Time Synchronization in Wireless Sensor Network Applications

Y.S. Hong and J.H. No

Department of Computer Engineering, Dongguk University, Seoul, Korea
{hongys, jhno}@dgu.ac.kr

Abstract. In most sensor network applications, events are time stamped with node's local time. However, energy is highly constrained resource in sensor networks. The purpose of this paper is to present a time-synchronization algorithm for sensor networks that aims at reducing the computation and communication energy expended by the algorithm. We use MAC-layer time stamping and estimate the clock drift rate and the offset in order to obtain high precision performance. Our algorithm works in two steps. In the first step, a spanning tree is built in the sensor network. In the second step, all nodes in the network synchronize their clocks to their parent nodes. We analyze and implement our time synchronization algorithm on Berkeley MicaZ platform and show that it can synchronize a pair of neighboring motes to an average accuracy of around one microsecond with communication complexity of *O(log n)*.

Keywords: time synchronization, sensor network, wireless communication.

1 Introduction

Applications such as environmental monitoring deploys a sensing network consisting of a large number of sensor nodes with limited energy resource. These sensor nodes need to maintain local clocks in order to time-stamp events. Due to the severe resource constraints in sensor nodes, the traditional time synchronization algorithms for distributed systems should be reevaluated for the sensor network. Register clocks used in wireless sensor networks, even initially synchronized with a standard clock, gradually deviate from each other over a period of time. Due to the unavoidable deviation of local clocks, network-wide time synchronization can be achieved by synchronizing clocks.

This paper proposes a time synchronization algorithm for the wireless sensor network. This approach is based on the accumulated time information in order to estimate the clock drift rate and the clock offset of sensor nodes. Our algorithm works in two steps. In the first step, a hierachical structure is built in the sensor network. Finally, all nodes in the network synchronize their clocks to their parent nodes.

This paper is organized as follows: Section 2 briefly describes existing time synchronization algorithms. In Section 3, we present the proposed time synchronization algorithm in detail and analyze the proposed algorithm. In Section 4, we

R. Obermaisser et al. (Eds.): SEUS 2007, LNCS 4761, pp. 429–435, 2007.

describe implementation and results from the experiment. The paper concludes in Section 5.

2 Approaches to Time Synchronization Schemes

Clock synchronization algorithms have been extensively studied in the past to ensure that the deviation between clocks remains bounded. Most clock synchronization algorithms try to guarantee on the maximum clock deviation by deterministic algorithms. Probabilistic clock synchronization algorithm is based on a remote clock reading to read the clock at a remote node with a minimum error [2]. Another probabilistic approach used the time transmission protocol to estimate the time at a remote node. In the time transmission protocol, a sequence of clock synchronization messages containing the transmitting node's time-stamp are sent to the target node [1]. The target node estimates the time on the transmitting node's clock based on the time-stamps on the synchronization messages and the message delay statistics. An approach to synchronize clocks via OS- or middleware architecture mechanism tried to reduce the scheduling delay [10]. The Network Time Protocol have been widely used to synchronize clocks in the internet domain.

In wireless sensor networks, however, nondeterminism in transmission time caused by the Media Access Control(MAC) layer of radio stack can introduce unexpected delay at each hop. In the Reference Broadcast Synchronization(RBS) algorithm [5], a reference message is broadcasted. The receiver nodes record their local time and exchange the recorded time between neighboring nodes. In this approach, additional message is necessary to communicate the local time-stamp between nodes. The Timing-sync Protocol(TPSN) [6] first creates a hierachical structure in the network and then performs pair-wise synchronization between parent and children nodes. Each node synchronize its local time to its reference node by exchanging two synchronization messages with its parent node. The Flooding Time Synchronization Protocol(FTSP) [12] synchronizes the time of a sender by exchanging a single time-stamp message between the sender and the receivers. Ideas from these protocols were used and enhanced in the proposed time synchronization protocol.

3 Time Synchronization Using the Accumulated Time Information

Our algorithm works in two steps. The first step of the algorithm is to create a spanning tree in the sensor network by broadcasting a *make-tree* packet starting from a root node. After a spanning tree is created, the root node initiates the synchronization stage by broadcasting a *sync* packet.

Our algorithm makes the following assumptions.

(a) There are n sensor nodes in the network and each sensor node has a unique identifier.

(b) Each sensor node maintains a 16-bit register as a clock that is triggered by a crystal oscillator.

(c) The clock drift rate between two physical clocks is bounded.

(d) Delays of successive synchronization message are independent of each other.

The time synchronization messages are periodically broadcasted by the sensor nodes at the same level in the tree structure.

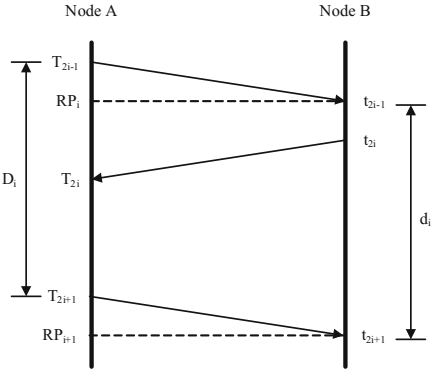

Fig. 1. Message exchange for time synchronization

Figure 1 shows the message exchange between node 'A' and node 'B' for time synchronization. Here, T_{2i-1}, T_{2i} and T_{2i+1} represent the time measured by local clock of node 'A'. t_{2i-1}, t_{2i} and t_{2i+1} represent the time measured by local clock of node 'B' . At time T_{2i-1}, 'A' sends a *sync* packet to 'B' . Node 'B' receives the packet at t_{2i-1}. At time t_{2i}, 'B' sends back an acknowledgement packet to 'A'. Node 'A' receives the packet at T_{2i}. These time variables satisfy following equations:

$$\begin{aligned} T_{2i-1} + p &= at_{2i-1} + b \\ T_{2i} &= at_{2i} + b + p \end{aligned} \tag{1}$$

Here p, a, and b denote the propagation delay, the clock drift rate and the clock offset between the nodes respectively. Node 'B' can calculate the clock drift rate, the clock offset and the accumulated time information M_i as:

$$M_i = (T_{2i+1} - T_{2i-1}) - (t_{2i+1} - t_{2i-1}) \tag{2}$$

$$\approx (D_i + d_i) \cdot \frac{\sum_{k=1}^{i}(D_k + d_k)}{\sum_{k=1}^{i}(D_k - d_k)} \tag{3}$$

$$a = \frac{T_{2i+1} - T_{2i-1}}{T_{2i+1} - T_{2i-1} - M_i} \tag{4}$$

$$b = \frac{(T_{2i+1} + T_{2i-1}) - a(t_{2i+1} - t_{2i-1})}{2} \tag{5}$$

Knowing the clock drift rate and the clock offset, node 'B' can synchronize to node 'A'. The proposed protocol reduces the jitter of interrupt handling by maintaining the accumulated time information M_i in each node.

The message exchange for the time synchronization at the sensor network begins with the root node of level 0. The time synchronization starts by broadcasting a *sync* packet. The *sync* packet contains 5 fields: the *refTimeStamp*, the *sendID*, the *childID*, the *levelID*, and the *driftRate*. The *refTimeStamp* contains the reference time for synchronization. The *sendID* and the *childID* contain the ID of the sender and one of children nodes, respectively. The *levelID* contains the level of the sender in the network. The *driftRate* is the estimated value of the clock drift.

On receiving this *sync* packet, children nodes calculate the clock drift rate and the clock offset and adjust their clock to the parent node. The children nodes broadcast *sync* packets to the grand children node and the parent node will overhear the *sync* packet. The parent node calculates a reference time-stamp by utilizing the packet as an acknowledgement packet from the child node.

This process is carried out throughout the network and we can achieve the network-wide time synchronization. Algorithm 1 describes the proposed time synchronization algorithm.

Algorithm 1. Time synchronization

Step 1. Create a spanning tree.

Assign level number to each sensor node by

broadcasting a *make-tree* packet

Step 2. Synchronize to the parent node

The root node initiates time synchronization

by broadcasting a *sync* packet

Repeat

On receiving the *sync* packet, children

nodes at the same level calculate

the clock drift rate and the clock offset

and broadcast a *sync* packet

until all sensor nodes are synchronized.

Let us analyze the communication complexity of the Algorithm 1. Step 1 takes *O(log n)* message exchanges. Step 2 also takes *O(log n)* message exchanges. Hence, The communication complexity of Algorithm 1 will become *O(log n)*.

Table 1 shows communication complexities for RBS, TPSN, FTSP and the proposed algorithm. If the resynchronization period is T seconds, then each node sends 1 message per T seconds in the proposed protocol. Each node sends 2 messages per T seconds in TPSN, 1.5 message per T seconds in RBS and 1 message per T seconds in FTSP [12].

Table 1. Communication complexities

Time synchroniztion Algorithm	Communication Complexity
RBS	$O(n^2)$
TPSN	$O(n)$
FTSP	$O(n)$
Proposed Algorithm	$O(logn)$

Since the proposed protocol employs a single broadcast message, it does not compensate for the propagation delay which is less than 1 microsecond for up to 300 meters.

There may be failures in sensor nodes or links. This situation may arise, when a level i node does not receive any *sync* packet from any neighbor at level *i-1*. When a node does not receive new *sync* packet for *TimeOut* number of resynchronization periods, it starts the root election process to select a new root node. In addition, when a new node is introduced to the network, the root election process will be needed in order to build a new spanning tree in the network.

4 Implementation

The proposed algorithm is implemented in the platform of Berkeley motes. Figure 2 shows the system for evaluating the time synchronization scheme described here.

The test system consists of four Berkeley MicaZ motes running Tiny OS 1.1.14. Three motes were used for time synchronization test and one of the motes was designated as a data collector connected to a PC running Windows XP in order to record the measured data. One of three motes was designated as the parent node and was responsible for broadcasting the synchronization message after every 10 seconds. The parent node was broadcasting time-stamped synchronization message to children nodes. The time-stamps were recorded on children nodes and the clock drift rate and the clock offset were calculated based on the accumulated time information. The synchronization error between a pair of motes is the absolute value of the difference of the recorded time-stamp and the corrected time-stamp.

The synchronization error from the experiment is summarized in Table 2. The results are obtained after averaging over 100 independent runs. The average and maximum time-stamping errors were 0.81 microsecond and 1.72 microsecond.

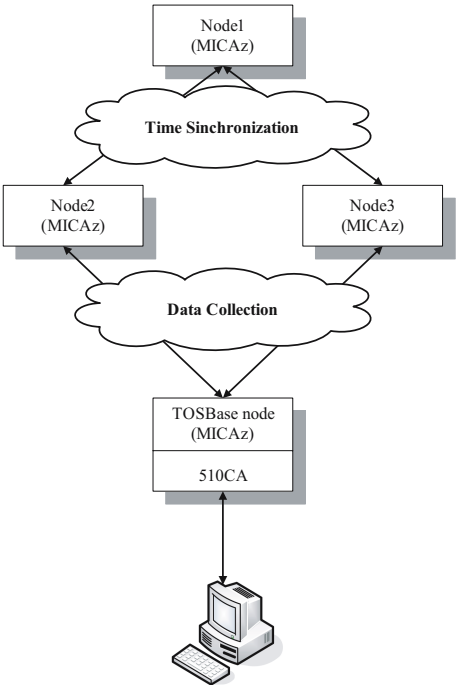

Fig. 2. Test environment

Table 2. Synchronization error from the experiment

	Synchronization error
Mean	0.81
Maximum	1.72
Variance	0.65

5 Conclusion

We have presented a time synchronization protocol for wireless sensor networks. The protocol uses the accumulated time information in order to estimate the clock drift rate and the offset. We have tested our protocol on the Berkely MicaZ platform with four motes. The measurements indicate that the synchronization error is in the range of one microsecond with communication complexity of $O(logn)$.

We plan to extend our experiment to the multi-hop networks of motes.

Acknowledgments. This research was supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Assessment).

References

1. Arvind, K.: Probabilistic Clock Synchronization in Distributed Systems. In: IEEE Trans. On Parallel and Distributed Systems, 5th edn, pp. 474–487 (1994)
2. Cristian, F., Fetzer, C.: Probabilistic Internal Clock Synchronization. In: Proc. of thirteenth Symposium on Reliable Distributed Systems, pp. 22–31 (October 1994)
3. Dai, H., Han, R.: TSync: A Lightweight Bidirectional Time Synchronization Service for Wireless Sensor Networks. Mobile Computing and Comm. Review 8, 125–139 (2004)
4. Dam, T.V., Langendoen, K.: An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks. In: The First ACM Conference on Embedded Networked Sensor Systems (Sensys03), Los Angeles, CA, USA, pp. 171–180 (2003)
5. Elson, J.E., Girod, L., Estrin, D.: Fine-Grained Network Time Synchronization using Reference Broadcasts. In: Proc. 5th Symp. Op. Sys. Design and Implementation, Boston, vol. 36, pp. 147–163 (2002)
6. Ganeriwal, S., Kumar, R., Srivastava, M.: Timing Sync Protocol for Sensor Networks. In: ACM SenSys, Los Angeles, CA, pp. 138–149 (2003)
7. Greunen, J.V., Rabaey, J.: Lightweight Time Synchronization for Sensor Networks. In: Proc. 2nd ACM Int'l. Conf. Wireless Sensor Networks and Apps, San Diego, CA, pp. 11–19 (2003)
8. Hong, Y.S., No, J.H.: Clock Synchronization in Wireless Distributed Embedded Applications. In: IEEE Workshop on Software Technologies for Future Embedded Systems, pp. 101–104. IEEE Computer Society Press, Los Alamitos (2003)
9. IEEE Computer Society. IEEE 802.15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs) (2003)
10. Kim, K.H(K.), Im, C., Athreya, P.: Realization of a Distributed OS Component for Internal Clock Synchronization in a LAN Environment. In: Proc. of the fifth IEEE Symposium on Object-Oriented Real-Time Distributed Computing, pp. 263–270 (2002)
11. Kopetz, H., Ochsenreiter, W.: Clock Synchronization in Distributed Real-Time Systems. IEEE Transactions on Computers C-36(8), 933–939 (1987)
12. Maroti, M., Kusy, B., Simon, G., Ledeczi, A.: The flooding time synchronization protocol. In: Proc. 2nd international conference on Embedded networked sensor systems, pp. 39–49 (2004)
13. Mills, D.L.: Internet time synchronization: the Network Time Protocol. IEEE Transactions on Communications 39, 1482–1493 (1991)
14. Mock, M., Nett, E., Frings, R., Trikaliotis, S.: Clock Synchronization for Wireless Local Area Networks. In: Proc. of the 12th Euromicro Conference on Real-Time Systems, Stockholm, pp. 183–189 (2000)
15. Romer, K.: Time Synchronization in Ad Hoc Networks. In: ACM MobiHoc '01, Long Beach, CA, pp.173-182 (October2001)
16. The TinyOS Project, `http://webs.cs.berkeley.edu/tos`