Improving Fault Tolerance in High-Precision Clock Synchronization

Georg Gaderer, Patrick Loschmidt, and Thilo Sauter, *Senior Member, IEEE*

***Abstract*—The very popular Precision Time Protocol (PTP or IEEE 1588) is widely used to synchronize distributed systems with high precision. The underlying principle is a master/slave concept based on the regular exchange of synchronization messages. This paper investigates an approach to enhance PTP with fault tolerance and to overcome the transient deterioration of synchronization accuracy during a recovery from a master failure. To this end, a concept is proposed where a group of masters negotiates a fault-tolerant agreement on the system-wide time and transparently synchronizes the associated IEEE 1588 slaves. Experimental verification on the basis of an Ethernet implementation shows that the approach is feasible and indeed improves the overall synchronization accuracy in terms of fault tolerance.**

***Index Terms*—Clock synchronization, computer network reliability, fault tolerance, IEEE 1588, time measurement.**

I. Introduction

THE distribution of time information is an essential element for distributed systems in many application domains like automation, test and measurement, or telecommunications. Rather than establishing a dedicated synchronization infrastructure, it is common practice today to use communication networks that link the distributed network nodes anyway. The spatially distributed clocks are then synchronized by means of dedicated network messages and protocols. This approach is the basis for the well known and widely used Network Time Protocol (NTP), which is one of the standard protocols used in distributed systems.

One drawback of NTP is its relatively moderate achievable accuracy in the range of 1 ms [1]. For distributed computer systems in IT environments, this is fairly sufficient. In recent years, however, there is a trend to employ networks (notably Ethernet-based real-time networks) in other domains that require higher synchronization accuracies. The test and measurement industry needs synchronized clocks for distributed and correlated data collection [2], [3]. The numbers of nodes in large-scale applications like high-energy physics experiments can reach several hundreds [4]. In other domains, like factory automation, synchronized clocks are needed in order to coordinate medium access in real-time networks [5]–[8] or for highly dynamic applications like electric drive control. Here, accuracies in the ms or even μs range are often desired and node numbers can be in the range of hundreds. To meet these needs, the Precision Time Protocol (PTP) was designed and standardized as IEEE 1588 in 2002 and IEC 61588 in 2004 [9], respectively. It was devised for highly accurate clock synchronization focused especially on test and measurement, as well as power engineering (e.g., substation automation) and factory automation. New requirements not foreseen in the original work soon led to a revision of the protocol and a new version of the standard [10].

PTP adopts a straightforward master/slave synchronization method. The master is elected upon system startup based on the accuracy of the node's local clock and subsequently distributes its local time to the slaves on a periodical basis. Transmission delays over the network are measured and can therefore be taken into account. In fault-free environments, PTP shows a good long-term stability and is deterministic and convenient to use. The simplicity of the approach is however also a problem, since the master is a single point of failure. If this happens, a master re-election similar to the procedure for power-up has to be conducted and leaves the network without timing for several synchronization periods.

The purpose of this paper is to propose a remedy for this obvious deficiency. In the suggested democratic approach using so-called *Master Groups*, the master is distributed and virtualized in a group of nodes with sufficiently accurate clocks. Contrary to the single-master standard, failure of one group member is not a problem, since the group can still find a fault-tolerant time value. Apart from increased fault tolerance, the benefit of the proposed solution is its backward compatibility with standard PTP slaves. This paper investigates the behavior and efficiency of the democratic solution and compares it to standard IEEE 1588. Section II reviews the most essential elements of standard PTP with respect to fault tolerance. Section III discusses general limitations for the accuracy achievable by the standard IEEE 1588 protocol. Section IV introduces the proposed democratic Master Group approach. The experiments described in Section V compare the standard and distributed master behavior by generating faults into the master or the Master Group and measuring their effect on an attached group of slaves. Finally, Section VI presents conclusions and an outlook.

II. The Precision Time Protocol

The starting point for the protocol specified in IEEE 1588 is a distributed set of nodes. The protocol is defined in an abstract way and the underlying communication network as such is a priori not relevant. Nevertheless, a detailed analysis of actual implementations of clock synchronization protocols shows that

Manuscript received September 30, 2009; accepted February 21, 2010. First published March 22, 2010; current version published May 05, 2010. This work was supported in part by the province of Lower Austria (Amt der NÖ Landesregierung – WST3) and the European Regional Development Fund. Paper no. TII-09-09-0220.

The authors are with the Institute for Integrated Sensor Systems, Austrian Academy of Sciences, 2700 Wiener Neustadt, Austria (e-mail: Georg.Gaderer@OEAW.ac.at; Patrick.Loschmidt@OEAW.ac.at; Thilo.Sauter@OEAW.ac.at).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2010.2044580

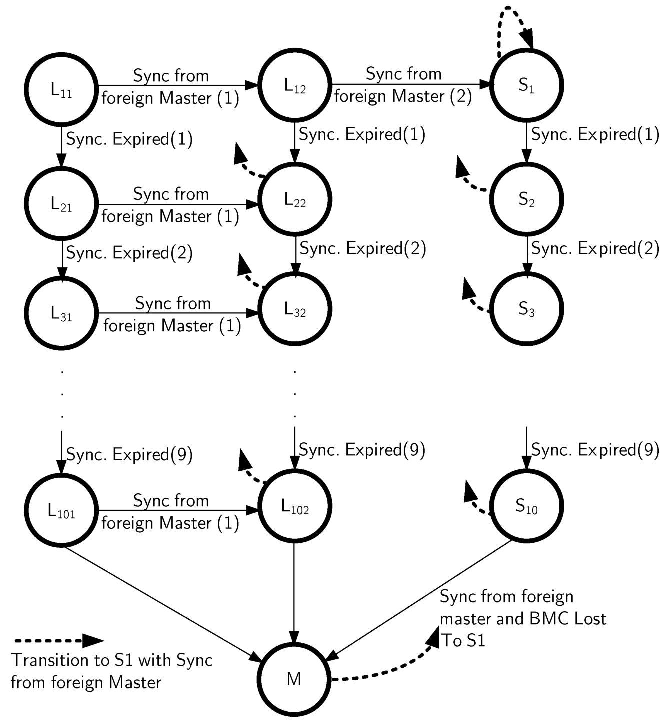

L11
L12
S1
Sync from foreign Master (1)
Sync from foreign Master (2)
Sync. Expired(1)
L21
L22
S2
Sync. Expired(2)
L31
L32
S3
Sync. Expired(9)
L101
L102
S10
Sync from foreign master and BMC Lost To S1
Transition to S1 with Sync from foreign Master
M

Fig. 1. State transition diagram of a PTP master election.

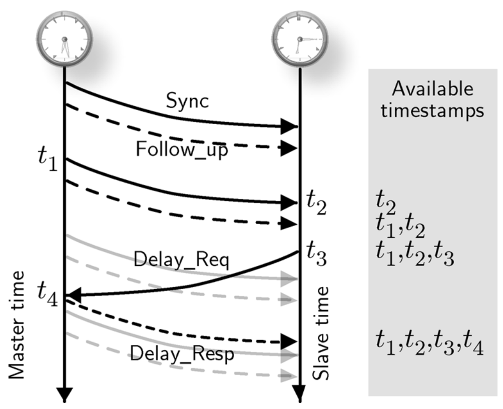


Fig. 2. Message sequence during normal PTP operation.

certain definitions are needed to solve interoperability issues. Among these are exact specifications of where timestamps are drawn upon message arrival (in order to accurately measure the transmission delays), as well as specific assumptions of the underlying transport protocol – such as the port number if the User Datagram Protocol (UDP) is used. As a matter of fact, most implementations of PTP are based on Ethernet and UDP/IP, even though the basic mechanisms are rather generic.

A. PTP Master Election

In standard PTP, a dedicated master clock synchronizes a group of slaves. This clock can be either a so-called *boundary clock*, which is a clock synchronizing itself to another PTP clock or a PTP *grandmaster*, which is the ideally best synchronized node in a whole system, as for instance a node attached to a GPS receiver or using an atomic clock as a reference. After system startup, this master needs to be identified or, in PTP terminology, elected. The concept of master election presumes that a node itself has knowledge about its accuracy, which is implemented in a statically configured variable (called *stratum* in PTP version 1 or *clockClass* in version 2).

Each node starts in the status INITIALIZED. The only valid transition from this state is to the state LISTENING (denoted $L_{i,j}$ in Fig. 1). In this state, the node listens for subsequent synchronization messages. In these messages, nodes broadcast their stratum values to compete for the master role. If a node receives two messages indicating better stratum values than its own clock, it changes into the SLAVE state. If the stratum value is equal to its local one, the decision is based on a unique node identifier (layer 2 MAC address in the Ethernet case). On the other hand, if no synchronization message is received for ten synchronization intervals, the node considers itself the master and switches to the MASTER state.

In the MASTER state (M in Fig. 1) a PTP node starts multicasting synchronization messages to all other nodes in the PTP domain. The MASTER state can only be left if a synchronization message from another node is received and the analysis shows that the node's clock is less accurate than the clock from which the synchronization message was received. The node then transits into the SLAVE state. A node which is in the slave state (S_i) may transit into MASTER state as soon as no synchronization event is received for ten subsequent synchronization periods. This is what happens if a master fails during normal operation.

B. Clock Synchronization in PTP

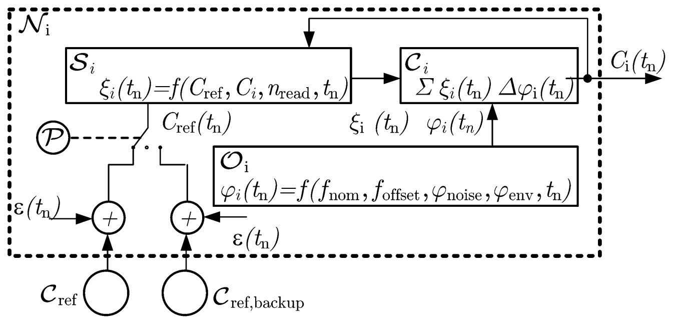
After the proper master is elected, the actual synchronization can take place. The key to achieving high accuracy is the knowledge of network timings, in particular the delays. The synchronization process in IEEE 1588 networks is therefore organized in two stages. A so-called *delay request/delay response* process measures the round-trip delay [11] to determine the packet delay between master and slave (Fig. 2). This is done upon initiation by the slave with a telegram which is returned by the master with its local timestamp of reception. Together with the timestamps drawn by the slave when sending the request, the slave may easily calculate the communication delay (which is assumed to be symmetric).

In addition, the master regularly sends synchronization packets. The precise timing information is sent in a two-step process. First, the packets are timestamped with an estimate of the sending time. The actual sending time (which consequently can only be known after the completion of the transmission of a packet) is sent with a so-called *follow_up* packet in the second step. A detailed description of this process together with implementation issues can be found in [12].

III. System Model

A. Formal Clock Synchronization Node Model

To analyze the behavior of a clock for the following analysis of the PTP, a model of a node has to be established first. Fig. 3



$\mathcal{N}_\mathrm{i}$
$\mathcal{S}_i$
$\xi_i(t_\mathrm{n})=f(C_\mathrm{ref}, C_i, n_\mathrm{read}, t_\mathrm{n})$
$\mathcal{C}_i$
$\Sigma\, \xi_i(t_\mathrm{n})\, \Delta\varphi_\mathrm{i}(t_\mathrm{n})$
$C_\mathrm{i}(t_\mathrm{n})$
$C_\mathrm{ref}(t_\mathrm{n})$
$\xi_\mathrm{i}\ (t_\mathrm{n})$
$\varphi_i(t_n)$
$\mathcal{P}$
$\mathcal{O}_\mathrm{i}$
$\varphi_i(t_\mathrm{n})=f(f_\mathrm{nom}, f_\mathrm{offset}, \varphi_\mathrm{noise}, \varphi_\mathrm{env}, t_\mathrm{n})$
$\varepsilon(t_\mathrm{n})$
$\varepsilon(t_\mathrm{n})$
$\mathcal{C}_\mathrm{ref}$
$\mathcal{C}_\mathrm{ref,backup}$

Fig. 3. Structural model of the time generated by a node.

shows the principle structure of a node $\mathcal{N}_i$ under a general synchronization scheme. It consists of three basic elements: An oscillator $\mathcal{O}_i$ is taken as frequency reference for the actual clock $\mathcal{C}_i$. The general operation principle is to control the clock value by adding increments at every oscillator tick given by $\mathcal{O}_i$. This variable increment ξ_i – defined and set by the servo $\mathcal{S}_i$ – is subject to adjustments in order to compensate for imperfections of the oscillator.

To adjust the increments and thus the clock, the servo – as a controller – must be able to compare the actual clock value $C_i(t)$ to a reference time $C_{\mathrm{ref},i}(t)$ acting as set value. The generation of this reference is the task of the clock synchronization scheme $\mathcal{P}$. This includes not only the actual transmission of the reference time over the network, but also a possible combination of different sources, assessment, and selection of an appropriate value on an algorithmic basis. Thus, the reference clock $\mathcal{C}_\mathrm{ref}$ and possibly also a backup reference $\mathcal{C}_\mathrm{ref,backup}$ can only be considered as *remote* clocks. In the simple case of master/slave based clock synchronization according to PTP, the reference clock will be the master and $\mathcal{C}_\mathrm{ref,backup}$ can be seen as the clock on another node which will be elected as new master by the (essentially distributed) algorithm $\mathcal{P}$ if $\mathcal{C}_\mathrm{ref}$ fails.

On the other hand, if a democratic approach is assumed, the actual time is a combination of several clock values in the network (including the local $\mathcal{C}_i$) and the backup reference is not used. The reference time is therefore not derived from a physical quantity with continuous behavior, but calculated on a regular basis. $\mathcal{C}_\mathrm{ref}$ can therefore be considered only as the remote readout of a clock or the remote readout of an *ensemble* time. This is why the absolute time in Fig. 3 is denoted t_n, so as to indicate that the values in general can be taken only at discrete points in time, rather than continuously. This in turn stems from the fact that only packet-oriented networks are considered for the transmission of clock values, which necessitates some form of discrete-time sampling. Therefore, adjustments to the local clock can be made only at certain points in time.

Outside the node, also the influence of the clock reading error $\varepsilon(t)$ has to be considered. Owing to this systematic error, a remote clock cannot be read out in a deterministic way, but only as a stochastic process. Not only does this include the delay jitter, it is also affected by the fact that remote clocks, even if they might run internally as a perfect clock, are not readable without error. It is important to model the clock reading error as a property of the node which reads the reference clock and not of the reference clock itself since it can be different at every node.

B. Influence Factors for Clock Synchronization Accuracy

For further analysis we assume two nodes $\mathcal{N}_1$ and $\mathcal{N}_2$, with clocks $\mathcal{C}_1$ and $\mathcal{C}_2$. The clocks shall be synchronized by the exchange of clock synchronization messages, so that for discrete sampling points $\{t_n = T_\mathrm{sync} n, n \in \mathbb{N}\}$ the value $C(t_n)$ can be corrected to fulfill $C(t_n) = t_n$ for external clock synchronization (ideal accuracy), or $C_1(t_n) = C_2(t_n)$ for internal clock synchronization (ideal precision). These periodic synchronization intervals are scheduled every T_sync seconds.

In order to calculate the offset of a clock after each synchronization interval, its local oscillator has to be considered as a time reference. If one assumes a stochastic model for the oscillator imperfection, the phase $\varphi(t_n)$ of an oscillator running with nominal frequency f_nom and the frequency offset f_off can be modeled as in [13],

$$\varphi(t_n) = (f_\mathrm{nom} + f_\mathrm{off})(t_n - t_0) + \frac{1}{2}a(t_n - t_0)^2 + \\ + \varphi_n(t_n) + \varphi_\mathrm{e}(t_n), \quad (1)$$

where f_nom is the nominal oscillator frequency, f_off is the frequency offset, a models the aging, $\varphi_n(t_n)$ the contribution of the nonenvironmental noise and $\varphi_\mathrm{e}(t_n)$ the environmental noise. In this consideration, the main contributing error factors are functions of the absolute time t_n or the time elapsed since the start of the observation of the oscillator $t_n - t_0$.

A clock using such an oscillator as reference computes a weighted sum of the oscillator ticks. Real implementations usually perform this summation not only once per synchronization interval, but at every rising edge of the oscillator output. However, considering that during a synchronization period no corrections in the increment ξ_i are reasonably made, the calculation of the clock value can be done in one step taking only synchronization intervals into account,

$$C(t_n) = \sum_{i=0}^{n} \xi_i \left(\varphi(t_i) - \varphi(t_{i-1})\right). \quad (2)$$

If adjustments to the clock increments ξ_i are made only after a synchronization interval T_sync, the clock value after n intervals (and neglecting environmental noise terms) is given by (3) at the bottom of the page.

Imperfections of the oscillator like frequency offset or (time-varying) drift must be leveled out by the servo $\mathcal{S}$ by controlling the increment ξ_i.

$$C(t_n) = \sum_{i=0}^{n} \xi_i \left\{(f_\mathrm{nom} - f_\mathrm{off})T_\mathrm{sync} + \frac{1}{2}a(t_i^2 - t_{i-1}^2 + 2t_0(t_{i-1} - t_i)) + \varphi_n(t_i) - \varphi_n(t_{i-1})\right\} \quad (3)$$

Despite this influence of the local oscillator, also the behavior of the network and timestamping itself have to be considered. This is usually done in the clock reading error $\varepsilon(t)$. The error is caused by several systematic inaccuracies which degrade the capability of reading a remote clock and, therefore, the ability of adjusting a local clock to a reference. It heavily depends on the actual hardware implementation and physical layer technology but, in general, the following factors influence it:

- The communication jitter, uncertainty of the delay of a message: Although absolute delay can be measured by round trip measurements, the statistical influence of jitter on this measurement and on synchronization messages has to be considered as well [14].
- The clock reading error of outgoing messages: Usually, outgoing messages are timestamped at the moment when they are leaving the node. This accuracy of timestamping (including the ability to sample the actual leaving of the message) contributes to the uncertainty.
- The clock reading error of inbound messages, which is generally the same effect as above, referring to the ability and uncertainty of timestamping incoming messages.
- Finally, the asymmetry of the communication channel [15]: As round-trip delay measurements rely on a communication channel which is considered to have equal delays in both directions, any asymmetry contributes to errors. In the Ethernet case, they are caused by the usage of different cable pairs.

C. Delay Corrections

The PTP defines that unlike normal synchronization, delay measurements are performed not every synchronization period. The reason for this is the assumption that the message transmission delay does not change as fast as the nodes' clock offset. The delay compensation is initiated by the slaves. However, if a delay request would be issued after every n^{th} synchronization packet, the master would receive a delay request storm within a relatively short period of time from a potentially large number of nodes. This is a problem as the round-trip delay measurement relies on a relatively short residual time at the master node.

To overcome the problem of a delay request storm at the master, the messages are issued in a random manner over a defined interval. PTP does this in two stages. First, a random integer number is drawn from the range between 2 and 30 (which is the upper bound PTP_DELAY_REQ_INTERVAL as defined in the standard) and multiplied with the synchronization interval to give the delay request interval T_{DRQ}. Furthermore, it is preceded by the slot time T_{SLOT}, a fraction of the synchronization interval. The delay measurement is thus taken every

$$T_{\text{DM}} = \text{rand}(2,30)\,T_{\text{sync}} + \frac{T_{\text{sync}}}{\text{rand}(2,18)} = T_{\text{DRQ}} + T_{\text{SLOT}} \tag{4}$$

seconds. The random numbers $\text{rand}(a,b)$ are supposed to be uniformly distributed and they are computed anew for every measurement cycle. This randomization, therefore, ensures that the interval for delay measurements varies between minimum $2\,T_{\text{sync}}$ and maximum $31\,T_{\text{sync}}$.

D. Failure Model

The reasons for a clock failure can be manifold: either the communication link to a node can fail or the clock of a node itself. A faulty clock can have two possible results: either the *clock rate* is arbitrarily (and maybe changing with time) wrong or the clock is stuck at a certain value. The former case can be dealt with by the clock synchronization algorithm $\mathcal{P}$, as long as the PTP master is not affected (because the slaves would be synchronized to this faulty clock). The latter case can be reduced to a communication failure. The algorithm can do this by checking the readout values of the clock. If the same value is read twice, the clock must be faulty and the algorithm can declare the node as unable to communicate. Probably the most complicated case is the Byzantine fault: A clock can fail in a way that it shows completely wrong time. In the generalized case, the readout of $C_i(t)$ by different nodes can even show different values. This can be avoided by the introduction of broad- or multicasts within a system. However, the problem that a clock can report randomly wrong values has to be tackled by $\mathcal{P}$.

The failure model relevant to the discussion and improvement of fault-tolerance mechanisms in PTP may therefore concentrate on failures of the communication link between the master node and the network. Faults occurring in the synchronization scheme can be reduced to a packet loss of one or more synchronization messages from the master to the slave(s). As it is always assumed that a master which is externally synchronized (e.g., via GPS) synchronizes its slaves internally, a stopped master clock would have the same effect (because of the time triggered synchronization messages) as the loss of all master packets.

IV. Democratic Clock Synchronization for PTP

As mentioned in the introduction, a failure of the PTP master causes a temporary loss of synchronization until a new master is elected. Detecting that the master is no longer present takes the slaves ten synchronization intervals. The subsequent phase during which a new master is established may take another few synchronization periods depending on the network complexity. When the new master is settled, it may take up to 31 synchronization periods until the delay measurements are completed and synchronization is regained. During this time, the clocks of the individual nodes run freely or in partial synchronization in subgroups.

The master failure in the context of these considerations need not necessarily be a failure of the node itself. It is already sufficient if the communication to the master is lost – even as a transient effect. As soon as the master is not reachable for ten synchronization intervals (or if ten subsequent synchronization messages are lost), a master failure has occurred because this will provoke a master re-election and thus a synchronization failure.

A. Democratic Efficiency Considerations

The obvious solution is a democratic approach, where the reference time is not determined by one single master, but by a fault-tolerant averaging of all clock values in the network. Depending on the actual algorithm, such a strategy can tolerate a number of failures without impairing the quality of the resulting network-wide time. Over the years, a number of algorithms have been defined for computer networks [16]–[18].

A completely democratic architecture, however, does not seem reasonable for network load reasons. In a network with n participants, every node has to tell the remaining $(n-1)$ peers its local time together with a confidence interval. All other nodes need to distribute that time too, and therefore $M_{\text{democ}} = n(n-1)$ unidirectional communication links have to be established to distribute the synchronization data. This is significantly more than the $(n-1)$ communication relationships required for the strict master/slave principle of IEEE 1588. Furthermore, the telegram efficiency η decreases with the number of nodes

$$\eta_{\text{democ}} = \frac{n-1}{n(n-1)} = \frac{1}{n}. \quad (5)$$

The efficiency here is the fraction of the communication overhead in terms of required synchronization messages for the purely master/slave and the democratic approach. The master/slave approach has the least communication requirements and therefore serves as a reference. A fully democratic solution is therefore rather inefficient.

B. Master Groups

The previous considerations lead to a three-level architecture consisting of hierarchically structured synchronization subnets instead of a flat democratic structure. The reference time is broadcasted by GPS satellites and atomic clocks, respectively. The GPS receivers, which are considered as reference clocks, are coupled directly to the nodes of the Master Group which can be interpreted as a fully democratic subnet where each member of the group talks to all others. This approach has the advantage that a failure of a single master has hardly any influence on the associated slaves, except that the overall accuracy of the Master Group is reduced. From a technical viewpoint, the communication inside the Master Group and between Master Group and the slaves is implemented via two separate multicast domains.

The internal synchronization used within in the Master Group employs the fault-tolerant average [16] which synchronizes all nodes. The nodes are tied to the Temps Atomique International (TAI) timescale in order to avoid the irregularities of the common Coordinated Universal Time (UTC) timescale. The members of the Master Group regularly broadcast their local time values. Each incoming broadcast message is timestamped at the receiver and the receiving nodes can, therefore, calculate the offset of their local clocks with respect to the broadcast time values. The resulting information about the mutual offsets inside the ensemble is then evaluated using the Fault-Tolerant Average Algorithm (FTA). The algorithm sorts the differences between the clock values and discards the k lowest and highest values. The remaining data sets are taken and an average is calculated. Within a group of n nodes, up to k faulty ones can therefore be tolerated.

In the proposed approach the transmission of the fault-tolerant time average computed within the Master Group to the lowest level takes place via a so-called Master Group speaker, represented by the switches in Fig. 4 for an Ethernet-based implementation. This speaker ensures compatibility with the PTP

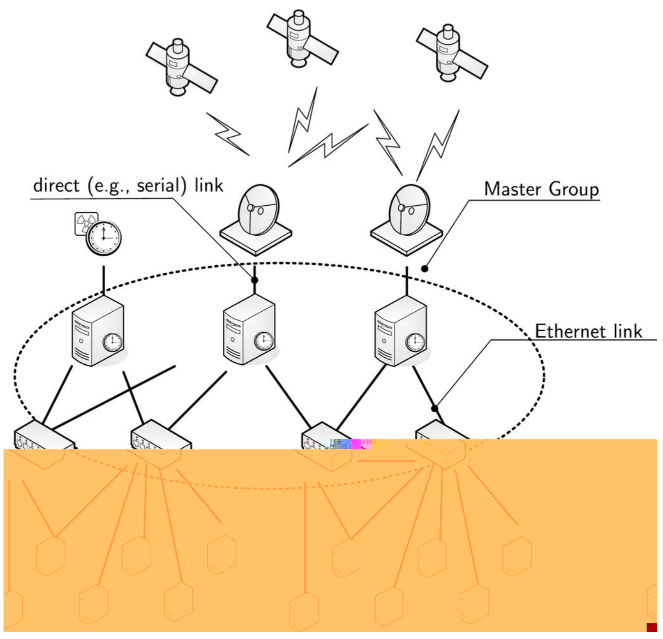


Fig. 4. Master Group concept.

standard. To further enhance fault tolerance, the switches for the cross-linking of the Master Group can also offer the possibility for redundancy on the physical and on the protocol layer.

The group speaker communicates with a set of standard IEEE 1588 slaves, which ensures low traffic volume (compared to the reference clock interconnection) even for large numbers of slaves. The associated master for each node is the speaker of the superordinate group which acts transparently like an IEEE 1588 master and passes the group-wise agreed time from the group downward. The core idea of this approach is to enhance IEEE 1588 networks with this transparently integrable Master Group to a hybrid architecture in order to increase stability and fault tolerance.

The efficiency of the communication overhead for the hybrid approach of m masters (including the group speaker) interacting with n slaves is then

$$\eta_{\text{hybrid}} = \frac{(m-1)+n}{m(m-1)+n} = \frac{m+n-1}{m^2-m+n}. \quad (6)$$

Note that for the likely case of only a few masters synchronizing a substantially larger number of client nodes ($m \ll n$) the complexity approaches the one of the conventional master/slave method.

As in every distributed algorithm, the membership problem is an issue for the Master Group and the FTA. Depending on the actual network topology, it might happen that due to particular link failures, the Master Group is separated into two or even more subgroups, which might each be synchronized internally, but not with respect to each other. The FTA can address this membership problem only to a certain extent because it was designed to tolerate the loss of group members. Nevertheless, with respect to the group of PTP slaves associated with the Master Group, the problem is alleviated in the sense that only the subgroup containing the group speaker will be relevant for the overall synchronization. This means that the basic functionality of the Master Group consisting of at least k functional nodes will be retained, albeit with reduced fault tolerance.

V. Experimental Verification

In order to test the proposed Master Group approach and compare it with the standard fault tolerance mechanisms in PTP, experiments need to be conducted in an IEEE 1588 test bed. The experiments aim at manually generating faults in a master/slave and a Master Group configuration with the goal of disturbing or disrupting synchronization between master(s) and slaves. This is done in three different ways:

- Emulating master clock failures by disconnecting the network: In the standard PTP case, the loss of the master is expected to trigger a master re-election with an intermediate loss of synchronization between the nodes. In the Master Group implementation, disconnection of one Master Group member is expected to have no appreciable effect at all.
- Environmental changes by influencing node temperatures: Although not representing an actual failure, temperature changes distort the frequency of the oscillator. Although the clock synchronization algorithm is designed to correct this, a sudden change in a single PTP master configuration is expected to disturb the slave nodes. In the Master Group, the time value of the affected group member will be discarded by the fault-tolerant averaging algorithm.
- Insertion of new nodes: In the standard PTP setup, inserting a node with higher clock accuracy than the current master will trigger a master re-election which has the same effect as a master failure. In the Master Group, a new member is not expected to cause a disturbance because it will be simply integrated in the group.

Overall, the experiments are expected to demonstrate that the Master Group approach will be able to tolerate the generated faults and thus provide more fault tolerance than standard PTP.

A. *Experimental Setup*

Measurements in a distributed clock synchronized system are not straightforward. The main problem is that the actual clock value is usually not directly observable, because the clock registers inside the node controllers can only be partially read. For this reason, signals triggered at each full second (pulse per second, 1 PPS) are made available for observation. Thus, the difference between the node clocks is represented by the mutual offset of the 1 PPS pulses. This timing can then easily be measured by a high-resolution time-interval counter. Usually, those systems are limited in the number of concurrent measurements. Therefore, it is more convenient to sample all 1 PPS pulses with a high-speed sampling data acquisition system. However, a full coverage of all nodes is not always necessary and economically implementable.

The strategy employed for the experimental investigation of the clock synchronization behavior was to use a four-channel digital storage oscilloscope (LeCroy wavepro 7300A, 3 GHz, 10 GS/s) to collect and observe one reference clock (trigger) and four devices under test, i.e., four Master Group members and one randomly chosen slave node. Fig. 5 shows the structural design of the testbed.

For the actual measurement the trigger is typically connected to a reference node in the Master Group. However, a compensation for node errors has to be done: the frequency of the oscillator of the reference node has to be monitored in order to identify instabilities, which of course influence the trigger-to-trigger time of the test setup. This frequency stability measurement is not done directly, but by a resolution enhancing method. The signal of the oscillator is multiplied with a rubidium (Stanford Research Systems FS725) sourced frequency standard with the same nominal frequency. Thus, if the oscillator under observation has a small offset $\Delta\omega$ to its nominal frequency ω_1, a multiplication of a virtually ideal and GPS-disciplined rubidium (ω_2) and an imperfect oscillator on the boards gives a beat signal

$$U(t) = A_1 A_2 \left\{ 2\sin\left(\frac{\omega_1 + \Delta\omega + \omega_2}{2} t + \frac{\varphi}{2}\right) \cdot \cos\left(\frac{\Delta\omega}{2} t - \frac{\varphi}{2}\right)\right\} \quad (7)$$

where φ is the arbitrary phase offset between the two oscillators at $t = 0$ s and A_1 and A_2 the amplitudes of the two mixed signals. The resulting beat signal can be considered as an amplitude modulation of a sine function with the frequency $(\omega_1 + \Delta\omega + \omega_2)/2$ with a sine with the much lower beat frequency $\Delta\omega/2$. With proper low-pass filtering, the envelope can be extracted, and the frequency offset $\omega_1 - \omega_2$ can be measured with a much higher resolution than by observing the 10 MHz signal directly. The final measurement of this frequency is done with a rubidium and GPS-disciplined frequency counter (Stanford Research Systems, SR620).

The rest of the evaluation platform contains a backbone network which is used for managing of nodes in order to maintain a stable network connection even if the timestamping network cards are shut off. Further, a Network File System (NFS) server provides a common directory for a consistent code basis for all nodes within this test bed.

The most important parameter for the PTP synchronization algorithm is the synchronization interval T_{sync}, which was set to 2. This value is typical for actual implementations of PTP on Ethernet and is also the default value defined in the IEEE 1588 standard.

B. *Results*

1) Pure IEEE 1588 Clock Failure: As discussed before, clock failures are emulated by disconnecting the respective nodes from the network. With respect to the PTP structure, two possibilities have to be evaluated. On the one hand, if a slave fails, the other nodes are hardly influenced. The only observable consequence for the master would be that no more *delay_req* messages are issued by this slave. Therefore, only the random runaway of the faulty slave can be observed.

The case of a master failure, on the other hand, is completely different. This error causes a master re-election. In a real-world implementation this condition means that the new master drifts freely away from the other nodes, which after some time start synchronizing again. In Fig. 6, this resynchronization takes place after a master election at approximately $t = 50$ s. After this, a signal step in the control loop behavior can be seen, where the two other slaves try to catch up with the new master. During this time ($t = 50 - 80$ s), node 1 stabilizes at a constant

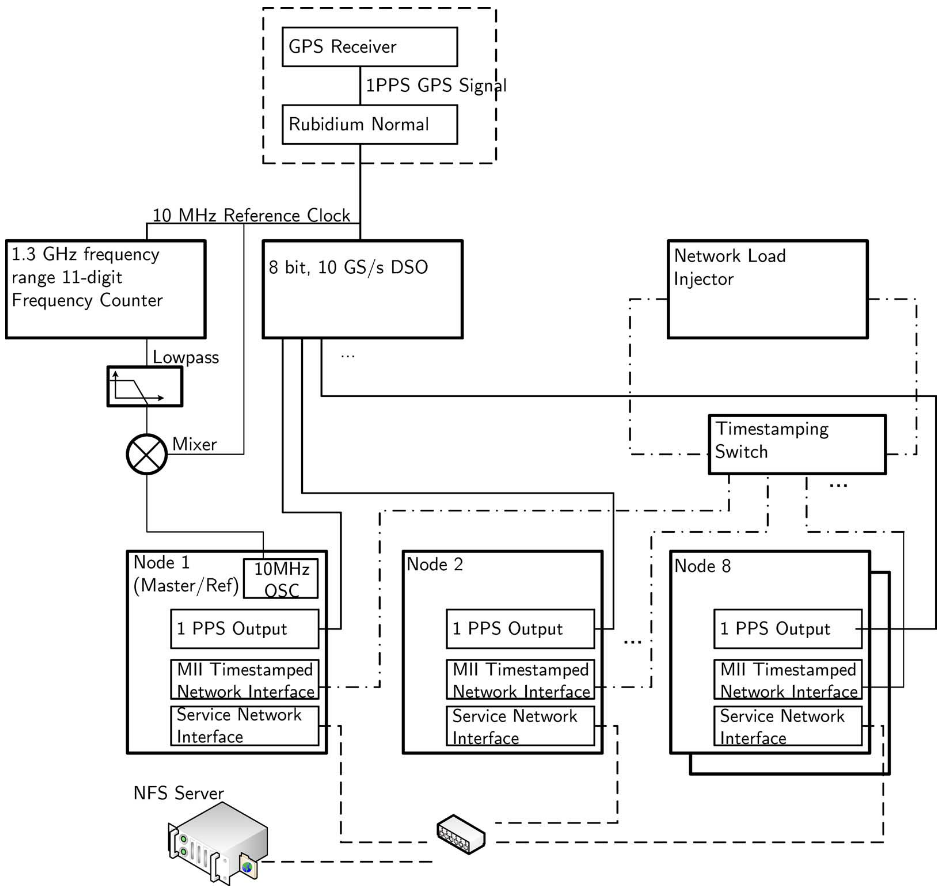


Fig. 5. Hardware measurement setup and testbed. This figure shows only the principle of the setup. In the actual measurements 4 devices under test were connected to the data acquisition device and four more were connected to the network.

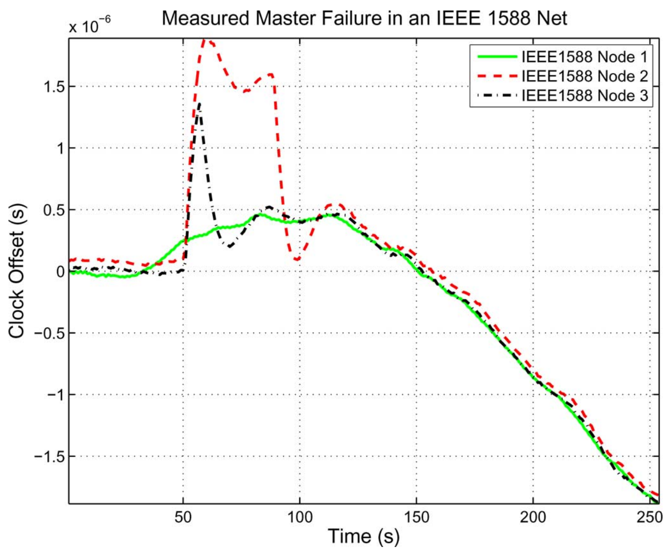


Fig. 6. Master failure in an IEEE 1588 network. For this experiment the communication of the master was cut off. The (former) master defines the trigger for the measurements. Therefore, the nodes depart from the zero-offset line after synchronizing to the new master (node 1).

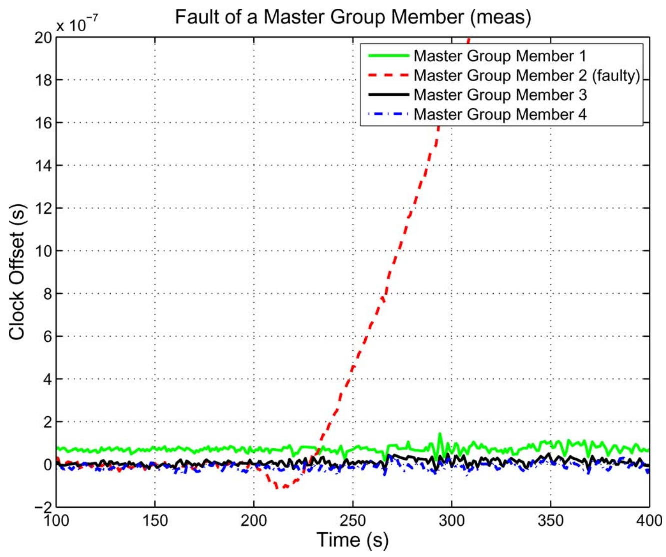
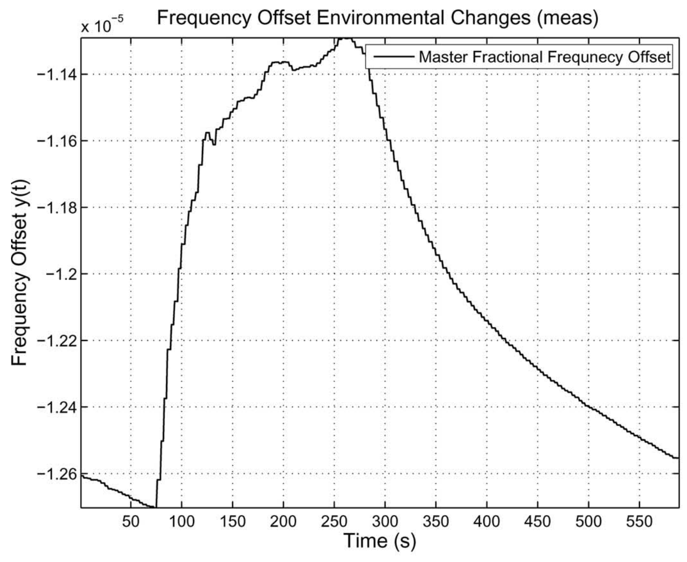
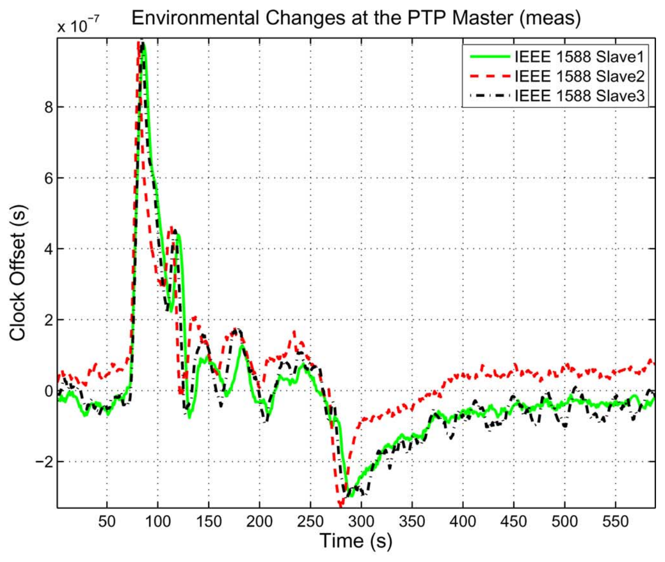


Fig. 7. Error of a Master Group member. The faulty node has no influence on the rest of the ensemble.

offset which is caused by an improper delay measurement between node 2 and the (new) master. The fact that this is not observable for node 3 can be explained by the (randomly chosen) earlier delay measurement of this node. Afterwards ($t \geq 120$ s), all remaining nodes in the network are synchronized. The reason that they depart from zero offset is that the measurement is done with respect to the old (now disconnected) master. The noise in the steady-state offset of the nodes stems from the clock reading error as discussed in Section III.



(a)



(b)

Fig. 8. Influence of a frequency change at the master oscillator. For this experiment, the master PC was cooled down by 2.8 °C with compressed air. (a) Fractional frequency change. (b) Slave clock offset.

2) Clock Failure in Master Group: Like in the experiment with the master failure in a standard IEEE 1588 network, the failure of a Master Group member can be forced by simply unplugging it from the network. Fig. 7 shows the resulting behavior. After pulling of the plug at $t = 200$ s, the faulty node strides away. As in the case with the free running clocks, the time shown by the disconnected clock exhibits a random walk pattern.

The rest of the Master Group shows the expected behavior. The three remaining nodes remain synchronized all the time. Not even at the instant of unplugging node 2 can a transient synchronization error be noticed. A PTP slave attached to the Master Group would not experience a disturbance either. This demonstrates that the Master Group is superior to the standard PTP master election algorithm.

3) Temperature Change in Standard IEEE 1588: Changes of environmental parameters at a node bear similarities to some clock faults. If the temperature inside the nodes changes, the frequency is influenced as well because of the simple oscillators used. This obvious imperfection can be used to provoke frequency changes at a node. For standard PTP, the investigation can again be limited to an analysis of a frequency change at the master.

For the experiments shown in Fig. 8, a PTP master node was disturbed by blowing pressurized air into the housing, which caused the internal temperature to drop from 30.6 °C to 27.8 °C. After this phase (blow of air from $t = 80-280$ s) the roughly exponential thermal relaxation takes place. The frequency change effect can be observed in Fig. 8(a). The fractional frequency offset is defined as $y(t) = 1 - f(t)/f_{\mathrm{nom}}$.

As observed in Fig. 8(b), the clock offset reaches almost 1 μs at the moment of the sudden temperature change. Although this error is corrected by the clock synchronization algorithm within 50 s below a 400 ns bound, the problem of the offset peak still remains. Also the beginning of the warming phase – which is again characterized by a fast change of temperature – can be seen at the slave side ($t \geq 280$ s).

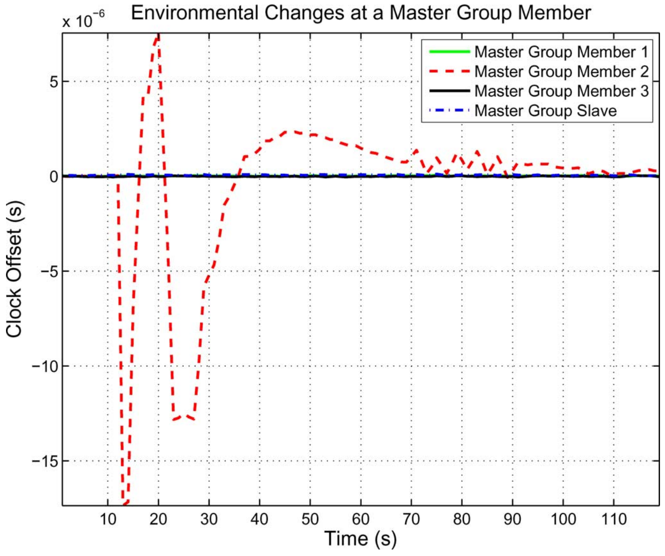


Fig. 9. Environmental change at a Master Group member. The faulty node has no influence on the rest of the ensemble and the Master Group slave. At the time $t = 12$ s, the Master Group member was changed in temperature.

4) Temperature Change in Master Group: Like in the standard PTP case, if the temperature and thus the clock frequency of a node is changed, also the clock values get off the linear timescale agreed and accepted within the ensemble. The system under test is influenced similarly to the experiment with standard PTP. The results are depicted in Fig. 9. To create a larger temperature drop, the cool-down was intensified by using a cooling spray. Thus, the clock offset of the affected node with respect to the ensemble average could be shifted to the range of several μs. Nevertheless, the fault-tolerant averaging algorithm detects the significant deviation of the clock value and discards

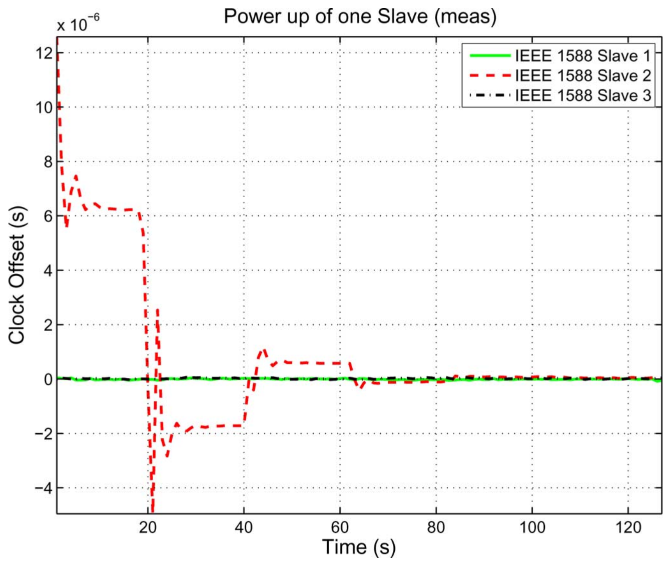


Fig. 10. Transient error of a slave node (node 2) which is added into a running IEEE 1588 network. It is shown that the running nodes are not influenced and that the final accuracy is reached after several delay measurements.

it. Therefore, this even stronger disturbance (compared to the master/slave approach) has no observable influence, neither on the rest of the ensemble, nor on the slave.

5) Insertion of New PTP Nodes: The addition of a node to a running PTP network (irrespective of whether it is operated in standard mode or with a Master Group) does not influence the rest of the network as long as the new node does not take over the master position. Fig. 10 shows the power-on of a slave. Again, the offset shown in this figure has to be interpreted with respect to the master. The synchronization behavior of the node, which is powered on at $t = 0$ s, exhibits three steps in the clock offset. These steps can be explained by delay measurements, which are taken during a period when the clock is accelerated in order to catch up with the absolute time. This is an implementation-specific issue of the clock controller algorithm and could be avoided by taking only delay measurements after the first clock correction phase into account.

Not shown as a diagram is the corresponding experiment of adding a new master to the single-master network, since the behavior is the same as the power-on of a whole ensemble, but with only one master running. The power-on cycle until the master is fully integrated (i.e., takes over the master role from the previous one) was measured in an experiment to be 20 s, i.e., ten clock synchronization rounds. In the democratic Master Group approach, the insertion of a new Master Group member does not have an effect on the ensemble clock (and thus on the slave clocks) because the value will be discarded by the averaging algorithm until the new member is properly synchronized. How long this takes essentially depends on the clock controller strategy.

VI. Conclusion

A consistent, accurate time base in a distributed system is mandatory for many applications such as real-time networking, data acquisition and control, or the simple analysis of the precise timing of a sequence of (failure) events. Clock synchronization has the advantage that this shared time base can be used to timestamp data at their origin and thus relieve the actual data processing from stringent real-time requirements. The standardized PTP is a widely used and useful solution particularly for the case of a very accurate clock synchronization in Ethernet networks. Nevertheless, the hierarchical master/slave synchronization has the drawback that a failure of the master leaves the slave clocks unsynchronized for some time. For sure the determination of the master (the Best Master Clock algorithm) could be made more efficient, so that less time is lost and synchronicity can be regained faster, but the principle problem remains.

The solution proposed in this paper, i.e., a democratically synchronized Master Group, improves the fault tolerance of the master/slave approach, while maintaining backward compatibility with the standard. The distributed fault-tolerant averaging of the nodes' local clocks also prevents the network-wide time base from being affected by a failure of a group member. Experiments comparing the conventional single-master approach and the democratic scheme clearly demonstrate that the latter is superior. The increase in fault tolerance naturally comes at the expense of an increased network traffic needed to synchronize the Master Group internally. However, the hierarchical synchronization scheme proposed keeps the overhead moderate. Of course, the goal of maintaining backward compatibility with the IEEE 1588 standard and the resulting need to keep the master/slave scheme on the lowest hierarchy level obviously shifts the fault-tolerance bottleneck to the Master Group speaker. Nevertheless, this functionality is not very complex and can be implemented in a network switch (in Ethernet networks), which is a single point of failure anyway. On this level, fault tolerance can be increased further only with a system-wide approach affecting the physical links and also the slave nodes. This is an issue left for future research and also standardization efforts. The main problem of the (at least) transient deterioration of synchronization accuracy caused by a master failure in PTP can, however, be solved by the proposed method.

References

[1] D. L. Mills, "Internet time synchronization: The network time protocol," *IEEE Trans. Commun.*, vol. 39, no. 10, pp. 1482–1493, Oct. 1991.
[2] J. C. Eidson, "The application of IEEE 1588 to test and measurement systems," *White Paper*, pp. 9–13, Dec. 2005.
[3] D. Owen, "Wired trigger bus physical aspects," *White Paper,* 1.0 ed., 2005.
[4] P. Loschmidt, G. Gaderer, N. Simanic, A. Hussain, and T. Sauter, "White Rabbit – Sensor/actuator protocol for the CERN LHC particle accelerator," in *Proc. 2009 IEEE Sensors Conf.*, Oct. 2009, pp. 781–786.
[5] T. Sauter, "Fieldbus systems – Embedded networks for automation," in *Networked Embedded Systems Handbook*. Boca Raton, FL: CRC Press, 2009, ch. 20, pp. 20–1–20–64.
[6] R. W. Brennan, J. H. Christensen, W. A. Gruver, D. B. Kotak, D. H. Norrie, and E. H. van Leeuwen, "Holonic manufacturing systems: A technical overview," in *The Industrial Information Technology Handbook, Part II (Industrial Information Technology), Section 7 (Integration Technologies)*. Boca Raton, FL: CRC Press, 2005, ch. 106, pp. 106–1–106–15.
[7] H. Hansson, M. Nolin, and T. Nolte, "Real-time systems," in *The Industrial Information Technology Handbook, Part II (Industrial Information Technology), Section 6 (Real-Time Embedded Systems)*. Boca Raton, FL: CRC Press, 2005, ch. 81, pp. 81–1–81–28.
[8] P. Ferrari, A. Flammini, D. Marioli, and A. Taroni, "IEEE 1588-based synchronization system for a displacement sensor network," *IEEE Trans. Instrum. Meas.*, vol. 57, pp. 254–260, Feb. 2008.

[9] "Precision clock synchronization protocol for networked measurement and control systems," *IEC 61588 First Edition 2004–09; IEEE 1588*, pp. 0_1–156, Sep. 2004.

[10] *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, , Jul. 2008, IEEE Std 1588–2008 (Revision of IEEE Std 1588–2002).

[11] L. De Vito, S. Rapuano, and L. Tomaciello, "One-way delay measurement: State of the art," *IEEE Trans. Instrum. Meas.*, vol. 57, pp. 2742–2750, Dec. 2008.

[12] J. C. Eidson, *Measurement, Control, and Communication Using IEEE 1588*. New York: Springer, 2006, pp. 45–50.

[13] G. Gaderer, A. Nagy, P. Loschmidt, and N. Kerö, "A novel, high resolution oscillator model for DES systems," in *Proc. 2008 IEEE Int. Freq. Control Symp.*, 2008, pp. 178–183.

[14] B. R. Calder and A. McLeod, "Ultraprecise absolute time synchronization for distributed acquisition systems," *IEEE J. Ocean. Eng.*, vol. 32, no. 4, pp. 772–785, Oct. 2007.

[15] S. Lee, "An enhanced IEEE 1588 time synchronization algorithm for asymmetric communication link using block burst transmission," *IEEE Commun. Lett.*, vol. 12, pp. 687–689, Sep. 2008.

[16] J. Welch-Lydelius and N. Lynch, "A new fault-tolerant algorithm for clock synchronization," *Inf. Comput.*, vol. 1, pp. 1–36, 1977.

[17] C. Fetzer and F. Christian, "Integrating external and internal clock synchronization," *Real-Time Systems*, vol. 12, no. 2, pp. 123–171, 1997.

[18] K. Schossmaier, "Interval-based clock state and rate synchronization," Ph.D. dissertation, Technische Universität Wien, Wien, Austria, 1998.

Georg Gaderer received the M.S. degree in electrical engineering and informatics both with honors from Vienna University of Technology, Vienna, Austria, in 2002 and 2004, respectively. In 2008 he finished his Ph.D. thesis.

After finishing his studies in 2002, he became a Research Assistant at the Institute of Computer Technology at TU Vienna. During that time his research interests where among various industry projects focused on remote energy meter reading and clock synchronization in powerline networks. In 2005, he joined the Austrian Academy of Sciences where he is currently Head of the Clock Synchronization Group, leading several related national and international research projects. He is part-time lecturer at the Vienna University of Technology and the University of Applied Sciences FH-Campus Wien.

Dr. Gaderer is coordinator of the FP7 ICT STREP $^{\mathrm{flex}}\mathrm{WARE}$, Program Co-Chair of the ISPCS 2007/2009/2010 Conference, active Member of the IEEE P1588 Standardization Group and General Co-Chair of ISPCS 2008.

Patrick Loschmidt was born in Vienna, Austria, in 1977. He received the Dipl.-Ing. degree in electrical engineering from the Vienna University of Technology, Vienna, Austria, in 2002.

From 2001 to 2002, he was a Research Assistant at the Institute of Communication Networks, Vienna University of Technology, working in the area of FPGA design for high-speed optical network nodes. Since 2004, he has been with the Institute for Integrated Sensor Systems, Austrian Academy of Sciences. He is currently leading research projects in the area of network-based high-accuracy clock synchronization. Besides his main interest on hardware design, supporting software drivers and stacks are as well part of his daily work. Further, current activities also focus on his Ph.D. theses dealing with enhanced clock synchronization performance through dedicated Ethernet hardware support.

Thilo Sauter (M'93–SM'09) received the Dipl.-Ing. and Doctorate degrees in electrical engineering from the Vienna University of Technology, Vienna, Austria, in 1992 and 1999, respectively.

From 1992 to 1996, he was a Research Assistant at the Institute of General Electrical Engineering, working in the area of programmable logic and analog ASIC design. Since 1996, he has been with the Institute of Computer Technology, where he was Head of the Center of Excellence for Fieldbus Systems and leading the factory communications group. Since 2004, he is Head of the Institute for Integrated Sensor Systems of the Austrian Academy of Sciences. In 2005, he was appointed Assistant Professor at the Vienna University of Technology. His current research interests are integrated sensor systems and communication networks in automation, with a focus on interconnection issues of fieldbus systems and IP-based networks as well as industrial Ethernet. He is author of more than 170 technical papers and has been involved in the organization of several IEEE conferences.

Prof. Sauter is member of the Austrian Technical Committee ÖVE MR65SC and Delegate in the CENELEC Committee TC65CX, both concerned with fieldbus standardization. Furthermore, he is AdCom Member of the IEEE Industrial Electronics Society, Vice Chair of the IEEE IES TC on Factory Communication, IES Representative in the Administrative Committee of the IEEE Sensors Council, and Treasurer of the IEEE Austria Section.