

# Clock Synchronization of TinyOS-based Sensor Networks with DCF77

Lars Niemann, Marcus Venzke, Christian Renner, and Volker Turau

Institute of Telematics  
Hamburg University of Technology  
Hamburg, Germany  
venzke@tu-harburg.de

**Abstract**—The paper presents an approach of applying DCF77 time radio signals to provide a clock with global time in sensor networks based on TinyOS. Some nodes containing DCF77 receiver hardware reliably decode time signals even if these are distorted to some extent. The clock with global time is provided by compensating clock drift. Time is distributed in the network with a protocol generating timestamps on the MAC layer.

## I. INTRODUCTION

Sensor network nodes contain their own local clocks. Frequently these are synchronized in the network or between neighboring nodes as required for several algorithms (internal synchronization). In most cases synchronization is not performed to global time because this would require a suitable time source. On the other hand, global time is required if data is measured on several nodes, aggregated, and should be interpreted outside the network.

This paper thus analyses how clocks in sensor networks can be synchronized to global time (external synchronization). It assumes the use of radio time signals as time source, which are transmitted in several countries. It discusses how these signals can be evaluated reliably in hardware and software, how local clocks can be synchronized to it, and how time can be distributed in the network. TinyOS and IRIS nodes are assumed as platform for a demonstrator.

## II. CLOCK SYNCHRONIZATION IN SENSOR NETWORKS

Two important protocols for time synchronization in sensor networks are the Flooding Time-Synchronization Protocol (FTSP) [1] and the TimeSync Protocol for Sensor Networks (TPSN) [2]. Both select a leader node whose local clock acts as time source. Its time is periodically broadcasted in packets also containing identification information. Other nodes receive and store it in conjunction with their local reception time and estimate the global time. With FTSP each node immediately retransmits the packets. In contrast, with TPSN the local estimation of global time is sent to other nodes in a tree structure.

TinyOS implements its local clock in software incrementing an integer from system start with 1kHz, 32kHz, or 1MHz. It implements clock synchronization in module TimeSyncC [3] as a combination of TPSN and FTSP. A tree of nodes is established using the node with lowest ID as root. Timestamps are generated at MAC layer when packets are sent or received [4].

This makes transmission time for timestamps small and deterministic, resulting in a higher accuracy clock synchronization, since the delay for waiting for a free channel is not included.

## III. DCF77 RECEIVER HARDWARE

To implement a clock with global time, a sensor network needs a source of the global time to be distributed in the network. Candidates are satellite systems (e.g. GPS) as well as terrestrial radio transmitters available in several countries. The latter requires receiver hardware with less complexity and lower energy consumption and is more suitable for sensor networks. The paper thus assumes the use of Germany's terrestrial long wave time transmitter DCF77. Its utilization requires receiver hardware contained in one, some, or all sensor network nodes.

The DCF77 time signals are transmitted on 77.5 kHz by the Physikalisch Technische Bundesanstalt close to Frankfurt. Its transmission range of 2000 km spans most of central Europe. Date and time are transmitted in packets with 1 bit/s. Bits and packets are exactly aligned to seconds and minutes of global time respectively. It is modulated with amplitude (AM) and phase modulation (PM). The latter permits decoding with higher accuracy and reliability. On the other hand, decoding AM allows much simpler hardware and three orders of magnitude less energy. It is thus more suitable for sensor networks [5].

The DCF77 receiver hardware module for IRIS nodes shown in Fig. 1 has been developed using AM demodulation [5]. It is based on the receiver module EM6 DCF with a ferrite



Figure 1. DCF77 receiver hardware developed for IRIS nodes

antenna of HKW-Elektronik requiring 250  $\mu\text{A}$  in a voltage range between 1.1 V and 3.6 V. The module is directly connected to the IRIS node's 51-pin expansion connector without voltage regulator. An IO pin is used as power supply allowing power-down to save energy. An interrupt line reads the demodulated, digital DCF77 signal. Two more switches connected to IO pins are used to enable test software for debugging and adjusting antenna. The antenna has to be mounted at least 1cm from the IRIS node to not obstruct reception.

#### IV. DECODING DISTORTED SIGNALS WITH HIGH ACCURACY

The receiver's digital signals have to be decoded by the sensor node's controller in software. Signals can be distorted, if radio reception is poor, e.g. due to unfavorable orientation of the antenna. The decoder needs to gracefully handle distortion and still regenerate time with high accuracy.

A combination of interrupt driven edge detection and periodic sampling should be applied to detect the edges of the receiver's distorted, digital signals with high accuracy and low processing time. Distortion results in quick erroneous changes between high and low as shown in Fig. 2 depicting two bits. Detecting all edges with interrupts would lead to a high interrupt rate requiring a lot of processing time. On the other hand, sampling with a reasonable rate reduces accuracy of edge detection and thus clock accuracy. The combination applies interrupts to detect falling edges marking the exact start of a second of global time. Afterwards a low sampling rate is used to determine the intended rising edge distinguishing 1 and 0 bits.

Distortion and processing in the receiver can shift the falling edge's position reducing clock accuracy. The latter is a property of the receiver, leading to a delay of up to 3 ms for module EM6 DCF in case of good signal quality. Distortion leads to varying delays. Measurements have shown that it can be reduced to 1 ms using Exponentially Weighted Moving Average (EWMA).

#### V. A GLOBAL TIME CLOCK FOR TINYOS

Node time synchronization is performed using timestamps received every minute and the falling edges of the DCF77 signal marking every second. Local node time depends on the internal or external oscillator of the microprocessor, so that clock drift is inferred. The latter has been observed to be as large as multiple seconds per day on the IRIS platform. Furthermore, clock drift is not a constant scaling factor, but varies among nodes and depends on the temperature.

As a result, we estimate the clock drift by comparing the local and global times elapsed since the last synchronization, i.e., timestamp generation via the DCF77 module. Smoothing of the estimation is performed using EWMA. An update-parameter  $\alpha=0.8$  has been empirically determined and used in

$$c = \alpha \cdot c_{old} + (1 - \alpha) \cdot c_{measure} \quad (1)$$

Besides updating the clock drift compensation factor  $c$ , the synchronization module stores the local time  $t_{sync}$  and the global time  $T_{sync}$  of the current synchronization process and discards



Figure 2. Digital signal with two distorted bits: 0 (left) and 1 (right)

earlier values. Global time  $T_t$  can thus be estimated at any (local) time  $t \geq t_{sync}$  via

$$T_t = T_{sync} + c \cdot (t - t_{sync}) \quad (2)$$

#### VI. DISTRIBUTING TIME IN THE SENSOR NETWORK

The global time must be distributed in the sensor network by radio if some nodes do not have a DCF77 receiver or cannot receive the time signal, e.g. due to bad reception conditions. In TinyOS this can be implemented based on the available module TimeSyncC (see section II). The module must be adapted to forward received packets as described for FTSP. In case of several nodes having access to external global time, the node with the best signal quality should be selected as leader.

Energy restrictions make leader selection an awkward issue. Nodes should remain in sleep mode and with radio receiver switched off as much as possible to save energy. Thus the reception of packets cannot be guaranteed. Dynamic changes in the network topology, such as the announcement of a new leader, are therefore hard to treat. An algorithm can deal with the issue in two ways. It can ignore energy issues leaving the radio transceiver in operation all times. Alternatively it can allow power save operation assuming a fixed leader sending timesync packets in fixed time intervals among which receiving nodes can sleep.

#### VII. CONCLUSION

DCF77 and similar radio time signals are an advisable global time source for sensor networks. Some nodes containing low power receivers listen to the omnipresent signals free of charge and distribute time to other nodes by radio. Good accuracy and low processing time are feasible even if signals are distorted to some extent. Simple arithmetic models allow synchronizing local clocks by adjusting clock drift.

#### REFERENCES

- [1] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi, "The Flooding Time Synchronization Protocol," In Proc. of the 2nd international Conference on Embedded Networked Sensor Systems, pp. 39-49, ACM, Nov. 2004.
- [2] K. Römer, P. Blum, and L. Meier, "Time Synchronization and Calibration in Wireless Sensor Networks," In: Ivan Stojmenovic (ed.): Handbook of Sensor Networks: Algorithms and Architectures, John Wiley & Sons, pp. 199-237, 2005.
- [3] "Component: tos.lib.ftsp.TimeSyncC," In TinyOS 2.0.2 Documentation of Interfaces and Components, University of California, Berkeley, July 2007. <http://www.tinyos.net/tinyos-2.x/doc/nesdoc/telosa/>
- [4] M. Maroti, and J. Sallai, "Packet-level time synchronization," Version 1.1, Draft Documentary, TinyOS Core Working Group, May 2008. <http://www.tinyos.net/tinyos-2.x/doc/pdf/tep133.pdf>
- [5] L. Niemann: Zeitsynchronisation mit DCF77 in TinyOS basierten Sensornetzen. Projekt work. Hamburg University of Technology, 2009.