Temperature-Assisted Clock Synchronization and Self-Calibration for Sensor Networks

Zhe Yang, *Member, IEEE*, Liang He, *Member, IEEE*, Lin Cai, *Senior Member, IEEE*, and Jianping Pan, *Senior Member, IEEE*

***Abstract*—Synchronization is a pre-requisite for many sensor network applications. However, it remains challenging in sensor networks due to both the limited resources and the dynamic environments. In this paper, we propose a new two-phase clock synchronization scheme. The first one is the external clock synchronization phase, during which nodes update their clock by exchanging timestamp messages with the reference clock. Different from the conventional solutions, we propose to directly remove the clock skew during the external synchronization to achieve a higher synchronization accuracy and lower computational complexity. The second one is the clock self-calibration phase, as the accumulated clock skew will make the synchronized clock drift away again, we need to compensate the clock skew to maintain the clock synchronization accuracy. However, the compensation is non-trivial as the clock skew may not be constant due to the changing environment. Thus we propose the temperature-assisted clock self-calibration (TACSC) to dynamically compensate the clock skew according to the working temperature. Extensive simulation demonstrates that the proposed synchronization scheme can achieve a much lower root mean square error in the external synchronization phase. Furthermore, during the clock self-calibration phase, the TACSC scheme can improve the synchronization accuracy by more than one order of magnitude, which is verified by both simulation and testbed experimentation.**

***Index Terms*—Time synchronization, clock skew, wireless sensor networks.**

I. Introduction

HAVING a synchronized clock, i.e., all nodes having a common notion of *time*, is one of the most fundamental requirements in wireless sensor networks [1]–[3]. Clock synchronization is critical for data transmission, localization, sleep scheduling and information fusion, etc. However, due to the imperfect crystal cutting and clock circuit, the output value of a node's local clock, namely the *local time*, is often different from others' local time, leading to the clock offset. An unregulated clock offset will degrade the network performance and even endanger the proper network functioning.

Clock synchronization, the process of removing the clock offset w.r.t. the reference clock, still remains a challenging issue [4]–[8]. First, sensor nodes usually have limited computational capacity and power supply and thus the synchronization algorithms for sensor networks should be with moderate computational complexity and communication overhead. Besides the limited resources, in many sensor network applications such as environment monitoring, wild animal tracking, surveillance systems, etc., the environments surrounding sensor nodes are usually highly dynamic and/or even hostile. Thus the environment information must be considered to improve the synchronization of sensor nodes.

In modern electronic systems, a crystal oscillator is usually used to generate the clock ticks, which are periodical signals with certain frequency. However, the output frequency of the clock tick is not always the same as the nominal frequency changes according to the working temperature. The deviation of the output frequency results in different clock tick durations and thus the clock skew, which is the instantaneous clock drift rate between two or more clocks. It is the inherent reason for clock de-synchronization [9]. The output value of a clock is the counting number of its ticks and therefore the effect of clock skew will be accumulated and lead to an unbounded clock offset. Thus the clock needs to be re-synchronized to maintain the clock synchronization accuracy. It is obvious that if we can estimate and compensate the clock skew, we can improve the clock synchronization accuracy and therefore prolong the clock re-synchronization period [10]. However, the estimation on the clock skew is non-trivial because the output frequency of a crystal is affected by the working temperature of sensor nodes, which may not be stable [11]. Furthermore, the reference clock may also be temporarily unavailable due to the movement of sensor nodes.

We propose a two-phase clock synchronization scheme in this paper. The first phase is the *temperature-assisted external clock synchronization*, during which the local node exchanges timestamps to eliminate the clock offset. Different from the previous solutions which try to estimate both clock offset and clock skew simultaneously, such as using the maximum likelihood estimators [9], [12], we propose to leverage the relationship between the clock skew and the temperature to estimate the current clock skew, which allows us to directly remove the clock skew and thus improves the clock offset estimation accuracy and reduces the computational complexity. During the second phase, there is no timestamp exchanged and the time is updated by the local clock only. Therefore, we propose the *temperature-assisted clock self-calibration* (TACSC), with which the clock skew is dynamically compensated according to the current working temperature.

Manuscript received February 9, 2013; revised July 16 and November 30, 2013; accepted January 31, 2014. The associate editor coordinating the review of this paper and approving it for publication was L Liu.

Z. Yang is with Northwestern Polytechnical University, Xi'an, Shaanxi, China (e-mail: zyang@ece.uvic.ca).

L. Cai and J. Pan are with the University of Victoria, Victoria, BC, Canada (e-mail: cai@ece.uvic.ca; pan@uvic.ca).

L. He, corresponding author, is with Singapore University of Technology and Design, Singapore (e-mail: he_liang@sutd.edu.sg).

Digital Object Identifier 10.1109/TWC.2014.051414.130270

The main contributions of this paper are threefold. First, we propose to directly remove the clock skew during a clock synchronization process by exploring the relationship between the crystal frequency and the working temperature. It can not only improve the estimation accuracy but also reduce the computational complexity. Second, during the period when the local clock cannot obtain the timestamps from the reference clock, we propose the TACSC scheme to dynamically compensate the clock skew. Thus we can maintain the synchronization of the local clock and therefore significantly prolong the period between two synchronization processes under certain synchronization error tolerance. The proposed TACSC scheme relies on local information only and does not require any timestamp exchange, and thus the communication overhead in clock synchronization is reduced. Third, extensive simulation and testbed verification are conducted to demonstrate the efficacy of the proposed scheme. For the synchronization phase, the proposed scheme can reduce the mean square error (MSE) of clock offset estimation by more than 50%. For the duration when there is no timestamp message exchanged, the proposed TACSC scheme can reduce the synchronization error by more than one order of magnitude.

The rest of the paper is organized as follows. Section II introduces the related work. The important notations and system model are summarized in Section III. The clock skew estimation and the temperature-assisted clock synchronization schemes are presented in Section IV. Performance evaluations by simulation and experiment are given in Section V, followed by concluding remarks and further research issues in Section VI.

II. Related Work

In the previous research work, many clock synchronization approaches have been proposed for different scenarios. For example, the Network Time Protocol (NTP) [13] has been widely used in the Internet, which allows computers to extract the timestamp information from NTP packets to update their local clocks. However, it is not well suited for sensor networks due to the complexity. The Timing-sync Protocol for Sensor Networks (TPSN) was proposed for sensor networks instead in [14] to correct the clock offset. However, this scheme requires frequent re-synchronizations because it does not compensate clock skew, which makes the clock drift away quickly. The Flooding Time Synchronization Protocol (FTSP) [10] has been proposed for multi-hop wireless networks. It calculates the clock skew in a moving window and uses linear-regression algorithms to mitigate both clock offset and clock skew in a hierarchical way. FTSP takes the clock skew into account. Therefore, it can achieve a relatively high synchronization accuracy. In [15], an interesting scheme called FLIGHT was proposed, which can explore the light intensity changes with a stable period that equals half of the alternating current to perform clock synchronization in the indoor environment.

With the estimation of clock skew, it can be removed and/or compensated to improve the synchronization accuracy and/or prolong the re-synchronization period. A linear optimization problem was formulated and solved to estimate clock skew in [16]. A convex hull algorithm for both offline and online clock skew estimation was proposed in [17], and it was also used for clock skew mitigation. In [12], the joint estimation of clock offset and clock skew with unknown synchronization delay was addressed. In [18], a direct clock skew removal technique was proposed. However, the relationship between clock skew and working environment has not been considered in their work, and the skew estimation using a moving window might not be able to track the change of clock skew accurately. In [11], the authors proposed an interesting on-demand synchronization scheme that can give an uncertainty analysis of the clock skew estimation and compensation. According to this uncertainty analysis, the synchronization error can be statistically estimated and therefore the resynchronization can be triggered wisely. However, since this scheme ignored the impact of temperature, the analytical model can only work well in a stable environment. As demonstrated in [19], [20], clock skew changes dramatically when the working temperature changes.

In general, crystal oscillators are sensitive to the temperature and the changing temperature will deviate their output frequency. Although the temperature compensated crystal oscillator (TCXO) can compensate the temperature effect, e.g., MAXIM DS32KHz, the cost is still too high ($2.81 each at the volume of 1K [21]) for the low-cost sensor nodes. Besides, the TCXO also increases the circuit complexity, energy consumption and footprint of wireless devices. In addition, the TCXO cannot totally eliminate the clock skew instability w.r.t. temperature (e.g., up to $\pm 7.5\mathrm{ppm}$ [21]). Thus, we are motivated to devise low-cost, flexible software-based solutions.

In [22], a temperature-compensated time synchronization algorithm was proposed to compensate the clock skew according to the environment temperature. Since it dynamically compensates the clock skew, it can achieve a longer re-synchronization period. The limitation of that work is that the adopted clock skew estimation algorithm treats the clock skew as a stationary random process, which may not be true in dynamic environments.

Different from the previous approaches, in this work, we first explore the relationship between clock skew and dynamic temperature, and propose to estimate the clock skew with the assistance of temperature information. According to the estimation of clock skew, we directly remove it in the external clock synchronization phase, which results in both higher synchronization accuracy and lower computational complexity for clock offset estimation. In the clock self-calibration phase, we use the correlation between temperature and clock skew to compensate the instantaneous clock skew, so it can maintain the clock synchronization level for a much longer duration without exchanging timestamp messages. The proposed solution can be easily extended to any other wireless devices equipped with temperature sensors.

III. Temperature-assisted Clock Skew Estimation

A. Notations

For presentation clarity, we first define the important terms used in this paper, which are consistent with our previous work [19].

Clock is the device to measure time. It consists of a periodic component that ticks at a given frequency and a counter that

counts the number of ticks from a starting time instant. Denote by $C_A(t)$ the output value of clock A at a given reference time instant t.

Clock frequency is the frequency of clock signal generated by the clock circuit, which is usually determined by the crystal and the corresponding peripheral circuit components. Let $f(T)$ denote the actual output frequency of clock circuit at temperature T. It may not be the same as the nominal frequency f_n with perfect crystal and peripheral circuit at the targeted temperature. The difference of $f(T)$ and f_n results in the clock skew, which will be discussed soon.

Clock offset is the difference between the reported time of two or more clocks. If the time reported by clock A and clock B are $C_A(t)$ and $C_B(t)$, respectively, the offset of clock A w.r.t. clock B can be written as

$$\theta_B^A(t) = C_A(t) - C_B(t). \tag{1}$$

Clock skew is the differential coefficient of the clock offset. The clock skew reflects the difference of tick durations of two or more clocks, and is denoted by

$$\alpha_B^A(t) = \frac{d\theta_B^A(t)}{dt} \approx \frac{\theta_B^A(t+\tau(t)) - \theta_B^A(t)}{\tau(t)}, \tag{2}$$

where $\tau(t)$ is the sampling interval.

Using a uniform sampling where $\tau(t) = \tau$, the clock offset and clock skew can be discretized as

$$\theta_B^A[n] = C_A[n] - C_B[n], \tag{3}$$

and

$$\alpha_B^A[n] = \frac{\theta_B^A[n+1] - \theta_B^A[n]}{\tau}. \tag{4}$$

B. Crystal Oscillator

With its low cost, relatively good stability and accuracy, crystal oscillator is widely used in modern electronic systems, especially in embedded systems, e.g., the Mica2 sensor nodes [23] are equipped with the 32KHz crystal SE2412CT-ND [24], a tuning fork crystal unit. The output frequency of the clock circuit depends on the crystal, i.e., the shape or the cut of the crystal, and the peripheral devices, such as the capacitor [20]. However, due to the imperfect cutting techniques and the load capacitance of the peripheral circuits, the actual output frequency $f(T)$ may not be the same as the nominal frequency f_n even at the targeted temperatur T [20] (typically between 20°C$\sim$ 25°C [24]).

Besides, the output frequency is also highly related to the working temperature and is usually modeled as a parabolic function w.r.t. temperature as

$$f(T) = f(T_0)(1 - \beta(T - T_0)^2), \tag{5}$$

where β is the parabolic coefficient (or temperature coefficient) and T_0 is usually called the turn-over temperature (might be different from the targeted temperature). This model has been verified by measurement results and is widely used in both academia and industry [20], [24]–[30]. A common parabolic coefficient for a regular 32KHz tuning fork crystal is around 0.035±$0.01ppm/$°C [24], [25]. It indicates that the crystal will resonate close to $f(T_0)$ when the temperature is close to T_0, and it will slow down when either the temperature increases or decreases from the turn-over temperature. As $f(T)$ over T is a quadratic function, it is sensitive to the operating temperature, e.g., if $T - T_0 = 10$°C, the frequency drift can be up to 4ppm (around 4μs per second). Specifically, our empirical measurement results indicate the values of β for our testbed sensors fall in the range of 0.034±$0.001ppm/$°C, as shown in Fig. 1.

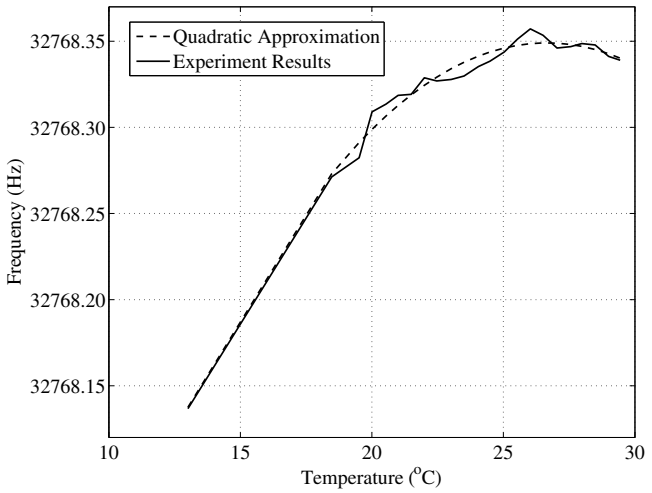


Fig. 1. Frequency vs. temperature quadratic curve fitting.

In the measurement, we let the Mica2 sensor node, which is connected to a laptop (DELL Latitude E5400) via a serial cable, send the timestamp message along with the temperature measurement (from the temperature sensor) to the laptop periodically, e.g., every 10 seconds. The time duration is measured by the sensor node and is represented by t_{sensor}. At the laptop side, whenever it receives the timestamp, a system level interrupt will be triggered and the time will be recorded. The duration between the two timestamp messages, which is measured by the laptop and is represented as t_{laptop}, may not be the same as t_{sensor} due to the clock skew between the sensor node and the laptop. Let Δ_d represent the difference between t_{sensor} and t_{laptop}, which are measured by the laptop and the sensor node, respectively. Hence, we can obtain the clock skew measurements by dividing Δ_d by t_{laptop}.

The clock output is the count of clock ticks, and the duration of a clock tick is determined by the frequency of the crystal oscillator. Therefore, the difference between $f(T)$ and f_n results in a clock skew of

$$\alpha(T) = f_n/f(T) - 1. \tag{6}$$

Given the fact that the temperature sensor becomes the standard element of many sensor boards [31], we can substitute the measured temperature (may contain noise) into (5) to estimate the current clock frequency and to obtain the corresponding clock skew by (6). Since the measurement of temperature is noisy, it will result in the estimation noise in the frequency and clock skew estimation, which will be discussed in Section III-C.

C. Clock Skew Estimation

As mentioned in Section III-B, we can estimate the crystal output frequency based on (5) and the current working temperature. Then the clock skew can be estimated by (6). Since the temperature information from the temperature sensor also contains noise, it will introduce errors in the crystal frequency estimation and thus the clock skew estimation. Similar to the majority of existing research efforts [32], [33], we assume that the temperature measurement noise δ_T is 0 mean with the variance of σ_T^2. We analyze the estimation error performance first and demonstrate that the estimation accuracy is related to the current working temperature.

As presented in Section III-B, the output frequency of a crystal can be captured by a quadratic function w.r.t. temperature. Therefore, an intuitive solution is to substitute the measured temperature into (5) to obtain the frequency estimation. However, this estimation is biased.

Lemma 1: The frequency estimation $\hat{f}(T)$ obtained by substituting the measured temperature into (5) is not an unbiased estimation of the actual frequency $f(T)$ at temperature T.

Proof: The measured temperature $\tilde{T}$ is expressed as

$$\tilde{T} = T + \delta_T, \tag{7}$$

where T denotes the actual temperature and δ_T is the measurement noise as mentioned before. Therefore, the estimation of frequency directly from (5) is

$$\tilde{f}(T) = f_0(1 - \beta(\tilde{T} - T_0)^2). \tag{8}$$

Then, we obtain the mean value of the frequency estimation at temperature T as

$$\begin{aligned} E[\tilde{f}(T)] &= E[f_0(1 - \beta(\tilde{T} - T_0)^2)] \\ &= f_0 - f_0\beta E[((T + \delta_T - T_0)^2)] \\ &= f_0 - f_0\beta E[(T - T_0)^2 + \delta_T^2 + 2\delta_T(T - t_0)] \\ &= f_0(1 - \beta(T - T_0)^2) - f_0\beta\sigma_T^2. \end{aligned} \tag{9}$$

As we know, the actual crystal output frequency at T should be $f_0(1-\beta(T-T_0)^2)$ [20], [24], [25], which is different from $E[\tilde{f}(T)]$ as we calculated above. This finishes the proof. ■

From the proof of Lemma 1, we can see that the difference between $E[\tilde{f}(T)]$ and $f(T)$ is $-f_0\beta\sigma_T^2$, which can be calibrated prior to the deployment stage. Therefore, we can take it into consideration and obtain the unbiased crystal output frequency estimation.

Theorem 1: The unbiased estimation of crystal output frequency based on the temperature measurement is

$$\tilde{f} = f_0(1 - \beta(\tilde{T} - T_0)^2) + f_0\beta\sigma_T^2, \tag{10}$$

and the mean value of the estimated frequency is $E[\tilde{f}(T)] = f(T)$.

The proof of Theorem 1 is similar to that of Lemma 1 and therefore is omitted here due to the space limit.

The estimation error of the crystal output frequency at certain temperature is also a random variable and we can calculate the corresponding variance as

$$\begin{aligned} \sigma_{f(T)}^2 &= E[(\tilde{f} - \bar{f})^2] \\ &= E[(f_0(1 - \beta(\tilde{T} - T_0)^2) - f_0(1 - \beta(T - T_0)^2))^2] \\ &= f_0^2\beta^2(3\sigma_T^4 + 4(T - T_0)^2\sigma_T^2). \end{aligned} \tag{11}$$

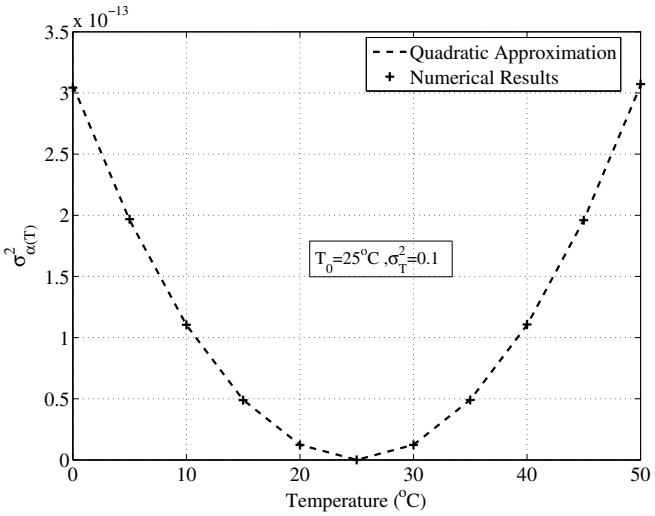


Fig. 2. Variance of clock skew estimation error.

From (11), the variance of the frequency estimation error is also related to temperature, and the further away from the turn-over temperature T_0, the larger the variance is.

From (6), we can obtain the current clock skew based on the estimation of the crystal output frequency

$$\alpha(T) = \frac{f_n}{\tilde{f}(T)} - 1. \tag{12}$$

Unfortunately, we cannot obtain the closed-form expression of the mean value and variance of the estimation error of the clock skew. The clock skew estimation obtained from (12) can be biased. However, from the numerical results, the mean value of the clock skew estimation error is below 10^{-10} level (below 10^{-9} from experiments), which is negligible. For the variance of the clock skew estimation error, we find that it can be well captured by a quadratic function w.r.t. temperature by numerical results and experiments, as shown in Fig. 2 and Fig. 3, respectively. It has the same trend as the frequency estimation error, i.e., the further away from the turn-over temperature T_0, the larger the estimation error variance. This is because the parabolic parameter errors and the temperature measurement noise have larger impact on the frequency estimation and therefore on the clock skew estimation. We will evaluate its impact on the synchronization performance through simulation in Section V.

Discussions: The clock skew changes in dynamic environments. To address this challenge, we advocate to estimate clock skew in advance based on the relationship between crystal output frequency and the temperature. Besides the temperature measurement error, the errors contained in the parabolic parameter β, the actual turn-over temperature T_0 and the corresponding crystal output frequency $f(T_0)$ also affect the clock skew estimation performance. These parameters can be obtained by pre-deployment system profiling [20] or by the curve fitting from the online clock skew estimation [19]. We will evaluate the impact of the imperfect parameters in Section V.

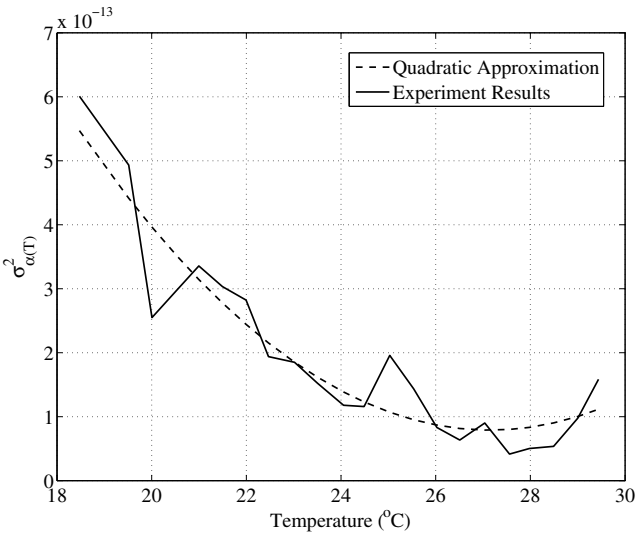


Fig. 3. Variance of clock skew estimation error (experiment results).

IV. Two-phase Temperature-assisted Synchronization and Self-calibration

In this section, we present the design of the two-phase temperature-assisted clock synchronization and self-calibration algorithm. We first estimate the clock skew according to the relationship between the crystal output frequency and the working temperature, as presented in Section III-B. Then it can be directly removed in the external synchronization phase and also dynamically compensated thereafter till the following re-synchronization. Different from the previous work [11], [12], [20], [22], we take the environment information into account.

A. External Clock Synchronization Phase

Clock synchronization is the process to mitigate the clock offset among two or multiple clocks based on the timestamps. During this phase, local nodes exchange timestamp messages with the reference clock as presented in Section IV-A1 and the procedure is shown in Fig. 4.

1) Two-way Timestamp Exchange: The two-way timestamp exchange mechanism between the local node and the reference node is usually adopted to estimate the clock offset [12], [34], as shown in Fig. 4. Several timestamps might be exchanged to achieve a higher clock offset estimation accuracy, in order to eliminate the measurement noise caused by delay, clock jitters, and so on.

In Fig. 4, θ_0 is the initial clock offset to be estimated. The absolute value of clock offset might change as time elapses due to the clock skew, i.e.,

$$\theta(\Delta t) = \theta_0 + \alpha(T)\Delta_t, \tag{13}$$

where the clock skew $\alpha(T)$ can be assumed as a constant during the timestamp exchanges, since the timestamp message exchange can be finished within a very short time. In the ith round of timestamp exchange, node A first sends the timestamp $t_{A,1}^{(i)}$ at $t_{A,1}^{(i)} + d_{p,A}$, where $d_{p,A}$ is the processing and queuing delays at node A. The message arrives at node B at $t_{B,1}^{(i)} - d_{p,B}$ and node B records the time $t_{B,1}^{(i)}$, where $d_{p,B}$

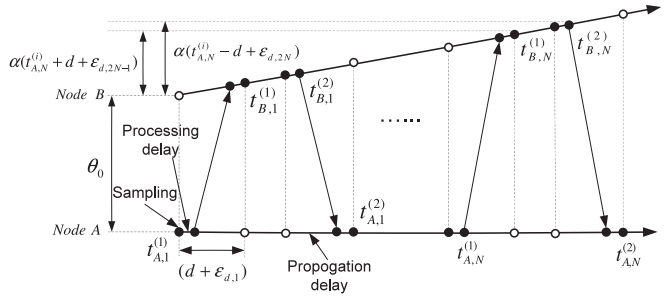


Fig. 4. Two-way timestamp exchange.

is the processing and queuing delays at node B. The reply message contains timestamp $t_{B,2}^{(i)}$ and is sent at $t_{B,2}^{(i)} + d_{p,B}$ along with the previous recorded timestamp $t_{B,1}^{i}$. Then node A records the reception time of the reply message as $t_{A,2}^{i}$.

Denote $D_1[i]$ as the difference between the timestamps $t_{A,1}^{i}$ and $t_{B,1}^{i}$, i.e., $D_1[i] = t_{B,1}^{i} - t_{A,1}^{i}$. However, we cannot simply use $D_1[i]$ as the estimation or measurement of clock offset $\theta[i]$, since there are several delays introduced in different stages during the message transmission, which make it deviate from the actual clock offset. Instead, the clock offset can be expressed as

$$D_1[i] = \theta_1[i] + (1 + \alpha(T))(t_c[i] + t_t[i] + t_p[i]), \tag{14}$$

where $\theta_1(i) = \theta_0 + \alpha(T)(t_{A,1}^{i} - t_{A,1}^{0})$ denotes the current clock offset; $t_c[i]$ and $t_p[i]$ are the message construction delay at the sender side and the processing delay at the receiver side, respectively; $t_t[i]$ is the propagation delay. $t_c[i]$ and $t_p[i]$ may be different from system to system due to the CPU capacity and the system load. Their variations are usually at μs level or even smaller, by setting the corresponding processes at a high priority. Hence, we can approximate them as constants and their variations can be included in the system observation noise. $t_t[i]$ can also be approximated as a constant if the channel is stable, and thus its variance can be included in the system observation noise as well. Therefore, we can attribute all these delays as a constant delay d_1 plus a zero-mean random variable and rewrite $D_1[i]$ as

$$D_1[i] = \theta_1[i] + (1 + \alpha(T))(d_1 + \omega_{d,1}[i]), \tag{15}$$

where $\omega_{d,1}[i]$ is the delay jitter during the transmission from node A to node B. Similary, let $D_2[i]$ denote the difference between timestamps $t_{B,2}^{i}$ and $t_{A,2}^{i}$, i.e., $D_2[i] = t_{B,2}^{i} - t_{A,2}^{i}$, and it can be written as

$$D_2[i] = \theta_2[i] - (1 + \alpha(T))(d_2 + \omega_{d,2}[i]), \tag{16}$$

where $\omega_{d,2}[i]$ is the delay jitter during the transmission from node B to node A. Since the aforementioned process happens within a short period, we assume that the delay is symmetric, i.e., $d_1 = d_2 = d$ as in [9], [12]. Therefore, we can model this two-way timestamp exchange as

$$\begin{aligned} D_1[i] &= t_{B,1}^{i} - t_{A,1}^{i} \\ &= \theta_0 + \alpha(T)(t_{A,1}^{i} - t_{A,1}^{1} + (1 + \alpha(T))(d + \omega_{d,1}[i]), \\ D_2[i] &= t_{B,2}^{i} - t_{A,2}^{i} \\ &= \theta_0 + \alpha(T)(t_{A,2}^{i} - t_{A,1}^{1}) - (1 + \alpha(T))(d + \omega_{d,2}[i]), \end{aligned} \tag{17}$$

where $\omega_{d,1}[i]$ and $\omega_{d,2}[i]$ are assumed to be zero mean independent Gaussian distributed random variables with variance σ_d^2, which has been experimentally verified in [35]. If the two-way timestamp exchange is conducted for N rounds, we can obtain a sequence of recorded timestamps as $\{t_{A,i}^1, t_{B,i}^1, t_{A,i}^2, t_{B,i}^2\}_{i=1}^N$. Then, the clock offset θ_0 can be estimated based on these N equation sets similar to (17) by different algorithms [9], [12].

2) Temperature-assisted Clock Synchronization: Based on the two-way timestamp exchange, we can obtain a sequence of equation sets as (17). Because the timestamp exchange can be accomplished within a short period, similar to existing research efforts [9], [12], we simplify the investigation by assuming the two-way transmission between the local node and the reference node undergoes the same delay with independent delay jitters. We can rewrite the two-way timestamp exchange model (17) by adding the two equations together, and the new model is

$$\begin{aligned} D_1[i] + D_2[i] &= 2\theta_0 + \alpha(T)(t_{A,1}^i + t_{A,2}^i - 2t_{A,1}^1) \\ &\quad +(1+\alpha(T))(\omega_{d,1}[i] - \omega_{d,2}[i]). \quad (18) \end{aligned}$$

Different from the previous work which tries to estimate both clock offset θ_0 and clock skew α simultaneously, such as using maximum likelihood estimators [9], [12], we leverage the temperature information to estimate clock skew, as presented in Section III-C. Then we directly remove the clock skew accordingly to improve the clock offset estimation accuracy. On the other hand, since we only need to estimate clock offset instead of two unknown variables, we can reduce the computational complexity of the estimator. Let $\tilde{\alpha}(T)$ denote the estimated clock skew, we can obtain the ith clock offset estimation $\tilde{\theta}_0[i]$ from (18) as

$$\tilde{\theta}_0[i] = \frac{1}{2}[D_1[i] + D_2[i] - \tilde{\alpha}(T)(t_{A,1}^i + t_{A,2}^i - 2t_{A,1}^1)]. \quad (19)$$

As we mentioned before, the two-way timestamp exchange is conducted multiple times sequentially to smooth the noise out. Therefore, the estimation of clock offset θ_0 is

$$\tilde{\theta}_0 = \frac{1}{2N}\sum_{i=1}^{N} \tilde{\theta}_0[i], \quad (20)$$

where N is the number of the two-way timestamp. This estimator only involves scalar operations. From (18) and (20), we can see that there are $9N$ add operations and N multiply operations involved, which is much simpler than the previous work [9], [12], where there are at least $14N$ add operations and $7N$ multiply operations.

B. Clock Self-calibration Phase

As the timestamp messages may not be always available due to the communication overhead or when the reference clock is temporarily unreachable, especially for WSN applications. Therefore, the sensor node needs to maintain the synchronization by its local clock. Unfortunately, as we discussed before, the uncompensated, time-varying clock skew will make the clock offset drift away quickly and unbounded.

An effective compensation of clock skew can improve the clock synchronization accuracy and prolong the clock re-synchronization period and thus less-frequent timestamp exchange is needed, which is particularly desirable for WSNs where communication cost is high and frequent re-synchronization may not be feasible due to mobility, sleep scheduling, etc. However, as demonstrated in previous work [19], [20], [22] and in the crystal datasheet [24], [25], the clock skew is not constant and varies with the surrounding environment. Therefore, the compensation of clock skew for WSNs is non-trivial, so we propose the temperatur-assisted clock self-calibration (TACSC) scheme which takes the temperature information into account to dynamically compensate clock skew.

During the self-calibration phase, a local variable $\tilde{\theta}[n]$ is used to indicate the estimated clock offset in the n-th time slot, which is the accumulated clock skew since the last external synchronization phase. Due to the thermal inertia, temperature usually does not change quickly in a small time period, such as a few seconds. We assume that clock skew does not change during one sampling period. Thus, the instantaneous temperature measurement can represent the temperature for the whole sampling period. We can use the current temperature to obtain the instantaneous clock skew $\tilde{\alpha}(T_n)$. Then we can update the estimated clock offset as

$$\tilde{\theta}[n] = \tilde{\theta}[n-1] + \tau\tilde{\alpha}(T_n) = \tilde{\theta_0} + \tau\sum_{i=1}^{n}\tilde{\alpha}(T_i), \quad (21)$$

where T_i is the temperature measurement in the ith slot and τ is the temperature (clock skew) sampling duration. As proved in [11], (21) is an unbiased estimation of clock offset $\theta_i[n]$. We are also interested in its error uncertainty.

Theorem 2: The variance of the clock offset estimation by (21) at Nth slot is

$$\sigma_{\theta[N]}^2 = \sigma_{\theta_0}^2 + \tau^2\sum_{i=1}^{N}\sigma_{\alpha(T_i)}^2, \quad (22)$$

where $\sigma_{\theta_0}^2$ is the variance of clock offset estimation in the external synchronization phase.

Proof: The true value of clock offset at the Nth time slot is

$$\theta[N] = \theta_0 + \tau\sum_{i=1}^{N}\alpha(T_i). \quad (23)$$

Therefore, from (21) and (23), the clock offset estimation error is

$$\delta_{\theta[N]} = \tilde{\theta}[N] - \theta[N] = \delta_{\theta_0} + \tau\sum_{i=1}^{N}\delta_{\alpha(T_i)}, \quad (24)$$

where δ_{θ_0} is the estimation error in the external clock synchronization phase and $\delta_{\alpha(T_i)}$ is the clock skew estimation error. Therefore, from (24), we have the error variance as

$$\sigma_{\theta[N]}^2 = \sigma_{\theta_0}^2 + \tau^2\sum_{i=1}^{N}\sigma_{\alpha(T_i)}^2.$$

This finishes the proof. ■

From Theorem 2, given the elapsed time of TACSC ($\Delta t = N\tau$), we can see that a smaller sampling period τ leads to a smaller error variance at the cost of a higher energy

consumption. Without loss of generality, we set the sampling period τ as 1s for simplicity in simulation.

Given the estimation of clock offset $\tilde{\theta}[n]$, we can compensate it to the local clock as follows:

$$C_i[n] = C_i[n] - \tilde{\theta}[n]. \tag{25}$$

Given the unavoidable clock offset and clock skew estimation errors, the clock will eventually drift away but at a much slower pace, as we will show in Section V.

V. Performance Study

In this section, we first present the numerical results of the clock offset estimation, i.e., the results of the external clock synchronization phase. We then evaluate the performance of the proposed TACSC scheme by simulation and by experiments using our sensor testbed.

A. Performance of the External Clock Synchronization Phase

The external clock synchronization is based on the two-way timestamp exchange shown in Fig. 4. The transmission delay D and the standard deviation of delay jitter σ_D are set as 100ms and 10μs, respectively. The temperature measurement noise is zero mean Gaussian distributed with the variance $\sigma_T^2 = 0.1$. Considering the tuning folk crystal oscillator used in Mica2 nodes [23], the nominal frequency is 32,768Hz and the parabolic parameter β is set to 0.04ppm. The turn-over temperature and the corresponding output frequency are 25°C and $32,768.5$Hz, respectively.

The root mean square errors (RMSE) of the clock offset estimation based on the two-way timestamp exchange are shown in Fig. 5. We compare the proposed temperature-assisted external clock synchronization scheme with the existing solution [12], which tries to estimate the clock offset and clock skew simultaneously. We also evaluate the Cramer-Rao Lower Bound (CRLB), assuming that there is no clock skew, as a benchmark [9]. As presented in Section III-B, the estimation variance of clock skew is also related to the temperature, which will affect the synchronization accuracy. Therefore, we evaluate the synchronization performance under different temperatures (0°C and 10°C).

From Fig. 5, without the parabolic parameter error, the proposed temperature-assisted synchronization can significantly outperform the existing algorithm and reduce the RMSE by more than 90% and close to the performance bound. This is because the previous solutions estimate the clock offset and clock skew simultaneously, while we only need to estimate the clock offset and therefore the information contained in the measurements is dedicated to one unknown variable. Besides, the computational complexity is reduced. We just show the results at $T = 0$°C and omit the results at $T = 10$°C, which largely overlap with the performance bound.

In reality, in order to obtain the parameters that define the parabolic function of crystal output frequency v.s. temperature, i.e., f_0, T_0 and β, we need clock skew values w.r.t. different temperature values as the training data. We can obtain the parabolic parameters with at least three clock skew v.s. temperature estimation pairs, and the estimation accuracy will be improved with more training data. Then we can use the minimum mean square error (MMSE) algorithm to estimate these parabolic parameters. However, such estimation also contains errors, which will affect the synchronization performance. The standard deviation of the clock skew estimation error can be as low as $10^{-2}\mu$s [11], [19]. Assuming that the standard deviation of the clock skew estimation error is 0.1μs and that of the temperature is 0.1°C, we used 8 clock skew v.s. temperature pairs as the training data set to estimate the parabolic parameters and then use them to dynamically estimate the crystal output frequency and thus the corresponding clock skew. From the results shown in Fig. 6, we can see that the RMSE is lower at $T = 10$°C than that at $T = 0$°C. This is because that the parabolic parameter errors have a larger impact at the temperature further away from the turn-over temperature (25°C). Besides, the variance of the clock skew estimation error is also larger at $T = 0$°C than that at 10°C. Nevertheless, even with the estimation errors in parabolic parameters, the proposed scheme can still significantly outperform the existing algorithm.

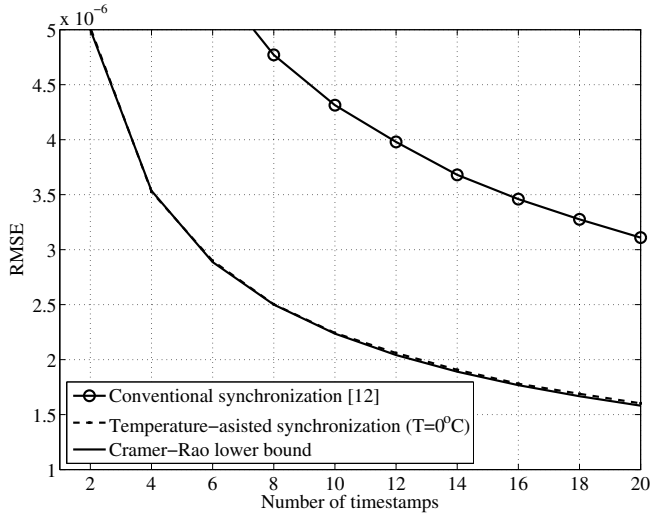


Fig. 5. Clock offset estimation, without parabolic parameter error.

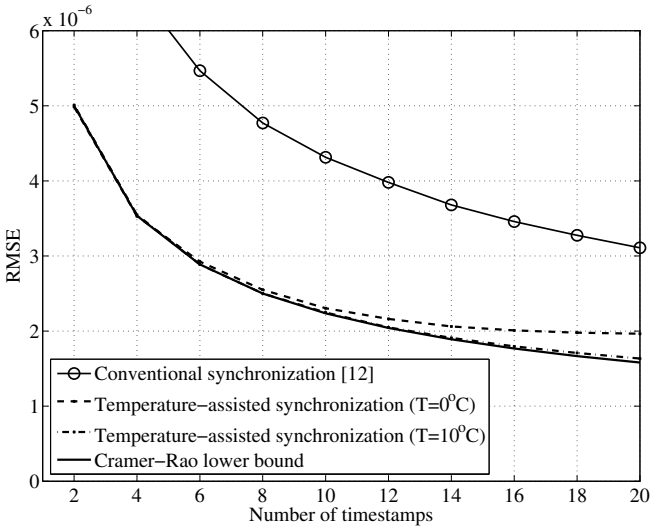


Fig. 6. Clock offset estimation, with parabolic parameter error.

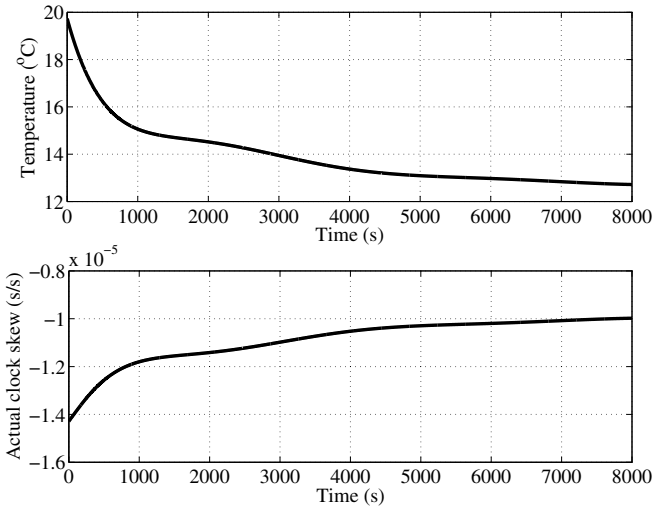


Fig. 7. Simulation data for TACSC.

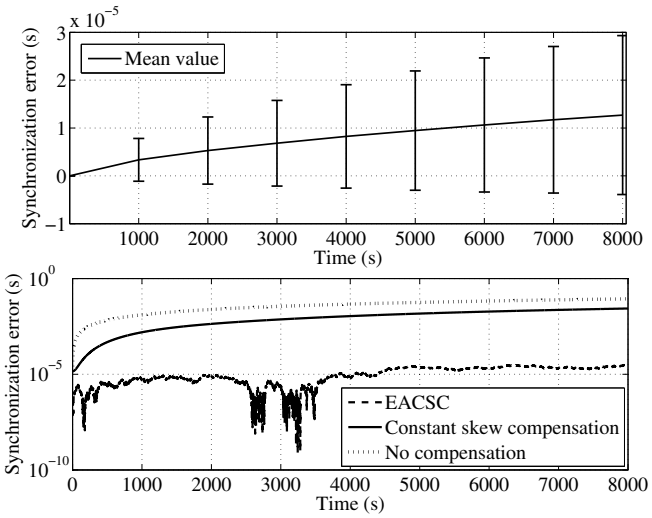


Fig. 8. Temperature-assisted clock self-calibration, without parabolic parameter error.

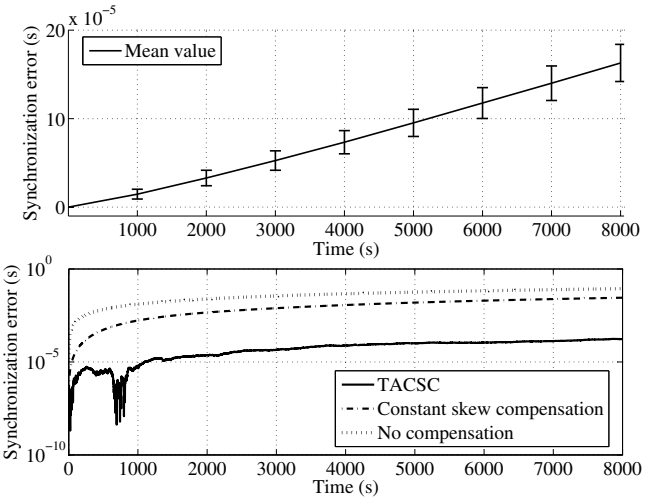


Fig. 9. Temperature-assisted clock self-calibration, with parabolic parameter error.

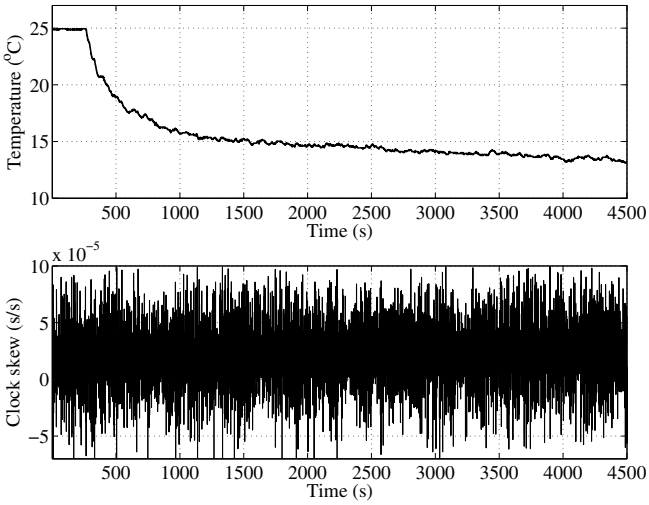


Fig. 10. Training trace.

B. Performance of the Clock Self-calibration Phase

To evaluate the performance of the TACSC scheme and compare it with the constant skew compensation solution, we generated simulation data as shown in Fig. 7, which simulate the temperature changing from the sunset to midnight (summer time, Victoria, BC) and the corresponding clock skew. We set the sampling period T to be 1 s. We set the parabolic parameter of the sensor node, where β is 0.03469ppm, f_0 is $32,767.41$Hz and T_0 is 26.4°C. The variance of the temperature measurement is set as σ_T^2 is 0.01.

The results of TACSC and the constant clock skew compensation are compared in Fig. 8. As shown in the figure, the constant clock skew compensation only works for certain circumstances and the performance degrades severely once the working environment changes because it cannot adaptively compensate the changing clock skew. On the other hand, we can see that the clock offset is well compensated using the proposed TACSC and the clock synchronization error is more than two orders of magnitude lower than that by the constant compensation algorithm.

Different from the external synchronization phase, the TACSC scheme is more sensitive to the errors of parabolic parameters. As demonstrated in Fig. 9, the mean synchronization error of TACSC is up to 160μs, which is much higher than that in Fig. 8 (less than 15μs). This is because the clock skew estimation error caused by the imperfect parabolic parameters will be accumulated over a much longer calibration phase. However, the synchronization error of the proposed TACSC scheme is still much lower than that of the constant clock skew compensation scheme, which is up to 0.115s, and the improvement is about three orders of magnitude.

We also conducted experiments using the Mica2 testbed to demonstrate the applicability and feasibility of the proposed scheme in a real world. The experiment setup is the same as that presented in Section III-B. We used the temperature and clock skew measurements shown in Fig. 10 as the training dataset to obtain the clock skew v.s. temperature and estimate the parabolic parameters by curve fitting. Then, several months later, we used the same sensor and the laptop (DELL Latitude

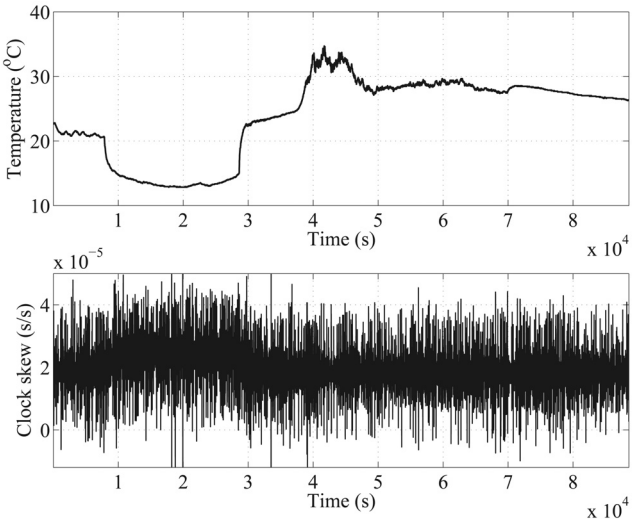


Fig. 11. Verification trace.

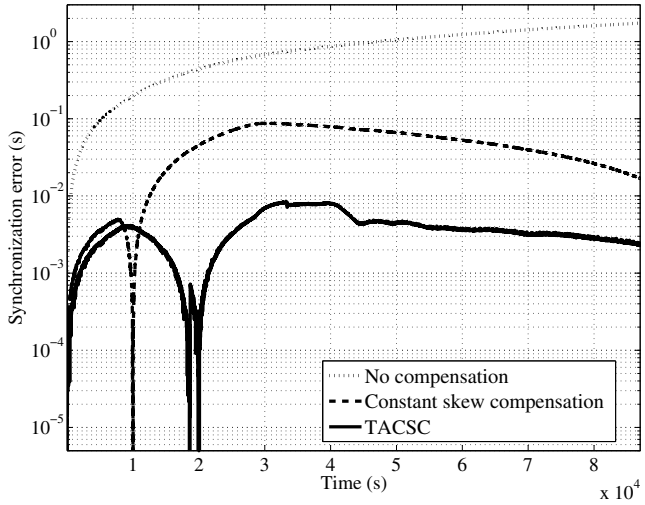


Fig. 12. Verification results.

E5400) to conduct the same measurement, where the sensor was working in the outdoor environment in Victoria, BC, and the traces of temperature and clock skew are shown in Fig. 11. According to the parabolic parameters estimated by the training data, we can estimate the crystal output frequency (clock skew) and then dynamically compensate the clock skew of the sensor node.

The compensation results are shown in Fig. 12. From the figure, the clock offset by TACSC is much lower than those without compensation or with a constant skew compensation. The clock offset by TACSC is always below 8.5ms over the $88,593$s (more than 24h) test duration, which is an order of magnitude improvement to the constant skew compensation, where the clock offset is up to 95ms.

On the other hand, it is not surprising that the experiment results are worse than those of simulations due to the following reasons. First, the temperature measurement noise may not always be Gaussian in practical systems, which will degrade the estimation accuracy. More importantly, we used a laptop in a lab environment as the reference node, as the clock of the laptop is much more stable than that of the sensor node. However, the laptop clock circuit also suffers from thermal noise and other uncertain factors, and the lab room temperature is not constant as our lab has no temperature control, which also introduces errors. Furthermore, the estimation error of parabolic parameters will be accumulated. These imperfect practical conditions degrade the performance of the clock skew estimation in the real test. Even with these impairments, the test results show that TACSC can meet our design objective, i.e., reduce the clock offset and prolong the re-synchronization period by an order of magnitude, and its performance is reliable over several months (and probably longer).

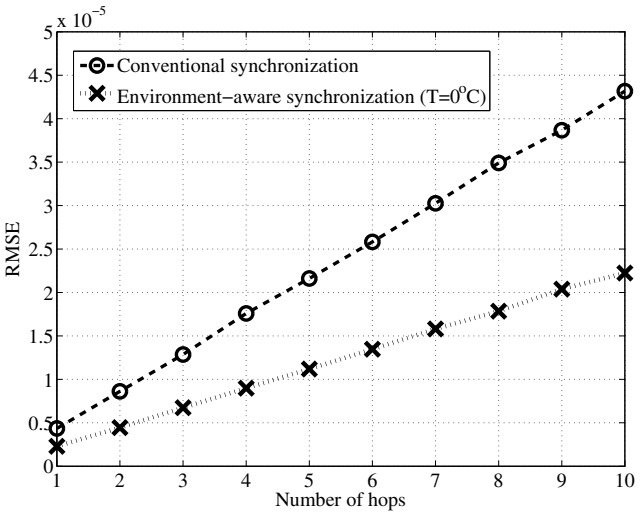


Fig. 13. Clock offset estimation in the multi-hop network.

C. Performance in Multi-hop Networks

We have demonstrated the effectiveness of the two-phase clock synchronization for pair-wise synchronization. As the WSN applications usually involve a large number of sensor nodes which may not be all reached by one-hop communications and thus the multi-hop communication is adopted to let the reference clock information propagate to the entire network [34], [36]. However, the clock synchronization accuracy will degrade as the number of hops increases. Therefore, the scalability issue is a grand challenge in WSN synchronization. To evaluate the performance of the proposed scheme, we conduct a simulation study with a multi-hop network. The simulation setting is the same as that in Sec. V-A except that we consider a multi-hop network with 10 hops. There are $1,000$ nodes in the network. Without loss of generality, we assume each hop contains 100 nodes. We use the MAC layer-time stamp [14], [37] to minimize the delay uncertainties.

The RMSEs of the external clock synchronization in the multi-hop network are shown in Fig. 13. We can see that the RMSEs of both the conventional synchronization and proposed synchronization schemes increase, which indicate the deterioration of the synchronization accuracy, as the number of hops increases because more uncertainties such as delay, clock jitter, etc. are introduced. However, as we can see from this figure, the RMSE of the proposed scheme is not only still much lower than that of the conventional scheme, which is

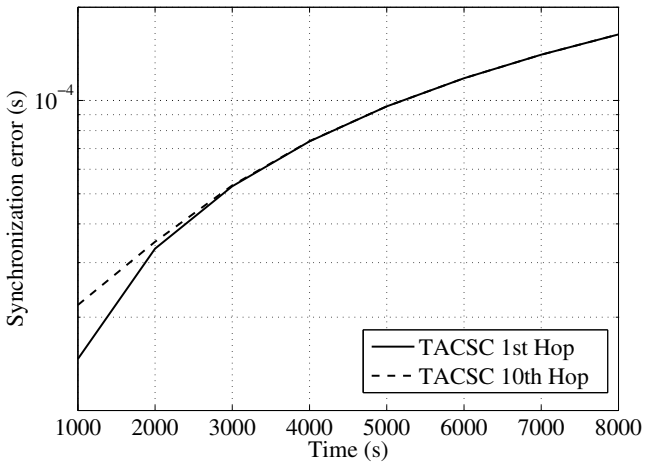


Fig. 14. Clock self-calibration in the multi-hop network.

reduced by around 50% at the 10th hop, but also increases in a slower pace. It means that the proposed scheme can still significantly outperform the existing algorithm in multi-hop networks.

The results of TACSC in multi-hop networks are compared in Fig. 14. We show the results of the 1st hop and the 10th hop and the results from 2nd to 9th hops are omitted because they are not very distinguishable. We can see that the performance only degrades a little bit (less than 10^{-5}s) at the 10th hop compared with that of the first hop, which is not even noticeable. This is because the TACSC is based on the local information and the synchronization error introduced by the clock-self calibration phase is topology independent. Therefore, TACSC is very suitable for large-scale multi-hop WSNs. The performance loss is mainly due to the synchronization error in the external clock synchronization phase, which is not distributed. Even with the performance loss at the 10th hop, synchronization accuracy can still outperform the constant compensation by more than one order of magnitude.

From the above discussion, we can conclude that although the proposed scheme is designed for pair-wise clock synchronization, it is suitable for clock synchronization in large-scale networks and can achieve a decent performance in multi-hop networks.

VI. Conclusions

In this paper, we have investigated the temperature-assisted clock synchronization and clock self-calibration synchronization for wireless sensor networks, which are usually working in the dynamic working environments. The crystal output frequency has been modeled as a quadratic function w.r.t. temperature, so we can estimate it according to the current working temperature and thus the clock skew. We have then proposed to directly remove the clock skew during the external clock synchronization process to improve the synchronization accuracy with a much lower computational cost. For the interval between synchronization actions, we have proposed a TACSC scheme to conduct the real-time dynamic clock skew compensation. Simulation and experiment results have demonstrated the effectiveness and efficiency of the proposed TACSC scheme.

How to further improve the accuracy of the parabolic parameter estimation especially with an online estimation remains an open issue. Meanwhile, more research work about the uncertainty analysis of TACSC scheme under imperfect parabolic parameter estimation is beckoned since it can give the error bound of TACSC and thus the re-synchronization can be triggered appropriately according to the synchronization requirement. Besides the temperature, further research is needed to investigate the impact of other environment factors (such as the power supply voltage, humidity, electromagnetic interference, and vibration) on clock skew, which are treated as noise in this work as their influence is typically less than that of the temperature.

Acknowledgment

This work was supported in part by NSERC Canada and the National Natural Science Foundation of China under project 61272461.

References

[1] Y. Wang, F. Nunez, and F. J. Doyle III, "Energy-efficient pulse-coupled synchronization strategy design for wireless sensor networks through reduced idle listening," *IEEE Trans. Signal Process.*, vol. 60, no. 10, pp. 5293–5306, 2012.

[2] Q. M. Chaudhari, "A simple and robust clock synchronization scheme," *IEEE Trans. Commun.*, vol. 60, no. 2, pp. 328–332, 2012.

[3] Y. Gu, L. He, T. Zhu, and T. He, "Achieving energy-synchronized communication in energy-harvesting wireless sensor networks," *ACM Trans. Embedded Comput. Syst.*, vol. 13, no. 2, pp. 68:1–68:26, 2014.

[4] J. Liu, Z. Zhou, Z. Peng, J. Cui, M. Zuba, and L. Fiondella, "Mobi-sync: efficient time synchronization for mobile underwater sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 2, pp. 406–416, 2013.

[5] C. Liao, "Time-synchronization in mobile sensor networks from difference measurements," in *Proc. 2010 IEEE Conf. Decision Control.*

[6] J. Beauquier and J. Burman, "Self-stabilizing synchronization in mobile sensor networks with covering," in *Distributed Computing in Sensor Systems*, volume 6131 of *Lecture Notes in Computer Science*, R. Rajmohan, M. Thomas, D. Adam, and S. Anna, Eds. Heidelberg: Springer, 2010, pp. 362–378.

[7] K. Cheng, K. Lui, Y. Wu, and V. Tam, "A distributed multihop time synchronization protocol for wireless sensor networks using pairwise broadcast synchronization," *IEEE Trans. Wireless Commun.*, vol. 8, no. 4, pp. 1764–1772, 2009.

[8] K. Noh, E. Serpedin, and K. Qaraqe, "A new approach for time synchronization in wireless sensor networks: pairwise broadcast synchronization," *IEEE Trans. Wireless Commun.*, vol. 7, no. 9, pp. 3318–3322, 2008.

[9] K. L. Noh, Q. M. Chaudhari, E. Serpedin, and B. W. Suter, "Novel clock phase offset and skew estimation using two-way timing message exchanges for wireless sensor networks," *IEEE Trans. Commun.*, vol. 55, no. 4, pp. 766–777, 2007.

[10] M. Maroti, B. Kusy, G. Simon, and Á. Lédeczi, "The flooding time synchronization protocol," in *Proc. 2004 ACM SenSys.*

[11] Z. Zhong, P. Chen, and T. He, "On-demand time synchronization with predictable accuracy," in *Proc. 2011 IEEE INFOCOM.*

[12] M. Leng and Y. C. Wu, "On clock synchronization algorithms for wireless sensor networks under unknown delay," *IEEE Trans. Veh. Technol.*, vol. 59, no. 1, pp. 182–190, 2010.

[13] D. L. Mills, "Internet time synchronization: the network time protocol," *IEEE Trans. Commun.*, vol. 39, no. 10, pp. 1482–1493, 1991.

[14] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *Proc. 2003 ACM SenSys.*

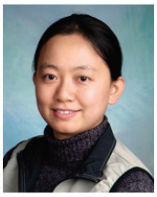
[15] Z. Li, W. Chen, C. Li, M. Li, X. Li, and Y. Liu, "Flight: clock calibration using fluorescent lighting," in *Proc. 2012 ACM MobiCom.*

[16] S. B. Moon, P. Skelly, and D. Towsley, "Estimation and removal of clock skew from network delay measurements," in *Proc. 1999 IEEE INFOCOM.*

[17] L. Zhang, Z. Liu, and C. H. Xia, "Clock synchronization algorithms for network measurements," in *Proc. 2002 IEEE INFOCOM.*
[18] H. Khlifi and J. C. Gregoire, "Estimation and removal of clock skew from delay measures," in *Proc. 2004 IEEE Conf. Local Comput. Netw.*
[19] Z. Yang, L. Cai, Y. Liu, and J. Pan, "Environment-aware clock skew estimation and synchronization for wireless sensor networks," in *Proc. 2012 IEEE INFOCOM.*
[20] R. Sugihara and R. Gupta, "Clock synchronization with deterministic accuracy guarantee," *Wireless Sensor Netw.*, pp. 130–146, 2011.
[21] MAXIM, "32.768kHz temperature-compensated crystal oscillator." Available: http://datasheets.maxim-ic.com/en/ds/DS32kHz.pdf
[22] T. Schmid, Z. Charbiwala, R. Shea, and M. B. Srivastava, "Temperature compensated time synchronization," *IEEE Embedded Sys. Lett.*, vol. 1, no. 2, pp. 37–41, 2009.
[23] Crossbow Technology, "MICA2 data sheet." Available: http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf
[24] Epson Electroincs Inc., "SE2412CT-ND data sheet." Available: http://www.eea.epson.com/portal/pls/portal/docs/1/1407484.PDF
[25] Seiko Instruments Inc., "Quartz crystal product catalogue." Available: http://www.sii.co.jp/compo/catalog/crystal_en.pdf
[26] Z. Yang, J. Pan, and L. Cai, "Adaptive clock skew estimation with interactive multi-model Kalman filters for sensor networks," in *Proc. 2010 IEEE ICC.*
[27] D Brunelli, D Balsamo, G Paci, and L Benini, "Temperature compensated time synchronisation in wireless sensor networks," *Electron. Lett.*, vol. 48, no. 16, pp. 1026–1028, 2012.
[28] P. Marchetto, A. Strickhart, R. Mack, and H. Cheyne, "Temperature compensation of a quartz tuning-fork clock crystal via post-processing," in *Proc. 2012 IEEE Int. Frequency Control Symp.*
[29] E. Gerber and R. Sykes, "State of the art: quartz crystal units and oscillators," *Proc. IEEE*, vol. 54, no. 2, pp. 103–116, 1966.
[30] K. Sundaresan, G. Ho, S. Pourkamali, and F. Ayazi, "Electronically temperature compensated silicon bulk acoustic resonator reference oscillators," *IEEE J. Solid-State Circuits*, vol. 42, no. 6, pp. 1425–1434, 2007.
[31] Crossbow Technology, "MTS-MDA sensor board users manual." Available: http://www.xbow.com/Support/Support_pdf_files/MTS-MDA_Series_Users_Manual.pdf
[32] Y. Zhang and A. Srivastava, "Adaptive and autonomous thermal tracking for high performance computing systems," in *Proc. 2010 ACM/IEEE Design Automation Conf.*
[33] S. Sharifi and T. S. Rosing, "Accurate direct and indirect on-chip temperature sensing for efficient dynamic thermal management," *IEEE Trans. Computer-Aided Des. Integr. Circuits Syst.*, vol. 29, no. 1, pp. 1586–1599, 2010.
[34] Y. C. Wu, Q. Chaudhari, and E. Serpedin, "Clock synchronization of wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 28, no. 1, pp. 124–138, 2011.
[35] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *ACM SIGOPS Operating Syst. Rev.*, vol. 36, pp. 147–163, 2002.
[36] J. Van Greunen and J. Rabaey, "Lightweight time synchronization for sensor networks," in *Proc. 2003 Wireless Sensors Networks*, pp. 11–19.
[37] H. Kopetz and W. Ochsenreiter, "Clock synchronization in distributed real-time systems," *IEEE Trans. Comput.*, vol. 100, no. 8, pp. 933–940, 1987.

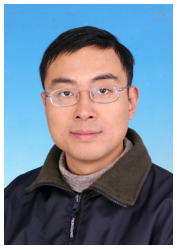
Zhe Yang (S'09) received his B.S. degree in information engineering in 2005 and the M.S. degree in control theory and engineering in 2008, both from Xi'an Jiaotong University, Xi'an, China. He received the Ph.D. degree in electrical and computer engineering from the University of Victoria, Victoria, British Columbia, Canada, in 2013. He then joined the Department of Computer Science at Northwestern Polytechnical University, Xi'an, China, with the exceptional promotion to associate professor. His research areas include protocol design, optimization, and resource management of wireless communication networks.

Liang He (S'09-M'12) received his B.Eng. degree in 2006 and Ph.D. degree in 2011, both in computer science, from Tianjin University and Nankai University, respectively. During Oct. 2009 to Oct. 2011, he worked at Panlab at the University of Victoria, Canada, as a visiting research student. He is currently a research scientist at Singapore University of Technology and Design, Singapore. He is a recipient of the best paper awards of IEEE WCSP 2011 and IEEE GLOBECOM 2011. His current research interests mainly reside in mobile wireless sensor networks and cyber-physical systems.



Lin Cai (S'00–M'06–SM'10) received her M.A.Sc. and Ph.D. degrees (awarded Outstanding Achievement in Graduate Studies) in electrical and computer engineering from the University of Waterloo, Waterloo, Canada, in 2002 and 2005, respectively. Since 2005, she has been an Assistant Professor and then an Associate Professor with the Department of Electrical & Computer Engineering at the University of Victoria. Her research interests span several areas in wireless communications and networking, with a focus on network protocol and architecture design supporting emerging multimedia traffic over wireless, mobile, ad hoc, and sensor networks.

She received the NSERC Discovery Accelerator Supplement Grant in 2010, and the best paper awards of IEEE ICC 2008 and IEEE WCNC 2011. She served as a TPC symposium co-chair for IEEE Globecom'10 and Globecom'13, and is an Associate Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, the *EURASIP Journal on Wireless Communications and Networking*, the *International Journal of Sensor Networks*, and the *Journal of Communications and Networks (JCN)*.



Jianping Pan (S'96–M'98–SM'08) is currently an associate professor of computer science at the University of Victoria, Victoria, British Columbia, Canada. He received his bachelor's and Ph.D. degrees in computer science from Southeast University, Nanjing, Jiangsu, China, and he did his postdoctoral research at the University of Waterloo, Waterloo, Ontario, Canada. He also worked at Fujitsu Labs and NTT Labs. His area of specialization is computer networks and distributed systems, and his current research interests include protocols for advanced networking, performance analysis of networked systems, and applied network security. He received the IEICE Best Paper Award in 2009, the Telecommunications Advancement Foundation's Telesys Award in 2010, the WCSP 2011 Best Paper Award, and the IEEE Globecom 2011 Best Paper Award, and has been serving on the technical program committees of major computer communications and networking conferences including IEEE INFOCOM, ICC, Globecom, WCNC, and CCNC. He is a senior member of the ACM and a senior member of the IEEE.