An Architectural Charge Management Interface for Energy-Harvesting Systems

Emily Ruppel[*‡], Milijana Surbatovich[†], Harsh Desai[†], Kiwan Maeng[§], Brandon Lucia[†]
[*] Bosch Research and Technology Center [§]Pennsylvania State University [†] Carnegie Mellon University
emily.ruppel@us.bosch.com kmaeng91@gmail.com {milijans, harshd, blucia}@andrew.cmu.edu

***Abstract*—Energy-harvesting devices eliminate batteries, instead collecting their operating energy from environmental sources. A device stores energy into a capacitor, drawing energy to perform tasks and powering off to recharge when the energy is exhausted. State-of-the-art charge management systems for these devices aim to avoid power failure during task execution by reasoning about task energy cost. We identify that the innate equivalent series resistance (ESR) in energy storage capacitors breaks energy-based systems' guarantees; running high current load on a high-ESR capacitor causes a substantial voltage drop that rebounds once the load is removed. This voltage drop is disregarded by systems that only reason about energy. If the drop lowers the voltage below the system's operating threshold, however, the device powers off *while stored energy remains.* Though ESR is well understood in hardware design, this is the first work to argue that software for batteryless devices must also be aware of ESR.**

This work presents Culpeo, a hardware/software mechanism and architectural interface to relay the effect of ESR in the power system to software. We develop static and dynamic implementations of Culpeo and demonstrate on real batteryless devices that considering ESR restores correctness guarantees broken by energy-only charge management. We then demonstrate how to integrate Culpeo's safe voltage into state-of-the-art schedulers, restoring task deadline guarantees for applications with predictable energy harvesting. Finally, we propose an on-chip Culpeo hardware implementation that allows for runtime monitoring of the effects of ESR to respond to changes in harvestable power.

***Keywords*-Intermittent computing, energy-harvesting power system, equivalent series resistance**

I. Introduction

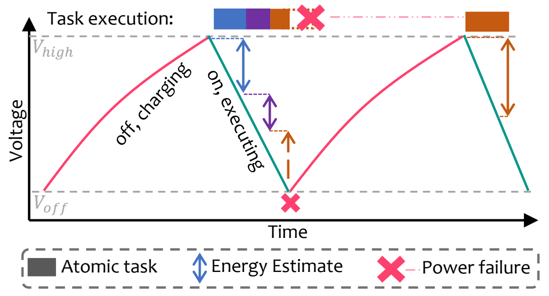
With the advent of intermittent computing [67], [68], [70], [87], [105], [113], energy-harvesting devices have matured as a platform for deeply embedded sensing and data processing applications [32], [66], [82], [108]. Such intermittent systems eliminate batteries, instead collecting their operational energy from environmental sources such as solar power, radio waves, vibrations, and thermal gradients. A device stores energy into a capacitor, drawing energy to compute, sense, and communicate. When the power system can no longer furnish energy, the device turns off to recharge, repeating the cycle. To execute long programs through power failure, intermittent systems must save execution state — such as by hardware triggered checkpoints [12], [13], [50], [76], [85], software defined atomic regions [67], [68], [87], [121], or hybrid model [60], [70], [105].

A key challenge in programming intermittent systems arises because some program tasks—such as using peripherals or sending a radio packet—must complete without being interrupted by power failure [14], [18], [70], [105], but power failure is frequent. Many prior systems [44], [67], [68], [70], [87], [105], [121] provide constructs for such "atomic" re-execution, which is illustrated in Figure 1(a). The plot shows capacitor voltage over time. As tasks execute across the top (represented by the colored blocks), they draw current and consume energy, decreasing the voltage. If voltage reaches the minimum threshold, V^{off}, the device powers off. After recharging, the orange task will fully re-execute. These existing systems opportunistically execute tasks if the capacitor's voltage level is above V^{off}. However, trying to execute a task with insufficient stored energy dooms the device to fail and not only imposes the cost of powering off, recharging, restarting, and re-execution, but risks prolonged non-termination [29], [70].

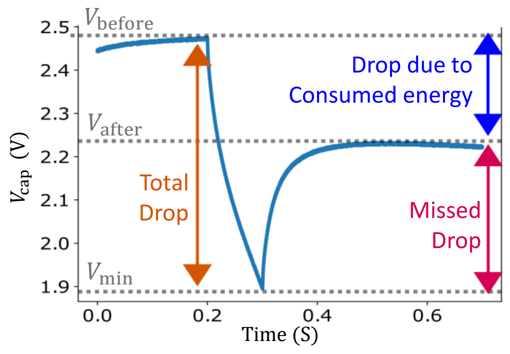
Thus, systems have started to manage charge to avoid unexpected power failures using compilers [29], [69], hardware-aware runtimes [16], [118], or schedulers [47], [71], [77], [88], [121]. Such existing charge management systems reason about energy to size tasks appropriately and execute them only when sufficient energy is available. To estimate task energy, these systems use direct energy measurements, energy modeling, or programmer intervention [16], [92], [118]. Some systems use changes in capacitor voltage as a reasonable approximation of energy ($E_{cap} = \frac{1}{2}CV^2$). Whatever the estimation method, these systems implicitly depend on energy being the sole quantity of interest for safe task execution. In this paper, we show that reasoning about stored *energy* is insufficient. Intermittent software systems must also independently reason about the *voltage* of the energy storage buffer.

The voltage of a device's energy storage buffer changes independently of energy consumption, because a capacitor's voltage varies with current draw (or applied *load*). Even with oracular knowledge of task energy and stored energy, software may experience unexpected failures because of this load-dependent capacitor behavior. The key oversight is that a capacitor has an *equivalent series resistance* (ESR). In a load circuit, a capacitor behaves both capacitively and resistively, with ESR as its resistance. Because of the resistance, the

[‡]This work was completed while the author was affiliated with Carnegie Mellon University.



(a) **Intermittent Execution**



(b) **ESR Drop**

Fig. 1: 1(a) shows an intermittent execution trace. Consuming energy lowers the device's voltage level; if it drops below V^{off}, the device powers off. Software atomic tasks must re-execute from the beginning after such a power failure. 1(b) shows voltage drop and rebound due to ESR on a real trace, which causes the problem addressed by this work. Considering only energy consumption misses this key voltage drop, leading to incorrect charge management.

capacitor experiences a drop in its voltage which "rebounds" to the original level once the load is removed, minus the energy used by the load.

While ESR is a well-known electrical engineering concept, no intermittent system has considered how these voltage changes impact software execution. Prior systems had low loads and low-ESR capacitors, resulting in negligible ESR-drops. However, batteryless systems must be geometrically small, so they are increasingly adopting low-profile but energy-dense *supercapacitors* that have (relatively) high ESR [30], [32], [38], [82], [88], [118]. Furthermore, as applications become more sophisticated, they mix computing with use of sensors [8] and radios [45], [95] that have (relatively) high load. If either the load or ESR is high, the voltage drop due to ESR is substantial and cannot be safely discounted. Figure 1(b) illustrates this drop using a real trace of voltage over time. The energy consumed accounts for only the end-to-end drop, about 0.25 volts. The ESR-induced voltage drop (or *ESR drop*) spans a further 0.35 volts, a drop that is completely missed if a system only considers energy consumption. If the voltage drop due to ESR causes the capacitor voltage to go below the system's minimum operating voltage, the system powers off *even when ample stored energy remains.*

ESR drop breaks the central assumption of charge management systems that reason solely about energy, which is that if the device has enough energy to run a task, the task will not fail. Instead, due to ESR-induced voltage drop, a task safely executes only if there is sufficient energy *and* a high enough voltage to survive the ESR-induced drop. Unexpected task failures that stem from ignoring ESR compromise correctness and degrade performance.

For future energy harvesting systems to be correct, performant, and reliable, their designers must reason about the effects of voltage on software execution. However, considering low-level physical circuit properties such as capacitor ESR is burdensome for software developers. The goal of this work is to enable integrating energy- and voltage-based reasoning into the charge management systems of energy harvesting devices. Accomplishing this goal, we present Culpeo, which is a hardware/software mechanism and architectural interface that provides the minimum safe voltage at which a task can execute without dropping below the operating minimum. We show the value of Culpeo in two different designs of its mechanisms and abstractions. The first is a compile-time, profile-guided analysis to reason about the safe starting voltage for regions of a program (e.g., software tasks [28], [68], [71]). The second implementation uses runtime software and microarchitectural support to dynamically produce accurate, safe starting voltage estimates for tasks. Culpeo's profile-guided design variant avoids the need to profile the load on the target device's power system, separating the concerns of the power system designer from the software developer. Instead, the power system's ESR characteristics are profiled independently of the load, and the load is characterized (e.g. on continuous power) independently of the power system. The runtime software and microarchitectural design variant exposes a simple hardware/software interface that decouples low-level power system dynamics from high-level software schedulers and applications.

We build prototypes of both Culpeo designs and apply them to real energy harvesting systems and application workloads. We show quantitatively that Culpeo produces safe starting voltages and demonstrate that disregarding voltage leads to wildly inaccurate estimates from state-of-the-art schedulers. We then show that integrating Culpeo's runtime design variant into a charge management system restores the charge management system's correctness guarantee—which ignoring ESR previously violated—for three complex, event-driven applications. In summary, this work's key contributions are:

- The Culpeo hardware/software interface definition, which captures both task voltage and energy requirements
- A profile-guided program analysis that combines power system models with load current measurements to compute safe starting voltages for application tasks at compile time.
- Two runtime analyses and implementations in software and with microarchitectural support that calculate safe starting voltages.
- A charge management system from prior work that is broken by ESR corrected through its integration with Culpeo.

- An evaluation on real applications demonstrating that scheduling tasks based on energy is flawed, while scheduling based on Culpeo's computed safe voltage restores application correctness and performance.

II. Background & Motivation

Culpeo is motivated by the need for a hardware/software interface that exposes the power system characteristics of an energy-harvesting device (EHD) to software. Underlying this motivation is the increasing prevalence of high-energy-density supercapacitors, the electrical properties of which present new challenges to designers of EHD hardware and software. Culpeo exists at the intersection of energy-harvesting power systems, profile-guided static and dynamic analysis, and task-based intermittent execution. This section provides a gentle introduction to EHD power systems and the reason for and consequences of the shift to high-ESR supercapacitors. This shift breaks existing charge management systems that reason about energy only, creating a need for Culpeo's charge management interface, which considers the consequences of ESR as well as energy.

A. Energy-Harvesting Power Systems

An EHD has a power system that harvests energy from its surroundings and accumulates the energy in an energy buffer. Later, the EHD consumes the stored energy to compute, sense, and communicate. The hardware components of an EHD can be split in two parts: *supply-side* power system components (regulators, capacitors) that collect and store energy and *load-side* components (microcontrollers, sensors) that execute software. Figure 2 is a simplified schematic of the energy-harvesting power system that we test in this work. It is typical of EHDs that support a large capacitor bank (e.g. millifarads of capacitance) [30], [82], [118] and uses two off-chip voltage regulators: an input booster and an output booster. The input booster regulates fluctuating voltage from the energy harvester to steadily charge the capacitor, up to a maximum voltage level (V^{high}). Using such an input booster decouples the charging behavior from the limitations of the energy harvester, allowing, for instance, V^{high} to exceed the harvester's maximum output voltage. Once the capacitor's voltage level, V^{cap}, reaches V^{high}, the output booster is enabled by the voltage monitor, allowing software to execute. The output booster provides a stable voltage to the *load-side* components as it discharges the capacitor, decreasing the capacitor's voltage level. This output booster is only enabled when V^{cap} is above a device-specific minimum value (V^{off}). In other words, software executes only when V^{cap} is between V^{high} and V^{off} (e.g. between 2.4V and 1.6V). When software deactivates (i.e. when V^{cap} falls below V^{off}), the system uses hardware to fully recharge to V^{high} before the output booster is re-enabled [30], [31], [43], [66], [82], [90].

EHD power systems increasingly use supercapacitors to buffer energy instead of batteries or other (e.g., ceramic) capacitors. The shift to supercapacitors has benefits and drawbacks. Compared to batteries, supercapacitors provide an attractive

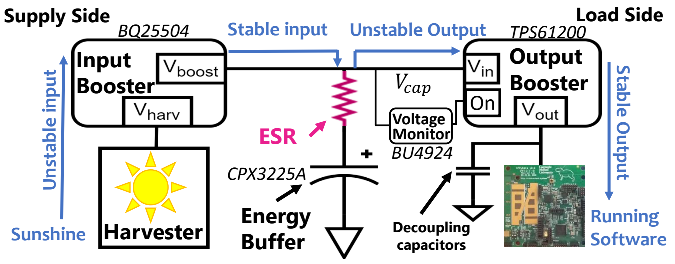


Fig. 2: An Annotated energy-harvesting power system schematic. An EHD harvests energy into a (super)capacitor and uses the energy to run software tasks. Input and output boosters regulate voltage to the energy buffer and the load, respectively.

balance between energy capacity and lifetime; a supercapacitor can last for decades [57], [83], [126] while a rechargeable battery lasts only a few months under a high computing duty cycle. Compared to other capacitors, supercapacitors are more energy dense, with a greater capacitance in a smaller volume. The main drawback of supercapacitors is that they usually have a higher ESR than other capacitors, especially in volumetrically small packages. Volume constraints matter, because EHD deployments often optimize for small volume, such as an implantable medical sensing application requiring a small form factor. The shift to supercapacitors enables volumetrically small devices with high energy density (compared to capacitors) and long lifetimes (compared to batteries), but requires a system to tolerate high ESR.

B. Capacitor Trends: Size, Leakage, and ESR

Culpeo specifically addresses volume constrained EHDs, where the entire sensing and computing platform might be the size of a business card [30], [90], or even much smaller [63]. Given the fixed, tight volume budget of such a device, a power system designer should build an energy buffer that satisfies an application's energy requirements while minimizing ESR, but ESR and volume are not the only considerations. To reduce losses when harvestable power is weak, EHD designers also need to minimize intrinsic leakage current (or "direct current leakage", usually "DCL") from the energy buffer. Further, building a bank out of fewer, higher capacity capacitors reduces the total number of parts installed on the EHD and its associated cost. In short, the ideal energy buffer for an EHD would be composed of tiny, high-capacity, low-leakage, low-ESR capacitors; however, such capacitors do not exist.

Figure 3 shows the trends of ESR versus volume for 45 mF capacitor banks formed from different capacitor technologies. Each point represents a 45 mF bank composed of a specific capacitor (e.g. identified by part number) available from Digikey [33]. Banks are formed by combining multiple of each capacitor until the total capacity is 45 mF (e.g. a stack of 45 1 mF capacitors). To acquire capacitor data, we limited our search to capacitors between 1 μF and 45 mF and then downloaded the summary metadata for the 500 shortest parts in each capacitor type category. The ESR of ceramic capacitors is not included in the metadata since it is typically very low [107], so we approximate each individual ceramic capacitor part to have an ESR of 10 mΩ. The banks range from supercapacitors that are roughly the size of a grain of Kyrgyz rice [81] to electrolytic capacitors optimized for low ESR that are larger than a standard US pint glass.

Figure 3 illustrates how different types of capacitors align with the needs of a volume constrained EHD's power system. The data show that typical electrolytic capacitors do not meet the needs of energy harvesting platforms, consuming too much volume for too little energy, with moderately high ESR. The lowest volume tantalum and ceramic banks would meet the needs of an EHD's power system, if *only* size mattered. However, the smallest tantalum banks have extremely high leakage current (e.g., 26 mA), and the ceramic banks require an impractical number of parts (e.g., $> 2,000$) to furnish 45 mF of capacitance. In contrast, supercapacitors meet the capacitance requirement with the smallest volume of all options, with low leakage current ($20nA$), and a practical part count (six) compared to other technologies. The figure also clearly illustrates the relatively high ESR cost that comes with the low volume, leakage, and part count of supercapacitors. However, unlike the unavoidable costs of high leakage and part count imposed by non-supercapacitors, the high ESR cost of supercapacitors *is* directly addressed by Culpeo's new ESR-aware mechanism for charge management.

C. *ESR Induced Voltage Drops*

ESR makes a capacitor behave like a resistor (reducing current flow) as well as a capacitor (storing energy). When current flows from a capacitor, ESR induces a voltage drop, as in a resistor. This voltage drop does *not* actually consume (much) energy as the voltage rebounds to its original level as load decreases. If the drop causes the capacitor's voltage to sink beneath V^{off}, however, the system will power down regardless of the remaining stored energy. We illustrate this problem in Figure 4. With a 10Ω ESR capacitor and a 50mA current draw similar to a LoRa radio [94], the voltage drop is 500mV. With a capacitor voltage range of 2.4V to 1.6V, this 500mV ESR drop is 62.5% of the device's operating range. This radio transmission may consume 50mA for a short duration,

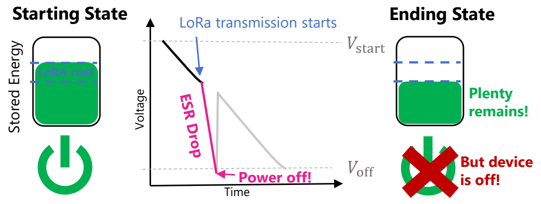


Fig. 4: Voltage drop due to ESR can cause the device to power down even when there is plenty of stored energy

requiring far less energy than is stored in the capacitor (e.g., 5% of the stored energy). However, if the operation begins with a voltage lower than 64.5% of its operating voltage range (i.e., 2.12V), the ESR drop causes the system to shut down.

This ESR drop across the energy buffer in an EHD power system is thus distinct from noise on the voltage supply of a microprocessor [19] because of its duration. ESR drop lasts up to hundreds of milliseconds, while a load is applied, in contrast to the microsecond transients caused by voltage noise.

D. *Disregarding Voltage Breaks Past Systems*

Prior energy-harvesting systems only modeled incoming (recharging) and outgoing (computation) energy, without considering circuit-level characteristics like ESR. Considering only energy and disregarding ESR drop causes charge management systems like schedulers to fail frequently.

ESR breaks schedulers. ESR-induced voltage drops violate the core assumption of schedulers for intermittent systems, which is that a task will execute completely [24], [25], [40], [64], [71], [77] if the energy buffer contains more energy than the task consumes. As a concrete example, we consider the scheduler CatNap [71], which adapts RTOS's feasibility scheduling [123] for intermittently powered systems. CatNap looks at the energy consumed by high priority tasks and their deadlines, determining if it is possible to schedule tasks and recharges so that there is always energy to run the tasks at the appropriate time.

Figure 5 shows how CatNap's careful scheduling still results in a task failure. CatNap must determine if it is possible to schedule two tasks: `radio` repeats every 6.5 ticks and `sense`

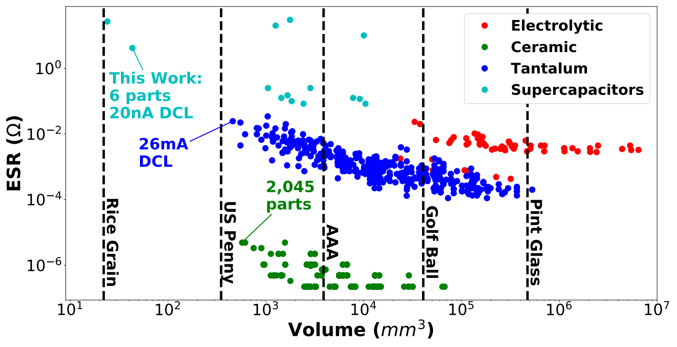
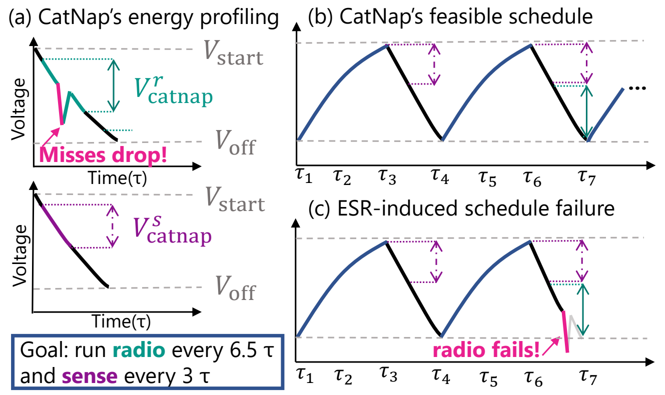


Fig. 3: Volume vs. ESR for 45 mF capacitor banks using different capacitor technologies. Supercapacitors enable the smallest design point, but must grapple with high ESR.



(a) CatNap's energy profiling
Voltage
V_{start}
V^r_{catnap}
V_{off}
Misses drop!
Time(τ)
V_{start}
V^s_{catnap}
V_{off}
Time(τ)
Goal: run radio every 6.5 τ and sense every 3 τ
(b) CatNap's feasible schedule
τ_1 τ_2 τ_3 τ_4 τ_5 τ_6 τ_7
(c) ESR-induced schedule failure
radio fails!
τ_1 τ_2 τ_3 τ_4 τ_5 τ_6 τ_7

Fig. 5: CatNap's feasibility test will lead to failed executions

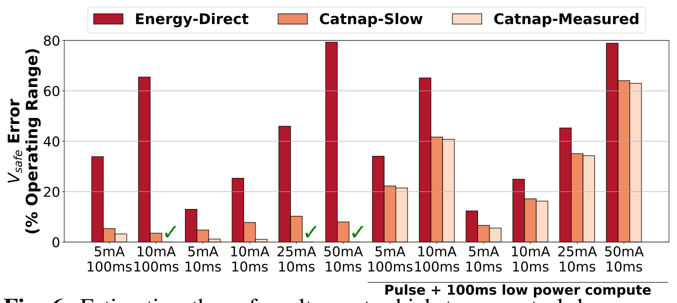


Fig. 6: Estimating the safe voltage at which to run a task by energy costs alone results in wildly incorrect predictions.

repeats every 3 ticks. CatNap estimates the energy costs of the tasks by measuring voltage at the start and end of each task's execution, in Figure 5 (a). The graphs are of voltage over time, with the energy estimate for `radio` indicated by a green solid arrow and `sense` with a purple dashed arrow. Figure 5 (b) shows Catnap's feasible schedule of the tasks interspersed with recharges. Based on energy estimates alone, `sense` followed by `radio` should complete in one discharge at τ_6 to τ_7. Figure 5 (c) shows how this schedule will fail due to ESR. While there is sufficient *energy* for `radio`, the scheduler executes it at a *voltage* too low to survive the ESR drop, causing a failure. To be correct, CatNap and other schedulers must ensure that the starting voltage level is high enough to satisfy ESR drops as well as the consumed energy.

Failures are Common. Running a task at a voltage too low to sustain ESR drops is not an edge case. Figure 6 shows the error between voltages at which it is actually safe to start running a task and those voltages predicted by energy-based estimates, for a series of load profiles run on the Capybara power system. If the error is positive, the task fails to complete. We provide details on the task profiles in Section VII; at a high level, the profiles comprise different combinations of pulse width, intensity, and load shape. Direct energy estimates fail across the board, and voltage-based energy approximations like Catnap are highly dependent on how quickly they measure capacitor voltage after the task completes. A quick measurement can capture the voltage level before rebound, resulting in a highly conservative energy estimate that accounts for the voltage drop as a side-effect. Catnap-Measured reports the voltage estimates by the published Catnap implementation [71], and Catnap-Slow reports estimates if there is a 2 ms delay between a task's completion and the measurement. Whether a task's energy cost is obtained through direct measurement or through using voltage as a proxy, determining the safe starting voltage by energy cost alone results in task failure most of the time.

Simple HW and SW approaches do not fix ESR drops. Power-system designers commonly account for ESR drops due to quick, transient spikes in load-side current by adding small decoupling capacitors (around 10-100μF) close to the load-side components [7], [42], [73], [114], [119]. While adding a large amount of decoupling capacitance is the "go-to" circuit fix for load-dependent voltage drop, decoupling capacitance does not address the problem that Culpeo solves. Transient spikes draw their current from the decoupling capacitors instead of the high-ESR supercapacitors. However, our work targets *sustained* high current loads. Decoupling capacitors are typically too small to supply these sustained loads, which draw mainly from the supercapacitor. We quantitatively evaluated this effect by testing a wide range of decoupling capacitance (400uF to 6.4mF) with the Capybara [30] power system, running a 50mA-100ms load (similar to a LoRa packet) from a 33mF supercapacitor. Even with an abnormally high 6.4mF of decoupling capacitance, we still observed an ESR drop of 200mV, which is 20% of the device's operating range.

Simply adding a safety margin (e.g., provisioning extra energy) is also an inadequate solution. Provisioning unnecessary energy will make the entire system inefficient while still not guaranteeing correctness; a larger ESR drop that spills over the safety margin could still happen. Further, the programmer has little guidance on how much extra energy to provision for safety. The limited data in capacitor datasheets make handling ESR a guessing game for application developers, even if they are aware of the voltage drop effect [9], [35], [55], [78], [93]. While industry hardware designers perform expensive characterizations of ESR across frequency, temperature, humidity and lifetime, this information is not accessible to software designers [114]. A more practical approach, and the one adopted by Culpeo, is to provide software developers with an interface to reason about load-dependent ESR drops.

III. Culpeo Overview

Culpeo is a hardware-software interface and collection of system and microarchitectural mechanisms that captures the ESR-aware voltage and energy requirements of a software task. Figure 7 shows a high-level overview of Culpeo, illustrating how the system and the programmer interact with Culpeo. The goal of Culpeo is to determine the lowest possible voltage at which it is safe to start executing a task without failing, which we call V^{safe}. Culpeo determines V^{safe} based on the voltage drop due to ESR for a task, V^{δ}, and the voltage drop due to the energy consumed by the task. Culpeo's output is a set of per-task V^{safe} values that can be used by both software systems and programmers to reason about task completion in the presence of ESR induced voltage drops.

The V^{safe} estimates that Culpeo produces can be integrated directly into intermittent runtimes, or they can be used by the programmer to reason about task completion at compile time. An intermittent runtime or task scheduler records per-task V^{safe} values, using those values to determine when to execute a task. Such a software system may reason directly about a single task by comparing its V^{safe} to the energy buffer's voltage just before the task runs, or it may use V^{safe} as part of a feasibility test for a sequence of task executions scheduled into the future. A programmer can use Culpeo as a complement to an energy model [29] to reason about how to subdivide a program's code into reasonable tasks. For instance, if a task's V^{safe} value is higher than what the energy buffer can provide, the programmer knows they must correct the task division. If using a device with

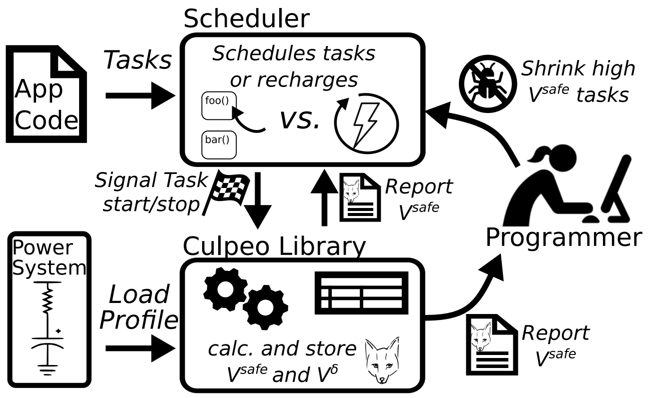


Fig. 7: Schedulers use the Culpeo library to safely schedule recharging and application tasks. A scheduler runs tasks, indicating their start and stop to Culpeo, then Culpeo reports V^{safe} to the scheduler and the programmer using task profiles gathered by the power system.

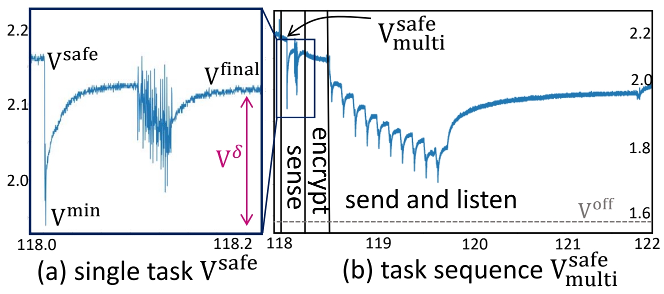


Fig. 8: A measured V^{cap} trace over a single (a) and multiple (b) tasks: V^{safe} guarantees that a task will complete, but the V^{δ} parameter is required to calculate V^{safe}_{multi}, a safe voltage for a sequence of tasks.

a configurable energy storage array [30], [118], the programmer can also use V^{safe} as a guide to configure the energy buffer. Furthermore, V^{safe} values are useful during development for understanding how different device load conditions affect task completion, e.g., testing if operating a radio at the end of a compute task results in a higher V^{safe} than operating it at the beginning.

Culpeo brings together several models and mechanisms to provide useful estimates of V^{safe}. At the core of the technique is the Culpeo Voltage-Aware Charge Model (Section IV), which uses information about a program's current load and a target device's power system to analytically derive V^{safe}. To collect the necessary program and power system information, Culpeo must observe a device's power system while the program runs. Culpeo includes several alternatives for making these observations: program analysis (Section V-A), runtime support relying on software interrupts (Section V-C), and microarchitectural support (Section V-D).

The three alternatives for power system monitoring make the Culpeo Charge Model useful at both design time and run time, but they require separate mathematical implementations. Culpeo defines two underlying methods for implementing the Culpeo Charge Model. The first ingests load current profiles to produce V^{safe} at compile-time, which we call Culpeo-Profile-Guided (Culpeo-PG) (Section IV-C). The second, Culpeo-Runtime (Culpeo-R), takes in online V^{cap} measurements and calculates V^{safe} onboard the MCU (Section IV-D).

IV. The Culpeo Voltage-Aware Charge Model

The core of Culpeo is its voltage-aware charge model, which it uses to capture the voltage and energy requirements of a software task. The model defines a safe starting voltage for each software task, V^{safe}, that accounts for ESR drop. To calculate V^{safe}, both Culpeo charge model implementations (Culpeo-PG and Culpeo-R) require a model of the target device's power system and insight into the load profile of each software task.

A. *Defining V^{safe}*

V^{safe} is the minimum energy buffer voltage level at which a task will complete without experiencing a power failure, accounting for voltage drops due to both consumed energy and ESR. Figure 8 shows a real voltage trace of a task execution, annotated with the various voltage values that Culpeo uses in its V^{safe} calculation. V^{δ} is the difference between the minimum voltage during a task (V^{min}) and the final voltage once the task completes and the voltage rebounds (V^{final}). Culpeo defines both V^{safe} for a single task and V^{safe}_{multi} for a series of tasks, allowing a scheduler to determine the feasibility of a task sequence, not just a single task. As with V^{safe}, starting execution at V^{safe}_{multi} guarantees that all tasks in the sequence will complete. While calculating V^{safe} for a single task depends only on the voltage levels during the task's own execution, calculating V^{safe}_{multi} requires composing per-task V^{δ} information across the sequence.

Formulating V^{safe}_{multi} requires determining a voltage level for each task that will satisfy its voltage requirements *and* meet the voltage requirements of subsequent tasks. For the initial task of a sequence, Task 0:

$$V_0^{safe} = V(E_0) + penalty_0 + V_1^{safe}$$

Here, $V(E_0)$ is the voltage required to satisfy the energy consumed by Task 0. Regardless of ESR, the voltage must be at least $V(E_0)$ before Task 0 starts, or the energy buffer's voltage will drop below the system's power-off threshold, V^{off}, before the task completes. The voltage after this drop due to consumed energy must be high enough to satisfy the requirement of the subsequent task, V_1^{safe}. If ESR is not a factor, V^{safe} for Task 0 followed by Task 1 would be $V(E_0) + V_1^{safe}$. Accommodating a non-zero ESR drop requires increasing V_0^{safe}, which Culpeo accomplishes by adding a *penalty* term, $penalty_0$. The penalty is a corrective term that ensures the temporary voltage drop due to ESR, V_0^{δ}, will not force the voltage below V^{off}. However, not all tasks in a sequence require a penalty term. If V_1^{safe} is high enough to tolerate Task 0's V^{δ} without crossing the power-off threshold, the ESR-drop will rebound after Task 0 completes. This rebound "repays" the penalty, imposing no corrective requirement on V_0^{safe}. On the other hand, if V_1^{safe} is not high enough, a sufficient penalty must be added to keep the voltage above V^{off} during Task 0. The following expression describes the penalty computation:

$$penalty_0 = \begin{cases} V^{\mathsf{off}} + V_0^{\delta} - V_1^{\mathsf{safe}}, & \text{if } V^{\mathsf{off}} + V_0^{\delta} > V_1^{\mathsf{safe}}. \\ 0, & \text{otherwise.} \end{cases}$$

Computing $V_{multi}^{\mathsf{safe}}$ thus requires combining the V^{safe} terms and their penalty values for tasks in the sequence. At the end of the task sequence, the voltage must be high enough that meeting the last task's voltage requirements results in a voltage at or above the minimum operating threshold.

$$V_{\mathsf{final}}^{\mathsf{safe}} = V(E_{\mathsf{final}}) + penalty_{\mathsf{final}} + V^{\mathsf{off}}$$

$V_{multi}^{\mathsf{safe}}$ can thus be formulated as the summation of the voltage needed to satisfy the energy and ESR drops for each task in a sequence:

$$V_{multi}^{\mathsf{safe}} = \sum_{i=0}^{n} V(E_i) + \sum_{i=0}^{n} penalty_i + V^{\mathsf{off}}$$

If the voltage at the start of a sequence of tasks ε is $\geq V_{\varepsilon}^{\mathsf{safe}}$, then the voltage will not dip below V^{off} while executing the tasks. As a proof sketch that $V_{multi}^{\mathsf{safe}}$ is correct, assume that for some task i in the sequence, the voltage after running i is less than the threshold, i.e., $V_i^{\mathsf{safe}} - V(E_i) - penalty_i < V^{\mathsf{off}}$. By definition, $V_i^{\mathsf{safe}} = V(E_i) + penalty_i + V_{i+1}^{\mathsf{safe}}$. Simplifying the equations results in $V_{i+1}^{\mathsf{safe}} < V^{\mathsf{off}}$. As no part of V_{i+1}^{safe} is negative, and the base case is at least V^{off}, this equation results in a contradiction.

B. Modeling the Power System

To calculate V^{safe}, all Culpeo implementations model the device's energy buffer and output booster (Figure 2). For the input booster, Culpeo-PG assumes a worst case of no incoming power, and Culpeo-R assumes the input is stable.

The energy buffer (i.e., capacitor) is modeled differently in each Culpeo variant due to differences in the V^{safe} calculation algorithms. Culpeo-PG models the energy buffer based on its capacitance, C, and its ESR as an ideal capacitor in series with a resistor. The value of C comes from the capacitor's datasheet and is generally conservative [9], [55], [93]. Using datasheet ESR values is too inaccurate; the ESR experienced by a load changes with the load's *frequency* (i.e., how often the load current is applied per unit time), but many datasheets do not supply the full spectrum [9], [55], [93]. Furthermore, small decoupling capacitors throughout the power system also affect ESR. We instead derive a curve of ESR versus frequency via direct measurement of the power system. To choose a representative ESR value from the curve, Culpeo-PG uses the width of the largest current pulse, excluding high frequency noise. In contrast, Culpeo-R models the capacitor with no knowledge of the exact capacitance or resistance. Culpeo-R ignores some nonidealities of supercapacitors, e.g., leakage [53] and charge speed effects [6], relying on the ideal $I = C\frac{dV}{dt}$ equation for capacitor analysis.

To model the output booster, the power system designer sets V^{high}, the capacitor's highest voltage, and V^{off}, the voltage at which the output booster turns off. The designer also sets V^{out}, the output voltage of the output booster, which is used in conjunction with the current profile to determine P^{out}, the power delivered to the load. Culpeo uses datasheet booster efficiency curves to relate P^{in}—the power drawn from the energy buffer to the output booster—to P^{out} as the energy buffer voltage V^{cap} declines over the course of an operation. We assume the output booster has little change in efficiency w.r.t. current [2], [3], so efficiency can be modeled as a line relating input voltage to efficiency (i.e. $\eta = mV + b$) at a single current value. Combining this output booster model with the energy buffer model, Culpeo can predict the behavior of the power system in response to an arbitrary task load.

C. Culpeo-PG V^{safe} Calculation

In addition to the power system model, Culpeo-PG requires the application developer to input a current profile (captured using any power system) of each program task, a process described in detail in Section V-A. Culpeo-PG finds V^{safe} by iteratively calculating the voltage drop due to *energy consumption* and due to *ESR drop*. Essentially, Culpeo-PG applies each step of the current trace to the power system model and predicts the combined V^{safe}.

Algorithm 1 describes how the energy and voltage penalty are calculated. The algorithm starts using the power system model provided by the power system designer (P) and the current trace collected by the application developer (I). At each time step, dt, Culpeo calculates E using the output booster efficiency, η, given that $P^{\mathsf{in}} = P^{\mathsf{out}}/\eta$. Next, Culpeo estimates V^{cap} to calculate the current drawn from the capacitor, because $P^{\mathsf{in}} = I_{in} \times V^{\mathsf{cap}}$. Culpeo must consider V^{cap} when assessing the current from the capacitor to the output booster, because as V^{cap} decreases, the booster draws more current from the capacitor; as current increases, so too does ESR drop. Finally, Culpeo calculates the voltage penalty, which guarantees that the new V^{safe} satisfies the energy requirements of the next step, $V[i+1]$ and can sustain the ESR drop in the present step.

Algorithm 1 Culpeo V^{safe} algorithm

```
1:  function CULPEOVSAFE(CurrentTrace $I$, PowSys $P$)
2:      $V \leftarrow \varnothing$                                  ▷ Initialize safe starting voltages
3:      $C \leftarrow$ GETCAP($P$)                                  ▷ Get capacitance value
4:      $R \leftarrow$ GETESR($P$,$I$)                              ▷ Get freq. dependent ESR
5:      for $i \leftarrow len(I)..0$ do                             ▷ Reverse through trace
6:          $E \leftarrow I[i] * V^{out} * dt/\eta$                 ▷ Energy consumed by step i
7:          $V^{cap} \leftarrow$ ESTVCAP($P$,$I[i]$,$V[i+1]$)       ▷ Estimate $V^{cap}$
8:          $I_{in} \leftarrow I[i] * V^{out}/\eta(V^{off}) * V^{cap}$   ▷ Current out of cap.
9:          $V^{\delta} \leftarrow I_{in} * R$                      ▷ Voltage drop from ESR
10:         $V_{penalty} \leftarrow \max V^{off} + V^{\delta}, V[i+1]$   ▷ Voltage penalty
11:         $V[i] \leftarrow \sqrt{2 * E/C + V_{penalty}^2}$
12:     end for
13:     return $V[0]$
14: end function
```

As shown in Section VII, Culpeo-PG produces accurate V^{safe} calculations for a recently profiled capacitor. However, Culpeo-PG assumes a static ESR model, but supercapacitor ESR and nominal capacitance change over the device lifetime (years). Capacitance can reduce to less than 80% of nominal and ESR can increase to double its nominal, beyond which the capacitor is considered dead [35], [78], [97]. A runtime V^{safe} calculation captures these aging effects by rerunning periodically.

D. Culpeo-R Calculation

To calculate V^{safe} at runtime, Culpeo-R profiles the capacitor voltage online, but storing a full time series of voltage data is memory intensive for the highly constrained devices Culpeo targets. Instead, Culpeo-R only captures the starting (V^{start}), minimum, and final voltages during an event execution. Sections V-C and V-D describes Culpeo-R's profiling in detail, but, at a high level, it is achieved by repeatedly sampling from an ADC connected to V^{cap} and recording the minimum observed voltage.

The goal of the Culpeo-R V^{safe} calculation is to allow the system to profile tasks starting at an arbitrary capacitor voltage and produce a useful V^{safe} estimate. Changes in efficiency as the input voltage declines make the mapping non-trivial to compute, particularly on constrained devices. Culpeo-R makes several assumptions to keep the code running on the MCU practical. The first is that efficiency decreases monotonically with voltage; since Culpeo approximates efficiency as a line, this assumption holds as long as the slope of the line is positive. The second is that harvested power is roughly constant during the event execution. This assumption is reasonable as the supercapacitor-enabled devices Culpeo targets generally rely on more powerful, slowly changing energy sources (e.g. solar power) than low-end batteryless motes. Culpeo-R produces different V^{safe} values for different levels of incoming power, so it is best to use Culpeo-R in conjunction with scheduler policies that re-profile as harvestable power changes [71].

Culpeo-R separates the worst case ESR drop, V^{δ}, from the energy induced voltage drop, V_E^{safe}, and calculates them independently before adding the effects back together. First, we calculate the new V^{δ} for a given event in terms of the current, i_{load}, the ESR R, and the efficiency at the event's V^{min}, $\eta(V^{\text{min}})$, as shown in Equation 1a. This expression for V^{δ} is rooted in Ohm's law and converter efficiency, namely $V^{\text{out}} I_{out} = V^{\text{cap}} I_{in} \eta(V^{\text{cap}})$. Intuitively, as efficiency decreases with V^{min}, V^{δ} gets larger. V^{δ}_{safe}, the worst case V^{δ}, is calculated by substituting V^{off} for V^{min} in Equation 1a to form Equation 1b. The problem, however, is Equation 1b requires an accurate measurement of the load current and ESR when V^{cap} reaches V^{off}. Instead, Equation 1c defines V^{δ}_{safe} in terms of the observed V^{δ} without directly measuring the current trace.

$$V^{\delta} = \frac{i_{load} * R * V^{\text{out}}}{V^{\text{min}} * \eta(V^{\text{min}})} \tag{1a}$$

$$V^{\delta}_{safe} = \frac{i_{load} * R * V^{\text{out}}}{V^{\text{off}} * \eta(V^{\text{off}})} \tag{1b}$$

$$V^{\delta}_{safe} = V^{\delta}\left(\frac{V^{\text{min}} \eta(V^{\text{min}})}{V^{\text{off}} \eta(V^{\text{off}})}\right) \tag{1c}$$

In addition to the voltage drop caused by ESR, Culpeo-R must consider the voltage drop caused by actual energy expenditure. Instead of predicting a drop due to energy, Culpeo-R solves for V_E^{safe} based on the assumption that the energy delivered to the load, E_{out}, is constant across all input voltages. Equation 2a defines E_{out} by integrating output power ($P_{out} = V^{\text{cap}} I_{in} \eta(V^{\text{cap}})$) over time. We then apply the relationship between current and capacitance ($I = C\frac{dV}{dt}$) and change variables from time to voltage to redefine E_{out} in Equation 2b. Finally, we set as equal the integrals that represent the measured execution (V^{start} to V^{final}) and what would be the execution starting at V^{safe} (V^{safe} to V^{off})(Eq. 2c). The goal now is to solve for V^{safe} by resolving both definite integrals, since V^{off} is given and V^{start} and V^{final} are quantities the Culpeo interface can measure.

$$E_{out} = \int_{t_{start}}^{t_{end}} V(t) i_{in}(t) \eta(V(t)) dt \tag{2a}$$

$$E_{out} = C \int_{V^{start}}^{V^{final}} \eta(V) V dV \tag{2b}$$

$$C \int_{V^{\text{off}}}^{V^{\text{safe}}} \eta(V) V dV = C \int_{V^{\text{final}}}^{V^{\text{start}}} \eta(V) V dV \tag{2c}$$

However, even with a linear efficiency function ($\eta(V)$), solving Equation 2c requires multiple cubic root operations that are expensive for the low power microcontrollers that Culpeo targets. Instead, we approximate the solution as:

$$(V_E^{\text{safe}})^2 = \frac{\eta(V^{\text{start}})}{\eta(V^{\text{off}})}((V^{\text{start}})^2 - (V^{\text{final}}))^2 + (V^{\text{off}})^2 \tag{3}$$

Effectively, we solve Equation 2c after collapsing $\eta(V)$ into a constant. We use $\eta(V^{\text{start}})$ on the left and $\eta(V^{\text{off}})$ on the right because they can be known quantities that can be calculated at compile time. V^{off} is set by the power system designer, and Culpeo-R may choose a known V^{start} to run the event. We finally define V^{safe} as $V^{\text{safe}} = V_E^{\text{safe}} + V^{\delta}_{safe}$

V. Culpeo System Design

A Culpeo system implementation operationalizes the Culpeo voltage-aware charge model to produce V^{safe} values by analyzing power system characterization data and task energy characterization observations. Culpeo-PG is a profile-guided analysis framework for producing V^{safe} values, including support for profiling tasks before deployment. Culpeo-R is a dynamic analysis for estimating V^{safe} at run time using either an interrupt-based software system or a combination of a runtime library and microarchitectural support. Culpeo-PG and Culpeo-R expose the same API, which is listed in Table I. These API calls allow both systems to profile software tasks, perform V^{safe} calculations, and access V^{safe} and V^{δ} data, while accommodating Culpeo's breadth of implementations.

TABLE I: Culpeo calls grouped by function. `id` is a task identifier.

Profile	Calculate	Access
`profile_start()`	`compute_vsafe(id)`	`get_vsafe(id)`
`profile_end(id)`		`get_vdrop(id)`
`rebound_end(id)`		

A. Culpeo-PG Design

Culpeo-PG profiles an application's tasks offline, before deployment. Culpeo-PG implements `profile_start` and `profile_end` operations that interface with current measurement instruments [101] and capture a task's worst-case current trace. Capturing a task's current trace is reasonable because traces will be manageably short; a task's total energy consumption cannot exceed the device's energy buffer capacity. Profiling to cover a wide range of operating points (including the worst-case behavior) is also reasonable because, as prior work showed, "knob" values [70], [71] often determine task energy consumption (e.g., the input dimension of a matrix in a matrix-matrix multiplication computation.) Culpeo-PG collects a task current trace (at 125kHz in our prototype) and selects an ESR value from the power system's ESR curve to use in the V^{safe} calculation. The analysis then calls its offline version of `compute_vsafe` to produce V^{safe} and V^{δ} estimates using the math from Section IV. Once Culpeo has computed a V^{safe} and a V^{δ} value for each task, these values may be used by the developer or code directly. For instance, a programmer may include these values in a program to be read at runtime, allowing a program to use its own logic to compare these values to the voltage of the device's energy buffer, for instance, to control task dispatch. The advantage of Culpeo-PG is that it allows application developers to calculate V^{safe} values prior to deployment using a continuously powered system, but the estimates are limited by the accuracy of the statically profiled inputs.

B. Culpeo-R Design

Culpeo-R is a dynamic analysis that profiles tasks while an application runs in deployment and uses the profiles to compute V^{safe} and V^{δ} estimates. An intermittent runtime system or scheduler can then use Culpeo's API calls to access V^{safe} and V^{δ} data to make task scheduling decisions. Culpeo-R has several operations that control its behavior. `profile_start()` begins profiling a segment of code and `profile_stop(id)` ends profiling, storing profile results in an in-memory table of per-task measurements that Culpeo indexes by task identifier `id`. `compute_vsafe(id)` performs V^{safe} and V^{δ} calculations at runtime on the device's CPU, using the profile data stored in the task's entry in the profile table. If a task's profile table entry is unpopulated, `compute_vsafe(id)` is a `no-op`. Culpeo then stores per-task V^{safe} and V^{δ} values in an in-memory table. Culpeo's `get_` functions retrieve V^{safe} and V^{δ} values from the table if valid values exist, otherwise returning V^{high} and -1. An intermittent runtime system or scheduler can explicitly call these Culpeo API functions and use the retrieved values to make task scheduling decisions.

By operating online, Culpeo-R can adapt to changing environmental and power system conditions. For schedulers that monitor charge rate [71], a change in incoming power that exceeds a threshold can be used to trigger re-profiling and re-collection of V^{safe} and V^{δ}. Culpeo-R also allows capturing V^{safe} values for changing configurations in a reconfigurable energy buffer [30], [118]. Culpeo models a system's energy buffer as a capacitor in series with a variable resistor, capturing the effect of low resistance connections between individual banks and the shared capacitor voltage rail [118]. To handle data for multiple capacitor bank configurations, Culpeo-R tags per-task data with a buffer identifier. Future `get` queries must then specify a buffer configuration.

Culpeo-R has two implementations, one is an interrupt-based software implementation and the other is a combination of software and microarchitecture support.

C. Culpeo-R-ISR Implementation

The interrupt-driven implementation (Culpeo-R-ISR) relies on an interrupt service routine (ISR) triggered by a hardware timer that reads from the ADC and updates the minimum observed voltage as a task progresses. `profile_start()` sets the minimum observed voltage to infinity and enables a 1 ms timer to trigger the profiling ISR. The function then configures an ADC (on- or off-chip) that will be read from quickly in the ISR, and reads from the ADC to record V^{start}. Once the profiled task is complete, calling `profile_end(id)` disables the timer interrupt and ADC and puts the MCU in a low-power sleep mode to allow the capacitor voltage to recover from any ESR drop. The MCU awakens every 50 ms to read from the ADC and update a *maximum* observed voltage. Sleeping between ADC samples minimizes the MCU's power draw to ensure an accurate V^{final}. Finally, the scheduler runs `rebound_end(id)` to exit sleep when the capacitor voltage stops increasing, and V^{final} is set to the maximum value.

We implemented a Culpeo-R-ISR interface on an MSP430 microcontroller and show in Section VII that it substantially improves the performance of event driven applications. However, Culpeo-R-ISR has several drawbacks. The first is that the on-chip ADC in most microcontrollers is relatively high power [116] which limits the profiling frequency and accuracy for tasks with small ESR-drops (e.g. compute tasks). Second, the MCUs we target are in-order, single-threaded cores, so time spent sampling the ADC in software is time taken from the application. Further, not all applications tolerate other interrupts, leading to bugs [1]. Third, monopolizing the only ADC is not an option if a task needs it. Many MCUs can multiplex ADC access [111], but this can increase the sampling delay for an application and force a programmer to rewrite their ADC driver.

D. Culpeo-R Microarchitecture

Culpeo-μArch is a custom microarchitectural mechanism that allows a system to collect the profile data needed to compute V^{safe} and V^{δ} without involving the device's MCU. Figure 9 shows a detailed view of the proposed microarchitectural additions, which integrate into a generic MCU architecture. Culpeo-μArch measures V^{cap} using an 8-bit ADC and uses a digital comparator to automatically capture a minimum (or maximum) voltage value. Sampling V^{cap} at high frequency using hardware captures start, minimum, and final voltage values required by Culpeo to produce V^{safe} and V^{δ} estimates without continuously involving the MCU. The Culpeo

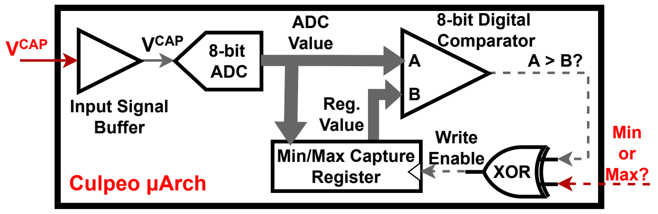


Fig. 9: The Culpeo-μ-Arch is a low overhead design that uses an 8-bit ADC, a comparator and a single register to track capacitor voltage for Culpeo-R. Red arrows indicate inputs, solid arrows are analog signals, dashed arrows are boolean and wide arrows are 8-bit buses.

peripheral block includes a high impedance input buffer to minimize leakage from the capacitor, and an 8-bit register to capture minimum/maximum values. The MCU interacts with the block by writing to a memory mapped control register and provides a clock (100kHz in our prototype) to trigger ADC sampling. The MCU can read out measurements through a memory mapped data register.

Table II shows the low-level driver commands that interface with the block's control signals and are used to implement the Culpeo-R runtime library. `configure` enables or disables the block. `prepare` writes a value to to the "min/max" capture register, 0xFF for minimum, 0x00 for maximum, in preparation for sampling. `sample` starts minimum or maximum sampling, and `read` reads from the capture register. To implement `profile_start`, the core issues `configure(on)`, reads the current ADC value to use as V^{start}, and then issues `prepare(min)` and `sample(min)` to start minimum sampling during the task. In `profile_end`, software uses `read` to extract the minimum before switching to maximum tracking, i.e. issuing `prepare(max)` and `sample(max)`. Unlike Culpeo-R-ISR, the peripheral block will not end the rebound tracking until it receives a `rebound_done(id) call` that reads the maximum voltage and disables the block. Waiting until a rebound done call gives the scheduler more flexibility in capturing V^{final}. The block, as we show shortly, is low power and may be kept enabled indefinitely, so the scheduler may choose to run another task immediately instead of waiting to capture a more accurate (higher) V^{final}.

TABLE II: Culpeo on-chip peripheral command interface Culpeo on-chip is a memory-mapped peripheral with control and data registers.

Function	Description
`configure([on/off])`	Enable or disable ADC
`sample([min/max])`	Start repeated ADC sampling, storing the min or max value
`prepare([min/max])`	Set the capture reg. to 0xFF (for min) or 0x00 (for max)
`read()`	Read from the capture reg.

Culpeo-μArch eliminates limitations imposed by an interrupt-based approach. The comparator eliminates software interaction during the task; the scheduler only interacts with the peripheral before tasks begin and after they complete. Moving to a dedicated, modern, 8-bit ADC, instead of using the MCU's existing ADC, reduces power substantially and eliminates resource contention while adding minimal area. Recent work demonstrated an 8-bit ADC in a 130 nm process that consumes only 140 nW at an area of $0.01mm^2$ [36], [79]. The ADC we use to implement Culpeo-R-ISR on an MSP430 is also built in 130 nm, but consumes over 180 μW [17]. Therefore, Culpeo-R-μArch reduces the power consumption of ADC sampling to just 0.003% of the total MCU power, down from 4.2% with Culpeo-R-ISR[1], without degrading the accuracy of V^{safe} estimates, as shown quantitatively in Section VII-A. Since Culpeo-R includes its ADC sampling cost in a task's voltage drop, reduced power additionally allows Culpeo-R-μArch to profile lower energy tasks than Culpeo-R-ISR.

VI. Methodology

We first evaluate Culpeo's ability to generate V^{safe} values for synthetic and real-peripheral load profiles, showing benefits by direct comparison to a voltage-as-energy baseline system. We then implement a state-of-the-art scheduler and integrate Culpeo voltage reasoning to test the end-to-end value of Culpeo in full, event-based applications. Our methodology relies on testing on real energy harvesting devices to demonstrate that Culpeo is practically useful and that accurate V^{safe} values improve application performance.

A. V^{safe} Evaluation

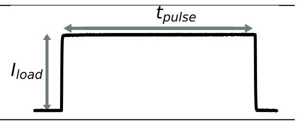
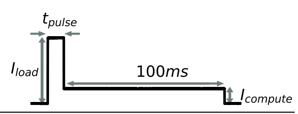
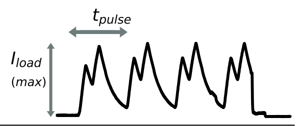
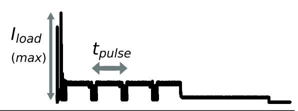
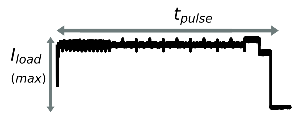
We used careful hardware and software coordination to confirm that Culpeo's V^{safe} predictions are safe.

Hardware Setup. Our experiments use the Capybara energy-harvesting platform [30] because its power system architecture supports high-ESR supercapacitors and it is preconfigured with several sensors, an ultra-low-power MCU, and a BLE radio. We disabled Capybara's reconfigurable energy storage so that its power system closely resembles the architecture in Section II-A, with a V^{off} of 1.6 V, a V^{high} of 2.56 V and V^{out} of 2.55 V. Unless otherwise noted, the energy buffer was a 45mF capacitor bank composed of dense supercapacitors [93]. To facilitate automated testing while validating V^{safe}, we modified Capybara to isolate the power system from the load side components by default. A test harness controls incoming, harvestable energy and explicitly triggers the power system to begin delivering power. In full application tests, Capybara is unmodified except for attaching an external capacitor bank. For all tests we simulate harvested solar energy using a 2.2V output in series with a potentiometer. A measurement harness collects time-series traces of energy buffer voltage and load current [89], [110].

Load Profiles. We used load current profiles from synthetic applications and real peripherals, shown in Table III, to validate V^{safe}. Synthetic profiles are generated by toggling resistor-transistor circuits tuned to sink specific loads from V^{out} under two load shapes to explore their affect on V^{safe}: Uniform and Pulsed. (t_{pulse}), representing a high-powered peripheral. The Pulsed load applies a high current pulse (I_{load} for t_{pulse}) followed by $100ms$ at $I_{\text{compute}} = 1.5mA$, representing peripheral

[1]This calculations assumes an MSP430FR5994 MCU operating at 8 MHz, with Vcc = 2.5V and a 50% SRAM hit rate [112].

TABLE III: Description of different loads used in our evaluation.

Load Type	Parameters	Current Profile
Uniform	$I_{\text{load}} = \{5mA, 10mA, 25mA, 50mA\}$ $t_{\text{pulse}} = \{1ms, 10ms, 100ms\}$	
Pulse	$I_{\text{load}} = \{5mA, 10mA, 25mA, 50mA\}$ $t_{\text{pulse}} = \{1ms, 10ms, 100ms\}$ $I_{\text{compute}} = 1.5mA$	
Gesture Recognition	$I_{\text{load}}(max) = 25mA$ $t_{\text{pulse}} = 3.5ms$	
BLE Radio	$I_{\text{load}}(max) = 13mA$ $t_{\text{pulse}} = 17ms$	
Compute Acceleration	$I_{\text{load}} = 5mA$ $t_{\text{pulse}} = 1.1s$	

activation followed by low-power computing. The peripheral traces were captured from the gesture-recognition sensor [8] and BLE radio [109] on Capybara as well as an external ARM Cortex-M4 [100] running a digit recognition workload [20], [62], [103].

Test Harness Operation. We tested the utility of Culpeo's V^{safe} estimation by monitoring whether a software task completes without power failure when started at V^{safe}. Our test harness charges the supercapacitor bank to V^{high}, disables the charging circuit, discharges the capacitor to the V^{safe} value, and then applies a load profile. Disabling incoming power represents a worst-case scenario where the V^{safe} value must ensure that the task completes using only the stored energy. We ran the real profiles using this approach and compared the accuracy of the values produced by Culpeo-PG and Culpeo-R to two baselines, a direct energy estimate and CatNap.

We also use the harness to produce known-good V^{safe} values for the synthetic load profiles. Via a brute-force binary search, the test harness finds a profile's V^{safe} by repeatedly running the profile at different V^{safe} levels until the minimum voltage is within 5 mV of V^{off}. We validated that values below the test harness' V^{safe} cause failures by running multiple trials of each synthetic load profile with V^{start} above and below the known V^{safe}. Based on our analysis, estimates more than 20 mV below V^{safe} will reliably cause failures, and estimates from V^{safe} to 20 mV below will cause failures some of the time. The validated brute-force methodology allows mathematically comparing the V^{safe} calculations of Culpeo-PG and Culpeo-R with CatNap. We also separate the Culpeo-R implementations to determine the effect of an 8-bit ADC on Culpeo-$\mu Arch$ versus the 12-bit precision used by Culpeo-R-ISR.

B. Application-Level Comparison

During scheduler tests, Capybara is not connected to the test harness. It charges and discharges based on the scheduling policy under test and provides constant, weak harvestable power, matched to a solar harvester [49], [82].

Scheduler implementation. To understand V^{safe}'s direct benefit to applications, we integrated a Culpeo-R-ISR interface into the energy-based scheduler CatNap and tested full applications on harvested energy. We modified the CatNap implementation to support a larger capacitor bank than CatNap originally targeted, which required changing the low priority scheduling policy to account for longer recharge times, and disabled additional features, e.g., adaptive voltage measurements, that would interfere with measuring the effect of voltage-based V^{safe} estimates. Further, we disabled CatNap's feasibility test, replacing it with a check that the current voltage is above V^{safe} before running a high priority task. We then added the Culpeo-R-ISR task profiling runtime as described in Section V-B and replaced CatNap's V^{safe} and "energy bucket" (V^{safe}_{multi}) calculation with Culpeo-R's. Since harvested power is stable in our evaluation setup, Culpeo-R-ISR profiles tasks one time, before the application starts.

Applications. The full applications span a range of load characteristics and requirements for success, but they all are event-driven. To guarantee that each application is feasible, we degraded the event frequency until the application successfully meets its requirements with V^{safe} for each task set to a safe value. We test each application by running three five minute trials and report the fraction of events that are successfully completed.

Periodic Sensing (PS) reads 32 samples from an IMU [102] every 4.5 seconds and has a background task that reads from a photoresistor and keeps an average of the value when extra energy is available. PS uses a 15 mF energy buffer to explore Culpeo's performance with smaller buffers. An event is considered lost if the intersample deadline is not met.

Responsive Reporting (RR) triggers three high priority tasks in response to an interrupt triggered by a GPIO pin that arrives based on a Poisson distribution with λ =45 s. The first event reads from the IMU, as in PS, the second encrypts the IMU samples, and the third sends the encrypted samples over a BLE radio and performs a low-power listen for 2 seconds awaiting a response [39]. Like PS, a background task captures light levels from a photoresistor. RR must respond to interrupts within 3 seconds or the event is lost.

Noise Monitoring & Reporting (NMR) reads 256 sample from a low power microphone [4] at 12kHz every 7 seconds, while a low priority task performs an FFT on the samples in the background. Interrupts arrive with a Poisson distribution of λ =30 s, and trigger a BLE response containing the FFT data followed by low-power listen that must respond within 15 seconds.

VII. Evaluation

Our evaluation shows that Culpeo generates V^{safe} values enabling correct operation when prior systems grossly under-

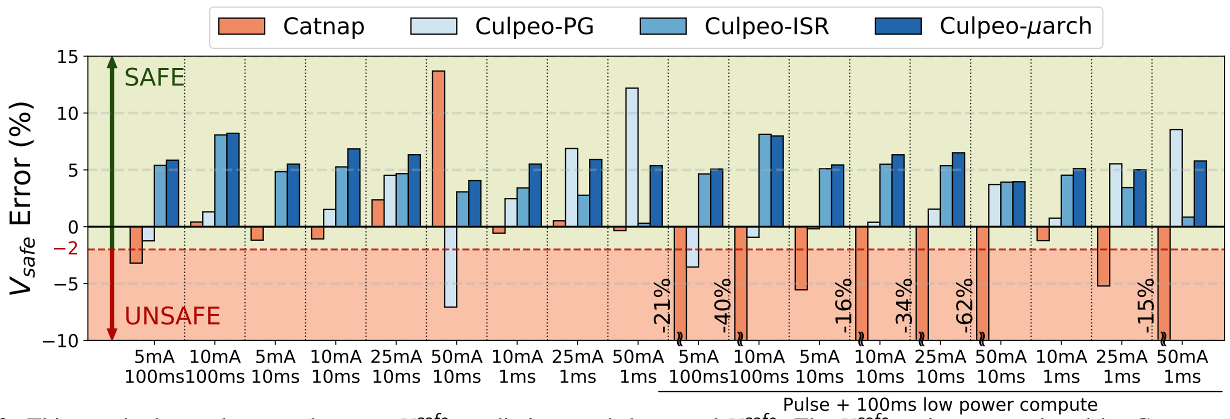


Fig. 10: This graph shows the error between V^{safe} predictions and the actual V^{safe}. The V^{safe} estimates produced by Catnap and other energy-based methods produce radically incorrect estimates. All Culpeo variants produce safe (> 0) and performant ($< 10\%$ error) estimates.

estimate a task's safe starting voltage. We also show that integrating Culpeo into a scheduler allows capturing events that would otherwise be missed.

A. *Culpeo's V^{safes} are Accurate*

Figure 10 shows the difference between the known-good V^{safe} value and the V^{safe} predicted by each approach for each synthetic load as a percentage of the total capacitor voltage range (2.5V-1.6V). For correctness, the difference must be above -2% and greater than 0% is best. Overall, the results show that ESR-aware V^{safe} estimates are much more accurate than state-of-the-art voltage-as-energy approximations. Specifically, the results show that CatNap fails when a workload has a low current "tail" after a high current pulse. Since CatNap's V^{safe} estimate ignores ESR, as load current and ESR drop grow (from $5mA$ to $50mA$), CatNap's estimates degrade. The 50 mA, 10 ms pulse shows an important side effect of CatNap's approach– for very large ESR drops, CatNap will *over*estimate the energy required as it observes a voltage drop before rebound and treats it as consumed energy. In instances where Culpeo-PG's model is accurate, it produces more accurate V^{safe} estimates than Culpeo-R because it profiles with higher sampling frequency and precision. However, Culpeo-PG fails in instances when the total load energy is high, such as for both of the 100ms load pulse + 100ms compute workloads and the 50mA,10ms pulse. These failures are likely due to compounding errors in the output booster efficiency model.

Compounding errors also cause Culpeo-PG's estimates to get more conservative as current increases at a given frequency. Such failures will become more likely over time as ESR increases; a fact that is not reflected in Figure 10 because this evaluation was performed shortly after profiling the energy buffer. In contrast, both Culpeo-R implementations provide safe estimates for all load profiles, demonstrating the robustness of the online approach. Like Culpeo-PG, Culpeo-R-ISR's estimates are less accurate as energy increases, but its estimates are always safe. The Culpeo-R-μArch is more conservative than Culpeo-R-ISR due to its lower precision. The difference is not large, except for the 50mA,1ms pulses where, ironically, Culpeo-R-ISR's slower clock rate results in an aggressive estimate because it misses the minimum voltage.

We also show that Culpeo produces safe V^{safe} estimates for three real-world peripherals (Figure 11): a gesture recognition sensor, BLE, and a compute accelerator running an MNIST digit-recognition DNN. In the graph, the top of each arrow is the V^{safe} at which each systems begins the peripheral operation, and the bottom of the arrow is the minimum observed voltage. The closer the bottom of the arrow is to V^{off} (1.6V) without going below, the more accurate. The results show that Energy-V, an end-to-end voltage based approximation that closely tracks with direct measurements, and CatNap are not safe for realistic peripheral workloads. The Energy-V estimates force the voltage so low the output booster falls into a non-operational region. CatNap fares better, but all of its estimates are still below V^{off}, triggering a power-off under normal operating conditions. In contrast, both Culpeo versions perform well. Culpeo-PG provides slightly more conservative estimates than Culpeo-R as it selects a single ESR value to use for the entire operation. Culpeo-R's estimates are very accurate– they never result in a V^{min} higher than 1.7V and never fail. Overall, the data show that Culpeo produces correct V^{safe} values, but existing systems do not.

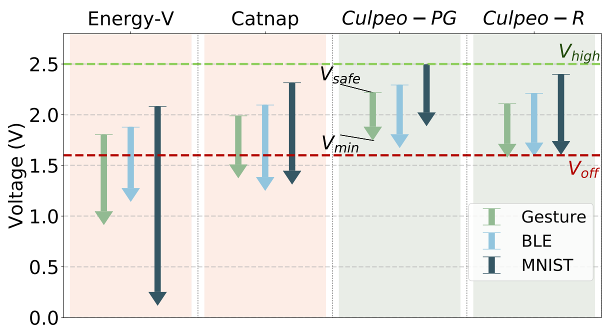


Fig. 11: Culpeo-R and Culpeo-PG's V^{safe} values (arrow tops) complete with V^{min} (arrow points) above V^{off} for tests on three real peripherals. The graph shows that Energy and CatNap V^{safe} estimates are unsafe, because they cause the device to turn off unexpectedly.

B. V^{safe} Fixes Schedulers

We first show analytically how to correct CatNap's feasibility test with V^{safe} to ensure that tasks will not fail due to ESR drop. CatNap's feasibility test can be written as $\forall t \geq 0, e_{\text{cap}}(t) > 0$. In other words, at any time, there is always energy in the capacitor *after* executing the task scheduled at time t. This test is assumed to also mean that the system will never fail to execute a task if the schedule is determined feasible, but having sufficient energy is *not* synonymous with lack of failure. Catnap's test only considers energy consumption for a task, implicitly assuming that voltage to satisfy the energy consumption is a sufficient level, i.e., $\forall t, V_t \geq V(E_t)$. Looking at the formulation for V^{safe}, which is $V^{\text{safe}} = \sum_{i=0}^{n} V(E_i) + \sum_{i=0}^{n} penalty_i + V^{\text{off}}$, it becomes clear why Catnap's test is incorrect. If for any operation i in the task $penalty_i > 0$, then $\sum_{i=0}^{n} penalty_i > 0$ and $V_t^{\text{safe}} = (\sum_{i=0}^{n} V(E_i) + \sum_{i=0}^{n} penalty_i) > V_t^{\text{catnap}}$. So, the CatNap scheduler does not meet the voltage correctness constraint. Instead the feasibility test must be expanded as:

Theorem 1: Tasks $\{\varepsilon_0, ...\varepsilon_n\}$ are feasible if $\forall\ t :: 0 \leq t \leq n, V_t \geq V_t^{\text{safe}} \wedge e_{\text{cap}}(t) > 0$, where V_t is the voltage level before executing task ε_t and $e_{\text{cap}}(t)$ is the energy after executing.

If a scheduler uses this feasibility test, then the voltage will not dip below the power-off threshold while running any task, *and* there will always be sufficient energy.

C. Culpeo Corrects Applications

A Culpeo enabled scheduler uses V^{safe} to eliminate the unexpected power failures that prevent CatNap from meeting application requirements. Figure 12 shows the percentage of captured IMU events in PS, report triggering events in RR, and both the microphone (-mic) and reporting(-BLE) events in NMR. "Events captured" is a critical, application specific performance metric that describes the fraction of events a device responds to. The data demonstrate that Culpeo prevents applications from unnecessarily missing events. CatNap misses PS and NMR-mic events because of unexpected voltage drops that trigger power failure. Powering down requires the Capybara to spend time recharging that may cause further events to be missed. The missed NMR-mic events are thus actually caused by ESR drops and recharges during the BLE reporting task, not by accesses to the low power microphone. RR fails the vast majority of its responses in CatNap because the threshold level at which to run low priority work is too low *and* the V^{safe} is too low. As a result, CatNap discharges the capacitor too far when running low priority work. When an interrupt arrives, the process of sensing, encrypting and transmitting begins, but fails, and the system transmits the sensed data on the next reboot, after the deadline has passed. CatNap performs slightly better in NMR-BLE than in RR because the BLE event stands alone– the capacitor voltage is higher when starting so the misprediction in V^{safe} matters less and thus results in fewer (but still over 50%) lost events. Culpeo eliminates the vast majority of missed events sustained by CatNap. Culpeo does experience some lost events for NMR-BLE because it waits charge to V^{safe} for the radio task and does not always charge fast enough to meet the deadline.

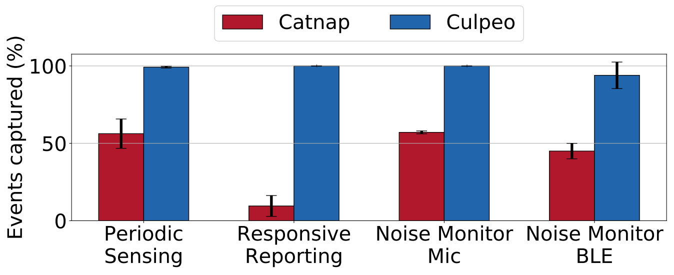


Fig. 12: Culpeo's accurate V^{safe} estimates enable high event capture rates where CatNap's estimates cause it to fail.

Finally, we examine the effect of event-interarrival time on scheduler performance. Figure 13 shows the missed event rate for PS and RR given three sampling rates– slow (6 and 60 seconds for PS and RR respectively), achievable (4.5 and 45 sec.), and too fast (3 and 30 sec.). Running the applications at a range of interarrival times demonstrates how Culpeo and CatNap react to an energy surplus or deficit. Overall, Culpeo makes the plot make sense– once the frequency drops to an achievable level given the incoming power, Culpeo guarantees high event capture rates. CatNap, however, experiences little or inverted benefits from reducing the event frequency. This phenomenon occurs because CatNap discharges the capacitor too far performing background work. The more time between events, the further CatNap will discharge the capacitor and the more likely it will fail.

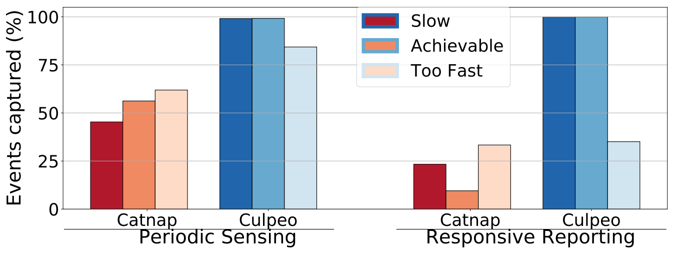


Fig. 13: Culpeo has nearly ideal event capture for achievable event rates. Catnap misses events because its V^{safe} predictions are wrong. CatNap discharges the capacitor too low while performing background work in between events, so the more time between events (i.e. the slower the event arrival rate) the more likely the task is to fail.

VIII. Related Work

Culpeo relates to work spanning a wide range of topics including intermittent systems, supercapacitor enabled sensors and energy-aware programming.

Intermittent systems. Culpeo relates to strides made in prior work to expand the capabilities of batteryless, energy-harvesting systems. The initiative relies on checkpointing techniques [11], [12], [13], [50], [51], [69], [70], [85], [113] and task-based methods [28], [29], [67], [68] to tolerate failures during operation. Able to tolerate power failures, subsequent works focused on adding functionality such as system flexibility [30], [43], timeliness [44], [60], failure resistant timers [31], interrupts [87], multitasking [121] and peripheral handling through power failures [14], [18], [86]. Additional work has sought to provide guarantees for intermittent systems, including formal checks for data correctness and freshness [105], [106] and formal models of peripheral correctness [15], assuming that tasks terminate. Prior works also reduce the difficulty of developing batteryless systems with tools to eliminate bugs specific to the domain [27], [72], [104], model intermittent execution [98], [117] and enable beginner-friendly languages [59] and repeatable energy traces [124]. As a result, advanced applications on batteryless devices have proliferated, including DNN processing [41], [48], image capture [80] and recognition [82], environmental monitoring [5], and gaming [32]. Finally, recent works [47], [71], [121], [122] present schedulers for periodic and reactive intermittent operation, which we demonstrate to fail with ESR voltage drops. Thus, Culpeo closes a gap in the literature between programming models and hardware to enable higher power peripherals and compute acceleration. Culpeo likewise complements work providing static termination checking for intermittent programs [29], [37]. As these works base their probabilistic guarantees on energy consumption models only, they can incorrectly conclude a task likely terminates when ESR drops will actually pull the voltage beneath the power-off threshold. Programmers using these tools should also use Culpeo-PG to check that a task's safe voltage when accounting for ESR is not too high for the device to support (i.e., will cause non-termination).

Supercapacitor Enabled Embedded Systems. Culpeo is motivated by numerous energy-harvesting platforms that rely solely on supercapacitors for energy storage. No prior work examines the effect of high ESR on applications that run on supercapacitor-only systems. Early batteryless supercapacitor motes, such as Everlast [96] and Ambimax [84] focus on charging supercapacitors efficiently and use large supercapacitors (>10F) for which ESR is not a primary concern. Additional work defined principles for building efficient power systems in energy-harvesting, supercapacitor based devices [21], [58], and are complementary to Culpeo's efforts to improve the capability of supercapacitor based devices. More recent systems either seek out low ESR capacitors, increasing volume or cost as in the Camaroptera [82] and TA-1 [66], or engineer their applications to compensate [38]. Both the sensor node and EdbSat instantiations of the Capybara power system use compact, high ESR supercapacitors for energy storage [30], making them primary targets for Culpeo. The Capybara power system [30] uses an output booster to compensate for voltage drops due to high ESR, but the work does not describe the limitations that ESR places on how energy can be extracted from the supercapacitor. Capybara's task based programming model makes no guarantees about completion, and requires system developers to test all tasks before deployment. Several hybrid supercapacitor-battery sensing nodes including Prometheus [52], Trio [34], HypoEnergy [75] and numerous others as described in [56], use a supercapacitor to reduce the primary battery cycling. Culpeo, in contrast, targets supercapacitor only systems.

Culpeo is aligned with work that modeled supercapacitors in wireless sensor networks (WSNs) to provide closed-loop device simulations [53], [120], analysis of energy over time [21], [23], [74], [115], [119] and the effect of the charging routine on state-of-charge [6]. None of these models focus on the immediate ESR drop because they target low current, long-lifetime operations where other effects (e.g. charge redistribution, leakage) are more prominent.

Energy-aware Programming. Culpeo is designed to aid energy-aware programming languages and models in successful interactions with supercapacitor based systems. Energy Types [26] and ENT [22] are energy-aware type systems whose guarantees fail in the presence of ESR as neither considers the rapidly changing energy state of a batteryless system nor has constructs to easily support ESR drop. Eon associates tasks with energy levels [99] and could better evaluate available energy in batteryless systems using Culpeo. Levels permits "optional" code to run based on energy availability, estimated using a simple battery model, and would require an awareness of ESR to run on a batteryless system [61]. Pixie is a WSN programming model that allocates energy to tasks within a dataflow graph via resource tickets [65]. Pixie's energy allocator and energy broker abstractions would benefit from this work's treatment of ESR as a first class concern. Additional efforts allow programmers to trade accuracy for energy via approximation [10], [46], [91] and adaptation to QoS requirements [54], [125] that all rely on a means of measuring energy consumption. While these efforts are not designed to run on energy-harvesting systems, they demonstrate the importance of accurate energy models to enable optimization at higher levels of the system stack.

IX. Conclusion & Future Work

This work is the first to identify ESR as an important factor in designing intermittent software systems. ESR-induced voltage drops break correctness assumptions of prior work, which reason about energy without considering voltage. As a remedy, we present Culpeo, a hardware/software interface that enables intermittent system designers to reason about power system effects like ESR. Culpeo's evaluation shows that disregarding ESR results in unexpected system failures and missed events, which can be alleviated by integrating Culpeo into intermittent system schedulers. Future work should explore uses of Culpeo beyond scheduling:

Language Constructs. Disregarding voltage can break energy-aware language constructs. Energy-Types [26] provides a type system to enable energy-aware programming. A well-typed program preserves an invariant that program elements associated with high energy availability (e.g. battery full) may interact with elements associated with low availability (e.g. battery nearly empty), but not vice-versa. This invariant is insufficient for intermittent systems. A program element could take little energy but have a high ESR drop. Calling this element with little energy respects the invariant but could cause the system to fail. To enable sophisticated resource-aware programming on intermittent systems, languages must have abstractions for voltage-awareness.

Probabilistic Resource Reasoning. To make forward progress, a task must be able to complete when starting execution with a full capacitor. Compile-time tools use probabilistic energy models [29] to give bounds on completion probability and to aid the programmer in sizing tasks. As we have shown, energy modelling is *not enough* as a task could with all likelihood have enough energy to run and still fail. Future efforts in probabilistic reasoning must model voltage as a resource.

Acknowledgment

We thank the anonymous reviewers for their valuable feedback. We thank members of the Abstract research group and Limin Jia for insightful feedback on these ideas. This work was supported in part by National Science Foundation Award #1751029 and by the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.

References

[1] "Msp430f5529 dma hangs in uart rx," https://e2e.ti.com/support/microcontrollers/msp-low-power-microcontrollers-group/msp430/f/msp-low-power-microcontroller-forum/310488/msp430f5529-dma-hangs-in-uart-rx, accessed: 2022-04-20.

[2] "Tps61200 0.3-v input voltage, adjustable ouput voltage boost converter with 1.3-a switches, 3-mm x 3-mm qfn," https://www.ti.com/product/TPS63000, accessed: 2022-04-21.

[3] "Tps61200 0.3-v input voltage, adjustable ouput voltage boost converter with 1.3-a switches, 3-mm x 3-mm qfn," https://www.ti.com/product/TPS61200, accessed: 2022-04-21.

[4] "Spu0414hr5h-sb amplified sisonictm microphone," https://media.digikey.com/pdf/Data%20Sheets/Knowles%20Acoustics%20PDFs/SPU0414HR5H-SB.pdf, 2012, accessed: 2022-04-21.

[5] M. Afanasov, N. A. Bhatti, D. Campagna, G. Caslini, F. M. Centonze, K. Dolui, A. Maioli, E. Barone, M. H. Alizai, J. H. Siddiqui, and L. Mottola, "Battery-less zero-maintenance embedded sensing at the mithræum of circus maximus," in *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, ser. SenSys '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 368–381. [Online]. Available: https://doi.org/10.1145/3384419.3430722

[6] J. I. Ahn, D. Kim, R. Ha, and H. Cha, "State-of-charge estimation of supercapacitors in transiently-powered sensor nodes," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2021.

[7] T. Ashworth, "Characterizing the esr of a supercapacitor with the mfia," https://www.zhinst.com/americas/en/blogs/using-mfia-impedance-analyzer-characterize-esr-super-capacitor, 2016, accessed: 2021-11-11.

[8] Avago Technologies, "Apds-9960 digital proximity, ambient light, rgb and gesture sensor– data sheet," https://cdn.sparkfun.com/assets/learn_tutorials/3/2/1/Avago-APDS-9960-datasheet.pdf, 2013, accessed: 2020-03-06.

[9] AVX, "BestCap® Ultra-low ESR High Power Pulse Supercapacitors," http://catalogs.avx.com/BestCap.pdf, 2019.

[10] W. Baek and T. M. Chilimbi, "Green: a framework for supporting energy-conscious programming using controlled approximation," in *ACM Sigplan Notices*, vol. 45. ACM, 2010, pp. 198–209. [Online]. Available: http://dl.acm.org/citation.cfm?id=1806620

[11] S. S. Baghsorkhi and C. Margiolas, "Automating efficient variable-grained resiliency for low-power iot systems," in *Proceedings of the 2018 International Symposium on Code Generation and Optimization*, ser. CGO 2018. New York, NY, USA: Association for Computing Machinery, 2018, p. 38–49. [Online]. Available: https://doi.org/10.1145/3168816

[12] 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.

[13] D. Balsamo, A. S. Weddell, G. V. Merrett, B. M. Al-Hashimi, D. Brunelli, and L. Benini, "Hibernus: Sustaining computation during intermittent supply for energy-harvesting systems," *IEEE Embedded Systems Letters*, vol. 7, no. 1, pp. 15–18, 2015.

[14] G. Berthou, T. Delizy, K. Marquet, T. Risset, and G. Salagnac, "Peripheral state persistence for transiently-powered systems," in *2017 Global Internet of Things Summit (GIoTS)*, June 2017, pp. 1–6.

[15] G. Berthou, P.-E. Dagand, D. Demange, R. Oudin, and T. Risset, "Intermittent computing with peripherals, formally verified," in *The 21st ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems*, ser. LCTES '20, 2020, pp. 85–96. [Online]. Available: https://doi.org/10.1145/3372799.3394365

[16] N. A. Bhatti and L. Mottola, "HarvOS: efficient code instrumentation for transiently-powered embedded sensing." ACM Press, 2017, pp. 209–219. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3055031.3055082

[17] J. Borgeson, "Ultra-low-power pioneers: Ti slashes total mcu power by 50 percent with new "wolverine" mcu platform," https://www.ti.com/lit/wp/slay019a/slay019a.pdf, 2012, accessed: 2022-04-21.

[18] A. Branco, L. Mottola, M. H. Alizai, and J. H. Siddiqui, "Intermittent asynchronous peripheral operations," in *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*, 2019, pp. 55–67.

[19] D. Brooks, R. P. Dick, R. Joseph, and L. Shang, "Power, thermal, and reliability modeling in nanometer-scale microprocessors," *IEEE Micro*, vol. 27, no. 3, pp. 49–62, 2007.

[20] J. Brownlee, "Handwritten digit recognition using convolutional neural networks in python with keras," 2016. [Online]. Available: https://machinelearningmastery.com/handwritten-digit-recognition-using-convolutional-neural-networks-python-keras/

[21] D. Brunelli, C. Moser, L. Thiele, and L. Benini, "Design of a solar-harvesting circuit for batteryless embedded systems," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 56, no. 11, pp. 2519–2528, 2009.

[22] A. Canino and Y. D. Liu, "Proactive and adaptive energy-aware programming with mixed typechecking." ACM Press, 2017, pp. 217–232. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3062341.3062356

[23] R. Chai, Y. Zhang, G. Sun, and H. Li, "Self-aware power management for maintaining event detection probability of supercapacitor-powered cyber-physical systems," *ACM Trans. Cyber-Phys. Syst.*, vol. 4, no. 4, Jul. 2020. [Online]. Available: https://doi-org.cmu.idm.oclc.org/10.1145/3375407

[24] W.-M. Chen, T.-S. Cheng, P.-C. Hsiu, and T.-W. Kuo, "Value-based task scheduling for nonvolatile processor-based embedded devices," in *Real-Time Systems Symposium (RTSS), 2016 IEEE*. IEEE, 2016, pp. 247–256.

[25] M. Chetto, "Optimal scheduling for real-time jobs in energy harvesting computing systems," *IEEE Transactions on Emerging Topics in Computing*, no. 1, pp. 1–1, 2014.

[26] M. Cohen, H. S. Zhu, E. E. Senem, and Y. D. Liu, "Energy types," in *ACM SIGPLAN Notices*, vol. 47. ACM, 2012, pp. 831–850. [Online]. Available: http://dl.acm.org/citation.cfm?id=2384676

[27] A. Colin, G. Harvey, B. Lucia, and A. P. Sample, "An energy-interference-free hardware-software debugger for intermittent energy-harvesting systems," *ACM SIGPLAN Notices*, vol. 51, no. 4, pp. 577–589, 2016.

[28] A. Colin and B. Lucia, "Chain: tasks and channels for reliable intermittent programs," in *Proceedings of the 2016 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications*. ACM, 2016, pp. 514–530.

[29] A. Colin and B. Lucia, "Termination checking and task decomposition for task-based intermittent programs," in *Proceedings of the 27th International Conference on Compiler Construction*, ser. CC 2018, 2018, pp. 116–127.

[30] 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*, ser. ASPLOS '18. New York, NY, USA: ACM, 2018, pp. 767–781. [Online]. Available: http://doi.acm.org/10.1145/3173162.3173210

[31] 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.

[32] J. de Winkel, V. Kortbeek, J. Hester, and P. Pawełczak, “Battery-free game boy,” *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 4, no. 3, Sep. 2020. [Online]. Available: https://doi.org/10.1145/3411839

[33] Digikey, “Capacitors categories,” https://www.digikey.com/en/products/category/capacitors/3, 2022, accessed: 2022-06-30.

[34] P. Dutta, J. Hui, J. Jeong, S. Kim, C. Sharp, J. Taneja, G. Tolle, K. Whitehouse, and D. Culler, “Trio: enabling sustainable and scalable outdoor wireless sensor network deployments,” in *2006 5th International Conference on Information Processing in Sensor Networks*, 2006, pp. 407–415.

[35] Eaton, “Application Guidelines,” https://www.eaton.com/content/dam/eaton/products/electronic-components/resources/technical/eaton-supercapacitor-application-guidelines.pdf, 2017, accessed August 12, 2021.

[36] M. ElAnsary, J. Xu, J. Sales Filho, G. Dutta, L. Long, A. Shoukry, C. Tejeiro, C. Tang, E. Kilinc, J. Joshi *et al.*, “28.8 multi-modal peripheral nerve active probe and microstimulator with on-chip dual-coil power/data transmission and 64 2 nd-order opamp-less $\delta\sigma$ adcs,” in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64. IEEE, 2021, pp. 400–402.

[37] F. Erata, A. Goknil, E. Yıldız, K. S. Yıldırım, R. Piskac, J. Szefer, and G. Sezgin, “Etap: Energy-aware timing analysis of intermittent programs,” 2022. [Online]. Available: https://arxiv.org/abs/2201.11433

[38] F. Fraternali, B. Balaji, Y. Agarwal, L. Benini, and R. Gupta, “Pible: Battery-free mote for perpetual indoor ble applications,” in *Proceedings of the 5th Conference on Systems for Built Environments*, ser. BuildSys ’18. New York, NY, USA: Association for Computing Machinery, 2018, p. 168–171. [Online]. Available: https://doi.org/10.1145/3276774.3282822

[39] P. Gavrikov, P. E. Verboket, T. Ungan, M. Müller, M. Lai, C. Schindelhauer, L. M. Reindl, and T. Wendt, “Using bluetooth low energy to trigger an ultra-low power fsk wake-up receiver,” in *2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, 2018, pp. 781–784.

[40] H. E. Ghor, M. Chetto, and R. H. Chehade, “A real-time scheduling framework for embedded systems with environmental energy harvesting,” *Computers & Electrical Engineering*, vol. 37, no. 4, pp. 498–510, 2011.

[41] G. Gobieski, B. Lucia, and N. Beckmann, “Intelligence beyond the edge: Inference on intermittent embedded systems,” in *Proceedings of the International Symposium on Architecture Support for Programming Languages and Operating Systems*, 2019.

[42] M. S. Halper and J. C. Ellenbogen, “Supercapacitors: A brief overview,” *The MITRE Corporation, McLean, Virginia, USA*, pp. 1–34, 2006.

[43] J. Hester and J. Sorber, “Flicker: Rapid prototyping for the batteryless internet-of-things,” in *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys ’17. New York, NY, USA: ACM, 2017, pp. 19:1–19:13.

[44] 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*. ACM, 2017.

[45] R. Heydon, *Bluetooth low energy: the developer’s handbook*. Prentice Hall Upper Saddle River, 2013, vol. 1.

[46] H. Hoffmann, “Jouleguard: energy guarantees for approximate applications,” in *Proceedings of the 25th Symposium on Operating Systems Principles*, 2015, pp. 198–214.

[47] B. Islam and S. Nirjon, “Scheduling computational and energy harvesting tasks in deadline-aware intermittent systems,” in *2020 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2020, pp. 95–109.

[48] B. Islam and S. Nirjon, “Zygarde: Time-sensitive on-device deep inference and adaptation on intermittently-powered systems,” *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 4, no. 3, Sep. 2020. [Online]. Available: https://doi.org/10.1145/3411808

[49] N. Jackson, J. Adkins, and P. Dutta, “Capacity over capacitance for reliable energy harvesting sensors,” in *Proceedings of the 18th International Conference on Information Processing in Sensor Networks*, ser. IPSN ’19. New York, NY, USA: Association for Computing Machinery, 2019, p. 193–204. [Online]. Available: https://doi.org/10.1145/3302506.3310400

[50] 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 *VLSI Design and 2014 13th International Conference on Embedded Systems, 2014 27th International Conference on*. IEEE, 2014, pp. 330–335.

[51] H. Jayakumar, A. Raha, J. R. Stevens, and V. Raghunathan, “Energy-aware memory mapping for hybrid fram-sram mcus in intermittently-powered iot devices,” *ACM Trans. Embed. Comput. Syst.*, vol. 16, no. 3, Apr. 2017. [Online]. Available: https://doi.org/10.1145/2983628

[52] X. Jiang, J. Polastre, and D. Culler, “Perpetual environmentally powered sensor networks,” in *Proceedings of the 4th international symposium on Information processing in sensor networks*. IEEE Press, 2005, p. 65. [Online]. Available: http://dl.acm.org/citation.cfm?id=1147765

[53] A. Kailas, D. Brunelli, and M. A. Ingram, “A simple energy model for the harvesting and leakage in a supercapacitor,” in *2012 IEEE International Conference on Communications (ICC)*, 2012, pp. 6278–6282.

[54] A. Kansal, S. Saponas, A. B. Brush, K. S. McKinley, T. Mytkowicz, and R. Ziola, “The latency, accuracy, and battery (LAB) abstraction: programmer productivity and energy efficiency for continuous mobile context sensing.” ACM Press, 2013, pp. 661–676. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2509136.2509541

[55] Kemet Electronics Corporation, “Supercapacitors FT Series,” https://content.kemet.com/datasheets/KEM_S6014_FT.pdf, p. 2, 2020.

[56] J. A. Khan, H. K. Qureshi, and A. Iqbal, “Energy management in wireless sensor networks: A survey,” *Computers & Electrical Engineering*, vol. 41, pp. 159–176, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045790614001773

[57] B. K. Kim, S. Sy, A. Yu, and J. Zhang, “Electrochemical supercapacitors for energy storage and conversion,” *Handbook of Clean Energy Systems*, pp. 1–25, 2015.

[58] S. Kim and P. H. Chou, “Size and topology optimization for supercapacitor-based sub-watt energy harvesters,” *IEEE Transactions on Power Electronics*, vol. 28, no. 4, pp. 2068–2080, 2013.

[59] V. Kortbeek, A. Bakar, S. Cruz, K. S. Yildirim, P. Pawełczak, and J. Hester, “Bfree: Enabling battery-free sensor prototyping with python,” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 4, no. 4, pp. 1–39, 2020.

[60] V. Kortbeek, K. S. Yıldırım, A. Bakar, J. Sorber, J. Hester, and P. Pawełczak, “Time-sensitive intermittent computing meets legacy software,” in *In Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS’20. New York, NY, USA: ACM, 2020. [Online]. Available: https://doi.org/10.1145/3373376.3378476

[61] A. Lachenmann, P. J. Marrón, D. Minder, and K. Rothermel, “Meeting lifetime goals with energy levels,” in *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*, ser. SenSys ’07. New York, NY, USA: Association for Computing Machinery, 2007, p. 131–144. [Online]. Available: https://doi.org/10.1145/1322263.1322277

[62] Y. Le Cun, L. Jackel, B. Boser, J. Denker, H. Graf, I. Guyon, D. Henderson, R. Howard, and W. Hubbard, “Handwritten digit recognition: applications of neural network chips and automatic learning,” *IEEE Communications Magazine*, vol. 27, no. 11, pp. 41–46, 1989.

[63] Y. Lee, G. Kim, S. Bang, Y. Kim, I. Lee, P. Dutta, D. Sylvester, and D. Blaauw, “A modular 1mm 3 die-stacked sensing platform with optical communication and multi-modal energy harvesting,” in *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2012 IEEE International*. IEEE, 2012, pp. 402–404.

[64] S. Liu, Q. Qiu, and Q. Wu, “Energy aware dynamic voltage and frequency selection for real-time systems with energy harvesting,” in *Proceedings of the conference on Design, automation and test in Europe*. ACM, 2008, pp. 236–241.

[65] K. Lorincz, B.-r. Chen, J. Waterman, G. Werner-Allen, and M. Welsh, “Resource aware programming in the pixie os,” in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, 2008, pp. 211–224.

[66] B. Lucia, B. Denby, Z. Manchester, H. Desai, E. Ruppel, and A. Colin, “Computational nanosatellite constellations: Opportunities and challenges,” *GetMobile: Mobile Comp. and Comm.*, vol. 25, no. 1, p. 16–23, Jun. 2021. [Online]. Available: https://doi.org/10.1145/3471440.3471446

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

[68] K. Maeng, A. Colin, and B. Lucia, “Alpaca: Intermittent execution without checkpoints,” in *Proceedings of the 2017 ACM SIGPLAN*

International Conference on Object-Oriented Programming, Systems, Languages, and Applications. ACM, 2017.

[69] K. Maeng and B. Lucia, "Adaptive dynamic checkpointing for safe efficient intermittent computing," in *Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation*. USENIX Association, 2018, pp. 129–144.

[70] 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.

[71] K. Maeng and B. 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*, 2020, pp. 1005–1021.

[72] A. Maioli, L. Mottola, M. H. Alizai, and J. H. Siddiqui, "Discovering the hidden anomalies of intermittent computing," in *Proceedings of the 2021 International Conference on Embedded Wireless Systems and Networks*, ser. EWSN '21, 2021.

[73] P. Mars, "Coupling a supercapacitor with a small energy harvesting source," https://www.eetimes.com/coupling-a-supercapacitor-with-a-small-energy-harvesting-source, 2012, accessed: 2021-11-15.

[74] G. V. Merrett, A. S. Weddell, A. P. Lewis, N. R. Harris, B. M. Al-Hashimi, and N. M. White, "An empirical energy model for supercapacitor powered wireless sensor nodes," in *2008 Proceedings of 17th International Conference on Computer Communications and Networks*, 2008, pp. 1–6.

[75] A. Mirhoseini and F. Koushanfar, "Hypoenergy. hybrid supercapacitor-battery power-supply optimization for energy efficiency," in *2011 Design, Automation Test in Europe*, 2011, pp. 1–4.

[76] A. Mirhoseini, E. M. Songhori, and F. Koushanfar, "Idetic: A high-level synthesis approach for enabling long computations on transiently-powered asics," in *Pervasive Computing and Communications (PerCom), 2013 IEEE International Conference on*. IEEE, 2013, pp. 216–224.

[77] C. Moser, D. Brunelli, L. Thiele, and L. Benini, "Real-time scheduling for energy harvesting sensor nodes," *Real-Time Systems*, vol. 37, pp. 233–260, 10 2007.

[78] Murata, "Murata supercapacitor technical note," 2020. [Online]. Available: https://www.murata.com/~/media/webrenewal/products/capacitor/edlc/techguide/electrical/c2m1cxs-053.ashx

[79] B. Murmann, "Adc performance survey 1997-2021," http://web.stanford.edu/~murmann/adcsurvey.html, accessed: 2022-04-21.

[80] S. Naderiparizi, A. N. Parks, Z. Kapetanovic, B. Ransford, and J. R. Smith, "Wispcam: A battery-free rfid camera," in *RFID (RFID), 2015 IEEE International Conference on*. IEEE, 2015, pp. 166–173.

[81] M. Nádvorníková, J. Banout, D. Herák, and V. Verner, "Evaluation of physical properties of rice used in traditional kyrgyz cuisine," *Food Science & Nutrition*, vol. 6, no. 6, pp. 1778–1787, 2018.

[82] M. Nardello, H. Desai, D. Brunelli, and B. Lucia, "Camaroptera: A batteryless long-range remote visual sensing system," in *Proceedings of the 7th International Workshop on Energy Harvesting & Energy-Neutral Sensing Systems*, ser. ENSsys'19. New York, NY, USA: Association for Computing Machinery, 2019, p. 8–14. [Online]. Available: https://doi.org/10.1145/3362053.3363491

[83] E. W. Online, "Supercapacitor esr, specification and life time," 2020. [Online]. Available: https://passive-components.eu/supercapacitor-esr-specification-and-life-time/

[84] C. Park and P. H. Chou, "Ambimax: Autonomous energy harvesting platform for multi-supply wireless sensor nodes," in *Sensor and Ad Hoc Communications and Networks, 2006. SECON'06. 2006 3rd Annual IEEE Communications Society on*, vol. 1. IEEE, 2006, pp. 168–177. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/4068119/

[85] B. Ransford, J. Sorber, and K. Fu, "Mementos: System support for long-running computation on rfid-scale devices," *Acm Sigplan Notices*, vol. 47, no. 4, pp. 159–170, 2012.

[86] 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.

[87] E. Ruppel and B. Lucia, "Transactional concurrency control for intermittent, energy harvesting, computing systems," in *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*. ACM, 2019.

[88] A. Sabovic, A. K. Sultania, and J. Famaey, "Demonstration of an energy-aware task scheduler for battery-less iot devices," in *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 586–587. [Online]. Available: https://doi.org/10.1145/3485730.3493358

[89] Saleae, Inc, "Saleae logic 8 tech specs," https://saleae.com, 2020, accessed: 2020-03-06.

[90] A. P. Sample, D. J. Yeager, P. S. Powledge, A. V. Mamishev, and J. R. Smith, "Design of an rfid-based battery-free programmable sensing platform," *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no. 11, pp. 2608–2615, 2008.

[91] A. Sampson, W. Dietl, E. Fortuna, D. Gnanapragasam, L. Ceze, and D. Grossman, "EnerJ: Approximate data types for safe and general low-power computation," in *ACM SIGPLAN Notices*, vol. 46. ACM, 2011, pp. 164–174. [Online]. Available: http://dl.acm.org/citation.cfm?id=1993518

[92] J. San Miguel, K. Ganesan, M. Badr, and N. E. Jerger, "The eh model: Analytical exploration of energy-harvesting architectures," *IEEE Computer Architecture Letters*, vol. 17, no. 1, pp. 76–79, 2017.

[93] Seiko Instruments Inc., "CPX Capacitors CPX3225A752D," https://www.sii.co.jp/en/me/datasheets/chip-capacitor/cpx3225a752d/.

[94] SemTech, "Sx1276/77/78/79 - 137 mhz to 1020 mhz low power long range transceiver," https://www.mouser.com/datasheet/2/761/sx1276-1278113.pdf, p. 14, 2016, accessed: 2022-06-30.

[95] SEMTECH, "LoraWAN Standard," https://www.semtech.com/lora/lorawan-standard, 2021, visited August 11, 2021.

[96] F. Simjee, D. Sharma, and P. H. Chou, "Everlast: Long-life, supercapacitor-operated wireless sensor node," in *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, ser. SenSys '05. New York, NY, USA: Association for Computing Machinery, 2005, p. 315. [Online]. Available: https://doi-org.cmu.idm.oclc.org/10.1145/1098918.1098980

[97] Skeleton Technologies, "White Paper: Equivalent Series Resistance," https://cdn2.hubspot.net/hubfs/1188159/Whitepapers/160809_whitepaper_ESR.pdf, 2016, accessed August 12, 2021.

[98] S. T. Sliper, W. Wang, N. Nikoleris, A. S. Weddell, and G. V. Merrett, "Fused: Closed-loop performance and energy simulation of embedded systems," in *2020 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 2020.

[99] J. Sorber, A. Kostadinov, M. Garber, M. Brennan, M. D. Corner, and E. D. Berger, "Eon: a language and runtime system for perpetual systems," in *Proceedings of the 5th international conference on Embedded networked sensor systems*. ACM, 2007, pp. 161–174.

[100] STMicroelectronics, "NUCLEO-G474RE," https://www.st.com/en/evaluation-tools/nucleo-g474re.html.

[101] STMicroelectronics, "Stm32 power shield, nucleo expansion board for power consumption measurement (um2243)," https://www.st.com/en/evaluation-tools/x-nucleo-lpm01a.html.

[102] StMicroelectronics, "Nemo inertial module: always-on 3d accelerometer and 3d gyroscope, datasheet - production data," https://www.st.com/resource/en/datasheet/lsm6ds3.pdf, 2017, accessed: 2019-07-15.

[103] STMicroelectronics, "Ai expansion pack for stm32cubemx," 2020. [Online]. Available: https://www.st.com/en/embedded-software/x-cube-ai.html

[104] M. Surbatovich, L. Jia, and B. Lucia, "I/o dependent idempotence bugs in intermittent systems," in *Proceedings of the 2019 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications*. ACM, 2019.

[105] M. Surbatovich, L. Jia, and B. Lucia, "Automatically enforcing fresh and consistent inputs in intermittent systems," in *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, ser. PLDI 2021. New York, NY, USA: Association for Computing Machinery, 2021, p. 851–866. [Online]. Available: https://doi.org/10.1145/3453483.3454081

[106] M. Surbatovich, B. Lucia, and L. Jia, "Towards a formal foundation of intermittent computing," *Proceedings of the ACM on Programming Languages*, vol. 4, no. OOPSLA, Nov. 2020. [Online]. Available: https://doi.org/10.1145/3428231

[107] Taiyo Yuden, "Multilayer ceramic capacitors," https://media.digikey.com/pdf/Data%20Sheets/Taiyo%20Yuden%20PDFs%20URL%20links/AMK325ABJ337MM-P_SS.pdf, p. 2, 2022, accessed: 2022-06-30.

[108] V. Talla, B. Kellogg, S. Gollakota, and J. R. Smith, "Battery-free cellphone," *Proc. ACM Interact. Mob. Wearable Ubiquitous*

Technol., vol. 1, no. 2, Jun. 2017. [Online]. Available: https://doi.org/10.1145/3090090

[109] Texas Instruments, "SimpleLink™ 32-bit Arm Cortex-M3 multiprotocol 2.4 GHz wireless MCU with 128kB Flash," https://www.ti.com/product/CC2650.

[110] Texas Instruments, "10ua-100ma, 0.05% error, high-side current sensing solution reference design tipd135," http://www.ti.com/tool/TIPD135, 2020, accessed: 2020-03-06.

[111] TI Inc., "MSP430FR59xx Mixed-Signal Microcontrollers (Rev. F)," http://www.ti.com/lit/ds/symlink/msp430fr5969.pdf, p. 19, 2017.

[112] TI Inc., "Msp430fr599x, msp430fr596x mixed-signal microcontrollers (rev d)," http://www.ti.com/lit/ds/symlink/msp430fr5969.pdf, p. 31, 2021.

[113] J. Van Der Woude and M. Hicks, "Intermittent computation without hardware support or programmer intervention," in *Proceedings of OSDI'16: 12th USENIX Symposium on Operating Systems Design and Implementation*, 2016, p. 17.

[114] R. Vicentini, L. M. Da Silva, E. P. Cecilio Junior, T. A. Alves, W. G. Nunes, and H. Zanin, "How to measure and calculate equivalent series resistance of electric double-layer capacitors," *Molecules*, vol. 24, no. 8, p. 1452, 2019.

[115] A. S. Weddell, G. V. Merrett, T. J. Kazmierski, and B. M. Al-Hashimi, "Accurate supercapacitor modeling for energy harvesting wireless sensor nodes," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 58, no. 12, pp. 911–915, 2011.

[116] H. Williams, M. Moukarzel, and M. Hicks, "Failure sentinels: ubiquitous just-in-time intermittent computation via low-cost hardware support for voltage monitoring," in *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2021, pp. 665–678.

[117] S. C. Wong, S. T. Sliper, W. Wang, A. S. Weddell, S. Gauthier, and G. V. Merrett, "Energy-aware hw/sw co-modeling of batteryless wireless sensor nodes," in *Proceedings of the 8th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*, ser. ENSsys '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 57–63. [Online]. Available: https://doi-org.proxy.library.cmu.edu/10.1145/3417308.3430272

[118] F. Yang, A. S. Thangarajan, S. Michiels, W. Joosen, and D. Hughes, "Morphy: Software defined charge storage for the iot," in *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, 2021, pp. 248–260.

[119] H. Yang and Y. Zhang, "Analysis of supercapacitor energy loss for power management in environmentally powered wireless sensor nodes," *IEEE transactions on power electronics*, vol. 28, no. 11, pp. 5391–5403, 2013.

[120] H. Yang and Y. Zhang, "A task scheduling algorithm based on supercapacitor charge redistribution and energy harvesting for wireless sensor nodes," *Journal of Energy Storage*, vol. 6, pp. 186 – 194, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2352152X16300494

[121] K. S. Yıldırım, A. Y. Majid, D. Patoukas, K. Schaper, P. Pawelczak, and J. Hester, "Ink: Reactive kernel for tiny batteryless sensors," in *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2018, pp. 41–53.

[122] J. Zhan, A. S. Weddell, and G. V. Merrett, "Adaptive energy budgeting for atomic operations in intermittently-powered systems," in *Proceedings of the 8th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*, ser. ENSsys '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 82–83. [Online]. Available: https://doi.org/10.1145/3417308.3430277

[123] F. Zhang and A. Burns, "Schedulability analysis for real-time systems with edf scheduling," *IEEE Transactions on Computers*, vol. 58, no. 9, pp. 1250–1258, 2009.

[124] H. Zhang, M. Salajegheh, K. Fu, and J. Sorber, "Ekho: bridging the gap between simulation and reality in tiny energy-harvesting sensors," in *Proceedings of the 4th Workshop on Power-Aware Computing and Systems*. ACM, 2011, p. 9.

[125] Y. Zhu and V. J. Reddi, "Greenweb: Language extensions for energy-efficient mobile web computing," *SIGPLAN Not.*, vol. 51, no. 6, p. 145–160, jun 2016. [Online]. Available: https://doi.org/10.1145/2980983.2908082

[126] D. Zogbi, "Supercapacitors: A 25-year market review," https://www.tti.com/content/ttiinc/en/resources/marketeye/categories/passives/me-zogbi-20200806.html, 2020, accessed: 2021-11-11.