# Energy-adaptive Buffering for Efficient, Responsive, and Persistent Batteryless Systems

Harrison Williams
hrwill@vt.edu
Virginia Tech
USA

Matthew Hicks
mdhicks2@vt.edu
Virginia Tech
USA

## Abstract

Batteryless energy harvesting systems enable a wide array of new sensing, computation, and communication platforms untethered by power delivery or battery maintenance demands. Energy harvesters charge a buffer capacitor from an unreliable environmental source until enough energy is stored to guarantee a burst of operation despite changes in power input. Current platforms use a fixed-size buffer chosen at design time to meet constraints on charge time or application longevity, but static energy buffers are a poor fit for the highly volatile power sources found in real-world deployments: fixed buffers waste energy both as heat when they reach capacity during a power surplus and as leakage when they fail to charge the system during a power deficit.

To maximize batteryless system performance in the face of highly dynamic input power, we propose *REACT*: a responsive buffering circuit which varies total capacitance according to net input power. *REACT* uses a variable capacitor bank to expand capacitance to capture incoming energy during a power surplus and reconfigures internal capacitors to reclaim additional energy from each capacitor as power input falls. Compared to fixed-capacity systems, *REACT* captures more energy, maximizes usable energy, and efficiently decouples system voltage from stored charge—enabling low-power and high-performance designs previously limited by ambient power. Our evaluation on real-world platforms shows that *REACT* eliminates the tradeoff between responsiveness, efficiency, and longevity, increasing the energy available for useful work by an average 25.6% over static buffers optimized for reactivity and capacity, improving event responsiveness by an average 7.7*x* without sacrificing capacity, and enabling programmer directed longevity guarantees.

**ACM Reference Format:**
Harrison Williams and Matthew Hicks. 2024. Energy-adaptive Buffering for Efficient, Responsive, and Persistent Batteryless Systems. In *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1 (ASPLOS '24), April 27-May 1, 2024, La Jolla, CA, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3620666.3651370

## 1 Introduction

Ever-shrinking computing and sensing hardware has pushed mobile Internet-of-Things (IoT) type devices beyond the limits of the batteries powering them. A typical low cost/power microcontroller [21] drains a 1 $cm^3$ battery nearly 14x its size in just over 8 weeks of active operation [39], rendering the system useless without a potentially costly replacement effort. Cost, maintenance, and safety concerns make batteries further incompatible with massive-scale (one million devices per square kilometer [5]) and deeply-deployed (infrastructure [2], healthcare [37]) applications. IoT engineers are turning to batteryless energy harvesting platforms to power low-cost, perpetual systems capable of driving a ubiquitous computing revolution. Increasingly efficient energy harvesting circuits enable batteryless systems across a range of IoT use cases including feature-rich batteryless temperature sensors 500x smaller than a grain of rice [51] and batteryless flow-meters [15] supporting deep-sea drilling or geothermal plants for decades without maintenance.

The energy harvesting design model both enables new deployments previously limited by batteries and places new demands on system developers. Harvested energy is highly unreliable: sensitive environmental factors such as shadows over a photovoltaic cell or shifts in the orientation of a rectenna produce rapid, outsized changes in the energy scavenged by the harvester. Energy harvesters mitigate this unreliability by charging an energy buffer to a given enable voltage, which the system periodically discharges to supply a useful quantum of work despite potential power loss.

Buffer capacity is a key design element of any batteryless system. Past work [7] explores the tradeoff between buffer sizes: small buffers are highly *reactive*—charging rapidly and quickly enabling the system to address time-sensitive events—but sacrifice *longevity* because they rapidly discharge during operation, guaranteeing only a short burst of uninterrupted operation. Large buffers store more energy at a given voltage, improving longevity by supporting a longer or more energy-intensive burst of operation at the cost of reactivity because they require more energy to enable the system at all. Matching buffer size to projected energy demand is critical

*ASPLOS '24, April 27-May 1, 2024, La Jolla, CA, USA*
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0386-7/24/04.
https://doi.org/10.1145/3620666.3651370

to ensuring the system is both reactive enough to address incoming events/deadlines (e.g., periodic sensor readings) and long-lived enough to support uninterruptible operations (e.g., radio transmissions). Designers choose the minimum size necessary to power all atomic operations on the device, maximizing reactivity given a required level of longevity.

In this work, we explore static energy buffer *efficiency* as a third metric for buffer performance and find that it varies dramatically with *net energy input* rather than simple energy demand. Small buffers reach capacity quickly if power input exceeds instantaneous demand—burning off hard-won energy as heat to prevent overvoltage. Large buffers capture all incoming power, but enable slowly and lose more harvested energy to leakage below the minimum system voltage. The volatile nature of harvested power means that fixed-size buffers experience *both* problems over the course of their deployment, discharging energy during a power surplus and losing large portions of energy to leakage during a deficit.

To make the most of incoming energy in all circumstances, we propose *REACT*[1]: a dynamic energy buffering system that varies its capacitance following changes in *net* power. *REACT* maximizes system responsiveness and efficiency using a small static buffer capacitor, quickly enabling the system to monitor events or do other low-power work under even low input power. If input power rises beyond the current system demand and the static buffer approaches capacity, *REACT* connects additional capacitor banks to absorb the surplus, yielding the capacity benefits of large buffers without the responsiveness penalty. When net power is negative, these capacitors hold the system voltage up and extend operation beyond what is possible using the small static capacitor.

While expanding buffer size to follow net power input ensures the system can capture *all* incoming energy, increasing capacitance also increases the amount of unusable charge stored on the capacitor banks—charge which could power useful work if it were on a smaller capacitor and therefore available at a higher voltage. As supply voltage falls and approaches a minimum threshold, *REACT reclaims* this otherwise-unavailable energy by reconfiguring capacitor banks into series, shifting the same amount of charge onto a smaller equivalent capacitance in order to boost the voltage at the buffer output and ensure the system continues operating for as long as possible. *REACT* effectively eliminates the design tradeoff between reactivity and capacity by tuning buffer size within an arbitrarily large capacitance range, only adding capacity when the buffer is already near full. *REACT*'s charge reclamation techniques maximize efficiency by moving charge out of large capacitor banks onto smaller ones when net input power is negative, ensuring all energy is available for useful work.

We integrate a hardware prototype of *REACT* into a full energy harvesting platform to evaluate it against previous work, operating under different input power conditions and with different power consumption profiles. Our results indicate that *REACT* provides the "best of both worlds" of both small- and large-buffer systems, rapidly reaching the operational voltage under any power conditions while also expanding as necessary to capture all available energy and provide software longevity guarantees as needed. Maximizing buffer capacity and reclaiming charge using *REACT*'s reconfigurable capacitor banks eliminates the efficiency penalties associated with both small and large static capacitor buffers, increasing the portion of harvested energy used for application code by an average 39% over an equally-reactive static buffer and 19% over an equal-capacity one. We also compare *REACT* to recent work exploring dynamic-capacitance batteryless systems [53] using a fully-interconnected capacitor architecture; our evaluation on real-world energy harvesting traces shows that prior work targeting the reactivity-longevity tradeoff *underperforms* baseline static capacitance systems as a result of lossy switching between capacitance modes. *REACT* improves performance by an average 26% over the state of the art owing to its efficient bank-based charge management structure. This paper makes the following technical contributions:

- We evaluate the power dynamics of common batteryless systems in real deployments and explore how common-case volatility introduces significant energy waste in static or demand-driven buffers (§ 2).
- We design *REACT*, a dynamic buffer system which varies its capacitance according to system needs driven by net input power (§ 3). *REACT*'s configurable arrays combine the responsiveness of small buffers with the longevity and capacity of large ones, enables energy reclamation to make the most of harvested power, and avoids the pitfalls of energy waste inherent in other dynamic capacitance designs (§3.3).
- We integrate *REACT* into a batteryless system and evaluate its effect on reactivity, longevity, and efficiency under a variety of power conditions and workloads (§5). Our evaluation indicates that *REACT* eliminates the responsiveness-longevity tradeoff inherent in static buffer design while increasing overall system efficiency compared to *any* static system.
- We publish our design and evaluation data for *REACT* and baseline systems in an open-source repository at https://github.com/FoRTE-Research/REACT-Artifact to enable further evaluation and integration of *REACT* into batteryless systems.

## 2 Background and Related Work

Scaling sensing, computation, and communication down to smart dust [25] dimensions requires harvesting power on-demand rather than packaging energy with each device using a battery. Many batteryless systems use photovoltaic [10, 51]

[1] **R**econfigurable, **E**nergy-**A**daptive **C**apaci**T**ors

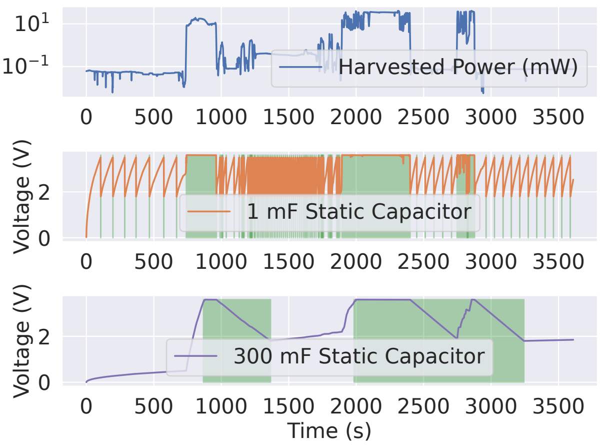


**Figure 1.** Static buffer operation on a simulated solar harvester. Highlighted blocks indicate the system is running.

or RF power [43], while other use-cases are better suited for sources such as vibration [45], fluid flow [15], or heat gradients [29]. Commonalities between ambient power sources have inspired researchers to develop general-purpose batteryless systems, regardless of the actual power source: ambient power is *unpredictable*, *dynamic*, and often *scarce* relative to the active power consumption of the system.

Batteryless systems isolate sensing and actuation components from volatile power using a buffer capacitor. The harvester charges the capacitor to a pre-defined enable voltage, after which the system turns on and begins consuming power. Because many environmental sources cannot consistently power continuous execution, systems operate intermittently—draining the capacitor in short bursts of operation punctuated by long recharge periods. This general-purpose intermittent operation model has enabled researchers to abstract away the behavior of incoming power and focus on developing correct and efficient techniques for working under intermittent power [19, 28, 30, 31, 48, 50].

## 2.1 Choosing Buffer Capacity

Buffer size determines system behavior in several important ways. Supercapacitors provide inexpensive and small-form-factor bulk capacitance [26], enabling designers to choose a capacitor according to performance rather than cost or size concerns. Two metrics motivate past work: *reactivity* refers to the system's ability to rapidly charge to its enable voltage and begin operation. High reactivity ensures a system is online to execute periodic tasks or address unpredictable input events. *Longevity* refers to the energy available for an uninterrupted period of work with no additional power input; long-lived systems support high-power and long-running uninterruptible operations and reduce the overhead incurred with state recovery after a power loss.

### 2.1.1 Reactivity and Longevity:

A batteryless system's reactivity and longevity depend primarily on the charge and discharge rate of the buffer capacitor. We illustrate the tradeoff using a simulated solar harvester with a 22% efficient, 5 $cm^2$ panel, based on a pedestrian trace from the EnHANTs solar dataset [14]. The system runs from 3.6V down to 1.8V and draws 1.5 mA in active mode, representative of a typical deployment [27]. Figure 1 illustrates the reactivity-longevity tradeoff inherent in static buffer systems at two design extremes, using a 1 mF and 300 mF capacitor. The 1 mF system charges rapidly and is therefore highly reactive, reaching the enable voltage over $8x$ sooner than the 300 mF version. However, the smaller capacitor also discharges quickly—the mean length of an uninterrupted power cycle using the 1 mF capacitor is 10 seconds versus 880 seconds for the 300 mF capacitor, indicating the 300 mF system is far longer-lived once charged. The relative importance of reactivity and longevity depends on the use case, but often changes over time for a complex system—complicating design further.

### 2.1.2 Power Volatility and Energy Efficiency:

Buffer capacity is also a major driver of end-to-end energy *efficiency*: using the 300 mF capacitor our system is operational for 49% of the overall power trace, compared to only 27% for the 1 mF platform. This stems from the high volatility of incoming power—82% of the total energy input is collected during short-duration power spikes when harvested power rises above 10 mW, despite the system spending 77% of its time at input powers below 3 mW. A large buffer captures this excess energy to use later while the smaller buffer quickly reaches capacity and discharges energy as heat to avoid overvoltage.

Large buffers, however, are not always more efficient: the energy used to charge the capacitor to the operational voltage cannot power useful work, and is eventually lost to leakage while the system is off. When power input is low, this "cold-start" energy represents a significant portion of total harvested energy. For the system described above powered by a solar panel at night [14], the 1 mF buffer enables a duty cycle of 5.7% versus only 3.3% using a 10 mF buffer. This low power environment highlights another risk of oversized buffers: the system using the 300 mF capacitor never reaches the enable voltage and so never begins operation.

Improvements in harvester efficiency and low-power chip design are closing the gap between harvester output and chip power consumption. Power is increasingly limited by volatile environmental factors rather than scarcity induced by low efficiency; the result is that energy harvesters experience periods of both energy scarcity and surplus. Rapidly changing power conditions place opposing demands on batteryless systems, which must remain responsive with low input power, provide longevity for long-running operations, and maximize efficiency by avoiding energy waste.

### 2.2 Power-Responsive Performance Scaling

One solution to volatile energy input is modulating throughput according to incoming power, increasing execution rate when power is plentiful and decreasing it to maintain availability when power is scarce. Limiting net input power to the buffer by matching power consumption with input enables systems to use small buffer capacitors without reaching the buffer capacity, ensuring no power is wasted with an over-full buffer. Past work realizes power-responsive scaling using heterogeneous architectures [11] or by adapting the rate and accuracy of software execution [1, 4, 32, 52].

Unfortunately, we find the assumptions underlying power-performance scaling often do not apply to batteryless systems. Increasing energy consumption by accelerating execution only improves systems which *have useful work to do exactly when input power is high*, but many batteryless systems focus on periodic sensing and actuation deadlines which do not correlate with ambient power supply. Further, resource-constrained platforms may have few on-chip operations which can be delayed until power is plentiful; when these operations do exist, they are often not amenable to scaling (e.g., transmitting data to a base station may be delayed but always requires a fixed-cost radio operation). Flexible batteryless systems must capture energy and use it on demand rather than fit operation to unreliable power input.

### 2.3 Multiplexed Energy Storage

Rather than match power consumption to incoming supply, systems may charge multiple static buffers according to projected demand. Capybara [7] switches capacitance using an array of heterogeneous buffers: programmers set capacitance modes throughout the program, using a smaller capacitor to maximize responsiveness for low-power or interruptible tasks and switching to a larger capacitor for high-power atomic operations. UFoP and Flicker [17, 18] assign each peripheral on the system a separate buffer and charging priority, enabling responsive low-power operation while waiting to collect sufficient energy for high-power tasks. These systems increase overall energy capacity by directing excess power to capacitors not currently in use.

Static arrays increase capacity without reducing responsiveness, but waste energy when charge is stored on unused buffers. Reserving energy in secondary capacitors 1) requires error-prone [32] speculation about future energy supply and demand to decide charging priorities, which can change between when energy is harvested and when it needs to be used; and 2), wastes energy as leakage when secondary buffers are only partially charged, failing to enable associated systems and keeping energy from higher-priority work. To minimize programmer speculation, decouple tasks which compete for buffered energy, and minimize leakage, energy must be *fungible*: the buffer must be capable of directing *all* harvested energy to *any* part of the system on demand.

### 2.4 Unified Dynamic Buffering

Past work has also explored varying the behavior of a single unified buffer to capture the fungibility requirement described above. Dewdrop [6] varies the enable voltage to draw from a single capacitor according to projected needs (e.g., begin operation at 2.2V instead of 3.6V)—providing complete energy fungibility—but still suffers from the reactivity-longevity tradeoff of capacitor size. Morphy [53] replaces static buffers using a set of capacitors in a unified switching network; software can connect and disconnect arbitrary sets of capacitors in series or parallel to produce different equivalent capacitances.

We evaluate *REACT* alongside Morphy because the Morphy architecture also targets the reactivity and longevity challenges discussed in § 1, but several key design choices differentiate each system. Morphy's fully-connected capacitor network (Figure 4) enables a wide range of equivalent capacitance configurations for a given set of capacitors, particularly for low capacitances: every capacitor in the network can be combined in series for an equivalent capacitance of C/N (assuming N C-sized capacitors), producing a highly-reactive system. This design also enables Morphy to act both as bulk energy storage and a large charge pump, boosting low-voltage inputs from an energy harvester by up to the number of capacitors in the system by alternating between fully-parallel and fully-series. In contrast, *REACT*'s bank-based design (Figure 3) limits the number of potential configurations as all banks are effectively permanently in parallel—instead, *REACT* realizes high reactivity using a small static capacitor (§3.2).

The key advantage of *REACT*'s design is *energy efficiency*. As software reconfigures Morphy's fully-connected network to vary capacitance, current flows between capacitors *within* the network to equalize the output voltage of the array. This internal current dissipates a significant portion of stored energy; energy is lost on every reconfiguration of the network, dramatically reducing overall efficiency when faced with the highly volatile power inputs typical of many real-world deployments (§2.1.2) as the system rapidly varies capacitance to match net power input. Our evaluation in §5 shows that this internal energy dissipation reduces common-case end-to-end performance below that of even static capacitors, making this approach impractical for energy-constrained devices. *REACT* eliminates this loss by dividing sets of capacitors into mutually isolated banks and ensuring no current flows between banks even as software varies capacitance. This energy-focused approach prioritizing minimal energy loss is key to developing intermittent systems that simultaneously maximize reactivity, longevity, and overall efficiency.

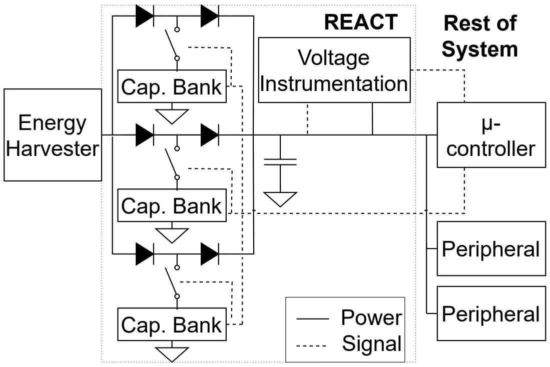


**Figure 2.** *REACT* diagram and signal flow between components.

# 3 Design

An intelligent energy buffering strategy is key to effective and efficient batteryless systems. Three performance objectives, informed by the advantages and limitations of prior approaches, drive *REACT*'s design:

- **Minimize charge time:** Rapidly reaching the operational voltage, even when buffered energy cannot support complex operation, maximizes reactivity and enables systems to reason about power or sensor events from within low-power sleep modes.
- **Maximize capacity:** System-wide longevity and efficiency require buffering large amounts of incoming energy when power supply exceeds demand, either to power long-running uninterruptible operations or support future power demand when supply is low.
- **Maximize energy fungibility:** Unpredictable power demand patterns mean that energy cannot be preprovisioned to specific operations at harvest-time; systems need the ability to draw all harvested energy from the buffer and direct it as needed.

## 3.1 *REACT* Overview

*REACT* buffers energy using a fabric of reconfigurable capacitor banks that software adjusts as needed. Figure 2 shows a high-level overview of *REACT*'s hardware design. We design *REACT*'s hardware component as a drop-in replacement for a typical buffer between the harvester and the rest of the system, while the buffer management software requires no code modification or programmer input. The only system prerequisite is a set of digital I/O pins to configure capacitor banks and receive voltage monitoring information.

## 3.2 Cold-start Operation and the Last-level Buffer

From a cold start ($E(t) = 0$), *REACT* minimizes overall capacitance in order to rapidly charge to the operational voltage and enable the system with minimum energy input (high reactivity). The minimum capacitance is set by the smallest quantum of useful work available on the system (minimum required longevity), such as a short-lived software operation or an initialization routine that puts the system into a low-power responsive sleep mode. *REACT* provides this rapid charge time using a small static buffer referred to hereafter as the *last-level buffer*. Additional capacitor banks are connected using normally-open switches and only contribute to overall capacitance when configured to do so in software, after the system is able to reason about buffered energy.

The last-level buffer sets the minimum capacitance at power-on when all other banks are disconnected. This enables simple tuning of the energy input required to enable the system (reactivity) and the guaranteed energy level when the system does begin work (minimum longevity). It also smooths voltage fluctuations induced by capacitor bank switching (§ 3.3.4). Finally, the last-level buffer serves as the combination point between the different capacitor banks and the rest of the system. Although energy may be stored in multiple banks of varying capacity at different voltages, combining it at the last-level buffer simplifies system design by presenting harvested power as a unified pool of energy which the system taps as needed (i.e., harvested energy is fungible).

### 3.2.1 Monitoring Buffered Energy.

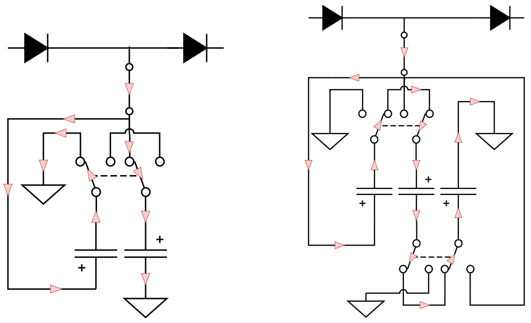
Despite mutual isolation, bank voltages tends to equalize: the last-level buffer pulls energy from the highest-voltage bank first, and current flows from the harvester to the lowest-voltage bank first. This enables *REACT* to measure only the voltage on the last-level buffer as a surrogate for remaining energy capacity. If voltage rises beyond an upper threshold—the buffer is near capacity—*REACT*'s voltage instrumentation hardware signals the software component running on the microcontroller to increase capacitance using the configurable banks. Voltage falling below a lower threshold indicates the buffer is running out of energy and that *REACT* should reconfigure banks to extract additional energy and extend operation. *REACT*'s instrumentation only needs to signal three discrete states—near capacity, near undervoltage, and OK—so two low-power comparators is sufficient for energy estimation.

## 3.3 Dynamic Capacitor Banks

The last-level buffer on its own enables high reactivity and minimizes cold-start energy below the operational minimum, maximizing efficiency during power starvation. However, when net power into the buffer is positive—such as during a period of high input power or low workload—the small last-level buffer rapidly reaches capacity. *REACT* provides the energy capacity required to both maximize efficiency and support long-running operation by connecting configurable capacitor banks when the last-level buffer reaches capacity, as shown in Figure 2.

### 3.3.1 Capacitor Organization.

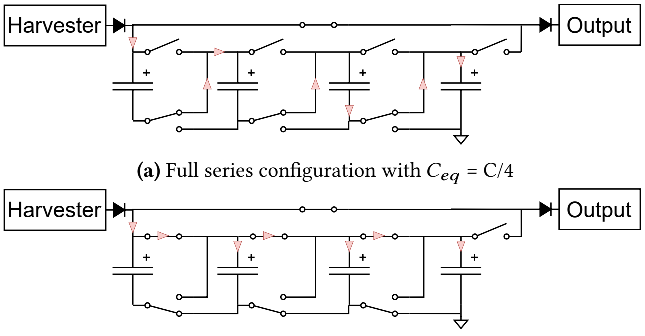
Careful management of the connections between each capacitor is key to maximizing energy efficiency while also presenting a valid operational voltage for the computational backend. Morphy [53] presents



**(a)** Two capacitors, parallel configuration

**(b)** Three capacitors, series configuration

**Figure 3.** *REACT* capacitor banks in different bank sizes and configurations. Arrows indicate charging current path.



**(a)** Full series configuration with $C_{eq}$ = C/4

**(b)** Full parallel configuration with $C_{eq}$ = 4C

**Figure 4.** Structure of the unified approach presented by Yang et al. [53]. Arrows indicate charging current path.

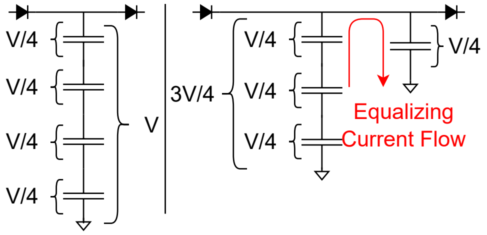


**Figure 5.** Dissipative current flow in a fully-unified buffer during reconfiguration. Energy is dissipated by current spikes after capacitors at different voltages are placed in parallel.

one approach: by connecting a set of equally-sized capacitors through switches similar to a charge pump, overall buffer capacitance can be varied across a wide range of capacitance values. Different switch configurations produce intermediate buffer sizes between the extremes shown in Figure 4; gradually stepping through these configurations smoothly varies capacitance through software control.

A fully interconnected array enables a wide range of equivalent capacitances, but introduces significant waste through dissipative heating when the charged capacitor array is reconfigured. Figure 5 illustrates how energy is lost when charged capacitors are connected in a new configuration. Before reconfiguration, the energy contained in the system is $E_{old} = \frac{1}{2}(C/4)V^2$; when a capacitor is taken out of series and placed in parallel with the remaining capacitors to increase equivalent capacitance to $4C/3$, current flows to the lower-voltage newly-parallel capacitor to equalize output voltage. The final output voltage is $3V/8$, and the remaining energy is $E_{new} = \frac{1}{2}(4C/3)(3V/8)^2$. The portion of energy conserved is $E_{new}/E_{old} = 0.75$—i.e., 25% of buffered energy is dissipated by current in the switches during reconfiguration. Larger arrays are increasingly inefficient: the same scenario with an 8-capacitor array wastes 56.25% of its buffered energy transitioning from an 8-parallel to a 7-series-1-parallel configuration. Similar waste occurs when reducing equivalent capacitance by placing capacitors in series.[2] Our evaluation in § 5.6 indicates that the energy loss caused by switching often outweighs any advantage from dynamic behavior, causing the fully-connected approach to underperform even static buffers.

### 3.3.2 Bank Isolation.

The switching loss discussed above stems from charge flowing between capacitors within the power network as they switch into different configurations. *REACT* eliminates unnecessary current flow by organizing capacitors into independent, mutually isolated banks as shown in Figure 2. Figure 3 illustrates in detail two example capacitor banks in each possible configuration: capacitors within a bank can only be arranged in either full-series (low capacitance) or full-parallel (high capacitance) so that no current flows between capacitors *within* a bank. Isolation diodes on the input and output of each bank prevent current *between* banks: when a charged parallel-configured bank is reconfigured into series (reducing its capacitance and boosting its output voltage), isolation diodes prevent it from charging other banks in the array. Similarly, banks re-configured into parallel cannot draw current from anywhere except the energy harvester. Isolation reduces the number of potential capacitor configurations compared to a fully-connected network, but dramatically increases energy efficiency.

*REACT*'s isolation diodes direct the flow of current: intermediate capacitor arrays are only charged directly from the energy harvester and only discharge to the last-level buffer. This also means that all current from the harvester

[2] Charge pumps avoid this waste by never connecting capacitors at different potentials in parallel; in this use case, however, parallel capacitance is always necessary to smooth voltage fluctuations during switching and keep the output voltage within the computational backend's acceptable range.

flows through two diodes before reaching the system, so minimizing power dissipation in the diodes is essential to maintaining overall system efficiency. To maximize charging efficiency, we design *REACT* using ideal diode circuits incorporating a comparator and pass transistor, rather than typical PN or Schottky diodes. Active ideal diodes are far more efficient at typical currents for batteryless systems: the circuit we use [22] dissipates 0.02% of the power dissipated in a typical Schottky diode [46] at a supply current of 1 mA.

**3.3.3 Bank Reconfiguration.** The range of buffer sizes depends on the number of capacitor banks and the number of capacitors in each bank. *REACT*'s capacitor banks are effectively connected in parallel, so the overall capacitance is the sum of each bank's contribution. Each *REACT* bank containing $N$ identical capacitors of capacitance $C$ may be configured to contribute no capacitance (disconnected), series capacitance $C/N$, or parallel capacitance $NC$.

*REACT* must increment buffer capacitance in small steps in order to keep voltage within the operational range while capturing all incoming power. A large increase in capacitance pulls output voltage down and introduces cold-start energy loss if net power input is low; for extreme cases, the system may run out of energy and cease execution while the new capacitance charges *even if incoming power would be sufficient to power operation. REACT* first connects banks in the series configuration to contribute a small capacitance and avoid large jumps in overall buffer size. If the buffer continues to charge and reaches the upper voltage limit $V_{high}$, *REACT* further expands capacitance by toggling double-pole-double-throw bank switches to configure the capacitors in parallel. Expanding the buffer by reconfiguring charged capacitors rather than adding new ones reduces the time the system is cut off from input power while current flows exclusively to the new capacitance, because it is already charged to $V_{high}/N$. Because no current flows between capacitors or banks, bank reconfiguration changes capacitance seen on the common rail without dissipative loss. *REACT* uses break-before-make switches to ensure no short-circuit current flows during switching; incoming current flows directly to the last-level buffer during the momentary open-circuit in the bank.

**3.3.4 Charge Reclamation.** Reconfiguring a bank from series to parallel allows *REACT* to efficiently increase capacitance without dropping output voltage. When voltage on the last-level buffer appraches the threshold value $V_{low}$, indicating net power is leaving the buffer, *REACT* needs to reduce equivalent capacitance to boost voltage and keep the backend running. *REACT* accomplishes this by transitioning charged $N$-capacitor banks from the parallel to the series configuration, reducing equivalent capacitance from $NC$ to $C/N$ and boosting output voltage from $V_{low}$ to $NV_{low}$. This boosts voltage on the last-level buffer and extracts more energy from the capacitor bank than would otherwise be available once voltage falls below $V_{low}$.

The remaining energy unavailable after the parallel⟶series transition depends on the number $N$ of $C_{unit}$-size capacitors in the bank. Before switching, the cold-start energy stored on the parallel-mode bank is $E_{par} = \frac{1}{2} N C_{unit} V_{low}^2$. Switching to the series configuration conserves stored energy: $E_{ser} = \frac{1}{2}(C_{unit}/N)(NV_{low})^2 = E_{par}$, but boosts voltage to enable the digital system to continue extracting energy. If net power remains negative, the system eventually drains the series-configuration bank down to $V_{low}$. This is energetically equivalent to draining the parallel-configuration bank to $V_{low}/N$, leaving $E_{par} = \frac{1}{2} N C_{unit} (V_{low}/N)^2 = \frac{1}{2} C_{unit} V_{low}^2 / N$ unusable; the overall result is that *REACT* reduces energy loss by a factor of $N^2$ when reducing system capacitance compared to simply disconnecting the capacitor.

**3.3.5 Bank Size Constraints.** Increasing the number of capacitors $N$ in a bank improves efficiency by reclaiming more energy when switching a bank from parallel to series. However, it also introduces voltage spikes when the bank output voltage is temporarily multiplied by $N$. Because *REACT* measures overall energy at the last-level buffer, the software component may interpret this voltage spike as a buffer-full signal and incorrectly add capacitance despite low buffered energy. In extreme cases, the voltage spike may exceed component absolute limits.

The size of the last-level buffer $C_{last}$ constrains the number $N$ and size $C_{unit}$ of each capacitor in a bank in order to keep voltage below *REACT*'s buffer-full threshold during a parallel⟶series transition. A larger $C_{unit}$ contains more energy and thus pulls voltage higher when switched from parallel to series. Equation 1 gives the last-level buffer voltage after switching a bank to series at a trigger voltage $V_{low}$:

$$V_{new} = \frac{(NV_{low})(C_{unit}/N)}{C_{last} + C_{unit}/N} + \frac{V_{low} * C_{last}}{C_{last} + C_{unit}/N} \tag{1}$$

Constraining $V_{new} < V_{high}$ and solving for $C_{unit}$ yields the absolute limit for $C_{unit}$ (Equation 2). Note that $C_{unit}$ is only constrained if the parallel⟶series transition at $V_{low}$ produces a voltage above $V_{high}$:

$$C_{unit} < \frac{NC_{last}(V_{high} - V_{low})}{NV_{low} - V_{high}} \tag{2}$$

### 3.4 *REACT* Software Interface

*REACT*'s standalone hardware design means that the software component running on the target microcontroller is minimal. The software subsystem monitors for incoming over- or under-voltage signals from *REACT*'s voltage instrumentation and maintains a state machine for each capacitor bank. Each capacitor bank is disconnected at startup; on an overvoltage signal from *REACT*'s hardware, the software directs *REACT* to connect a new capacitor bank in the series configuration. A second overvoltage signal[3] causes *REACT*

[3] *REACT* polls the over/undervoltage signals using an internal timer rather than edge-sensitive interrupts to handle cases such as a high enough power input that the capacitance step does not pull supply voltage below $V_{low}$.

| **Bank** | **0** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|---|
| **Capacitor Size ($\mu F$)** | 770 | 220 | 440 | 880 | 880 | 5000 |
| **Capacitor Count** | 1 | 3 | 3 | 3 | 3 | 2 |

**Table 1.** Bank size and configurations for our *REACT* test implementation. Bank 0 is the last-level buffer.

to reconfigure the newly-connected bank to parallel; on the next overvoltage signal, *REACT* connects a second capacitor bank, and so on. *REACT* similarly steps capacitor banks in the opposite direction when an undervoltage signal arrives.

Our proof-of-concept *REACT* implementation uses external voltage detectors with hard-wired thresholds to supply over/undervoltage signals to the software component and varies capacitance to keep supply voltage within a range corresponding to the limits for our load microcontroller (2.0-3.4V). One possible extension to *REACT* is to instead use on-chip analog comparators or ADCs and redefine these thresholds in software, enabling developers to dynamically change the target operating voltage at different points in the program. Decoupling supply voltage from buffered energy in this way has both functional benefits (some operations, like writes to Flash memory, require a higher supply voltage) and performance benefits (e.g., keeping the input to a voltage regulator at its most efficient operating point).

**3.4.1 Software-Directed Longevity.** *REACT*'s software component requires no active programmer intervention or code changes aside from setting voltage thresholds, initializing each bank state machine, and setting the order to connect and disconnect banks. Software does not need to know the details ($N$, $C_{unit}$) of each bank, although this information with the state of each bank gives a coarse idea of the current buffered energy. Because *REACT* only changes capacitance when the bank is near-full or near-empty, capacitance level is an effective surrogate for stored energy. Application code can use this feedback to set longevity guarantees through *REACT*'s software interface. In preparation for a long-running or high-energy atomic operation, software sets a *minimum capacitance level* corresponding to the amount of energy required and then enters a deep-sleep mode keeping *REACT*'s capacitor polling time active. As the system charges, *REACT* eventually accumulates enough energy to reach the minimum capacitance level—indicating that enough energy is stored to complete the planned operation, and pulling the system out of its deep-sleep with enough energy to complete execution regardless of future power conditions.

# 4 Implementation

We explore *REACT*'s impact on overall efficiency, reactivity, and longevity using a hardware prototype integrated into a real batteryless platform. Our testbed is based on the MSP430FR5994 [23], a popular microcontroller for energy harvesters [30, 48, 49]. For each buffer configuration we evaluate, an intermediate circuit power gates the MSP430 to begin operation once the buffer is charged to 3.3V and disconnects it when the buffer voltage reaches 1.8V.

Our *REACT* implementation has a range of 770 $\mu F$-18.03 $mF$ using a set of 5 dynamic banks, in addition to the last-level buffer, detailed in Table 1. We implement the capacitors in banks 0-4 using combinations of 220 $\mu F$ capacitors with max leakage current of 28 $\mu A$ at their rated voltage of 6.3V [35]. Bank 5 uses supercapacitors with approximately 0.15 $\mu A$ leakage current at 5.5V [36].

## 4.1 Baseline Systems

We evaluate *REACT* against three fixed-size buffers spanning our implementation's capacitance range—770 μF, 10 mF, and 18 mF—to ensure the realized improvement is a result of energy-adaptive behavior rather than simply different buffer capacity. To compare *REACT*'s capacitor architecture to prior work on dynamic energy buffers, we also implement and evaluate Morphy [53] for a similar capacitance range. Our Morphy implementation uses eight 2 $mF$ capacitors with leakage current of approximately 25.2 $\mu A$ at 6.3V [38] (i.e., slightly lower leakage than the capacitors in *REACT*).

Morphy uses a secondary microcontroller powered by a battery or backup capacitor to control the capacitor array; we use a second MSP430FR5994 powered through USB, corresponding to Morphy's battery-powered design. Accordingly, we expect our results to slightly overestimate Morphy's performance in the fully-batteryless case as the system does not have to power the Morphy controller or charge a backup capacitor in our implementation. Seven of the eight capacitors in the array are available to reconfigure, with one task capacitor kept in parallel to smooth voltage fluctuations from switching. We evaluate the same subset of eleven possible configurations for the remaining seven capacitors as is done in the original Morphy work, resulting in a capacitance range for our Morphy implementation of 250 $\mu F$-16 $mF$.

## 4.2 Computational Backend

To explore how *REACT* affects performance across a range of system demands—focusing on diverse reactivity and longevity requirements—we implement four software benchmarks:

- **Data Encryption (DE)**: Continuously perform AES-128 encryptions in software. This application has no reactivity requirements, low persistence requirements, and a predictable power draw; we use it as a baseline to explore *REACT*'s software and power overhead. Figure of Merit: Number of AES encryptions completed, not including encryptions interrupted by power loss.
- **Sense and Compute (SC)**: Exit a deep-sleep mode once every five seconds second to sample and digitally filter readings from a microphone [13]. This benchmark represents systems which value high reactivity

and can accept low persistence; individual atomic measurements are low-energy, but the system must be online to take the measurements. Figure of Merit: Number of sensing tasks completed, not including samples or processing interrupted by power loss.

- **Radio Transmission (RT)**: Send buffered data over radio [20, 34] to a base station. Data transmission is an example of an application with high persistence requirements (radio transmissions are atomic and energy-intensive) and low reactivity requirements (transmitting data may be delayed until energy is available). Figure of Merit: Number of transmissions completed, not including transmissions interrupted by power loss.
- **Packet Forwarding (PF)**: Listen for and retransmit incoming data over the radio. Timely operation demands both high persistence and reactivity to successfully receive and retransmit data. Figures of Merit: Packets received and transmitted, not including operations interrupted by power loss.

We emulate the power consumption of the necessary peripherals for each benchmark by toggling a resistor connected to a digital output on the MSP430, with values for each benchmark chosen to match the relevant peripheral. The reactivity-focused benchmarks (SC and PF) have deadlines that may arrive while the system is off; we use a secondary MSP430 to deliver these events. A deployed system may use remanence-based timekeepers [9] to track internal deadlines despite power failures for the SC benchmark, while incoming packets as in the PF benchmark would arrive from other systems. Although we evaluate each benchmark in isolation, full systems are likely to exercise combinations of each requirement—one platform should support all reactivity, persistence, and efficiency requirements.

### 4.3 Energy Harvesting Frontend

Energy volatility makes repeatable experimentation with batteryless devices difficult; uncontrollable environmental changes often have an outsized effect on energy input and obfuscate differences in actual system performance. We make our experiments repeatable and consistent using a programmable power frontend inspired by the Ekho [16] record-and-replay platform. The power controller supplies the energy buffer using a high-drive Digital-to-Analog Converter (DAC), measures the load voltage and input current using a sense resistor, and tunes the DAC to supply a programmed power level. We evaluate *REACT* emulating both solar (5 $cm^2$, 22% efficient cell [47]) and RF energy (915 MHz dipole antenna [42]).

## 5 Evaluation

We evaluate *REACT* alongside the baseline buffers running each benchmark under three RF and two solar traces from publicly available repositories [3, 14], representative of power dynamics for small energy harvesting systems. We record the RF traces in an active office environment using a commercial harvester and transmitter [40, 41] and use solar irradiance traces from the Enhants mobile irradiance dataset [14]; Table 3 gives a short summary of each trace. These traces show the power variability common for IoT-scale harvesters: environmental changes (e.g., ambient RF levels, time of day) affect *average* input power, while short-term changes such as orientation cause instantaneous variation even if the environment is unchanged. We apply each trace using the power replay system described in § 4.3; once the trace is complete, we let the system run until it drains the buffer capacitor.

### 5.1 Software and Energy and Overhead

Figure 6 illustrates *REACT*'s behavior through the last-level buffer voltage when varying capacitance; the inset focuses on *REACT*'s voltage output as it expands to capture energy (also shown is the voltage of the comparable Morphy array). From a cold start *REACT* only charges the last-level buffer—rapidly reaching the enable voltage and then the upper voltage threshold (3.5V). *REACT* then adds a series-configured capacitor bank to capture excess incoming energy. Voltage drops as the system temporarily operates exclusively from the last-level buffer while harvested energy goes towards charging the new capacitance. As power input falls, *REACT*'s output voltage falls below the upper threshold voltage—indicating *REACT* is operating at an efficient capacitance point. At $t \approx 450s$ the last-level buffer is discharged to the lower threshold and *REACT* begins switching banks into series mode to boost their output voltage and charge the last-level buffer, visible in Figure 6 as five voltage spikes corresponding to each capacitor bank—sustaining operation until no more energy is available at $t \approx 500s$.

We characterize *REACT*'s software overhead by running the DE benchmark on continuous power for 5 minutes with and without *REACT*'s software component, which periodically interrupts execution to measure the capacitor bank. At a sample rate of 10 Hz, *REACT* adds a 1.8% penalty to software-heavy applications. We measure *REACT*'s power overhead by comparing the execution time of systems running the DE benchmark using *REACT* and the 770 $\mu F$ buffer after charging each to their enable voltage. Based on this approach we estimate that our implementation of *REACT* introduces a 68 $\mu W$ power draw, or ~14 $\mu W$ per bank.

### 5.2 Characterization

Table 4 details the cost at scale of *REACT* compared to Morphy for a capacitor network consisting of eight 100 $mF$ supercapacitors. Considering the same supercapacitors and switch ICs as in the original Morphy implementation [53], *REACT* using four two-capacitor banks is approximately 25% less expensive than Morphy owing to its simpler capacitor switching structure: *REACT* requires one Double-Pole-Double-Throw (DPDT) and one Single-Pole-Single-Throw (SPST) switch per *bank*, corresponding to six switch ICs (as

| | Data Encrypt | | | | | Sense and Compute | | | | | Radio Transmit | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Buffer | 770$\mu$ | 10m | 17m | Morphy | REACT | 770$\mu$ | 10m | 17m | Morphy | REACT | 770$\mu$ | 10m | 17m | Morphy | REACT |
| RF Cart | 1275 | 1574 | 1831 | 1745 | 1711 | 50 | 81 | 104 | 77 | 83 | 22 | 53 | 56 | 38 | 48 |
| RF Obs. | 666 | 472 | 0 | 357 | 576 | 44 | 28 | 0 | 39 | 49 | 4 | 6 | 0 | 0 | 3 |
| RF Mob. | 810 | 1004 | 645 | 801 | 1038 | 52 | 50 | 40 | 53 | 84 | 4 | 13 | 12 | 4 | 15 |
| Sol. Camp. | 6666 | 7290 | 7936 | 8194 | 9756 | 330 | 353 | 367 | 398 | 439 | 1376 | 1457 | 1542 | 1059 | 1426 |
| Sol. Comm. | 2168 | 2186 | 2554 | 2399 | 2232 | 88 | 110 | 130 | 133 | 154 | 8 | 40 | 48 | 31 | 34 |
| Mean | 2317 | 2505 | 2593 | 2699 | 3063 | 113 | 124 | 128 | 140 | 162 | 283 | 314 | 332 | 226 | 313 |

**Table 2.** Performance by figure of merit on the DE, SC, and RT benchmarks, across traces and energy buffers.

| Trace | Time (s) | Avg. Pow. (mW) | Power CV* |
|---|---|---|---|
| RF Cart | 313 | 2.12 | 103% |
| RF Obstruction | 313 | 0.227 | 61% |
| RF Mobile | 318 | 0.5 | 166% |
| Solar Campus | 3609 | 5.18 | 207% |
| Solar Commute | 6030 | 0.148 | 333% |

**Table 3.** Details of each power trace. *CV = Coefficient of Variation.

| Device | 1000x | Morphy | | REACT | |
|---|---|---|---|---|---|
| | Unit Cost | Count | Cost | Count | Cost |
| Supercap. [36] | \$1.54 | 9 | \$13.87 | 8 | \$12.33 |
| Switch IC [12] | \$1.53 | 12 | \$18.36 | 6 | \$9.18 |
| Ideal Diode [22] | \$0.27 | 1 | \$0.27 | 8 | \$2.16 |
| Ceramic Cap. [35] | \$0.45 | 0 | \$0 | 4 | \$1.82 |
| Comparator IC [24] | \$0.77 | 0 | \$0 | 1 | \$0.77 |
| Detector IC [44] | \$0.26 | 0 | \$0 | 2 | \$0.52 |
| Control MCU [33] | \$2.37 | 1 | \$2.37 | 0 | \$0 |
| RTC [8] | \$1.08 | 1 | \$1.08 | 0 | \$0 |
| **Platform Cost** | | | **\$35.94** | | **\$26.77** |

**Table 4.** Cost comparison between *REACT* and Morphy implementations targeting the same capacitance range.

| Buffer | 770 $\mu F$ | 10 mF | 17 mF | Morphy | REACT |
|---|---|---|---|---|---|
| RF Cart | 6.65 | 17.73 | 31.27 | 5.51 | 6.65 |
| RF Obs. | 14.58 | 223.07 | - | 6.50 | 16 |
| RF Mob. | 6.90 | 148.10 | 239.88 | 5.65 | 6.38 |
| Sol. Camp. | 42.11 | 737.39 | 741.42 | 35.59 | 41.26 |
| Sol. Comm. | 119.60 | 196.30 | 213.00 | 108.10 | 130.6 |
| Mean | 37.97 | 264.92 | 306.39 | 32.27 | 40.18 |

**Table 5.** System latency (seconds) across traces and energy buffers. - indicates system never begins operation.

each IC contains two SPDT switches which can be combined to form one DPDT switch). The Morphy network requires one DPDT and one SPST switch per *capacitor*, plus an additional SPST switch to fully connect the network. The battery-free version of Morphy requires an additional capacitor and diode to power the Morphy controller; following the original Morphy design, we use the same type of capacitor as in the switching network. Other cost differences (i.e., *REACT*'s diodes and voltage instrumentation versus Morphy's secondary microcontroller and real-time clock) are relatively minor as capacitor and switch costs dominate overall cost.

Designers can tune the structure of the capacitor network to vary aspects of *REACT*'s performance. One alternative for the eight-capacitor network described above is to organize *REACT* using two four-capacitor banks; this design does not significantly change unit cost (each four-capacitor bank requires one SPST and three DPDT switches, and the overall system requires half as many diodes) and increases efficiency by extracting more energy from each capacitor bank (§3.3.4). The tradeoff is that using fewer and larger banks reduces the number of possible capacitance configurations and requires increasing the size of the last-level buffer to smooth voltage spikes, limiting reactivity from a cold start. In practice, application demands will determine the ideal *REACT* network and corresponding balance of efficiency and flexibility.

### 5.3 *REACT* Minimizes System Latency

Table 5 details the time it takes each system to begin operation, across power traces and energy buffers (charge time is software-invariant and constant across benchmarks). Latency is driven by both capacitor size and environment—the 10$mF$ buffer is ~13$x$ larger than the 770$\mu F$ buffer and takes on average 7$x$ longer to activate the system across our traces. High-capacity static buffers incur a larger latency penalty even if mean power input is high if much of that power is contained in a short-term spike later in the trace (e.g., for the Solar Campus trace), but these dynamics are generally impossible to predict at design time. By exclusively charging the last-level buffer while the rest of the system is off, *REACT* matches the latency of the smallest static buffer—an average of 7.7$x$ faster than the equivalent-capacity 17 mF buffer, which risks failing to start at all. Morphy further reduces system latency because its smallest configuration is smaller than *REACT*'s last-level buffer (250 $\mu F$ vs 770 $\mu F$), although the limited reduction in average latency compared to the reduction in capacitance (Morphy realizes an average 20% reduction in latency over *REACT* using a 68% smaller capacitance) suggests that further reducing capacitance yields diminishing latency returns in realistic energy environments.

Minimizing latency improves reactivity-bound applications such as the SC and PF benchmarks; this effect is visible in Table 2 as the 770 $\mu F$ buffer outperforms larger ones in the SC benchmark for relatively low-power traces (RF Mobile/Obstructed). *REACT* inherits the latency advantage

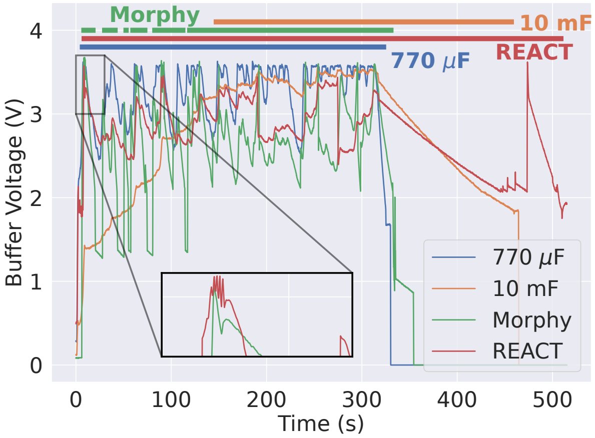


**Figure 6.** Buffer voltage and on-time for the SC benchmark under RF Mobile power. Solid bars indicate when the system is operating.

due to the small last-level buffer, similarly improving performance on each power trace. Morphy realizes a similar performance improvement over the static systems, but ultimately underperforms *REACT* as a result of inefficient capacitor switching (§ 5.6). Small static buffers enable low-latency operation, but at the cost of energy capacity. As power input increases, the latency penalty of large buffers fades and their increased capacity enables them to operate for longer—resulting in higher performance for larger static buffers under high-power traces (RF Cart, Solar Campus). Smaller buffers, in turn, become less efficient as they must burn more incoming energy off as waste heat.

### 5.4 *REACT* Maximizes Energy Capacity

Figure 6 illustrates the system-level effects of the capacity-latency tradeoff, and how *REACT* avoids this tradeoff through energy-adaptive buffering. The small 770$\mu F$ buffer charges rapidly, but reaches capacity and discharges energy when it does not have work to match incoming power (illustrated by clipping at 3.6V on the 770 $\mu$F line). The 10 mF buffer sacrifices latency for capacity—starting operation 21x later than the smaller buffer, but avoiding overvoltage. Morphy begins execution early with a small capacitance, but its lossy switching mechanism means it does not overall outperform the 770$\mu F$ buffer. In contrast, *REACT* achieves low latency, high efficiency, *and* high capacity by efficiently expanding capacitance as necessary after enabling the system.

Tables 2 and 6 show that high capacity is valuable when average input power exceeds output power (e.g., DE and SC benchmarks executed under the RF Cart trace), or when peak power demand is uncontrollable and uncorrelated with input (e.g., the PF benchmark executed on Solar Campus, where both power supply and demand are concentrated in short bursts). In both cases, high-capacity systems store excess energy to continue operation even if future power input falls or demand rises. *REACT* efficiently expands to capture all incoming energy during periods of high net input power, matching or beating the performance of the 10 mF and 17 mF systems when they outperform the small 770 $\mu F$ buffer.

| Buffer | 770 uF | | 10 mF | | 17 mF | | Morphy | | REACT | |
|---|---|---|---|---|---|---|---|---|---|---|
| Packets | Rx | Tx | Rx | Tx | Rx | Tx | Rx | Tx | Rx | Tx |
| RF Cart | 22 | 10 | 49 | 49 | 48 | 48 | 55 | 22 | 53 | 52 |
| RF Obs. | 4 | 4 | 4 | 4 | 0 | 0 | 2 | 0 | 3 | 0 |
| RF Mob. | 11 | 4 | 14 | 13 | 9 | 9 | 19 | 0 | 38 | 5 |
| Sol. Camp. | 163 | 163 | 240 | 240 | 196 | 196 | 206 | 204 | 284 | 277 |
| Sol. Comm. | 72 | 8 | 35 | 35 | 33 | 33 | 85 | 14 | 84 | 63 |
| Mean | 54 | 38 | 68 | 68 | 57 | 57 | 73 | 48 | 92 | 80 |

**Table 6.** Packets successfully received and retransmitted during the Packet Forwarding benchmark.

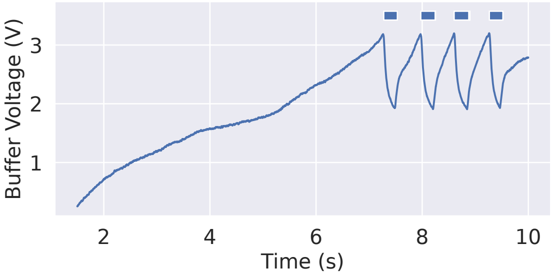


**Figure 7.** Buffer voltage and on-time for the 770 $\mu$F system running the RT benchmark under the RF Cart trace. Solid bars indicate when the system is operating.

### 5.5 *REACT* Provides Flexible, Efficient Longevity

We evaluate *REACT*'s software-directed longevity guarantees (§ 3.4.1) on the longevity-bound RT and PF benchmarks. We compare *REACT* to the 770 $\mu F$ buffer, which cannot sustain a full transmission without additional input power. Figure 7 illustrates this limitation, as the 770 $\mu$F system charges quickly but wastes power on doomed-to-fail transmissions when incoming power cannot make up for the deficit and the system powers off before completing a transmission. This shortcoming is reflected in Table 2, where the 770 $\mu$F system ultimately significantly underperforms the other buffers. We augment the RT benchmark code for our *REACT* implementation to include a minimum capacitance level for *REACT*, below which the system waits for more energy in a low-power sleep mode. Leveraging *REACT*'s variable capacitance allows software to buffer energy to guarantee completion, more than doubling the number of successful transmissions and ultimately outperforming even the larger buffers.

We use the same approach to execute the RT benchmark on our Morphy implementation. Similar to *REACT*, Morphy varies capacitance to keep supply voltage within an acceptable level for the application microcontroller while also waiting to gather enough energy to power a full transmission. Morphy's underperformance compared to both *REACT* and the static buffers is a result of Morphy's capacitor network design—as Morphy reconfigures the capacitor array to increase capacitance, stored energy is dissipated as current flows between capacitors in the network. This energy dissipation dramatically reduces Morphy's end-to-end performance, particularly in systems where Morphy *must* switch capacitance to ensure success (i.e., the RT and PF benchmarks). *REACT*'s isolated capacitor banks eliminate this problem by restricting current flow during switching; the energy savings are reflected in the end-to-end performance, where *REACT* completes on average 38% more transmissions than Morphy.

**5.5.1 Fungible Energy Storage.** A unified buffer means that energy is fungible, and *REACT* is flexible: software can re-define or ignore previous longevity requirements if conditions or priorities change. The PF benchmark (Table 6) shows the value of energy fungibility using two tasks with distinct reactivity and longevity requirements. Receiving an incoming packet requires a moderate level of longevity, but is uncontrollable and has a strict reactivity requirement (the system can only receive a packet exactly when it arrives). Re-transmission requires more energy but has no deadline. Software must effectively split energy between a controllable high-power task and an uncontrollable lower-power task.

We again use the minimum-capacitance approach to set separate longevity levels for each task, using a similar approach for our Morphy implementation. When the system has no packets to transmit, it waits in a deep-sleep until receiving an incoming packet. If *REACT* contains sufficient energy when the packet arrives, it receives and buffers the packet to later send. *REACT* then begins charging for the transmit task, forwarding the buffered packet once enough energy is available. If another packet is received while *REACT* is charging for the transmit task, however, software disregards the transmit-associated longevity requirement to execute the receive task if sufficient energy is available.

Table 6 shows that *REACT* outperforms all static buffer designs on the PF benchmark by efficiently addressing the requirements of both tasks, resulting in a mean performance improvement of 54%. *REACT*'s maximal reactivity enables it to turn on earlier and begin receiving and buffering packets to send during later periods of high power, while its high capacity enables it to make the most of incoming energy during those high periods. Software-level longevity guarantees both ensure the system only begins receive/transmit operations when enough energy is available to complete them, and that software can effectively allocate energy to incoming events as needed. Although Morphy enables the same software-level control of energy allocation, the energy dissipated when switching capacitors in the interconnected array reduce its performance on the PF benchmark to below that of the best performing static buffer.

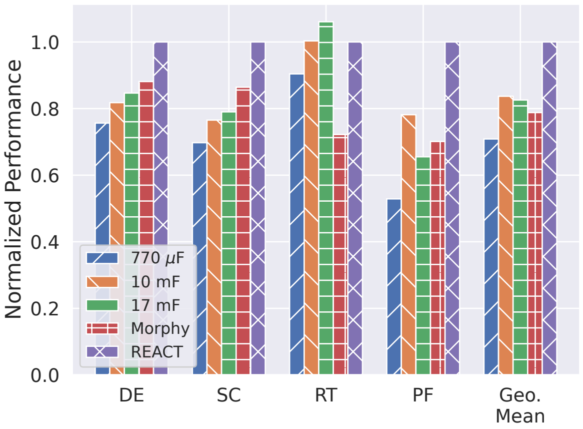


**Figure 8.** Average buffer performance quantified by figures of merit across power traces for each benchmark, normalized to *REACT*.

### 5.6 *REACT* Improves End-to-End System Efficiency

Optimizing buffer behavior maximizes energy available to the end system for useful work. Figure 8 illustrates the aggregate performance of *REACT* compared to the baselines across the benchmarks and power traces we evaluate; we find that *REACT* improves performance over the equally-reactive 770 $\mu F$ buffer by an average of 39.1%, over the equal-capacity 17 mF buffer by 19.3%, and over the next-best-efficient 10 mF buffer by 18.8%. Compared to Morphy, *REACT* improves aggregate performance by 26.2%—demonstrating the necessity of *REACT*'s bank isolation approach and boosting performance where the state of the art underperforms static approaches. In extreme cases where the system is *always* operating in an energy surplus or deficit, *REACT*'s quiescent power consumption causes it to underperform suitable static buffers. In the common case, however, volatile power conditions expose the latency, longevity, and efficiency-related shortcomings of static buffer designs and expose the value of *REACT*'s efficient variable-capacitance approach.

## 6 Conclusion

Batteryless systems operate on unreliable and volatile power, but use fixed-size buffers which waste energy and functionally limit systems when allocated capacity is a poor fit for short-term power dynamics. *REACT* stores energy in a fabric of reconfigurable capacitor banks, varying equivalent capacitance according to current energy supply and demand—adding capacitance to capture surplus power and reclaiming energy from excess capacitance. *REACT*'s energy-adaptive approach maximizes reactivity and capacity to ensure all incoming energy is captured and efficiently delivered to sensing, computing, and communication devices. Our hardware evaluation on real-world power traces shows that *REACT* reduces system latency by an average of 7.7x compared to an equivalent-sized static buffer and improves throughput by an average of 25.6% over *any* static buffer system, while incorporating software direction allows *REACT* to provide flexible task longevity guarantees. Compared to state-of-the-art switched capacitor systems, *REACT*'s efficient switching architecture improves performance by an average of 26.2%.

*REACT*'s runtime-configurable buffering technique eliminates the tradeoff between system latency and longevity, and affords designers greater control over how batteryless devices respond to incoming power. Our results indicate that energy-responsive reconfiguration of hardware is an effective approach to both maximizing energy efficiency and system functionality, opening the door for future work leveraging energy-adaptive hardware and reconfiguration.

## 7 Acknowledgments

We thank our shepherd Kiwan Maeng for his guidance and the anonymous reviewers for their helpful suggestions. The project depicted is sponsored by the Defense Advanced Research Projects Agency. The content of the information does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. Approved for public release; distribution is unlimited.

## A Artifact Appendix

### A.1 Abstract

This artifact contains the source code for the software component of *REACT*, baseline platforms and application benchmarks. All code is publicly available as a GitHub repository and on the open-access repository Zenodo. This artifact is for *availability only*: the full performance evaluation requires building the circuits described in §4 and modifying source code according to the specific hardware configuration.

### A.2 Artifact check-list (meta-information)

- **Algorithm:** Energy and dynamic capacitance management library.
- **Program:** Custom-developed embedded benchmarks.
- **Compilation:** Texas Instruments MSP430 C Compiler as part of Code Composer Studio version 11. Not included.
- **Data set:** Harvested RF power traces.
- **Run-time environment:** Developed and tested on Ubuntu 20.04.
- **Hardware:** Texas Instruments MSP430FR5994.
- **How much disk space required (approximately)?:** 30 MB.
- **How much time is needed to prepare workflow (approximately)?:** 30 minutes.
- **How much time is needed to complete experiments (approximately)?:** 60 hours.
- **Publicly available?:** Yes.
- **Code licenses (if publicly available)?:** MIT License
- **Data licenses (if publicly available)?:** MIT License
- **Archived (provide DOI)?:** https://doi.org/10.5281/zenodo.10798308

### A.3 Description

**A.3.1 How to access.** *REACT*'s software component is publicly available through its git repository at https://github.com/FoRTE-Research/REACT-Artifact and on Zenodo at https://zenodo.org/records/10798308.

**A.3.2 Hardware dependencies.** *REACT*'s software component and the baseline systems for evaluation target the Texas Instruments MSP430FR5994 and were developed using the MSP-EXP430FR5994 development kit.

**A.3.3 Software dependencies.** Texas Instruments MSP430 C compiler included with Code Composer Studio.

**A.3.4 Data sets.** The ambient RF power traces used in the evaluation are provided and described in the artifact. The solar power traces are part of the Enhants [14] dataset.

### A.4 Installation

***REACT Software Component:*** Clone the artifact repository from https://github.com/FoRTE-Research/REACT-Artifact. Benchmarks with *REACT*'s runtime library integrated are included as Code Composer Studio projects.

***Code Composer Studio:*** We do not provide CCS as part of the artifact repository; install it by following the instructions at https://www.ti.com/tool/CCSTUDIO.

### A.5 Evaluation and expected results

Clone the artifact repository from https://github.com/FoRTE-Research/REACT-Artifact, or download it from Zenodo at https://zenodo.org/records/10798308. Each benchmark can be imported into Code Composer Studio as a project targeting the MSP430FR5994 and compiled to produce a binary.

### A.6 Methodology

Submission, reviewing and badging methodology:

- https://www.acm.org/publications/policies/artifact-review-badging
- http://cTuning.org/ae/submission-20201122.html
- http://cTuning.org/ae/reviewing-20201122.html

## References

[1] Saad Ahmed, Qurat ul Ain, Junaid Haroon Siddiqui, Luca Mottola, and Muhammad Hamad Alizai. Intermittent computing with dynamic voltage and frequency scaling. In *Proceedings of the 2020 International Conference on Embedded Wireless Systems and Networks*, EWSN '20, page 97–107, USA, 2020. Junction Publishing.

[2] Miran Alhaideri, Michael Rushanan, Denis Foo Kune, and Kevin Fu. The moo and cement shoes: Future directions of a practical sense-control-actuate application, September 2013. Presented at First International Workshop on the Swarm at the Edge of the Cloud (SEC'13 @ ESWeek), Montreal.

[3] Anon. Rf traces, October 2022. https://anonymous.4open.science/r/rf_traces-4B3E/README.md.

[4] Abu Bakar, Alexander G. Ross, Kasim Sinan Yildirim, and Josiah Hester. Rehash: A flexible, developer focused, heuristic adaptation platform for intermittently powered computing. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 5(3), sep 2021.

[5] James Blackman. What is mmtc in 5g nr, and how does it impact nb-iot and lte-m, October 2019. https://enterpriseiotinsights.com/20191016/channels/fundamentals/what-is-mmtc-in-5g-nr-and-how-does-it-impact-nb-iot-and-lte-m.

[6] Michael Buettner, Ben Greenstein, and David Wetherall. Dewdrop: An energy-aware runtime for computational rfid. In *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, NSDI'11, page 197–210, USA, 2011. USENIX Association.

[7] Alexei Colin, Emily Ruppel, and Brandon 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*, ASPLOS '18, page 767–781, New York, NY, USA, 2018. Association for Computing Machinery.

[8] Abracon Corporation. AB08X5 Real-Time Clock Family, October 2014. https://abracon.com/Precisiontiming/AB08X5-RTC.PDF.

[9] Jasper de Winkel, Carlo Delle Donne, Kasim Sinan Yildirim, Przemysław Pawełczak, and Josiah Hester. Reliable timekeeping for intermittent computing. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '20, page 53–67, New York, NY, USA, 2020. Association for Computing Machinery.

[10] Jasper de Winkel, Vito Kortbeek, Josiah Hester, and Przemysław Pawełczak. Battery-free game boy. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 4(3), sep 2020.

[11] H. Desai and B. Lucia. A power-aware heterogeneous architecture scaling model for energy-harvesting computers. *IEEE Computer Architecture Letters*, 19(1):68–71, 2020.

[12] Analog Devices. 0.5 Ω CMOS 1.65 V to 3.6 V Dual SPDT/2:1 Mux in Mini LFCSP Package, March 2012. https://www.analog.com/media/en/technical-documentation/data-sheets/adg824.pdf.

[13] Knowles Electronics. SPU0414HR5H-SB, December 2012. https://www.mouser.com/datasheet/2/218/knowles_01232019_SPU0414HR5H_SB-1891952.pdf.

[14] M. Gorlatova, A. Wallwater, and G. Zussman. Networking low-power energy harvesting devices: Measurements and algorithms. In *2011 Proceedings IEEE INFOCOM*, pages 1602–1610, 2011.

[15] Wang Song Hao and Ronald Garcia. Development of a digital and battery-free smart flowmeter. *Energies*, 7(6):3695–3709, 2014.

[16] Josiah Hester, Timothy Scott, and Jacob Sorber. Ekho: Realistic and repeatable experimentation for tiny energy-harvesting sensors. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, SenSys '14, page 330–331, New York, NY, USA, 2014. Association for Computing Machinery.

[17] Josiah Hester, Lanny Sitanayah, and Jacob Sorber. Tragedy of the coulombs: Federating energy storage for tiny, intermittently-powered sensors. In *ACM Conference on Embedded Networked Sensor Systems*, SenSys, pages 5–16, 2015.

[18] Josiah Hester and Jacob Sorber. Flicker: Rapid prototyping for the batteryless internet-of-things. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, SenSys '17, New York, NY, USA, 2017. Association for Computing Machinery.

[19] Matthew Hicks. Clank: Architectural support for intermittent computation. In *International Symposium on Computer Architecture*, ISCA, pages 228–240, 2017.

[20] Fraunhofer IIS. RFicient Basic, Ultra-Low-Power WakeUp Receiver, January 2019. https://www.iis.fraunhofer.de/content/dam/iis/en/doc/il/ics/ic-design/Datenblaetter/Factsheet_WakeUp_v4.pdf.

[21] Texas Instruments. Msp430g2x52, msp430g2x12 mixed signal microcontroller datasheet (rev. g), May 2013. https://www.ti.com/lit/ds/symlink/msp430g2252.pdf.

[22] Texas Instruments. Lm66100 5.5-v, 1.5-a 79-milliohm, low iq ideal diode with input polarity protection, June 2019. https://www.ti.com/lit/ds/symlink/lm66100.pdf.

[23] Texas Instruments. MSP430FR599x, MSP430FR596x Mixed-Signal Microcontrollers, January 2021. https://www.ti.com/lit/ds/symlink/msp430fr5994.pdf.

[24] Texas Instruments. Tlv703x and tlv704x small-size, nanopower, low-voltage comparators, July 2021. https://www.ti.com/lit/ds/symlink/tlv7044.pdf.

[25] Joseph Kahn, Randy Katz, and Kristofer Pister. Next Century Challenges: Mobile Networking for "Smart Dust". In *Conference on Mobile Computing and Networking (MobiCom)*, 1999.

[26] Kemet. Supercapacitors fm series, July 2020. https://www.mouser.com/datasheet/2/212/1/KEM_S6012_FM-1103835.pdf.

[27] Silicon Labs. EFM32 Gecko Family EFM32WG Data Sheet, December 2021. https://www.silabs.com/documents/public/data-sheets/efm32wg-datasheet.pdf.

[28] Brandon Lucia and Benjamin Ransford. A simpler, safer programming and execution model for intermittent systems. In *Conference on Programming Language Design and Implementation*, PLDI, pages 575–585, 2015.

[29] K. Ma, Y. Zheng, S. Li, K. Swaminathan, X. Li, Y. Liu, J. Sampson, Y. Xie, and V. Narayanan. Architecture exploration for ambient energy harvesting nonvolatile processors. In *IEEE International Symposium on High Performance Computer Architecture*, HPCA, pages 526–537, Feb 2015.

[30] Kiwan Maeng, Alexei Colin, and Brandon Lucia. Alpaca: Intermittent execution without checkpoints. In *International Conference on Object-Oriented Programming, Systems, Languages, and Applications*, OOPSLA, pages 96:1–96:30, October 2017.

[31] Kiwan Maeng and Brandon Lucia. Adaptive dynamic checkpointing for safe efficient intermittent computing. In *USENIX Conference on Operating Systems Design and Implementation*, OSDI, pages 129–144, November 2018.

[32] Kiwan Maeng and Brandon Lucia. Adaptive low-overhead scheduling for periodic and reactive intermittent execution. In *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI 2020, page 1005–1021, New York, NY, USA, 2020. Association for Computing Machinery.

[33] Microchip. PIC18(L)F65/66K40, November 2017. https://ww1.microchip.com/downloads/en/DeviceDoc/40001842D.pdf.

[34] Microsemi. ZL70251 Ultra-Low-Power Sub-GHz RF Transceiver, March 2018. https://www.microsemi.com/document-portal/doc_view/132900-zl70251-datasheet.

[35] Murata. GRM31CR60J227ME11L Chip Monolithic Ceramic Capacitor for General. https://search.murata.co.jp/Ceramy/image/img/A01X/G101/ENG/GRM31CR60J227ME11-01.pdf.

[36] Murata. Supercapacitors FM Series, July 2020. https://www.mouser.com/datasheet/2/212/1/KEM_S6012_FM-1103835.pdf.

[37] Phillip Nadeua, Dina El-Damaj, Deal Glettig, Yong Lin Kong, Stacy Mo, Cody Cleveland, Lucas Booth, Niclas Roxhed, Robert Langer, Anantha P. Chandrakasan, and Giovanni Traverso. Prolonged energy harvesting for ingestible devices. *Nature Biomedical Engineering*, 1(0022), Feb 2017.

[38] Nichicon. ALUMINUM ELECTROLYTIC CAPACITORS. https://www.nichicon.co.jp/english/products/pdfs/e-kl.pdf.

[39] Panasonic. Panasonic coin type lithium batteries, August 2005. https://datasheet.octopart.com/CR1616-Panasonic-datasheet-9751741.pdf.

[40] Powercast. P2110B 915 MHz RF Powerharvester Receiver, December 2016. https://www.powercastco.com/wp-content/uploads/2016/12/P2110B-Datasheet-Rev-3.pdf.

[41] Powercast. TX91501B – 915 MHz Powercaster Transmitter, October 2019. https://www.powercastco.com/wp-content/uploads/2019/10/User-Manual-TX-915-01B-Rev-A-1.pdf.

[42] Powercast. 915 mhz dipole antenna datasheet, November 2020. https://www.powercastco.com/wp-content/uploads/2020/11/DA-915-01-Antenna-Datasheet_new_web.pdf.

[43] Benjamin Ransford, Jacob Sorber, and Kevin Fu. Mementos: System Support for Long-Running Computation on RFID-Scale Devices. In *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2011.

[44] ROHM Semiconductor. Low Voltage Standard CMOS Voltage Detector ICs, October 2021. https://fscdn.rohm.com/en/products/databook/datasheet/ic/power/voltage_detector/bu48xxg-e.pdf.

[45] Henry Sodano, Gyuhae Park, and Daniel Inman. Estimation of Electric Charge Output for Piezoelectric Energy Harvesting. In *Strain, Volume 40*, 2004.

[46] ST. Small signal schottky diode, October 2001. https://www.st.com/content/ccc/resource/technical/document/datasheet/group1/11/76/e4/a3/df/07/49/14/CD00000767/files/CD00000767.pdf/jcr:content/translations/en.CD00000767.pdf.

[47] Voltaic. Voltaic systems p121 r1g, April 2020. https://voltaicsystems.com/content/VoltaicSystemsP121R1G.pdf.

[48] Harrison Williams, Xun Jian, and Matthew 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*, ASPLOS '20, page 69–84, New York, NY, USA, 2020. Association for Computing Machinery.

[49] Harrison Williams, Michael Moukarzel, and Matthew Hicks. Failure sentinels: Ubiquitous just-in-time intermittent computation via low-cost hardware support for voltage monitoring. In *International Symposium on Computer Architecture*, ISCA, pages 665–678, 2021.

[50] Joel Van Der Woude and Matthew Hicks. Intermittent computation without hardware support or programmer intervention. In *USENIX Symposium on Operating Systems Design and Implementation*, OSDI, pages 17–32, November 2016.

[51] X. Wu, I. Lee, Q. Dong, K. Yang, D. Kim, J. Wang, Y. Peng, Y. Zhang, M. Saliganc, M. Yasuda, K. Kumeno, F. Ohno, S. Miyoshi, M. Kawaminami, D. Sylvester, and D. Blaauw. A 0.04mm316nw wireless and batteryless sensor system with integrated cortex-m0+ processor and optical communication for cellular temperature measurement. In *2018 IEEE Symposium on VLSI Circuits*, pages 191–192, 2018.

[52] Fan Yang, Ashok Samraj Thangarajan, Wouter Joosen, Christophe Huygens, Danny Hughes, Gowri Sankar Ramachandran, and Bhaskar Krishnamachari. Astar: Sustainable battery free energy harvesting for heterogeneous platforms and dynamic environments. In *Proceedings of the 2019 International Conference on Embedded Wireless Systems and Networks*, EWSN '19, page 71–82, USA, 2019. Junction Publishing.

[53] Fan Yang, Ashok Samraj Thangarajan, Sam Michiels, Wouter Joosen, and Danny Hughes. Morphy: Software defined charge storage for the iot. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, SenSys '21, page 248–260, New York, NY, USA, 2021. Association for Computing Machinery.