HARC: A Heterogeneous Array of Redundant Persistent Clocks for Batteryless, Intermittently-Powered Systems

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

***Abstract*—Batteryless sensing devices powered solely by ambient energy sources are expected to operate in an *intermittent* manner, since they do not have a predictable, or even continuous, energy supply. When such an intermittent system is powered off, it cannot keep track of time using conventional means. However, a continuous sense of time is critical for any system running real-time or time-sensitive applications. In this paper, we present HARC (Heterogeneous Array of Redundant Persistent Clocks), a novel solution to the problem of timekeeping for batteryless, intermittently-powered systems. HARC uses a heterogeneous, redundant array of capacitor-based persistent clocks that each decay in parallel, but at different rates, to provide variation-resilient high accuracy over a wide range of power off-times. We demonstrate the feasibility and effectiveness of HARC using experimental evaluations on a HARC prototype, and trace-based simulations of HARC-supported communication directly between two devices intermittently-powered by RF harvesting.**

***Index Terms*—Batteryless Intermittent Devices, Energy Harvesting, Persistent Clocks, Hardware-Software Integration, System Prototype, Lifecycle Management Protocol**

I. Introduction

Batteryless sensing devices, which are powered *solely* by ambient energy sources such as radio frequency (RF) energy, solar, thermal and vibrations [1], have received more and more attention in the era of the Internet of Things (IoT) due to several advantages over conventional battery-powered sensing devices. For instance, batteryless devices could operate much longer than battery-powered devices, whose lifetime is limited to that of the battery; batteryless devices could be deployed in IoT applications where replacing batteries is not necessarily possible; and batteryless devices eliminate the waste of non-biodegradable batteries. Batteryless devices have the potential to enable a plethora of new sensing applications, including wearable technologies [2], water monitoring through microbial energy harvesting [3], environmental monitoring [4], greenhouse monitoring, and more.

However, batteryless devices come with significant challenges. Primarily, most batteryless devices do not have a predictable, or even continuous, energy supply. Instead, batteryless devices typically store harvested energy in a capacitor or an array of capacitors [5] until sufficient energy is available to turn on the system (i.e., the capacitor's voltage reaches an "on threshold"). The system then performs tasks such as sensing, computing, and/or communication, until the capacitor reaches a voltage that can no longer support operation (i.e., its "off threshold"), and the system "dies." Then, the harvested energy recharges the capacitor to the on threshold, and the *lifecycle* repeats. This type of lifecycled device operation, depicted in Figure 1, is what makes these devices *intermittent*.[1]

Intermittent operation creates many practical problems. We focus on the challenge of maintaining a continuous sense of time, which is important for any real-time or time-sensitive applications, such as time-stamping of sensed data, synchronization, and communication between devices [13]. Timekeeping is challenging for an intermittent system for two reasons. First, when an intermittent system dies, it loses all volatile state, including the current clock value. Second, when the device is not powered (during the *off-time*), there is no active clock that can track time. The first problem can be resolved with system checkpointing [14], but the second is more challenging. Recent works have addressed the problem by using known rates of capacitor discharge or data remanence to keep time during off-times [15]. However, the state-of-the-art solutions are not tolerant to real-world variations encountered by deployments of intermittent systems.

In this paper, we present a novel solution to the problem of timekeeping for batteryless, energy-harvesting, intermittently-powered systems. It uses a parallel hardware architecture of persistent clocks that provides accuracy proportional to off-time over a wide range of off-times. The contributions of this paper are as follows:

- We propose a class of persistent clocks, called HARC, that are based on a heterogeneous, redundant array of capacitor-based persistent clocks that each decays in parallel but at a different rate to provide variation-resilient high accuracy over a wide range of power off-times.

[1]The intermittent systems we target are different from duty-cycled systems. Although systems with larger energy stores (e.g., many battery-powered systems) may "intermittently" duty-cycle microcontrollers and radios, they often allow some components such as real-time clocks, retention SRAMs, and interrupt circuits to be continuously powered. In the batteryless systems we target, none of these components are powered after the energy storage capacitor is depleted. During this period the storage capacitor is recharged from the extremely limited power of an energy harvester such as an RF harvester. This is what we refer to as an intermittent system [6]–[9] , which is sometimes also known as a transiently-powered system [10]–[12].

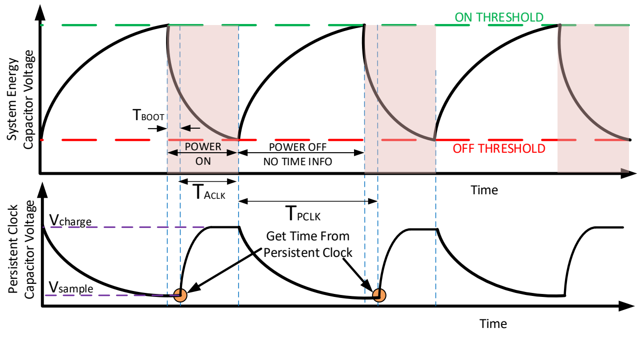

ON THRESHOLD
System Energy Capacitor Voltage
T_BOOT
POWER ON
POWER OFF NO TIME INFO
OFF THRESHOLD
Time
T_ACLK
T_PCLK
Get Time From Persistent Clock
Persistent Clock Capacitor Voltage
Vcharge
Vsample
Time

Fig. 1. Typical operation of an intermittent batteryless system with persistent clock, where T_{BOOT} is the system bootup time, T_{ACLK} the time period measured by an active clock when the system is on, and T_{PCLK} the time period measured by a persistent clock when the system is off.

- We prototype and evaluate several HARC implementations that provide a continuous sense of time in an intermittent batteryless system powered solely by RF harvesting. The highest-accuracy HARC prototype provides a maximum average error of 7.2% across a wide range of power off-times (10 ms to 135 s), while a low-overhead version provide a maximum average error of 58%. Single-capacitor baselines always demonstrate regions with greater than 100% error.
- We propose a *lifecycle management protocol* (LMP) for batteryless intermittent systems powered solely by RF-harvesting. The LMP leverages a continuous sense of time to enable node-to-node communications. We use trace-based simulations to show that the robust accuracy benefits of our HARC prototypes can allow the LMP to provide up to a 54% reduction in median intercommunication delay compared to single-capacitor clocks.

II. Time-keeping in Intermittent Systems

The intermittent nature of the operation of batteryless devices poses several significant challenges. First, since typical intermittent systems contain a mix of volatile memory that loses state when the system is off (e.g., registers and SRAM), and non-volatile memory that retains its state when the system is off (e.g., FRAM and NAND Flash), there is a challenge of making reliable forward progress in the application. Solutions to this either perform a form of careful checkpointing [10], [11], [16], [17], or are task-based, where each task appears to complete atomically [8], [18]–[20]. Second, setting up volatile state in I/O peripherals such as radios and sensors takes a significant fraction of the limited on-time (typical on-times observed are milliseconds to seconds). Approaches to solve this include the design of a middleware to retain the state of peripherals across the power failures [21], reliance on a kernel support to make atomic peripheral accesses and interrupt handling over power failures [12], and usage of energy profiling to make sure of completion of an atomic function which may include peripheral operations [22]. Third, many applications (e.g., communication protocols, staleness of sensed data, and security) require maintenance of a sense of time while the system is powered off.

In order to determine how much time has passed, a device can maintain an active clock (i.e., a real-time clock or RTC) with its own reserve capacitor store, or it can measure residual charge on a component (e.g., capacitor) where the rate of decay is predictable (i.e., a persistent clock). Unfortunately, neither of these approaches provides high precision and accuracy across large and variable off-time periods, which will be discussed in the following sections.

A. Low-power Real-Time Clock (RTC)

One approach to measuring off-time is to use an RTC powered by its own small capacitor energy store. During the off-time when the rest of the system is unpowered, the RTC runs off of the reserve capacitor and tracks time. Once the energy harvester can harvest enough energy to fill the main system storage capacitor and turn on the full system, the RTC's reserve capacitor can be refilled in preparation for the next off-time. If the reserve capacitor's voltage falls below the RTC's minimum operating voltage, all timing information is lost and the oscillator ceases to function and must be restarted during the next (potentially too short) on-time. Additionally, while current commercial RTCs can provide high accuracy in time measurement, this comes with fixed power overheads—the power to maintain a *stable* oscillation, and the time and energy required to stabilize upon start-up. The batteryless intermittent systems we target have extreme energy constraints due to the limited and variable amount of energy that can be harvested. For these systems, having some sense of time throughout their entire deployment time is more important than always having a relatively high accuracy time at a fixed energy cost. Therefore, we investigate a time-keeping mechanism that allows the energy overhead to gracefully scale with accuracy (even dynamically in response to observed conditions – see Section V-E) as well as one that can be rapidly restarted.

B. Persistent Clocks

Recent works present an alternative approach to measure off-time. In general, the idea is to use a known decay of charge on capacitance within the intermittent system to estimate how long the system has been powered off. As shown in the bottom plot of Figure 1, during an on-period, a clock capacitance of known value, C, is charged to a known voltage, V_{charge}. When the system powers off, the capacitor begins to discharge over a fixed resistance, R. This discharging continues until the system powers on again, at which point the current voltage on the capacitor (V_{sample}) is measured and the clock capacitor is recharged. Since the time constant (RC) of the clock is fixed, V_{sample} can be used to estimate how long the clock had been discharging, and thus how long the intermittent system was powered off (i.e., without an active clock). A clock built using this approach is known as a *persistent clock*.

One example of a persistent clock is Time and Remanence Decay in SRAM (TARDIS) [23], which uses variation in SRAM cell retention voltages to read the voltage value of either an external capacitor or the built-in chip capacitance. While this has low hardware overhead, it has relatively poor accuracy [15]. Custom Time and Remanence Decay (cusTARD) [15] uses a capacitor connected to the analogue to digital converter (ADC) of the microcontroller on the intermittent system. This provides improved accuracy at the cost of more hardware. Unfortunately, the maximum off-time of such clocks is fixed once the RC constant is selected. Since errors in the time estimate increase as the discharge rate decreases (i.e., as the maximum duration of the clock increases), selecting a single RC clock that is provisioned for the maximum expected off-time means poor accuracy for shorter off-times. A recent work, Cascaded Hierarchical Remanence Timekeeper (CHRT) [24], developed a timekeeping mechanism that provisions an intermittent system with multiple clocks with different RC constants that decays sequentially. While this approach can provide high accuracy (relative to the off-time) over a wide range of off-times, it does not guard against the variation that can occur in the charging, discharging, and measurement of a single persistent clock. The goal of our work is to provide intermittent systems with a continuous sense of time that is highly accurate over a wide range of *unknown* and *variable* off-times, despite variations within the persistent clocks themselves.

III. Observations on Persistent Clocks

Our approach to improving the accuracy and reliability of persistent clocks is based on three observations about the source of variations in persistent clocks and the intermittent systems that use them.

A. Observation #1: Off-time Varies Significantly

A batteryless intermittent system's off-time varies both with the rate of energy harvesting and the amount of stored energy remaining on the capacitor when the system dies. For example, the rate of energy harvesting from an RF source such as the Powercast TX91501b Powercaster [25] (PTX) by a harvester such as the Powercast P2110B Powerharvester [26] (PRX) can vary with distance between the PTX-PRX pair, as well as on their relative reflection, diffraction, interference, and scattering [27]. Such variations may occur based on system deployment, changes in the environment around the system, or system motion.

To illustrate this variation, Figure 2 shows the distribution of off-times observed on an intermittent system comprised of a single PRX with a 50 mF capacitor powering a TI SensorTag [28] running Contiki OS [29]. The intermittent system is statically deployed at a typical university student residence, with the SensorTag deployed at different distances, r, from the PTX. Generally, as r increases, so does the off-time, since the theoretical rate of energy transmission falls off proportional to $\frac{1}{r^2}$. However, in this experiment, the average off-time at 1.5 m is 113.05 s, which is unexpectedly larger than the off-time at 2 m (71.29 s), and the 1.5 m variance is also higher. This represents just a taste of the unpredictability a deployed system may encounter—and these results are on

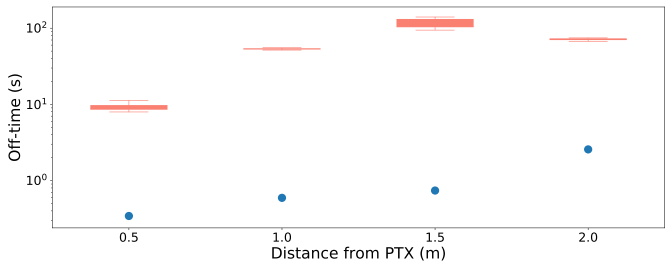


Fig. 2. Observed off-time versus PRX distance from the PTX over 100 lifecycles. The box plots show off-time corresponding to maximum on-time; the blue dots show off-time corresponding to minimum on-time, when the device turns itself off shortly after booting.

a static system. A deployment where an intermittent system moves (even slightly due to the environment) will experience even more unpredictable variation in off-times.

Finally, when a task completes or the harvesting device decides to conserve energy for a future task, the device may "die" early.[2] This further expands the range of off-times experienced by the device. To demonstrate, the blue dots in Figure 2 show the (observed) minimum off-time at each distance, obtained by having the SensorTag device turn itself off shortly after it boots. A device capable of controlling itself in this way could therefore be expected to see a wide, continuous range of off-times.

Insight #1: The large variation and uncertainty in the off-time of a deployed intermittent system means that a persistent clock must be able to provide accurate timing information across a wide (orders of magnitude) range of off-times, without prior knowledge of the off-time value.

B. Observation #2: Precision is Inversely Proportional to Maximum Duration

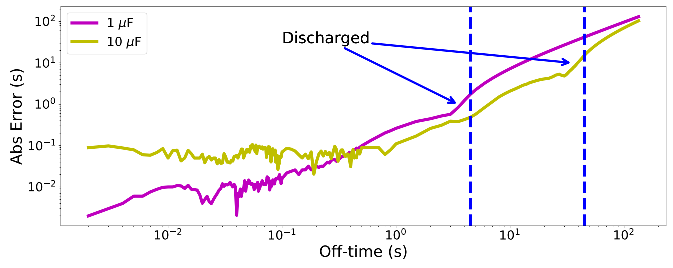
The precision of a single-capacitor clock (e.g. [15]) decreases as the maximum timekeeping duration of the clock increases. This occurs since a persistent clock provides timing by mapping a voltage value to a time estimate using a known rate of decay (i.e., $\frac{dV}{dt}$ is known across the persistent clock's entire decay range). The voltage value is sampled by an ADC, converting it to a discrete value. Therefore, the larger the rate of decay, the smaller the time value differences between two adjacent discretized values, and the more precise the time estimate can be. This also means that any noise caused by variations in the clock decay or sampling of the capacitor have a smaller impact on the resulting time estimate.

Figure 3 shows the absolute error[3] of two individual persistent clocks with different capacitor sizes. For small off-times (e.g., below 10 ms), the 1 μF clock has over an order of magnitude less absolute error than the 10 μF clock. In fact, the 10 μF clock has more than 100% relative error[4] in this region, meaning it encodes very little useful timing

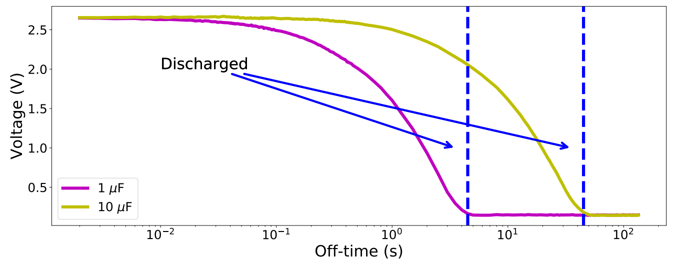
[2]The Powercast P2110B harvester has a RESET input which can be used to disable the regulated power supply.

[3]Absolute Error = abs(ground-truth off-time (T) - estimated off-time (T^*))

[4]Relative Error = Absolute Error/ground-truth off-time (T)



(a) Absolute error vs. off-time.



(b) Voltage vs. off-time.

Fig. 3. Comparison of persistent clocks with different capacitor sizes. The two clocks estimate time individually, and data is averaged over 10 samples. The dashed blue lines indicate when each clock is fully discharged, i.e., the maximum timekeeping duration of the clocks.

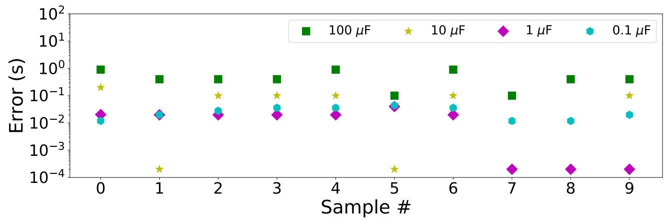
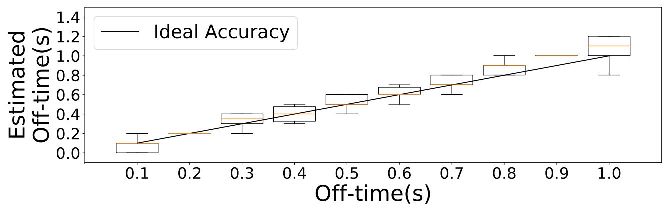
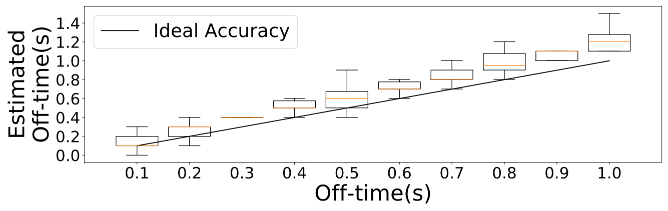


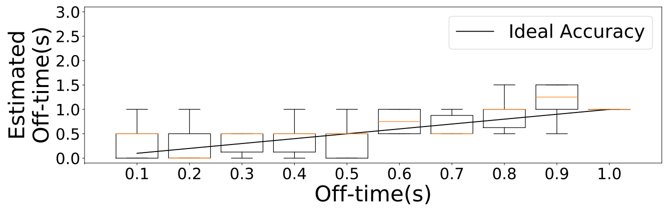
Fig. 4. Per-sample variation at 0.1 s off-time.



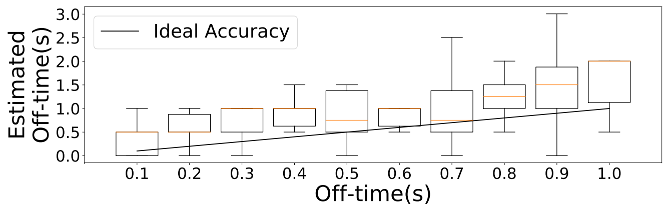
(a) 10 μF capacitor for Day 1.



(b) 10 μF capacitor for Day 2.



(c) 100 μF capacitor for Day 1.



(d) 100 μF capacitor for Day 2.

Fig. 5. Systemic persistent clock variation for 10 μF and 100 μF capacitors for test datasets taken on two separate days.

information. The relative error could exceed 100% if the clock estimate is larger than the ground-truth off-time; after reaching the discharge point, relative error is smaller than but will asymptotically approach 100% when the ground-truth off-time is larger than the frozen clock estimate. At an off-time of 4.5 s, the 1 μF clock's decay rate is so flat that it provides no timing information—it is completely decayed. Beyond this point, its time estimate does not increase, and, as the off-time increases, its relative error approaches 100%, indicating it provides almost no timing information. Although the 1 μF clock reaches its maximum off-time at 4.5 s, the 10 μF clock can still provide a timing estimate with an accuracy that is proportional to the off-time until its maximum off-time is reached at 45 s. Neither clock can provide an accurate, meaningful off-time estimate for the entire range between 2 ms and 135 s, although at least one clock can always provide an accurate time estimate up to 45 s, i.e., the discharge time of the 10 uF clock.

Insight #2: To accurately cover a wide and unknown range of off-times, multiple persistent clocks with heterogeneous decay rates must be used *in parallel*.

C. Observation #3: Clock Measurements Vary Significantly between Samples

Single-capacitor persistent clocks produce *local* variation in their time estimates due to noise in the decay rate and sampling process. Figure 4 shows a series of persistent clock samples taken for a fixed off-time using the same experimental setup as before. Each sample consists of voltage readings and corresponding time estimates from four clocks (0.1, 1, 10, and 100 μF capacitor sizes) that have decayed in parallel and are read sequentially by one ADC. At 0.1 s, the 1 μF clock has the best average accuracy across all sample points, although in samples 1 and 5, the barely-decaying 10 μF clock has significantly higher accuracy than the 1 μF clock. Additionally, the nearly-decayed 0.1 μF clock has better accuracy than the 1 μF clock for sample 0. From this, we observe that the redundant, less-accurate clocks for a given off-time can still retain some latent timing information, which could be used to correct for local variation.

Variations can also happen on a more systemic level, but crucially, the systemic variation affects the heterogeneous clocks differently. For example, Figure 5 shows the distribution of estimated off-time relative to actual off-time for both 10 μF and 100 μF clocks for two different days of data collected using the same deployment as before. The "ideal accuracy" line represents a zero-error reading (i.e., estimated off-time equals ground-truth off-time). On Day 1, both the 10 μF and 100 μF clocks had distributions whose averages were centered around the actual off-time. Then, on Day 2, both the 10 μF and the 100 μF clocks had distribution averages that were higher than the actual off-time—a systemic variation.

However, the systemic variation results in a significantly larger increase in error for the 100 μF clock than the 10 μF clock. This type of heterogeneous behavior in the presence of systemic variations may allow a statistical technique to mitigate the error.

Insight #3: In order to provide resiliency to local variations, multiple *redundant* clocks need to be used. In order to provide resiliency to systemic variations, multiple *heterogeneous* clocks (with different decay rates) can be used.

Given these insights on variations in off-time, precision vs. off-time, and single-capacitor clock estimate variation, we aim to develop a class of persistent clocks, based on a heterogeneous, redundant array of capacitors, that provides variation-resilient high accuracy over a wide range of off-times. Details will be presented in the next section.

IV. Heterogeneous Array of Redundant Persistent Clocks (HARC)

A Heterogeneous Array of Redundant Persistent Clocks (HARC) requires a combination of hardware and software to estimate time in an intermittent system. Figure 6 shows the essential components of a HARC and the process of generating a time estimate. The HARC hardware consists of a heterogeneous array of capacitors, each decaying simultaneously, thereby making them redundant. A microcontroller unit (MCU) with on-board ADC, timer, comparator, and non-volatile memory is used to interface with the capacitor array and run the HARC software. The HARC software consists of a set of pre-calibrated voltage-to-time mapping tables corresponding to each persistent clock in the array, and a program to produce a final time estimate based on the outputs of the mapping tables. As shown in Figure 6, HARC operates with the following steps:

1) When the system powers on after some off-time of length T, the MCU uses its ADC to read the individual voltages across all p capacitor clocks in the heterogeneous array of the HARC hardware.
2) The MCU uses the corresponding voltage-to-time mapping tables for each capacitor clock to generate p time estimates.
3) HARC uses one of the estimate fusion techniques described in the Section IV-B to combine p time estimates to produce a final time estimate T^*.
4) To maintain a continuous sense of time, the MCU's on-board timer can be used to keep track of the on-time, and a comparator can be used to detect a power failure and checkpoint the last on-time before dying.
5) The MCU re-charges the array of clock capacitors in preparation for the next off-time.

The following sections describe HARC hardware and HARC software in detail.

A. HARC Hardware

As shown in Figure 6, each clock circuit of the HARC hardware consists of a charge resistor R_{charge} followed by a Schottky diode, a capacitor, and a large discharge resistor $R_{discharge}$. Each capacitor clock's charge resistor is connected to a GPIO port of the microcontroller, while its discharge path is connected to an I/O port of the microcontroller's ADC. Internally, these I/O ports are multiplexed to the ADC, allowing a single ADC to read all p clocks in rapid succession.[5]

The selection of RC constants for a HARC is critical to provide high-accuracy coverage of the full range of desired off-times. Consider the design of a HARC array targeting the off-time range shown in Figure 7. Each candidate capacitor clock has a different time constant, $\tau_n = RC_n, n \in 1, \ldots, p$. To simplify the presentation, Figure 7 shows an example scenario with $p = 3$ clocks. Minimally, a HARC needs to have at least one clock discharging at a sufficient rate to provide an accurate time estimate, with error proportional to the off-time, across the entire range of off-times. That is, for every off-time T_i, there must be at least one clock n such that $\frac{\Delta V_{n,i}}{\Delta t} < \frac{\phi}{T_i}$, where ϕ is the target accuracy.

For example, let's consider $T_{\min}$, the smallest off-time in Figure 7. At this off-time, τ_1 has the steepest decay rate and, thus, should have the best accuracy. In fact, it is possible that no other candidate clock meets the target accuracy, meaning that τ_1 should be part of the HARC design. Likewise, for the largest off-time, $T_{\max}$, τ_3 has the steepest decay rate, since all other clocks have fully decayed and thus provide very little timing information. Therefore, to achieve accurate time estimates at the upper end of the off-time range, τ_3 should be part of the HARC design. At intermediate off-times, there may be multiple clocks that meet the target accuracy. For example, at T_i, both τ_2 and τ_3 meet the accuracy target, while τ_1 does not.

Including τ_2 in the HARC design can provide additional accuracy benefits, not only at T_i where it has the highest accuracy (because its discharge curve is the steepest), but also at $T_{\max}$, where it has some residual timing information that

[5]Other HARC charge and discharge architectures are possible, including ones with external switches and ADCs, single common charge GPIO, or even an on-chip capacitor array, depending on the overall intermittent system design constraints. We chose this particular architecture since each channel has a separate charge path and discharge path, which allows software to dynamically select the clocks to be charged and read. Dynamic selection of clocks is orthogonal to the core idea of HARC and is beyond the scope of this paper.

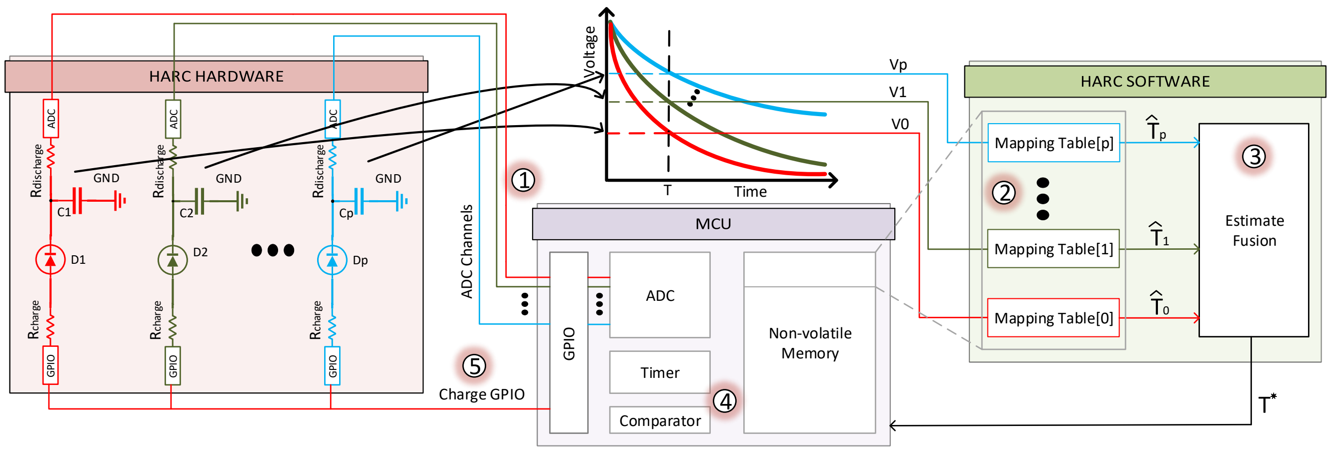


Fig. 6. HARC overview.

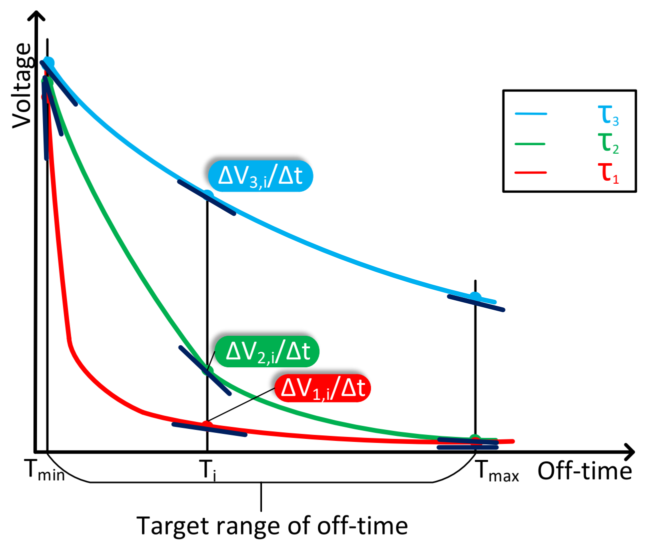


Fig. 7. Example HARC candidate clock discharge curves with $p = 3$. HARC selects heterogeneous time constants to ensure at least one discharge curve has a sufficiently steep discharge slope relative to a particular off-time.

can be leveraged by the HARC software to correct for noise in other clocks.

B. HARC Software

The HARC software uses the array of voltages coming from the capacitors of HARC hardware to provide an accurate estimated power-off time. This is accomplished in two main steps: voltage translation, and estimate fusion.

The first step, *voltage translation*, is the process of turning the array of capacitor voltages into an array of single-capacitor persistent clock time estimates. We perform this translation using voltage-to-time mapping tables. To create the mapping tables, we first log the HARC capacitor's voltage at selected known off-times in the target off-time range over multiple iterations. Then we remove the entries of each clock below its discharge point. Due to inherent variations in hardware components, we generate these mapping tables in a calibration process prior to deployment. During calibration, the voltage on each capacitor of the HARC is sampled multiple times for each off-time in a set of off-times that covers the target range for the application. For each capacitor, we calculate the mean of the voltage at each off-time and store it in a voltage-to-time mapping table, which is loaded onto the device as static data.

At runtime, the voltage value is used as the index to look up the table and retrieve the estimated time value. For voltages between entries in the mapping table, the time value is estimated by linear interpolation.

The second step, *estimate fusion*, takes the array of estimated times from the component clocks and produces a final time estimate. In this step, the heterogeneity and redundancy of HARC can be leveraged to produce high-accuracy, variation-resilient time estimates for a wide range of off-times. We propose three approaches to generate the final time estimate: *naive*, *reg*, and *lite*, each with a different accuracy, resiliency, and runtime overhead.

1) HARC-naive: HARC-naive is a straightforward implementation of estimate fusion for HARC, presented here as a straw-man proposal. In HARC-naive, the array of component clock off-time estimates is simply averaged to produce the final time estimate. By averaging the component time estimates, HARC-naive provides some resilience to variation within a sample, and still allows some measurement of time across the entire range of off-times. However, at small off-times, HARC-naive is dominated by the larger, less-accurate clocks, while at large off-times, HARC-naive is biased toward shorter estimates by including the smaller clocks that have already fully decayed.

2) HARC-reg: To deal with the wide range of off-times and variations in samples, HARC-reg divides the target off-time range into multiple smaller, non-overlapping sub-ranges. We use ℓ to denote the number of sub-ranges, which is set to 12 in our prototype. At each sub-range, HARC-reg first applies feature scaling [30] to the component clock time estimates to ensure that the variations between component clocks do not mislead the predictions. Then, HARC-reg uses a multidimensional linear regression to combine these time estimates in a robust manner. Since fully-decayed clocks are no longer contributing meaningful time information, we employ a LASSO regression technique, which can effectively select the contributing features and zero out the coefficients for non-contributing features (i.e., LASSO is a penalizing regression) [31]. A lasso regression is mainly used to train our model, and the trained coefficient matrix and regression intercepts are used to produce a time estimate T^* for that sub-range.

HARC-reg also needs to decide which sub-range a particular sample shall be mapped into. We use a multi-class Support Vector Machine (SVM) [32] based on a One vs One algorithm [33], for this purpose. Specifically, a binary SVM classification is performed between each pair of sub-ranges. The resulting binary classifications are used to vote for which sub-range to use. With this technique, we were able to predict the correct sub-range or an adjacent sub-range (which yields a similar estimate) with an accuracy of around 85-95%. Algorithm 1 summarizes the HARC-reg algorithm.

3) HARC-lite: Although HARC-reg can leverage heterogeneity and redundancy to provide high accuracy despite noisy clocks samples, its software robustness comes at the cost of run-time and energy overhead. For intermittent systems that have very short on-times, this overhead may be too high. We therefore propose HARC-lite, a lightweight and fast version of estimate fusion that aims to provide accuracy on par with the best single persistent clock for any given off-time. The design for HARC-lite (shown in Algorithm 2) is based on the insight that the most accurate time estimate from a capacitor clock occurs at the steepest voltage discharge slope ($\frac{dV}{dt}$) for the capacitor, as seen in Section III-B. Since the mapping table interpolation process already needs to calculate the slope about the sample point, the capacitor clock with the steepest discharge slope at the time of sampling can be determined with very little run-time overhead. HARC-lite returns this clock's time estimate as the final result.

Algorithm 1: HARC-reg

```
Input: $\hat{\mathbf{T}} = \hat{T}_1, ..., \hat{T}_p$, component clock time estimates
       $\boldsymbol{\gamma^0}$, $\frac{\ell(\ell-1)}{2}$ binary SVM intercepts vector
       $\boldsymbol{\gamma}$, $\frac{\ell(\ell-1)}{2}$ by $p$ binary SVM coefficients matrix
       $\boldsymbol{\beta^0}$, $\ell$ sub-range regression intercepts vector
       $\boldsymbol{\beta}$, $\ell$ by $p$ sub-range regression coefficients matrix
       $\boldsymbol{\sigma}$, $\ell$ by $p$ scale matrix
Output: $T^*$, final time estimate
Classifications $\mathbf{C} \leftarrow \hat{\mathbf{T}}\boldsymbol{\gamma} + \boldsymbol{\gamma^0}$ // Apply binary SVMs
// Convert binary SVM confidences to votes
Decision matrix, $\mathbf{D} \leftarrow d(\mathbf{C} < 0, -(\mathbf{C}))$
$s \leftarrow \underset{s \in [1,l]}{\text{argmin}}(\mathbf{D})$ // Identify sub-range with most votes
$\hat{\mathbf{T}}_{\mathbf{scale}} \leftarrow \boldsymbol{\sigma_s}\hat{\mathbf{T}}$ // Scale time estimate feature vector
$T^* \leftarrow (\hat{\mathbf{T}}_{\mathbf{scale}}\boldsymbol{\beta_s} + \boldsymbol{\beta^0_s})$ // Apply LASSO regression
```

Algorithm 2: HARC-lite

```
Input: $\hat{\mathbf{T}} = \hat{T}_0, ..., \hat{T}_p$, component clock time estimates
       $\mathbf{V} = V_0, ..., V_p$, component clock measured voltages
       $\mathbf{M} = M_0, ..., M_p$, component clock mapping tables
Output: $T^*$, final time estimate
$\boldsymbol{\alpha} \leftarrow \mathbf{M}.getMapTableSlope(\mathbf{V})$
$k \leftarrow \underset{k \in [1,p]}{\text{argmin}}(\alpha_k)$
$T^* \leftarrow \hat{T}_k$
```

V. Prototype Evaluation

To evaluate HARC, we implemented a prototype. Using the prototype we evaluated run-time, energy, and memory overheads as well as the accuracy across a range of off-times. Additionally, we reported the time estimation stability of HARC, and we showed that HARC can provide an energy versus accuracy trade-off.

A. Experimental Methodology

Our prototype HARC (pictured in Figure 8) consists of six component clocks with capacitors listed in Table II, selected to cover our observed range of off-times from Figure 2. These clocks are connected to the on-board ADC channels of a TI MSP430FR5994 [34] which runs the HARC software. The GPIO pins used to charge each capacitor are independently programmable and are pulled down externally to avoid unwanted harmonics and to compensate for incidental charging of the capacitors due to the high impedance state of the GPIO pins. We used class-1 and class-2 multilayer ceramic capacitors (MLCCs) as the clock capacitors due to their low leakage current and low cost. All the components used in the system are off-the-shelf.

As shown in Figure 8, for calibration and evaluation of our HARC prototype, we emulated an intermittent power supply using a USB-powered MSP430FR5994 LaunchPad [35] that controls the power of the HARC prototype via a MOSFET driver [36] powered by an external 3.3V power supply. The HARC prototype estimates each corresponding off-time and reports it to a host computer via UART.

We collected data in a climate-controlled university residence. For each dataset, the power controller swept through off-times from 10 ms to 135 s. Separate training and test

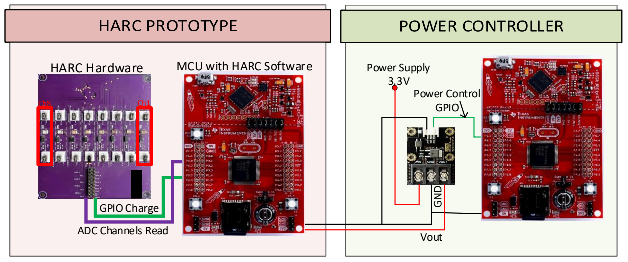


Fig. 8. Experiment setup for the evaluation of HARC prototype.

datasets were collected weeks apart. The training dataset was used to calibrate component persistent clocks (i.e., generate mapping tables) and to train HARC-reg's classifier and regression model. The testing set was used to perform accuracy evaluations.

B. HARC Overheads

Given the extremely small on-times and limited resources (e.g., non-volatile memory) that characterize batteryless intermittent systems, it is important to understand the overheads a persistent clock has at runtime.

A HARC incurs a run-time overheads from reading the voltage of clock capacitors using an ADC (*Read ADC channels*), calculating an estimate of the previous off-time (*HARC-reg* and *HARC-lite*), initializing an active clock with a previously checkpointed active clock and the off-time estimation (*Active time restore*), and recharging the component capacitors in anticipation of the next off-time. Table I shows the run-time and energy consumption of HARC as calculated from TI Code Composer Studio (CCS) using EnergyTrace [37].

TABLE I
HARC SOFTWARE RUN-TIME LATENCY AND ENERGY CONSUMPTION.

Function		Latency	Energy
Read ADC channels		0.665 μs	5.331 nJ
On-time restore		0.104 ms	0.586 μJ
HARC-reg	SVM Inference	30 ms	108.4 μJ
	Regression	80 μs	0.34235 μJ
HARC-lite		8.174 ms	48.95 μJ
Single-cap Clock		1.68 ms	9.435 μJ

HARC latency is dominated by the off-time estimation computation. HARC-reg's off-time estimation latency is 30.08 ms comprising 30 ms for SVM inference and 80 μs for regression. HARC-lite reduces off-time estimation latency by 73% to 8.174 ms. When added to the other time-keeping overheads, the HARC prototypes require 8.28 ms to 30.18 ms. This amounts to only 1.56% to 5.69% of the 530 ms maximum on-time observed when using a 50 mF system energy capacitor, which is charged by the Powercast Harvester and provides energy to *all* of the device's tasks – sensing, computation, communication, idling, and time-keeping.

HARC's energy overhead comes both from the above computation and from the charging of the clock capacitors. The Table II shows the energy consumption of each capacitor in the HARC prototypes to charge if they are completely discharged. In total the HARC prototypes consume a maximum of 2.38 mJ. This amounts to 18.56% of the 12.823 mJ needed to charge the 50 mF capacitor on PRX from 1 V to 1.23 V (its on and off thresholds). In typical operation, the largest clock capacitors are unlikely to fully discharge and require the full energy overhead.

The charging time of the largest component clock (4.5 ms) sets the minimum amount of on-time after the microcontroller boots and reads the ADC that must be met in order to

TABLE II
CHARGING TIME AND ENERGY CONSUMPTION OF INDIVIDUAL PERSISTENT CLOCK AND HARC.

Capacitor	Charging Time	Charging Energy
100 μF	4.5 ms	1.60 mJ
47 μF	2.7 ms	0.686 mJ
10 μF	1.1 ms	86.078 μJ
1 μF	125 μs	6.90 μJ
0.1 μF	10.8 μs	0.884 μJ
0.01 μF	10.4 μs	0.0972 μJ
HARC Total	4.5 ms	2.38 mJ

maintain a HARC at full accuracy[6]. Our HARC has orders of magnitude smaller worst-case start times compared with RTCs as described in Section II.

The major non-volatile memory overhead in the HARC prototypes comes from the component clock mapping tables. Table III shows the mapping table memory overheads for the six single-capacitor persistent clocks used by the prototype. Both HARCs require these mapping tables to be stored. HARC-reg additionally must store classification and regression coefficients. The total non-volatile memory overhead of HARC-reg is 8.512 KB or 3.3% of the MSP430's total non-volatile memory.

TABLE III
MEMORY FOOTPRINT OF INDIVIDUAL PERSISTENT CLOCK AND HARC.

Clock Version	# Entries	Memory (Bytes)
100 μF	558	2232
47 μF	558	2232
10 μF	558	2232
1 μF	276	1104
0.1 μF	100	400
0.01 μF	78	312
HARC-reg Coefficients	618	2489
HARC-reg Total	2746	11001
HARC-lite Total	2128	8512

C. HARC Accuracy

We evaluate the accuracy of our HARC prototype, relative to single capacitor clocks, using HARC-naive, HARC-reg, and HARC-lite. Figure 9 shows the average error (absolute value of the difference between ground-truth off-time and the clock time estimate) across the entire range of evaluated off-times.

As expected, the single capacitor persistent clocks provide meaningful time information in a narrow region. For example, the 10 μF clock has roughly 100% error under 80 ms, when it has not appreciably discharged, and again after 45 s, when it has completely discharged. It is most accurate between 200 ms to 4 s where it has an average error near 10%. While each single-capacitor clock may have a region over which it is the most accurate clock, no single-capacitor clock provides

[6]Note a partially charged HARC may still be able to produce some meaningful timing information. Such robustness evaluations are the subject of future work.

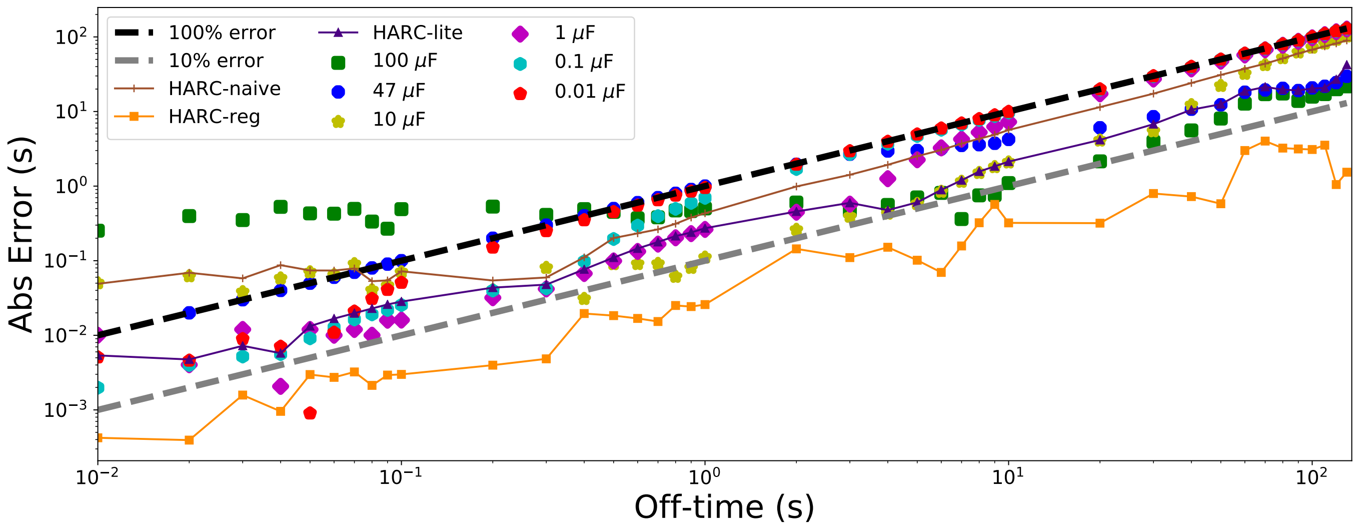


Fig. 9. Comparison of accuracy of HARC-naive, HARC-reg, and HARC-lite with single-capacitor persistent clocks.

less than 100% error across the entire range of off-times. Additionally, even in their most accurate off-time ranges, no single-capacitor clock can sustain an average error less than 10% (and in our wireless communications application in Section VI, performance rapidly improves when clock error is reduced below 10%).

In comparison, HARC-reg achieves sub-10% error across the entire range of off-times from 10 ms to 135 s, thanks to its use of heterogeneous capacitor values. Furthermore, at any given off-time, it achieves better accuracy than any individual component clock, thanks to its use of redundant capacitor clocks to provide resiliency to the measurement noise inherent to capacitor clocks.

HARC-lite can also meaningfully measure time across all off-time ranges, achieving average error from 8.5% to 59%. It does so by leveraging the heterogeneity in component clocks and attempting to select the best clock given their time measurements. Although it provides reasonable time estimates across the entire time range, the variations in samples can cause the heuristic to sometimes select a less-accurate clock, and HARC-lite does not leverage redundancy to correct this. Therefore, HARC-lite generally has a higher average error than the best single clock for any given off-time.

As expected, HARC-naive is the worst of the estimate fusion methods, never achieving higher accuracy than HARC-lite. While HARC-naive does use both the redundancy and heterogeneity of component clocks, it does so blindly. At small off-times (less than 100 ms), large clocks with very low precision dominate the average, causing large errors (i.e., greater than 100%). At larger off-times (greater than 1 s), the weight of multiple small, decayed clocks drives HARC-naive toward 100% error. However, over the range of off-times, HARC-naive is always more accurate than one or more single capacitor clocks.

D. *Time Estimate Stability*

The stability in estimating time across a wide range of off-times is an important factor to determine the reliability of clocks for intermittent systems. Traditional oscillating clocks, such as RTCs, use Allan Deviation to measure frequency stability. Since HARC does not use an oscillator, the conventional evaluation of Allan deviation (ADEV) can not be directly applied (i.e., there is no notion of free-running). Instead, we are using the following method that could provide an equivalent metric which we believe gives a reasonable sense of the stability of clock measurement.

We take N samples of estimated time x_n at a controlled off-time T. The τ is the time difference between each sample reading. Since HARC is not a free-running clock (i.e., it doesn't track time while the HARC's capacitors are charging or when the device is on), we do not include the charging period of the capacitors (otherwise this could be used to artificially increase HARC's stability). Therefore, for the calculation of ADEV, we first split off-time T's samples into N pairs of "sequential" samples where $\tau = T$ and then calculate the Overlapping Allan deviation at an averaging time of $\tau = m\tau_0$ with the following equation [38]:

$$\sigma^2_{OADEV}(m\tau_0) = \frac{1}{2(m\tau_0)^2(N-2m)} \sum_{n=1}^{N-2m} (x_{n+2m} - 2x_{n+1m} + x_n)^2 \quad (1)$$

where x_n is the estimated time samples measured at τ_0 interval of time, with length N.

The Figure 10 shows ADEV for HARC-lite and HARC-reg vs. power off-time. In general, at smaller off-times the smaller capacitors discharge rapidly, causing a higher ADEV value, while at larger off-times, the bigger capacitors discharge slowly. HARC-lite shows relatively higher deviations across off-times because it uses a single capacitor for time estimation

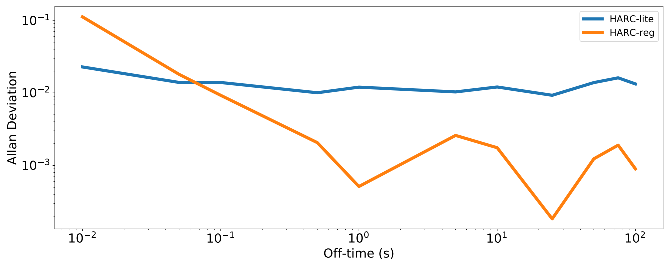


Fig. 10. Allan Deviation (ADEV) across power off-time for HARC.

while HARC-reg uses multiple capacitors with regression techniques at an off-time to estimate time. The lowest ADEV value that we measured for a HARC is 1.84×10^{-4}. The HARC's ADEV may be larger than state-of-the-art oscillators [39]–[42] but it provides an energy-accuracy operating point where the HARC consumes less energy at smaller off-time estimation and more energy at larger off-times.

E. Energy vs. Accuracy Trade-offs

HARC also offers a dynamic accuracy vs energy solution that an intermittent system can leverage to provide a sense of time throughout its entire deployment, even under severely constrained and variable energy-harvesting conditions. HARC can provide energy vs accuracy scalability by dynamically choosing to charge and read only a subset of its total capacitor clocks. Figure 11 shows the energy of HARC due to the underlying capacitor clocks array while using different subsets of the capacitors—a HARC configuration. Each gray line shows the energy-accuracy trade-off for a given off-time. The points within the line are for different configurations of our HARC prototype. Across a given off-time, configurations with fewer capacitors can provide timing information at a lower energy cost (in fact, orders of magnitude less). However, this generally comes at the cost of accuracy. For off-times at the lower end of the off-time range (e.g., 10^{-2}s), configurations with fewer large capacitors (e.g., HARC-reg($1\mu F, 0.01\mu F$) compared with HARC-reg($100\mu F, 1\mu F, 0.1\mu F, 0.01\mu F$) do not give up much accuracy since the small capacitors are actively discharging at small off-times and provide the highest accuracy while active, while the larger capacitors do not contribute much timing information. However, at larger off-times (e.g,. 10^2s) the small clocks are discharged and have nearly 100% error. Under these conditions, the clocks containing the largest capacitors (and thus largest energy overheads) can provide over an order of magnitude lower error in time estimate. In this way a HARC can provide a timing mechanism with an energy vs accuracy setting that is dynamically adjustable after deployment.

In summary, HARCs can provide high-accuracy, variation-resilient off-time estimates across a wide range of off-times at low run-time overheads relative to observed on-times and energies in an RF-powered intermittent system.

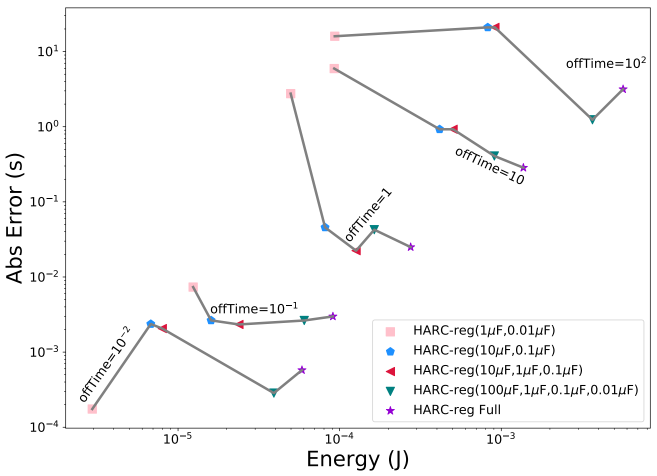


Fig. 11. Energy vs accuracy trade-off with HARC powered by different combinations of capacitor clocks for a dynamic range of off-times.

VI. Example Application: Direct Communication between Intermittent Nodes

In this section, we present an example application for accurate persistent clocks: direct communication between intermittent nodes. We first discuss the challenge, then briefly describe a protocol that uses a persistent clock to achieve regular node-to-node communication. Finally, we present simulation results illustrating the benefit of an accurate persistent clock for this type of communication.

A. Node-to-Node Communication

Communication between intermittent nodes (i.e., wireless sensor nodes that are intermittent systems) using active radios is hard. Both nodes must be powered on at the same time in order to communicate, but in general, intermittent nodes have little control over when they turn on. Assuming an intermittent system such as that described in Section I, the only control a node has over its lifecycle is the ability to "die early," as described in III-A. Dying early shortens the off-time, because the system will not take as long to recharge to the on threshold (i.e. *revive*). Thus, utilizing this control (which is available on the Powercast hardware used in this paper) can result in a wide range of off-times experienced by the node, further motivating the design of HARC.

We propose the concept of a *lifecycle management protocol* (LMP) to leverage a node's control over its lifecycle to enable tasks such as node-to-node communications. When considering LMP designs, a continuous sense of time in the intermittent system quickly becomes a useful and desired feature. Below, we will describe a simple clock-based LMP for node-to-node communications in order to demonstrate the usefulness of an accurate persistent clock, and HARC in particular. We will discuss and evaluate other designs for LMPs in future work.

B. Baseline vs. Clock-based LMP

Figure 12 illustrates a baseline approach to lifecycle management that does not use a persistent clock. In this approach, nodes always remain on until they run out of energy. The baseline relies on random alignment of node on-times (what we call *overlap*) to perform communication. Since the ratio of a node's on-time to its off-time may be very small (much smaller than shown in the figure), the probability of random alignment may be very low with the baseline protocol.

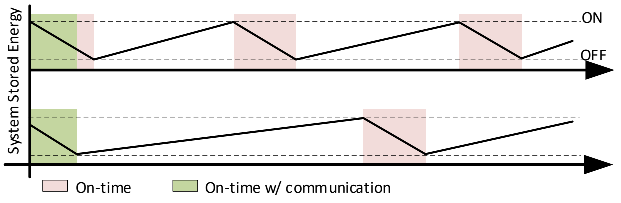


Fig. 12. A baseline lifecycle management protocol that does not use a persistent clock. Every time a node is revived, it attempts communication until it dies again. In this example, the nodes initially are communicating, but are unable to establish communication again.

We therefore propose an LMP that makes use of a persistent clock to roughly "schedule" communication, as shown in Figure 13. This protocol first uses a stateless mechanism, such as the above-described baseline, to establish an initial communication handshake. During this communication, the two nodes exchange estimates of the time of their next revival. This is calculated as an estimate of the node's remaining on-time, plus an estimate of the node's off-time. Using these estimates, the nodes agree on a target *rendezvous time* (t_R), a point in time in the future when both nodes will try to be on to communicate.

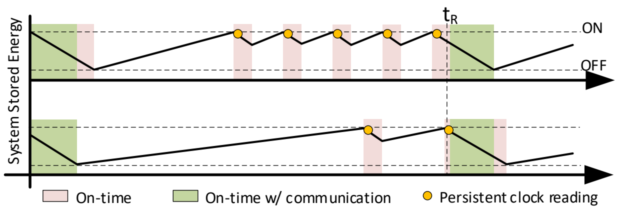


Fig. 13. A lifecycle management protocol in which nodes use a persistent clock to measure off-time in order to both be on at a pre-arranged rendezvous point, t_R.

The nodes then attempt to make the rendezvous by using their persistent clocks to maintain a continuous sense of time across off-times. Every time a node revives, it checks its continuous sense of time to see how close it is to the rendezvous. If the node can stay on until the rendezvous, it does so, turning on its radio and running its wireless MAC protocol for the full on-time. On the other hand, if the node's maximum on-time is not long enough to reach the rendezvous, it instead attempts a very quick communication (e.g., it broadcasts a probe packet and listens for a response), then immediately early-dies if no response is received. The motivation here is that, if the node uses as little energy as possible during the on-time, it will recharge faster, increasing the frequency of its on-times. We call this process of early dying immediately after checking the time and communication channel *fast-die cycling*. Clearly, having an accurate persistent clock is critical to the performance of proposed clock-based LMP protocol, because clock error can cause a node to miss the rendezvous, by taking a full on-time too early or too late.

C. Preliminary Results

We use simulations to quantify the effect of clock error on the performance of the proposed LMP. Our simulations use a custom Python event-based simulator. To simulate node charging and discharging behavior, we draw from distributions of on-times and off-times at various distances from the power transmitter, obtained experimentally using the same system deployment from Figure 2. We assume successful communication if the nodes have overlapping on-time. Sample results for communication between a node 1.5 m from the Powercast transmitter (PTX) and a node 2.0 m from the PTX are shown in Figure 14.

In Figure 14, the y-axis shows *communication success rate*, which we define as the number of successful communications divided by the number of possible communication opportunities (calculated as the total simulation time divided by the baseline lifecycle length of the energy-poor node). The x-axis shows the expected magnitude of the relative clock error, ϵ, used in the simulation. Every time a reading is taken from the persistent clock, a random value for relative error is chosen uniformly from the interval $[-2\epsilon, 2\epsilon]$ and applied to the persistent clock reading. As shown in the figure, the proposed LMP successfully communicates, on average, in over 25% of its opportunities in the presence of zero clock error, an over 5x improvement on the baseline. This demonstrates the critical role of the clock in the protocol.

As expected, the performance of the proposed LMP decreases as clock error increases. However, it also levels out at around 10–15% clock error, with around a 10% communication success rate. This means the protocol is better than the baseline regardless of the amount of clock error. This is

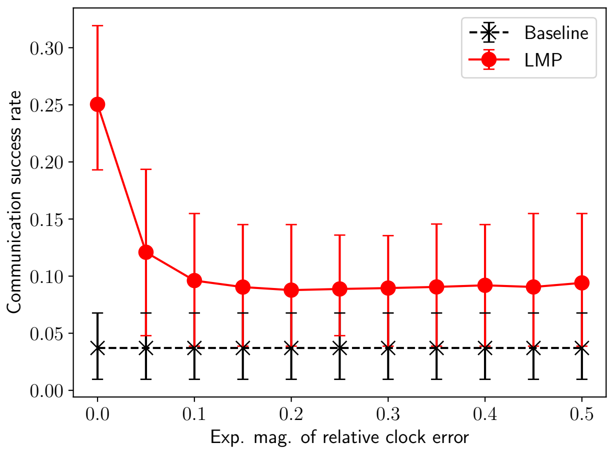


Fig. 14. Communication performance results over a range of the expected magnitude of simulated relative clock errors. Results are averaged over 200 simulations of one hour of simulated time. Errorbars show the 5th and 95th percentiles.

because, with large clock error, the proposed LMP regresses to a protocol where nodes generally perform fast-die cycling, but also sporadically take a full on-time. This "fallback" behavior is more effective than the baseline.

The shape of LMP in Figure 14 gives us a cutoff point: if using HARC with the proposed LMP yields better performance than LMP with 10% clock error, then HARC is aiding communication. From experimental results in Section V-C, we know that HARC-reg's average relative error is smaller than 10%. To evaluate whether this translates to communication performance, we used clock error data from the HARC experiments to simulate the proposed LMP running with HARC-reg and HARC-lite as the persistent clock. We also simulated LMP running with two different single-capacitor persistent clocks, with error values obtained with the same methodology. In these simulations, we fix one node (A) at 1.5 m from the PTX and vary the PTX distance of the other node (B) on the x-axis. Results for the median and 95th percentile intercommunication time, defined as the delay between successive successful communications, are shown in Figure 15.

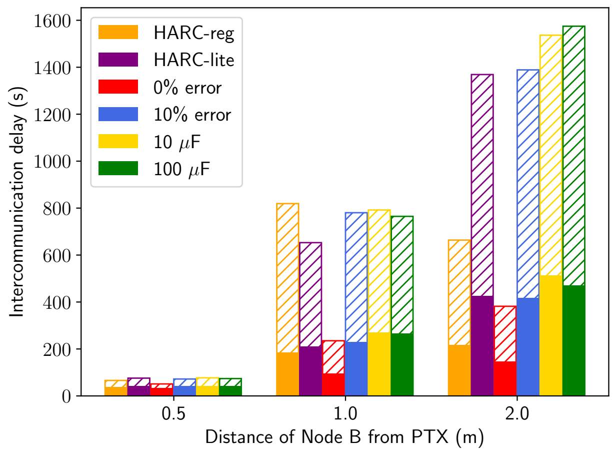


Fig. 15. Average intercommunication time vs. distance of node B from the PTX, with node A fixed at 1.5 m. The solid bar shows the median, and the hatched bar shows the 95th percentile. Results are averaged over 200 simulations with 10 intercommunication intervals each.

As expected, performance is better when node B is closer to the PTX, because this increases node B's charging rate and thus its ratio of on-time to off-time (which is approximately 25% at 0.5 m and 1.5% at 2.0 m). The baseline performs poorly (average over 1000 s at 2.0 m), and we omit it for readability. LMP with no error has the shortest intercommunication delay (note that the datapoint at 2.0 m corresponds to the zero-error datapoint shown in Figure 14). LMP with HARC-reg clearly outperforms LMP with 10% error over the range of distances, indicating the clock helps the protocol significantly. HARC-lite's performance is slightly better than LMP with 10% error, indicating it helps, but suggesting there is room for improvement in terms of this application. Finally, we also see that LMP with HARC-reg significantly outperforms either of the single-capacitor clocks, with 54% reduction in median intercommunication time over LMP with 100 μF at 2.0 m.

D. Next Steps

To validate our simulations, we are implementing our protocol and plan to test it with HARC on the TI SensorNode platform. This will allow us to verify the performance of the protocol, and HARC, with actual values for parameters that are currently simulated, such as node and radio boot-up time, charging rate variations, and low-level communication protocol behavior.

Our simulation results suggest that direct communication between intermittent nodes using active radios can be achieved regularly *if the persistent clock is accurate enough*. Going forward, we identify two paths to improving node-to-node communication with intermittent systems: (1) make the persistent clock more accurate, and (2) design LMPs that are more robust to persistent clock error, or that do not use persistent clocks. One alternative to persistent clocks for lifecycle management is detection of external events that are common across neighboring nodes, such as distinctive trends in energy harvesting rate. This could be done in software, using a sampling process similar to the fast-die cycling discussed here, or in hardware by designing a *wake-up harvester* circuit (inspired by the concept of wake-up radios).

VII. Conclusion and Future Works

In this paper, we present HARC (Heterogeneous Array of Redundant Persistent Clocks), a novel approach to timekeeping for intermittent, batteryless systems. HARC uses a heterogeneous, redundant array of capacitor-based persistent clocks that each decays in parallel, but at different rates, to provide variation-resilient high accuracy over a wide range of power off-times. We demonstrate the feasibility and effectiveness of HARC using experimental evaluations on a HARC prototype, and trace-based simulations of HARC-supported communication directly between two intermittent devices.

Our future work centers around developing the hardware and software components to build and deploy networks of batteryless nodes powered solely by ambient energy-harvesting. Going forward we will apply HARC techniques to tolerate more extreme environmental (e.g., temperature) variations that intermittent systems may encounter in real-world deployments. The current version of HARC is a prototype, we aim to further extend the idea by developing integrated VLSI implementations to achieve longer off-time measurement duration, improve the accuracy-energy trade-off, and reduce timekeeping overheads. We envision prototyping a testbed network of batteryless intermittent nodes built on top of our lifecycle management protocols.

Acknowledgment

We would like to thank Shen Fu for contributions to the LMP simulator. This work was supported in part by the U.S. National Science Foundation under Grants 1730275 and 2008548. We would also like to thank reviewers for their insightful comments and suggestions.

References

[1] A. Bakar and J. Hester, "Making sense of intermittent energy harvesting," in *Proceedings of the 6th International Workshop on Energy Harvesting & Energy-Neutral Sensing Systems*, 2018, pp. 32–37.

[2] D. C. Ranasinghe, R. L. S. Torres, A. P. Sample, J. R. Smith, K. Hill, and R. Visvanathan, "Towards falls prevention: a wearable wireless and battery-less sensing and automatic identification tag for real time monitoring of human movements," in *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2012, pp. 6402–6405.

[3] C. Donovan, A. Dewan, D. Heo, and H. Beyenal, "Batteryless, wireless sensor powered by a sediment microbial fuel cell," *Environmental science & technology*, vol. 42, no. 22, pp. 8591–8596, 2008.

[4] M. Minami, T. Morito, H. Morikawa, and T. Aoyama, "Solar biscuit: A battery-less wireless sensor network system for environmental monitoring applications," in *The 2nd international workshop on networked sensing systems*, vol. 2007. Citeseer, 2005.

[5] A. Colin, E. Ruppel, and B. Lucia, "A reconfigurable energy storage architecture for energy-harvesting devices," in *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems*, 2018, pp. 767–781.

[6] J. Hester, K. Storer, and J. Sorber, "Timely execution on intermittently powered batteryless sensors," in *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys '17. New York, NY, USA: Association for Computing Machinery, 2017. [Online]. Available: https://doi.org/10.1145/3131672.3131673

[7] A. Y. Majid, C. D. Donne, K. Maeng, A. Colin, K. S. Yildirim, B. Lucia, and P. Pawełczak, "Dynamic task-based intermittent execution for energy-harvesting devices," *ACM Trans. Sen. Netw.*, vol. 16, no. 1, Feb. 2020. [Online]. Available: https://doi.org/10.1145/3360285

[8] K. Maeng, A. Colin, and B. Lucia, "Alpaca: Intermittent execution without checkpoints," *arXiv preprint arXiv:1909.06951*, 2019.

[9] B. Lucia, V. Balaji, A. Colin, K. Maeng, and E. Ruppel, "Intermittent computing: Challenges and opportunities," in *2nd Summit on Advances in Programming Languages (SNAPL 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[10] D. Balsamo, A. S. Weddell, A. Das, A. R. Arreola, D. Brunelli, B. M. Al-Hashimi, G. V. Merrett, and L. Benini, "Hibernus++: a self-calibrating and adaptive system for transiently-powered embedded devices," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 12, pp. 1968–1980, 2016.

[11] H. Jayakumar, A. Raha, and V. Raghunathan, "Quickrecall: A low overhead hw/sw approach for enabling computations across power cycles in transiently powered computers," in *2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems*. IEEE, 2014, pp. 330–335.

[12] G. Berthou, T. Delizy, K. Marquet, T. Risset, and G. Salagnac, "Sytare: A lightweight kernel for nvram-based transiently-powered systems," *IEEE Transactions on Computers*, vol. 68, no. 9, pp. 1390–1403, 2018.

[13] V. Kortbeek, K. S. Yildirim, A. Bakar, J. Sorber, J. Hester, and P. Pawełczak, "Time-sensitive intermittent computing meets legacy software," in *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2020, pp. 85–99.

[14] E. Çürük, K. S. Yıldırım, P. Pawelczak, and J. Hester, "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.

[15] J. Hester, N. Tobias, A. Rahmati, L. Sitanayah, D. Holcomb, K. Fu, W. P. Burleson, and J. Sorber, "Persistent clocks for batteryless sensing devices," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 15, no. 4, pp. 1–28, 2016.

[16] B. Ransford, J. Sorber, and K. Fu, "Mementos: System support for long-running computation on rfid-scale devices," in *Proceedings of the sixteenth international conference on Architectural support for programming languages and operating systems*, 2011, pp. 159–170.

[17] H. Williams, X. Jian, and M. Hicks, "Forget failure: Exploiting sram data remanence for low-overhead intermittent computation," in *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2020, pp. 69–84.

[18] B. Lucia and B. Ransford, "A simpler, safer programming and execution model for intermittent systems," *ACM SIGPLAN Notices*, vol. 50, no. 6, pp. 575–585, 2015.

[19] J. Van Der Woude and M. Hicks, "Intermittent computation without hardware support or programmer intervention," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2016)*, 2016, pp. 17–32.

[20] A. Y. Majid, C. D. Donne, K. Maeng, A. Colin, K. S. Yildirim, B. Lucia, and P. Pawełczak, "Dynamic task-based intermittent execution for energy-harvesting devices," *ACM Transactions on Sensor Networks (TOSN)*, vol. 16, no. 1, pp. 1–24, 2020.

[21] A. Rodriguez Arreola, D. Balsamo, G. V. Merrett, and A. S. Weddell, "Restop: Retaining external peripheral state in intermittently-powered sensor systems," *Sensors*, vol. 18, no. 1, p. 172, 2018.

[22] K. Maeng and B. Lucia, "Supporting peripherals in intermittent systems with just-in-time checkpoints," in *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, 2019, pp. 1101–1116.

[23] A. Rahmati, M. Salajegheh, D. Holcomb, J. Sorber, W. P. Burleson, and K. Fu, "TARDIS: Time and remanence decay in SRAM to implement secure protocols on embedded devices without clocks," in *Proceedings of the 21st USENIX Security Symposium*, 2012, pp. 221–236.

[24] J. de Winkel, C. Delle Donne, K. S. Yildirim, P. Pawełczak, and J. Hester, "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.

[25] *TX91501B–915 MHz Powercaster Transmitter*, Powercast, 10 2018.

[26] *P2110B 915 MHz RF Powerharvester Receiver*, Powercast, 12 2016.

[27] H. J. Visser and R. J. Vullers, "RF energy harvesting and transport for wireless sensor network applications: Principles and requirements," *Proceedings of the IEEE*, vol. 101, no. 6, pp. 1410–1423, 2013.

[28] *Multi-Standard CC2650 SensorTag Design Guide*, Texas Instruments, 03 2015.

[29] S. Duquennoy *et al.*, "Contiki-NG: The OS for next generation IoT devices," 2019.

[30] E. Kreyszig, *Advanced Engineering Mathematics*, ser. Wiley International Edition. Wiley, 1972.

[31] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[32] N. Cristianini, J. Shawe-Taylor *et al.*, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[33] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[34] *MSP430FR58xx, MSP430FR59xx, and MSP430FR6xx Family User's guide*, Texas Instruments, 10 2017.

[35] *MSP430FR5994 LaunchPad™ Development Kit (MSP-EXP430FR5994)*, Texas Instruments, 09 2019.

[36] *Gravity: MOSFET Power Controller SKU: DFR0457*, DFRobot.

[37] "Energytrace technology," https://www.ti.com/tool/ENERGYTRACE, accessed: 2020-06-15.

[38] W. J. Riley, "Handbook of frequency stability analysis," 2008.

[39] A. Shrivastava, D. A. Kamakshi, and B. H. Calhoun, "A 1.5 nw, 32.768 khz xtal oscillator operational from a 0.3 v supply," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 3, pp. 686–696, 2016.

[40] H. Esmaeelzadeh and S. Pamarti, "18.4 a 0.55 nw/0.5 v 32khz crystal oscillator based on a dc-only sustaining amplifier for iot," pp. 300–301, 2019.

[41] J. Jung, I.-H. Kim, S.-J. Kim, Y. Lee, and J.-H. Chun, "A 1.08-nw/khz 13.2-ppm/° c self-biased timer using temperature-insensitive resistive current," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 8, pp. 2311–2318, 2018.

[42] Y. Zeng, T. Jang, Q. Dong, M. Saligane, D. Sylvester, and D. Blaauw, "A 1.7 nw pll-assisted current injected 32khz crystal oscillator for iot," pp. C68–C69, 2017.