2022 IEEE International Performance, Computing, and Communications Conference (IPCCC) | 978-1-6654-8018-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/IPCCC55026.2022.9894317

Toward a Shared Sense of Time for a Network of Batteryless, Intermittently-powered Nodes

Vishal Deep, Mathew L. Wymore, Daji Qiao, Henry Duwe
Electrical and Computer Engineering, Iowa State University, Ames, IA, USA
{vdeep, mlwymore, daji, duwe}@iastate.edu

***Abstract*—Wireless sensor nodes powered solely by energy-harvesting show promise in enabling truly pervasive, long-duration sensing by avoiding the fragility, cost, and maintenance limitations of batteries. Unfortunately, since the amount of energy harvested is often significantly less than the active consumption of the device, these devices operate intermittently with limited control of when they are on and how long they are off. Such uncontrollable intermittency first poses a challenge for traditional time synchronization error metrics, since nodes cannot reliably communicate time values at known intervals, resulting in the illusion that nodes are out-of-sync. Second, long duration off-times can exceed the inherent timing limit of persistent clocks that intermittent nodes rely on to measure off-times. Bursty groups of long off-times can cause traditional time synchronization mechanisms to re-converge slowly, incurring significant periods of high error in which nodes are effectively out-of-sync. In this paper, we define the meaning of a shared sense of time for intermittently-powered nodes, and propose two intermittency-aware synchronization error metrics. We then propose an intermittency-resilient time synchronization mechanism, called Levee, that exhibits more rapid re-convergence after losing time and a 2.12x reduction in maximum time synchronization error for a 48-hour period.**

***Index Terms*—Batteryless, Intermittent, Energy Harvesting, Time Synchronization, Wireless Sensor Networks (WSNs)**

I. Introduction

The number of pervasive devices continues to rapidly grow during the Internet of Things (IoT) era with predictions that by only 2025, there will be more than 25 billion deployed IoT devices [1]. If batteries are powering the vast number of IoT devices, the requisite periodic maintenance of batteries represents a significant hurdle in unleashing the full potential of ubiquitous sensing and computing [2]. A promising solution is to use small capacitors to store energy harvested from ambient sources, such as RF, thermal, or kinetic. However, the energy harvested is generally much smaller than the amount of energy a device may consume. This makes batteryless nodes intermittent in nature—nodes turn on when the voltage on their energy storage capacitor is above an *on threshold* and they die when the voltage is below an *off threshold*, as shown in Fig. 1 (Top). While this *lifecycling* allows nodes to function at all, it makes timekeeping particularly challenging since the active clocks within the node are powered off when no energy is available during the off-times, as shown in Fig. 1 (Middle).

Having an accurate sense of time is important to many applications, including security, data timestamping, and data staleness [3]. Maintaining a common sense of time across nodes is critical for scheduling communication and computation, coordinating sampling, and power management in wireless sensor networks [4, 5]. For example, a medium access control protocol (MAC) may make use of accurate timing on multiple nodes to avoid collisions [6], while intermittent communication benefits from accurate time information to manage power to coordinate on-times [7].

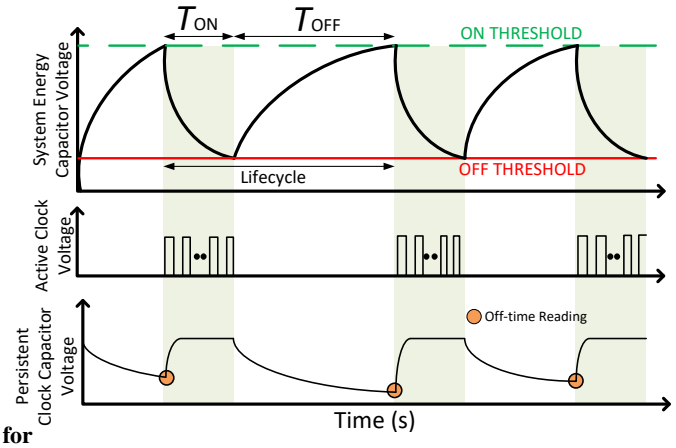


Fig. 1: Lifecycling behavior of batteryless, intermittent sensor nodes. MCU, oscillators, radios, and sensors are only on during green shaded regions. Off-times are estimated by persistent clocks that have an inherent maximum duration.

In order to provide any continuous sense of time, intermittent nodes may measure off-times using persistent clocks such as [7, 8, 9]. As shown in Fig. 1 (Bottom), these clocks measure the voltage decay of capacitive elements during the off-time and, using known decay models, estimate the off-time duration. However, even the best of such clocks is subject to higher error than oscillatory clocks and has a maximum off-time that can be measured before timing information is lost. These challenges in local timekeeping make it even harder to perform time synchronization between nodes. Despite these challenges, we observe that residual timing information still exists in the network. These disparate time shards can be smoothed and stitched together to form a useful *shared sense of time*.[1]

This work was supported in part by the U.S. National Science Foundation under Grants 1730275 and 2008548.

[1]The term time synchronization in wireless sensor networks is well established. Given the difference in target accuracy, in relative scarcity of high accuracy timing information in networks of intermittent nodes, and in different metrics required, we refer to the analogous problem of time synchronization as developing a *shared sense of time*.

In this paper, we first show that the conventional metrics of time synchronization fail to accurately represent the shared sense of time in a network of intermittent nodes. Therefore, we propose a novel set of intermittency-aware metrics that capture the amount of meaningful timing information held across intermittent nodes. Armed with new metrics, we develop an intermittency-resilient synchronization mechanism for maintaining a shared sense of time. The main contributions of this paper are as follows:

- We formally define a *shared sense of time* by presenting a couple of intermittency-aware metrics for time synchronization error in a network of intermittent nodes.
- We evaluate our synchronization error metrics with respect to availability and resiliency properties.
- We propose an intermittency-resilient time synchronization mechanism, *Levee*. Using our metrics in a two-node network, evaluations show that Levee re-converges rapidly after losing time information and yields a 2.12x reduction in maximum pairwise error for a 48-hour period, when compared with the direct application of flooding-based time synchronization protocol (FTSP) [4, 10].

II. Background

Even in traditional, continuously-powered wireless sensor networks, telling time is challenging.[2] Each node has its own local oscillator-based clock. Due to imperfections in the oscillator and environmental effects, each of these clocks reports time differently.

A commonly used clock time error model for oscillator-based clocks [11] relates node i's clock time, $C_i(t)$, to a reference time, $R(t)$, using three parameters: *offset (α_i), skew (β_i), and drift (γ_i)*, as follows [12]:

$$C_i(t) = \alpha_i + \beta_i R(t) + \frac{1}{2}\gamma_i R(t)^2, \tag{1}$$

where t is the ground truth time. Depending on the context, $R(t)$ may be the ground truth time t, a global clock (e.g., an atomic clock [13, 14]), or the local clock of one node (e.g., $C_{root}(t)$ in [15, 4]). These three parameters have physical meanings. Clock offset is the difference in time between the two clocks at $t = 0$. Clock skew is the difference in frequency between the two clocks. Skew is mainly caused by inaccuracies or differences in the oscillator (e.g., a crystal). Clock drift captures variations in the skew mainly caused by temperature, humidity, supply voltage, age of quartz, etc. In time synchronization studies for wireless sensor networks, clock drift is generally assumed to be relatively slow and not directly modeled.

[2]We consider battery-powered nodes as continuously-powered because some of the processor components are always rapidly bootable even if they are in low-power states. Indeed, these nodes either keep an internal or external RTC running. Of course, other components such as the radio could be turned off to save energy.

A. Time Synchronization Mechanisms

The mechanisms used to converge the clocks of nodes towards a reference clock are called *time synchronization*. To solve the problem of time synchronization, various methods have been proposed for how a node can generate a local estimate, $E_i(t)$, of the reference time $R(t)$ from its own local clock reading, $C_i(t)$, and past timing information.

These methods generally exchange various forms of timing information coupled with information from a local clock error model such as Equation (1). The time synchronization methods can be broadly classified as *flooding based*, *distributed*, and *cluster based*. In a flooding based approach, a reference node is selected that floods the network with time synchronization messages such as timestamp of the reference node. The receiver nodes adjust their time estimate based on the received timestamp relative to the local time [4, 16, 17, 18]. While these mechanisms have high accuracy, they tend to have poor robustness, poor scalability, and large synchronization overheads [19]. To reduce or eliminate these problems, distributed time synchronization mechanisms have been proposed. In these approaches, there is no single reference or root node; every node in the network only exchanges time information with its neighbors and tries to converge its time to an agreed-upon network time, such as the average or maximum time [20, 21, 22, 23]. Since these distributed mechanisms are relatively slow to converge [24], cluster based methods were proposed. These approaches create clusters of nodes and select cluster heads to localize synchronization operations that prevent the accumulation of errors and improve convergence speed [25, 26, 19, 27].

To our knowledge, Hourglass clocks [10] is the only prior work that studies time synchronization for batteryless intermittent nodes. It focuses on improving persistent clock measurements while assuming that all nodes can be on at the same time. Currently, no time synchronization mechanisms or metrics consider the limited control that batteryless, intermittent nodes can exert over their on-times.

B. Time Synchronization Metrics

The performance of a time synchronization mechanism is often evaluated based on several metrics, including accuracy, energy, number of packets, and time to converge. Of particular note, accuracy is usually measured using the *synchronization error metric*. Synchronization error is defined as the pairwise difference between time estimates of the nodes in the network:

$$\textit{Sync Error} = E_i(t) - E_j(t),\ \forall i, j \in N. \tag{2}$$

This error metric is measured at regular intervals of time T_m (i.e., the metric's measurement period). To take the measurement, a reference broadcaster can be used to initiate a query to all the nodes in the network and in response nodes report back their time estimates [4, 10].

[Time Synchronization Mechanism vs. Metric] An important difference between a synchronization mechanism (e.g., FTSP - Flooding based Time Synchronization Protocol [4])

and the metric (e.g., synchronization error) used to evaluate it in simulations or experiments is as follows: the mechanism is intended to be deployed and executed by sensor nodes in actual applications, while the metric is intended to evaluate the performance of the mechanism in a controlled lab environment or simulation, usually prior to actual deployment.

III. A Case for Intermittent-aware Time Synchronization

Despite time synchronization being well-studied in battery-powered wireless sensor networks, intermittent operation poses unique challenges to time synchronization. The challenges impact both the metrics used to evaluate timing synchronization accuracy and the mechanisms themselves.

A. Challenges in Local Timekeeping on Intermittent Nodes

Battery-powered nodes can keep track of the local time with a high level of accuracy using an oscillator-based clock/timer and can share time information with other nodes (and testbed infrastructure) using frequent, predictable communication. The synchronization errors in such networks are in μs to ms range [4, 27, 28]. In contrast, intermittently-powered nodes exhibit frequent power failures and non-deterministic off-times due to the stochastic nature of harvested energy that may result in errors of seconds or more [7, 29, 30, 31].

When intermittent nodes do not have enough energy for operation, all the active components on-board, such as MCUs, oscillating clocks/timers, sensors, radios, etc., are completely unpowered (e.g., during the white regions in Fig. 1). This causes traditional time information to be lost between on-times, fundamentally hindering time synchronization for intermittent nodes. In order to maintain a local continuous sense of time (i.e., a continuous $C_i(t)$) across multiple on-times), intermittent nodes can use active, oscillator-based clocks during on-times and persistent, remanence-decay clocks [7, 9] to estimate off-times[3]. As shown in Fig. 1, persistent clocks estimate off-times by measuring the remaining voltage on a capacitor (or capacitors) immediately after an off-time. Since the capacitor starts an off-time at a known voltage and decays at a known rate, the remaining voltage can be converted to an elapsed time. The elapsed time is summed with the final value of the previous on-time's active clock saved to non-volatile memory to become the initial value of the active clock for the current on-time. As shown in the left part of Fig. 2, the result is a local clock, $C_i(t)$, which can maintain time across off-times, but can only be read during on-times.

Even state-of-the-art persistent clocks suffer from relatively higher errors and variations compared to oscillator-based clocks/timers. In addition to the clock properties described in Section II, persistent clocks can add frequent large changes in offset, as shown in Fig. 3. These large changes in offset are due to a lack of energy and the limited time measurement

[3]Systems may also use a crystal oscillator based clock running on a decaying capacitance [32] to achieve higher accuracy comparable to persistent clocks, but this approach has a very short measurement duration because it can only measure time until the oscillator settles ($\approx$ 1 second).

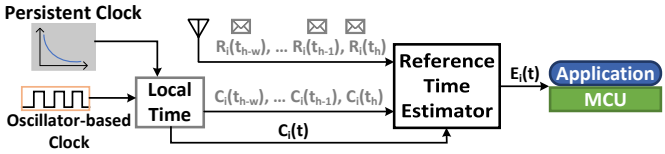


Fig. 2: Intermittent time synchronization. At time t, node i determines its local time, $C_i(t)$, using both oscillator-based and persistent clocks. At the time of each communication handshake, t_h, the local time, $C_i(t_h)$ and received reference time, $R_i(t_h)$, are recorded. The node can then calculate its estimated shared time, $E_i(t)$, by using the current local time and past time information.

duration of persistent clocks—i.e., cases where the persistent clock capacitance completely runs out of energy. This situation is referred as *clock dead period* in Fig. 3. Clock dead period induces a large negative change in the offset (i.e., α_i) of a node's clock reading. While the skew (i.e., the slope of the clock reading relative to the ground truth time – β_i) remains relatively similar before and after the clock dead period, it appears as a rapid temporary drift in the clock model described in Section II.

To concretely understand how frequent and severe these impacts are, Fig. 4a shows a 48-hour period of measured off-times and time estimates of an RF-harvesting node equipped with a lightweight state-of-the-art persistent clock (denoted as PCLK in the figure) deployed in our research lab, which has many computational devices of various scales in use and students moving about within the lab. We observe that the persistent clock generally tracks the externally-measured off-times well. However, during the most active times in the lab (region R3), off-times often exceeded the max duration that the persistent clock can measure. During these times, the node experiences frequent negative offsets in its local clock reading.

All these issues result in prior time synchronization metrics and mechanisms being incompatible with intermittent nodes. Next, we describe the challenges specific to the design of time

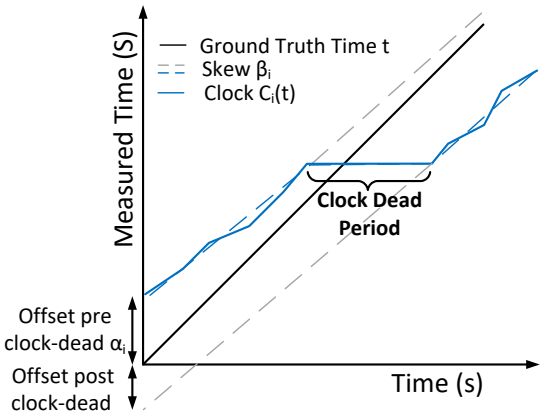
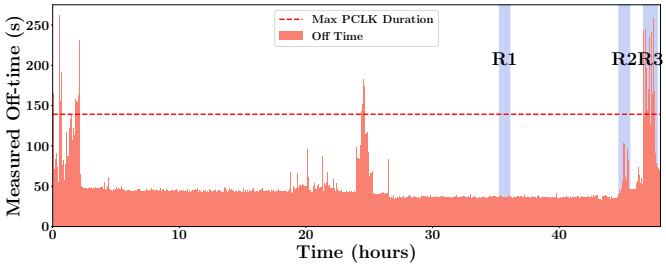
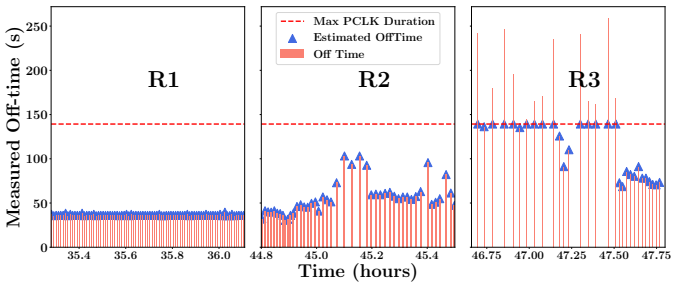


Fig. 3: Example clock time errors in an intermittently-powered node. Clock dead periods from unexpectedly-long off-times result in a large negative change in the offset of a node's clock.



(a) 48-hour measured off-time trace from an RF-powered node.



(b) Three different regions of measured and estimated off-time.

Fig. 4: Measured persistent clock errors of an RF-harvesting node deployed in our lab. During certain regions, observed off-times vary and can unexpectedly and repeatedly go higher than what a persistent clock can measure (e.g., R3).

synchronization metrics and mechanisms.

B. Challenges in Time Synchronization Error Metrics

Intermittent nodes pose challenge to conventional time synchronization error metrics in two ways: (i) the inability to get a representative error measurement at a specific sampling time, i.e., every T_m – the metric measurement period, and (ii) the inability to calculate pairwise time errors at a specific sampling time.

To illustrate the first challenge, consider the node behavior shown in Fig. 1. Clearly, in an experimental evaluation, a time estimate can only be produced and reported during the node's on-times shown in green. Since off-times are often much longer than on-times, it is likely that most nodes are not on at the end of a metric measurement period. This results in most nodes being incapable of participating in a conventional time synchronization error metric despite having a reasonably accurate time estimate.

The second challenge arises from the limited control that intermittent nodes can exert over their on-times. In order to calculate pairwise error between nodes during evaluation, both nodes must be on at the same time to report their current estimate of time (i.e., $E_i(t)$) at a particular sampling point. Any pair of nodes that cannot report their time are considered out-of-sync thus will not contribute to synchronization error. For intermittent nodes, this means that most, if not all, pairs could be considered out-of-sync. However, many of these nodes may still be maintaining a relatively accurate local estimate of time. The conventional metric for evaluating time synchronization cannot capture this important nuance in intermittent timekeeping.

Therefore, we need a new metric that can provide a meaningful understanding of the shared sense of time in a network of intermittent nodes.

C. Challenges in Time Synchronization Mechanisms

If intermittent nodes are energy rich with relatively short off-times and can communicate frequently with neighbor nodes, time synchronization mechanisms that were designed for battery-powered devices may provide a reasonable time synchronization between them [10]. However, it is common that one or multiple intermittent nodes could be energy-poor and thus have long off-times with higher clock error rates and rare communication (e.g., time region R3 in Fig. 4b). In this case, traditional time synchronization mechanisms could exhibit a very high error for a long duration of time.

Fig. 5 illustrates the particular challenge faced by traditional time synchronization mechanisms. We consider a two-node network where the nodes synchronize time using the prototypical Flooding Time Synchronization Protocol (FTSP) [4].[4] Node 1 is the root node of the synchronization spanning tree, with its local clock as the target reference time, $R(t)$, for the entire network. Node 2 is its only child node. As depicted in the center part of Fig. 2, in FTSP, node 2 keeps a window (W) of paired values—*local time* $C_2(t)$ *and received reference time* $R_2(t)$—and uses linear regression to estimate the reference time based on these values.

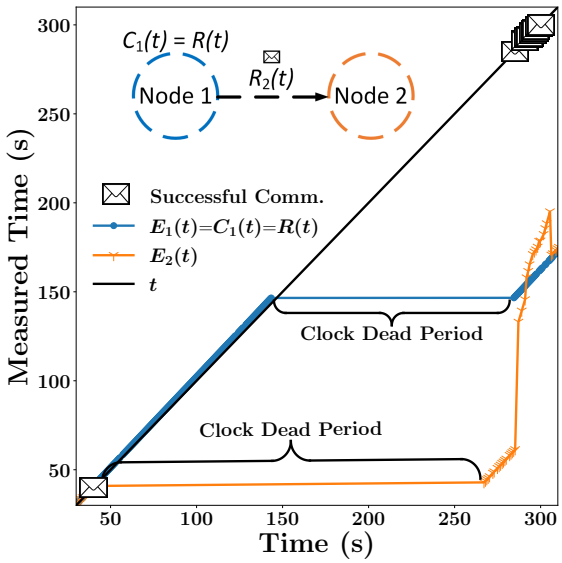


Fig. 5: Example Flooding Time Synchronization Protocol (FTSP) with two intermittent nodes each experiencing clock dead periods. Frequent clock dead periods cause a rapid drift in clock reading that FTSP is unable to quickly track/recover.

[4]We choose FTSP for illustrative purposes since it forms the base in the only prior work that evaluates time synchronization for intermittent nodes [10]. This work assumes that nodes can all be on every T_m time. Other mechanisms target improvements in error accumulation, scalability, and efficiency in communication, but are still susceptible to rapid drift in clock reading.

In this example, both nodes have low energy-harvesting rates and incur a clock dead period as a result. Soon after their clock dead periods, the nodes manage to communicate with each other. Despite this communication, it takes 13 on-times and nine communications for node 2 to fully re-synchronize with node 1. Indeed, when node 2 first communicates after the clock dead period, there is a significant error in node 2's time estimate (i.e., $E_2(t)$) as it attempts to converge to node 1's clock. This happens because the new, negative offset propagates through the window of regression pairs. In general, if long clock dead periods occur frequently (e.g., R3 in Fig. 4), this may cause an intermittent node to lose a shared sense of time for a prolonged period.

Therefore, we need a new time synchronization mechanism for intermittent nodes that considers the measured on-/off-times and the local energy harvesting conditions.

IV. Intermittency-Resilient Time Synchronization

Challenges described in Section III lead us to first develop *synchronization error metrics* that can be meaningfully evaluated even in the presence of frequent and long off-times. We then propose a *time synchronization mechanism* specifically for intermittently-powered nodes.

A. Intermittency-Resilient Synchronization Error Metrics

Based on the discussion of the conventional synchronization error metric and its limitations in batteryless, intermittently-powered nodes, we have concluded that an ideal synchronization metric shall have the following two desired properties. Recall that T_m is the metric measurement period; in other words, a metric measurement is produced every T_m time.

[Availability] – A good metric shall be able to quantify the shared sense of time between a network of sensor nodes at each pre-determined measurement time as much as possible. Metric availability is defined as the **percentage** of measurement time instances when a metric reading can be produced.

[Resiliency] – A good metric shall remain available regardless of the energy harvesting rates of sensor nodes. In other words, the metric shall remain available even when sensor nodes have relatively few lifecycles in a measurement period. We define resiliency of a metric as **one minus the absolute value of the correlation coefficient** between metric availability and the average number of lifecycles per measurement period for nodes in the network. A larger (i.e., closer to 1) resiliency value means that metric availability is less affected by the rate of energy harvesting, and hence is more resilient.

With these properties in mind, we propose two new metrics ($M_{\text{handshake}}$ and $M_{\text{lifecycle}}$) that can be used as per the application requirements to evaluate intermittently-powered wireless sensor networks. We also formally define the conventional metric as $M_{\text{conventional}}$, which serves as the baseline for comparison.

1) $M_{conventional}$: The first metric is the conventional pairwise metric, which is defined for conventional battery-powered sensor networks as follows:

$$M_{\text{conventional}}(t) = \begin{cases} \text{undefined}, & |P_B(t)| = 0, \\ \frac{1}{|P_B(t)|} \sum_{(i,j,t)\in P_B(t)} |E_i(t) - E_j(t)|, & |P_B(t)| > 0, \end{cases} \quad (3)$$

where $t \in \{T_m, 2T_m, \cdots\}$ is the metric measurement time, $P_B(t)$ is the set of all pairs of nodes that are both on at time t, and $E_i(t)$ is node i's estimation of time t. Note that in a conventional battery-powered sensor network, this metric is generally well-defined since most nodes are expected to be on at each measurement time. In comparison, in an intermittent batteryless sensor network, there is no guarantee that nodes are on at time t; therefore, the availability of this metric depends on factors such as lifecycle ratios, which we will evaluate later in Section V-B.

We use a toy example in Fig. 6 to explain how the metrics are measured and $M_{\text{conventional}}$'s corresponding availability problem. As shown in the figure, at time T_m, $M_{\text{conventional}}$ is undefined since only one node (node 2) is alive, thus there is no shared sense of time, while at time $2T_m$, one pair of nodes (nodes 2 & 3) contribute to and define $M_{\text{conventional}}$, as marked with gold stars in the figure.

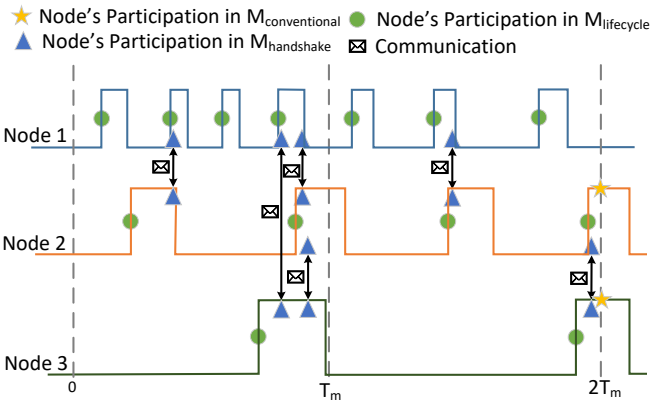


Fig. 6: An example network of three intermittent nodes. Nodes' participation in three different metrics–$M_{\text{conventional}}$, $M_{\text{handshake}}$, $M_{\text{lifecycle}}$– are marked as gold stars, blue triangles, and green circles, respectively. The metrics are measured every T_m time.

2) $M_{handshake}$: The second pairwise metric takes advantage of possible communications between nodes during the metric measurement period. It is defined as follows:

$$M_{\text{handshake}}(t) = \begin{cases} \text{undefined}, & |P_H(t)| = 0, \\ \frac{1}{|P_H(t)|} \sum_{(i,j)\in P_H(t)} |\bar{E}_i(t) - \bar{E}_j(t)|, & |P_H(t)| > 0, \end{cases} \quad (4)$$

where $t \in \{T_m, 2T_m, \cdots\}$ is the metric measurement time. $P_H(t)$ is the set of all pairs of nodes that communicate with each other (and hence are on at the same time) at

some instances between $t - T_m$ and t. For each pair of communicating nodes i and j, we define

$$\bar{E}_{i,j}(t) = \frac{1}{H_{i,j}(t)} \sum_{h=1}^{H_{i,j}(t)} |E_i(t_h) - E_j(t_h)| \quad (5)$$

as the average pairwise difference between their time estimates, where $H_{i,j}(t)$ is the total number of communication instances between $t - T_m$ and t.

We use the same scenario in Fig. 6 as an example. Pairs of nodes that contribute to $M_{\text{handshake}}$ are marked as blue triangles in the figure. In this example, $|P_H(T_m)| = 3$ since all three pairs of nodes (nodes 1 & 2, nodes 1 & 3, and nodes 2 & 3) communicate between time 0 and T_m, while $|P_H(2T_m)| = 2$ since only two pairs of nodes (nodes 1 & 2, and nodes 2 & 3) communicate between T_m and $2T_m$. Compared with $M_{\text{conventional}}$, $M_{\text{handshake}}$ is available at both T_m and $2T_m$.

3) $M_{lifecycle}$: The third pairwise metric extends the second metric by considering all nodes' lifecycles during the previous T_m duration. It is defined as follows:

$$M_{\text{lifecycle}}(t) = \begin{cases} \text{undefined}, & |P_L(t)| = 0, \\ \frac{1}{|P_L(t)|} \sum_{(i,j,t)\in P_L(t)} |\hat{E}_i(t) - \hat{E}_j(t)|, & |P_L(t)| > 0, \end{cases} \quad (6)$$

where $t \in \{T_m, 2T_m, \cdots\}$ is the metric measurement time, and $P_L(t)$ is the set of all pairs of nodes that are on (but not necessarily at the same time) between $t - T_m$ and t. For each node i, we define

$$\hat{E}_i(t) = \frac{1}{L_i(t)} \sum_{\ell=1}^{L_i(t)} |E_i(t_\ell) - t_\ell| \quad (7)$$

as the average difference between its estimated time and reference time, averaged over time estimates taken at the beginning of each lifecycle (denoted as t_ℓ) between $t - T_m$ and t, where $L_i(t)$ is the total number of lifecycles that i has experienced between $t - T_m$ and t.

Let's revisit the example in Fig. 6, where we mark nodes contributing to $M_{\text{lifecycle}}$ as green circles. We observe that all three pairs of nodes contribute to $M_{\text{lifecycle}}$ (i.e., $|P_L| = 3$ for both T_m and $2T_m$) and participate more often at each on-time. For instance, for $t = 2T_m$, $P_L(2T_m) = \{(1,2),(1,3),(2,3)\}$, $L_1(2T_m) = 3$, $L_2(2T_m) = 2$, and $L_3(2T_m) = 1$.

B. Intermittency-Resilient Time Synchronization Mechanism

Given our observation that clock dead periods cause the appearance of rapid drift in clock readings, we propose to estimate off-times with clock dead periods separately from those with non-depleted persistent clock readings. For the off-times short enough to be measured by the persistent clock (i.e., where enough energy is harvested for the node to turn on before the persistent clock completely decayed), the conventional time synchronization mechanism is used (e.g., FTSP). For the off-times where the energy harvesting rate is low and the persistent clock completely decays (i.e., $V_{\text{cap}} < V_{\text{dead}}$), we use the predicted energy harvesting rate to estimate the duration of the off-time. We refer such an intermittency-resilient time synchronization mechanism as *Levee* since it mitigates time overflowing the banks of persistent clocks. The core function of Levee is shown in Fig. 7.

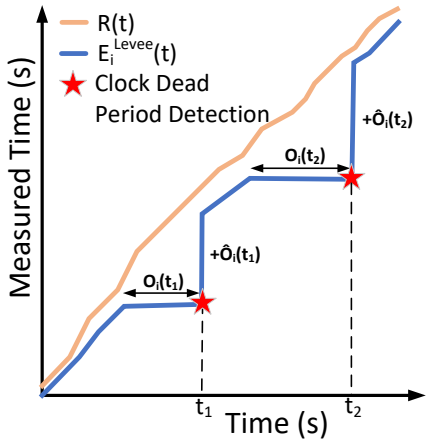


Fig. 7: Conceptual approach of Levee, an intermittency-resilient time synchronization mechanism. Levee detects clock dead periods and estimates their duration using prediction of energy harvesting rates.

While Levee's prediction mechanism may be less accurate than a non-dead persistent clock, it provides some estimate of the time even for long-duration off-times. The resulting time estimation for node i is:

$$E_i^{\text{Levee}}(t) = E_i^{\text{FTSP}}(t) + \sum_{d=1}^{D_i(t)} \hat{O}_i(t_d), \quad (8)$$

where t_d is the detection time of the d-th clock dead period, $\hat{O}_i(t_d)$ is its predicted off-time, and $D_i(t)$ is the total number of clock dead periods since the last handshake.

Prediction of energy harvesting rates is a frequently studied area for energy-harvesting sensor nodes and depends on the specific harvesting type and deployment environment [33, 34, 35]. The more accurate the prediction mechanism used by Levee, the more accurate its time measurement will be. Our implementation of Levee predicts that the energy harvesting rate during a clock dead period is similar to the most recent N clock dead periods in the past. At the time of each handshake, t_h, Levee estimates the actual clock dead periods of node i since the previous handshake according to:

$$O_i(t_d) = \frac{R_i(t_h) - E_i^{\text{FTSP}}(t_h)}{D_i(t_h)}, \quad (9)$$

where $R_i(t_h)$ is the received reference time. The most recent N such $O_i(t_d)$ values at time t are averaged to produce $\hat{O}_i(t)$. We will show in Section V-B that even this straight-forward prediction mechanism is able to yield significant improvements over FTSP in a shared sense of time for our RF-harvesting environment.

Finally, before a new pair of local time and received reference time is added to Levee's underlying FTSP regression window, the total time from any clock dead periods since the last received reference time must be added to the local time.

To avoid impacting the skew estimation of the local persistent clock, the clock dead time must be scaled by the current skew parameter, β_i.

V. Evaluation

A. Evaluation Methodology

We built a custom event-based simulator to simulate the behavior of a network of energy-harvesting intermittently-powered nodes. The simulator takes traces of experimental lifecycle on-times and off-times as input. When nodes' on-times sufficiently overlap, they communicate and exchange time information. The experimental lifecycle data was captured using a continuously-powered "sniffer" node that monitors prototype batteryless nodes consisting of a main application processor with FRAM persistent memory (MSP430FR5994 [36]), a radio (CC1352R [37]), and a PowerCast RF harvester [38] based on those described in [39]. The testbed power was generated by a PowerCast RF transmitter [40]. Various energy harvesting rates were observed by varying the distance between transmitter and nodes. We quantified a node's harvesting rate using the average *lifecycle ratio* $= \frac{T_{\text{on}}}{T_{\text{on}}+T_{\text{off}}}$. Nodes use a 16-bit onboard timer to measure on-time and a state-of-the-art persistent clock, HARC [7], to measure off-times. This version of HARC can measure off-time with high accuracy up to 139 seconds.

As a baseline mechanism, we use FTSP as described in [10]. We simulated a two-node network where node 1 acts as a reference node and node 2 behaves as a child to node 1. Node 2 tries to synchronize its estimated time to node 1's time. During each on-time, node 1 keeps sending packets to node 2 until an acknowledgement is received. For both FTSP and Levee, node 2 saves pairs of local time and reference time—($C_2(t_h)$, $R_2(t_h)$)—from the last 10 communication points and, using the linear regression, estimates the reference time $E_2(t)$. Levee also tracks the past five estimated clock dead period durations to estimate future clock dead periods. Synchronization errors for both mechanisms are measured at the start of each on-time.

B. Evaluation Results

Synchronization Error Metrics: We first evaluated the three metrics over the 48-hour experiment trace shown in Fig. 4a with T_m = 100 seconds. The nodes were synchronized using FTSP and the average lifecycle ratio of node 2 was 0.0035. Fig. 8 shows the CDF of each metric. The asymptotic value achieved by each metric represents its *availability*. $M_{\text{conventional}}$ has an availability of 0, since the nodes were never both on at the same measurement time—it is poorly suited to evaluate intermittent time synchronization. $M_{\text{handshake}}$'s availability is 56.4% since it gets a valid reading from any measurement period in which the pair of nodes communicates. As expected, the availability of $M_{\text{lifecycle}}$ is 98.3% since it produces an error for any period where at least two nodes turn on.

We further evaluated $M_{\text{lifecycle}}$ and $M_{\text{handshake}}$ for *resiliency*. Fig. 9 plots the availability of each metric relative to the average lifecycle ratio of node 2. As the lifecycle ratio decreases, node 2 has longer off-times between on-times and thus less frequent communication with node 1, rapidly degrading $M_{\text{handshake}}$'s availability. Decreasing the lifecycle ratio also reduces the number of on-times per measurement period and, eventually, the number of periods where both nodes turn on. While this causes $M_{\text{lifecycle}}$'s availability to degrade, it only happens at very low lifecycle ratios. These trends result in $M_{\text{lifecycle}}$ having a resiliency of 0.7 compared with $M_{\text{handshake}}$'s 0.19. Recall that, as defined in Section IV-A, a larger (i.e., closer to 1) resiliency value means that metric availability is less affected and hence more resilient to varying energy harvesting rates.

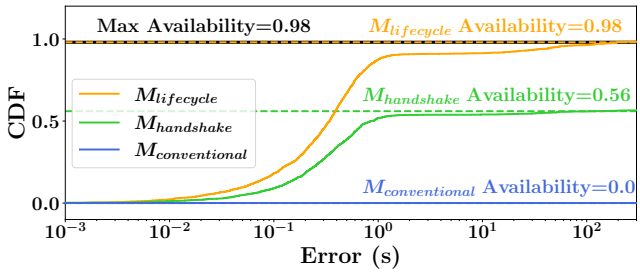


Fig. 8: Comparison of metric availability. Max availability represents the fraction of lifecycles where at least two nodes have an on-time in the measurement period – i.e., a shared sense of time exists.

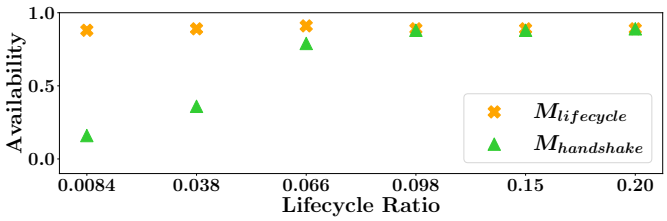


Fig. 9: Metric availability vs. lifecycle ratio. The resulting resiliency of $M_{\text{lifecycle}}$ and $M_{\text{handshake}}$ is 0.7 and 0.19, respectively.

Time Synchronization Mechanisms: With an available and resilient metric—$M_{\text{lifecycle}}$—we can now meaningfully compare the performance of Levee relative to FTSP. Fig. 10 shows the results of 2000 seconds simulation from region R3 of Fig. 4. Each blue diamond represents a single measurement period's synchronization error. Gaps without any error values occur when fewer than two nodes have an on-time during a measurement period—hence no shared sense of time exists. With FTSP, the initial measurement periods after a node experiences a clock dead period often incur large errors (up to 223.4 seconds. Furthermore, the error does not quickly converge back to a steady-state when the node experiences no clock dead periods. Conversely, Levee's maximum synchronization error is 106 seconds and re-converges within one measurement period. Levee can effectively dampen the large errors caused by long clock dead periods by using energy-harvesting predictions for clock dead periods.

For a more holistic comparison, we use the 48-hour trace data from Fig. 4 for node 2 and report $M_{\text{lifecycle}}$ results for the three representative regions in Table I. As expected, in regions

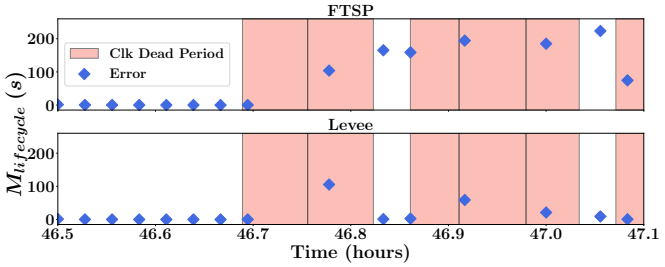


Fig. 10: $M_{\text{lifecycle}}$ synchronization error of FTSP and Levee for 2000 seconds of region R3.

R1 (little off-time variation and no clock dead periods) and R2 (high off-time variation, but no clock dead periods) exhibit low synchronization errors and both mechanisms (FTSP and Levee) work identically. However, in region R3 where there are high off-time variations that frequently cause clock dead periods, the mean and max synchronization error in FTSP is 55.3 and 223 seconds, respectively. In comparison, Levee reduces this error by over 2x, and provides an effective shared sense of time across all regions. Recall that our experimental data (see Fig. 4a) contained multiple hour-long regions with clock dead periods within a 48-hour time. During these relatively common times, Levee provides a more consistent and accurate sense of time compared to FTSP.

TABLE I: Comparison of synchronization error of FTSP and Levee during three representative time regions from Fig. 4, using the $M_{\text{lifecycle}}$ metric at T_m = 100s.

Time Region from Fig. 4	Mechanism	Mean (s)	Max (s)
R1	FTSP	0.52	2.98
	Levee	0.52	2.98
R2	FTSP	0.50	1.51
	Levee	0.50	1.51
R3	FTSP	55.3	223
	Levee	24.5	106

VI. Conclusion

In conclusion, this paper formally defined a shared sense of time for batteryless, intermittent wireless sensor nodes. The resulting definition of time synchronization error metrics were evaluated in terms of availability and resiliency with the lifecycle-based metric achieving both high availability and resiliency. This metric allowed us to propose and evaluate Levee, an intermittency-resilient time synchronization mechanism that uses energy-harvesting prediction to improve the accuracy of time synchronization of intermittent nodes. The metric and mechanism enable future exploration of energy-accuracy trade-offs, communication interactions, and lifecycle management implications for maintaining a *shared sense of time* in multi-node batteryless, intermittent sensor networks.

References

[1] Satyajit Sinha. *State of IoT 2021: Number of connected IoT devices growing 9% to 12.3 billion globally, cellular IoT now surpassing 2 billion*. 2021. URL: https://iot-analytics.com/number-connected-iot-devices/ (visited on 03/18/2022).

[2] Rafael Reyes. *Overcoming the Battery Obstacle to Ubiquitous Sensing — Finally*. 2021. URL: https://internetofbusiness.com/overcoming-the-battery-obstacle-to-ubiquitous-sensing-finally/ (visited on 03/27/2022).

[3] Waltenegus Dargie and Christian Poellabauer. *Fundamentals of wireless sensor networks: theory and practice*. John Wiley & Sons, 2010.

[4] Miklós Maróti et al. "The flooding time synchronization protocol". In: *Proceedings of the 2nd international conference on Embedded networked sensor systems*. 2004, pp. 39–49.

[5] Kaisheng Ma et al. "NEOFog: Nonvolatility-exploiting optimizations for fog computing". In: *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems*. 2018, pp. 782–796.

[6] Venkatesh Rajendran, Katia Obraczka, and Jose Joaquin Garcia-Luna-Aceves. "Energy-efficient collision-free medium access control for wireless sensor networks". In: *Proceedings of the 1st international conference on Embedded networked sensor systems*. 2003, pp. 181–192.

[7] Vishal Deep et al. "HARC: A Heterogeneous Array of Redundant Persistent Clocks for Batteryless, Intermittently-Powered Systems". In: *2020 IEEE Real-Time Systems Symposium (RTSS)*. 2020, pp. 270–282. DOI: 10.1109/RTSS49844.2020.00033.

[8] Josiah Hester et al. "Persistent clocks for batteryless sensing devices". In: *ACM Transactions on Embedded Computing Systems (TECS)* 15.4 (2016), pp. 1–28.

[9] Jasper de Winkel et al. "Reliable timekeeping for intermittent computing". In: *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*. 2020, pp. 53–67.

[10] Eren Çürük et al. "On the Accuracy of Network Synchronization Using Persistent Hourglass Clocks". In: *Proceedings of the 7th International Workshop on Energy Harvesting & Energy-Neutral Sensing Systems*. 2019, pp. 35–41.

[11] David W Allan et al. "Time and frequency(time-domain) characterization, estimation, and prediction of precision clocks and oscillators". In: *IEEE transactions on ultrasonics, ferroelectrics, and frequency control* 34.6 (1987), pp. 647–654.

[12] Francisco Tirado-Andrés and Alvaro Araujo. "Performance of clock sources and their influence on time synchronization in wireless sensor networks". In: *International Journal of Distributed Sensor Networks* 15.9 (2019), p. 1550147719879372.

[13] Peter Volgyesi et al. "Time synchronization services for low-cost fog computing applications". In: *2017 International Symposium on Rapid System Prototyping (RSP)*. IEEE. 2017, pp. 57–63.

[14] Robin E Kim et al. "Synchronized sensing for wireless monitoring of large structures". In: *Smart Structures and Systems* 18.5 (2016), pp. 885–909.

[15] Bharath Sundararaman, Ugo Buy, and Ajay D Kshemkalyani. "Clock synchronization for wireless sensor networks: a survey". In: *Ad hoc networks* 3.3 (2005), pp. 281–323.

[16] Christoph Lenzen, Philipp Sommer, and Roger Wattenhofer. "Optimal clock synchronization in networks". In: *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*. 2009, pp. 225–238.

[17] Kasim Sinan Yildirim and Aylin Kantarci. "Time synchronization based on slow-flooding in wireless sensor networks". In: *IEEE Transactions on Parallel and Distributed Systems* 25.1 (2013), pp. 244–253.

[18] Kasım Sinan Yıldırım and Önder Gürcan. "Efficient time synchronization in a wireless sensor network by adaptive value tracking". In: *IEEE Transactions on wireless communications* 13.7 (2014), pp. 3650–3664.

[19] Jie Wu et al. "Cluster-based consensus time synchronization for wireless sensor networks". In: *IEEE Sensors Journal* 15.3 (2014), pp. 1404–1413.

[20] Luca Schenato and Federico Fiorentin. "Average timesync: A consensus-based protocol for time synchronization in wireless sensor networks". In: *IFAC Proceedings Volumes* 42.20 (2009), pp. 30–35.

[21] Philipp Sommer and Roger Wattenhofer. "Gradient clock synchronization in wireless sensor networks". In: *2009 International Conference on Information Processing in Sensor Networks*. IEEE. 2009, pp. 37–48.

[22] Jianping He et al. "Time synchronization in WSNs: A maximum-value-based consensus approach". In: *IEEE Transactions on Automatic Control* 59.3 (2013), pp. 660–675.

[23] Kasim Sinan Yildirim and Aylin Kantarci. "External gradient time synchronization in wireless sensor networks". In: *IEEE Transactions on Parallel and Distributed Systems* 25.3 (2013), pp. 633–641.

[24] Zhaowei Wang et al. "Cluster-based maximum consensus time synchronization for industrial wireless sensor networks". In: *Sensors* 17.1 (2017), p. 141.

[25] Jeremy Elson, Lewis Girod, and Deborah Estrin. "Fine-grained network time synchronization using reference broadcasts". In: *ACM SIGOPS Operating Systems Review* 36.SI (2002), pp. 147–163.

[26] Qun Li and Daniela Rus. "Global clock synchronization in sensor networks". In: *IEEE Transactions on computers* 55.2 (2006), pp. 214–226.

[27] Baofeng Zhou and Mehmet C Vuran. "Cortis: Correlation-based time synchronization in Internet of Things". In: *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE. 2019, pp. 1–7.

[28] Tie Qiu et al. "A robust time synchronization scheme for industrial internet of things". In: *IEEE Transactions on Industrial Informatics* 14.8 (2017), pp. 3570–3580.

[29] Geoff V Merrett and Bashir M Al-Hashimi. "Energy-driven computing: Rethinking the design of energy harvesting systems". In: *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*. IEEE. 2017, pp. 960–965.

[30] Stephen P Beeby et al. "A comparison of power output from linear and nonlinear kinetic energy harvesters using real vibration data". In: *Smart Materials and Structures* 22.7 (2013), p. 075022.

[31] Dhananjay Jagtap and Pat Pannuto. "Reliable Energy Sources as a Foundation for Reliable Intermittent Systems". In: *Proceedings of the 8th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*. 2020, pp. 22–28.

[32] Arwa Alsubhi et al. "Can Crystal Oscillators Keep Time Without Power?" In: *Proceedings of the 8th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*. 2020, pp. 84–85.

[33] Selahattin Kosunalp. "A New Energy Prediction Algorithm for Energy-Harvesting Wireless Sensor Networks With Q-Learning". In: *IEEE Access* 4 (2016), pp. 5755–5763. DOI: 10.1109/ACCESS.2016.2606541.

[34] Faisal Ahmed et al. "Adaptive LINE-P: An Adaptive Linear Energy Prediction Model for Wireless Sensor Network Nodes". In: *Sensors* 18.4 (2018). ISSN: 1424-8220. DOI: 10.3390/s18041105.

[35] Daeyong Kim et al. "Ray Tracing-based Light Energy Prediction for Indoor Batteryless Sensors". In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5.1 (2021), pp. 1–27.

[36] *MSP430FR5994 LaunchPad™ Development Kit*. SLAU678B. Texas Instruments. Sept. 2019.

[37] *CC1352R SimpleLink™ High-Performance Multi-Band Wireless MCU*. swau127a. Texas Instruments. Jan. 2018.

[38] *P2110B – 915 MHz RF Powerharvester Receiver*. Powercast. Dec. 2016.

[39] Vishal Deep et al. "Experimental Study of Lifecycle Management Protocols forBatteryless Intermittent Communication". In: *2021 The 18th IEEE International Conference on Mobile Ad-Hoc and Smart Systems*. 2021.

[40] *TX91501B – 915 MHz Powercaster Transmitter*. Powercast. Oct. 2018.