Remote Measurement of Local Oscillator Drifts in FlexRay Networks

Eric Armengaud[1,2], Andreas Steininger[1]
[1] Vienna University of Technology, Embedded Computing Systems Group E182-2
[2] The Virtual Vehicle Competence Center
eric.armengaud@v2c2.at, steininger@ecs.tuwien.ac.at

***Abstract*—Distributed systems, especially time-triggered ones, are implementing clock synchronization algorithms to provide and maintain a common view of time among the different nodes. Such architectures heavily rely on the nodes' local oscillators to remain within given accuracy bounds. However, measuring the oscillator frequencies (e.g., for maintenance or diagnosis) is usually difficult to perform since it requires physical access to each single node and may interfere with the running application. Moreover, clock synchronization features tend to mask clock deviations. In this work, we propose a non-intrusive method for remote measurement of the individual oscillator drifts within a distributed system. Our approach is based on a tester that sends carefully aligned messages to stimulate the clock synchronization service and records the resulting bus traffic for an analysis of the nodes' synchronization behavior. This tester needs access to the communication bus only. We focus our work to FlexRay and validate our approach by experiments.**

I. Introduction

Distributed systems are usually composed of self-contained computing nodes, each having its individual independent clock source. Since the availability of a common view of time among all nodes has fundamental advantages with respect to task synchronization, bus access scheduling etc., a distributed clock synchronization algorithm is often employed to compensate the mismatch of the individual clock rates on a "macrotick" level. Central to all these clock synchronization algorithms, however, is the assumption of bounded oscillator drift [1], [2].

Excessive oscillator drift may long go unnoticed, since safety margins in the synchronization mechanism tend to mask it (particularly if rate correction is applied). In some cases it may lead to marginal bus timing behavior ("slightly-off-specification faults") that tends to cause subtle, even Byzantine, effects on application level [3], [4]. The motivation for measuring oscillator drifts is therefore to perform diagnosis of the system and both detect a faulty oscillator (error detection) or forecast an oscillator being to fail (preventive maintenance). Oscillator drift can result from ageing effects or be due to hostile environment (e.g., temperature, humidity, vibration). Crystal oscillators are generally considered less reliable than VLSI chips.

The main problem for this measurement is the requirement to physically access each local oscillator. This is often especially difficult in case of highly integrated and / or distributed systems (e.g. automotive, avionics) and might entail stopping the (safety-critical) application and even disassembling the system.

We propose a non-intrusive method for a remote measurement of the oscillator drifts within time-triggered systems [5]. Our approach is based on a single tester that synchronizes to a running system and then sends a precisely aligned test pattern (messages) to influence the clock synchronization algorithm. We will see that the response analysis allows measuring each individual oscillator drift within the system. This approach provides two main advantages: (1) *accessibility*, we only require a single access point to the bus line (in contrast to direct access to each oscillator), and (2) *transparency*, the measurement phase can be performed during normal system operation. This measurement requires few resources only (bandwidth for the stimulus) and is totally transparent for the application since the tester's messages are not further processed.

The method was developed as a supplement to our ExTraCT project[1] that aims at optimizing the generation of online tests for time-triggered communication protocols such as FlexRay [6]. This work is organized as follows: Section II describes the system model under consideration. Section III presents different approaches for remotely stimulating and measuring attributes of the clock synchronization. Further, Section IV presents two specific algorithms for measuring the oscillator drift. Finally, Section V concludes this work.

II. System under consideration

The system considered consists of nodes ("*self-contained computers with their own hardware and software*" [7] p.75) which cooperate to deliver services for their environment. Each node comprises a communication controller that is in charge of sending and receiving data over a communication network as shown in Figure 1. Time-triggered communication protocols such as TTP/C [8] and FlexRay [6] implement a Time Division Multiple Access (TDMA) scheme and divide the bus access into periodic, a-priori defined, time windows, which are statically assigned to the nodes for message transmission.

Evidently, the establishment of a global time base is of utmost importance for time-triggered systems. This global time base is required, amongst other things, for collision free communication in the TDMA scheme and for obtaining a consistent view of the environment among the different nodes.

[1]The ExTraCT-project received support from the Austrian FIT-IT Embedded Systems initiative under grant 810834, see http://ti.tuwien.ac.at/extract

978-3-9810801-5-5/DATE09 © 2009 EDAA

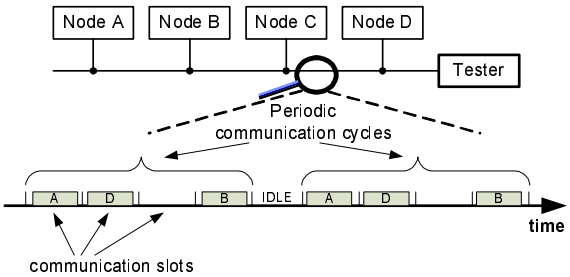

Fig. 1. System model

However, the nodes are self-contained and have their own (imperfect) quartz. Consequently, periodic re-synchronization is required to correct the different deviations and establish a system-wide synchronized time base.

There are several effects that can affect the quartz frequency, e.g. crystal structure, temperature, vibrations [9]. For the following, we will assume that for a small time window (few seconds) the oscillator frequency is constant. This means that the short-term variations are assumed to be negligible in comparison to the long-term variations and that the environment (temperature, acceleration) is assumed to be nearly constant during this time window (which is not unrealistic, see [10] Section 4.1 for further discussion). We define the *oscillator deviation* for node p at time t as $\delta_p(t) = \frac{f_p(t)}{f_p^0} - 1$ (with $f_p(t)$ and f_p^0 the current and nominal oscillator frequency, respectively). Each node presents its own oscillator deviation and we denote *clock deviation interval* (CDI) as the interval between the largest and the smallest oscillator deviations within the system.

Time-Triggered communication protocols provide a clock synchronization algorithm to correct their physical clock (oscillator) and construct an artificial, globally synchronized, *logical clock*. Two mechanisms are implemented in FlexRay. First, the *offset correction* corrects the current clock state difference measured between the nodes. Second, the *rate correction* artificially modifies the frequency of the nodes' logical clocks in order to correct the long term frequency deviation from the quartz. The rate correction mechanism would not be mandatory for synchronization but has been introduced to improve the effective precision.

FlexRay implements convergence-averaging clock correction and remote clock estimation based on the time transmission technique. Hence, a subset of nodes (e.g. with better oscillators) are transmitting *sync* frames (normal frames with the synchronization flag set), thus actively taking part in the clock correction process. The other nodes are passively synchronizing to the resulting global time. The correction value is computed as an average of the measured clock state (resp. rate) difference from selected nodes, and therefore depends on the oscillator deviations within the system. The (synchronized) logical clocks will deviate from their nominal values, too. The deviation is then a combination of the local oscillator deviation and of the clock correction. We define the *global time* as the set of fault-free logical clocks within the system. Since the logical clocks are synchronized within the system precision π, the global time is defined within a jitter of π, too. Notice, that the physical and the logical clocks might present different deviations. While the logical clocks present approximately the same deviation from a node to another (since they are synchronized), there is usually no relationship between the individual oscillator deviations within the system.

A bounded oscillator deviation is required to guarantee a bounded system precision, and in time-triggered communication systems is mandatory for collision free communication. The contribution of this work is a method to remotely measure each single oscillator deviation in the system without having to physically access or modify the nodes (remote, non-intrusive approach) and without disturbing the application (transparent measurement).

III. Cornerstones of the concept

A. Key principles

As outlined in the previous section, FlexRay also implements a rate correction that artificially modifies the rate of the logical clock. This effect is absolutely positive for the clock synchronization itself (the logical clocks are running closer to another, thus yielding a better precision) but negative for our diagnosis purpose since it masks the physical clock deviation. Notice that without rate correction (as, e.g., in TTP/C) the nodes' logical and physical clocks present the same deviation and therefore the oscillator deviation can be easily deduced from the clock state difference (measured against a reference tester node).

FlexRay's rate correction mechanism implements a damping factor to avoid common mode drift (without this measure the nodes could agree on any arbitrary logical clock rate by appropriate matching of their rate corrections, thus losing the reference to real time): The rate correction (absolute value) that should be applied according to the clock measurement is decreased by the value of the damping factor, which slightly shifts the rate of the logical clock from its ideal value towards the physical clock deviation. For example, if the damping factor is one microtick (clock period) and the clock synchronization algorithm requests a correction value of 40 (respectively -40) microticks, only 39 (respectively -39) microticks will be actually corrected during the next cycle. Figure 2 illustrates that this effect leads to a non-linearity between the rate measurement (requested correction) and the (applied) rate correction around zero.

This mechanism leads to several important properties that will be further used to measure the physical clock deviation:

P1 : In a stable environment the logical clock deviation for each node cannot be outside of the clock deviation interval (CDI) [11]. Should it be artificially moved outside these bounds by some "force" (faults, e.g.) then it will automatically return to within the bounds as soon as the force ceases.

P2 : Due to the damping factor a node with a positive rate correction will not correct the entire physical

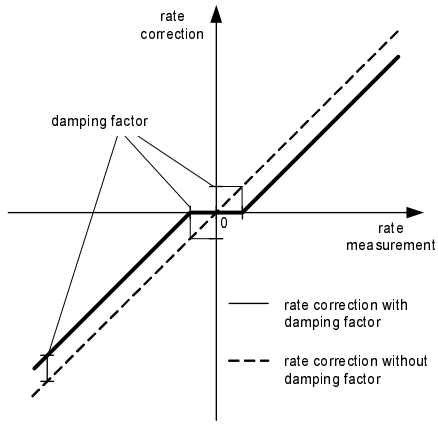

Fig. 2. The effects of the damping factor for the rate correction

clock deviation and therefore will require a positive offset correction. Similarly, nodes with a negative rate correction will exhibit negative offset correction.

In Section IV these properties will be exploited for measuring the deviations of the individual nodes' physical clocks. For this purpose we need means (a) to force the global time to follow a given trajectory and (b) to observe a node's reaction, in particular to identify where the non-linearity of its rate correction is located. For this purpose we introduce a dedicated tester node connected to the FlexRay network. Since the tester is a single dedicated node, one can afford a more sophisticated design.

B. Deterministic replay

We propose deterministic replay as a method to move the global clock – in other words the logical clocks of all non-faulty nodes – along a desired trajectory. To this end we use a tester node with the capability to send a pre-defined bus traffic regardless how the other nodes react. This introduces a two-hierarchy clock synchronization scheme. The tester node only listens to its own frames while the standard nodes take each sync frame within the network into account for synchronization and thus ultimately follow the tester.

In order to contribute to the correction term in a fault tolerant clock synchronization algorithm, it is not sufficient for the tester to send just one sync frame. FlexRay's fault-tolerant midpoint (FTM) algorithm with resilience f will discard the f highest and f lowest values before actually applying the convergence function, i.e. averaging two values. The tester must therefore send $T > f$ frames per cycle. For example, a system of four nodes can tolerate $f = 1$ fault and the tester then sends two sync frames per cycle.

In a typical scenario both these frames might be equally delayed against the ones provided by the standard nodes, so one of them will be discarded, while the other one is taken into account as one input of the convergence-average calculation. The second input is provided by one of the standard nodes. Relative to the delay imposed by the tester all standard nodes are tightly synchronized to each other. Therefore they will all correct approximately half the clock state difference to the tester at each re-synchronization point. The tester, however, does not correct its logical clock, so the standard nodes will gradually approach the logical clock trajectory dictated by the tester.

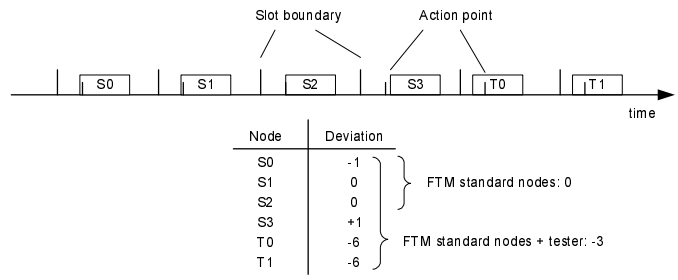

Fig. 3. Deterministic replay example

Figure 3 illustrates this scenario of a deterministic replay operation. Six sync frames are transmitted: four from standard nodes (S0, S1, S2, S3) and two (T0, T1) from a tester node. The standard nodes are tightly synchronized to each other, while the tester frames are way too early but – in order not to be dropped before the clock correction computation – still within the slot boundaries. The FTM function applied to the four standard nodes returns an offset correction of zero. However, when the tester is taken into account, the correction result is shifted to -3, which represents half the clock state difference between the standard nodes and the tester, as expected in this setting.

This provides evidence, that influencing the clock synchronization by a tester is indeed possible. More especially, the replay approach proposed here enables to remotely dictate the logical clock deviation within the system. It can be shown that this kind of influence can be safely controlled such that the system can still operate correctly within known precision bounds and with its full fault tolerance [12]. In our experiments we have used a FlexRay bus analyzer [13] with extended functionalities for replay. It is able to synchronize to the bus traffic and then send own sync frames shifted with a temporal resolution of 25ns.

C. Remote offset correction measurement

We have seen in Section III-A that a node's offset correction reflects the non-linearity of its rate correction (change of sign according to property P2). Unfortunately, the offset correction value $OC_p(m)$ at cycle m is a local parameter for node p and hence not directly available to our tester. In the following we propose a method to extract $OC_p(m)$ from the bus traffic monitored by a single tester, i.e. without any support from the node under observation.

First, the cycle length $cl_p(m)$ is approximated using the continuous measurement of the distance between the frames sent by a node p, see Figure 4 (in case p occupies more than one static slot per round, those with same identifier must be chosen). These cycle length measurement results are then split

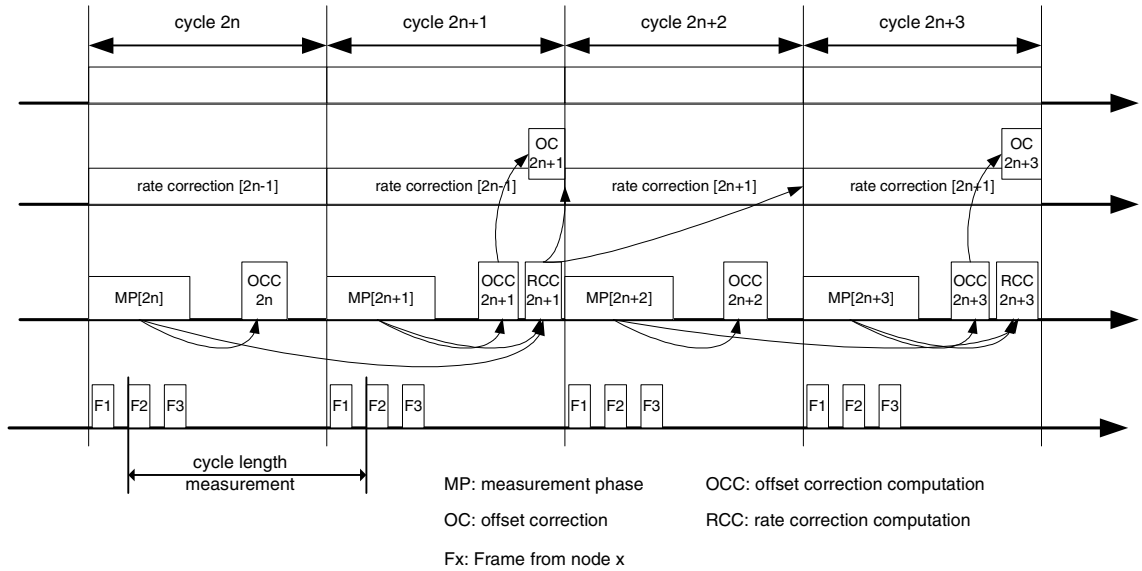

Fig. 4. Offset and rate correction computation measurement for FlexRay [6]

into odd and even cycle according to the cycle number (this information is packed in the frame header and thus globally available). As illustrated in Figure 4 the correction values are updated every two cycles and the offset correction is applied only during the odd cycles. Consequently, a direct comparison between two successive cycles suffices to get a node's local offset correction. This measurement assumes physical clock deviation and rate correction changes between consecutive cycles to be negligible, which is the case with our assumptions (see Section II). Assuming a constant rate correction for cycle $2n+1$ and $2n+2$ in Figure 4, we measure the offset correction of node 2 as $OC_2(2n+1) = cl_2(2n+2) - cl_2(2n+1)$.

IV. Clock deviation measurement in FlexRay

A. Measuring the clock deviation interval

According to property P1 of Section III-A the rate of the logical clocks for all fault-free nodes will remain in the frequency interval spanned by the CDI, and if moved outside by some "external force", they will finally return to this interval. Our first approach will be to measure the CDI, thus yielding an overview of the range of clock deviations (without being able to identify individual deviations though). For this purpose we use our deterministic replay to force the global time to follow a slope that is likely to move the rate of the logical clocks out of the CDI. Next, our tester stops transmitting and monitors the movement of the global time. The slope at which the global time first stabilizes (i.e. the rate of the logical clocks upon which the collective of nodes first finds agreement, within their precision π) represents one boundary for this interval. The same is repeated in the other direction to identify the other border of the CDI. Figure 5 illustrates the tester operation and system reaction.

In this experiment the physical clock deviations of the four nodes under test were set to $200 \cdot 10^{-6}$, $200 \cdot 10^{-6}$, $-200 \cdot 10^{-6}$ and $-200 \cdot 10^{-6}$. The upper part of the figure shows the trajectory of the global clock (more precisely the cycle length) enforced by the tester, and the lower part the reaction of the system. During cycles 50 to 100 the tester synchronizes to the running system. Then it forces the cycle length to gradually increase to $5002\mu s$ (cycle 220) and stops transmitting. We can observe that in the "force free" system the cycle length stabilizes at $5001\mu s$ (cycle 280), which corresponds to the configured $5000\mu s$ plus $200 \cdot 10^{-6}$. The same experiment is started at cycle 470 for the lower boundary. We can observe that the cycle length stabilizes at the value of $5000\mu s$ minus $200 \cdot 10^{-6}$, as expected.

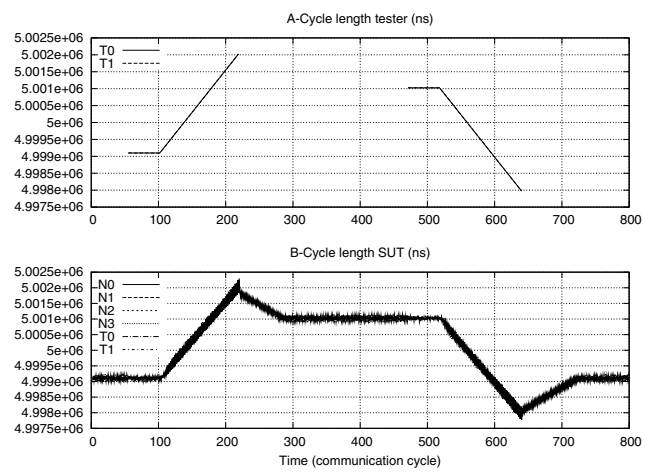

Fig. 5. CDI measurement for FlexRay

This approach has the main benefit of being quite simple in execution and result interpretation, but facilitates only a limited level of detail. In particular, it determines an interval in

which the physical clock deviation of all sync nodes is located – except for the extrema discarded in the FTM calculation. This can be fixed by keeping the tester occupy the position of one extremum during the "force free" phase. Then, the sync nodes that were rejected earlier are now taken into account for the computation. Still, however, diagnosis is restricted to sync nodes. In the following, we will present another method that facilitates a clock deviation measurement for all nodes, including none sync nodes and extrema.

B. Stepping the global clock

For our second approach we use the observation that the damping factor slightly shifts the rate correction towards the physical clock deviation (property P2 of Section III-A). This means that for a given (stable) global time, the nodes with a faster physical clock will also exhibit a slightly faster logical clock after the rate correction and thus have a positive offset correction, while the nodes with a slower physical clock will remain slightly slower and have a negative offset correction. This property leads to the following approach: The tester node performs a deterministic replay dictating a given cycle length. Then for each node the measured offset correction value provides information on whether its logical clock is faster or slower than the dictated global clock. This procedure is then iterated with different cycle lengths to determine an upper and a lower bound for each node's physical clock deviation. The steps applied between two successive measurements determine the size of the interval enclosed between these bounds and thus allow a trade-off between fine resolution and minimum experiment time.

For the following experiment, the oscillator deviations of the four nodes under test were set to $150 \cdot 10^{-6}$, $50 \cdot 10^{-6}$, $-50 \cdot 10^{-6}$ and $-150 \cdot 10^{-6}$. This corresponds to a raw cycle length (i.e. the uncorrected cycle length that would result from the physical clock solely) of 5000,75μs, 5000,25μs, 4999,75μs and 4999,25μs respectively. Figure 6 illustrates the results of our measurement: The upper part (A) represents the cycle length dictated by our tester, the second graph (B) the response of the system. The third graph (C) shows the offset correction values of the four nodes (measured as outlined in Section III-C), and the last graph (D) the filtered offset correction values (node 0 is highlighted).

The results confirm our concept. The overreaction of the offset correction observed at the beginning of each step is a result of the immediate change of the cycle length dictated by the tester. The interesting part, however, is to see whether the node's offset correction stabilizes in the positive or in the negative range and to observe for which step a sign change occurs. During the first phase of the experiment (until cycle 300), the raw cycle lengths of all four nodes are longer than the dictated cycle length. The resulting offset correction is therefore negative for all four nodes. Then, the offset correction of the first node gets positive, indicating that this node's raw cycle length has been crossed. Its raw cycle length can therefore be narrowed to the interval between 4999,15μs and 4999,35μs. This phenomenon can be observed for the

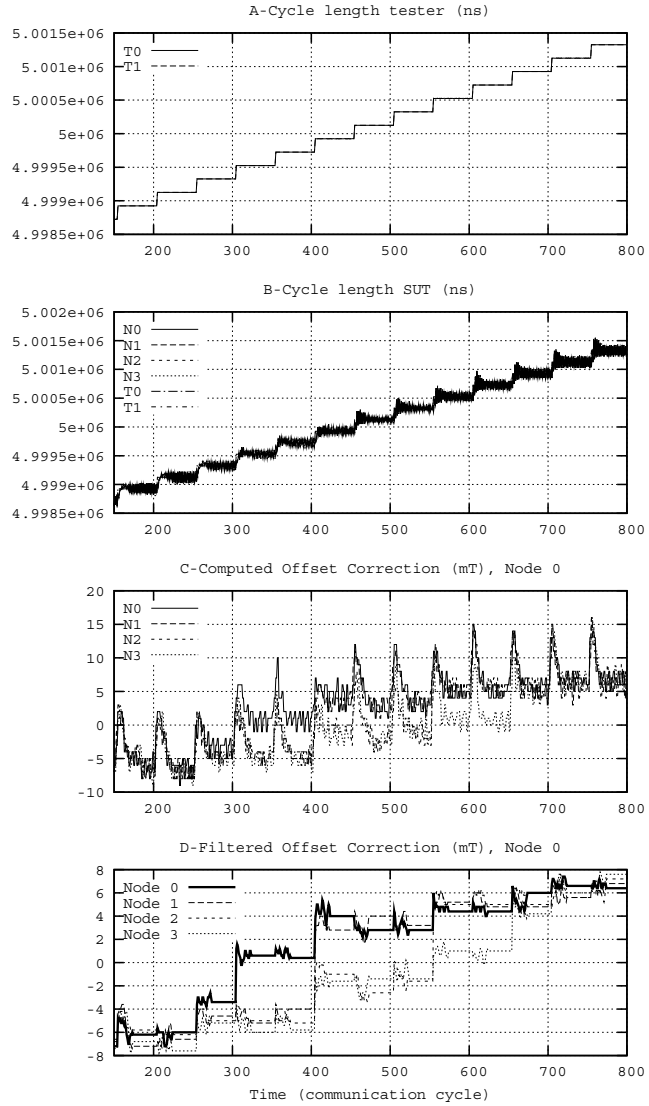

Fig. 6. Clock deviation measurement with steps

three other nodes around their raw cycle length, as well. The conclusion from the raw cycle length to the physical clock deviation is straightforward.

The main advantage of this second approach is to facilitate the detection of each individual physical clock deviation. It makes no difference whether the nodes are sending sync frames or not, or which nodes' values are discarded during the fault tolerant midpoint. The experiment duration depends on three parameters: (1) the cycle length (re-synchronization period), (2) the test range (interval spanned between the minimal and maximal values tested), and finally (3) the step size between consecutive measurements (which defines the measurement accuracy). For this example, the experiment lasted 4 seconds (800 cycles) for a test range of $\pm 1,5\mu s$ ($\pm 300 \cdot 10^{-6}$) around the configured cycle length of 5ms and a step size of 0,2μs (corresponding to an accuracy of $40 \cdot 10^{-6}$). Notice that a trade-off exists between test range, measurement

accuracy and experiment duration. However, the test duration does not depend on the number of nodes being tested.

The costs for the two methods presented in this work are three communication slots (since up to two time measurements might be thrown away during FlexRay clock synchronization) for the tester to send sync frames. The configuration parameter *gSyncNodeMax* (maximal number of sync nodes) has to be adapted accordingly. Notice that this requirement concerns the system architecture solely. Neither the node's hardware nor software needs to be modified, thus making this approach totally non-intrusive from the nodes' point of view. Moreover, the proposed methods are transparent for the system: The standard communication between the nodes is not disturbed and the additional tester frames are discarded by the application (since receive buffers are not explicitly configured to receive the tester frames).

V. Conclusion

Oscillators play a crucial role for the operation of time triggered distributed systems. Therefore it seems appropriate to spend some efforts for checking their integrity. The oscillators usually employed in digital circuits tend to exhibit accuracy errors due to ageing or adverse environmental conditions. These kinds of errors are usually more problematic than a complete failure of the oscillator. At the same time, however, testing an oscillator is usually very difficult in a deployed system, since it entails physical access to each node and often requires the application to be stopped.

We have devised methods for checking the deviation of an oscillator from its nominal value. These methods are based on stimulation and observation of the bus traffic, and therefore need access to the communication bus (at one arbitrary point) only. They can be applied on-line without disturbing the running application and are thus suitable for continuous monitoring. Their online applicability makes the proposed checks particularly promising for preventive maintenance: The drift behavior of all oscillators can be monitored, and in case of abnormal drift or excessive deviation repair can be initiated before an error actually occurs.

References

[1] J. Lundelius-Welch and N. Lynch, "A New Fault-Tolerant Algorithm for Clock Synchronization," *Information and Computation*, vol. 77, no. 1, pp. 1–36, 1988.

[2] T. K. Srikanth and S. Toueg, "Optimal Clock Synchronization," *Journal of the ACM*, vol. 34, no. 3, pp. 626–645, 1987.

[3] A. Ademaj, "Slightly-Off-Specification Failures in the Time-Triggered Architecture," in 7th *IEEE International Workshop on High Level Design Validation and Test , October, 2002, Cannes, France*, Oct. 2002, pp. 7–12.

[4] P. Milbredt, A. Steininger, and M. Horauer, "Automated Testing of Flexray Clusters for System Inconsistencies in Automotive Networks," in *Proceedings of the Fourth IEEE International Symposium on Electronic Design, Test & Applications*, Jan. 2008.

[5] H. Kopetz and G. Bauer, "The Time-Triggered Architecture," *Proceedings of the IEEE*, vol. 91, no. 1, pp. 112 – 126, Jan. 2003.

[6] , "Flexray Communications Systems – Protocol Specification Version 2.1a, http://www.flexray.com," FlexRay Consortium, 2005.

[7] H. Kopetz, *Real-Time Systems: Design Principles for Distributed Embedded Applications*. Norwell, MA, USA: Kluwer Academic Publishers, 1997.

[8] , "Time-Triggered Protocol TTP/C High Level Specification Document Protocol Version 1.1, http://www.tttech.com/technology/specification.htm," TTTech, 2005.

[9] H.-P. Company, "Fundamentals of Quartz Oscillators," in *HP Application Note 200-2*, 1997.

[10] E. Armengaud, "A transparent online test approach for time-triggered communication protocols," Ph.D. dissertation, Technische Universität Wien, Institut für Technische Informatik, Treitlstr. 3/3/182-1, 1040 Vienna, Austria, 2008.

[11] E. Armengaud, A. Steininger, and A. Hanzlik, "The Effect of Quartz Drift on Convergence-Average based Clock Synchronization," in *12th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2007)*, Greece, Sept. 2007, pp. 1123–1130.

[12] E. Armengaud, M. Fuegger, and A. Steininger, "Safe Deterministic Replay for Stimulating the Clock Synchronization Algorithm in Time-Triggered Systems," in *7th IEEE International Workshop on Factory Communication Systems (WFCS 2008)*, Dresden, Germany, May 2008, pp. 277–286.

[13] W. Forster and E. Armengaud, "A Novel Interconnection Approach for Globally Asynchronous Locally Synchronous Circuits," in *The 15th Austrian Workhop on Microelectronics (Austrochip 2007)*, Austria, Oct. 2007, pp. 107–114.