization Algorithm. 8th international CAN in Automation Conference, Las Vegas, pages 07/10–07/16, 2002.

[3] *R. Johannson*. Time and event triggered communication scheduling for automotive applications. Technical Report 17, Chalmers Lindholmen University College, Göteborg, Sweden, 2004.

[4] *A. Albert and W. Gerth*. Evaluation and Comparison of the Real-Time Performance of CAN and TTCAN. 9th international CAN in Automation Conference, Munich, pages 05/01–05/08, 2003.

[5*] A. Albert*. Comparison of Event-Triggered and Time-Triggered Concepts with Regard to Distributed Control Systems. Embedded World 2004, pages 235–253, Feb. 2004.

[6] *B. Wolter*. Messung der Dienstgüte von Echtzeitbetriebssystemen durch Walsh-Korrelation. PhD thesis, Fortschrittberichte VDI, Reihe 8, Nr. 964, VDI-Verlag, Düsseldorf, 2002.

Dr.-Ing. Amos Albert
Robert Bosch GmbH
71701 Schwieberdingen
++49 (0)711-811/43607
amos.albert@de.bosch.com

Robert Hugel
Robert Bosch GmbH
71701 Schwieberdingen
++49 (0)711-811/8517
robert.hugel@de.bosch.com