19th Symposium IMEKO TC 4 Symposium and 17th IWADC Workshop
Advances in Instrumentation and Sensors Interoperability
July 18-19, 2013, Barcelona, Spain

Time-Synchronous Sampling in Wireless Sensor Networks

Jürgen Funck[1], Clemens Gühmann[1]

[1] TU Berlin, Chair of Electronic Measurement and Diagnostic Technology, 10587 Berlin, Germany, E-Mail: juergen.funck@tu-berlin.de

Abstract- A large variety of time synchronization protocols for wireless sensor networks has been suggested. Yet, setting up a sensor network for synchronized data acquisition based on those algorithms is not a trivial task. This paper outlines two different approaches to time synchronous sampling in wireless sensor networks, discusses their advantages and disadvantages and gives recommendations on when to chose which approach.

Keywords: wireless sensor network, time synchronization, sampling

I. Introduction

In recent years intensive research has been done on the synchronization of clocks in wireless sensor networks. Summaries of the state-of-the-art protocols are given in [1-3]. Descriptions of individual time synchronization protocols can be found in [4-6]. Publications on synchronized sampling using wireless sensor networks are, however, much rarer. Yet some examples can be found [7-9].
When setting up a wireless sensor network for time synchronous data acquisition the design goals usually are to minimize the latency of data transmission while maximizing the network's availability as well as the quality and integrity of the data. Common challenges are the changing quality of wireless links over time [10] as well as the limited capacity of the batteries that often power the sensor nodes. Thus usually a tradeoff has to be found between the latency and data quality that benefit from frequent communication and the network availability that is mainly determined by the amount of energy used for communication and computation. In this context it is interesting to note that in a wireless sensor network the energy cost of computation is generally small compared to the cost of communication [11]. The software for wireless sensor nodes should be efficient and have a small memory footprint since wireless sensor nodes are usually equipped with microprocessors that are energy efficient but have little computational power and memory. Furthermore, the software should be reliable and easily extendible to ease its adaption to new sensing applications. This is best achieved by a modular software design with little or no interdependencies between the individual software modules.
Section II of this paper describes a wireless sensor network in terms of a generic multi-channel sampling system and outlines two different approaches to synchronized sampling. An overview of the effects of synchronization errors and clock updates by the synchronization protocol on the acquired signals is given in section III. Section IV contains a theoretical analysis of the approaches' properties. In section V conclusions are drawn from this analysis and the direction of future research is outlined.

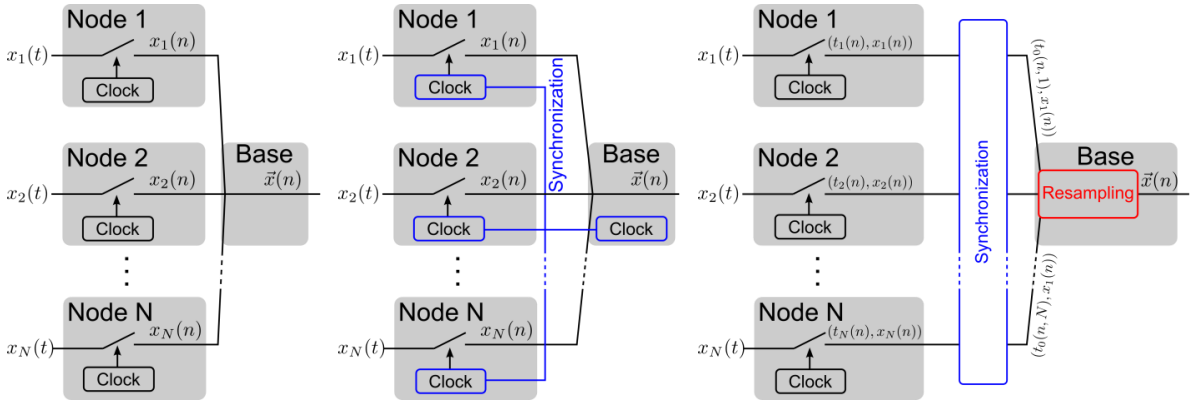
II. Approaches to Time Synchronized Sampling

A. Model of a synchronized sampling system

According to [12] a wireless sensor network can be modeled as a generic multi-channel sampling system. In such a system every channel samples a continuous signal $x_i(t)$ based on a series of trigger pulses. The trigger pulses are generated from a monotonically nondecreasing clock at predetermined values. At the output of the system the sampled signals $x_i(n)$ are combined to form a vector-valued signal $\vec{x}(n) = (x_1, x_2, \dots, x_N)$. We define synchronized sampling as the acquisition of samples $x_i(t(n))$ from all input signals for the same timing instants $t(n)$.
In a wireless sensor network one or more channels may be located on a single node. All nodes transmit the acquired data to a base station where the combined output signal $\vec{x}(n)$ is formed (see figure 1a).

B. Proactive and Reactive Synchronized Sampling

The literature on time synchronization for wireless sensor networks describes two classes of protocols [2,3]: a-priori and a-posteriori. A-priori synchronization protocols continuously synchronize clocks of the nodes in the network, so that every node always has an estimate of the network's global time. Well known examples of a-priori synchronization protocols are FTSP [4] and TPSN [5].



a) generic multi-channel sampling system

b) proactive synchronized sampling system

c) reactive synchronized sampling system

Figure 1. synchronized Sampling with wireless Sensor networks

A-posteriori protocols only synchronize the clocks once an event occurs, i.e. the estimate of the event's time of occurrence on the network's global timescale is calculated only after the event has occurred. An example of an a-posteriori synchronization protocol is RITS [6].
Based on those two classes of synchronization protocols we derive two approaches to synchronized sampling: a proactive and a reactive one. Proactive synchronized sampling uses a priori time synchronization to synchronize the clocks of the sensor nodes. Those clocks form a virtual common clock allowing for all acquisition channels to be sampled at the same timing instants (see figure 1b). In reactive synchronized sampling all sampling channels acquire samples independently from each other using their unsynchronized local clocks (see figure 1c). On acquisition the samples are timestamped according to the local clock of the sensor node. Later, the sample's local timestamps $t_i(n)$ are translated to timestamps $t_0(n,i)$ on the network's global timescale using a-posteriori synchronization. The samples acquired this way will in general not have been sampled at the same timing instants, i.e. $t_0(n,i) \neq t_0(n,j)$ for two nodes i and j. In order to obtain a truly synchronously sampled output signal $\vec{x}(n)$, the signals $x_i(n)$ from the individual sampling channels have to be interpolated and resampled at common timing instants $t_0(n)$.

C. Frequency of timing information

The triggering of a sample based on the common virtual clock in proactive synchronized sampling and the acquisition of a synchronized timestamp in reactive synchronized sampling are equivalent in the respect that both create a known point in time that can be used for further processing.
Either way the frequency with which this time information is included in the sampling process can be varied. It is possible to only trigger/timestamp the beginning or end of a block of n samples with a synchronized clock. The other samples can be triggered based on an unsynchronized local clock or left without timestamps. If needed the missing timing information can be reconstructed by interpolation.
The advantage of this is that in proactive sampling a clock with a lower but more stable frequency (e.g. watch crystal) may be used for the virtual common clock, while the triggering of the individual samples is done with a higher frequency but less stable one (e.g. RC-oscillator). In reactive sampling taking less timestamps reduces the amount of data to that has to be transmitted.

III. Effects of Synchronization Errors and Clock updates

As the clock of a wireless sensor node is used to generate the trigger pulses for sampling, the properties of the clock as well as any changes to the clock's value, i.e. by a synchronization protocol, have a direct influence on the timing instances at which samples are acquired. Ideally, the series of trigger pulses used for sampling is equally spaced with the same period and phase on all sensor nodes. The real series of trigger pulses will differ from this ideal due to clock errors. In real world data acquisition systems the true sampling instances are usually not known. Instead it is simply assumed that the samples were taken at the ideal sampling instances. In this case the errors in the sampling instances will translate into distortions in the acquired signal. In this section we analyze the signal distortions caused different kinds of clock errors or the synchronization protocol. Furthermore,

ISBN-10: 84-616-5438-2 | ISBN-13: 978-84-616-5438-3

we outline the possibilities to correct those distortions if information about the true sampling instances is available.

A. Effects of Clock Errors

In general, clocks exhibit three kinds of synchronization errors: offset, drift and jitter [1-3]. To illustrate the general effects of those errors, MATLAB simulations were done. The values for offset, drift and jitter are chosen arbitrarily and are not related to any real-world measurement setup. The simulation uses a sine signal with a frequency of $f_i = 5\ Hz$ and a sampling rate of $f_s = 128\ \frac{1}{s}$. The signal is sampled at ideal sampling instants t_{ideal} equally spaced in the interval $[0,1]$ as well as at non-ideal sampling instants t_{real} with 128 Samples each. The sampling instants, the acquired signal and the magnitude spectrum of the acquired signal for the ideal and non-ideal sampling are shown in figure 3.

A clock offset ΔT causes a time shift in the sampling instants (see equation 1). As can be seen from figure 3a it causes a phase shift in the acquired signal while leaving its magnitude spectrum unchanged.

$$t_{offset}(n) = t_{ideal}(n) + \Delta T \tag{1}$$

According to [2] a clock's drift ρ is defined as the difference between its rate f and the ideal rate of 1 (see equation 2). In this case the non-ideal sampling instances are given by equation 3. The drift is usually influenced by slowly changing quantities like temperature or battery voltage [2]. Thus it can be treated as constant for sufficiently short periods of time. In this case it only causes a frequency shift of the acquired signal (see figure 3b). If the drift changes during the time of acquisition it causes a non-linear distortion of the acquired signal (see figure 3c).

$$\rho(n) = f(n) - 1 \tag{2}$$

$$t_{drift}(n) = [1 + \rho(n)] \cdot t_{ideal}(n) \tag{3}$$

Every clock is affected by some degree of jitter, i.e. the clock's period randomly fluctuates around its nominal value (see equation 4). This, in general, causes a non-linear distortion of the acquired signal. However, the errors induced by jitter being usually small can be translated into amplitude errors and treated as additional amplitude noise degrading the signals signal to noise ratio (see figure 3e).

$$t_{jitter}(n) = t_{ideal}(n) + \mathrm{w(n)},\ \ \mathrm{w - white\ gaussian\ noise} \tag{4}$$

B. Effects of Clock Updates

In order to compensate for clock errors like offset or drift an a-priori synchronization protocol needs to update the clock from time to time [1-3]. The most straightforward way is to directly correct the value of the clock. This causes a step in the clock's value and effectively results in a high drift during one sampling period. The result is a non-linear distortion of the acquired signal (see figure 3e).

$$t_{step}(n) = \begin{cases} t_{ideal}(n) - \Delta T & n < N_0 \\ t_{ideal}(n) & n \geq N_0 \end{cases} \tag{5}$$

Alternatively the clock's value may be changed gradually, e.g. through controlling the clock's frequency with a software phase-locked loop [2]. While this method is less disruptive than the former, it still introduces a changing drift into the sampling clock and causes the acquired signal to be non-linearly distorted (see figure 3c).

C. Correction of Signal Distortions caused by Clock Errors

The knowledge of the exact acquisition times $t(n)$ of all samples $x(n)$ offers the possibility to correct the distortions to the acquired signal through interpolation and resampling.
If the sampling instants were affected only by offset and constant drift and the signal $x(t)$ was sampled at a rate greater or equal to its nyquist rate, the continuous signal $x(t)$ can be perfectly reconstructed from its samples $x(n)$ using sinc-interpolation. More practical interpolation methods using digital filters are discussed in [13]. A common feature of those interpolation methods is that the interpolation error decreases if the oversampling ratio is increased [13].

ISBN-10: 84-616-5438-2 | ISBN-13: 978-84-616-5438-3

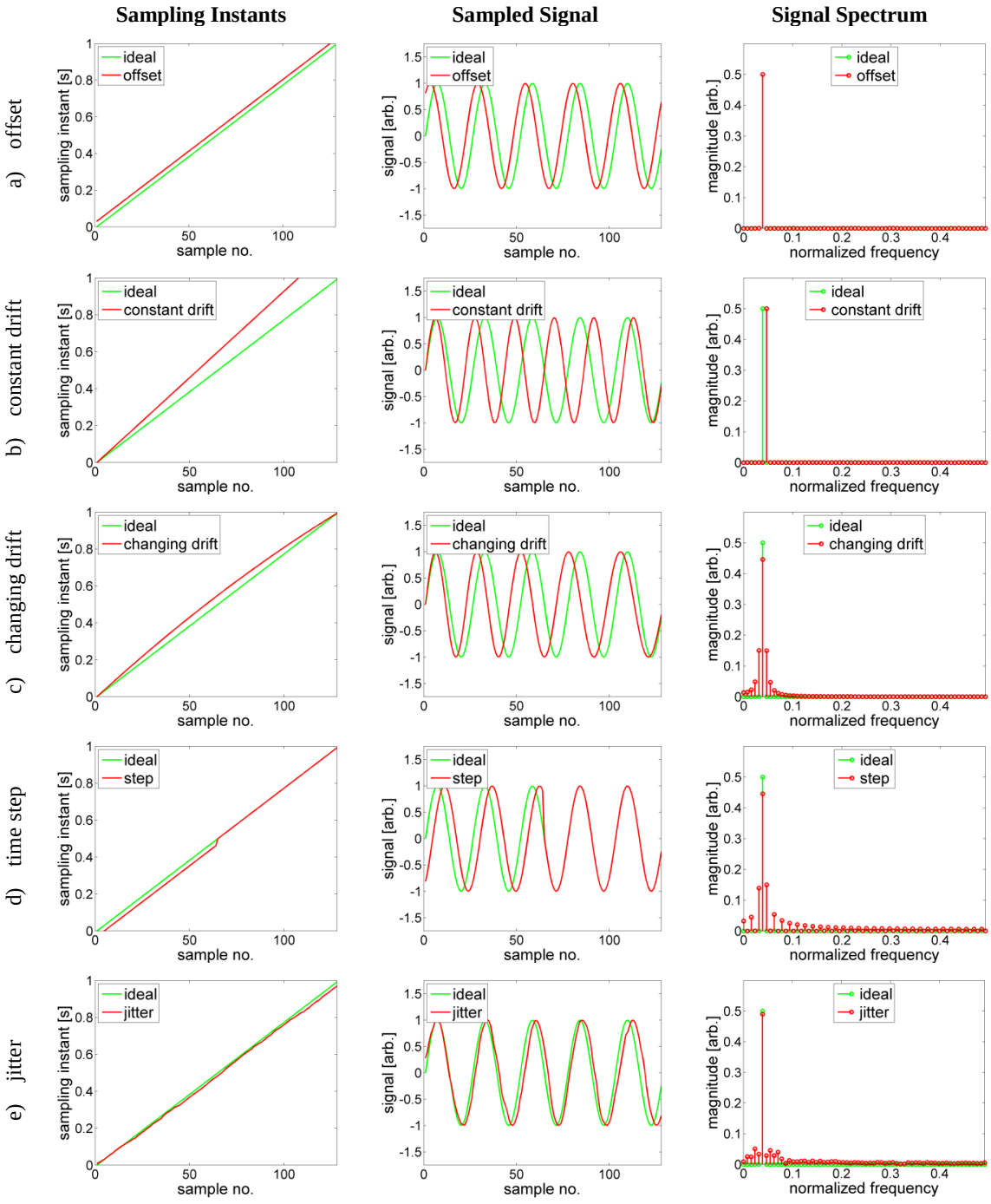


Figure 3: effects of synchronization errors and clock updates on the acquired signal

If the sampling instants were affected by a changing drift or jitter the samples are no longer uniform. In this case a perfect reconstruction of the continuous signal is in theory still possible, provided the average sampling rate was above the signals nyquist rate [14]. However, the Lagrange interpolation necessary for the prefect reconstruction from nonuniform samples is not practical [15]. Thus interpolation algorithms giving approximate only results have to be used in this case, e.g. sinc-interpolation [15]. Again the interpolation error of the approximate methods decreases as the oversampling ratio is increased [15].

ISBN-10: 84-616-5438-2 | ISBN-13: 978-84-616-5438-3

IV Analysis of Approaches

The proactive approach to synchronized sampling is the more intuitive one. Real-world implementations of synchronized sampling use the proactive approach, as it seems, exclusively [8,10,11]. All samples are acquired ideally at the same time on all channels so that no additional signal processing, i.e. interpolation and resampling, is needed after acquisition. Yet synchronization needs to be set up before and maintained during data acquisition. In networks using a regular communication scheme, as it is the case in many measurement setups (e.g. [8,11]), this comes at little additional cost as the synchronization information can be included in the beacons that are send periodically by the base station. If, however, the communication in the network is irregular, as it is often the case in sensor networks with a highly dynamic topology (e.g. [12]), the setup and maintenance of synchronized clock comes at a significant additional cost. Furthermore, the synchronization precision deteriorates the more time has passed since the last exchange of synchronization messages [2,3]. Thus to achieve a high precision in synchronization a high synchronization rate is needed. This increases the amount of communication and creates an additional dependency between data acquisition and communication which is contrary to the design of a modular sensor software.

The advantage of the reactive approach is that it largely decouples data acquisition and time synchronization, thus being advantageous for modular software implementations. No synchronization messages have to be exchanged when no data acquisition is done. There are no signal distortions due to clock updates, because no clock updates have to be done. Because a-posteriori synchronization is used there is no setup time needed to acquire synchronization before data acquisition [15]. Furthermore, the synchronization precision is independent of the rate of communication and comparatively high as the synchronization is always performed close to the time of sampling. The major disadvantage of the reactive approach is that timestamps have to be transmitted together with the samples, increasing the amount of communication. Another disadvantage is that the acquired signals have to be interpolated and resampled at the base station. This causes an additional computational burden. Keeping in mind that communication is energy-wise much more expensive than computation [4], this may still be a good tradeoff. The necessary interpolation also causes additional errors in the interpolated samples, as only approximate interpolation methods are practically feasible. This error, however, can be reduced by increasing the oversampling ratio for the acquired signal. If samples are lost from time to time on the wireless link the interpolation needed for the reactive approach may even be advantageous because missing samples can be reconstructed this way.

The frequency with which synchronized timing information is included in the sampling process mainly depends on the property of the underlying clock. Basically the inclusion of timing information is equivalent to sampling the clock's value. The reconstruction of synchronized timing information for the samples without explicit timing information is then equivalent to the interpolation of samples. Thus it is necessary to include synchronization timing information at a frequency higher than the rate of the changes in the clock's drift including clock updates by the synchronization algorithm. Significant oversampling is desired in this case too as it increases the quality of the interpolation. A higher frequency for the inclusion of synchronized timing information, however, also increases the computational burden on the sensor node and has to be well below the processors clock frequency in order to be feasible.

V Conclusions and Outlook

Our analysis shows two general approaches to time synchronized sampling: a proactive and a reactive one. The proactive approach is best suited for the acquisition of signals sampled close to the nyquist frequency over reliable wireless links with regular and frequent communication. If wireless links a less reliable, communication less frequent and significant oversampling is an option the reactive approach is more robust and easier to implement. Interdependencies in the sensor software can be greatly reduced by using the reactive approach as it largely decouples data acquisition from communication. An interesting possibility is the combination of both approaches. Samples would be taken based on a proactive sampling scheme and later their times of acquisition would be refined using a-posteriori time synchronization. The frequency with which timing information is included in the sampling process has to be chosen according on the clock's stability.

Future work will include the verification of our theoretical analysis by applying the two approaches to real-world synchronous sampling applications on a machine test bench. Furthermore, a comparative study of the applicability of the various interpolation methods for signal reconstruction in wireless sensor networks will be done.

ISBN-10: 84-616-5438-2 | ISBN-13: 978-84-616-5438-3

References

[1] S. M. Lasassmeh, J. M. Conrad, "Time Synchronization in Wireless Sensor Networks", *IEEE SoutheastCon 2010*, pp. 242-245, 2010.
[2] K. Römer, P. Blum, L. Meier, "Time synchronization and calibration in wireless sensor networks", *Handbook of Sensor Networks: Algorithms and Architectures*, Wiley, 2005.
[3] H. Karl, A. Willig, *Protocols and Architectures for Wireless Sensor Networks*, Wiley, 2005
[4] M. Maróti, B. Kusy, G. Simon, A. Lédeczi, "The flooding time synchronization protocol", *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pp. 39 – 49, 2004
[5] S. Ganeriwal, R. Kumar, M. B. Srivastava, "Timing-sync protocol for sensor networks", *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pp. 138 – 149, 2003
[6] B. Kusy, P. Dutta, P. Levis, M. Maroti, A. Ledeczi, D. Culler, "Elapsed time on arrival: a simple and versatile primitive for canonical time synchronisation services", *Int. J. Ad Hoc Ubiquitous Comput.*, Inderscience Publishers, vol. 1, pp. 239 – 251, 2006
[7] S. Chen, A. Dunkels, F. Österlind, T. Voigt, M. Johansson, "Time synchronization for predictable and secure data collection in wireless sensor networks", *6th Annual Mediterranean Ad Hoc Networking Workshop*, pp. 165 – 172, 2007
[8] H.-C. Lee, Y.-M. Fang, B.-J. Lee, C.-T. King, "The Tube: A Rapidly Deployable Wireless Sensor Platform for Supervising Pollution of Emergency Work", *IEEE Transactions on Instrumentation and Measurement*, vol. 61, pp. 2776 – 2786, 2012
[9] C. Medina, J. C. Segura, A. de la Torre, "A Synchronous TDMA Ultrasonic TOF Measurement System for Low-Power Wireless Sensor Networks", *IEEE Transactions on Instrumentation and Measurement*, vol. 62, pp. 599-611, 2013
[10] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, P. Levis, "Collection tree protocol", *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, pp. 1-14, 2009
[11] G. J. Pottie, W. J. Kaiser, "Wireless integrated network sensors", *Commun. ACM*, ACM, 2000.
[12] J. Funck, C. Gühmann, "Verfahren zur Messung von Synchronisationsfehlern in mehrkanaligen Messsystemen", *XXVI. Messtechnisches Symposium des Arbeitskreises der Hochschullehrer für Messtechnik e.V. (AHMT)*, 2012.
[13] R. W. Schafer, L. R. Rabiner, "A Digital Signal Processing Approach to Interpolation", *Proceedings of the IEEE*, vol. 61, pp. 692 – 702, 1973
[14] F. Marvasti (Editor), *Nonuniform Sampling*, Plenum Publishers, 2001.
[15] S. Maymon, A. V. Oppenheim, "Sinc Interpolation of Nonuniform Samples", *IEEE Transactions on Signal Processing*, vol. 59, pp. 4745 – 4758, 2011

ISBN-10: 84-616-5438-2 | ISBN-13: 978-84-616-5438-3