Hindawi Publishing Corporation
Journal of Computer Networks and Communications
Volume 2012, Article ID 152071, 11 pages
doi:10.1155/2012/152071

Research Article

Highly Accurate Timestamping for Ethernet-Based Clock Synchronization

Patrick Loschmidt, Reinhard Exel, and Georg Gaderer

Institute for Integrated Sensor Systems, Austrian Academy of Sciences, 2700 Wiener Neustadt, Austria

Correspondence should be addressed to Patrick Loschmidt, patrick.loschmidt@oeaw.ac.at

Received 16 October 2011; Accepted 10 December 2011

Academic Editor: Liansheng Tan

It is not only for test and measurement of great importance to synchronize clocks of networked devices to timely coordinate data acquisition. In this context the seek for high accuracy in Ethernet-based clock synchronization has been significantly supported by enhancements to the Network Time Protocol (NTP) and the introduction of the Precision Time Protocol (PTP). The latter was even applied to instrumentation and measurement applications through the introduction of LXI. These protocols are usually implemented in software; however, the synchronization accuracy can only substantially be improved by hardware which supports drawing of precise event timestamps. Especially, the quality of the timestamps for ingress and egress synchronization packets has a major influence on the achievable performance of a distributed measurement or control system. This paper analyzes the influence of jitter sources remaining despite hardware support and proposes enhanced methods for up to now unmatched timestamping accuracy in Ethernet-based synchronization protocols. The methods shown in this paper reach sub-nanosecond accuracy, which is proven in theory and practice.

1. Introduction

In instrumentation and measurement, the General Purpose Interface Bus (GPIB) was for a long time *the* system for data collection and networking of equipment. This bus system has a dedicated wiring for triggering devices and to simultaneously start measurements. The reason for the continuous usage of this relatively old technology is the excellent tool and driver support and the simplicity of the system. Despite these arguments, GPIB has several drawbacks in the handling (connectors, cable) and generality of the approach. First of all GPIB is limited in terms of cable length and number of bus devices. The parallel data transfer and strict arbitration scheme also limit the achievable data rate and make handling and configuration quite complicated for the user. Second, GPIB is also limited in terms of its functionality and does not comply to modern networked systems.

A solution for the test and measurement industry to tackle the drawbacks of GPIB can be found in the LAN extensions for instrument (LXI) [1] approach. This de facto standard uses the well-established Ethernet technology to network measurement devices. The advantage is clearly that one can embed such a system seamlessly into office and lab networks having all advantages of a full network functionality. The application in test and measurement is however only feasible if it can be ensured that the devices are properly triggered. The approach of LXI is to use synchronized clocks for this: a device which detects a trigger condition sends out the time of the trigger condition causing all other devices to a-posteriori save the data at that previous instant of time. This requires that all devices keep some backlog of historic data. It is clear that the precision and the usability of the data highly depend on the accuracy of the clocks in such systems and therefore the synchronization technology. For synchronizing computer clocks, the most prominent approach is the Network Time Protocol (NTP) [2]. This protocol, which is widely used in the Internet, manages accuracies of several milliseconds, with certain extensions even microseconds [3]. As this accuracy is not sufficient for high-precision measurements, the need for a new protocol arose: the Precision Time Protocol (PTP) [4].

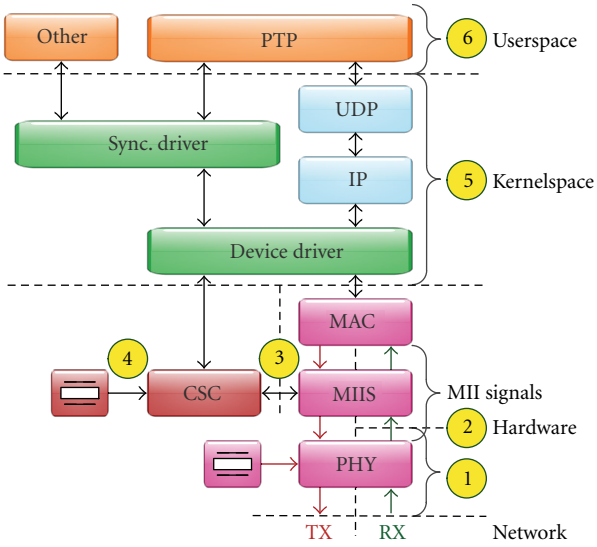

Figure 1: Synchronization node overview and possible internal sources of jitter.

The PTP periodically transmits synchronization messages to update the time on a master-slave basis. For that an algorithm for master election, management, and delay compensation is set on one of the upper layers of the communication stack. In the LXI case, this is the application layer where PTP communicates over User Datagram Protocol (UDP). The standard itself, however, is independent of the communication technology. Although the protocol can be implemented in software, high-precision timestamping has to be done in hardware in order to cancel out protocol stack jitter. This paper outlines the possibilities, analyzes different jitter sources, and proposes new approaches for this highly accurate timestamping. For that, first the state of the art in Ethernet-based clock synchronization is summarized. The motivation points out the reasons for seeking higher precision of timestamps. The following chapters show the influence of the different parameters and required components of a timestamping network interface. Existing accurate methods are given in Section 5 together with their pros and cons in Table 1. Section 6 then presents our new method for highly accurate timestamping, which is then proven to be working in reality by measurements shown in Section 7. The paper is finally rounded up by a conclusion.

2. State of the Art

Figure 1 shows the typical software and hardware structure of an Ethernet-based clock synchronization node. The protocol stack, for example, NTP or PTP, is typically implemented in userspace; see Figure 1(6). Thus, event detection (timestamping) on that level [5] suffers from the jitter induced by all operations required to detect an external asynchronous hardware event (reception of a packet) at or below this layer. The main sources are typically the scheduling behavior of the operating system, data-(length-) dependent processing, or variable execution times, for example, due to caching. Similar reasons are also valid for the kernelspace Figure 1(5) usually hosting the network, for example, Internet Protocol (IP), and transport layer, for example, UDP of the protocol stack [3].

Due to the timely uncertainties in software (except for specially designed real-time systems), almost all high-accuracy synchronization systems rely on event detection close to the physical layer. Since in Ethernet even the datalink layer, that is, Media Access Control (MAC), has variable processing time on the transmit path due to the Carrier Sense Multiple Access (CSMA) mechanism and a possible back-off delay [6], accurate solutions rely on timestamping on the Media Independent Interface (MII). The necessary data scanner, that is, the Media-Independent Interface Scanner (MIIS) block, can be attached to the MII as a separate device [7] or integrated within the functionality of the MAC.

The advantage of the MII is that all receive signals are already in the digital domain but are still source synchronous with respect to the analog data on the line. This gives the synchronization node the possibility to determine timestamps with high precision as the interface is phase-locked to the opposite transmitter. In contrast, interfaces synchronous to the local oscillator, for example, R(G)MII, introduce additional jitter as indicated by (2) of Figure 1 because elastic buffering is required to compensate for clock frequency offsets.

All effects that can deteriorate the performance via the physical layer (see Section 4.3) are summarized by Figure 1(1). These mainly include the analog properties of the physical layer entity (PHY), the cable, and the transmission standard. Timestamping can also be done directly at the physical layer, as shown in [8].

Beside the event detection itself, Figure 1 also outlines the integration of necessary synchronization functions in hardware with the Clock Synchronization Cell (CSC). This element is responsible for timekeeping and timestamping. It is driven by an oscillator which itself is again subject to instabilities indicated by Figure 1(4). In this case, due to the separate oscillator (e.g., for higher stability), an additional clock transition, Figure 1(3), is introduced which again adds jitter. This issue can be solved by using a single oscillator for media transmission as well as timekeeping.

To build a complete Ethernet networks, one element is essential: the switch. In principle Ethernet switches use the same PHY and MAC as ordinary nodes. Concerning clock synchronization, the residential time of a packet on a switch has to be measured to compensate for varying switching decision or queuing times. However, it can be said that the principles and influences are similar to a node, and therefore the results of this paper can be applied accordingly.

3. Motivation

As outlined in the introduction, synchronization is, among other areas, mandatory for test and measurement applications. Precise timestamping is required, because accurate synchronization over packet-oriented depends on a common event that can be detected by the synchronizing nodes. Such events are special messages, regularly sent out by PTP

Table 1: Comparison of the different approaches.

Approach	Type	Advantage	Disadvantage	Complexity	Impl. effort	Accuracy
HSC	Single shot	Linearity	High-speed clocks	Low	Low	Medium
PSC	Single shot	No high-speed clocks	Linearity issues	Low	Low	Medium
TDL	Single shot	Single-event capturing	Temperature dependency	High	High	High–very high
TSA	Single shot avg.	Simplicity	Leakage effects	Low	Low	Low–medium
DPE	Phase estimator	High freq. range	Analog parts req.	High	Medium	High
PFE	**Phase estimator**	**Purely digital, size**	**Narrow freq. range**	**Medium**	**Low**	**Medium–very high**
PFE + TSA	**Phase estimator avg.**	**At least PFE perf.**	**Correlation dependency**	**Medium**	**Low**	**Very high**

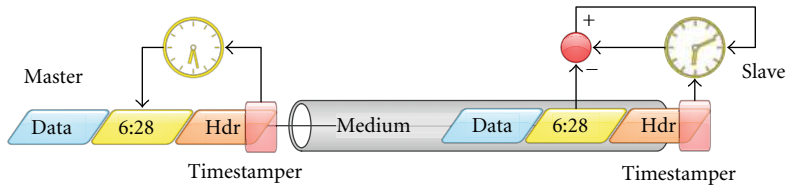

Figure 2: Basic method for clock synchronization in packet-based networks and the necessity of event detection (timestamping).

together with the absolute time of the event. Moreover, PTP uses a master-slave approach, where one clock master synchronizes several slaves. The basic principle of timestamping is indicated in Figure 2. For packet-oriented networks, the synchronization protocols define the occurrence of a packet (mostly, the start of the data frame) on the medium as the common event. As shown in the figure, the master node catches the transmission time and copies it into the synchronization packet. On the other end, the slave detects the reception and compares the event time with the information contained in the packet and adjusts its clock. The remaining offset between the clocks, which is due to the transmission delay on the network, can be compensated by round-trip delay measurements. The latter use the same timestamping techniques by sending a packet from slave to master or vice versa. Assuming that the delay on one link is symmetrical; that is, it takes the same time to transmit a message from node A to B as from node B to A, the line delay can be calculated by using the time difference between send and receive event, reducing it by the residential time at the remote side, and taking the half of the result as the delay of the link.

The alternative to precise timestamping—long-term averaging to enhance the precision—cannot be applied due to nonstationary jitter of several elements within the synchronization loop, in particular the oscillator. Thus, accurate timestamping is the key for any high-accuracy clock synchronization method. This paper focuses on the influence factors affecting the system precision despite the usage of hardware timestamping in Ethernet to develop methods able to acquire precise timestamps. Further, tradeoffs are identified that allow to tune different parameters to achieve a predefined accuracy boundary efficiently.

4. Sources for Inaccuracy

Since timestamping at a certain network layer avoids timely influences on the accuracy of all layers above, hardware timestamping cancels out software dependencies. Still, also for hardware implementations, certain limitations due to remaining jitter sources exist that influence the achievable accuracy [9]. As the first two of the following aspects are influenced by a wide range of parameters, the impact is summarized in this section while more detail is presented in the appendix.

4.1. Oscillator. An ideal oscillator serving as a timebase for a node would require only a single synchronization at startup to compensate for the initial offset. Due to the fact that every oscillator is subject to a number of physical phenomena, the progress of time is not constant; even worse, the accuracy is dependent on the considered hold-over time, which is the interval between two synchronization events. Periodic resynchronization is therefore indispensable. Several short-time noise phenomena, for example, phase noise, additionally complicate precise timestamping.

4.2. Synchronization Interval. Considering only the stability characteristics of a selected oscillator, an optimal synchronization interval can be chosen. Usually the longest interval with the lowest absolute clock jitter is chosen to minimize necessary network traffic between the nodes. However, as inaccuracies in timestamping cannot be distinguished from oscillator jitter, both have to be as low as possible. While the timestamp inaccuracy is independent on the synchronization interval, the timebase error caused by the oscillator instability increases with time. Hence, depending

on the interval either the oscillator or the timestamping can be identified as the limiting factor.

4.3. Physical Layer Properties. Since it is not (cost) efficient to replace commercial off-the-shelf (COTS) PHYs with a proprietary solution supporting timestamping, the most reasonable way to add high-precision timestamping to a system is to use the interface of the PHY to the MAC. Thus, the delay behavior of the PHY still has influence on the achievable performance of the timestamping method.

The most important properties are the line coding, the translation to MII, and the internal phase-locked loop (PLL). Since Ethernet is designed as an asynchronous network, the receive side of the physical layer has to recover the transmission clock in order to correctly decode the data. The other direction, the data transmission can be performed with the locally available clock. This does not introduce a clock transition and therefore does not increase the jitter (The reason why clock transitions add jitter is given in Section 4.4).

Beside dynamic link delay changes, also the absolute delay of the PHY-to-PHY system can vary each time the link is newly established. This is due to the fact that, for example, in Fast Ethernet, the 125 MBaud on the line have to be translated to four bit symbols on the 25 MHz MII, which allows five different locking positions [10].

The delay behavior for the three most popular copper-based Ethernet transmission standards is illustrated in Figure 3 using two Marvell 88E1111 PHYs over a 3 m direct connection. Since 10 Base-T keeps the line idle when there is no data to be transmitted, the PLL of the receiver has to resynchronize to the transmission clock with each packet. If the packet rate is low, the behavior is similar to a link reestablishment since the PLL can take any of the possible locking positions, that is, two different, 100 ns apart for Figure 3. On high packet rates, the drift of the receive PLL between two packets is small and thus the PHY can stay synchronized, which then results only in the jitter of the clock recovering process.

Compared to the original Ethernet, Fast Ethernet introduced 4B/5B line encoding and the *Idle* code-group (clause 24.2.2.1.2 of [6]). The coding replaces four bit by five bit groups, which are coded in a way that long constant bit sequences are avoided to ease clock recovery. Additionally, it is possible to insert control codes, for example, to denote the start of a transmission (/J/, /K/). Hence, the synchronization can be maintained continuously, independent of the data transfer, which results in significantly lower standard deviation of the transmission delay.

In contrast to the enhancement from the original 10 Base-T to 100 Base-T, Gigabit Ethernet does not give a performance boost for clock synchronization. Resulting from the fact that the physical layer uses a single clock for both directions, there is a master and a slave PHY.

Thus, there is a clock transition on the receiver side to the local clock. Due to the 125 MHz clock rate, this gives an equally distributed communication delay over a window of eight nanoseconds on the slave side, while the master only

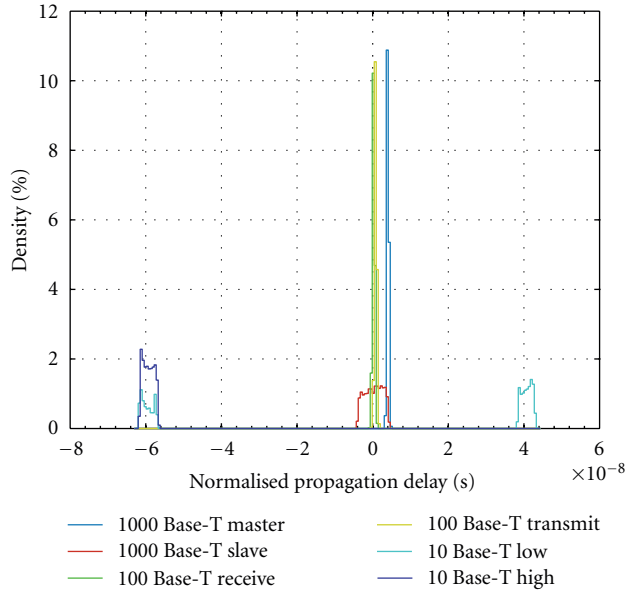

Figure 3: Comparison of various Ethernet physical layer transmission standards.

shows PLL and oscillator phase noise. Also the assumption that the problem of asymmetric delays of Ethernet can be solved with Gigabit Ethernet due to the bidirectional usage of all copper-lines does not turn out to be true, as shown in this figure.

Summarizing the findings, it can be said that apart from the initial locking with a specific absolute delay, the communication jitter for Fast Ethernet is by far the lowest (with a standard deviation $\sigma = 0.286$ ns compared to 1.387 ns for 10 Base-T and theoretic 2.309 ns for 1000 Base-T).

4.4. Timestamp Resolution. The resolution of the timestamp basically influences the quality of the timebase comparison for the control loop of the synchronization system. While in general it is no problem to gain enough resolution in hardware, the difficulty arises from the fact that the timestamp has to be transferred through various network layers to the application [11]. Linux, for example, started supporting timestamps with nanosecond resolution from version 2.6.22 on. The necessary structure to transfer hardware timestamps to the applications were added in version 2.6.30. Currently, the resolution is limited to one nanosecond, which makes it difficult to safely achieve synchronization accuracies below the nanosecond.

Figure 4 illustrates the main problem for highly accurate timestamping. Since Ethernet is an asynchronous network, the ingress packets are asynchronous to the local clock of the timebase. Thus, the issue boils down to detecting the occurrence of a packet, that is, frame active signal, with high precision. In synchronous digital designs, the asynchronous activation of a single event between two clock edges can be detected at the earliest with the next edge. Highly accurate solutions therefore have to measure/estimate

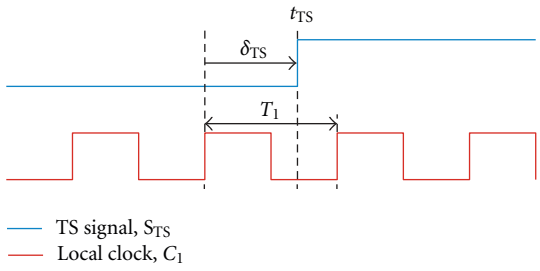

Figure 4: Timestamping error due to phase/frequency shift.

the exact occurrence of the event with respect to the local clock [12].

5. Existing Single-Shot Methods

Single-shot methods are one solution to the problem of accurate timestamping and measure δ_{TS} directly. That is done by determining how long after the rising edge of the local clock the timestamp signal $\mathcal{S}_{TS}$ has been asserted. For that purpose, a clock cycle T_l is divided into $n \in \mathbb{N}^+$ equally spaced fractions, which reduce the timestamping variance (the variance of a uniform distribution) to $\sigma^2_{TS} = T_l^2/(12n^2)$. The main advantage is that the timestamp signal is not required to occur in regular intervals. Events can be detected without any further reference even on their first occurrence.

5.1. High-Speed Counter (HSC). This approach divides T_l by a short-width high-frequency counter with a period $T_h = T_l/n$. The counter is reset at every rising edge of the local clock, and its value is frozen when $\mathcal{S}_{TS}$ is active. The sampled counter value divided by n then describes the relative phase offset δ_{TS}. Since the period of the local clock is exactly a multiple of T_h, the clock transition between these two clocks can be designed without any additional jitter, and consequently the timestamping variance reduces by n^2. The result is similar to a design completely clocked with $1/T_h$ with the advantage that only a few logic elements have to run at a high frequency.

With very low additional effort, n can be doubled using registers sampling with the opposite edge of the clock. Such register pairs using both edges, called Double Data Rate (DDR) registers, are available in many devices to be used for communication links. The additional improvement by a factor of two is achieved without change in the clock rate and is a special case of the next option requiring only an inverter.

5.2. Phase-Shifted Clocks (PSCs). The use of phase-shifted clocks is another method to partition T_l. For this technique, $n-1$ additional clocks are generated, which are phase shifted by $2\pi/k$ with $k = 0, 1, \ldots, n-1$ with respect to C_l. The timestamp signal is registered into n registers, each with a different generated clock. If the $\mathcal{S}_{TS}$ gets active, the first i registers still sample the old state. This generates a so-called thermometer code, which is converted to a binary code and used in the same manner as the HSC. Since $\mathcal{S}_{TS}$ drives n registers, special care has to be taken that the clock at the registers has the designed phase shift (i.e., the registers are timely equally spaced), since otherwise the thermometer code becomes nonlinear. This effect and the number of output clocks per PLL limit n to a value of about 10 in state-of-the-art Field Programmable Gate Array (FPGA) devices (e.g., Altera Stratix III family [13]).

5.3. Tapped Delay Lines (TDLs). TDLs are a common approach for digitizing times with sub-nanosecond accuracy. The basic configuration of a TDL consists of a serial chain of n latches having a delay τ_L, a second chain of noninverting buffers with delay $\tau_B < \tau_L$, and an output logic as described in [14]. The signal to be timestamped is then fed through all latches, which freeze its current state at the rising edge of a second signal (clock). The resulting thermometer code can then be evaluated after the next clock edge.

Nevertheless, it has to be considered that such a design uses asynchronous logic, and therefore the delays τ_L and τ_B are not only placement but also temperature dependent. The linearity of a TDL may be compromised by these effects, and special calibration logic may be required. The possible precision depends on the intrinsic switching speed of the latches and is typically in the range of 100 ps [15].

5.4. Proposed Phase-Estimating Solution. Phase-estimating methods do not measure δ_{TS} directly, but rather estimate it using the fact that $\mathcal{S}_{TS}$ is asserted synchronously to the communication clock. The relatively new approach for highly accurate time interval measurements is described in [16] for a active clock skew compensation in VLSI designs using analog components. The authors of [17] present a similar method using a 10 MHz atomic clock source sampled by an analog to digital converter (ADC) driven by the communication clock. The resulting waveform is used to perform a phase estimation by a 1024 pt Fast Fourier Transform (FFT). While the results show a very low timestamping standard deviation of about 10 ps, this approach requires one ADC per timestamper and an atomic clock to achieve the mentioned performance.

5.5. Approach. As with analog single-shot methods, for the targeted application area, timestamping in Ethernet, additional components (especially analog ones) are rather impractical. Consequently, a technique to implement the scheme using pure digital function blocks is presented by us in [12].

In this approach, the frequencies of the involved clocks have to fulfill several requirements in order to benefit from the method of phase estimation. First of all, the rising edge of the local clock T_l should occur equally distributed within the clock cycles of T_c averaged over a given time span (i.e., the rising edge of the local clock should cover all phases of the communication clock with equal probability). This includes that T_c must not match T_l or multiples thereof because in this case the rising edge would always coincide with a certain phase of T_c and the necessary averaging time span would be infinite. It is known from estimation and detection

theory that applying a randomization function may help [18]. However, such a randomization function (e.g., a clock with very high clock jitter) is an impractical solution as the clock jitter reduces the maximum attainable clock frequency in the same way.

In order to be able to sample T_c (or signals synchronous to it) at a number of different phase states, the local clock has to be a nonmultiple of the communication clock. To represent this criteria, the nominal local clock period T_l is given by

$$T_\mathrm{l} = T_c \frac{1-\beta}{n}, \quad 0 < \beta \ll 1, \tag{1}$$

with the design parameter β as the relative frequency offset factor and n as the nominal oversampling factor. A cycle slip (i.e., when the rising edge of two clocks pass each other) occurs, if the difference in the periods sums up to T_l as given by

$$\begin{aligned} T_\mathrm{l} &= \frac{1}{\varepsilon}\left[\frac{T_c}{n} - T_\mathrm{l}\right] \\ &= \frac{1}{\varepsilon}\left[\frac{T_\mathrm{l}}{1-\beta} - T_\mathrm{l}\right], \end{aligned} \tag{2}$$

$$\varepsilon = \frac{\beta}{1-\beta}. \tag{3}$$

Using (3) cycle slips will happen after every $1/\varepsilon$ local clock periods. This implies that $\mathcal{C}_c$ has been sampled at $\lceil 1/\varepsilon \rceil + 1$ different phase points over one cycle slip period and the maximal attainable precision is nominal βT_l. Selecting a very small β results in a small frequency offset and great resolution, but this can cause $T_c \leq nT_\mathrm{l}$ due to instabilities of the two involved oscillators during the long theoretical cycle slip period. Consequently, cycle slips might not happen at all, or even worse, reverse cycle slips can occur, resulting in possible data loss.

Furthermore, small frequency differences are unusable since the rising edge instance is only equally distributed within T_c over a long averaging window, and for short averaging windows the leakage effect [19] becomes dominant.

5.6. Timestamp Averaging (TSA). Timestamping a frame m times by means of the communication clock is one option to estimate the phase at the timestamping event based on the assumption that δ_TS averages to 0.5. Given that the timestamps are centered before and after the assertion of $\mathcal{S}_\mathrm{TS}$ (i.e., $(m-1)/2$ timestamps before and after the event) and that the clock rate is not changed during the timestamping period, the final timestamp TS is calculated by a weighted average over the timestamps TS_i following

$$\mathrm{TS} = \sum_{i=(1-m)/2}^{(m-1)/2} \alpha_i \mathrm{TS}_i, \quad \sum_i \alpha_i = 1. \tag{4}$$

This Finite Impulse Response (FIR) filter can be simplified to a window integrator with $\alpha_i = 1/m$ with some limitations, namely, leakage effects. The timestamping window should cover one cycle slip period or multiples thereof to get the timestamps equally distributed over one T_c period. Since the cycle slip period is dependent on the current frequency offset, it varies with the oscillator drift between the communication and local clock. One solution to this problem is to adjust m to cover always a multiple of the cycle slip periods or by capturing a big number of such periods and using a windowing function to minimize leakage effects. The leakage can also be reduced by selecting a rather large ε with the drawback of reduced resolution.

In the optimal case the resulting timestamping variance reduces to $\sigma^2_\mathrm{TS} = T^2_\mathrm{l}/(12m)$. For example, a typical IEEE 1588 frame with about 80 bytes frame length may create $m = 160$ timestamps on the 4-bit-wide MII. Given that T_l equals 10 ns and all timestamps can be considered uncorrelated, the standard deviation becomes 228 ps.

5.7. Digital Phase Estimation (DPE). The phase of the communication clock can be also directly estimated by phase detectors. Such a detector can be, for instance, based on a mixer, which shifts the spectrum of the clock to a low frequency similar to common superheterodyne receivers. The output of the mixer is low-pass filtered to remove aliases at multiples of the input frequency and is conducted into a phase estimator. Given that the duty cycle of the clock input signal is constant, the low frequency part of the downmixed signal then is a measure for the phase difference at the inputs. Nevertheless, real filters with a low bandwidth introduce significant group delay, which has to be taken into account for the calculation of the timestamp. In order to allow for a digital implementation, the mixer can be replaced by an XOR gate and the output can be filtered in the same way. A further solution would be to use an external analog antialiasing filter in combination with an ADC and only perform the second filtering digitally, similar to Zhu's method [17].

A pure digital implementation without requiring external parts can be achieved, but in such a processing scheme undersampling occurs. As the XORed signal is not bandwidth-limited, sampling results in alias frequencies that can dominate the signal (e.g., if the "1 bit" sampler is sourced from a clock correlated with $\mathcal{C}_\mathrm{l}$). One feasible solution is to sample the mixed signal by a clock which is uncorrelated to both inputs and apply the sampled signal to a low-pass filter with very low relative bandwidth. Such filters are typically Infinite Impulse Response (IIR) type since comparable FIR filters would need a big number of filter taps. IIR filters on the other hand have a frequency-dependent group delay, which means that the frequency offset between $\mathcal{C}_\mathrm{l}$ and $\mathcal{C}_c$ must be estimated in order to compensate for the filter's group delay.

5.8. Combined Phase/Frequency Estimation (PFE). Rather than estimating δ_TS directly, it is also possible to estimate the phase by its derivative, the frequency offset, together with a reference point. In the following, the hat notation (e.g., $\hat{x}$ for an estimation of x) is used to differentiate between the true value and its estimation.

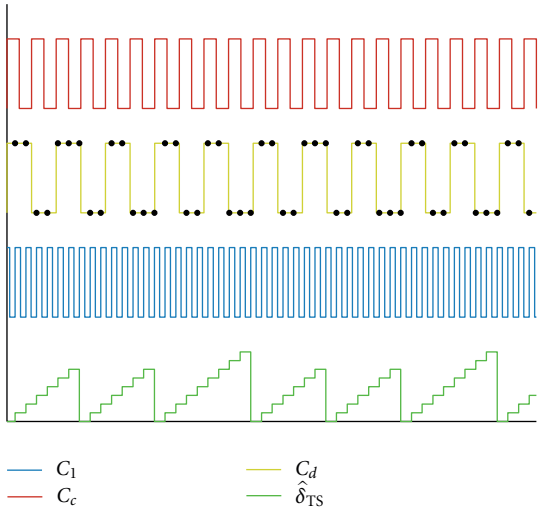

Figure 5: Phase estimation between cycle slips.

The principle of our phase estimation by frequency estimation approach can be implemented as follows. Whenever the communication clock is phase aligned to the local clock (i.e., when a cycle slip occurs), the phase estimation $\hat{\delta}_{TS}$ is set to zero. In every subsequent clock cycle $\hat{\delta}_{TS}$ is incremented by the estimated inverse cycle slip period $\hat{\epsilon}$. In other words, $\hat{\delta}_{TS}$ is the sampled integral of $\hat{\epsilon}$ over one cycle slip period. Given that the frequency is stable in the averaging interval, $\hat{\delta}_{TS}$ ideally would reach 1.0 at the next cycle slip as depicted in Figure 5 for $n = 2$. For the reason of better visibility, a relative high value of 0.13 was chosen for β resulting in a (rather short) cycle slip period of 6.7 local clock cycles.

The accurate detection of the cycle slip instant is critical for the start of the integration of the frequency offset to get $\hat{\delta}_{TS}$. To make the method independent of the communication clock duty cycle, a derived clock $\mathcal{C}_d$ with the period $2T_c$ is generated digitally. $\mathcal{C}_d$ is fed into a shift register with $5 + n$ taps clocked with the local clock. Further, this clock is used for cycle slip checking and performing edge detection. While the first two shift taps are used for buffering, the middle taps ($2 + n$ down to 2) are used for cycle slip detection. If all $n + 1$ middle taps contain the same binary value, a cycle slip must have happened. The last two taps (1 down to 0) are used to detect rising and falling edges of $\mathcal{C}_d$ at which the buffered timestamp signal $\mathcal{S}_{TS}$ is checked for a high level.

The dots on the derived communication clock, $\mathcal{C}_d$, mark the sampling points of this signal by the local clock, $\mathcal{C}_1$. Each time the signal is sampled three times with the same value, a cycle slip must have occurred and the phase is reset.

Due to the fact that the cycle slip period is in general not an integer number, the phase estimation cumulates an error with each reset. Thus, higher orders of the cycle slip period exist. In picture this can be seen; that, every third time, the cycle slip is detected one period (of $\mathcal{C}_c$) later in order to compensate for early detection at the previous two reset events. The higher order can be calculated by taking the remainder, which is $0.7 \cdot T_1$ and summing it up to one period of $\mathcal{C}_c$: $3 \times 0.7 \times T_1 \approx T_c$. Obviously, there is again a remainder, $0.1 \cdot T_1$, which again causes a higher order periodicity. Note that these higher orders can only be used to refine the timestamp if the remainder of the cycle slip period does not wrap over. Even for selected frequencies, this is only applicable for a very narrow frequency range. Considering the oscillator tolerance for Ethernet (50 ppm), the method is not feasible. However, even if the higher-order periodicities are neglected, TSA can be used to improve the accuracy. The relative frequency offset $\hat{\beta}$ can be calculated by monitoring the number of rising edges of $\mathcal{C}_c$ with respect to $\mathcal{C}_1$. In average, every $1/\hat{\epsilon}$ local clock cycles, a cycle slip will occur, which results in a missing rising edge with respect to the local clock. To achieve a continuous value of $\hat{\epsilon}$, a low-pass filter is used. The bandwidth of the filter must be narrow enough to track frequency changes of the oscillator while removing the cycle slip frequency. In general, IIR filters which have poles close to one are ideally suited for this application. Alternatively, the frequency offset $\hat{\epsilon}$ can be calculated by the cycle slip rate $1/\hat{\epsilon}$, but this requires a division block, which consumes a significant amount of logic resources. In any case, the calculation of the final timestamp involves summing up the last value of the rate-controlled timebase plus $\hat{\delta}_{TS}$ times the clock rate.

5.9. Summary. The selection of the cycle slip rate not only has to consider the accuracy requirements of the application but also the behavior of the physical layer. As mentioned, for example, 10 Base-T does not provide a continuous clock supply that can be measured by the phase/frequency estimation. Therefore, the factor β has to be chosen in a way that the measurement can be done within the reception/transmission time of a single packet, which is rather stringent. Alternatively, the precision can be enhanced by keeping the carrier active via the transmission of several (arbitrary) packets to allow the phase/frequency estimation to settle. The last packet of such a burst can then be timestamped with relatively high precision. Obviously, the provision of a continuous link implies much higher potential for accurate timestamping due to the permanent tracking of phase and frequency.

6. Implementation and Evaluation

One major advantage of the presented phase estimating method over single-shot techniques is the fact that the measured signal only passes one digital processing path. Unless for the latter techniques, only one sampling register is directly connected to the $\mathcal{S}_{TS}$ or the respective synchronous clock signal. Therefore, linearity problems due to unequal signal propagation delays to registers or between other logic elements, for example, buffers in TDLs, cannot occur. This makes the mentioned method also robust against temperature and other effects, which have direct influence on the signal propagation properties within an integrated circuit. Nevertheless, there might be a small (asymmetric) delay between the receive and the transmit path since two

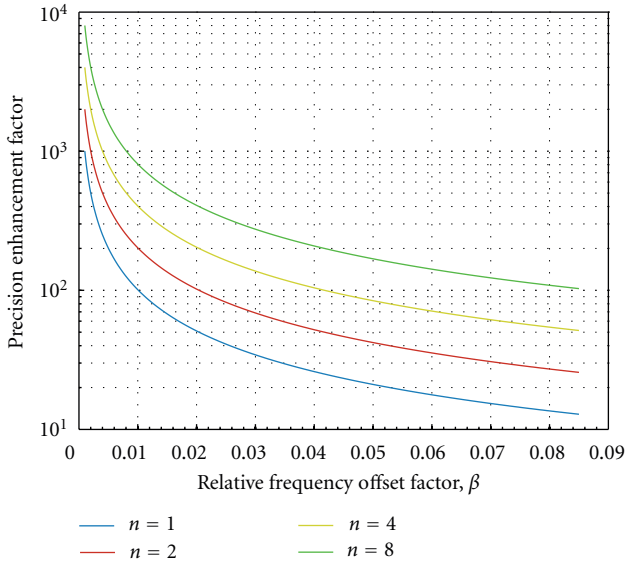

Figure 6: Dependency of the theoretical and the effective precision enhancement on the relative frequency offset.

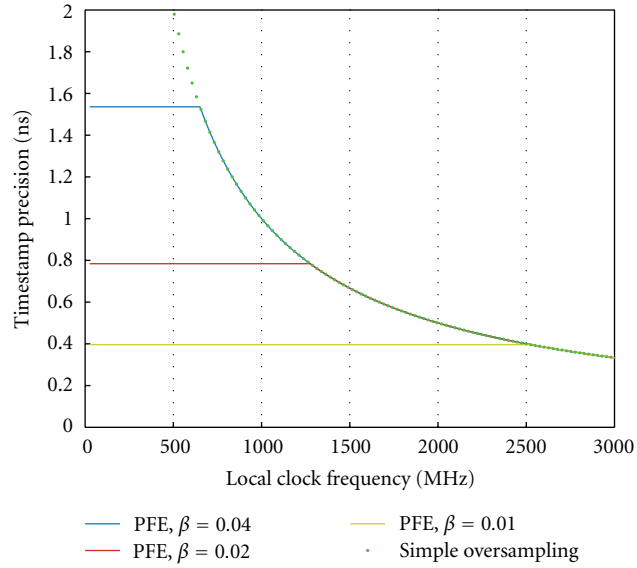

Figure 7: Timestamp precision with a 25 MHz communication clock for the PFE approach and simple oversampling.

separate instances of the mentioned method are required. For each chip, this delay has to be calibrated once to compensate for internal placement and routing issues.

6.1. Precision. Mathematically, the maximal attainable precision is given by $\beta T_l = T_c(\beta - \beta^2)/n$. Using (1) the enhancement over the period of the communication clock results in a factor of $n/(\beta - \beta^2)$. Figure 6 shows the shape of this function and the possible improvement due to oversampling. Unfortunately, the results for $n > 1$ are only theoretical values.

As the phase estimation is reset at every cycle slip, cycle slips and the interpolated phase value are correlated. Oversampling increases the number of interpolation steps for δ, yet, the number of reachable values within one cycle slip period stays the same. It can be shown that with an oversampling ratio n only every nth value can be reached by the rising edge of $\mathcal{C}_c$. Hence, the effective precision is only $T_c(\beta - \beta^2)$, independent of n, and only scales with the duration of the cycle slip period.

Still, if n is set anything below $1/\beta$, the combined phase-frequency estimation method offers improved precision. Figure 7 depicts an example for a 25 MHz communication clock with three frequency offset factors β. For simple oversampling with a high-speed clock, the timestamp precision is exactly the period of the clock. If PFE is used, the precision improves to βT_l until the frequency (or the oversampling factor n resp.) is highly increased. For $1 < n/\beta$, the PFE method degrades to the standard oversampling method. Frequencies in the range of multiple GHz cannot be reached with simple circuits, yet, the PFE method offers the same precision with design frequencies in the range of 50 MHz only.

6.2. Physical Limits. One critical parameter for the PFE method or in general for any form of highly precise timestamping is the phase noise of the clock. If the phase noise is low with respect to the calculated precision, then the considerations taken up to this point are directly applicable. However, measurements on PHY revealed that the phase noise is in the range of 100–250 ps RMS, 124 ps for the setup in [9]. While the phase noise does not alter the frequency estimation due to its long-term averaging, it causes the phase estimation circuit to see cycle slips too early or too late causing a timestamp jitter similar to the clock jitter. The clock can be cleaned by means of clock conditioners or the PLL inside the FPGA. The latter is readily available in FPGA-based solutions, but itself has a relatively high self-noise (around 200 ps RMS jitter) [13].

As discussed in Section 6, a randomization function is beneficial to remove clock correlation and leakage. Considering that the local clock frequency $1/T_l$ is typically around 50–300 MHz, the randomization by the clock jitter is far too low as it should cover one clock cycle uniformly distributed. For PFE with a high value of β (and therefore low precision), the randomization caused by the clock jitter can reduce the correlation between the timestamps. Hence, if multiple timestamps are averaged, the jitter may actually improve the precision of a packet's timestamp.

As already outlined, timestamping a frame multiple times can increase the precision, if the timestamps are not totally correlated. Since every Ethernet packet is at least 64 bytes long [6], a timestamper attached to the MII can draw 128 timestamps starting at beginning of the frame. These timestamps can then be fed into a minimum mean square error (MMSE) estimator calculating one timestamp for the frame using the least squares method. This combination of two methods PFE and TSA can guarantee at least the

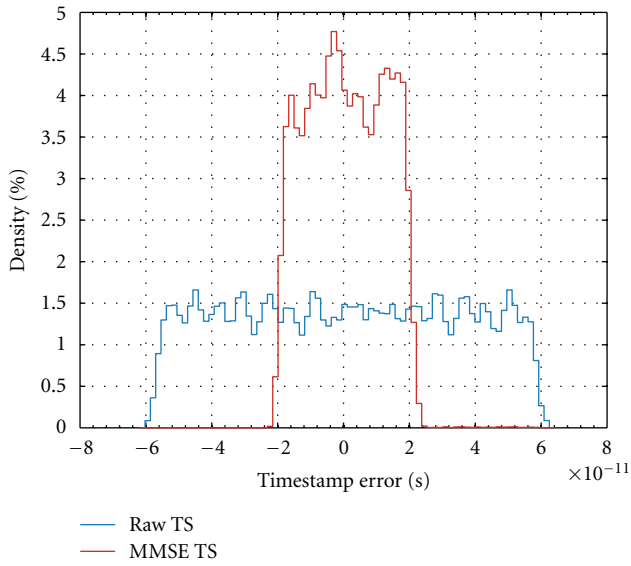

Figure 8: Timestamp error for a combined phase/frequency estimation ($\beta = 0:002$; $n = 2$) with linear MMSE estimator using 128 samples. The histogram shows a total of 50 kTS in 75 bins, giving a standard deviation of 12.036 ps.

performance of PFE only and offers improved precision for timestamps with lower correlation.

Measurement results of an FPGA-based implementation showing the feasibility of the described method are given in Figure 8. Both, communication and local clock, are sourced from a frequency generator with a rubidium clock as reference. The communication clock was set to 25 MHz and the local clock to 50.1 MHz resulting in $\beta = 0.002$ and $n = 2$. Every millisecond the generation of a timestamp set consisting of 128 timestamps within the communication clock domain is triggered. The captured timestamps in the local clock domain are then compared with the ideal ones, and the error is shown in the figure. The calculated precision for this setup should have a uniform distribution with $T_c(\beta - \beta^2) = 8$ ps width. The measured graph shows that it is actually about 120 ps wide. With MMSE averaging the timestamp precision improves to a width 40 ps or 12.036 ps standard deviation, respectively. The results can be interpreted in the following way. For precisions in the low picosecond range physical effects (e.g., clock jitter, thermal noise) dominate the timestamping. For larger values of β, for instance, $\beta = 0.0045$, as shown in [12], the calculated precision can be achieved even with the FPGA's PLLs.

7. Conclusion

A comparison of the methods analyzed within this paper is given in Table 1. Single shot is the method of choice if the event is not aligned to a clock. For events that appear synchronous to a clock, like the case for Ethernet with MII, phase estimation methods offer a similar or even superior performance with respect to, for example, a TDL, while requiring only low complexity and slow logic. The advantage of the purely digital design of the PFE overwhelms the restriction to a low-frequency range in particular for the intended timestamping application.

In conclusion it can be said that highly accurate clock synchronization is of utmost importance for test and measurement applications where a perse non-real-time network is used to sample/collect data as in the LXI approach. The latter uses PTP to achieve that functionality by distributing absolute time information to attached network nodes. The used messages have to be supplied with the actual ingress or egress time at the synchronizing nodes. Providing a precise value for that event was shown to be again an issue of timestamping.

It is further shown that there are several different possibilities to timestamp, and various jitter sources exist within a Ethernet IP/UDP node. It can be stated as a rule of thumb that, the closer one gets to the physical layer, the lower the jitter and therefore the influence on the precision of a timestamp.

As a further conclusion of this paper, it can be said that timestamping is a crucial issue for highly accurate clock synchronization: several methods like HSCs, PSC, TDLs, or the phase-estimating methods as DPE are limited in terms of their reachable accuracy. It was shown that this limitation is however a bound, which is much lower than previously published results in terms of accuracy. Finally, timestamping using a MMSE estimator together with PFE can bring the accuracy down to an equal distribution with a standard deviation of only 12 ps.

Appendix

A. Timebase Issues

A.1. Oscillator. The physical properties of oscillators are an essential parameter for the accuracy of a clock synchronization node. In general, the stability of oscillators is dependent on the sampling period and thus defines the criteria to select the right synchronization interval, the hold-over time, respectively.

The Allan variance [20], which is defined by the expectation value, $\langle \cdot \rangle$, of the normalized frequency y over a sampling period τ, that is,

$$\sigma_y^2(\tau) = \frac{1}{2}\left\langle (y_{n+1} - y_n)^2 \right\rangle, \tag{A.1}$$

is used to characterize oscillators. A low Allan variance means good stability over a certain measurement period τ. As the fractional frequency error can be also calculated by $y_n = 1/\tau(x_{n+1} - x_n)$, with x_n as the time error, at sample number n, an estimator of the Allan deviation can also be derived in the time domain by offset measurements from an ideal clock. A typical graph in double logarithmic scale for the Allan variance is shown in Figure 9. For short periods it will decrease by 100, if the sample time is increased by a factor 10. For large periods, long-term physical effects (e.g., temperature changes, aging) increase the value. One interesting area of sampling time for the case of clock synchronization is the minimum around 1 s. As

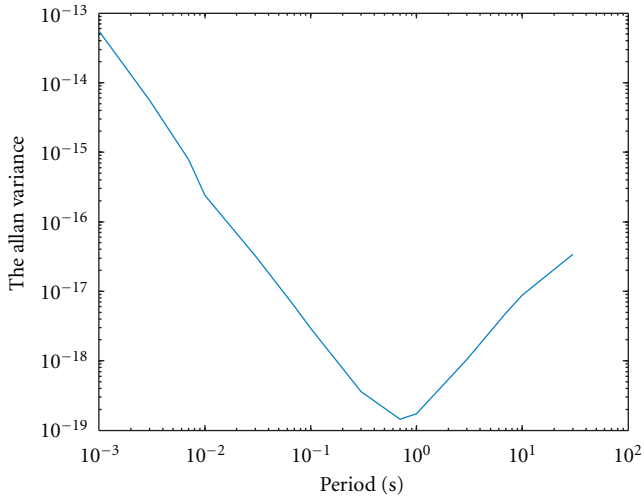

Figure 9: Typical Allan variance of an uncompensated crystal oscillator.

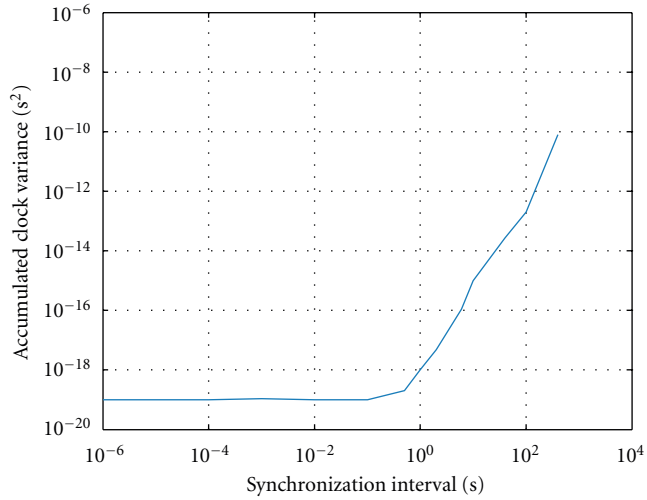

Figure 10: Absolute variance representing the accuracy with respect to the synchronization interval.

shown in the figure, the lowest values of the Allan variance can be observed. This so-called Allan-Floor results from a minimum in the sum of several different contributing noise contributions (white noise, flicker phase noise, white frequency noise, flicker frequency noise, and random walk frequency noise) of the oscillator. The individual power spectral density shapes add up in a way that, at a specific observation interval, a minimum as in Figure 9 can be observed.

Since clock synchronization seeks for a stable progress of time, the sample interval with the lowest deviation is of interest. The next section gives insight on how the synchronization interval should be chosen if a certain oscillator, that is, Allan deviation curve, is used.

A.2. Synchronization Interval. The most interesting and easiest changeable parameter in a synchronization system is the equidistant interval T_{sI} for exchanging synchronization messages. For further investigation of the influence, it is assumed that both, the master and the slave, are equipped with an oscillator that has a drift $D(t)$. The presence of varying drift of the oscillators will increase the offset between the nodes with the progression of time. If it is assumed that two oscillators were perfectly synchronous at the begining of a synchronization interval, the offset α after T_{sI} is [9]

$$\alpha = \int_{T_{\mathrm{sI}}} (D_M(t) + D_S(t))\mathrm{d}t. \tag{A.2}$$

If the Allan variance of the oscillator is known, the standard variance of α can be estimated. The two-sample Allan variance can be interpreted as a variance of the frequency change rate. Therefore, the frequency variance is the integral of the Allan variance over T_{sI} plus a frequency offset, the standard variance σ_D^2 for the drift. Another integration results in the time variance plus a time offset. If we assume that the frequency and time offsets are compensated in the synchronization system's servo, the variance of the time error is just the denormalised Allan variance (the normalized Allan variance multiplied by T_{sI}^2). Since the master and the slave oscillators are statistically independent, the resulting variance of α is the sum of the standard variances of both nodes.

For a typical oscillator, the accumulated clock variance remains constant for short intervals (as depicted in Figure 10). Thus, just from the oscillator's point of view, one gains in the first step no additional accuracy if the system synchronizes more often. For long synchronization intervals, the standard deviation rises significantly, which is mainly due to long-term effects like temperature changes. Consequently, the synchronization period should be selected as the maximum value that is available before the flicker floor of the Allan deviation is reached. This is equal to the end of the zero slope part of the accumulated clock variance at about 0.2 s. From the oscillator's point of view, there is no difference on which point of the zero slope part the synchronization period is chosen, which means that there is no direct further accuracy gain by exchanging more timing messages than necessary. Nevertheless it has to be mentioned that in principle an improvement of the resolution by averaging samples can be done where an increased synchronization rate has a positive influence due to that fact that the number of samples is increased.

PTP version 1 [21] has a default synchronization interval of 2 s and allows synchronization intervals between one and 16 s, incrementing by a factor of two. As pointed out by Figure 10, even the shortest possible period, according to the IEEE 1588 standard, of one second is rendered nonoptimal for the tested oscillator resulting in a degradation of accuracy.

A closer look at the basic data for the diagrams reveals that a synchronization interval of 0.5 s or even better 0.2 s can significantly reduce the jitter introduced by the clock source sampling. If PTP version 2 [4] is used in order to allow for the mentioned values of T_{sI}, the control loop is supplied with a higher quality of timestamps in slightly shorter intervals,

while the network load is still within a reasonable range of about five packets per second.

Acknowledgments

This project was partly financed by the province of Lower Austria, the European Regional Development Fund, the FIT-IT project ε-WiFi under contract 813319 in cooperation with Oregano Systems, and the EU under the FP7 STREP flexWARE Contract number 224350. Due to availability reasons, PHYs from Marvell were used in our measurements.

References

[1] LAN eXtensions for Instrumentation (LXI), LXI Standard, 2008. http://www.lxistandard.org.

[2] D. L. Mills, "Internet time synchronization: the network time protocol," *IEEE Transactions on Communications*, vol. 39, no. 10, pp. 1482–1493, 1991.

[3] J. Ridoux and D. Veitch, "Ten microseconds over LAN, for free (extended)," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 6, pp. 1841–1848, 2009.

[4] "IEEE standard for a precision clock synchronization protocol for networked measurement and control systems," in *Proceedings of the IEEE Standards Interpretations for IEEE Std 1588–2008 (Revision of IEEE Std 1588-2002)*, pp. c1–c269, Piscataway, NJ, USA, July 2008.

[5] A. Marco, R. Casas, J. L. S. Ramos, V. Coarasa, A. Asensio, and M. Obaidat, "Synchronization of multihop wireless sensor networks at the application layer," *IEEE Wireless Communications*, vol. 18, no. 1, pp. 82–88, 2011.

[6] "Specific requirements part 3: carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications," in *Proceedings of the IEEE Standard for Information Technology—Telecommunications and information exchange between systems—Local and metropolitan area networks*, IEEE Computer Society, New York, NY, USA, December 2008.

[7] P. Ferrari, A. Flammini, D. Marioli, and A. Taroni, "A distributed instrument for performance analysis of real-time ethernet networks," *IEEE Transactions on Industrial Informatics*, vol. 4, no. 1, Article ID 4475682, pp. 16–25, 2008.

[8] R. Ben-El-Kezadri and G. Pau, "TimeRemap: stable and accurate time in vehicular networks," *IEEE Communications Magazine*, vol. 48, no. 12, Article ID 5673072, pp. 52–57, 2010.

[9] P. Loschmidt, R. Exel, A. Nagy, and G. Gaderer, "Limits of synchronization accuracy using hardware support in IEEE 1588," in *Proceedings of the IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS '08)*, pp. 12–16, Ann Arbor, Mich, USA, 2008.

[10] D. Rosselot, "Application Note: DP83848 and DP83849 100Mb Data Latency," Tech. Rep. 1507, National Semiconductor, Santa Clara, Calif, USA, 2006.

[11] P. Loschmidt, *On enhanced clock synchronization performance through dedicated ethernet hardware support*, Ph.D. dissertation, Vienna University of Technology, Vienna, Austria, 2010.

[12] R. Exel and P. Loschmidt, "High accurate timestamping by phase and frequency estimation," in *Proceedings of the International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS '09)*, pp. 126–131, Brescia, Italy, October 2009.

[13] (2011, Mar) PLL Clock Management Features in Altera FPGAs. Altera Corporation, http://www.altera.com/support/devices/pll_clock/pll-overview.html.

[14] J. Kalisz, R. Szplet, J. Pasierbinski, and A. Poniecki, "Field-programmable-gate-array-based time-to-digital converter with 200-ps resolution," *IEEE Transactions on Instrumentation and Measurement*, vol. 46, no. 1, pp. 51–55, 1997.

[15] R. Szplet, J. Kalisz, and R. Szymanowski, "Interpolating time counter with 100 ps resolution on a single FPGA device," *IEEE Transactions on Instrumentation and Measurement*, vol. 49, no. 4, pp. 879–883, 2000.

[16] B. Amrutur, P. K. Das, and R. Vasudevamurthy, "0.84 ps Resolution clock skew measurement via subsampling," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pp. 1–9, 2010.

[17] X. Zhu, G. Sun, S. Yong, and Z. Zhuang, "A high-precision time interval measurement method using phase-estimation algorithm," *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no. 11, pp. 2670–2676, 2008.

[18] S. M. Kay, *Fundamentals of Statistical Signal Processing*, vol. 1 of *Estimation Theory*, Prentice Hall, New York, NY, USA, 1993.

[19] X. Dai and I. H. R. Gretsch, "Quasi-synchronous sampling algorithm and its applications," *IEEE Transactions on Instrumentation and Measurement*, vol. 43, no. 2, pp. 204–209, 1994.

[20] D. W. Allan, "Time and frequency (time-domain) characterization, estimation, and prediction of precision clocks and oscillators," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 34, no. 6, pp. 647–654, 1988.

[21] "IEEE Std. 1588–2002 IEEE standard for a precision clock synchronization protocol for networked measurement and control systems," in *Proceedings of the IEEE Standards Interpretations for IEEE Std 1588–2002*, pp. i–144, Piscataway, NJ, USA, November 2002, replaced by 61588-2004.