IEICE Electronics Express, Vol.9, No.7, 691–697

# Precise time synchronization based on ripple flooding in wireless sensor networks

**Jinyoung Yang**[a)] **and Daeyoung Kim**
*Korea Advanced Institute of Science and Technology*
*335 Gwahak-ro, Yuseong-gu, Daejeon, Korea*
a) *jyyang0308@kaist.ac.kr*

**Abstract:** Precise time synchronization is inevitable for duty-cycling and TDMA in wireless sensor networks. To achieve a precise synchronized clock between nodes, fast distribution of time information of a reference node to all other nodes in multi-hop without a scheduling is necessary. In this letter, we propose a time synchronization algorithm, RFTS (Ripple Flooding Time Synchronization), that presents the fastest distribution of time information of a reference node by using synchronized packet broadcasting instead of CSMA-CA based broadcasting. We show that error in any hops is not affected by a prior hop node in the evaluation, average error and distribution time of RFTS outperforms widely used FTSP by a factor of 2.5 and 2 respectively.
**Keywords:** sensor networks, time synchronization, linear regression
**Classification:** Wireless communication hardware

## References

[1] D. Yuan, J. Yang, and D. Kim, "Ripple Flooding in Wireless Sensor Networks," *Proc. 8th ACM PE-WASUN 2011*, Miami Beach, USA, pp. 41–48, Nov. 2011.
[2] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi, "The Flooding Time Synchronization Protocol," *Proc. 2nd SenSys '04*, Baltimore, USA, pp. 39–49, Nov. 2004.
[3] T. Schmid, P. Dutta, and M. B. Srivastava, "High-Resolution, Low-Power Time Synchronization an Oxymoron No More," *Proc. 9th IPSN '10*, Stockholm, Sweden, pp. 151–161, April 2010.
[4] L. Ma, H. Zhu, G. Nallamothu, B. Ryu, and Z. Zhang, "Impact of Linear Regression on Time Synchronization Accuracy and Energy Consumption for Wireless Sensor Networks," *Proc. MILCOM 2008*, San Diego, USA, pp. 1–7, Nov. 2008.
[5] J. L. Devore, "Simple Linear Regression and Correlation," *Probability and Statistics for Engineering and the Sciences*, 4th edition, pp. 474–522, Brooks/Cole Publishing Company, 1995.


DOI: 10.1587/elex.9.691
Received February 27, 2012
Accepted March 21, 2012
Published April 11, 2012

## 1 Introduction

WSNs (Wireless Sensor Networks) are networks of many sensor nodes that are distributed and wirelessly connected with multi-hop fashion. In WSNs, coordination of wake-up and sleeping time, TDMA schedules, ordering of collected sensor data and events and cooperation of multiple sensor nodes require sensor network nodes to have a precise common time. One of methods to achieve the precise common time is time synchronization. Each node in WSNs has a cheap and imperfect hardware clock and each clock in the nodes may drift away each other in time. We need to compensate drift and offset between each clock of nodes networked with multi-hop periodically for keeping a precise common timescale over the nodes by using time synchronization protocols.

Several time synchronization protocols have been proposed to achieve the precise common timescale over the nodes. By using a reference signal in a transceiver shared with 1-hop neighbor nodes and an elaborate time stamping method set on the signal, the Flooding Time Synchronization Protocol (FTSP, [2]) provides a precise common timescale to the neighbors. In FTSP, the reference node floods its global time periodically and 1-hop neighbor nodes receive the sync message. After receiving sync messages enough to estimate the global time, 1-hop neighbor nodes become synchronized status and start flooding with their estimation of global time. The next-hop neighbor nodes behave in the same manner. FTSP achieves average 1.48 $\mu$s and 3 $\mu$s synchronization error in 1-hop and in 6-hop respectively using a 7.37 MHz system clock, and convergence time is 14 minutes. As the result of FTSP indicates, the synchronization error grows average 0.5 $\mu$s per hop and convergence time also increases with the size of network because at least three sync messages are required to rebroadcast the global time estimated by its own clock.

In this letter, we propose a time synchronization algorithm RFTS based on the ripple flooding [1], which is our previous work. By using a synchronized packet broadcasting of multiple nodes instead of a CSMA-CA MAC based broadcasting, convergence time of the ripple flooding is faster than the simple flooding by a factor of 3.5. In RFTS, a global time of a reference node is flooded to all other nodes unchanged in an interval $(L + D) \cdot H$, where $L$ is the interval for PHY Protocol Data Unit, $D$ is a guard time for preparing next broadcasting and is shorter than an interval for clear channel assessment and random back-off, and $H$ is the number of hops. A reference node puts a global time synchronized to the local time of itself into a sync message and broadcasts it periodically. When nodes in any hops receive the sync message, the nodes save their local time and a modified global time which is a sum of the received global time and $(L + D) \cdot (H - 1)$. And the nodes simultaneously rebroadcast the global time of the reference node after the guard time measured by their own clock. And this flooding does not need any scheduling for broadcasting. Due to the timely consistent flooding scheme, nodes can validate input data by checking if prediction interval [5] with the


DOI: 10.1587/elex.9.691
Received February 27, 2012
Accepted March 21, 2012
Published April 11, 2012

input data and previous saved data is inside of a threshold value. With these features, average synchronization error of RFTS outperforms FTSP by a factor of 2.5.

## 2 Clock model and linear regression

Sensor nodes are equipped with a hardware oscillator-assisted clock, which implements an approximation of real time $t$ as

$$C(t) = \left\lfloor \int_0^t h(\tau)d\tau \right\rfloor + C(t_0) \tag{1}$$

, where $h(\tau)$ is the hardware clock rate at $\tau$ and $C(t_0)$ is the initial offset of the clock. We consider a pair of sensor nodes $x$ and $y$, where $x$ sends sync messages to $y$. Through time synchronization, node $y$ establishes a clock relationship between two nodes and translates $x$'s clock value into its own using the relationship. Node $x$ generates a time stamp $x(t_1)$ at time $t_1$ and sends a sync message including $x(t_1)$ to $y$ and node $y$ saves its local time $Y(t_1)$ and $x(t_1)$ as pair $(x_1, Y_1)$. After receiving $n$ sync messages, node $y$ has an observation set $\{(x_i, Y_i) \mid i = 1, 2, \ldots, n\}$ and infers the coefficients $\beta_0$ and $\beta_1$ in Eq. (2), which are the relative offset and the relative drift between the two nodes, based on linear regression theory [5] as in Eq. (3).

$$Y_i = \beta_0 + \beta_1 x_i + \varepsilon, \text{ where } \varepsilon \sim N(0, \sigma^2) \text{ and } i = 1, 2, \cdots, n \tag{2}$$

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 x. \tag{3}$$

The least-square estimates of the coefficients $\beta_0$ and $\beta_1$ are obtained as

$$\hat{\beta}_0 = \frac{\sum Y_i - \hat{\beta}_1 \sum x_i}{n}, \ \hat{\beta}_1 = \frac{n\sum x_i Y_i - \sum x_i \sum Y_i}{n\sum x_i^2 - (\sum x_i)^2}. \tag{4}$$

We are interested in a future time of node $y$ obtained with a value $x(t^*)$. In order to translate a future time of node $x$ at $x(t^*)$ into node $y$'s time in Eq. (5), node $y$ uses Eq. (3). A $100(1-\alpha)\%$ prediction interval (PI) for a future value of $Y$ when $x(t^*)$ is described in Eq. (6) and means that the future value of $Y$ when $x(t^*)$ will lie within the interval with probability $100(1-\alpha)\%$, where $t_{\alpha/2,n-2}$ is that the number on the measurement axis for which the area under a Student distribution $t_{n-2}$ curve to the right of $t_{\alpha/2,n-2}$ is $\alpha/2$. For example, $t_{0.025,6}$ is 2.447 with a 95% confidence interval and $n = 8$.

$$\hat{Y}(t^*) = \hat{\beta}_0 + \hat{\beta}_1 x(t^*). \tag{5}$$

$$(\hat{Y}(t^*) - w(x(t^*)), \hat{Y}(t^*) + w(x(t^*)))$$
$$, \text{ where } w(x(t^*)) = t_{\alpha/2,n-2}\sqrt{\frac{\sum(Y_i - \hat{Y}_i)^2}{n-2}}\sqrt{1 + \frac{1}{n} + \frac{(x(t^*) - \sum x_i/n)^2}{\sum(x_i - \sum x_j/n)^2}}. \tag{6}$$

## 3 Ripple flooding time synchronization

There is one reference node within WSN and all other nodes with multi-hop synchronize with the reference node using the RFTS. The reference node


DOI: 10.1587/elex.9.691
Received February 27, 2012
Accepted March 21, 2012
Published April 11, 2012

uses the MAC layer time-stamping scheme to distribute periodically its global time in a sync message to neighbor nodes which are located in RF range of the reference node. The level-1 nodes, which are the receivers of the sync message flooded from the reference node, save their local time and the global time to make their observation set and rebroadcast the same message that they received except a slot number [1] after the guard interval $D$ in Fig. 1. The slot number is a 1-byte size field contained in the sync message and the purpose of it is to let the nodes know their level. The level-1 nodes increase the slot number by 1 of the sync message and simultaneously rebroadcast the modified sync message to the next-level nodes. When nodes in any hops receive the sync message, the nodes save their local time and a modified global time which is a sum of the received global time and $(L+D)\cdot(H-1)$, and the nodes simultaneously rebroadcast the global time of the reference node after the guard time. In this manner, the sync messages propagate through the whole network in the interval $(L+D)\cdot H$.

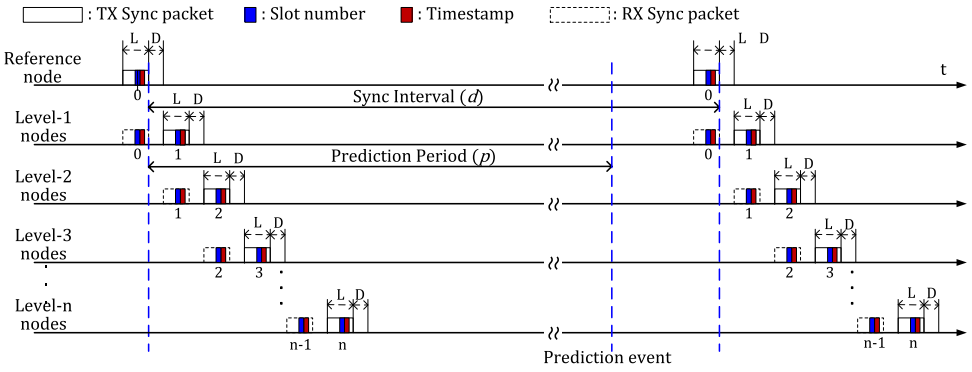


**Fig. 1.** RFTS scheme with a length of sync message $L$ and a guard time $D$

When a node receives a sync message, the node stores a pair of its local time $Y_i$ and the global time $x_{i,my\_level}$ according to its level according to Eq. (7). With $n$ pairs of observation set, the node estimates the coefficients in Eq. (3) and predict a future value of $Y$ for a particular value $x(t^*)$ with a PI $w(x(t^*))$ in Eq. (6).

$$x_{i,my\_level} = x_i + (L+D)\times(my\_level-1). \tag{7}$$

If the reference node floods a sync message every $d$ seconds, which is the sync interval, then the global time can be described with an initial value $x_1$ as follows

$$x_i = x_1 + (i-1)\times d,\ \ i = 1,2,\ldots,n \tag{8}$$

, where $n$ is the size of regression table. As we consider the RFTS in Fig. 1, the global time of a node in each level can be rewritten using the Eq. (7) as follows

$$\begin{aligned} x_{i,my\_level} &= x_1 + (L+D)\times(my\_level-1) + (i-1)\times d \\ &= x_{1,my\_level} + (i-1)\times d \end{aligned}. \tag{9}$$


DOI: 10.1587/elex.9.691
Received February 27, 2012
Accepted March 21, 2012
Published April 11, 2012

If we define the prediction period $p$ as the time difference between the time when the latest observation set is obtained and the time when the prediction event occurs, $p = x(t^*) - x_n$, then the PI $w(x(t^*))$ is described as shown in Eq. (10) [4]. While the PI of level-1 nodes only follows Eq. (10) in FTSP, the PI of all nodes in RFTS follows Eq. (10) and depends on the ratio of $p/d$ and the error sum of squares (SSE) of the node, i.e. $\sum(Y_i - \hat{Y}_i)^2$.

$$w(x(t^*)) = t_{\alpha/2,n-2}\sqrt{\frac{\sum(Y_i - \hat{Y}_i)^2}{n-2}}\sqrt{1 + \frac{1}{n} + \frac{3(n-1+2p/d)^2}{(n-1)n(n+1)}}. \quad (10)$$

If we decide $p/d$ and $n$ in the Eq. (10) as 3/5 and 8 respectively, the PI only depends on the SSE. To figure out a variation of the PI, we set up a 2-node simulation model for time synchronization and put a uniform random error $[-3, 3]$ in clock ticks for each observation set, where the amount of random error was determined by the result of the previous work. The average and the standard deviation of the PI in the simulation are 6.75 and 2.15 in clock ticks. When reception and transmission of the sync message include any errors exceeding the random error, it causes the SSE to exceed a threshold value in RFTS. The SSE can be calculated after nodes receive $n$ sync messages. When a node receives a new sync message, it deletes the oldest observation set and save a newly received one. Before saving a newly received one, the node validate it by checking if the SSE including the newly received one exceeds the threshold value, which is 113 in clock ticks and is set by a sum of the average and the standard deviation of the SSE in the simulation. If the result exceeds the threshold value, the node copies the latest observation set into the newly received one in the regression table. We call it sanity check. By adopting the sanity check, average synchronization error and standard deviation is improved by 15% and 35% under one inconsistent observation set occurring per hour.

## 4 Performance evaluation

We present experimental results for RFTS to verify that the synchronization error does not increase with the size of network and is confined due to the fastest distribution of a global time and adoption of the sanity check. To this end, we configure five sensor nodes on a line topology and the sensor nodes are equipped with the MSP430 microcontroller and the CC2520 transceiver. We use an 8.0 MHz quartz oscillator as the system clock source. The TI CC2520 is 2.4 GHz IEEE 802.15.4 compliant RF transceiver with 250 kbps data rate. For comparison, we implement the FTSP [3] and change distribution scheme of a global time to minimize an effect of slow convergence time on the result; When a node receives a time stamp, it immediately rebroadcasts the time stamp estimated by its own clock with the latest 2 pairs of observation set. The scenario of the experiment is as follows: The reference node broadcasts a sync message to nodes every 30 s. After sending 8 sync messages, the reference node sends a special command to require nodes' estimation of the global time every 18 s. After receiving the command, every node reports to the reference


DOI: 10.1587/elex.9.691
Received February 27, 2012
Accepted March 21, 2012
Published April 11, 2012

node the global time they have estimated. We collect measurement results during 2 hours.

With the ripple flooding scheme for time synchronization and adoption of the sanity check, RFTS shows better performance than FTSP in terms of average and max pairwise error by a factor of 2.5 and there are no spikes shown in Fig. 2. Average sync error of nodes along hops remains under 0.39 $\mu$s in RFTS, and standard deviation of sync error in any hops is under 0.28 $\mu$s shown in Fig. 3. Due to change of distribution scheme in FTSP, average error for single hop is improved from 1.48 $\mu$s in the paper [2] to 0.63 $\mu$s in Fig. 3 by a factor of 2. However, average sync error in FTSP is from 0.63 $\mu$s to 1.21 $\mu$s and standard deviation of sync error is over 0.5 $\mu$s in Fig. 3. While convergence time and average distribution time of time information in 4 hops are 4 minutes and 11.5 ms in CSMA-CA based FTSP, both value are 5.1 ms in RFTS.

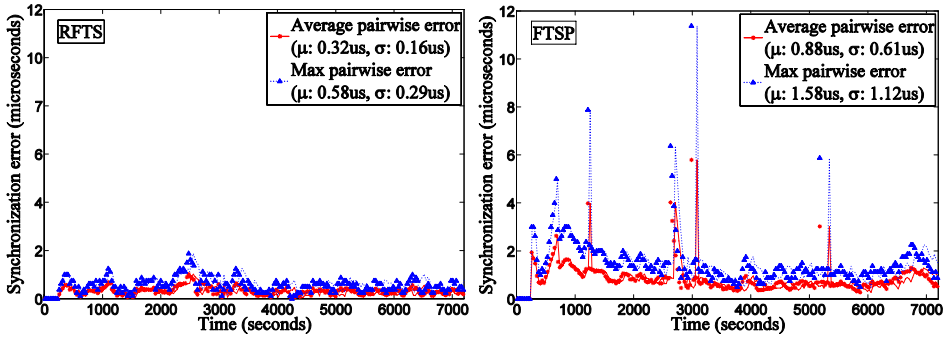


**Fig. 2.** Comparison for average and maximum synchronization error

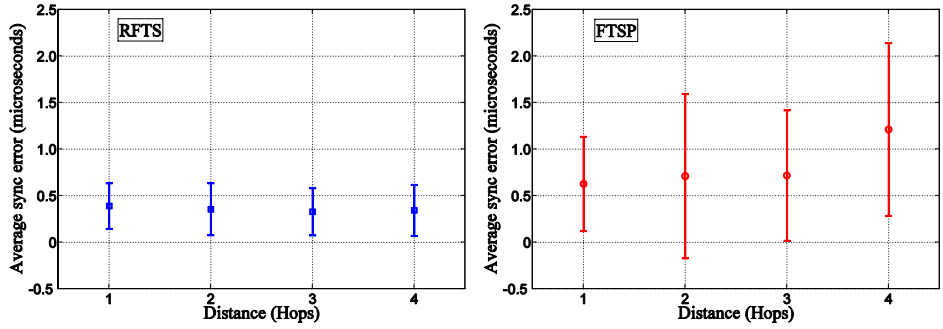


**Fig. 3.** Average sync error versus distance from the reference

## 5 Conclusion

In this letter, we propose a time synchronization RFTS, which presents the fastest distribution of time information of a reference node to all other nodes by using simultaneous packet broadcasting and adoption of the sanity check. Our evaluation shows that RFTS outperforms the improved FTSP by a factor


DOI: 10.1587/elex.9.691
Received February 27, 2012
Accepted March 21, 2012
Published April 11, 2012

of 2.5. Average sync error of nodes in any hops is under 0.39 $\mu$s and standard deviation of sync error also is under 0.28 $\mu$s. Due to the ripple flooding scheme for time synchronization, we can distribute the time information without an elaborate scheduling for nodes and we can perform the sanity check to improve accuracy.

## Acknowledgments

This work has been supported in part by National GNSS Research Center program of Defense Acquisition Program Administration and Agency for Defense Development. This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2011-0020408). This work was supported in part by the Smart IT Convergence System Research Center funded by the Ministry of Education, Science and Technology as Global Frontier Project (2011-0031860).


DOI: 10.1587/elex.9.691
Received February 27, 2012
Accepted March 21, 2012
Published April 11, 2012