JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 23, 985-997 (2007)

A Power-Efficient Timing Synchronization Protocol for Wireless Sensor Networks*

CHIH-MIN CHAO AND YUNG-CHANG CHANG+
Department of Computer Science and Engineering
National Taiwan Ocean University
Keelung, 202 Taiwan
+*Department of Information Management*
Tamkang University
Taipei County, 251 Taiwan

Timing synchronization is essential to wireless sensor networks. TDMA-based MAC protocols depend on the accurate synchronization to conduct transmissions. Synchronized with each other, sensor nodes are able to determine when they should be turned on/off to achieve power management which is especially important in wireless sensor networks. In this paper, we propose a power-efficient timing synchronization protocol (PETSP) which aims to fulfill the synchronization job with the minimum energy consumption. In PETSP, all sensor nodes synchronize their timing to a designated target node. This mechanism speeds up the synchronization process. When the clock oscillation difference to the target node is collected, it is possible for a sensor node to *self-correct* its timing. This self-correction avoids exchanging synchronization messages all the time. Based on this observation, our PETSP achieves power conservation by enlarging the synchronization period. Simulation results show that our PETSP achieves synchronization fast. It also decreases 47% and 75% the power consumption and average maximum clock drift as compared to the IEEE 802.11, respectively. We have also implemented the proposed timing synchronization mechanism on MICA Mote sensor boards. The experiments verify that the protocol achieve timing synchronization in a real system.

Keywords: timing synchronization, IEEE 802.11, wireless sensor networks, multihop, power-efficient

1. INTRODUCTION

The rapid progress of hardware and wireless communication technologies has made wireless sensor networks possible. Such networks consist of many inexpensive wireless sensor nodes, each capable of collecting, processing, and storing environmental information. Sensor nodes operate in a distributed way and coordinate to fulfill a common task. Sensor nodes are normally battery powered and communicate to each other in a multihop manner. Wireless sensor networks have attracted intensive attention recently. A lot of potential applications for wireless sensor networks have been discussed, including environment monitoring, mobile object tracking, and navigation applications.

Timing synchronization is important to the operation of wireless sensor networks. TDMA-based MAC protocol and power management mechanism require sensor nodes to

Received September 15, 2006; accepted February 6, 2007.
Communicated by Ten H. Lai, Chung-Ta King and Jehn-Ruey Jiang.
* This research was supported by the National Science Council of Taiwan, R.O.C., under grant No. NSC 95-2221-E-019-013.

be synchronized [4]. Synchronization is also needed for the above mentioned applications in that the data collected from different sensor nodes can be interpreted in a meaningful way. Although many timing synchronization protocols have been proposed for traditional wireless ad hoc networks, they cannot be used directly in sensor networks because of the unique features and requirements of sensor networks. The number of sensor nodes in a sensor network is usually much higher than that of an ad hoc network. As the number of nodes increases, the transmission contentions upraise accordingly. As a result, the *scalability problem* occurs [7, 9]. For a dense network, timing frames can hardly be successfully transmitted and some users may not be able to synchronize with others. Another unique feature is that sensor nodes are not powerful devices. They have limited computational capacity and memory. Most importantly, sensor nodes have limited and non renewable battery energy. Considering these differences, new time synchronization protocol that is simple and power-efficient is necessary.

In this paper, we propose a Power-Efficient Timing Synchronization Protocol (PETSP) for wireless sensor networks. PETSP aims to achieve fast timing synchronization and minimal energy consumption at the same time. Fast synchronization is achieved by distinguishing different transmission priorities. Energy conservation is realized through adjusting oscillator frequency and reducing timing information exchanges.

The rest of this paper is organized as follows. We review some timing synchronization protocols in section 2. Section 3 describes the details of the proposed timing synchronization protocol. Simulation results are in section 4. Conclusions are given in section 5.

2. RELATED WORKS

In ad hoc networks, a distributed *timing synchronization function* (*TSF*) is specified in IEEE 802.11 WLAN standard [8] to fulfill timing synchronization among users. Each mobile host maintains a TSF timer with modulus 2^{64} counting in increments of microseconds. All hosts contend for transmitting beacon frames periodically. The interval between beacon frames is defined as the aBeaconPeriod, which specifies the length of a beacon interval and is identical for all hosts in the MANET. In other words, time is divided into a series of beacon intervals which are exactly aBeaconPeriod time units apart. The host who wins the contention will send a beacon frame containing a timestamp which derives from its TSF timer. Other hosts synchronize to the received timing when it is faster than their own. The TSF timer in each host is the summation of a variable *offset* and the host's clock. When a host receives a beacon and finds its own timing is slower than the timestamp in the beacon, it will add the timing difference to its *offset*.

An *adaptive timing synchronization procedure* (*ATSP*) is proposed in [7] to solve the scalability problem in wireless ad hoc networks. The basic idea behind ATSP is from the observation that only later (faster) timing synchronizes others in the IEEE 802.11 TSF. Thus, ATSP gives the host that has the fastest timing the highest priority to transmit beacons (by allowing the fastest node contends to transmit beacon every beacon interval). On the other hand, slower hosts' beacon transmission frequencies are reduced. This reduction is achieved by allowing a slower node i contends to transmit beacon every $I_{\max}$ beacon intervals, if no faster timing value is received by node i during these $I_{\max}$ beacon

intervals. When the system is in a *stable state*, the fastest node has a very high probability of sending beacons successfully while the other nodes have the lowest priority.

ATSP successfully alleviate the scalability problem; however, in some cases such as the fastest mobile host leaves, some mobile hosts' clocks may still differ from others for hundreds of microseconds. To overcome these problems, a revision of ATSP, which is called *tiered adaptive timing synchronization procedure* (*TATSP*), is proposed in [9]. The TATSP protocol revises the ATSP by adopting two different $I_{\max}$ values. The idea of TATSP is to quickly identify a small set of fastest nodes and giving them higher priority to transmit beacons. Both ATSP and TATSP achieve clock synchronization for mobile hosts and scale well in a single hop network. Unfortunately, these two algorithms fail to function well when applying to a multihop MANET. In addition, both ATSP and TATSP take a long time to synchronize to users that are multiple hops away. For example, in a four-hop linear topology A-B-C-D where A is the fastest host, hosts B and C are responsible of spreading the time information. For the ATSP protocol, host A has the highest priority while host B has the lowest priority. In such a case, host B will have less chance to transmit a beacon and thus hosts C and D can synchronize to host A after a long time.

A clock synchronization algorithm called *automatic self-time-correcting procedure* (*ASP*) is proposed in [11]. ASP solves the scalability problem in a 802.11-based multihop ad hoc network. ASP enables hosts to calculate the oscillator differences to the fastest host. And then, each host adjusts its own timing periodically to catch up the fastest one. ASP successfully mitigates the clock asynchronism occurred in IEEE 802.11 networks.

A synchronization solution for wireless sensor networks called *reference-broadcast synchronization* (*RBS*) is proposed in [5]. In RBS, nodes periodically broadcast reference beacons to their neighbors. Each of the receivers records, according to its local clock, the time of receiving the reference. Then, the receivers exchange their records and calculate their timing offsets to each other. Specifically, the offset between nodes i and j can be obtained by $\text{Offset}[i, j] = \frac{1}{m}\sum_{k=1}^{m}(T_{j,k} - T_{i,k})$, where m is the number of reference packets and $T_{r,b}$ represents r's clock when it received reference packet b. RBS incurs $O(n^2)$ overhead to achieve synchronization, where n is the number of receivers. In addition, each node must keep the timing records from all its neighbors. For a dense network, it is a severe burden for a low-cost sensor. To provide multihop (different broadcast domains) synchronization, additional control packets and mechanism are needed.

The synchronization mechanism in the *timing-sync protocol for sensor networks* (*TPSN*) [6] consists of two phrases. In the first level discovery phrase, a node is elected as the *root* and organizes the other sensor nodes into a multiple level hierarchy structure. In the second synchronization phrase, nodes synchronize to their next lower level neighbors. Eventually, all nodes are synchronized to the root node. Constructing the hierarchy produces extra overhead. In addition, nodes in different levels or even the root node may fail. This can invalidate the scheme.

Another solution, the *Flooding Time Synchronization Protocol* (*FTSP*), is proposed in [10]. FTSP combines the RBS and TPSN: nodes synchronize to the single root node that periodically broadcasts reference packets. This protocol avoids the overhead of initial constructing phrase of TPSN; it also keeps away from the multiple broadcast domains problem of RBS. However, storage overhead and nodes failures still bother FTSP.

All these protocols, including IEEE 802.11 TSF, ATSP, TATSP, ASP, RBS, TPSN,

and FTSP, do not consider the energy issue which we believe is critical in wireless sensor networks.

3. PROTOCOL DESCRIPTION

Before describing our protocol, we first list some design issues of power-efficient timing synchronization protocols for wireless sensor networks.

- *Synchronization Target*: The target which all other nodes synchronized to can be either a single node (the fastest or a designated one) or multiple nodes. Using one single node as the synchronization target is relatively simple and efficient. The cost is initializations, such as the root/fastest node determination, may be necessary. In addition, the failure of the selected node can be a big problem. Employing multiple target nodes avoids the single point of failure problem. However, synchronized to multiple target nodes implies increased storage and transmission (energy) overhead, which is undesirable for wireless sensor networks.
- *Re-synchronization Period*: Since the frequencies of crystal oscillators in sensor nodes are not exactly the same, periodically re-synchronization is necessary to offset the clock drifts. Most solutions in the literature only rely on the received timing packets/references to synchronize sensor nodes. Such solutions have a short re-synchronization period and consume a lot of energy. If each node is able to obtain its timing difference to the target and adjust its own frequency by itself, the re-synchronization can be conducted less frequently. To conserve energy, prolonging the re-synchronization period is preferable. Note that it is impossible to remove re-synchronization totally because the oscillation frequency of each node's clock is time-variant.
- *Priority of transmission*: To synchronize a multihop network, the timing packets have to be forwarded to remote nodes. A naive scheme is to allow every node to forward the timing packet immediately after receiving it. This basic flooding scheme incurs a lot of contentions and takes a long time to synchronize the whole network. To relay the timing packet fast, giving priority to some nodes may help. A prioritization mechanism may reduce packet collisions and hence lower unnecessary energy consumptions.

3.1 Power-Efficient Timing Synchronization Protocol

Now we present the proposed PETSP. Similar to IEEE 802.11, time is divided into a series of beacon intervals. Each beacon interval is τ seconds apart. Timing packets (beacons) can be broadcasted at the beginning of each beacon interval. IEEE 802.11 CSMA/CA is adopted as the MAC mechanism. Each node has the same transmission range with radius r. The synchronization target is a single node. With frequency adjustment, sensor nodes' clocks are running at similar accuracy and thus re-synchronization period is extended. To rapidly forward timing packets, nodes that can cover more extra area are giving higher priorities.

We designate the node with the smallest ID as the synchronization target. The selection of target node is done before network deployment. This node is responsible to initiate beacon transmissions to synchronize the network. Although we use single target, to

avoid the single point of failure problem, a backup multiple-target mechanism is adopted. In the beginning, only the smallest-ID node will broadcast timing packets. In order not to consume too much energy, a parameter BT is defined to limit the number of beacon transmissions for a sensor node. If no timing packet is received in δ seconds, the original target node is assumed to be failed and multiple pre-selected targets (the nodes with smallest IDs next to the original target) will be activated to synchronize the networks. The number of necessary targets S depends on the stability of sensor nodes. This concept is shown in Fig. 1. Both τ and δ are system-dependent parameters. Note that the synchronization target is still a single node: the alive node with the smallest ID. However, to speed up the synchronization process, multiple nodes are activated concurrently instead of activating them one after another. When a node receives a beacon that has a higher-ID than that it has received, the newly received beacon will simply be discarded. Otherwise, the beacon will be forwarded to synchronize other nodes.

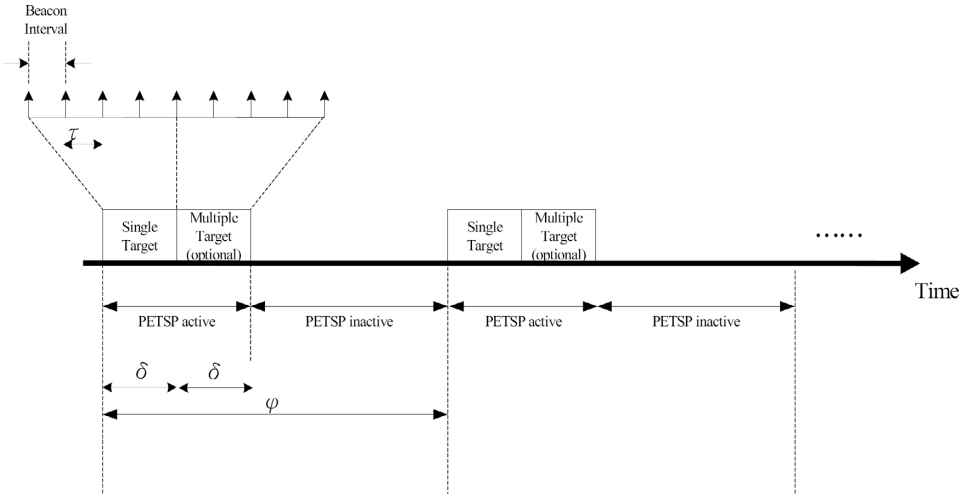

Fig. 1. PETSP active and inactive schedule.

To enable frequency adjustment, each node has to collect enough synchronization target's timing information. Each node i maintains a variable f_i which indicates the frequency to increase/decrease its clock in order to be synchronized to the target. Without loss of generality, we assume that the minimum time unit is microsecond in this paper. If f_i is positive/negative, one microsecond is increased/decreased to node i's clock every f_i microseconds. Assume that node i receives two beacons from the target node (or the same forwarding node) j at times $t1$ and $t2$, respectively. Let the timestamps in the beacons be $TS1$ and $TS2$, respectively. The adjustment frequency f_i can be calculated as follows.

$$f_i = \frac{t2 - t1}{(TS2 - TS1) - (t2 - t1)} \tag{1}$$

Enabling sensor nodes to correct their frequencies by themselves makes discontinuous re-synchronization possible. Discontinuous re-synchronization is equivalent to prolonging hosts' re-synchronization period. As shown in Fig. 1, PETSP is not always active. The synchronization process can be stopped after sensor nodes are synchronized and restarted when necessary. The re-synchronization period ϕ is also a system-dependent parameter and can be customized for different networks. It is believed that such a discontinuous re-synchronization mechanism can significantly reduce sensor nodes' power consumption. Note that a node may receive different timing information from different nodes. It is caused by each node's time-variant clock frequency. To avoid frequent adjustment in the PETSP active period, newly received smallest-ID beacons will be discarded when a node is able to adjust its frequency.

When a beacon is received, sensor nodes will help to forward it. The packet will be re-timestamped by each forwarding node. To achieve fast and efficient forwarding, we assign higher priorities to nodes that can relay the beacon farther. Specifically, several priority levels may be defined according to the estimated distance to the transmitter. A node that is nearer to the transmitter implies less extra area can be covered if the node re-broadcasts the timing information. Thus, lower priority is given. To keep our protocol simple, we define two priorities. The higher priority nodes generate their backoff numbers on the first half of the contention window when they try to forward the received beacon. On the other hand, the lower priority nodes obtain their backoff numbers on the second half. The number of beacon forwarding is also limited to BT. In each beacon interval, a node will stop forwarding the beacon if another beacon is received from another node.

The following variables and terms are maintained for each node i in our protocol.

1. An adjustment flag ADJ_i. This flag is set when node i is capable to adjust its clock frequency.
2. An adjustment variable f_i which is calculated from Eq. (1).
3. A counter bt_count_i which counts for the number of beacons that has been transmitted by node i. Initially, bt_count_i is set to zero.
4. A Clock_Table to keep track of meaningful timing information.

In the first 2δ of every ϕ seconds, the PETSP is active. Each node i will execute the following algorithm.

Algorithm PETSP
The smallest-ID node will activate itself as the synchronization target. Setting its transmission priority to high.

1. In each beacon interval, node i will attempt to transmit a beacon (following the IEEE 802.11 CSMA/CA operation), if the following conditions hold:
   (a) node i is a transmission target or ADJ_i is set,
   (b) node i has transmitted less than BT times, and
   (c) no beacon is received from other nodes in the same beacon interval.

Once a beacon is successfully transmitted, node i will increase its bt_count_i by one.

2. If a beacon is received, the timing information in the beacon is recorded in node i's Clock_Table if any of the following conditions holds:
   (a) node i has an empty Clock_Table,
   (b) the received beacon has a smaller ID than the one in node i's Clock_Table.

Once the timing information is recorded, clear ADJ_i and reset bt_count_i. If the received beacon has the same ID as the one in node i's Clock_Table, the following steps will be taken:

(i) calculate f_i (using Eq. (1)),
(ii) set ADJ_i, and
(iii) determine node i's transmission priority: low priority is given if it is within the circle of radius $r/2$ of the transmitter while high priority is given otherwise.

3. If no beacon is received in δ seconds, the backup nodes will activate themselves as target nodes and setting their transmission priorities to high.
4. If ADJ_i is set, sensor node i automatically adjusts its clock by one every f_i microseconds.

4. PERFORMANCE EVALUATION

We have implemented a simulator using *C#* to evaluate the performance of the proposed PETSP. We use 224 μs as the maximum tolerable clock drift since it is the duration, specified in the IEEE 802.11 standard, for the PHY to hop to another frequency in FHSS. A beacon interval is 0.1 second long. The PETSP active duration (δ) and the re-synchronization period (ϕ) are 10 and 40 seconds, respectively. The differences in accuracy of hosts' clocks are uniformly distributed in the range of [– 0.01%, + 0.01%]. Since hardware is not perfect, the oscillation frequency of each node's clock is time- variant. This time-variant clock drifts are uniformly distributed in the range of [– 3μs, + 3μs] per second. The number of sensor nodes is 500. Each of them is randomly located in a region of 1,000 × 1,000 square meters with a transmission range of 250 meters. The energy consumption model, described in Cardei *et al*. [3], was employed. Power consumption for transmit, receive, idle, and sleep modes was 700, 360, 340, and 30 mW, respectively. The initial energy was set to 500 Joules for each node.

4.1 Determining *S* and *BT*

When the single target node is failed, multiple backup targets will be activated. We have to decide the number of these backup nodes before deployment. Assuming that the failure probability of sensor nodes is p for a particular network. The success probability for S backup nodes, which is defined as at least one among the S nodes is not down, is $1 - p^S$. The results of success probability for different values of S and p are in Fig. 2. We found in all the tested failure probability, the value of S should be 6 or higher if we want to keep the success probability higher than 0.99. Thus, in the following simulation, we set S to be 6.

The most important job for PETSP is to achieve timing synchronization since it is a

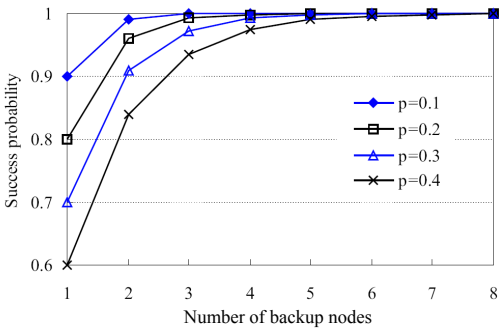

Fig. 2. Effect of S on success ratio.

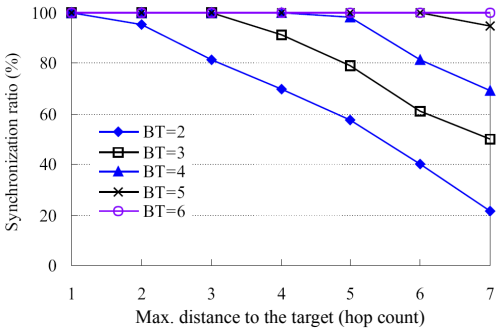

Fig. 3. Effect of BT on synchronization ratio.

timing synchronization protocol. In this experiment, we investigate the value of BT that can achieve this purpose. The metric is synchronization ratio which stands for the percentage that sensor nodes are synchronized to the target. Fig. 3 shows the result. The x-axis means the maximum distance (in hop count) a sensor node is away from the target. We can see the synchronization ratio decreases inversely proportional to the maximum distance. For example, when BT = 2, the synchronization ratios are 95.3% and 57.5% when the maximum hop counts are 2 and 5, respectively. A large BT achieves better synchronization ratio. It is reasonable since the transmitted beacon may collide with other transmissions. More transmissions increase the chances to successfully synchronize the network. Since our simulation environment is six hops at most, according to Fig. 3, we set BT = 5 because it is the minimum value of BT that can perfectly synchronize the network.

4.2 Maximum Clock Drift

In this experiment, we compare the synchronization performance of IEEE 802.11, ASP, and PETSP. We compare with IEEE 802.11 since it is a wide-accepted standard for wireless networks. It should be confirmed if the standard method works for wireless sensor network or not. The ASP is selected since it is capable of adjusting nodes' clock frequencies and performs well. The comparison is shown in Fig. 4. We can see that the values of the maximum clock drift are large and vary acutely for IEEE 802.11 TSF. The clock drift is over 224 μs for 378 times and the average is 337 μs for the entire simulated 500 beacon intervals. The performance of ASP is shown in Fig. 4 (b). It obviously performs better than IEEE 802.11 TSF; however, there is still some room for improvement. The average is 131 μs. In contrast to IEEE 802.11 TSF and ASP, PETSP performs quite well as shown in Fig. 4 (c). All nodes come to a stable state at a simulation time of 5 seconds. And then, since PETSP is inactive (at time 10 seconds) and there is oscillator error in each node, the maximum clock drift increases with time. The average clock drift is 86 μs. Our PETSP reduces up to 75% and 35% over the IEEE 802.11 and ASP, respectively.

4.3 Synchronization Speed

Here we survey the time it takes to synchronize the whole network. As shown in Fig. 5, our PETSP synchronize the network the fastest, which is followed by ASP and IEEE

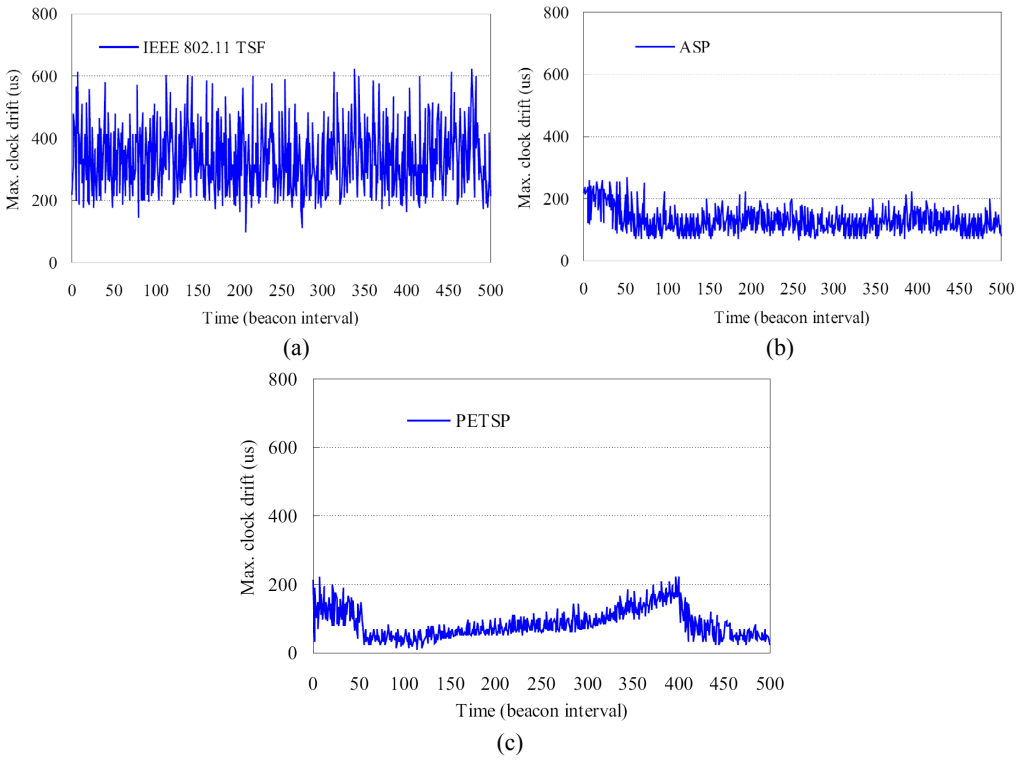

Fig. 4. Maximum clock drift vs. simulation time.

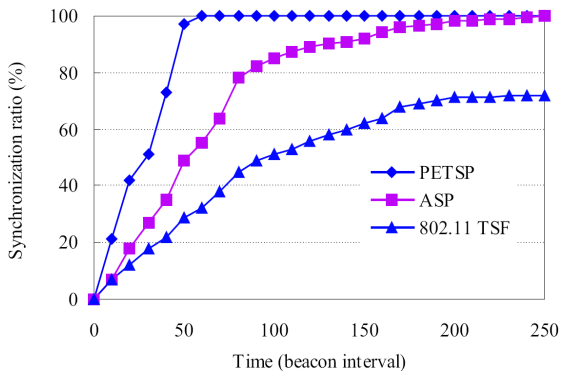

Fig. 5. Synchronization rate vs. simulation time.

802.11 TSF, respectively. For example, for PETSP and ASP, 10 and 25 seconds are needed to achieve nodes synchronization. For IEEE 802.11, only 72% of the nodes are synchronized to the target at a time of 25 seconds. We believe it is because of the severe collision problem caused by dense deployment of sensor nodes. Such out-of-synchronization condition is unsatisfactory, which means IEEE 802.11 can not apply to wireless

sensor networks directly. Spending less time to achieve synchronization means less beacon transmissions are needed for synchronization purpose. It is beneficial since it helps to reduce power consumption for sensor nodes.

4.4 Power Consumption

In this experiment, we examine the power consumption for different protocols. Again, as shown in Fig. 6, our PETSP has the best performance and IEEE 802.11 TSF is the worst. Nodes that run IEEE 802.11 TSF exhaust their energy at a time of 300 seconds. At the same time, all nodes that run PETSP remain alive. Nodes running ASP exhaust their energy at a time of 345 seconds while 73% of nodes running PETSP are still alive then. The proposed PETSP prolongs 47% and 28% of the lifetime when compared to IEEE 802.11 TSF and ASP, respectively. These results verify the excellence of our PETSP in energy-sensitive sensor networks.

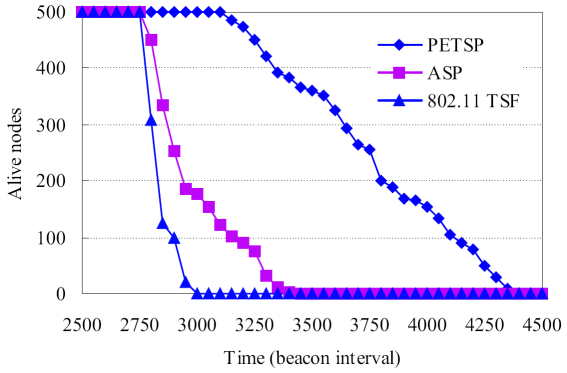

Fig. 6. Survival nodes vs. simulation time.

4.5 Implementation on Motes

In addition to simulations, we have also implemented the proposed clock synchronization mechanism on MICAz Motes [1] to verify its practicability. Motes are initially developed by U.C. Berkeley EECS Department. This battery-powered platform integrates sensors, computation, and communication. Specifically, the platform consists of processor radio boards, sensor and data acquisition cards, and gateways that interfaces Motes to personal computers. Motes runs TinyOS [2] which is written in NesC (a new language for programming structured component-based applications). TinyOS is an open-source operating system. It is event-based and is designed for wireless embedded sensor networks.

We have implemented our clock synchronization mechanism on five nodes. A beacon interval is 0.5 second long. The PETSP active duration (δ) is 30 seconds while different re-synchronization periods are simulated. Each node reports its timing to the gateway every minute. We found that each node's oscillator is indeed time-variant. The error is 0.8 to 1.2 μs per minute. Three different ϕ values are tested. The experiment results are

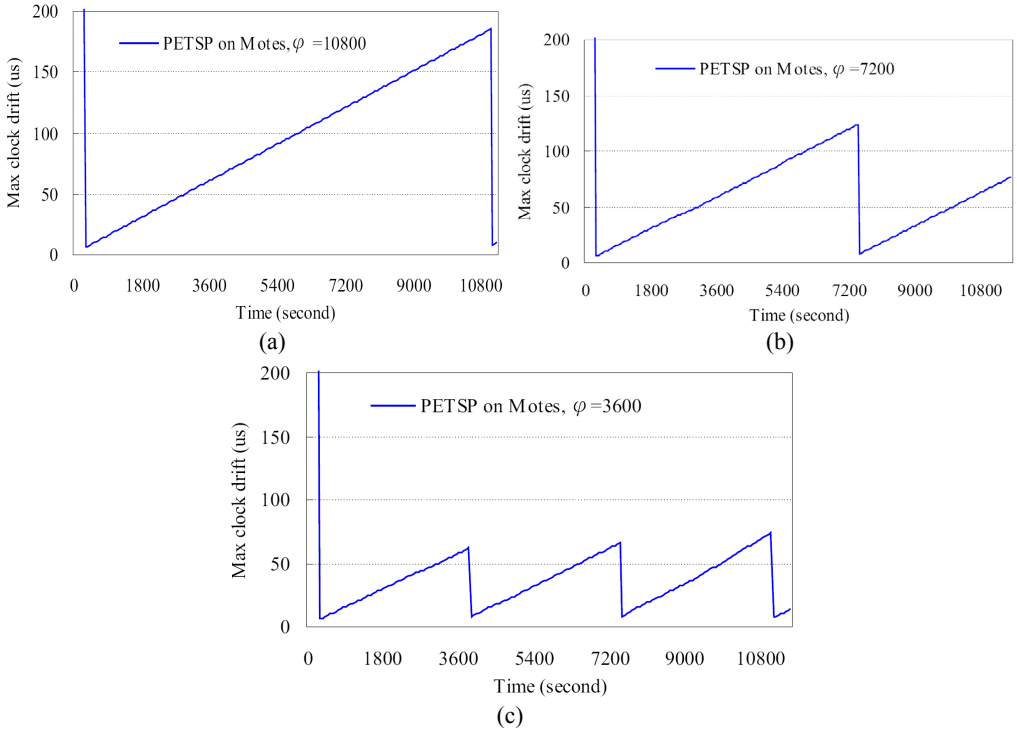

Fig. 7. Maximum clock drift in 5 Motes with (a) ϕ = 3,600 seconds, (b) ϕ = 7,200 seconds, and (c) ϕ = 10,800 seconds.

shown in Fig. 7. Generally, the increased clock drifts are proportional to simulation time. To keep the maximum clock drift between hosts below 200 μs, the value of ϕ can be as large as 10,800 seconds (180 minutes). If the time synchronization requirement becomes stringent, say, the maximum clock drift between hosts is 80 μs, the value of ϕ should be less than 3,600 seconds. This implementation confirms that our PETSP is a practical solution.

5. CONCLUSIONS

We proposed a power-efficient clock synchronization mechanism (PETSP) for wireless sensor networks. Sensor nodes running PETSP synchronize to a single target. With the frequency adjustment (self-correcting), hosts are able to extend the re-synchronization period and hence the power consumption is reduced. Simulation results show that PETSP is a fast and power-efficient timing synchronization protocol. We have also implemented PETSP on Motes. This implementation confirms our PETSP works well in reality, even through each node has a time-variant oscillator. For a wireless sensor node where energy is a critical issue, our PETSP is a promising solution to obtain clock synchronization with minimum power consumption.

REFERENCES

1. Crossbow Technology Inc., http://www.xbow.com, 2006.
2. TinyOS, http://www.tinyos.net, 2006.
3. M. Cardei, M. T. Thai, Y. Li, and W. Wu, "Energy-efficient target coverage in wireless sensor networks," in *Proceedings of the INFOCOM*, 2005, pp. 13-17.
4. C. M. Chao, J. P. Sheu, and I. C. Chou, "An adaptive quorum-based energy conserving protocol for IEEE 802.11 ad hoc networks," *IEEE Transactions on Mobile Computing*, Vol. 5, 2006, pp. 560-570.
5. J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," in *Proceedings of the 5th Symposium on Operating Systems Design and Implementation*, 2002, pp. 147-163.
6. S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, 2003, pp. 138-149.
7. L. Huang and T. H. Lai, "On the scalability of IEEE 802.11 ad hoc networks," in *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2002, pp. 173-182.
8. IEEE 802.11 Standard – IEEE Computer Society LAN MAN Standards Committee, Wireless LAN Medium Access Control and Physical Layer Specifications, 1999.
9. T. H. Lai and D. Zhou, "Efficient and scalable IEEE 802.11 ad-hoc-mode timing synchronization function," in *Proceedings of the 17th International Conference on Advanced Information Networking and Applications*, 2003, pp. 318-323.
10. M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The flooding time synchronization protocol," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, 2004, pp. 39-49.
11. J. P. Sheu, C. M. Chao, and C. W. Sun, "A clock synchronization algorithm for multihop wireless ad hoc networks," in *Proceedings of the 24th International Conference on Distributed Computing Systems*, 2004, pp. 574-581.

Chih-Min Chao (趙志民) received the B.S. and M.S. degrees in Computer Science from Fu Jen Catholic University and National Tsing Hua University in 1992 and 1996, respectively, and the Ph.D. degree in Computer Science and Information Engineering from National Central University in January of 2004. He was with SENAO International in 1996. He was an assistant professor at the Tamkang University, Taiwan, from 2004 to 2005. Since 2005, he has been an assistant professor with the Department of Computer Science and Engineering, National Taiwan Ocean University, Taiwan. His research interests include mobile computing and wireless communication.

Yung-Chang Chang (張永昌) received his B.S. degree in Information Management from Chinese Culture University in 2000 and received his M.S. degree in Information Management from Tamkang University in 2006, respectively. He worked for the Quanta Computer Inc. as a research and development engineer from 2006 until now.