TECHNISCHE UNIVERSITÄT DRESDEN

DOCTORAL THESIS

Methods and Tools for Battery-free Wireless Networks

Author:
Kai Geißdörfer

Advisor:
Prof. Dr. Marco Zimmerling

A thesis submitted in fulfillment of the requirements for the degree of Doktoringenieur (Dr.-Ing.)

in the

Networked Embedded Systems Lab
Fakultät Informatik

July 5, 2022

To Sarah.

Abstract

Embedding small wireless sensors into the environment allows for monitoring physical processes with high spatio-temporal resolutions. Today, these devices are equipped with a battery to supply them with power. Despite technological advances, the high maintenance cost and environmental impact of batteries prevent the widespread adoption of wireless sensors. *Battery-free devices* that store energy harvested from light, vibrations, and other ambient sources in a capacitor promise to overcome the drawbacks of (rechargeable) batteries, such as bulkiness, wear-out and toxicity. Because of low energy input and low storage capacity, battery-free devices operate *intermittently*; they are forced to remain inactive for most of the time charging their capacitor before being able to operate for a short time. While it is known how to deal with intermittency on a single device, the coordination and communication among groups of multiple battery-free devices remain largely unexplored. For the first time, the present thesis addresses this problem by proposing new methods and tools to investigate and overcome several fundamental challenges. Specifically, we make the following three main contributions:

- We present Shepherd, the first testbed with dedicated support for experiments with groups of distributed battery-free devices. Shepherd allows recording and replaying high-resolution voltage and current traces of real energy environments synchronously and at high rates across spatially distributed battery-free devices. It provides unprecedented visibility into energy environments across time and space, and faithfully reproduces those conditions for the systematic development and evaluation of distributed battery-free applications and services. We release Shepherd as an open-source tool, facilitating research into time synchronization, wireless networking, and other distributed algorithms for battery-free systems.

- We bootstrap battery-free wireless networks by presenting two new mechanisms that enable battery-free devices to discover each other quickly and efficiently. The first mechanism, Find, is a neighbor discovery protocol enabling devices to wake up synchronously despite a previous time offset by introducing random delays before becoming active. When waking up synchronously, neighboring devices can communicate and discover each other. At runtime, each device running Find dynamically adapts an optimized delay distribution to changes in its energy availability to maintain low discovery latency despite changes in energy availability. The second mechanism, Flync, is a hardware/software solution that phase-synchronizes solar energy harvesting devices to the powerline-induced flicker of state-of-the-art lamps; the proposed circuit draws only 5 µW of power. Using Find together with Flync, devices can implicitly align their activity phases to this external synchronization signal, dramatically increasing their chances to be active at the same time. Experiments with an open-source prototype built from off-the-shelf hardware components show that our techniques reduce the discovery latency by 4.3× (median) and 34.4× (99th percentile) compared with a baseline approach without waiting.
- Finally, we present Bonito, the first connection protocol for battery-free systems that enables reliable and efficient bidirectional communication between intermittently powered nodes. We collect and analyze real-world energy-harvesting traces from five diverse scenarios involving solar panels and piezoelectric harvesters, and find that the nodes' charging times approximately follow well-known distributions. Bonito learns a model of these distributions online and adapts the nodes' wake-up times so that sender and receiver are operational at the same time, enabling successful communication. Experiments with battery-free prototypes demonstrate that our design improves average throughput by 10–80× compared with the state of the art.

Acknowledgements

I would first like to thank my advisor Marco Zimmerling for providing the guidance, the resources and the freedom that facilitated the research presented in this work. Thanks for raising the right questions, for discussing countless ideas and for turning the most promising ones into successful projects with me. Your dedication and your aspiration for quality have substantially contributed to this thesis, and I am honored to become your first doctoral graduate.

I am grateful to Raja Jurdak for introducing me to the research field of energy harvesting and for hosting me in my second research group at CSIRO, Australia. The same applies to Brano Kusy whom I also thank for serving as the adjunct advisor for this dissertation. Special thanks to Sara Khalifa and Moid Sandhu for the exciting projects and the good times we shared. I am thankful to Fabian Mager and Carsten Herrmann, my fellow doctoral students at the Networked Embedded Systems Lab for providing a friendly and inspiring working environment. I have benefitted a lot from your input and our discussions. Thanks to our research engineer Ingmar Splitt for sharing your knowledge and for lifting Shepherd to new heights. Thanks also to Mikołaj Chwalisz for introducing me to testbeds and the Python universe. Thanks to our research assistants Justus Paulick and Friedrich Schmidt for your support in the lab and with our (literal) field experiments. I would also like to thank our administrative assistant Conny Okuma for her patience while guiding me through the paper jungle.

I owe my deepest gratitude to my parents and my sister for their ongoing encouragement and support for this project and beyond. Finally, I would like to thank my partner Sarah for her love and for accompanying me through the ups and downs of this journey.

Contents

List of Figures

List of Tables

1

Introduction

Cyber-physical Systems (CPS) deeply embed sensing, computing, storage, and control in the physical world. Adding networking capabilities to these systems enables humans and other devices to monitor and control processes remotely. Outside the academic world, the collective term internet of things (IoT) has emerged to describe such networked CPS. The IoT currently connects more than 12 billion devices for applications ranging from industrial control systems [90] over smart farming [41] to implantable biologgers [119]. Because the devices are tightly coupled to the physical world, they are often mobile, inaccessible or widely distributed in remote areas. Sensor networks, for example, monitor processes with high spatio-temporal resolutions by deploying tiny embedded computers equipped with one or more sensors in an environment of interest. In such scenarios, laying cables for power and communication is often unaffordable or impossible. Instead, devices rely on local energy sources, like batteries, and send the collected data over wireless networks. However, batteries have limited capacity and after running out, the corresponding device stops working.

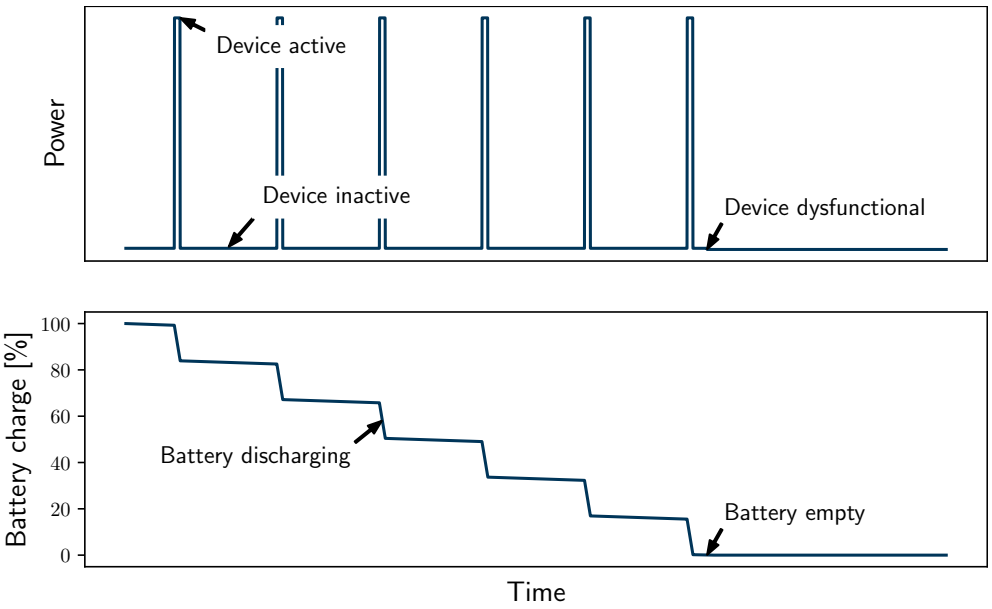


FIGURE 1.1: In order to prolong their lifetime, battery-powered IoT devices use *duty cycling*. They spend most of the time in a low-power sleep mode and only become active for a short time. Nonetheless, eventually the battery runs out and the device stops working.

Many IoT devices offer various levels of low-power operation where parts of the system are powered off to reduce the power consumption from tens of milliwatts when active to hundreds of nanowatts or less. To prolong their battery lifetime, devices use *duty cycling* where the system remains in a low-power mode for most of the time, conserving energy and only becomes active to read a sensor or communicate with the network [140]. Duty cycling is illustrated in Figure 1.1. The average duty cycle and exact wake-up times are set by the operator or the application to trade off performance with battery lifetime. Despite these efforts to conserve energy, batteries unavoidably run out, causing tons of hazardous waste and severe environmental problems [71]. Depending on the battery capacity and the power consumption of the device, this may happen after a few weeks or months, already [3]. Many applications on the other hand require much longer lifetimes. For instance, monitoring the restoration of former mining land with wireless sensor nodes is a decade-long endeavor [73]. Such long-term deployments would require frequent replacement of empty batteries,

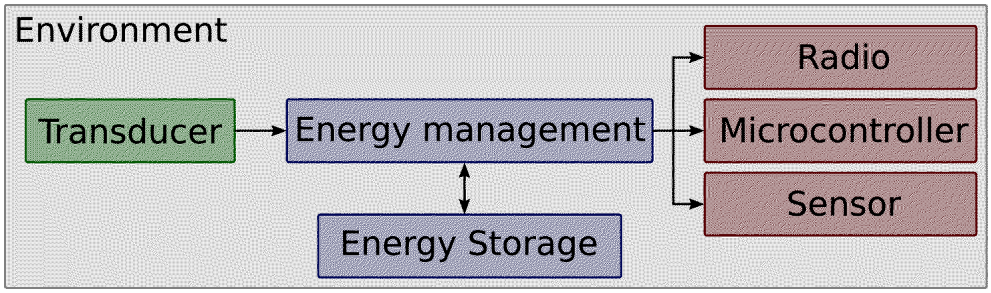


FIGURE 1.2: Architecture of an energy harvesting IoT device. The transducer extracts energy from the environment. The energy management supplies the system with the required voltage. Energy storage allows powering the system when the consumption exceeds the power available from the transducer.

but this is prohibitively expensive. For example, replacing the batteries of one single industrial IoT device can cost up to US$ 500 [5], much more than the price of the hardware. As a result, large-scale sensing applications with hundreds or thousands of individual battery-powered devices are infeasible. In other applications, like wildlife tracking with animal-borne devices [122], replacing batteries is not possible at all.

To operate *autonomously* and to reduce the environmental impact of disposable batteries, devices can instead use rechargeable batteries and replenish them by harvesting renewable energy from the environment. The ability for sustained operation without human intervention has facilitated some of the most influential applications of sensor networks, like ZebraNET which provided researchers with insights into migrational patterns of zebras [63]. After decades of research, energy harvesting based sensors are now also entering commercial contexts. For example, the mOOvement global positioning system (GPS) ear tag [97] uses solar energy to monitor the location, activity and health of cattle and sends the recorded data wirelessly to a base station, from where it is uploaded to a cloud for analysis. Relying on this unceasing source of energy, the tag can operate for years without battery replacement - a significant improvement over traditional battery-powered devices.

Figure 1.2 shows the architecture of an energy-harvesting device. The transducer extracts energy from the environment and converts it to

Harvesting technology	Application scenario	Power density
Solar cells	Outdoors at noon	$15\,\mathrm{mW\,cm^{-2}}$
Piezoelectric	Shoe inserts	$330\,\mu\mathrm{W\,cm^{-3}}$
Electromechanical	Small microwave oven	$116\,\mu\mathrm{W\,cm^{-3}}$
Thermoelectric	10° temperature gradient	$40\,\mu\mathrm{W\,cm^{-3}}$

TABLE 1.1: Common technologies for energy harvesting devices [105]. Due to its high energy density and wide availability, solar energy is the most used energy harvesting source.

electrical energy. Table 1.1 lists common examples of transducers, application scenarios and corresponding energy densities. Solar energy remains by far the most popular source due to its high energy density and wide availability. The power from a transducer varies and is often not sufficient to power the system at all times. For instance, when harvesting energy from the vibrations of a car, the intensity of vibrations changes with the road surface and the velocity of the car. To be able to operate when the instantaneous harvesting power is lower than the power consumption of the device, energy storage is required to accumulate surplus energy when the harvesting power is higher than the consumption. The energy management circuitry charges the energy storage and provides the microcontroller, radio and sensors with the required supply voltage.

With the advent of mobile consumer devices, rechargeable Lithium-based batteries have become cheap and widely available, and are the state-of-the-art energy storage element for commercial energy harvesting devices. However, they have various crucial disadvantages: The extraction of Lithium consumes large amounts of water and energy and can lead to the release of toxic chemicals into sensitive environments [136, 31]. Their cathodes contain cobalt that is often mined under catastrophic humanitarian and environmental conditions and fuels armed conflicts in the producing regions [10, 131]. Many other battery technologies have similar downsides. For example, all rechargeable batteries are subject to aging and must be replaced after a few hundred charging cycles [99]. They are also often the heaviest and largest of all components of an IoT device [55].

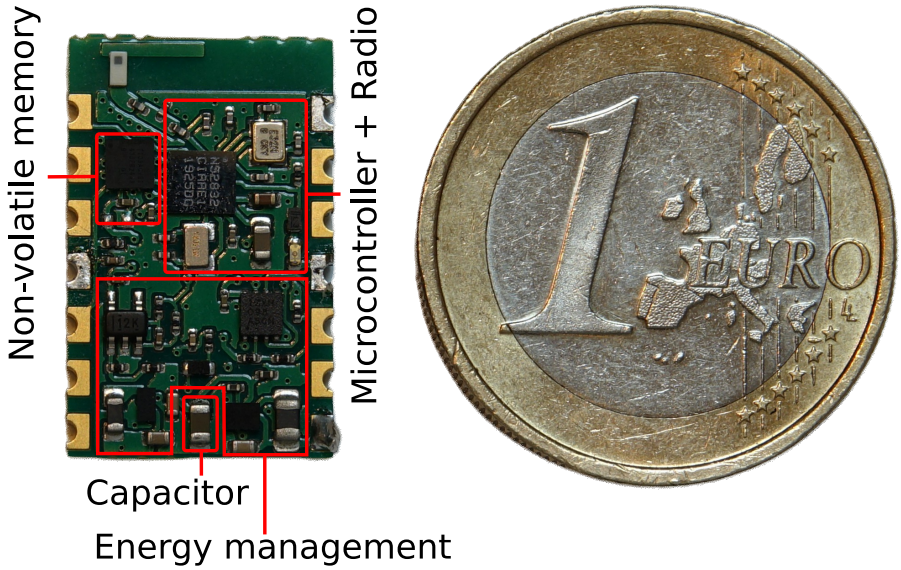


FIGURE 1.3: Our battery-free prototype is equipped with a capable MCU and a BLE radio, and uses a sustainable 0.5 mm^3 capacitor as energy storage.

1.1 Battery-free Devices

To overcome the drawbacks of rechargeable batteries, researchers started casting the vision of *battery-free* devices [112, 55]. Instead of storing harvested energy in rechargeable batteries, these devices are powered either directly from the harvester or use only tiny and eco-friendly capacitors as energy storage. Figure 1.3 shows a picture of our latest prototype battery-free device. The printed circuit board (PCB) measures only 2 cm^2 and integrates all necessary harvesting and energy management circuitry to charge and execute from the tiny 22 µF ceramic capacitor. The prototype is based on the popular nRF52832 microcontroller unit (MCU), which combines a 64 MHz ARM Cortex-M4F central processing unit (CPU) and a low-power bluetooth low energy (BLE) radio. The MCU is connected to a non-volatile memory chip that allows retaining application data across power failures (see Section 1.3).

In addition to reducing the environmental impact of existing applications, battery-free devices can also enable applications where batteries

are too expensive, too large or infeasible, such as sensing on live insects [58], in vivo [109], or in space [33]. However, without significant energy storage, the quality of service provided by a battery-free device crucially depends on the energy availability; it can only sensibly be used when its tasks coincide with the energy availability. In the simplest case, the energy availability itself is the sensing signal, known as energy harvesting based sensing [66]. One example is human activity recognition with a battery-free device powered by human motion [114]. In other scenarios, the information signal and the energy are from the same source, as could be the case in solar-powered vision systems [49, 35]. Finally, the utility and the energy source may simply correlate like when tracking diurnal animal species with solar-harvesting devices [125]. When neither of these conditions applies, the only alternative is to artificially inject energy into the environment to provide the sensor with energy when required. This approach, known as *wireless power transfer* [139], is fundamentally different from ambient energy harvesting, because the system can dynamically control the timing and extent to which the devices receive power.

1.2 Intermittency

Using rechargeable batteries, designers of energy-harvesting devices can dimension the harvester and energy storage capacity to achieve *energy-neutral operation* with a given duty cycle [18]. Alternatively, devices dynamically adapt their duty cycle such that the average energy consumption matches the average harvested energy [65, 45]. In both cases, surplus energy is accumulated in the energy storage to keep the device operational when energy is scarce. Figure 1.4 illustrates energy-neutral operation of a device with variable energy input operating with a constant duty cycle.

A battery-free device on the other hand cannot store sufficient energy in its small capacitor to compensate for low and fluctuating harvested

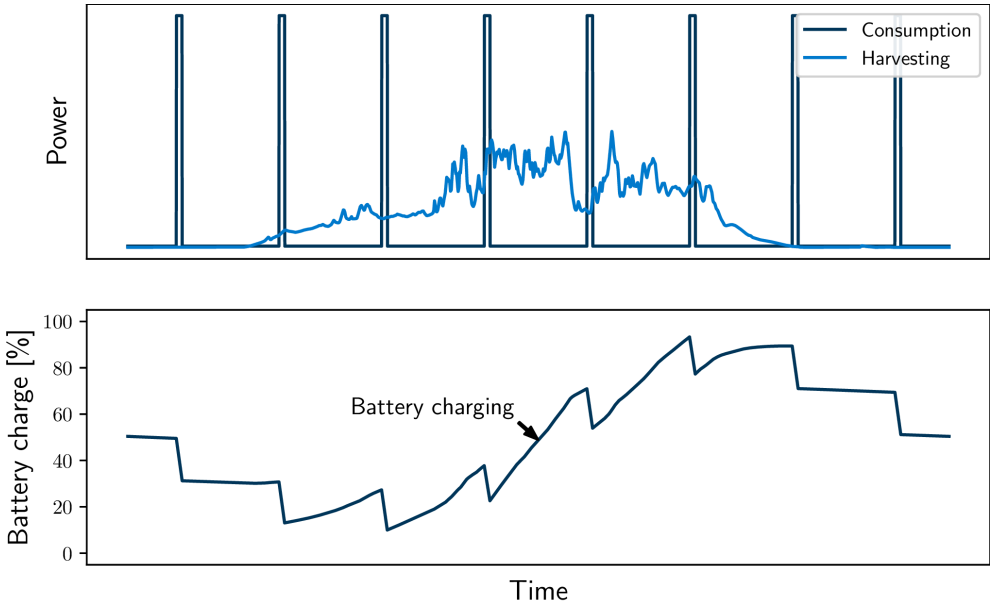


FIGURE 1.4: Energy-neutral devices recharge their battery using energy harvested from the environment. They adjust their average duty cycle in order to avoid depleting their battery.

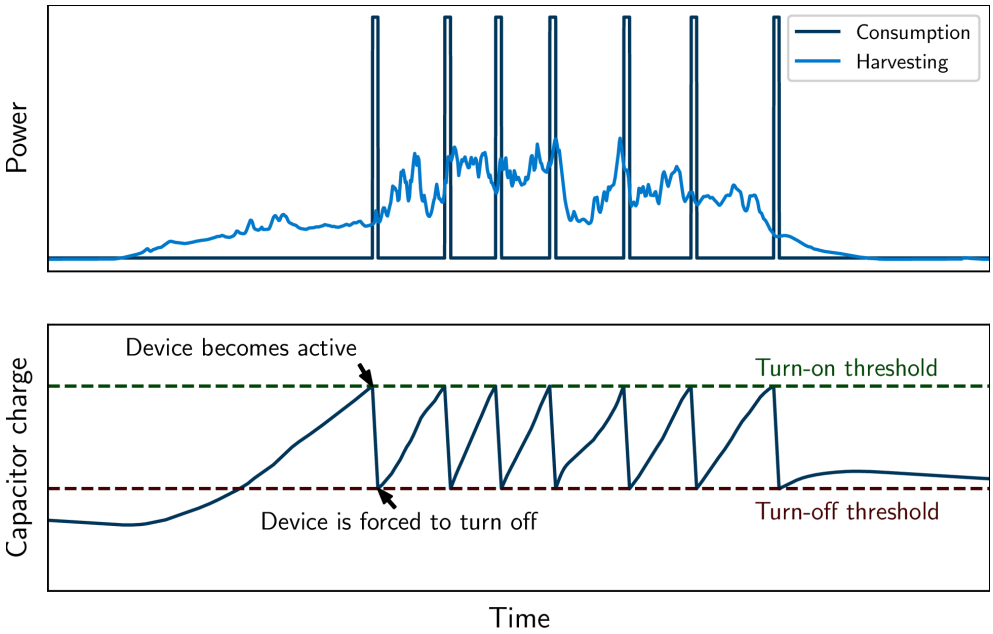


FIGURE 1.5: Battery-free devices operate *intermittently*. They are forced to remain inactive for most of the time and have limited control over their exact wake-up times.

energy. Instead, the device operates *intermittently*, as illustrated in Figure 1.5. While inactive, the device consumes little power in the order of hundreds of nanowatts and accumulates charge until the capacitor voltage reaches a turn-on threshold. Then, the device becomes active and starts to operate with a greatly increased power consumption. After executing for a short time, the capacitor voltage falls below the turn-off threshold and the battery-free device is *forced* to become inactive. The activity phases of a battery-free device are often short compared to the time it takes to recharge the capacitor. For example, when harvesting energy from indoor light, our prototype battery-free node shown in Figure 1.3 needs to stay off and recharge, on average, for hundreds of milliseconds before it can operate for at most one millisecond.

Intermittency is in stark contrast to conventional duty cycling and energy-neutral systems, where devices can become active *at any point in time* subject only to an average duty cycle constraint to avoid running out of energy. Various technological parameters influence the extent to which a device is subject to intermittency:

Energy harvester. Provisioning a system with an energy harvester that provides the power required to run the system at any point in time avoids intermittency, but has significant drawbacks in terms of size, weight and costs. For example, a device may consume only 10 µW on average, but 10 mW when active, requiring to over-provision the harvester by a factor of 1000. In many practical scenarios, the harvestable energy becomes zero for extended periods, for example, when harvesting energy from the vibrations of a car that stops and switches off the engine at a red light. A system without significant energy storage cannot become active during these times; It operates *intermittently*.

Energy storage. Despite efforts to formalize the design space of intermittently powered devices [60, 96], up to date, there exists no clear definition of the maximum capacity up to which a device operates intermittently rather than under severe time-varying duty cycle constraints. When the instantaneous harvesting power is too low to power

the system, the energy storage capacity must be high enough to support the largest atomic operation like, for example, sending a packet with the wireless radio. Increasing the energy storage beyond the minimum may allow executing multiple operations without turning off in between and thereby alleviate some challenges associated with intermittent operation [143]. But due to the low energy density of capacitors, a moderate increase in capacity comes with a relatively large increase in cost, size and weight of the overall device. Beyond these practical considerations, studying the lowest end of the storage capacity spectrum also raises exciting research questions and substantially pushes the boundaries of what is possible.

Energy consumption. While using lower power hardware, which, for example, reduces the average power consumption in sleep mode or for time-keeping can reduce the charging time, it does not generally avoid intermittency. This would require lowering also the active power consumption of the system below the harvested power. Despite significant progress over the past decades, transmitting and receiving remain among the most power-intensive tasks of an IoT device. Low power wireless radios enable communication at data rates up to $2\,\mathrm{Mbit\,s^{-1}}$ or ranges of multiple kilometers, but consume between 10 mA and 100 mA. The active current draw of the CPU instead is often less than 1 mA. By omitting the power-hungry oscillator on the low-power device and instead modulating the radio frequency (RF) signal from a carrier generator, *backscatter* can drastically reduce the power consumption for transmission at the cost of reduced throughput and range [83, 59, 92]. *Wake-up receivers* on the other hand use an ultra low-power, low sensitivity radio to listen for incoming transmissions and only activate a more powerful receiver to decode the data [43, 102]. This greatly reduces the power for idle listening, but still requires sender and receiver to activate their high power radios *at the same time* in order to be able to exchange data. These alternative physical layers help to reduce the extent to which intermittency affects a battery-free device by trading off throughput, range or autonomy for lower power consumption and, as a result, increased active time.

In summary, when designing a battery-free device for minimum cost, size and weight, while maintaining practical capabilities, its active power consumption often exceeds the harvested power. When the device also does not have enough storage capacity to account for the discrepancy, intermittency arises as a key challenge.

1.3 Intermittent Computing

The relevance of this challenge is reflected by the growing number of works in the domain of intermittent computing. The majority of recent literature assumes that a battery-free device powers off unpredictably when reaching the turn-off threshold and studies the resulting challenges. For example, while powered off, the content of volatile memory and registers is not retained. To ensure forward progress of tasks across power failures, the application state must be checkpointed to non-volatile memory [91]. When recovering after a power failure, the software has to establish memory consistency [88]. Similarly, reliable timekeeping [56, 27] and retention of the state of peripherals [16, 89] are challenging across power failures. A growing number of programming abstractions [91, 69] and intermittent runtimes [141] aim to abstract the challenges of programming devices operating under these conditions.

A less well-studied approach to dealing with intermittency is to monitor the supply voltage and signal an impending power loss to the microcontroller. Upon this signal, the microcontroller can checkpoint any volatile application state to non-volatile memory and gracefully transition to a deep sleep mode instead of powering off completely [9, 61]. The associated energy overhead for monitoring the supply is often outweighed by the savings from not having to checkpoint preventatively and by retaining peripherals and clocks across the off-periods.

Using these methods and supporting tools for debugging [21] and experimentation [52, 53, 121], this research has culminated in a number of real-world applications of battery-free technology [79, 3, 28].

1.4 Battery-free Networks

While these applications demonstrate impressive progress in the domain of intermittent computing, today's potential scenarios for battery-free systems are limited to individual devices. Shifting the focus to *multiple* rather than individual battery-free devices would not only enable exciting new applications (*e.g.*, swarms of nanosatellites [98]) but may also offer a new perspective on the reliability issue; while an individual device may not be able to deliver sufficient quality of service at all times, a group of battery-free devices might. Additionally, many practical applications rely on direct device-to-device communication for a variety of applications and services including time synchronization [77], contact tracing [103], coordination of sampling [93] or multi-hop communication, where messages are relayed between devices to increase coverage, reliability and efficiency [76]. Despite being frequently mentioned as one of the greatest barriers to the adoption of battery-free devices [55, 142, 82], there has been little work on coordination and communication among groups of battery-free devices. This is mainly due to three key challenges:

1. Without significant energy storage, the behavior of battery-free devices is directly tied to the instantaneous energy availability. While the energy availability and the behavior of a single battery-free device have been studied extensively, there is little understanding of the spatio-temporal patterns of energy harvesting and its implications on operating groups of distributed battery-free devices.

2. To communicate successfully, two devices must be active *at the same time*. Because the short activity phases of different nodes are generally not aligned, it takes a long time until nodes wake up simultaneously and can communicate. After an extended period without energy input, battery-free devices lose track of time and need to resynchronize.

3. At every encounter, devices can only exchange a small amount of data before being forced to power off again. Because of fluctuating and different charging times between the nodes, they are out of sync at the following wake-up. Having to resynchronize before the exchange of every small chunk of data renders communication extremely inefficient.

1.5 Contribution

This thesis addresses these challenges and enables battery-free devices to efficiently exchange data, forming a basis for large-scale networked applications. Specifically, we make the following contributions, which are also illustrated in Figure 1.6.

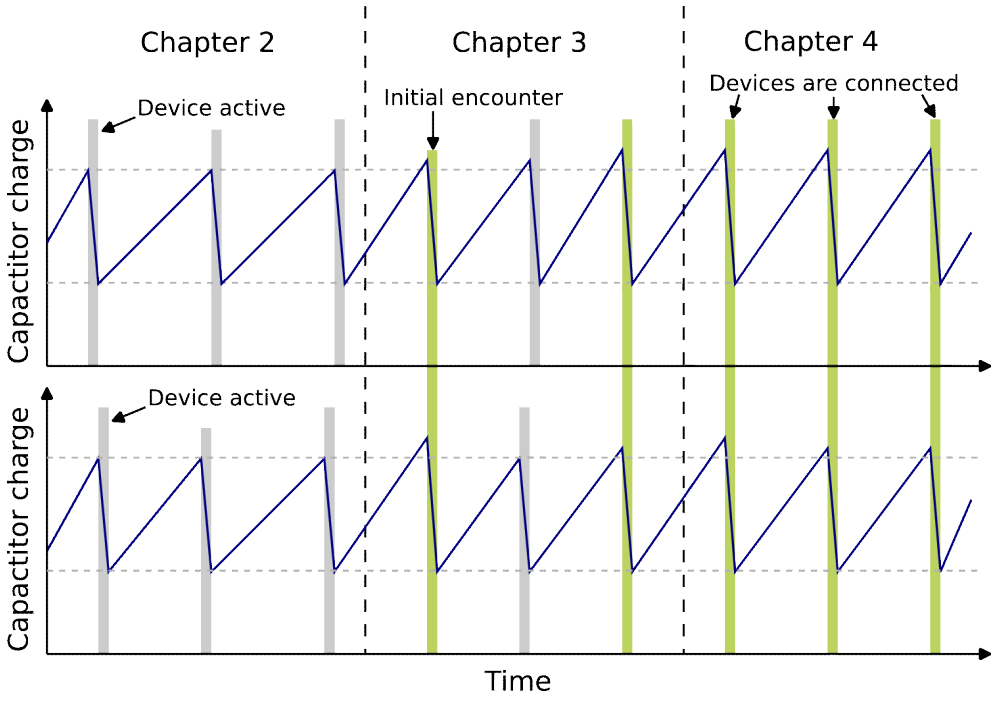


FIGURE 1.6: Illustration of our contributions. Chapter 2 presents a tool for experiments with groups of battery-free devices. Chapter 3 enables battery-free devices to encounter each other for the first time. Chapter 4 proposes a protocol for efficient data exchange over long-lasting connections.

A testbed for battery-free devices (Chapter 2). Existing tools for experiments with energy harvesting devices consider single devices only [52, 53, 121]. Distributed testbeds for experiments with multiple battery-powered devices on the other hand have no notion of energy harvesting [81, 118]. We tackle challenge 1 and enable experiment-driven research into the challenges and opportunities of operating groups of battery-free devices by presenting an open-source tool that can record and reproduce spatio-temporal characteristics of real energy environments.

Neighbor discovery (Chapter 3). To establish communication, two devices need to discover each other during a first encounter, known as neighbor discovery. Existing neighbor discovery protocols require devices to be able to wake up at any point in time [37, 64, 8, 67] and hence do not apply to battery-free devices. Addressing challenge 2, we propose a method of how battery-free devices can arrange an initial encounter despite their short and interleaved activity phases. By exploiting power line flicker in commonly used lamps as a synchronization signal, we further reduce the time to the first encounter for indoor light harvesting devices.

Efficient device-to-device communication (Chapter 4). Various MAC protocols have been proposed for battery-powered [140, 32] and energy neutral wireless sensor networks [39, 2]. These protocols are based on fixed communication schedules and excessive sampling of the wireless channel, both of which do not apply to battery-free devices. In response to challenge 3, we postulate that devices can exploit the initial encounter to establish a connection that enables them to exchange data at every wake-up, dramatically increasing throughput. We analyze the statistical properties of real-world harvesting traces from diverse scenarios and devise a method of how devices can online learn models of their wake-up times to select connection parameters that work reliably and efficiently.

2

Shepherd: A portable testbed for the batteryless IoT

Prelude. This chapter covers the paper with the same title co-authored by Mikołaj Chwalisz and Marco Zimmerling that I presented at the 17th ACM Conference on Embedded Networked Sensor Systems in 2019 [44]. Motivated by the need to understand the challenges and opportunities of operating groups of battery-free sensor nodes, the paper presents Shepherd, a testbed for the battery-free IoT. Shepherd allows recording synchronized energy traces with a resolution of 3 µA and 50 µV at a rate of 100 kHz, and faithfully replaying these traces to any number of sensor nodes to study their behavior. Shepherd is released as an open-source tool for the community, facilitating research into time synchronization, wireless networking, and other distributed algorithms for battery-free systems.

2.1 Introduction

As the IoT grows to trillions of devices [124], sustainability and reliability of this computing infrastructure become matters of utmost importance. One possible path to sustainability is the adoption of *batteryless* devices that buffer harvested energy in a capacitor, and execute when there is energy available in the capacitor. Batteryless devices promise to overcome the drawbacks of (rechargeable) batteries, such as bulkiness, wear-out, toxicity, uncertain remaining charge, etc. The limited energy capacity of capacitors, however, requires intermittently executing the software, which may harm reliability [60] despite checkpointing techniques [88, 13], platform support [23, 54], and dedicated programming models [22].

Focusing on *multiple* rather than individual batteryless devices not only enables exciting new applications (for example, swarms of nanosatellites [98]), but may also offer a new perspective on the reliability issue. Fault tolerance in conventional (*i.e.*, continuously-powered) distributed systems often relies on exploiting redundancy [117]: If the level of fault tolerance provided by a single server is unacceptable, then multiple servers executing replicas must be used. From this experience, we may ask research questions like the following: Is it possible to operate a distributed collection of batteryless devices so that the group is more reliable than each device alone? How would efficient and reliable programming models, wireless communication protocols, and runtimes for collections of batteryless devices look like? Can the cooperation of multiple distributed batteryless devices bring about benefits in terms of overall efficiency and effectiveness, similar to cooperation in multi-agent systems [108]?

Motivating example. To illustrate a possible way to investigate those questions, Figure 2.1 shows real traces of harvesting current we synchronously recorded at three devices with piezo-electric elements. The devices are mounted at different locations of a car, and harvest energy from the car's vibrations as it drives through a suburban area. This setting is akin to, for example, devices mounted on a large machine in

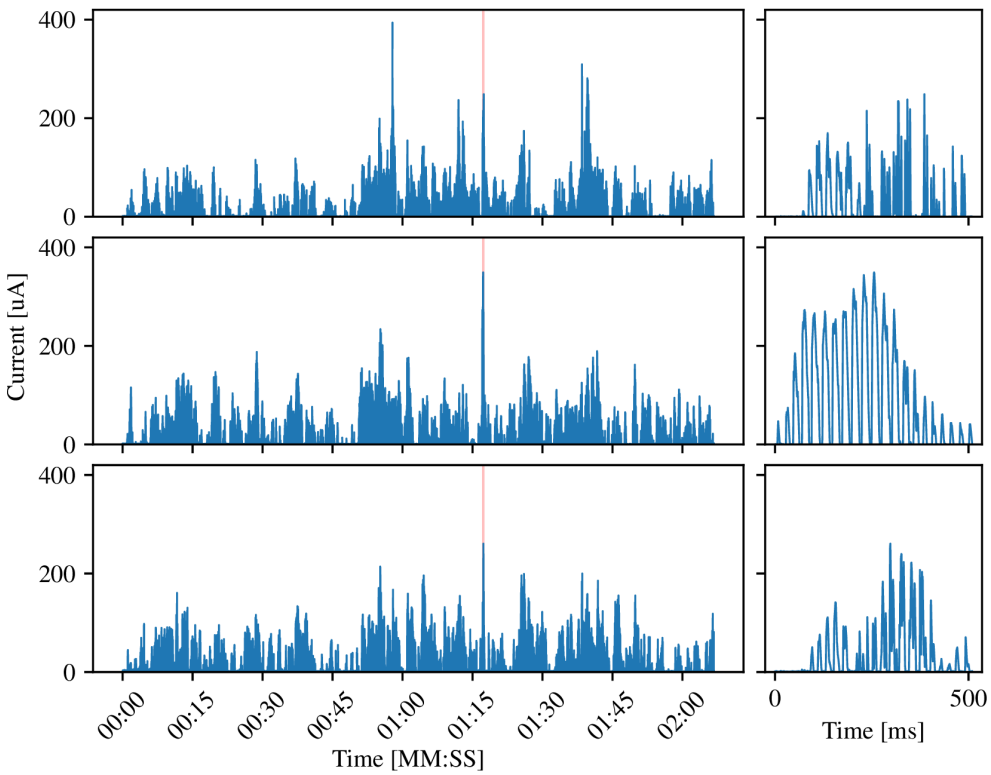


FIGURE 2.1: Synchronized traces of kinetic harvesting current, recorded with three Shepherd nodes mounted at different locations of a car. Shepherd can replay such harvesting current and voltage traces to distributed batteryless devices.

a factory (*e.g.*, for predictive maintenance).

Considering any of the traces individually, we observe that the instantaneous energy availability of a device varies significantly and unpredictably over time. This is indeed one of the key challenges in the design of useful, reliable batteryless applications [85, 55]: In the absence of a large energy storage (*e.g.*, a battery), fine-grained variations in energy availability cannot be abstracted away. Rather, the matter of *when* energy is available may be critical to the correct functioning of a batteryless system, which is comparable to the crucial role of time for the correctness of a cyber-physical system [34].

Looking at all three plots to the left together, we instead notice a similar macroscopic shape of the traces. The spatial proximity of the devices and systematic properties of the energy-harvesting environment

result in a positive correlation between the time-varying energy availabilities: If one device has energy, then the other two tend to have energy as well. This also holds in other environments, such as indoor solar energy harvesting (see Section 2.7). Although there are non-negligible differences in energy availability across the devices, as visible in the zoomed-in plots to the right, exploring ways to exploit such correlations (*e.g.*, for synchronizing and networking batteryless devices) may eventually provide answers to the challenging research questions posed above and elsewhere [20, 85, 55].

Problem. Unfortunately, the research community lacks a tool that enables such scientific endeavors. An appropiate tool needs to synchronously *record* the rapidly changing energy conditions at different points in space. Even having such traces, it is hard to accurately model and predict the performance and behavior of a real batteryless system because of the complex behavior of circuits exposed to an intermittent power supply. To develop and compare novel designs, it is thus necessary to experiment under the constraints of time-varying energy availability by faithfully *reproducing* energy environments from recorded traces or spatio-temporal models.

Recording and replaying harvested energy is hard, and few solutions exist for individual devices. For example, using a source measure unit (SMU) one can profile and emulate a single harvester. However, such equipment is expensive (*i.e.*, thousands of USD per unit), while the sampling speed may not be sufficient to capture rapidly changing energy availability. To address this problem, Ekho [53] uses custom-designed, affordable hardware to record and emulate an energy source with limited accuracy and resolution.

Testbeds support synchronous recording of current draw [81] or energy consumption [118] at multiple distributed devices. Although this is one piece of the puzzle, it is not possible to profile the complex behavior of an energy-harvesting system with existing testbeds by reproducing an energy environment. The FlockLab testbed offers the possibility to vary the supply voltage, emulating a discharging battery [81].

However, as described in Section 2.2, an energy harvester has a characteristic IV curve that determines the current flowing at a particular voltage. Emulating this behavior requires an inherently different approach than what is found in existing testbeds.

Contribution. This paper presents Shepherd, a portable testbed for the batteryless IoT that fills this gap. Shepherd's main novelty is the combined capability of accurately recording and replaying high-resolution voltage and current traces synchronously and at high rates across spatially distributed batteryless devices. With this, Shepherd provides unprecedented visibility into energy environments across time and space (see Figure 2.1), and faithfully reproduces those real-world conditions for the systematic development and evaluation of distributed batteryless applications and services.

Shepherd is a complement of hardware and software. Its modular hardware architecture rests upon a powerful observer platform with a custom-designed analog frontend, deep local storage, and real-time processing capabilities. Different harvesting sources, energy buffers, and sensor nodes can be attached to an observer using well-defined interfaces. Shepherd's software architecture tackles the challenges of tight synchronization among observers that may be kilometers apart (*e.g.*, batteryless LPWAN), and by providing reliable, high-throughput data transfer subject to timing constraints.

Beyond record and replay, Shepherd's harvesting traces may be analyzed offline or fed into simulators. Conversely, Shepherd can also replay traces generated in software or recorded with other tools, such as Ekho [53], RocketLogger [121], or a SMU. Shepherd is affordable (about 200 USD per observer) and portable (supporting mobile outdoor scenarios) as it does not rely on heavy infrastructure, yet it offers all amenities of existing testbeds, including GPIO tracing, serial logging, and remote programming.

To summarize our main contributions:

- We identify a workflow for the development and evaluation of solutions for distributed batteryless devices, and derive the key

requirements of a testbed that supports this workflow.

- We design and build Shepherd, the first testbed that meets those requirements. We open-source Shepherd's hardware/ software stack together with extensive documentation and tools that aid users during installation and experimentation.[1]
- We demonstrate Shepherd's utility and capabilities using a real-world distributed batteryless application scenario.
- We evaluate Shepherd's performance and show, for example, that it records traces with a resolution of 3 µA/50 µV at a rate of up to 100 kHz, it replays traces with a mean error below 0.1 %, while ensuring a synchronization accuracy of 2.4 µs or better.

2.2 Background

This section provides some background on the device and energy-harvesting architecture of batteryless systems.

2.2.1 Batteryless Device Architecture

In its simplest form, a batteryless device consists of a harvester and a sensor node. The *harvester* is a transducer that converts some form of ambient energy, such as solar radiation or movement, into electrical energy. The *sensor node* operates from the energy extracted by the harvester, and typically includes a MCU, sensors, volatile and non-volatile memory, and a wireless radio.

In this configuration, the harvester must deliver the minimum voltage and power required to operate the sensor node (*e.g.*, about 1.8 V to operate the MCU and about 10 mW to send a packet over an active IEEE 802.15.4 radio). Adding an *energy buffer*, usually a fixed-size capacitor, allows to decouple the node operation from the instantaneous energy availability. While the node is inactive, energy accumulates in the buffer. When the energy in the buffer reaches a threshold, the node

[1] https://shepherd.nes-lab.org

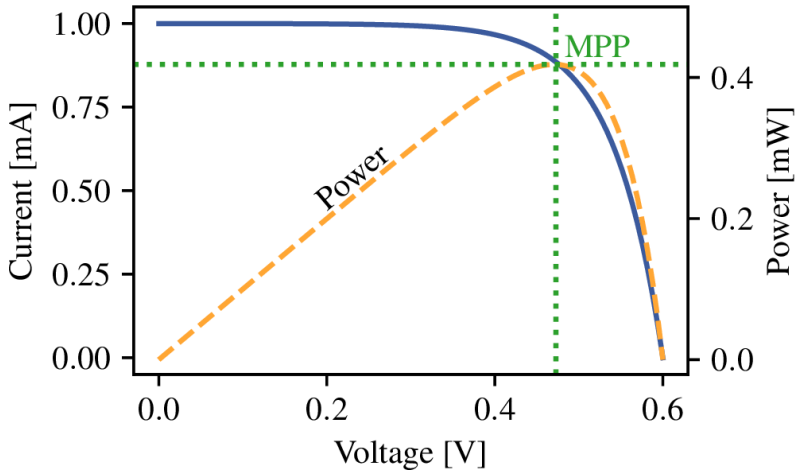


FIGURE 2.2: Characteristic IV curve of a solar cell. The voltage (V) on its output depends on the current (I) that is drawn and vice versa. The maximum power point (MPP) is the operating point where the extracted power reaches its maximum.

operates and consumes the buffered energy. This way, the node can operate although the voltage or power from the harvester is momentarily insufficient.

2.2.2 Characteristics of Harvesting Source

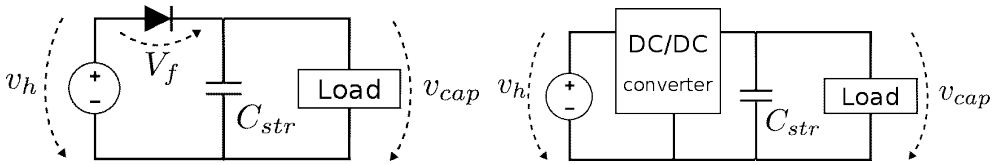
An ideal voltage source provides unlimited current. Instead, a real harvesting source has a distinctive IV characteristic: The voltage (V) on its output depends on the current (I) that is drawn and vice versa. The voltage together with the corresponding current determine the harvester's operating point. Figure 2.2 shows the characteristic *IV curve* (solid line) of a solar cell, illustrating that the extracted *power* (dashed line) crucially depends on the operating point. The operating point where the extracted power reaches its maximum is called *maximum power point (MPP)*. This has important implications for the extraction of energy from the harvesting source.

2.2.3 Energy-harvesting Architecture

There exist two fundamental approaches to extract energy from a harvester, converter-less and converter-based, each with their own strengths that can be exploited for different batteryless applications.

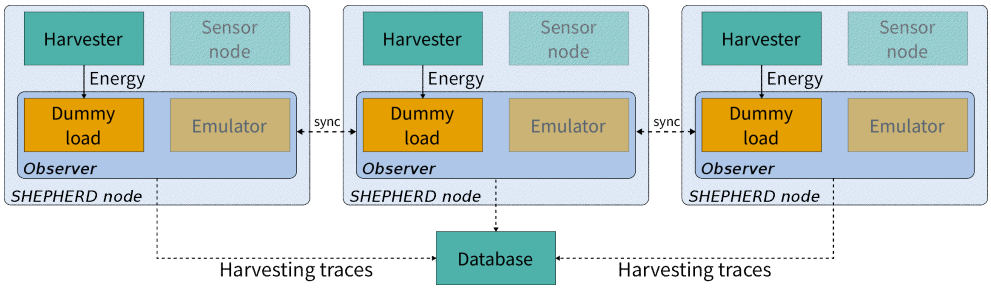
Converter-less. As shown in Figure 2.3a, a *converter-less* architecture consists of the energy-harvesting source, a diode, a capacitor, and the load. The operating point of the harvester depends on the state of charge of the capacitor as the harvesting voltage v_h is the sum of the capacitor voltage v_{cap} and the diode drop V_f. This prevents effective energy extraction as the harvester's operating point can be far off its maximum power point. Moreover, the harvester must be carefully selected and dimensioned to ensure minimum voltage and power conditions. Otherwise, the sensor node may never be able to operate, because the harvester delivers no current at a high enough voltage. For example, a typical solar cell delivers current only up to a voltage of about 600 mV, whereas a typical MCU needs at least 1.8 V to operate. Nevertheless, a converter-less approach requires only a minimum number of components and is thus highly cost-efficient, robust, and allows for extremely small form factors.

Converter-based. As shown in Figure 2.3b, a *converter-based* architecture uses a DC/DC converter in order to operate the harvester at an operating point different from the load: The harvesting voltage v_h can be set independently of the capacitor voltage v_{cap}. Thus, with knowledge about the IV characteristic of the source, the system can optimize power yield by dynamically adapting the operating point of the harvester, which is known as *maximum power point tracking*. This allows to efficiently harvest energy from a variety of low-voltage sources independent of the state of charge of the capacitor. Drawbacks of this approach include increased complexity, size, and cost because adding a DC/DC converter involves adding an integrated circuit and a handful of passive components.

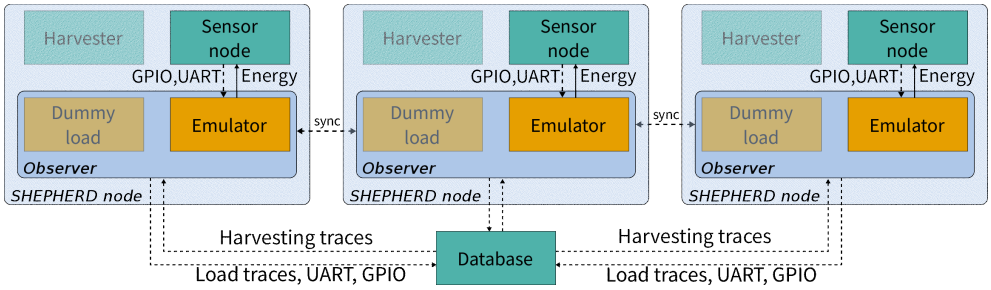


(A) Converter-less. (B) Converter-based.

FIGURE 2.3: Two fundamental energy-harvesting architectures.



(A) Recording.



(B) Emulation.

FIGURE 2.4: Essential components of Shepherd and their interactions during Shepherd's two main modes of operation.

2.3 Requirements and Overview

Shepherd is the first testbed for distributed batteryless systems. This section outlines the key requirements for such a testbed and provides an overview of how Shepherd addresses them.

2.3.1 A Typical Workflow

We envision the following typical workflow for the development and evaluation of distributed batteryless systems. A number of testbed nodes equipped with the desired energy-harvesting technology are deployed in the energy environment of interest. The testbed records the harvested energy at each node for a user-defined period of time. The user retrieves the data and analyzes them to gain an understanding of the characteristics of the recorded energy environment. With the help of the testbed, the user can then develop, test, and validate ideas involving (one or) multiple batteryless devices by repeatedly replaying the recorded energy traces to the device(s). This enables repeatable, experiment-driven research into open problems such as time synchronization, wireless networking, or distributed sensing and actuation using collections of batteryless devices [85, 54, 20]. A solution can then be validated by deploying it to the testbed in the target RF environment and analyzing the behavior and performance of the system. Replaying the same energy conditions allows to rigorously compare different solutions.

2.3.2 Key Requirements

From this envisioned workflow we derive the following key requirements for a useful testbed for distributed batteryless systems.

High accuracy and resolution. Harvested energy must be recorded and replayed accurately and precisely to be able to draw meaningful conclusions. Typical voltages of harvesting transducers like solar cells or piezo-electric elements range from hundreds of mV to a few V. Harvesting currents have an even wider range, typically from µA to tens

of mA. Voltage and current draw of a common sensor node are in a similar range. From these figures, we can derive our first two requirements: Current should be recorded with a resolution of 1 µA within a range up to 50 mA, and voltage should be recorded with a resolution of 1 mV within a range up to 3 V.

High sampling rate. Current and voltage need to be sampled at a rate high enough to capture the fine-grained characteristics of a harvesting source. For example, we found that a solar cell changes its voltage within tens of µs in response to a light being switched on. A sampling rate of at least 100 kHz is needed to capture such rapid changes in energy-harvesting conditions.

Time synchronization. To record and replay the energy environment and behavior of a batteryless network, the testbed nodes need to be tightly time-synchronized. In particular, the synchronization error must be significantly less than the sampling interval to unambiguously map samples from different nodes on a common timeline. We therefore target a synchronization accuracy of 1 µs.

Debugging facilities. Next to remote programming, the testbed should offer state-of-the-art debugging facilities such as synchronized tracing of GPIO pins and serial logging (*e.g.*, via `printfs`).

Portability, affordability, and customizability. A testbed for distributed batteryless devices must be exposed to different energy environments with unique characteristics and requirements. Thus, unlike conventional testbeds that are installed at a fixed location, a portable testbed is needed that users can afford to build and easily set up in various locations, which poses strict limitations on infrastructure and costs. This includes the ability to support new harvesting modalities and node platforms with minimal effort.

2.3.3 Shepherd Overview

To meet the above requirements, Shepherd consists a network of *Shepherd nodes* that are synchronized and operate in two modes:

- During *recording* (see Figure 2.4a), energy flows from the harvester to a dummy load, which is part of a powerful *observer* platform. The observer measures current and voltage, and timestamps the data with respect to the testbed-wide timeline. The timestamped data are first buffered locally on the observers, and then shipped to a remote database.
- During *emulation* (see Figure 2.4b), data are sent from the database to the Shepherd nodes from where they are fed into a harvesting emulator that outputs the corresponding voltage and current to an attached node. The data can be from previous recordings with Shepherd or some other tool, or generated using, for example, a spatio-temporal model of an energy environment. While replaying, the observer also monitors the sensor node's power draw, samples the GPIO pins, and records any serial messages from the node.

In the following two sections, we detail Shepherd's hardware and software architecture. Section 2.6 describes how users interact with Shepherd. Section 2.7 illustrates the capabilities and utility of Shepherd based on a real-world use case, while Section 2.8 systematically evaluates the performance characteristics of Shepherd's hardware/software stack using a series of controlled experiments.

2.4 Shepherd Hardware

As shown in Figs. 2.5 and 2.6, a *Shepherd node* consists of an *observer* and three *capelets*, and measures 90 mm×55 mm×40 mm. The *harvesting capelet* hosts the harvesting transducer and all components required to operate it; the *storage capelet* hosts the energy buffer, usually a capacitor with the desired size; and the *target capelet* hosts the sensor node. The three capelets are connected to the observer through well-defined interfaces, which makes for a modular design that users can easily customize depending on their needs in terms of harvesting modality, energy storage, and sensor node platform.

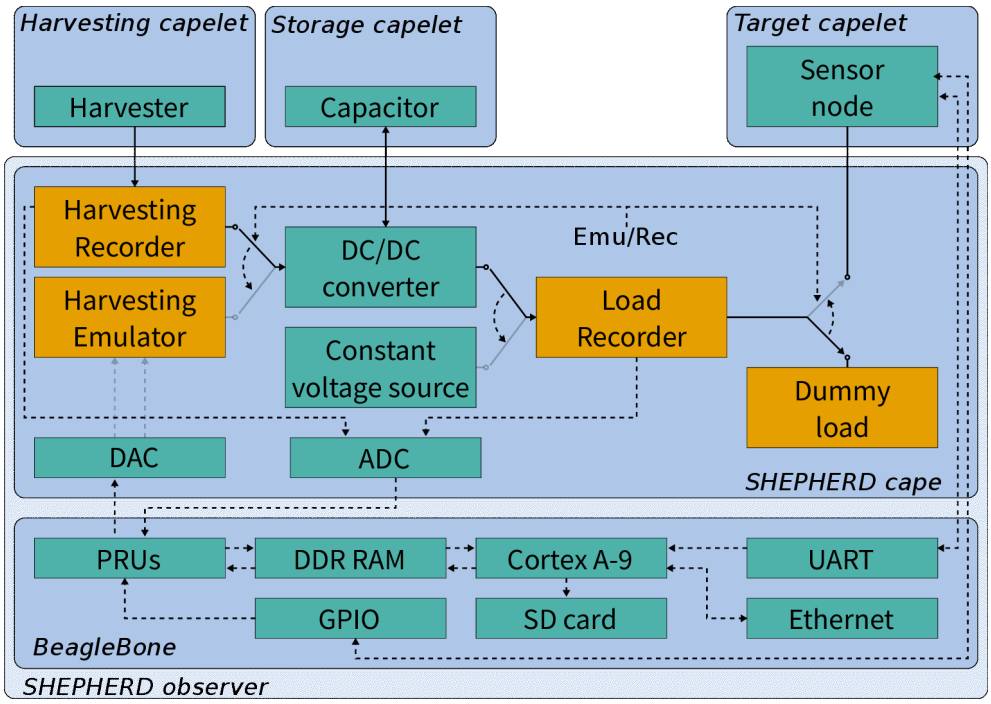


FIGURE 2.5: A Shepherd node consists of a BeagleBone SBC, the custom Shepherd cape, and three attached capelets.

The observer includes a custom-designed analog frontend, the *Shepherd cape*, and a *BeagleBone* single-board computer (SBC). Multiple observers connect via their BeagleBones' Ethernet ports with each other and to a host that stores the data and runs a tool we provide for orchestrating a collection of distributed Shepherd nodes. The Shepherd cape hosts all components and circuitry required for the recording and replaying of energy-harvesting traces.

2.4.1 BeagleBone

The BeagleBone is responsible for time synchronization, hardware interfacing, and data processing. We base our design on this platform as it is a mature single-board computer with superb software support and a living community. Two features make the BeagleBone particularly suitable for our needs compared to similar platforms.

First, the Ethernet controller of the BeagleBone's system-on-chip supports timestamping of Ethernet packets. As described in Section 2.5.2,

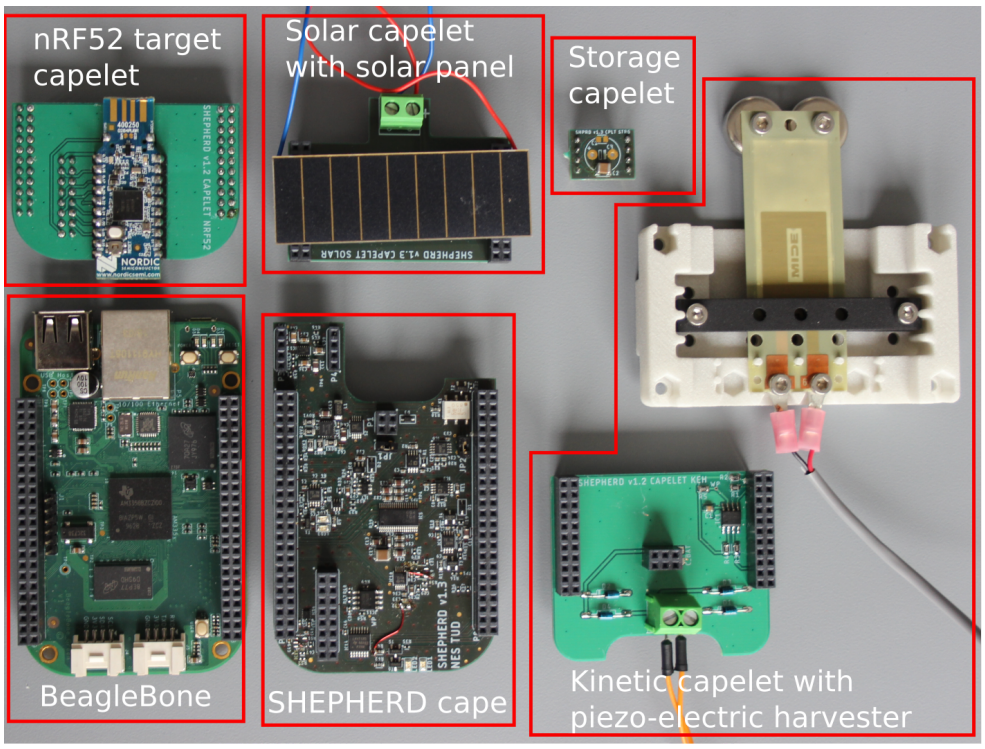


FIGURE 2.6: A Shepherd node is 90 mm×55 mm×40 mm in size, including the harvesting, storage, and target capelets.

we use this feature to tightly time-synchronize a collection of distributed Shepherd nodes using precision time protocol (PTP).

Second, the programmable real-time unit sub-system (PRUSS) of the BeagleBone's system-on-chip includes two deterministic RISC cores, the programmable real-time units (PRUs), which we dedicate to time-critical tasks, such as interacting with the analog-to-digital converter (ADC) and the digital-to-analog converter (DAC) on the analog frontend, as illustrated in Figure 2.5. Less critical tasks, such as storage and networking, are instead handled by the high-throughput ARM Cortex-A9. The PRUs and the ARM core can exchange data and control signals through shared memory and the system bus, which we use to implement a bidirectional communication protocol (see Section 2.5.1). The PRUs also have direct, low-latency access to some peripherals including the GPIOs. This is essential to achieve a GPIO sampling latency in the low µs range.

Using expansion headers, the analog frontend is stacked onto the BeagleBone as a cape (hence the name Shepherd cape), which in turn serves as the base board for the capelets discussed next.

2.4.2 Capelets

Shepherd supports different harvesting sources, energy buffers, and node platforms. To this end, we physically separate the Shepherd cape (*i.e.*, the analog frontend) from these components by introducing harvesting, storage, and target capelets, making it easy to exchange them (modularity) without affecting the behavior of the Shepherd cape or the BeagleBone (composability).

Target capelets. Target capelets are similar to adapter boards in Flocklab [81]: They provide a hardware interface that allows connecting a specific sensor node to Shepherd. It essentially connects the node to the capacitor-buffered output voltage of Shepherd's DC/DC converter. The well-defined connector includes all signals required to control and monitor a wide spectrum of sensor nodes through Shepherd. For example, serial wire debug (SWD) signals allow remote programming and debugging of many modern ARM-based nodes. universal asynchronous receiver transmitter (UART) pass-through allows programming of targets with a serial bootloader (*e.g.*, TelosB) in addition to serial logging and injection of commands. Finally, four general purpose input/output (GPIO) lines facilitate high-resolution monitoring of logical program states.

We provide a *nRF52840 capelet* as reference implementation of a target capelet. It interfaces the increasingly popular off-the-shelf nRF52840 dongle from Nordic Semiconductor with Shepherd. This target capelet merely serves as an adapter to connect the pins of the dongle to the 16-pin target connector on the Shepherd cape.

Harvesting capelets. Harvesting capelets provide Shepherd nodes with the energy-harvesting source of choice, such as a solar panel or piezoelectric element. Harvesting capelets may have a small flash memory

that can be used to store an ID and specific parameters for the mounted type of harvester. We provide two harvesting capelets.

The *solar capelet* allows to mount a solar panel to a screw terminal, directly connecting this DC source to Shepherd. Using a voltage divider, the operating point of the maximum power point tracker of the DC/DC converter (see Section 2.4.3) is set to 80 % of the open-circuit voltage, which is typical for solar energy harvesting.

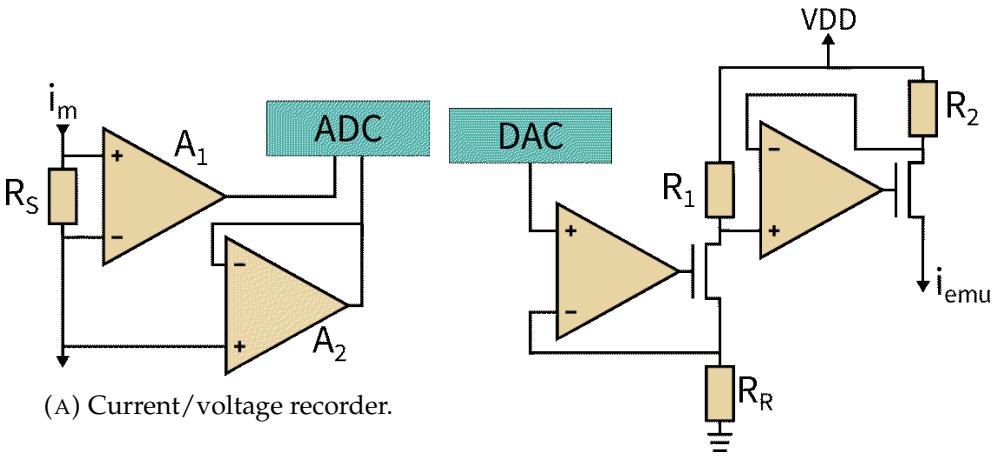
The *kinetic energy harvesting (KEH) capelet* is a more complex example of a harvesting capelet. The AC voltage from the piezo-electric element connected to the screw terminal is rectified with a full-bridge rectifier before connecting it to Shepherd's DC/DC converter. In contrast to solar energy harvesting, the optimal operating voltage of a piezo-electric harvester is often not derived from the open-circuit voltage, but set to a fixed value. For this reason, the KEH capelet has a flash memory that is used to inform Shepherd about the type of capelet and the desired operating voltage.

Storage capelets. Storage capelets host a capacitor as energy buffer that is directly connected to the DC/DC converter of the Shepherd cape. We provide a storage capelet with a 150 µF ceramic capacitor.

2.4.3 Analog Frontend: The Shepherd Cape

During recording the observer measures the power extracted from the harvester, and during emulation it replays (this or some other) power trace to the sensor node while measuring the node's consumption. The custom-designed analog frontend hosts the circuitry and components required to support these two modes of operation, node programming and debugging, as well as ensuring a defined initial filling level of the energy buffer before an experiment starts. Our design also caters for use cases where Shepherd serves as a portable testbed for non-harvesting sensor nodes.

DC/DC converter. Existing work on recording and emulating harvesting traces considers converter-less systems [53]. Because the harvested



(A) Current/voltage recorder.

(B) Current emulator.

FIGURE 2.7: High-level schematics of recorder and emulator.

energy depends on the load voltage in this case, as explained in Section 2.2.3, the whole IV characteristic has to be sampled for every point in time by rapidly altering the load voltage.

With Shepherd's converter-based approach, the harvester can be operated at a defined point, independent of load behavior and capacitor voltage. Therefore, it is sufficient to sample current and voltage at the defined operating point for any single point in time, allowing orders of magnitude faster sampling. During replay the DC/DC converter is then forced into the very same operating point.

We use TI's BQ25504, an industry-standard boost converter that charges a storage element from a harvesting source with a voltage of 100 mV or higher. It provides over- and under-voltage protection, and a *storage-voltage-in-operating-range signal* that indicates whether the attached storage element is in a configurable operating range. This signal changes to high when the voltage on the storage element rises above the upper threshold and changes to low when the voltage falls below the lower threshold. An attached load, such as a sensor node, can use this signal to schedule its activity based on the amount of energy in the buffer. We also use it to control our dummy load, as explained in the following paragraph.

Dummy load. The DC/DC converter operates the harvester independent of the state of charge of the capacitor, but this only holds within certain limits. For example, the BQ25504 can only charge a storage element up to 5.5 V. Above this threshold it switches off the regulator so no more current is flowing from the harvester. Similarly, once the voltage on the capacitor falls below 1.8 V, the BQ25504 operates in cold-start mode, where a charge pump operates the harvester at a fixed operating voltage with reduced efficiency. Thus, it is important to keep the voltage on the storage element within these limits to guarantee independence of the harvester's operating point and the capacitor voltage during recording. To this end, we use a *dummy load* that consumes the harvested energy from the capacitor when the voltage reaches a defined threshold (currently 2.8 V). We make use of the storage-voltage-in-operating-range signal of the BQ25504 to switch on an electric load consisting of two parallel light emitting diodes (LEDs). This load remains on until the capacitor voltage falls below a lower threshold (currently 2.3 V).

Harvesting recorder. The harvesting recorder measures voltage and current flowing from the harvester to the DC/DC converter.

Accurately measuring current at high rates over a wide dynamic range is challenging. The challenge lies in converting the range of current to a range of voltage that matches the input range of the ADC without negatively affecting the signal source and without introducing excessive noise into the signal chain.

We use a shunt am-meter (R_S and A_1 in Figure 2.7a) that measures the voltage drop as current flows through a resistor. Using a large resistor results in a large voltage drop, effectively reducing the voltage seen by the load (burden voltage). Conversely, a small resistor produces a small voltage drop, requiring significant amplification before the ADC. Since the collective noise at the input stage of the amplifier is also amplified this results in a lower signal-to-noise ratio (SNR). The noise is usually broadband and can be reduced by downsampling; however, this reduces the effective sampling rate.

We use a small shunt resistor of $R_S = 1\,\Omega$ to keep the burden voltage low. The voltage drop is amplified with an ultra-low noise instrumentation amplifier (A_1), keeping noise levels to a minimum. The amplified voltage is sampled using a high-resolution successive approximation register ADC that has a second-order low-pass filter on the input to band-limit the noise entering the ADC.

Measuring voltage is relatively easy as typical harvesting voltages are well within the range of commonly used ADCs. However, the input impedance of the ADC is relatively low, allowing current to flow from the harvesting source into the inputs of the ADC affecting the measurement. Therefore, we use a low-noise op-amp with very low input bias current as voltage buffer (A_2 in Figure 2.7a).

Harvesting emulator. For replaying traces previously recorded or generated in software, we must independently set voltage and current at the input of the DC/DC converter. The BQ25504 can be supplied with a reference voltage to which it regulates the input. We use a high-speed, high-resolution DAC to dynamically generate this reference voltage according to the sequence of digital values in the recorded/generated trace. We set the current using the precision high-side voltage-to-current converter shown in Figure 2.7b. This circuit provides a current to a ground-referenced load (R_R) proportional to the voltage on the input. This voltage is generated with a second DAC that is identical to the one used for the reference voltage. The resulting current flows through resistor R_1, causing a voltage drop. The amplifier A_2 regulates the current through R_2 such that the corresponding voltage drop equals the voltage over R_1, effectively setting i_{emu} proportional to the input voltage from the DAC.

Load recorder. To study the collective behavior of a group of harvesting nodes running a distributed application, it is essential to monitor their power draw. To this end, we provide a *load recorder* that measures the voltage on the capacitor and the instantaneous current drawn by the sensor node. The load recorder has similar requirements as the harvesting recorder and is thus almost identical to the one shown in

Figure 2.7a. The only difference is a slightly cheaper instrumentation amplifier with a wider common-mode range supporting current measurements at load voltages above 4 V.

Constant voltage source. We also add a *constant voltage source* that can be dynamically enabled and disabled in software at runtime. The voltage can be set to values in the range from 2.2 V to 3.3 V using a potentiometer. The constant voltage source serves three main purposes: (*i*) it supplies the sensor node with a stable voltage during programming/debugging; (*ii*) it allows to pre-charge the energy buffer to a defined initial filling level; (*iii*) it allows to use Shepherd as a portable testbed for non-harvesting sensor nodes.

EEPROM. The Shepherd cape also features a 256 kB electrically erasable programmable read-only memory (EEPROM) to store the name of the cape, a unique ID, and hardware calibration parameters. Based on the name of the cape, the software running on the BeagleBone can automatically configure peripherals; for example, it configures the corresponding GPIO pins as input. Storing calibration data on the cape hardware also relieves the user from the task of keeping track of which cape is mounted on which BeagleBone.

2.5 Shepherd Software

The Shepherd software stack consists of four main components: The *PRU firmware* controls the hardware on the Shepherd cape. The *kernel module* provides an interface between the PRU firmware and the user-space code, and synchronizes the PRU clock to the Linux host clock. The *user-space code* handles data storage and retrieval, and exposes a high-level user interface to manage all underlying software and hardware. Finally, the *user interface* provides a convenient way to start/stop recording and emulation from the user's machine on a collection of distributed Shepherd nodes.

2.5.1 Data Handling

The key challenges in Shepherd's software architecture include time synchronization and high-throughput data exchange among user-space code, Linux kernel code, and the PRUs subject to timing constraints. Our solution uses four mechanisms: (*i*) interrupts between the ARM Cortex-A9 and the two PRUSS cores; (*ii*) the Linux *remoteproc* framework that manages resources and controls the state of the PRUs; (*iii*) the remote processor messaging (RPMSG) protocol, a standardized messaging solution for communication with the PRUs; and (*iv*) shared access to the main DDR RAM.

In the following, we describe the data exchange during emulation as the more general case that involves bi-directional data exchange: harvesting data is transferred from the database to the frontend, and load recordings are sent from the frontend to the database. We use the *remoteproc* framework to allocate an area in the DDR RAM that can be accessed from Linux and from the PRUs. We divide this memory into 64 buffers that can each store 10,000 current and voltage samples. The user-space code starts by copying data from the database into memory, buffer by buffer. After writing a complete buffer, the corresponding index is sent to the PRU core that is responsible for data acquisition (PRU1) as an RPMSG. The RPMSG communication is double-buffered such that both sides can keep writing into the corresponding queues, while the other side is busy. PRU1 retrieves the buffer index from the queue and transfers the samples one by one to the DAC based on a sampling trigger generated by PRU0. After sending one sample to the DAC, it samples the ADC and overwrites the memory from which the DAC sample was read. After processing a complete buffer, PRU1 returns the buffer index to the RPMSG queue. The user-space code receives the buffer index from the queue, copies the data to the database, and fills the buffer with the next block of emulation data.

Database. We currently use the *hdf5* file format to store data locally on each Shepherd observer, using an SD card or USB flash storage. Together with the raw data, we store calibration values retrieved from

the EEPROM on the Shepherd cape. This approach allows to easily share data that can be viewed and replayed independent of the hardware it was recorded with. After recording, data can be conveniently downloaded from individual nodes and merged into a single file using the command-line utilities we provide (see Section 2.6). Similarly, for emulation, the corresponding data are uploaded to each Shepherd node before the experiment starts.

2.5.2 Time Synchronization

Meaningful experimentation with a collection of batteryless, energy--harvesting devices requires that both the recording and the emulation of harvested energy happens synchronously on all devices under test. Similarly, recordings of load voltage and current have to be time-synchronized in order to interpret node interactions.

We use an approach where each sample is scheduled at a defined point in time. At a sampling rate of 100 kHz, samples are always taken at the wrap of full 10 µs. This has two advantages: (*i*) While synchronized, all nodes take the same number of samples on average. (*ii*) It is sufficient to timestamp the first sample in any 'block' of samples: The timestamp of all following samples in that block can be derived from this first timestamp using the sampling interval.

The goal of time synchronization is to take and replay samples at the same specified time on a number of Shepherd observers with as little jitter as possible. In essence, this means that the PRUSS cores on the observers, which handle the direct interaction with the hardware on the analog frontend, need to act in concert. We describe the synchronization on two levels: (*i*) How to synchronize the Linux host clocks of a number of Shepherd observers using (a) GPS and (b) PTP, and (*ii*) how to locally synchronize the PRUSS cores to the Linux host clock.

GPS. Using GPS allows to globally absolutely synchronize clocks with high accuracy: GPS fundamentally relies on accurate time-of-flight measurements between the receiver and multiple satellites. For this purpose, the receiver has to be tightly synchronized to the satellites.

Many receivers output a pulse per second (PPS) signal, a sharp rising or falling edge of an electrical signal with the wrap of every second. Together with information about the global time of that second, it is relatively easy to synchronize another node to the globally accurate GPS time. Using *gpsd* and *chrony*, the host clock of the Linux OS running on the BeagleBone can be synchronized.

The advantage of using GPS is that the Shepherd nodes do not need to be physically connected. For example, Shepherd can be used to explore solar energy harvesting LPWAN networks where nodes can be kilometers apart making physical connections infeasible. However, GPS receivers can be expensive and always require line of sight to the satellites, severely limiting deployment options.

PTP. As an alternative, we consider PTP, which is a time synchronization protocol based on transmission time measurements over standard Ethernet links. A number of slave nodes synchronize to a common master, which can be elected automatically based on clock quality estimation. TI's AM3358 SoC, the core of the BeagleBone, implements hardware time-stamping of Ethernet packets, significantly improving the achievable accuracy of PTP. PTP is readily available in Linux and, combined with *phc2sys* allows to synchronize the Linux host clocks of multiple Shepherd observers. When combining GPS and PTP, every partition that does not have an Ethernet connection to all other nodes must contain one GPS master to achieve global time synchronization across all nodes.

PRUs. The last stage of the synchronization hierarchy is the synchronization between the Linux host clock and the PRUs. To the best of our knowledge, there exists no standard approach so we developed a custom synchronization procedure.

At a fixed time in every interval (currently after 5 ms in a 100 ms interval), the Linux kernel module timestamps the Linux host clock and immediately sends an interrupt to one of the PRU cores (PRU0). On reception, PRU0 immediately timestamps its own clock and requests the corresponding timestamp from the kernel module. With these two

values, the PRU estimates its offset from the host clock. By storing the last pair of timestamps, it also estimates the clock drift with respect to the host clock. This way, the PRUs can accurately schedule all samples until the next synchronization interval.

2.6 Using Shepherd

Public testbeds expose an application programming interface (API) or graphical user interface (GUI) to the user, while the underlying implementation is only relevant to the testbed maintainers. By contrast, users of Shepherd are exposed to the full hardware/software stack. We anticipate two different types of users: The first group has no specific hardware requirements and wants to get started quickly by using the reference hardware and software we provide. Ideally, a user from this group boots up the BeagleBones and runs the provided software to record and emulate harvesting traces. The second group of users requires dedicated hardware and/or software solutions to achieve their goal. For this group of users, modularity and composability are key: They must be able to quickly change parts at every level of the hardware/software stack.

We aim for a flexible, yet robust solution that satisfies the needs of both types of users. We implement Shepherd using standard Linux interfaces and the most high-level programming language possible for a given task. That is, only the SPI transfer routines are written in assembly; we use C for the Linux kernel module and the firmware of the PRUs; software running in user-space is written in Python. Users interact with Shepherd in four ways.

1) Installation: After mounting the Shepherd cape onto the BeagleBones, users download and flash the latest Ubuntu image to the SD cards by following the instructions on the BeagleBone website. Then they log into each node using the default credentials to set a unique hostname and configure public/private key based `ssh` access. Finally, they deploy the Shepherd software stack by installing two Debian packages, which we provide as release artifacts on the Shepherd repository. These

steps can be executed manually for each node or by running the provided Ansible playbook against all nodes with one command from the user's machine.

2) Calibration: Although Shepherd can work with default values derived from the hardware specs, the recording and emulation accuracy is greatly improved when applying per-node calibration, as evaluated in Section 2.8. We provide one walk-through example on how to fully automate the calibration process using a Keithley SMU, which can be controlled remotely over Ethernet or USB. We also provide a second example for users with access to a stock lab-bench power supply and multimeter. The example guides the user through the calibration process in a step-by-step manner, prompting the required reference values on the command line. Using these scripts, the calibration procedure takes roughly 10 minutes per node.

3) Usage: To record and replay harvesting traces, users invoke two command-line utilities. *shepherd-sheep* is run locally on each Shepherd observer node and provides a rich interface to start recording or emulation, run a GUI webserver, or to start a remote procedure call (RPC) interface. A user may use this interface to investigate the behavior of a single battery-less node. Nevertheless, the key functionality of Shepherd is the ability to orchestrate a number of time-synchronized Shepherd nodes. To this end, users can run the *shepherd-herd* tool on any host with `ssh` access to the network of Shepherd observers, which provides similar functionality as *shepherd-sheep* but takes as an argument a list of Shepherd nodes on which the corresponding commands should be executed.

4) Development: We make Shepherd's hardware and software available as open source, and encourage users to provide feedback and develop new features. We help users getting started with three measures: (*i*) modularity and extensive documentation, (*ii*) a test suit for catching software bugs early and showcasing functionality, (*iii*) automation of build and deployment tasks using standard tools.

2.7 Shepherd in Action

This section demonstrates the utility of Shepherd when testing a distributed algorithm for batteryless nodes in a real environment.

Scenario. We consider an indoor solar energy harvesting scenario. The testbed comprises three Shepherd nodes. Each node consists of a BeagleBone Green, the Shepherd cape, a 7 cm $\times$ 2 cm solar panel mounted onto a solar capelet, a nRF52840 capelet, and a storage capelet that hosts a 150 µF ceramic capacitor. The three Shepherd nodes are placed on tables inside a room without windows, receiving only little light through a glass door. Node 2 is slightly tilted, facing that glass door, while nodes 1 and 3 are oriented horizontally.

We develop an example application for the nRF52840 that wakes up from system-off mode when it sees a rising edge on the voltage-in-operating-range pin of the BQ25504, which is triggered when the capacitor voltage exceeds 2.8 V. After initialization, the application senses the supply (*i.e.*, the capacitor) voltage every 125 ms. When the supply voltage reaches 3 V, it executes a task that involves sampling the temperature sensor and transmitting the reading to a remote base station by embedding it into BLE advertisement packets sent on the three corresponding channels. The application continues to sample its supply voltage until the 3 V threshold is again exceeded or the voltage-in-operating-range pin of the BQ25504 is pulled low as the supply voltage falls below 2.3 V. The latter causes an interrupt that puts the nRF52840 into system-off mode from which it only wakes up on a rising edge of the voltage-in-operating-range pin. The application indicates state changes (system-off, sleeping, sampling supply voltage/ADC, transmitting BLE advertisement packets) using GPIO pins. In this way, we can track the logical state of each node with high resolution.

Recording. We record harvesting current and voltage for 60 seconds synchronously on the three Shepherd nodes. Initially, the room lights are switched off. After 30 seconds, we turn the room lights on and keep recording for another 30 seconds.

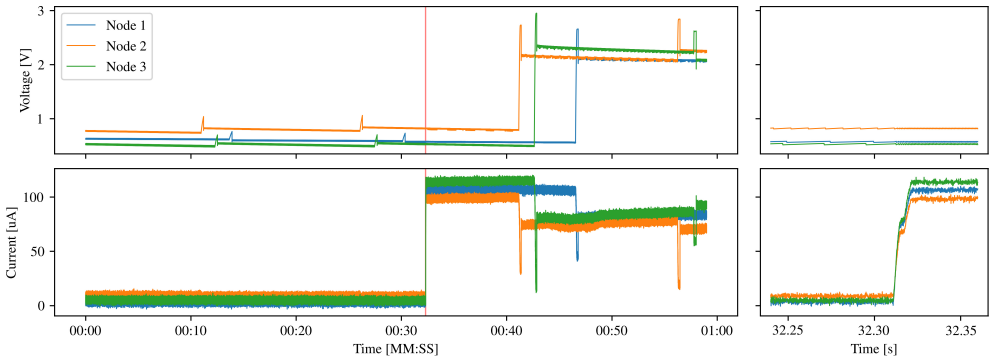


FIGURE 2.8: Voltage and current traces recorded by three Shepherd nodes in an indoor solar energy-harvesting scenario. The plots on the right zoom into the moment the room lights are switched on (marked in red on the left).

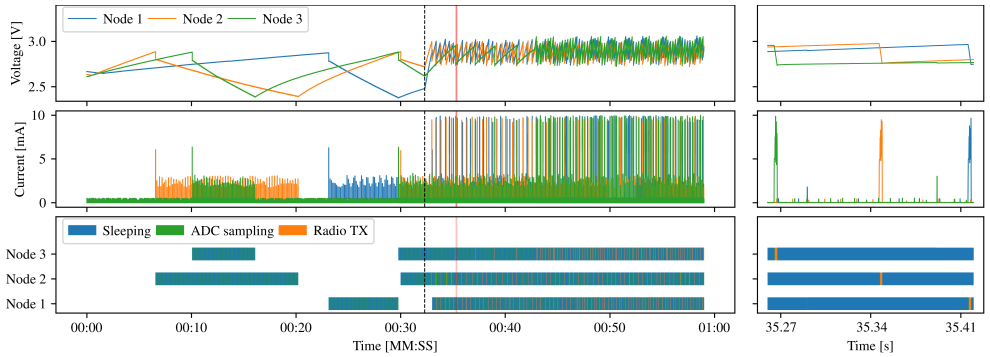


FIGURE 2.9: Capacitor voltage, current, and logical states when replaying the traces from Figure 2.8 to three sensor nodes. The black dashed line indicates when the lights are switched on. The plots on the right zoom into the time marked with a red solid line.

Figure 2.8 plots the voltage and current recorded on the three nodes. Initially, the voltages are low. The spikes at around 12, 30, 45, and 60 seconds are due to MPP tracking. To this end, the BQ25504 shortly disconnects the harvester and samples the solar panel's open-circuit voltage. For the next 16 seconds, it regulates the voltage to 80 % of that voltage. As soon as the light is switched on, the current sharply rises on all three nodes, as shown in the zoomed-in plots to the right. However, the voltage remains low as the converter keeps it at the regulation point. Only after the next MPP tracking at around 40 to 45 seconds, the voltage is also increased to track the new MPP.

Emulation. We replay the recorded traces to the nRF52840 devices running our example application, while recording capacitor voltage,

current, and GPIO traces. Figure 2.9 shows that all three nodes start in system-off mode. The small amount of energy extracted from the solar panels is sufficient to slowly charge the capacitors while the nodes are still powered off. Following the recorded harvesting traces, node 2 receives most energy and is thus the first to reach the power-on threshold of 2.8 V. The wake-up causes a significant current spike and corresponding voltage drop, while the MCU initializes memory and peripherals. However, as long as the room lights are switched off, the periodic sampling of the supply voltage using the ADC is not sustainable, leading to a decreasing capacitor voltage and eventually power-off. After switching on the room lights, the nodes power up quickly and accumulate enough energy to read out the temperature sensors and send the readings to the base station. In the zoomed-in plots to the right we see that the nodes execute the task at different times and that the execution drains the capacitor voltage much faster than it rises during charging.

Reproducibility. One of Shepherd's strengths is its ability to emulate spatio-temporal energy availability. Given a deterministic application, this should also lead to a consistent behavior across successive emulation runs with the same harvesting traces. To quantify this reproducibility, we consider three parameters of our example application that may be of interest to a developer: (*i*) *#packets* is the total number of packets sent by a node during a 60-second experiment with the recorded traces from Figure 2.8; (*ii*) *wake-up time* is the time from the start of an experiment until the node executes the sense-and-send task for the first time; (*iii*) *sleep time* is the total time spent in sleep mode (*i.e.*, powered on and waiting for an interrupt).

We replay the traces ten times and measure the three application-level parameters for all three nodes. Table 2.1 lists for each parameter and node the mean and the *error*, defined as the maximum absolute difference between any two repetitions. The errors are very small across all nodes and parameters. This demonstrates the ability of Shepherd to accurately reproduce application behavior, which can greatly aid in the development and evaluation of batteryless applications and system

TABLE 2.1: Application parameters when replaying the same energy-harvesting traces to three nodes ten times. The *error* is the max. absolute difference between any two repetitions.

	#packets		Wake-up time		Sleep time	
	Mean	Error	Mean	Error	Mean	Error
Node 1	238.5	4	33.4 s	0.1 s	23.6 s	0.7 s
Node 2	177.1	6	32.8 s	0.3 s	36.3 s	0.8 s
Node 3	226.6	9	35.1 s	0.5 s	26.4 s	0.9 s

services. Overall, the observations and insights presented throughout this section would be very difficult, if at all possible, to attain without a tool like Shepherd.

2.8 Performance Evaluation

The previous section showed some of Shepherd's capabilities, suggesting that Shepherd meets the performance requirements from Section 2.3. This section shows that this is indeed the case for the current hardware/software stack by answering the following questions:

- What is the time difference between two nodes when taking or replaying a sample at the supposedly same time?
- What are the electrical characteristics affecting the resolution, accuracy, and sampling rate of voltage and current?
- What is the resolution, frequency, and synchronization with which Shepherd can trace GPIO pin changes?
- Does the average power draw of a Shepherd node allow for extended experiments without mains power supply?

For reference, Table 2.2 summarizes the main performance characteristics of our current Shepherd implementation.

2.8.1 Time Synchronization Accuracy

Ideally, we would measure synchronization accuracy by comparing the times when the ADCs on two Shepherd nodes close their sample

TABLE 2.2: Summary of Shepherd performance specification.

Sampling rate		100 kHz
Range	Harvesting voltage	100 mV-3 V
	Harvesting current	0 mA-50 mA
	Load voltage	0 V-4 V
	Load current	0 mA-50 mA
	Emulation voltage	100 mV-3 V
	Emulation current	0 mA-50 mA
24 h DC Accuracy	Harvesting voltage	19.53 µV $\pm$ 0.01 %
	Harvesting current	381 nA $\pm$ 0.07 %
	Load voltage	19.53 µV $\pm$ 0.01 %
	Load current	381 nA $\pm$ 0.01 %
	Emulation voltage	11 µV $\pm$ 0.012 %
	Emulation current	191 nA $\pm$ 0.025 %
RMS Noise (@1 kHz)	Harvesting voltage	50 µV (14 µV)
	Harvesting current	3 µA (0.4 µA)
	Load voltage	48 µV (10 µV)
	Load current	4.5 µA (0.9 µA)
Bandwidth	Recording channels	15 kHz
Risetime	Emulation voltage	65 ms
	Emulation current	19.2 µs
Max. burden voltage	Harvesting recorder	50.4 mV
	Load recorder	76.1 mV
Min. GPIO sampling rate		580 kHz
Avg. current draw		345 mA
Max. synchronization error		2.4 µs

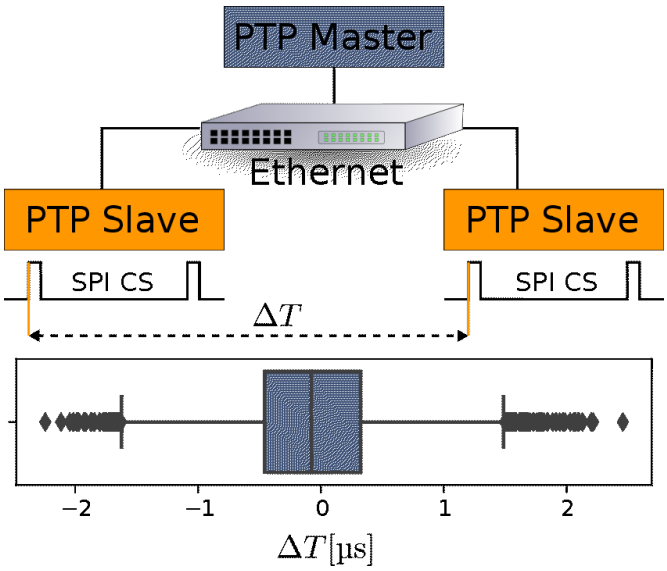


FIGURE 2.10: In **setup A**, two Shepherd nodes are synchronized to one PTP master over a COTS Ethernet switch.

and hold switch. However, this switch is not accessible from outside. According to the datasheet, the ADC starts the conversion on the falling edge of the SPI chip select (CS) signal. We thus measure the delay between 10,000 consecutive falling edges of the SPI CS signal on two nodes to determine the synchronization accuracy. There may be jitter introduced by the circuitry inside the ADC/DAC, but we expect it to be small. We consider two setups:

- **Setup A** (Figure 2.10): In this setup, we use two nodes that are connected over an off-the-shelf Ethernet switch and synchronized to a common PTP master within the same network.
- **Setup B** (Figure 2.11): We use four nodes in two separate Ethernet networks. Each network consists of a Shepherd node acting as PTP slave that is connected over Ethernet to a PTP master with a GPS reference clock. The two networks are physically separated, representing scenarios where nodes are mobile or too far from each other for a direct, wired connection.

We plot the median and 25th/75th percentiles of the synchronization error at the bottom of Figs. 2.10 and 2.11; the dots to either side of the

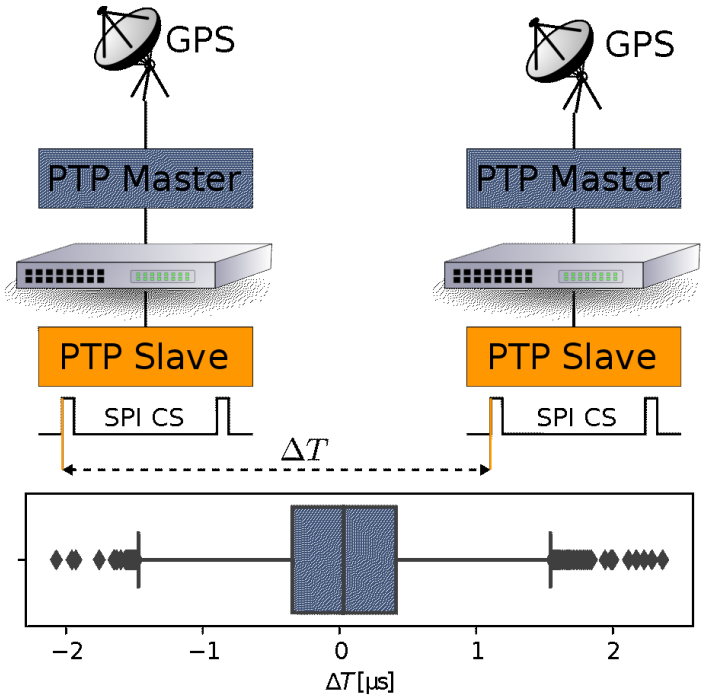


FIGURE 2.11: In **setup B**, two Shepherd nodes are part of two separate Ethernet networks with their own PTP master that is synchronized to global GPS time.

whiskers are outliers. Even with the outliers, we find that the maximum synchronization error across both setups is as small as 2.4 µs: 91 % of the measurements are within our targeted synchronization accuracy of 1 µs, also for setup B in which the Shepherd nodes have no wired connection between each other (see Figure 2.11).

2.8.2 Electrical Characteristics

Zero-input root-mean-square (RMS) noise. The noise floor crucially determines the effective resolution, that is, the smallest signal change that can be differentiated. There are various sources of noise in the signal acquisition chain, including switching noise from the DC/DC converter and thermal noise from the shunt resistors. Assuming that the noise has zero mean, is uncorrelated, and uncorrelated to the measured signal, the noise power can be reduced, for example, by a factor of two by averaging over four consecutive samples at the cost of a decreased data rate.

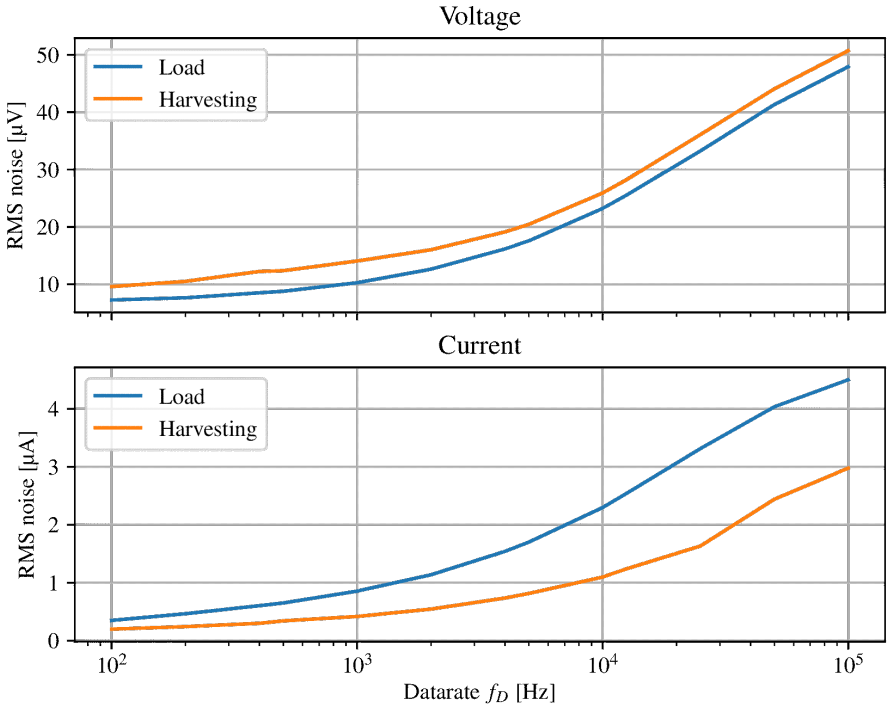


FIGURE 2.12: RMS noise against data rate for all four channels.

We use a Keithley SMU2604B SMU to apply a zero-input signal to each of the four channels of a Shepherd node (*i.e.*, harvesting voltage/current, and load voltage/current) and sample for 10 seconds at Shepherd's fixed sampling rate of $f_S = 100\,\text{kHz}$. Figure 2.12 plots the RMS of all four channels against the data rate f_D after averaging over f_S/f_D samples. We see that Shepherd's voltage resolution is much better than the required 1 mV, and that the current resolution is within the required 1 µA for data rates below 8 kHz.

Burden voltage. Measuring current with a shunt resistor introduces a voltage drop between the source and the load. Similarly, the switches used to select between recording and emulation for the harvesting

TABLE 2.3: Maximum burden voltage and impedance.

Channel	Maximum burden voltage [mV]	Impedance [Ω]
Harvesting	50.4	1.008
Load	76.1	1.342

channel as well as between dummy load and sensor node for the load channel also cause a voltage drop that would not be present in an ideal system. To measure the burden voltage, we use the Keithley SMU from before to perform a current sweep with a resolution of 1 mA. For every value, we measure the resulting voltage drop over the measurement circuitry and the control logic.

The results reveal a linear dependency between the burden voltage and the current, and that the burden voltage is dominated by the respective shunt resistor. Table 2.3 lists the maximum burden voltage and the corresponding impedance. Because current and voltage on a harvester typically correlate, the impact of losses due to burden voltage are small; for example, the loss is less than 2 % for the maximum current and voltage supported by Shepherd.

Recording bandwidth. We use a 51 Ω resistor as load and an AIM-TTI TG5011 function generator to measure the bandwidth of all four recording channels by configuring it for an amplitude of 2 V and a linear frequency sweep of up to 50 kHz. We define the bandwidth as the frequency at which the measured amplitude falls below −3 dB with respect to the 2 V input. We find that the bandwidth of all four channels ranges between 15.1 kHz and 15.2 kHz, which corresponds to the nominal bandwidth of the low-pass filter built into the ADC.

Emulation rise time. A critical parameter for the harvesting emulator is the time it takes to change its operating point. We measure the rise time for the current source by inserting a resistor between its output and ground, and then applying a low-frequency square wave with an amplitude from 0 A to 50 mA through the DAC. For the voltage emulation path, we use a 220 µF capacitor and the dummy load, and apply a constant current of 500 µA using a Keithley 2604B. We measure the regulated voltage at the input of the DC/DC converter while applying a low-frequency square wave with an amplitude from 100 mV to 3 V through the DAC.

We measure the time it takes for the signal to rise from 10 % to 90 % of its range with an Agilent MSO7104B oscilloscope. The 19.2 µs rise

TABLE 2.4: DC accuracy in terms of mean absolute percentage error (MAPE) without calibration and 24 hours after calibration. LSB represents the measurement resolution.

Channel		LSB	MAPE after 24 h [%]	MAPE uncalibr. [%]
harv. rec.	voltage	19.53µV	0.011	0.150
	current	381nA	0.072	0.918
load rec.	voltage	19.53µV	0.028	0.051
	current	381nA	0.018	0.714
harv. emu.	voltage	11µV	0.012	5.962
	current	191nA	0.025	20.450

time of the current source is within expectations and short enough to accurately emulate rapidly changing conditions. For the voltage channel, we observe a long rise time of 64 ms. A closer examination reveals that the set-point voltage provided by the DAC rises to its value within a few µs, but the BQ25504 only applies the reference voltage with its internal duty-cycle of 64 ms. However, this is not critical as the voltage is regulated: it typically does not change in response to changing conditions.

DC accuracy. DC accuracy quantifies how close a value measured or emulated with Shepherd is to the true value. To measure it, we use the Keithley SMU to do a sweep over the full range of the four recording channels, and log the value measured with Shepherd together with the corresponding reference value. For the two emulation channels (harvesting load/current), we reverse the setup and apply the sweep using Shepherd. For the current source, we use a 91 Ω resistor as the load and the SMU as amperemeter. For the voltage emulation path, we use a 220 µF capacitor and the dummy load, and apply a constant current of 500 µA to the SMU. We measure the regulated voltage at the input of the DC/DC converter.

We do these measurements without calibration and then again 24 hours after calibration. The mean absolute percentage error (MAPE) and the least significant bit (LSB) indicating the measurement resolution are shown in Table 2.4. We see that the calibration process described in

Section 2.6 improves the DC accuracy considerably, in particular for the two emulation channels. The MAPE values 24 hours after calibration are below 0.1 % across the board, demonstrating that Shepherd can accurately record and replay harvesting traces.

2.8.3 GPIO Tracing Performance

Shepherd can sample and store the state of up to four GPIO pins during recording and emulation. The state is continuously polled by the PRU firmware. We measure the maximum polling interval by toggling a pin within the corresponding routine and measuring the maximum delay between two edges with a Saleae Logic 8 logic analyzer. The maximum delay is 1.7 µs. This is the minimum time a pin must be high or low for the change to be recorded. It is also the maximum delay between the event and the recorded timestamp. The size of internal buffers limits the maximum frequency to 163,840 events per second, where an event represents the state change of at least one pin. As GPIO timestamping is done with the same clock used for scheduling samples, the synchronization error is in the same region as the results in Figs. 2.10 and 2.11, that is, 2.4 µs at most.

2.8.4 Power Draw of a Shepherd Node

We expect Shepherd to be used in scenarios without mains power supply. We measure the current draw of a Shepherd node while recording data using a Keithley 2604B SMU, supplying the node with 5 V through the micro USB connector. The average current draw is 345 mA with a peak current of 395 mA. Using a u-blox M8F GPS sensor for time synchronization adds another 28 mA once the receiver has acquired the first fix. This allows for a theoretical recording duration of 19 h from a 10 000 mAh USB power bank.

TABLE 2.5: Comparison of Shepherd with Ekho [53].

	Ekho [53]	Shepherd
Sampling rate	0.135 kHz / 1 kHz	100 kHz
Nominal current resolution	10 µA	0.381 µA
Current emulation error	86.9(462) µA	0.51(29) µA

2.9 Related Work

Two areas of prior work are related to Shepherd: (*i*) tools to record and emulate energy-harvesting environments, and (*ii*) testbeds for battery-supported nodes, with or without energy-harvesting capabilities. Debugging techniques for intermittent systems, however, are orthogonal to our work; for instance, the platform proposed in [21] may be integrated into Shepherd for energy-interference-free debugging of distributed batteryless applications. Shepherd deals with energy harvesting across a collection of nodes, which is independent of the communication technology used to exchange data between nodes. By developing additional capelets, Shepherd can thus be used to experiment not only with active radios, but also with ambient backscatter [83] or visible light transceivers [68].

Recording and emulation of energy sources. Ekho [53] records IV curves of a harvester and recreates these characterstics in the lab as input to a converter-less node. Ekho supports different harvesting technologies and can reproduce the energy environment of a single node, but it cannot provide insights into the spatio-temporal energy environment and behavior of multiple distributed nodes. Shepherd, instead, offers synchronized recording and emulation of multiple harvesters. The performance of Ekho has not been characterized in terms of noise levels, DC accuracy, or dynamic range, making a quantitative comparison with Shepherd difficult. Nevertheless, Table 2.5 lists the performance results provided in [53] along with the corresponding values for Shepherd, obtained by computing the mean and standard deviation of the absolute current emulation error across 24 hours based on the measurements underlying Table 2.4. We find that Shepherd achieves

orders of magnitude better sampling rate, nominal current resolution, and current emulation error.

RocketLogger [121] is a hand-held device that enables in-situ measurements across four voltage and two current measurement channels with high accuracy and wide dynamic range. It also records temperature, illuminance, etc. to support the analysis of environmental statistics. Similar to Shepherd, RocketLogger aims to bring the capabilities of high-quality, expensive, wall-powered lab equipment into a compact and portable measurement device. Unlike Shepherd, RocketLogger cannot replay recorded or generated traces, and is only applicable to single energy-harvesting devices.

Recent work aims at repeatable indoor testing of solar-powered nodes by controlling the intensity of a light source (*e.g.*, based on real illumination data) to induce a solar cell or panel to generate a desired level of power [127, 52, 94]. Using enclosures for the node or photovoltaics, it is possible to create repeatable light conditions. All solutions are specifically designed for a certain battery-supported platform. By contrast, Shepherd's design is platform-agnostic and applicable to batteryless devices and different harvesting sources. Moreover, it can be used to record the harvesting conditions, which is not possible with any of the proposed solutions.

Testbeds for embedded wireless nodes. Testbeds for battery-supported nodes enable development, distributed debugging, and performance measurements. The capabilities of existing testbeds range from basic support for reprogramming and serial logging [36] through synchronized GPIO tracing/actuation [81] and JTAG debugging [123] to power profiling [81] and the generation of controllable Wi-Fi interference [118]. Shepherd is inspired by these testbeds and brings many of their services to batteryless, intermittent systems. In addition, it adopts a fundamentally different approach to provide services like recording and emulation of spatio-temporal energy availability that no existing testbed offers.

Indeed, a few papers outline the challenges and desired capabilities

of a testbed for energy-harvesting [128] or transiently powered sensor nodes [1]. The prototypes, however, lack even basic features such as time synchronization and energy consumption measurements [1] or record and replay of harvesting traces [128]. Shepherd provides a more powerful solution that is available as an affordable, portable, and open-source tool for the research community.

2.10 Conclusions

We have presented Shepherd, a testbed for collections of batteryless devices that can accurately record and replay the spatio-temporal characteristics of real energy environments. Shepherd's modular design is agnostic to harvesting source, energy storage, node platform, and communication technology; it is affordable and portable; and our experiments show that it provides adequate accuracy, resolution, sampling rate, and time synchronization. We believe Shepherd can be a valuable tool for the research community to investigate exciting questions that have been out of reach so far.

Postscript. Shepherd was developed out of a necessity for a tool to approach the key problems of battery-free networks. Since its completion, it has proven indispensable and was a key enabler for the research presented in Chapter 3 and Chapter 4 as well as research conducted with our partners in the space of energy-harvesting based sensing [113, 114].

Based on the feedback from users and our own experiences after three years of using Shepherd we are currently developing an improved version with the following new key features:

- Flexible recording frontend that allows sampling of voltage-current surfaces and emulation of different harvesting architectures
- Virtual energy storage for experiments with variable capacities
- Configurable turn-on and turn-off thresholds

We plan to build on the success of existing testbeds [36, 123, 81, 118] for battery-powered devices by operating a public instance of this improved version testbed consisting of dozens of nodes that are controlled remotely by users worldwide to conduct reproducible experiments.

3

Bootstrapping battery-free wireless networks: Efficient neighbor discovery and synchronization in the face of intermittency

Prelude. This chapter covers the paper with the same title co-authored by Marco Zimmerling that I presented at the USENIX Symposium on Networked Systems Design and Implementation in 2021 [46]. The paper addresses the unsolved problem of efficient device-to-device communication in the face of intermittency. It presents Find, the first neighbor discovery protocol for battery-free wireless networks that uses randomized waiting to minimize discovery latency. Additionally, it introduces Flync, a new hardware/software solution that synchronizes indoor light harvesting nodes to powerline-induced brightness variations of widely used lamps, which is exploited to further speed up neighbor discovery.

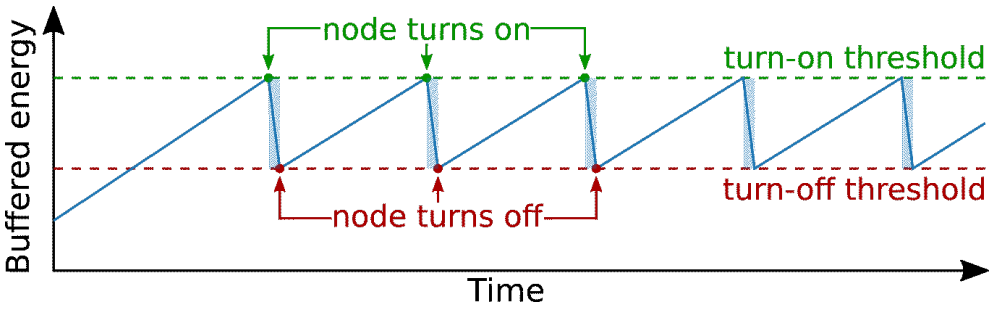


FIGURE 3.1: Because ambient power is often weak, a battery-free node must buffer energy before it can wake up and operate for a short time period. This is known as intermittent operation.

3.1 Introduction

Despite technological advances, the maintenance costs and environmental impact of batteries remain a major threat to the vision of a truly ubiquitous internet of things [6, 20]. *Battery-free devices* that store energy harvested from light, vibrations, radio-frequency (RF) signals, and other ambient sources in a capacitor are one of the most viable alternatives today [115]. Capacitors store electrical energy in an electrical field rather than in the form of chemical energy, and thus have negligible aging effects and are sustainable [25, 11]. Moreover, their favorable size, weight, and cost points enable new applications where batteries would be inconvenient or infeasible [79].

Challenge. The power that can be harvested from ambient energy sources can vary significantly across time and space [44], and is often too weak to directly power a battery-free node, such as a smart sensor [83]. Thus, as illustrated in Figure 3.1 and further discussed in detail in Section 3.7, a battery-free device first needs to buffer sufficient energy in its capacitor before it can operate for a short period of time; then the device turns off until the capacitor is sufficiently charged again. As a result, battery-free devices operate *intermittently*.

Intermittency is in stark contrast to conventional duty cycling. While duty cycling is intentionally introduced to save energy and thus predictable, intermittency is mainly dictated by uncontrollable environmental factors and thus impacts the device operation in unpredictable

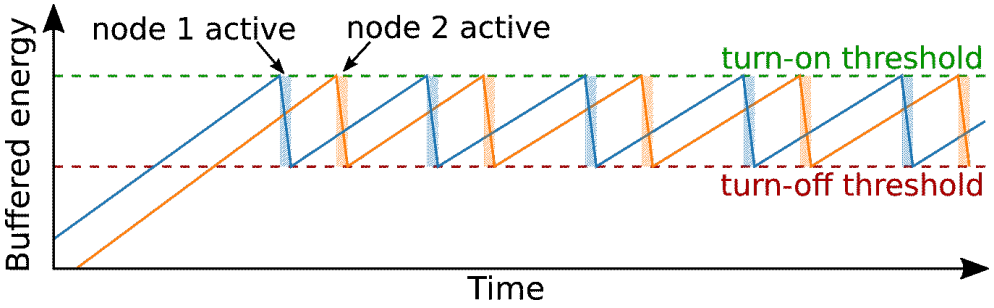
ways. The resulting challenges in terms of, for example, reliable time keeping [56, 27] or ensuring application progress and data consistency [88, 16] have been widely studied in the recent literature.

The impact of intermittency on wireless networking has instead received little attention. Just like in conventional battery-supported networks, direct communication between battery-free devices is desirable, for example, to increase the availability of the system [93], to enable novel applications [83, 59], and to reduce infrastructure costs [92]. However, to communicate with one another, sender and receiver must be active simultaneously for at least the airtime of one complete packet. This is challenging in battery-free networks for three reasons:

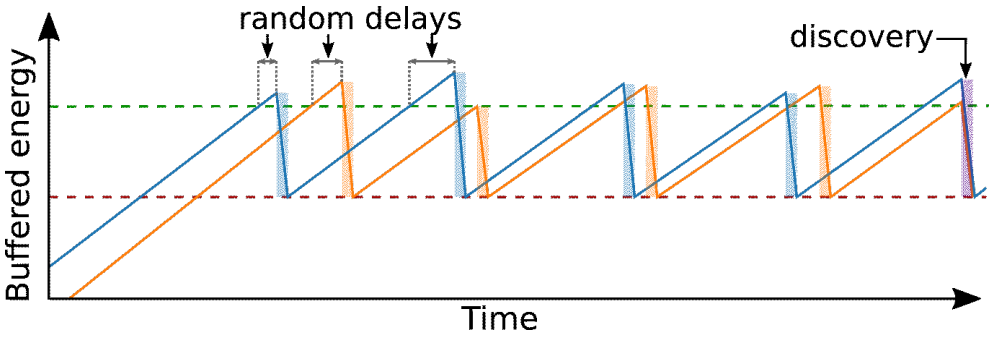
1. Battery-free nodes can only become active when they have accumulated sufficient energy in their capacitors.
2. They may only be active for a short period, which renders excessive sampling of the wireless channel infeasible.
3. Their duty cycles are often low and may change unpredictably due to varying availability of ambient energy.

For example, our prototype battery-free node needs to charge its capacitor for hundreds of milliseconds to sustain 1 ms of activity when harvesting from indoor light. Because the short activity phases of different nodes are generally interleaved, as shown in Figure 3.2a, it takes a long time until nodes encounter each other. And this is not a one-time endeavor: While nodes may attempt to synchronize their activity phases at the first encounter, they lose track of time during extended periods without energy [56, 27], which forces them to re-synchronize.

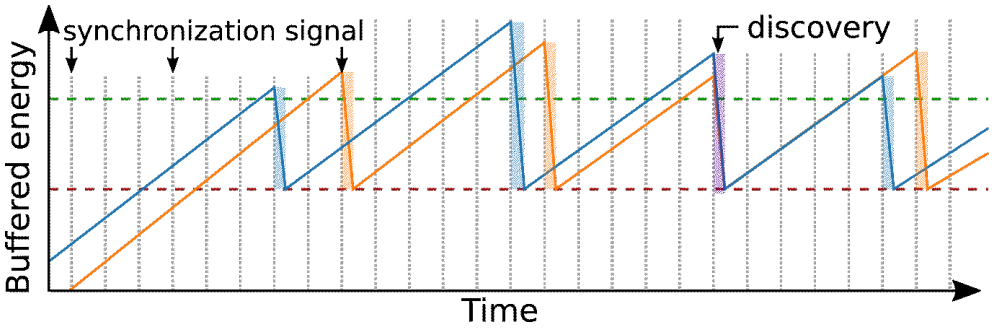
This challenge is fundamental and pertains to battery-free networks regardless of the type of wireless communication: While backscatter communication can lower the energy costs compared to active radio



(A) Battery-free nodes may need a long time to discover each other due to low duty cycles and the interleaving of short activity phases.



(B) Using Find, nodes randomly delay their wake-ups to avoid interleaving, thereby discovering each other faster and more efficiently.



(C) Using Find + Flync, nodes implicitly align their wake-ups to an external synchronization signal, further accelerating discovery.

FIGURE 3.2: Illustration of the battery-free neighbor discovery challenge in (a) and of our proposed mechanisms to address it in (b) and (c).

communication, sender and receiver still need to have sufficient energy at the same time. Prior work on backscatter has primarily focused on pushing the envelope of communication range and throughput, avoiding intermittency by evaluating the designs under high ambient energy availability [83, 59] or by powering the devices via USB or batteries to not disturb the measurements [92]. To our knowledge, direct radio communication between real battery-free devices has not been explored so far, as the overhead due to intermittency is considered too demanding [93].

Contribution. We set out to bootstrap battery-free wireless networks by presenting two mechanisms that enable battery-free nodes to discover each other quickly and efficiently.

The first mechanism, Find, is a neighbor discovery protocol. As illustrated in Figure 3.2b, the key idea behind Find is to address the interleaving problem by introducing random delays after the devices have sufficiently charged their capacitors before becoming active. We develop analytical models to determine an optimized delay distribution that minimizes discovery latency. At runtime, each Find node dynamically adapts the delay distribution to changes in its energy availability.

The second mechanism, Flync, is a hardware/software solution that further speeds up the discovery process. Flync phase-synchronizes solar energy harvesting devices to powerline-induced flicker of state-of-the-art lamps; the proposed circuit draws only 5 µW of power. As shown in Figure 3.2c, using Find together with Flync, nodes can implicitly align their activity phases to this external synchronization signal, dramatically increasing their chances to be active at the same time.

We prototype our mechanisms on a custom-designed ultra low-power battery-free node. It is based on a state-of-the-art MCU with a 2.4 GHz BLE radio, and buffers energy harvested via three small solar panels in a tiny 47 µF ceramic capacitor.

We use 6 of our prototype battery-free nodes to conduct extensive experiments and a contact-tracing case study. We summarize our key findings as follows:

- Find provides shorter discovery latencies than greedy and naïve random node activations. Find + Flync improves on greedy by 4.3× in terms of the median latency (141 s vs. 604 s); the 99th percentile improvement is 34.4×.
- Our hardware prototype works with 14 out of 19 fluorescent, halogen, and LED lamps we tested, demonstrating that Flync is broadly applicable in indoor environments. Flync provides a stable clock signal when nodes are deployed across different rooms, carried around, or exposed to temporary shadowing.
- We conduct a contact-tracing case study in an open-air pub with Find and in an office kitchen using Find + Flync. The median time between consecutive encounters of the same two nodes is 1.5 s and 7.5 s in the outdoor and indoor environment, respectively. This shows the potential of our battery-free designs for real-world applications.

Overall, this paper makes the following contributions:

- Find, the first neighbor discovery protocol for battery-free networks. Find is agnostic to the energy harvesting modality and the type of wireless communication.
- Flync, the first solution extracting a stable clock from solar harvesting current, whose amplitude changes due to powerline-induced flicker of state-of-the-art lamps. While we use Flync in tandem with Find to speed up discovery in indoor scenarios, Flync is useful for other purposes and also applicable to battery-supported nodes.
- A novel battery-free node design including an implementation of an efficient intermittent runtime.
- Empirical evidence that the proposed techniques work well under a diverse set of real-world conditions.

3.2 Battery-free Neighbor Discovery

This section presents the design of Find, the first neighbor discovery protocol for battery-free wireless networks. Find empowers battery-free nodes to quickly discover each other's presence despite intermittent operation and varying ambient energy availability. It is agnostic as to how the nodes harvest energy (from solar, vibrations, RF, etc.) and as to whether they communicate using backscatter or radio communication.

The design of Find is based on the observation that the only way battery-free nodes can reliably avoid interleaving is to not wake up and become active immediately after reaching the minimum energy level required to do so. We refer to this as the greedy approach. Instead, Find delays each wake-up for a random time. A crucial question is how to choose this random delay to ensure fast and energy-efficient discovery.

To answer this question, we devise a model that captures the impact of key parameters, such as the charging time needed to reach the minimum energy level and the random delay, on the discovery latency (Section 3.2.1). Using this model, we then determine an optimized delay distribution that minimizes the discovery latency (Section 3.2.2). Finally, we describe how these considerations materialize in the practical design of the Find protocol and its runtime operation (Section 3.2.3).

3.2.1 Modeling Discovery Latency

Suppose that a node needs to charge for c slots until it reaches the minimum energy level required to be active for one slot. Let k_0 denote the first slot in which a node reaches the minimum energy level. Using Find, a node waits for a random delay x in units of slots before it wakes up and becomes active. We model x as a discrete random variable X with probability mass function (pmf) $p_X(x)$. During an active slot, a node fully depletes its energy storage. The probability that a node

becomes active for the first time in slot k is given by

$$p_{wk,0}(k) = p_X(k - k_0) \tag{3.1}$$

Afterward, a node needs to recharge for c slots before it can become active again. The time of the second wake-up is the sum of the time of the first wake-up, the charging time, and the second random delay. The same reasoning applies recursively to all future wake-up times. Because the random delay is independently chosen across all wake-ups, we can use a recursive convolution to determine the probability that a node wakes up for the n-th time in the k-th slot

$$p_{wk,n}(k) = (p_{wk,n-1} * p_X)(k - c) \tag{3.2}$$

By summing over $n \to \infty$ we obtain the probability that a node is active in slot k

$$p_a(k) = \sum_{n=0}^{\infty} p_{wk,n}(k) \tag{3.3}$$

To model discovery latency, we consider a fully connected network of N nodes (*i.e.*, a clique of size N). Using a suitable sequence of message exchanges in active slots (see Section 3.2.3), one of the $M = N(N-1)/2$ bi-directional links $i \leftrightarrow j$ is discovered if nodes i and j are active in the same slot while all other nodes in the network are inactive. Otherwise, a collision occurs and no link is discovered, a typical assumption in neighbor discovery protocols [67]. The probability that link $i \leftrightarrow j$ is discovered within k slots is the complement of the probability that the link is not discovered in slots $0, \ldots, k$:

$$c_{i \leftrightarrow j}(k) = 1 - \prod_{\kappa=0}^{k} \left(1 - p_{a,i}(\kappa) \cdot p_{a,j}(\kappa) \cdot \prod_{l \neq i,j} (1 - p_{a,l}(\kappa)) \right) \tag{3.4}$$

$c_{i \leftrightarrow j}(k)$ can be regarded as the cumulative distribution function (cdf)

of the discrete random variable describing the slot in which link $i \leftrightarrow j$ is discovered. With $p_{i\leftrightarrow j}(k)$ denoting the corresponding pmf, we compute the expected fraction of links discovered up to slot k by averaging $p_{i\leftrightarrow j}(k)$ over all M links

$$d(k) = \frac{1}{M} \sum_{i\leftrightarrow j} p_{i\leftrightarrow j}(k) \tag{3.5}$$

If the nodes' charging times are finite, $d(k)$ is a valid cdf, and we define the discovery latency as

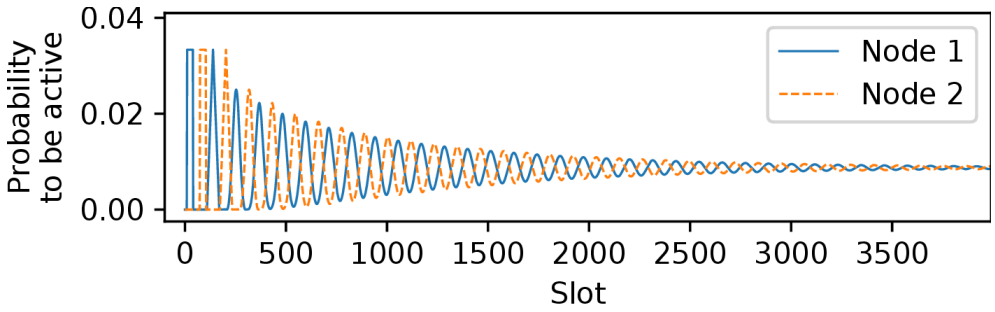
$$T_{nd} = \sum_{k=0}^{\infty} (1 - d(k)) \tag{3.6}$$

3.2.2 Optimized Delay Distribution

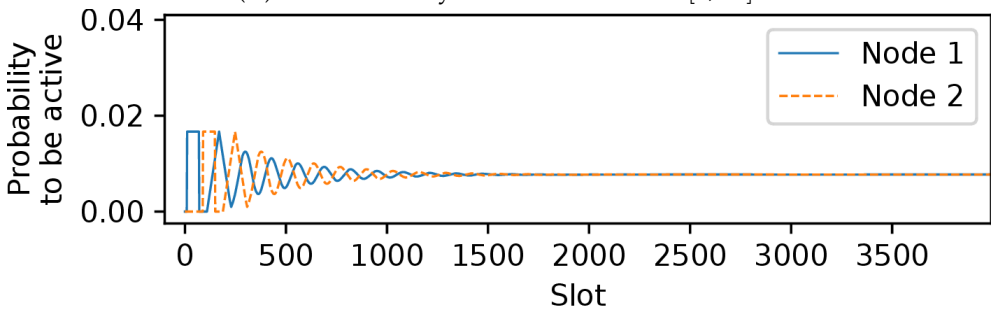
With the above model we are able to get a better understanding of how nodes should delay their wake-ups to help discovery.

Example. Suppose two nodes i and j with the same charging time of $c = 100$ slots, but different slots k_0 in which they reach the minimum energy level for the first time (*i.e.*, initial offset). Using (3.3) we plot in Figure 3.3a for both nodes the probability of being active in a slot when they pick random delays from the discrete uniform distribution $X \sim U[0, 30]$. We see that in the first thousand slots there is hardly any overlap in the activity of the nodes: Due to the initial offset, node i is likely active when node j is powered off, and vice versa. The probability of being active smears out over time and converges to an average duty cycle of $1/(c + \mathrm{E}[X]) \approx 0.0087$. Figure 3.3b plots the same when the two nodes pick random delays from $X \sim U[0, 60]$. Compared to Figure 3.3a we find that the probability of being active smears out sooner as nodes tend to choose more wide-spread delays. However, as nodes also tend to pick longer delays, they have a lower average duty cycle of 0.0077.

Figure 3.4 directly compares the two delay distributions by plotting the cdf of the slot in which nodes i and j discover each other according



(A) Random delay drawn from $X \sim U[0, 30]$.



(B) Random delay drawn from $X \sim U[0, 60]$.

FIGURE 3.3: Probability of being active in a slot for two nodes with identical charging times but an initial offset in their wake-ups. The more wide-spread the random delay, the faster nodes break up their interleaved wake-up pattern at the cost of a lower average duty cycle.

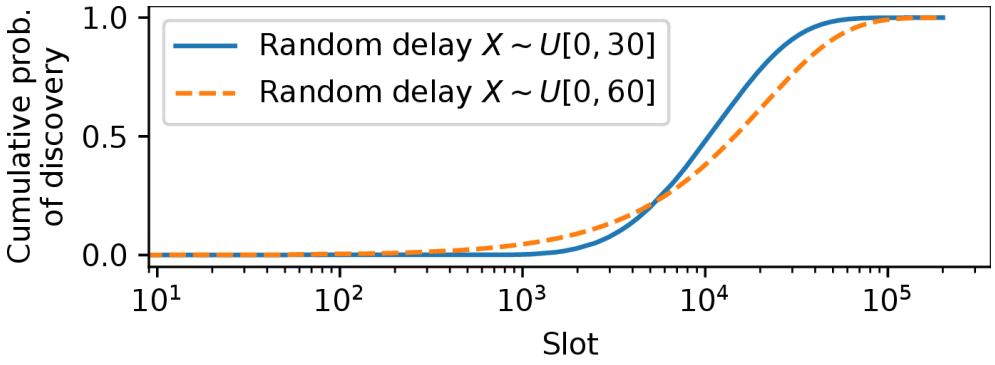


FIGURE 3.4: Cumulative distribution function of the slot in which two nodes discover each other, for the two delay distributions in Figure 3.3. A more wide-spread delay performs better initially, but leads to lower performance in the long run due to a lower average duty cycle.

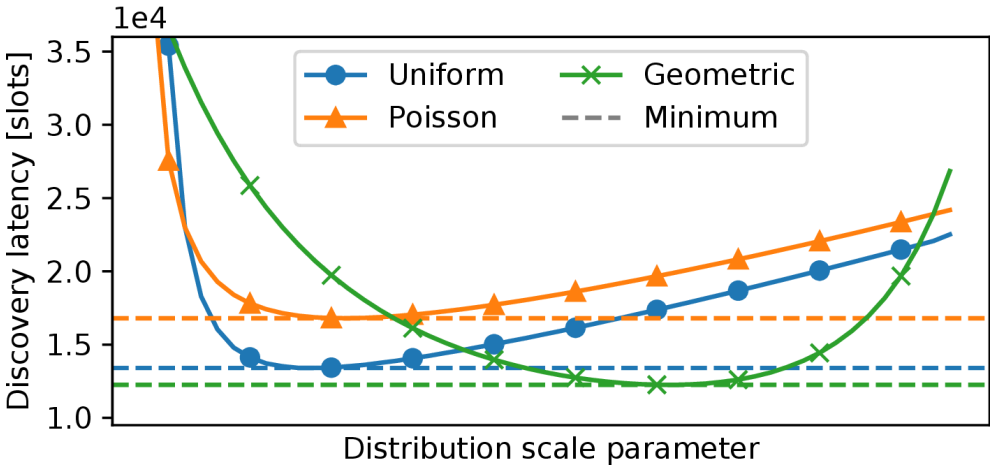


FIGURE 3.5: Discovery latency against scale parameter for three different probability distributions. The geometric distribution performs best as it yields delays with high randomness and low mean.

to (3.4). We observe that the more wide-spread delay induced by the second distribution $X \sim U[0, 60]$ initially provides a higher probability of discovery. In the long run, however, the higher average duty cycle of the first distribution $X \sim U[0, 30]$ leads to a higher probability of discovery.

Choosing a distribution. The above example suggests that a non-negative delay distribution with *high randomness and low mean* is preferable. Entropy is a commonly used measure of randomness. Maximizing the entropy of a general non-negative distribution with a given mean yields the exponential distribution [100]. Thus, in Find, we draw random delays from the geometric distribution, the discrete analogue of the exponential distribution, with scale parameter $1/r$ and pmf $(1 - r)^k r$ for $k \in \{0, 1, 2, \ldots\}$.

To confirm our reasoning, we compare the geometric distribution with other well-known distributions, namely the discrete uniform distribution and the Poisson distribution. We sweep the scale parameter of the three distributions and compute the discovery latency using (3.6) for the two-node case, where nodes i and j have equal charging times (25, 100, 500, or 1000 slots). We find that the geometric distribution

achieves the lowest discovery latency across all charging times. Figure 3.5 shows the resulting curves for a charging time of 100 slots. The differences in the minimum discovery latencies are relatively small. One reason for this is that, according to the central limit theorem, the probability that a node wakes up for the n-th time in slot k converges to a normal distribution for large n, irrespective of the underlying delay distribution.

Determining optimized distribution parameters. Having chosen a suitable delay distribution, we now turn to the problem of determining the scale parameter that minimizes the discovery latency. To formally state the optimization problem, we consider the worst case in terms of discovery latency: all N nodes have the same charging time c, and their initial wake-up times $k_{0,i}$ are all interleaved as in Figure 3.3, that is,

$$k_{0,i} = i \cdot \frac{c + 2\,\mathrm{E}[X]}{N} \tag{3.7}$$

where i is the node index and $\mathrm{E}[X]$ is the expected delay. For specific N and c, we minimize the discovery latency given by (3.6) and the initial offsets given by (3.7)

$$\min_r T_{nd}(N, c) \tag{3.8}$$

Numerical evaluation suggests that $T_{nd}(N, c)$ is convex (see Figure 3.5) and hence straightforward to optimize. We use Brent's method [17] to approximate the scale parameter $1/r^*$ that minimizes the discovery latency. The next section explains how we adapt the scale parameter at runtime on a real node.

3.2.3 Practical Protocol Design

The above analysis makes a number of simplifying assumptions that do not hold in practice. For example, the charging times are generally different across nodes and vary over time. A node typically only knows its own charging time c and is unaware of the total number N of nodes in the network.

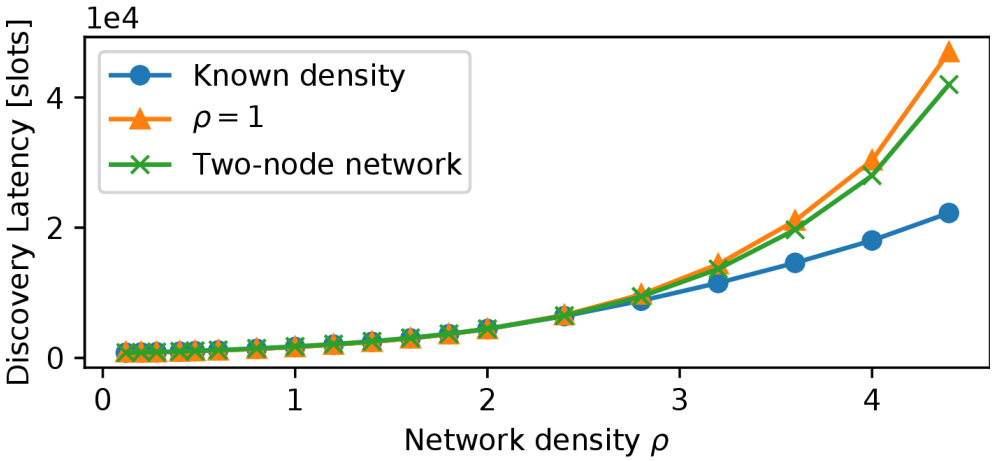


FIGURE 3.6: Discovery latency against network density ρ when optimizing for the known density, for a fixed density of $\rho = 1$, and for a two-node network. For $\rho \leq 2.5$, all approaches perform similarly.

Nevertheless, prior work has shown that neighboring nodes have similar energy availability because they harvest energy from the same ambient source(s) [44, 7]. Thus, *in the absence of any prior information*, a reasonable approach for a node is to assume that its neighbors harvest the same amount of energy and thus have the same charging time c like itself.

Moreover, we found that knowledge of the number of nodes N is often not required: optimizing for the case of a two-node network yields competitive performance across a wide range of network densities. In other words, in practice, it is often sufficient for a node to assume that it has only one neighbor (although over time it may discover that it has many more). To understand why, we plot in Figure 3.6 the discovery latency for a charging time of 25 slots when optimizing for (*i*) the known network density $\rho = N/c$, (*ii*) a fixed network density of $\rho = 1$, and (*iii*) a two-node network. We can see that for a network density of $\rho \leq 2.5$ the three approaches achieve almost the same performance. For realistic charging times, the network density rarely exceeds this threshold. For example, based on the charging times and beacon length in our real-world case study (see Section 3.6), a network density of $\rho = 2$ would require a network of around 4000 fully connected nodes.

Runtime operation. Prior to each wake-up, a Find node samples a geometric distribution to determine the random delay. A node dynamically adapts the scale parameter of the distribution to changes in its charging time, under the assumption that it has one neighbor with the same charging time, as explained above. To achieve an efficient runtime operation, we store a look-up table of optimized scale parameters in non-volatile memory and use inverse transform sampling to convert samples from a uniform pseudo-random number generator to the optimized, geometric distribution.

Frame structure. Taking inspiration from existing neighbor discovery protocols for battery-powered sensor nodes [37, 8], we adopt the frame structure shown in Figure 3.7. During each active slot, a node first transmits a beacon, then listens for potential beacons from neighboring nodes, and finally transmits another beacon at the end of the slot. The second beacon ensures that nodes can discover bi-directional links in one common active slot. Specifically, if the slot offset $\mathcal{T}$ between two nodes (see Figure 3.7) is uniformly distributed between $-T_{slot}/2$ and $T_{slot}/2$, where T_{slot} is the slot length, the probability that two nodes successfully discover each other's presence is

$$p = 1 - \frac{2 \cdot (T_{ta} + T_{tx})}{2 \cdot (T_{ta} + T_{tx}) + T_{rx}} \tag{3.9}$$

Here, as depicted in Figure 3.7, T_{tx}, T_{rx}, and T_{ta} denote the times needed to transmit a beacon, to listen for potential beacons, and to switch from receive to transmit mode (or vice versa). In order to maximize the success probability according to (3.9), Find keeps the beacon transmission time T_{tx} as short as possible to maximize the listening window T_{rx}.

3.3 Further Accelerating Neighbor Discovery

Find provides fast and energy-efficient neighbor discovery in battery-free networks. Nevertheless, if the ambient energy availability is low, discovery may still take a long time due to the low duty cycles. For example, according to our model, under dim indoor light conditions

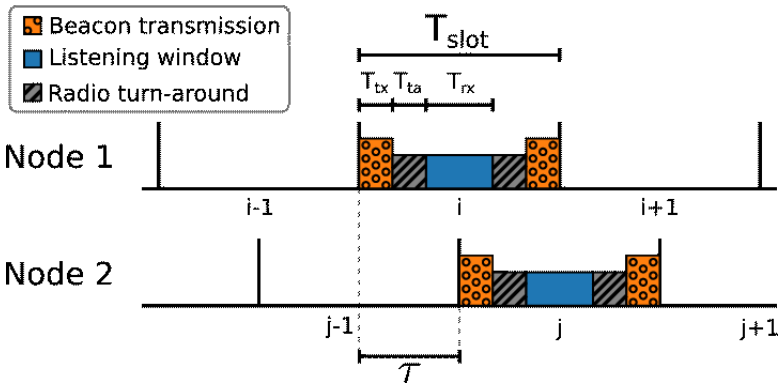


FIGURE 3.7: Find's frame structure specifying the sequence of beacon transmissions and the intermediate listening window during an active slot. Using our prototype implementation, nodes can successfully discover each other if the slot offset $\mathcal{T}$ is between 88 µs and 848 µs.

it takes on average 8 min until two of our prototype battery-free nodes (see Section 3.4) discover each other. Similar observations are to be expected when nodes harvest from weak RF signals or miniature vibrations [12]. The discovery latencies in those challenging energy environments can be prohibitively long for many applications.

This section introduces an approach that facilitates, according to our model, a 10× speed-up in the above-mentioned scenario, allowing two nodes to discover each other in 45 s on average instead of 8 min at an additional cost of only 5 µW. The underlying idea is that neighboring nodes harvest energy from the same ambient source(s) and may therefore have access to a common energy signal that can be used as a time reference. In combination with Find, nodes can exploit this common time reference to align their wake-ups, thereby increasing the chances that nodes are active in the same slot.

To assess the potential of this idea, we focus in this work on harvesting energy from indoor light. While this is a popular method for powering battery-free nodes due to the ubiquity of interior lamps, the energy density of indoor light is significantly lower than that of sunlight. As such, it represents both a challenging environment for battery-free neighbor discovery and a highly relevant setting for real applications. In the following, we provide answers to three key questions:

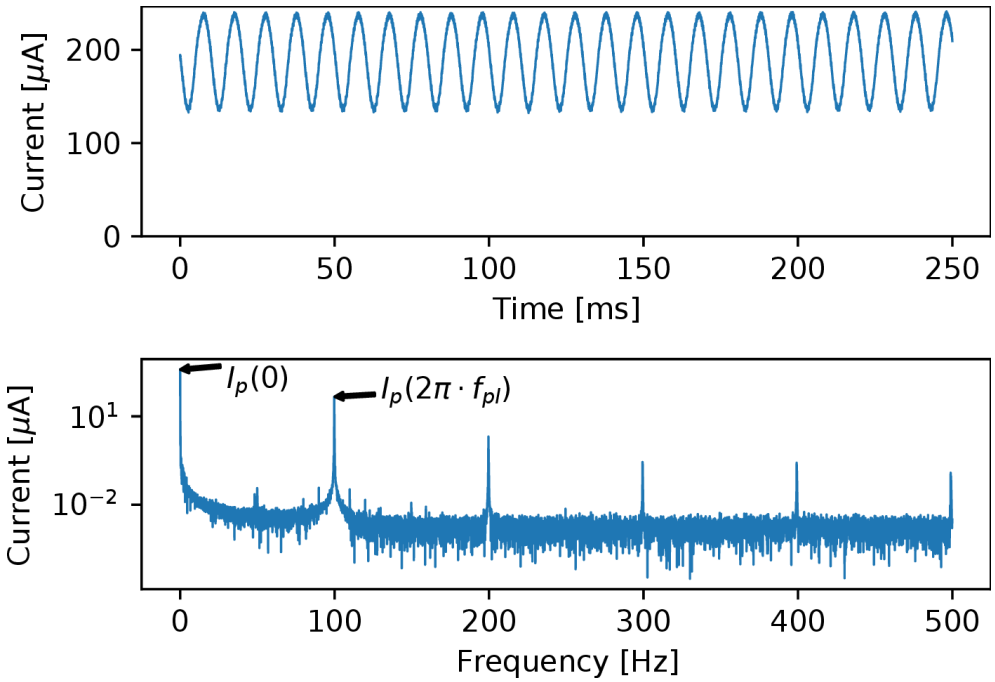


FIGURE 3.8: Time and frequency domain of solar panel current when harvesting energy from light emitted by a UP-PL30120-45W LED panel. The current varies with double the powerline frequency.

1. What common energy signal can nodes use? (Section 3.3.1)
2. How to efficiently extract a time reference? (Section 3.3.2)
3. How to exploit this for faster discovery? (Section 3.3.3)

3.3.1 Powerline Flicker in Solar Current

When harvesting energy from indoor light, we observed that the solar panel current varies with double the powerline frequency (50 or 60 Hz depending on the region). As an example, Figure 3.8 shows the solar panel current when harvesting energy from an LED panel light found in a typical office space.

Practically all indoor lamps are connected to mains power, which induces phase-synchronized brightness variations (*powerline flicker*) of the lamps through different effects. Despite their relatively high inertia, the alternating current through the filament of incandescent and

halogen lamps causes temperature and, as a result, brightness variations. A similar effect occurs in gas-discharge lamps like the ubiquitous fluorescent lamps, where the alternating current through the gas modulates the brightness. Due to the exponential relation between forward voltage and brightness, voltage-controlled LEDs are also sensitive to residual ripple of the rectified supply voltage. Because the power available from a solar panel is proportional to the brightness of the incident light, it also varies with double the powerline frequency, as visible in Figure 3.8.

To assess the potential of using powerline flicker as a common energy signal, we characterize the magnitude of powerline frequency induced fluctuations of the solar panel current for a wide variety of lamps. To compare lamps across diverse average brightness levels, we define the *flicker index FI* as the ratio of the amplitude of the powerline frequency component and the DC component of the solar panel current i_p

$$FI = \frac{I_p(2\pi \cdot f_{pl})}{I_p(0)} \tag{3.10}$$

where $I_p(\omega) = \mathcal{F}\{i_p(t)\}$ is the Fourier transform of the solar panel current and f_{pl} is the powerline frequency.

We attach an IXYS SM141K06L solar panel to a Shepherd node [44] and record 15 s of solar panel current at a sampling frequency of 100 kHz from each of the 19 lamps in Figure 3.9. For each trace we compute the flicker index using (3.10). The results in Figure 3.9 show that all lamps we tested exhibit varying levels of powerline flicker. We observe that all fluorescent and halogen lamps have a relatively large flicker index. The results for the tested LED lamps are more ambiguous. We suspect that highly integrated, bulb-shaped LED lamps tend to have high-quality current-controlled drivers with little flicker, whereas commercial panel-style LED lamps often rely on voltage-controlled drivers with significant levels of flicker.

We conclude that most types of lamps exhibit significant powerline flicker, which makes this an attractive common energy signal. Next,

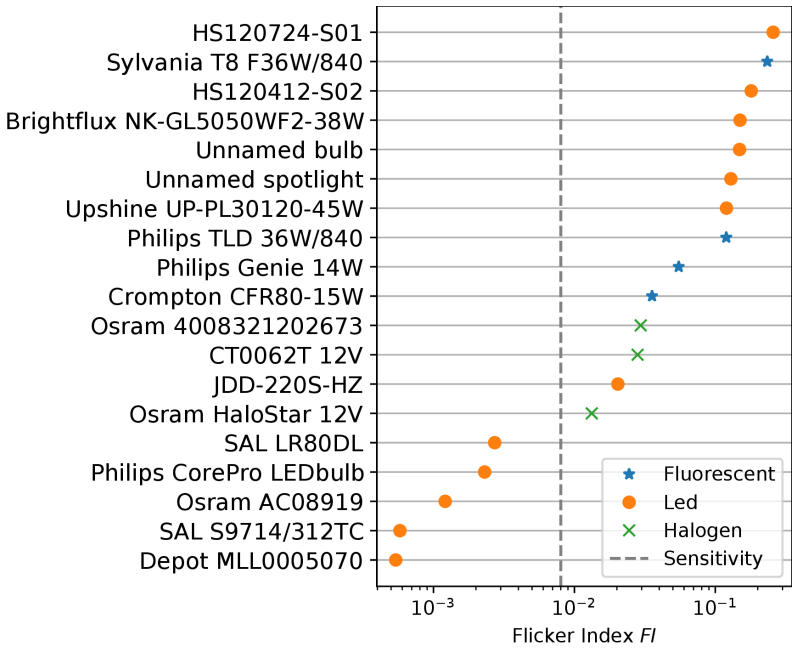


FIGURE 3.9: Flicker index for 19 tested lamps. The gray line marks the sensitivity of our Flync prototype. The proposed circuit works with all fluorescent and halogen lamps and the majority of LED lamps.

we present our design of Flync, a hardware/software solution that extracts a frequency- and phase-synchronized clock signal from this common energy signal on distributed battery-free nodes. The dashed line in Figure 3.9 is the measured sensitivity (see Section 3.5.2) of our Flync prototype, showing that the proposed design works with all fluorescent and halogen lamps and the majority of tested LED lamps.

3.3.2 Extracting a Clock from Solar Current

To be viable, Flync needs to provide a stable clock signal while keeping the required energy costs as low as possible.

Hardware. We propose the circuit shown in Figure 3.10, which converts the modulated current signal from the solar panel into a digital clock signal that can be connected to a GPIO pin of a MCU. The current through shunt resistor R_S causes a voltage drop that is filtered with a narrow-band bandpass filter to extract and amplify the powerline frequency component. We tune the band-pass filter to a gain of 36 dB at

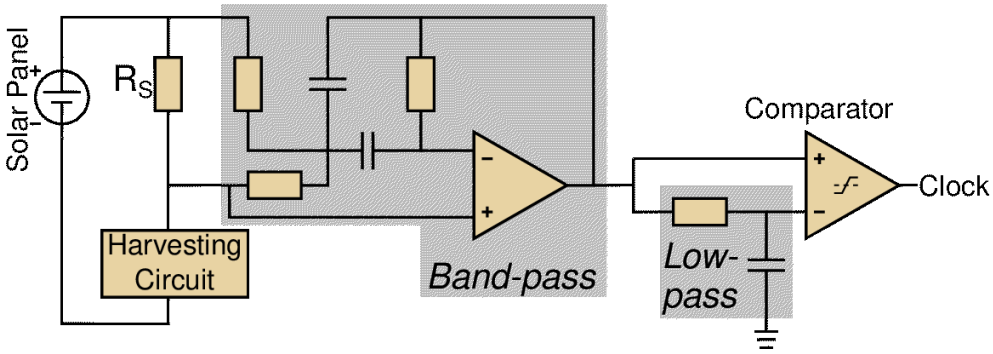


FIGURE 3.10: Flync circuit to extract a clock signal from the powerline-induced solar panel current variations (see Figure 3.8 for an example).

a center frequency of exactly double the powerline frequency, taking into account the limited gain-bandwidth product of the low-power operational amplifier. The resulting signal is connected to a comparator directly and through a low-pass filter to convert it into a digital signal.

The TI TLV521 operational amplifier used in the band-pass filter has a typical current draw of 350 nA, and the TLV7031 comparator has a typical current draw of 315 nA. Including the losses over the 300 Ω shunt resistor, the Flync circuit draws a total of around 5 µW under typical harvesting conditions. This is orders of magnitude lower than the power draw of related approaches, using a light sensor and an ADC (5.394 mW [80]) or an antenna to extract the signal from powerline radiation (300 µW [110]).

Software. To achieve a stable clock signal, we use a phase-locked loop (PLL) in combination with a proportional integral derivative (PID) controller to synchronize the MCU's real-time clock (RTC) to the powerline frequency signal extracted with our proposed circuit. In Section 3.4.2, we describe our software implementation of Flync in more detail.

3.3.3 Exploiting the Clock for Faster Discovery

Using Flync, neighboring battery-free nodes have access to a common clock. Nodes can use the phase information of this clock to implicitly

agree on times at which they potentially become active. For the powerline flicker, this could be the rising edges of the solar panel current (see Figure 3.8).

When using Find without Flync, we set the slot length to the duration of a node's active period. When using Find with Flync, we increase the slot length to $1/(2 \cdot f_{pl})$ and let nodes only become active at the beginning of a slot. This increases the probability that nodes become active in the same slot. For example, consider two nodes that randomly and uniformly wake up once within a 1 s time window. Using a slot length of 1 ms, the probability that both nodes wake up in the same slot is 1/1000. With a slot length of 10 ms, this probability is 10× higher, which speeds up the neighbor discovery process.

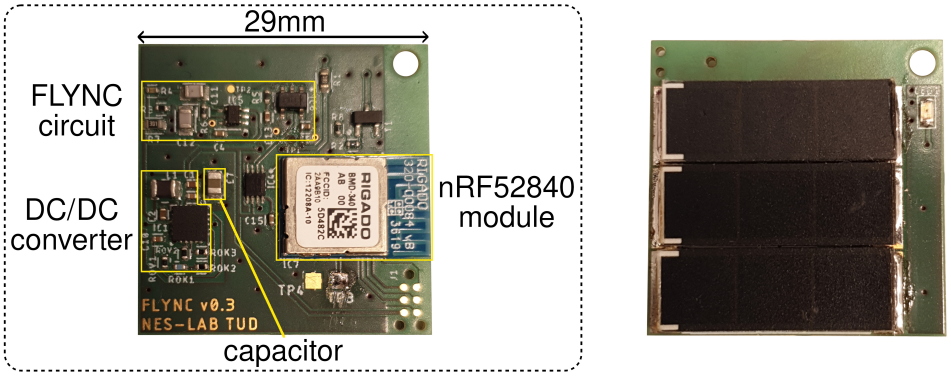
Flync exploits the well-behaved, widely available powerline flicker as sychronization source, but the concept applies to any phase-synchronized signal available on different nodes. Because the benefit in terms of a shorter discovery latency stems from increasing the effective slot length, the signal's period must be longer than the duration of a node's active period. The lower the frequency, the longer the slot length and the greater the potential benefit. If the period is longer than the charging time of a node, it can be divided down to avoid nodes wasting energy while waiting for the next slot.

3.4 Prototype Implementation

This section describes the hardware and software components of our prototype implementation.

3.4.1 Hardware

We design a low-power battery-free node that integrates the circuit from Figure 3.10. The node is based on a capable Nordic Semiconductor nRF52840 MCU, which features a 64 MHz ARM Cortex-M4F and a 2.4 GHz radio with support for Bluetooth 5.2 and IEEE 802.15.4. The node harvests energy using three 23 mm × 8 mm IXYS KXOB25-05X3F



(A) Front (B) Back

FIGURE 3.11: Prototype battery-free node based on the nRF52840 MCU. Solar panels on the back charge a tiny capacitor that powers the node.

solar panels. A TI BQ25505 DC-DC boost converter steps up the voltage of the solar panels and charges a 2 mm × 1.25 mm × 1.25 mm 47 µF multilayer ceramic capacitor (MLCC). However, due to DC bias, the capacitor has only an effective capacitance of around 17 µF at 3.3 V. The BQ25505 implements a maximum power point tracking (MPPT) mechanism that aims to operate the solar panels close to their optimal voltage of around 80 % of the panels' open-circuit voltage. The MPPT circuit obtains a new reference voltage every 16 s by disabling the charger for 256 ms and sampling the panels' open-circuit voltage. Once the capacitor voltage reaches a hardware-programmable threshold of 3.3 V, the BQ25505 sets one of its pins high. This pin is connected to a TI TS5A23166 analog switch that connects the MCU to the capacitor-buffered supply voltage.

The two-layer PCB shown in Figure 3.11 measures 29 mm × 29 mm. The total cost of all components is $13.89, including $8.11 for the relatively expensive, highly integrated nRF52840 module. Comparing our design to recently proposed battery-free platforms with similar capabilities in Table 3.1, we see that our prototype is indeed one of the first truly battery-free nodes in the sense that the energy storage is negligible in terms of cost, size, and environmental impact: The ceramic capacitor does not contain problematic materials, costs $0.024, and takes up only 0.3 % of the PCB area.

Platform	Year	Capacitor	Communication
Pible [42]	2018	220 mF super-cap	BLE
luxBeacon [62]	2019	1.5 F super-cap	BLE
Sigrist et al. [120]	2020	520 µF MLCC	BLE
Botoks [27]	2020	100 µF MLCC	868 MHz
This work	2021	47 µF MLCC	BLE PHY

TABLE 3.1: Our battery-free prototype node has a sustainable ceramic capacitor that is significantly smaller and cheaper than the energy storage of other recently proposed battery-free platforms.

3.4.2 Software

Next, we describe our implementation of an efficient runtime for battery-free nodes. We also detail the PLL implementation of Flync and key configuration parameters of Find.

Efficient runtime. Many existing battery-free runtimes discharge the capacitor until the voltage drops below the minimum and the MCU is powered off [27, 54]. To avoid the high energy costs of frequent hardware resets, we implement a different approach that we call *soft intermittency*. During charging, the MCU enters the lowest possible sleep mode, periodically waking up to sample the capacitor voltage with the built-in ADC. In this mode, we measure a total average power draw of 15 µW, including the power for the Flync circuitry and software processing. When the capacitor voltage reaches a software-defined turn-on threshold, the node arms the power-fail comparator, a dedicated peripheral that raises an interrupt when the capacitor voltage drops below a software-defined turn-off threshold. Then the node executes protocol and application code until it is notified by the power-fail comparator upon which it immediately transitions to deep sleep, drastically reducing its power draw until it has again buffered enough energy. While this soft intermittency approach cannot prevent hard resets when there is no energy input for several hundreds of milliseconds, it greatly increases the average efficiency without using additional comparators and switches.

Flync PLL. The comparator at the output of the circuit in Figure 3.10 has a relatively small hysteresis, occasionally causing flickering at signal transitions. Furthermore, while MPPT obtains a new reference value, the harvesting current approaches zero, causing the clock signal to pause for hundreds of milliseconds. To provide a stable clock signal despite these disturbances, we implement a PLL that synchronizes the MCU's RTC to the signal extracted with the Flync circuit. We configure the GPIO peripheral to generate an interrupt on a rising edge at the GPIO pin connected to the output of the comparator of the circuit. After a reset, we wait for the first GPIO interrupt. Upon this interrupt, we set up an RTC interrupt to reset the RTC counter after the nominal powerline frequency interval. Ideally, all following GPIO interrupts should coincide with that RTC interrupt. Thus, the counter value at the time of the GPIO interrupt can be interpreted as phase deviation between the external clock signal and the local timer. We implement a control loop to continuously adjust the timer period in order to minimize the phase deviation. In this way, we obtain a highly stable interrupt that is phase-synchronized with the variations of the solar panel current and works even during the MPPT sampling or other disruptions.

Find settings. Each beacon in Find's frame structure shown in Figure 3.7 consists of 2 B preamble, 3 B base address, 6 B payload, and 1 B cyclic redundancy check (CRC). When using the 2 Mbit BLE mode of the radio, this corresponds to a beacon transmission time of $T_{tx} = 48\,\mu s$. With 17 µF of capacitance, the time required to start the high-frequency oscillator, and a turn-around time of $T_{ta} = 40\,\mu s$, we can afford a maximum listening window of $T_{rx} = 800\,\mu s$. As a result, two nodes can successfully detect each other if they wake up with an offset $\mathcal{T}$ between 88 µs and 848 µs (see Figure 3.7).

3.4.3 Example Real-world Trace

Figure 3.12 shows capacitor voltage and activities over time while one of our prototype nodes runs Find. We see that the node charges its capacitor until reaching the turn-on threshold of 3.3 V. It wakes up and

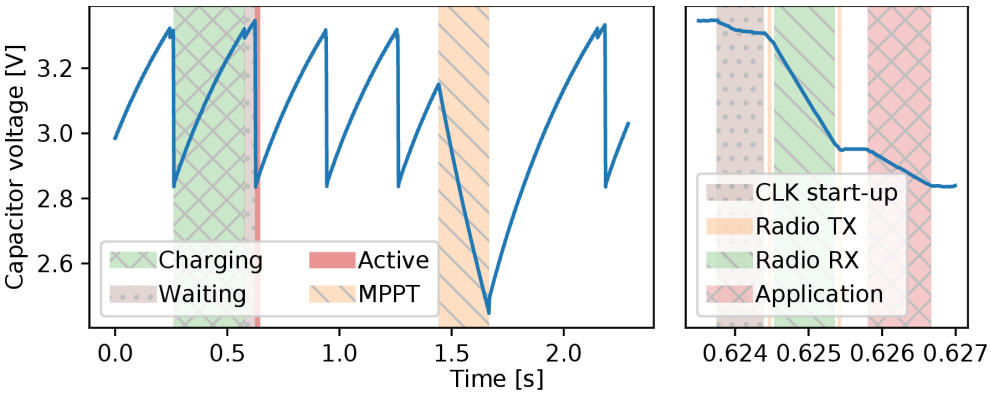


FIGURE 3.12: Example trace from a prototype node running Find.

samples a random delay from Find's optimized distribution. The necessary computations cause a noticeable drop in the capacitor voltage when transitioning from charging to waiting. After the random delay, the node becomes active and quickly drains its capacitor below the turn-off threshold of 2.8 V. The overview on the left side of Figure 3.12 also shows how the capacitor discharges during MPPT at around 1.5 s. The detailed view on the right side shows the individual stages while the node is active. We see that the node first starts the high-frequency clock required to run the radio. Then it sends the first beacon and starts to listen for potential beacons from other nodes. After listening for 800 µs, the node sends the trailing beacon. The remaining energy in the capacitor is assigned to the application that can run until the capacitor voltage hits the turn-off threshold.

3.5 Evaluation

We manufacture six prototype battery-free nodes to evaluate Find and Flync. We first look at their effectiveness in terms of discovery latency, followed by a detailed characterization of Flync's robustness and performance. Section 3.6 reports on the results of a contact tracing case study based on our techniques.

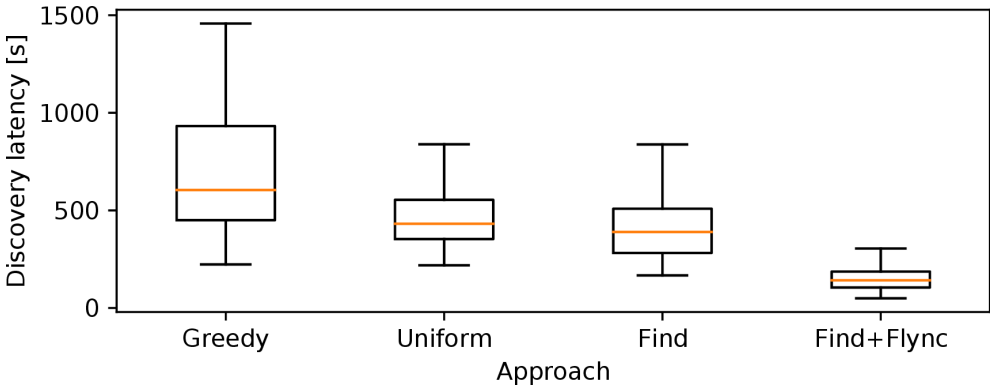


FIGURE 3.13: Discovery latency of four different approaches in a network of 6 battery-free nodes. Our techniques outperform the comparison approaches by up to 4.3× (median) and 34.4× (99th percentile).

3.5.1 Neighbor Discovery Performance

To fairly compare the neighbor discovery performance of our techniques against baseline approaches, we conduct experiments under controlled conditions. Section 3.6 reports on results when using Find and Flync in uncontrolled environments.

Setup. All experiments are conducted in a darkened room with a controllable light source. We place six prototype nodes next to each other on a flat surface. The nodes are programmed to output the ID of any discovered node over UART, while a logic analyzer logs the output of every node. For each run, we let nodes wake up with a random initial delay, and consider the measured time until all 15 bi-directional links are discovered as the discovery latency. We compare Find and Find + Flync with a *greedy* approach, where nodes become active as soon as their capacitor voltage reaches the turn-on threshold, and a *uniform* approach, where nodes randomly delay their wake-ups by a uniformly distributed time. Overall, the measurement campaign took more than 4 days, in which we performed between 48 and 128 runs for each of the four approaches.

Results. Figure 3.13 shows the measured discovery latency for each approach, including the median, the 25th and 75th percentiles, and

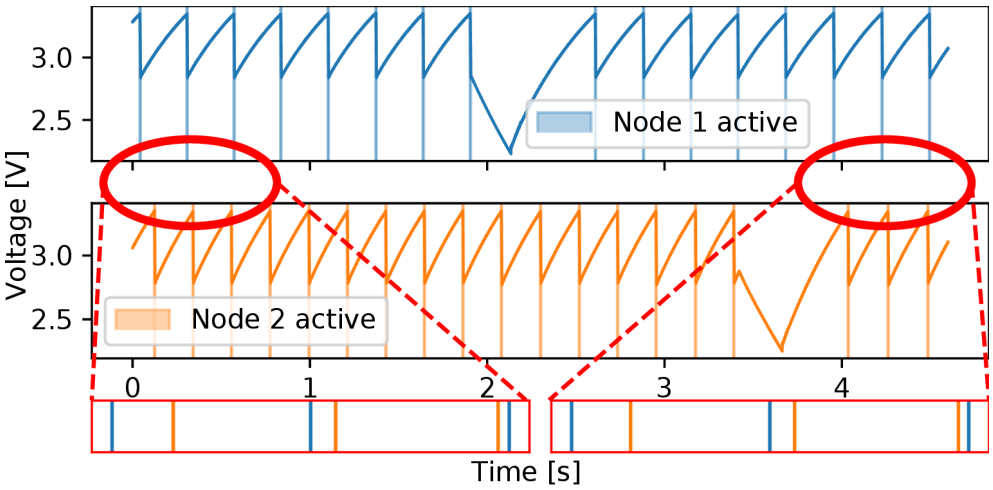


FIGURE 3.14: Interleaved activity phases of two nodes when using the *greedy* approach. The zoomed in plots on the bottom show that, despite the disturbances caused by MPPT, the two nodes repeatedly wake up with the same pattern, preventing successful discovery.

the 1.5× of the interquartile range. Clearly, the greedy approach performs worst. This is mainly because of interleaved activity phases of the nodes, as visible from the trace in Figure 3.14. If we zoom in on the first three and the last three wake-ups in the trace, we notice that nodes repeatedly wake up with the same pattern that prevents discovery despite different charging times and MPPT intervals. In Figure 3.15, instead, we see that when nodes use Find to randomly delay each wake-up, they are more likely to be active at the same time. For instance, at about 4.5 s, the nodes wake up with an offset of less than 848 µs and are therefore able to successfully exchange beacons as shown in the detailed plot on the right side of Figure 3.15. This explains the significant reduction in median discovery latency from 604 s with greedy to 390 s with Find, as visible in Figure 3.13. We also see that Find's optimized delay distribution performs slightly better than the uniform approach (median of 431 s), which matches the magnitude of improvement predicted by our model (see Figure 3.5). Find + Flync achieves the lowest median discovery latency of 142 s, which corresponds to an overall improvement of 4.3× (median) and 34.4× (99th percentile) compared with greedy.

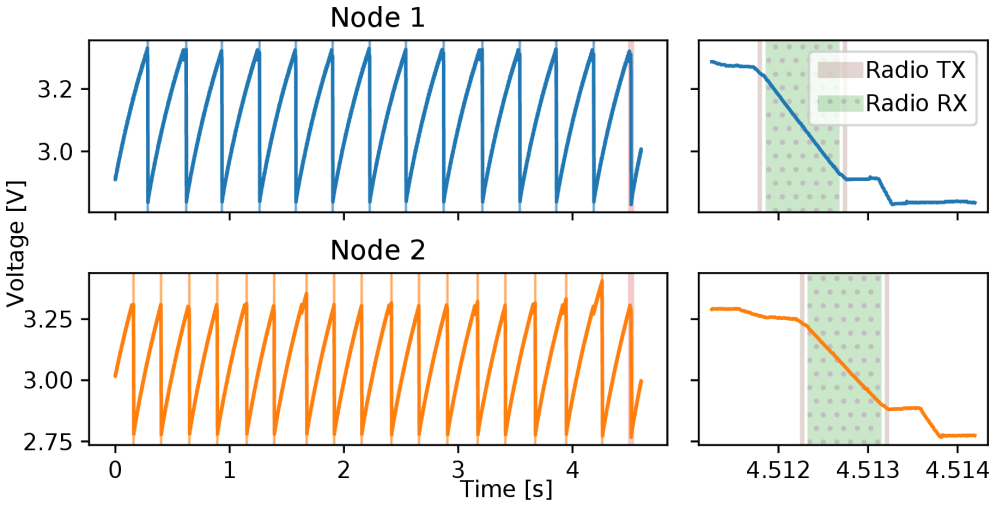


FIGURE 3.15: Using Find, nodes prevent *interleaving* by delaying each wake-up by a small random time, enabling quick discovery.

3.5.2 Flync Sensitivity

To extract a clock signal, the Flync circuit requires a minimum magnitude of the powerline frequency component in the solar panel current. We empirically determine the corresponding minimum flicker index for our hardware prototype.

Method. The magnitude of the powerline frequency component is proportional to the DC component and decreases with smaller panel size and increasing distance from the light source. We define the worst-case minimum flicker index as the flicker index sufficient to extract a clock signal even at the lowest possible harvesting current. The latter is defined by the minimum power requirements of our prototype when running Find, the panel voltage, and the corresponding efficiency of the DC-DC converter. Our solar panels have a typical panel voltage of 1 V at the maximum power point. At this voltage, our DC-DC converter has an efficiency of 80 %. Thus, the minimum harvesting current to cover the power requirements of our prototype of about 37.5 µW is 50 µA.

We use a Keithley 2600B sourcemeter to generate a current signal with a DC offset of 50 µA while sweeping the amplitude of the 100 Hz AC

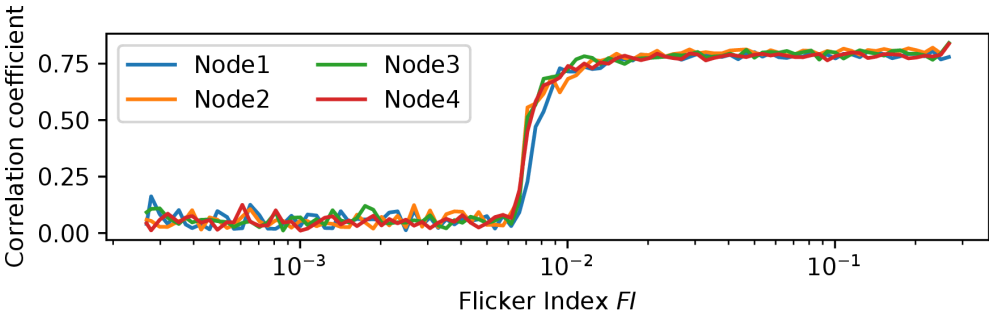


FIGURE 3.16: For a flicker index $\geq$0.008 Flync provides a stable clock.

component. The current is fed to the input of our prototype that is usually connected to the solar panel. By limiting the voltage at the output of the sourcemeter to 1.25 V, the MPPT circuit regulates the input to around 1 V. For every setting of the AC amplitude, we record 5 s of clock signal with a mixed-signal oscilloscope. To quantify the quality of the clock signal, we compute the correlation coefficient between the signal and a phase-aligned 100 Hz reference. We repeat these measurements for four of our prototype nodes.

Results. The results in Figure 3.16 show that there is a distinct threshold at around $FI = 0.008$ beyond which all nodes begin to output a clean clock signal. Comparing this with Figure 3.9, we conclude that, with the exception of 5 LED lamps, our prototype works with the vast majority of the lamps we tested.

3.5.3 Flync Robustness

We now assess the robustness of Flync when a node changes its position and orientation relative to the light source, when the solar panels of a node are temporarily covered, and when electrical loads are temporarily connected to the same power strip. To this end, we experiment with two nodes powered by a desk lamp and connect them to an oscilloscope. We quantify robustness by measuring the time difference between clock edges on the two nodes. As a benchmark, we note that our implementation can tolerate a time difference of up to 848 µs.

Mobility. We keep one node static and attach the other one to the wrist

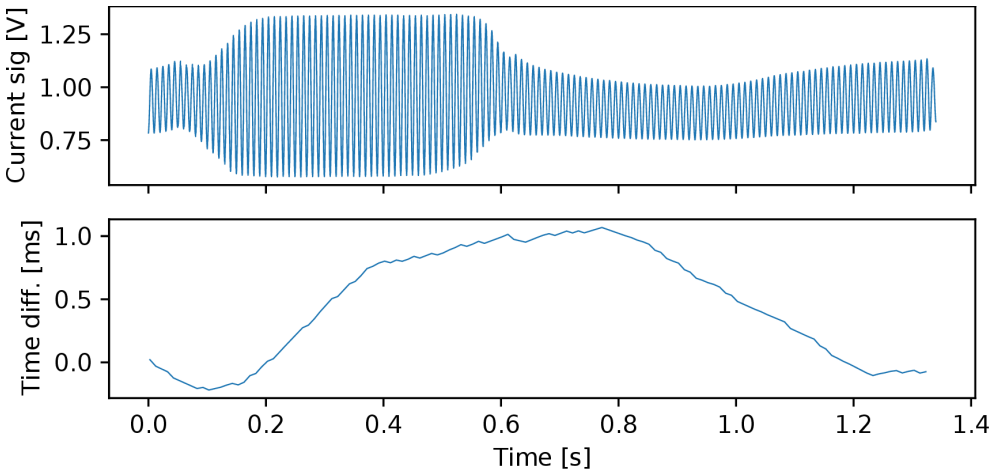


FIGURE 3.17: Current signal and time difference between two nodes while one node changes its distance and angle to the light source.

of a person. The person waves, changing distance and angle between the node's solar panels and the light source.

Figure 3.17 shows a period where the node moves closer and farther away from the lamp. The changes in the amplitude of the current signal affect the time difference between the nodes. The comparator that thresholds the sine wave uses a low-pass filter that reacts slowly to changes in the average amplitude. As a result, the clock signal deteriorates temporarily, causing an increased time difference of up to 1 ms. However, after a short while, the time difference recovers to previous levels.

Shadowing. To investigate the impact of shadowing, we put both nodes on a table and temporarily cover one of them by slowly moving a hand between the lamp and the node.

Figure 3.18 shows that the time difference increases after covering the panel as the PLL loses its reference signal. However, without significant energy input, the node does not reach the turn-on threshold, which renders communication infeasible anyhow. As soon as the panel is uncovered, the node quickly charges up again and, after less than a second, the clock returns with a small time difference.

Electrical loads. We repeatedly switch on and off a drilling machine

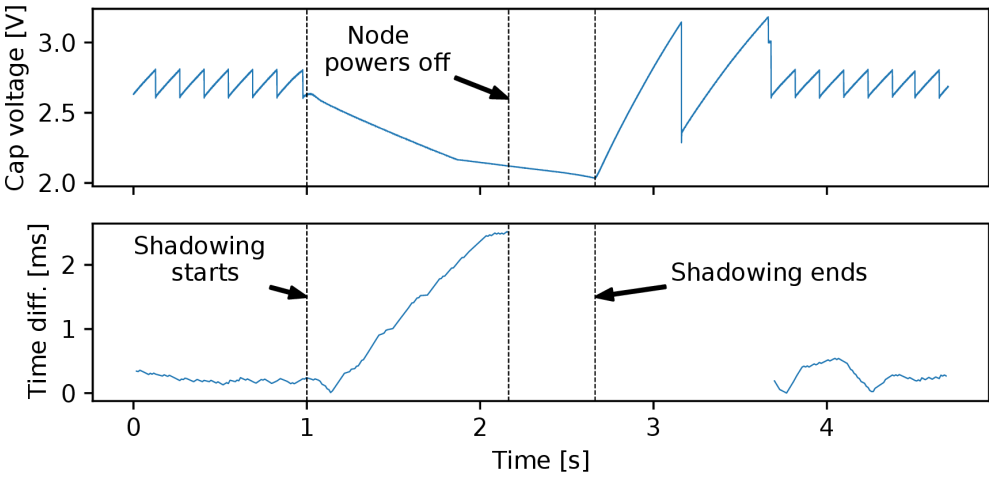


FIGURE 3.18: Capacitor voltage and time difference between two prototype nodes while temporarily covering the solar panel of one node.

and a vacuum cleaner connected to the same power strip as the lamp. We do not observe any noticeable effect of the loads on the time difference between the two nodes.

3.5.4 Flync Jitter

In a final set of experiments, we look at the time difference between the clock signals of different nodes when these are: (*i*) powered by a single light source, (*ii*) placed in different rooms, and (*iii*) powered by different types of light sources.

Testbed. For these experiments, we built a distributed testbed of observer nodes. The observer nodes are accurately time-synchronized to within 479 ns, and record the clock signals of the attached prototype nodes with a resolution of 62.5 ns.

Single light source. We place six of our prototype nodes in the same room with a single halogen lamp. The experiments are conducted during the day, and the nodes receive a mixture of natural sunlight and artificial light from the lamp. Using our testbed, we record the clock edges of all six nodes for 1 h.

Figure 3.19 shows the pairwise time difference between nodes. Because the phase offset resulting from propagation delays of light is

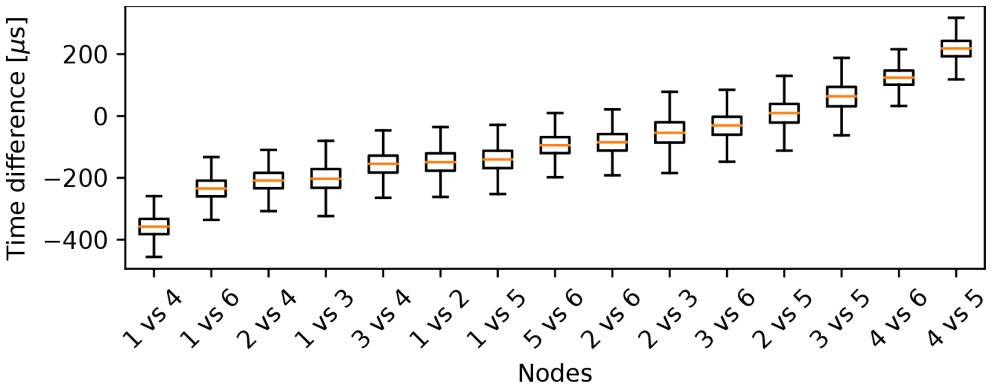


FIGURE 3.19: Pairwise time difference between clock edges on different prototype nodes when these are powered by a single light source.

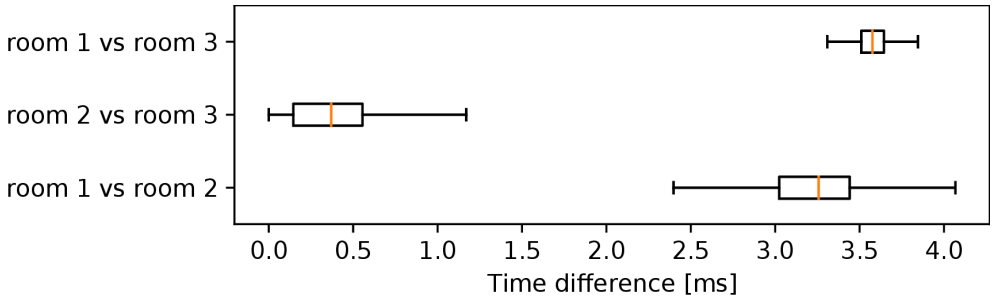


FIGURE 3.20: Pairwise time difference between clock edges on prototype nodes placed in different rooms with the same type of lamp.

negligible, the jitter must be introduced on each node. For example, a slight difference in the offset voltage of the comparator can lead to a significant mean difference of the resulting clock signal. Nevertheless, with 95 % of the more than five million recorded pairs below 244 µs, the jitter is well below the 848 µs tolerated by our Find implementation.

Different rooms. We conduct experiments in three rooms of an office building equipped with fluorescent tubes. The rooms are located on a long hallway with a distance of around 15 m between the middle room and the other two. We place two nodes in each room, and record with our testbed for 4 h while the nodes receive light from the tubes as well as sunlight.

Figure 3.20 shows that there is a small offset between rooms 2 and 3 with 95 % of the recorded values being smaller than 700 µs. The offsets

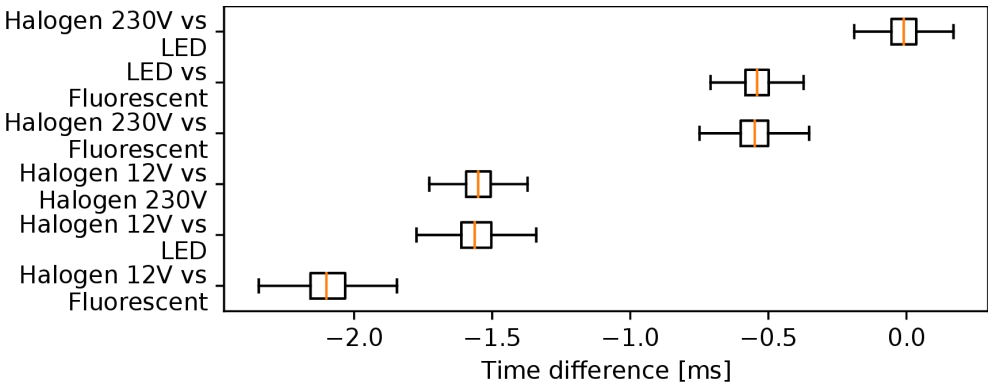


FIGURE 3.21: Pairwise time difference between clock edges on different prototype nodes when these are powered by different types of lamps.

between rooms 1 and 2 and rooms 1 and 3 are centered around 3.3 ms. While residential homes are often connected to a single phase, larger apartment blocks or commercial buildings are typically fed by three-phase power. Apparently, the lights in room 1 are connected to a different phase than the lights in rooms 2 and 3, leading to a 60° phase and 3.3 ms time shift between the light intensity variations. Thus, when nodes need to discover neighbors across rooms with lights potentially connected to different power phases, they must be able to become active not only at the edge of their own Flync clock signal, but also with a 60° phase shift.

Different types of light sources. We plug an LED, a fluorescent, and two halogen lamps into the same power strip. We place one node under each lamp so that it only receives light from this lamp, and record for 30 min with our testbed.

Figure 3.21 reveals large offsets between the clocks of nodes powered by different types of lamps. These offsets are due to varying phase shifts between the powerline voltage and the brightness variations of the lamp. For example, although the current through an incandescent lamp is in phase with the supply voltage, the filament may take some time to heat up and cool down, leading to the observed phase shift. Other types of lamps contain inductors or capacitive elements, a switching power supply, or an electronic ballast that cause different phase shifts. This shows that Flync does not work out of the box

(A) Node on shirt. (B) Setup of experiment in an open-air pub.

FIGURE 3.22: Battery-free contact tracing.

when different nodes are powered by different types of lamps. The static phase shifts would need to be measured during deployment or learned at runtime. On the other hand, Flync may not work reliably when individual nodes receive a mixture of light from different types of lamps. The results from the previous experiments (see Figs. 3.19 and 3.20) show that Flync works well when nodes receive a mixture of natural sunlight and artificial light from the same type of lamp.

3.6 Case Study: Contact Tracing

Automatic contact tracing is important to contain the spread of infectious diseases (*e.g.*, SARS-CoV2) in a scalable manner. It allows to quickly identify contacts of an infected person and to quarantine potentially infected individuals before they become contagious. To assess the potential of our proposed designs for real-world battery-free applications, we conduct a contact tracing case study with our prototype nodes.

Setup. We attach six nodes to the shirts of human participants, as shown in Figure 3.22a. The nodes run the Find protocol, logging the timestamp and ID of each discovered node to non-volatile memory. As we are only interested in relatively close contacts that would allow a virus to transmit from one person to another, we set the transmission power of the beacons to $-16\,\text{dBm}$. We run experiments indoors and outdoors, as detailed below. After each run, we dump the content of the non-volatile memory of each node to a computer for analysis.

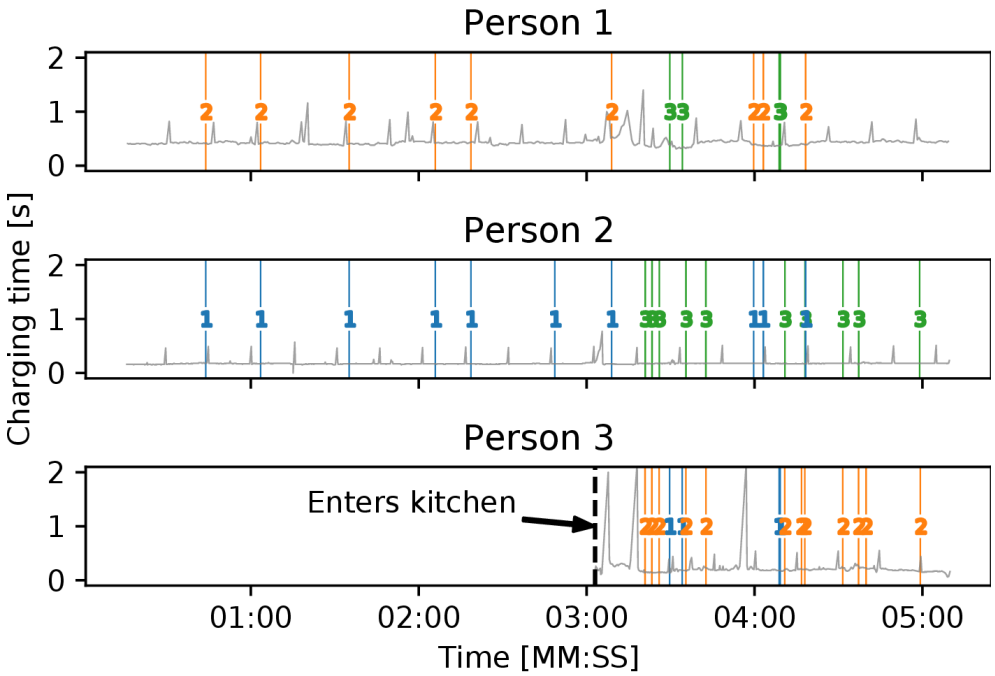


FIGURE 3.23: Charging times and rendezvous in coffee kitchen experiment. Vertical markers show rendezvous with the respective person.

Indoor experiment: coffee kitchen. Two persons sit at a table in a small coffee kitchen, roughly 1.5 m apart from each other. After 3 min a third person enters the kitchen and prepares a coffee for 2 min. The kitchen is equipped with fluorescent lamps, and we use Flync together with Find.

Figure 3.23 plots the charging times and recorded rendezvous of the three nodes over time. We see a total of 49 received beacons. All contacts are logged successfully with low latency, despite the relatively long charging times of hundreds of milliseconds. Specifically, the first contact between persons 1 and 2 is detected after 43.9 s. When person 3 enters the kitchen, it takes 26.6 s and 17.9 s until the contacts with persons 1 and 2 are detected, respectively. Overall, the median time between rendezvous of the same two nodes is 7.5 s.

Outdoor experiment: open-air pub. Three pairs of persons sit at opposite sides of three tables (see Figure 3.22b). Two tables are next to each other; the third table is at a distance of around 4.5 m. We perform

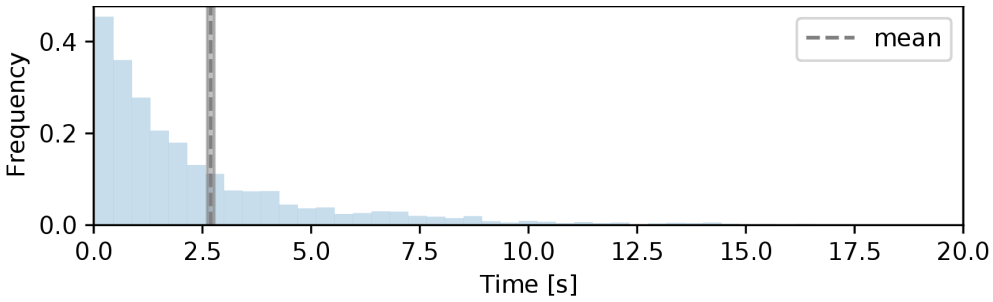


FIGURE 3.24: Histogram of the time difference between rendezvous of the same two nodes in the open-air pub experiment.

the experiments in the morning of a slightly overcast day at an open-air pub without direct sunlight. Receiving only natural sunlight, the nodes do not make use of Flync. We conduct three consecutive 15 min runs.

We measure a total of 4426 received beacons. All contacts between persons on the same table are successfully recorded. More importantly, contacts between persons on different tables in close vicinity are also reliably detected. Due to the low transmit power, we do not see any rendezvous between the first two tables and the third remote table, which is expected and in fact desirable because we only want to trace contacts that are associated with an actual risk of virus transmission. Figure 3.24 shows the histogram of the time between consecutive rendezvous between the same two nodes. As expected, the time between rendezvous is approximately exponentially distributed, and the mean is estimated between 2.61 s and 2.78 s with 95 % confidence. This means, under the given conditions, we are able to detect contacts with a resolution of around 2.67 s, allowing for fine-grained contact tracing.

Summary. The results from our contact tracing case study show that Find and Flync are also effective under uncontrolled real-world conditions. Outdoors, energy availability is high and therefore Find alone enables fast rendezvous and fine-grained contact tracing. Indoors, Flync can compensate for the significantly lower energy density of interior light, providing decent performance even under these challenging conditions.

3.7 Discussion

We have presented two novel techniques that enable for the first time efficient device-to-device communication in the face of intermittency. By introducing random delays, Find breaks interleaved activity patterns of battery-free devices to discover each other faster and more efficiently. By tapping into the powerline-induced flicker of state-of-the-art lamps, Flync phase-synchronizes devices that harvest energy from indoor light. While we have exploited Flync to further speed up discovery in battery-free networks, Flync is useful for other purposes and also applicable to battery-supported devices.

Recent work tackles the intermittency problem on individual battery-free devices in terms of, for example, computing and time keeping [56, 27, 88, 16]. We instead focus on communication between battery-free devices that operate intermittently. Like prior work, our techniques are relevant if intermittency makes traditional approaches inefficient or unreliable. To understand the scope of our work, we discuss intermittency and relevant impact factors below. Afterward, we discuss the influence of built-in randomness on our proposed techniques.

3.7.1 When Does Intermittency Occur?

A battery-free device goes through periods with low power requirements (*e.g.*, system-off and sleep modes) and high power requirements (*e.g.*, sensing, processing, and communication). Since the instantaneous power available from a harvester is often insufficient to support a battery-free device during periods with high power requirements, some form of energy storage is needed that buffers energy when the device is inactive to support a high-power workload for a short period of time.

The minimum size of the energy storage is determined by the demands of the largest atomic operation that must not be interrupted. For example, to transmit or receive a packet, the buffered energy needs to be sufficient to power the radio for at least the airtime of one complete packet; other examples of atomic operations include reading out

a sensor or executing a checkpoint [60]. In our proposed Find protocol, the largest atomic operation is the frame sequence depicted in Figure 3.7.

If a device with an active power draw higher than the harvesting power is equipped with an energy storage that does not support executing multiple iterations of the largest atomic operation from a single full charge, it is forced to go through periods of inactivity—the device is said to operate intermittently. Intermittency is in stark contrast to duty cycling, which is intentionally used on devices with primary or rechargeable batteries, yet the devices can become active at any point in time subject only to an upper bound on the average duty cycle. By contrast, intermittency prevents a device from becoming active at any point in time, and when a device enters and exits the inactivity phases is only partially controllable, at best.

3.7.2 What Factors Impact Intermittency?

Three key dimensions influence the extent of the intermittency problem: energy input, energy storage, and workload.

Energy input. An ambient energy source may exhibit intermittent behavior, including periods where it emits no energy. Clearly, a battery-free device can only harvest energy when the ambient source emits energy. In this case, provisioning a device with a harvester that provides the power required to continuously operate the device in high-power mode prevents the intermittency problem. This, however, would come with major drawbacks in terms of size, weight, and costs. For example, a battery-free device may draw only 10 µW on average but 10 mW when active, thus requiring to over-provision the harvester by a factor of 1000. While such over-provisioning is in theory always possible, it is severely limited in practice by the constraints imposed by the application requirements.

Energy storage. If permitted by the application requirements, an energy storage larger than the minimum required to execute the largest atomic operation may be used. For example, using a high-capacity

rechargeable battery can prevent intermittency. Such batteries have a high energy density, but their minimum physical dimensions are typically orders of magnitude larger than those of capacitors. Batteries are also more expensive and subject to aging, losing capacity over time and eventually malfunctioning with excessive heat and leakage of potentially toxic chemicals. By contrast, capacitors have low energy density, but are extremely cheap, readily available in sizes well below 0.1 mm^3, have negligible aging effects, and do not contain problematic materials (*e.g.*, toxic chemicals). Thus, despite advances in battery technology, alternative systems to store energy are being explored [6] and capacitors are widely regarded as a more sustainable option [115, 20].

When a device is inactive, it accumulates charge until the capacitor voltage reaches a turn-on threshold. The amount of energy that can be stored depends on the turn-on threshold, which is limited by the breakdown voltage of the capacitor and the device's maximum operating voltage. When a device is active, it discharges the capacitor until the voltage reaches a turn-off threshold, which is dictated by the device's minimum operating voltage. Thus, for the same capacitor, a device with a lower minimum operating voltage or a higher maximum operating voltage can increase the effective amount of buffered energy that can be used. This allows to either use a smaller capacitor or execute longer from a single full charge, potentially alleviating the intermittency problem.

Workload. While lower-power hardware can reduce the average power draw in sleep mode and thus the charging time, it does not generally avoid intermittency. This would require pushing also the active power below the harvesting power.

Reducing the transmission power of the radio can extend the time a device can operate from a single full charge. While this may alleviate the intermittency problem, it also reduces the communication range, which may render device-to-device communication infeasible or require multi-hop networking.

Similarly, using backscatter communication instead of active radio communication may bring the active power draw of a device below the harvesting power and thereby enable continuous operation. However, backscatter requires the presence of an external carrier and may pose limitations in terms of communication range and data rate. In particular, existing practical implementations of tag-to-tag backscatter receivers do not yet reach the point where the end-to-end power draw is negligible (*i.e.*, below sleep power of around 1 µW) [92, 104], thus leaving a significant region in the design space of battery-free backscatter devices where intermittency occurs.

3.7.3 Impact of Built-in Spatial Randomness

Find tackles interleaving by letting nodes randomly and independently delay their wake-ups. This approach is particularly effective in scenarios with little built-in spatial randomness, that is, when the harvested energy exhibits limited variability *between* nodes, regardless of a potentially high temporal variability in harvested energy. We believe this holds for a broad class of battery-free application scenarios, because nodes in a confined space often harvest energy from the same ambient source(s). On the other hand, a high built-in spatial randomness may alleviate the interleaving problem. Although our case study experiments exhibit built-in spatial and temporal randomness, it remains an open question how built-in spatial randomness may influence the choice of Find's delay distribution and scale parameter as well as its overall effectiveness.

3.8 Related Work

Battery-free device-to-device communication. Prior work on battery-free wireless device-to-device communication is mainly theoretical [70, 145], studying the capacity limits for different energy scheduling, transmission, and decoding policies. Understanding energy issues on the

Work	Type	Sensing Signal	Power
Syntonistor [110]	frequency	EM radiation	300 µW
Flight [80]	frequency	light sensor	5394 µW
Flync	freq.+phase	solar current	5 µW

TABLE 3.2: Compared with prior work using powerline frequency for synchronization, Flync provides frequency and phase synchronization from the solar panel current at significantly lower power draw.

receiver side [2] and the impact of intermittency have been open problems. On the other hand, practical work on tag-to-tag backscatter communication has primarily focused on physical-layer issues and considers intermittency an orthogonal problem [83, 59, 92].

Rendezvous and neighbor discovery protocols. Blind rendezvous is the process of establishing a communication link between nodes in a distributed system without any prior information [51]. Neighbor discovery protocols for wireless networks target a sub-class of the blind rendezvous problem with the goal of optimizing the trade-off between discovery latency and energy consumption. Deterministic protocols let nodes wake up according to a schedule based on (co-)prime numbers [37, 64], a quorum [75, 74, 57], or by systematically traversing slots [8, 135]. This way, they can provide guaranteed bounds on discovery latency [67]. Probabilistic protocols are stateless, robust to varying conditions, and offer low average discovery latency [19]. For example, the influential birthday protocol [95] and follow-up work [134, 133] analyze optimal transmit probabilities to maximize the fraction of links discovered in a given time. However, none of the existing neighbor discovery protocols are applicable to battery-free networks because they require nodes to be able to wake up at arbitrary points in time, not taking into account intermittency.

Powerline-based clock synchronization. We are not the first to exploit the powerline frequency signal for synchronization. The Syntonistor extracts a stable clock signal from electromagnetic (EM) powerline radiation using a large coil [110]. It draws 300 µW of power, 60× more than Flync. Flight samples a light sensor to synchronize a node's

oscillator to the powerline-induced brightness variations of fluorescent lamps [80]. Using Flight, synchronization takes 100 ms at a power draw of 5394 µW, 1000× more than Flync. As summarized in Table 3.2, both approaches only synchronize the frequency of local clocks, eliminating the need to periodically compensate for clock drift, but do not exploit phase information. They also use dedicated high-power sensors, whereas Flync uses a low-power circuit to extract the signal from the current of the solar panel.

Energy harvesters as sensors. Previous work has explored the use of the harvesting current or voltage as a sensing signal for indoor positioning [107], gait recognition [87], gesture recognition [132], activity classification [114], and transport-mode detection [113]. To the best of our knowledge, we are the first to exploit context information from harvested energy for synchronization. Furthermore, Flync is the first design that extracts the sensing signal from current variations of a solar panel that is simultaneously used to power the system.

Visible light communication. Flync exploits the powerline-induced brightness variations as an intrinsic property of ubiquitous types of lamps. When modifying existing lighting infrastructure, it is possible to encode arbitrary data into the brightness variations. This opportunity has been used for downlink communication [106], indoor positioning [72], and battery-free duplex visible light communication [78]. By modulating light with a well-defined synchronization signal, the efficiency and applicability of Flync could be further improved. Also, our approach to harvest energy while simultaneously demodulating encoded signals from the same panel may reduce the size and power of existing visible light communication receivers.

3.9 Conclusions

Leaving batteries behind allows for building cheap, tiny, and maintenance-free devices that can be embedded into smart textiles, intelligent surfaces, or even the human body. In this paper, we have addressed

the problem of enabling efficient battery-free device-to-device communication. Experiments with a prototype platform and implementation show that our proposed techniques empower battery-free devices to quickly and efficiently discover each other despite their unpredictable intermittent operation. By bootstrapping battery-free wireless networks, we believe that our work provides a stepping stone for future research toward full system and communication stacks for this emerging kind of networked system.

Availability

Artifacts are available to the public under a permissive MIT license at `https://find.nes-lab.org/`. These include a Python implementation of the Find model from Section 3.2, which can be used to reproduce the analytical results in Figs. 3.3 to 3.6, as well as the hardware design files and the firmware of our prototype implementation from Section 3.4, which we used for the experiments and case study described in Secs. 3.5 and 3.6.

Acknowledgments

Thanks to Brano Kusy, Carlos Pérez-Penichet, and Carsten Herrmann for their feedback that helped improve this paper, to Friedrich Schmidt for his help with the hardware design, to James Broadhead for the preliminary experiments, and to all participants of the contact tracing case study. Thanks also to the anonymous reviewers, and to our shepherd, Shyam Gollakota. This work was supported by the Emmy Noether project NextIoT (DFG grant ZI 1635/2-1) and cfaed.

Postscript. For the first time, Find provides the ability to exchange data between intermittently powered battery-free devices. Its versatility and low overhead make Find easy to implement, and we have used it as a stepping stone for the connection protocol Bonito presented in Chapter 4. Similarly, the key method of delaying device wake-up to change the wake-up offsets between two devices was adopted for Bonito. Flync further reduces the discovery latency for the specific application scenario of indoor light harvesting devices with sufficient flicker. Both methods are well suited to establish the initial encounter, but they are inefficient as sole communication primitive because devices need to wait for the duration of the discovery latency between the exchange of two consecutive messages. Chapter 4 demonstrates that communication efficiency can be significantly increased when devices use the initial encounter to exchange data and establish a connection.

4

Learning to Communicate Effectively Between Battery-free Devices

Prelude. This chapter covers the paper with the same title co-authored by Marco Zimmerling that I presented at the USENIX Symposium on Networked Systems Design and Implementation in 2022 [47]. The paper presents Bonito, the first connection protocol for battery-free systems that enables reliable and efficient bidirectional communication between intermittently powered nodes. As part of this work, we collected real-world energy-harvesting traces from five diverse scenarios involving solar panels and piezoelectric harvesters. The analysis reveals that the nodes' charging times approximately follow well-known distributions. Bonito learns a model of these distributions online and adapts the nodes' wake-up times so that sender and receiver are operational at the same time, enabling successful communication. Experiments with battery-free prototype nodes built from off-the-shelf hardware components demonstrate that the approach improves the average throughput by 10–80× compared with the state of the art.

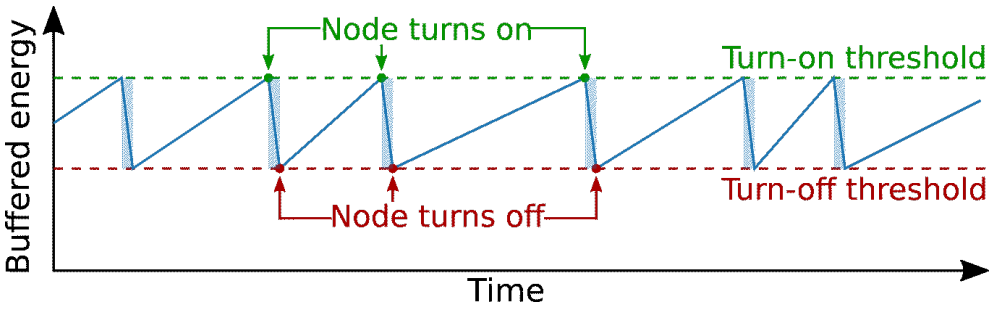
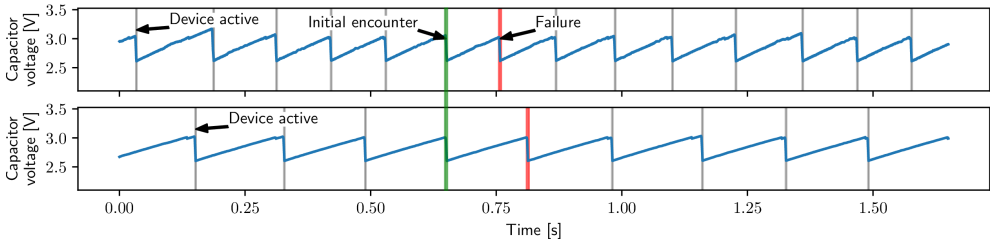


FIGURE 4.1: Because ambient power is often weak, a battery-free node must buffer energy before it can wake up and operate for a short time period. This is known as intermittent operation.

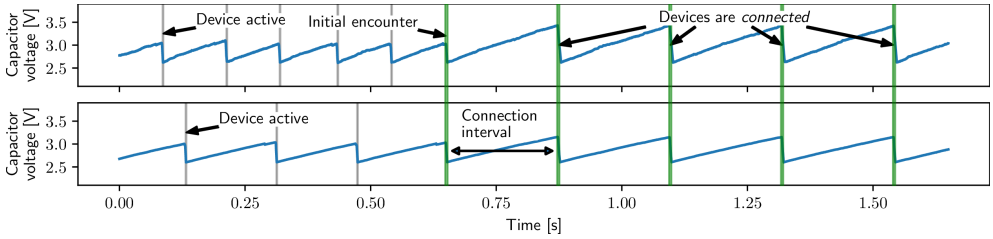
4.1 Introduction

The last few years have seen rapid innovation in battery-free systems [115], culminating in a number of real-world applications [3, 28, 79]. These systems pave the way toward a more sustainable IoT [20] by enabling small, cheap, and lightweight devices to perform complex tasks (*e.g.*, DNN inference [48]) off ambient energy while using tiny, environmentally friendly capacitors as energy storage [115]. However, to replace today's trillions of battery-powered IoT nodes, battery-free devices must learn to communicate.

Challenge. The power that can be harvested from solar, vibrations, or radio signals is typically insufficient to continuously operate a device. A traditional energy-neutral device buffers harvested energy in a rechargeable battery and can *freely control* its average duty cycle to avoid power failures. Instead, a battery-free device cannot avoid power failures, and has *very limited control* over when the power failures begin and end. Figure 4.1 illustrates this so-called *intermittent operation*. After executing for a short time, a battery-free device is *forced* to become inactive and wait for a long, fluctuating time until its capacitor is sufficiently charged again. For example, when harvesting energy from indoor light, our prototype battery-free nodes need to stay off and recharge, on average, for hundreds of milliseconds before they can operate for at most 1 ms.



(A) Because of their short and interleaved activity phases, battery-free devices often need a long time with hundreds of wake-ups until they encounter each other. Even after an initial encounter, the devices quickly get out of sync, rendering communication inefficient and unreliable.



(B) With Bonito, devices learn and exchange statistical models of their charging times and agree on a connection interval that ensures that both devices have sufficient energy at the same time. Maintaining a connection over multiple encounters enables efficient and timely communication.

FIGURE 4.2: The challenge of efficient battery-free device-to-device communication in (a) and our proposed protocol in (b).

Many techniques have been developed to deal with intermittency on a *single* battery-free device [91, 16, 27], but how to communicate *between* intermittently powered devices is one of the most pressing problems yet to be solved [55, 142, 82]. This is due to the fact that device-to-device communication is a fundamental building block for a variety of network and system services, including optimal clock synchronization [77], ranging and localization [24, 50], sensor calibration [116], distribution and coordination of sensing and computing tasks [86], collaborative learning [138], and efficient and reliable wireless networking [76]. Realizing these services across battery-free devices has the potential to enable novel and more sustainable IoT and sensor network applications, from automatic contact tracing to planetary-scale environmental monitoring.

To be able to communicate, sender and receiver must be active and have enough energy for at least one complete packet transmission *at*

the same time. However, since the nodes' activity phases are generally interleaved and short compared to their charging times, as visible from the real-world trace in Figure 4.2a, it often takes thousands of wake-ups until two nodes encounter each other and communication becomes possible [46]. Moreover, after an encounter, the nodes quickly get out of sync if they become active immediately after a recharge, as stipulated by the state of the art [23, 88] and apparent in Figure 4.2a. This is because ambient energy varies across time and space [7], which leads to fluctuating and different charging times between the nodes.

Besides establishing a first encounter [46], active radio communication has been considered too demanding for battery-free devices [93]. Conversely, work on backscatter communication has focused on physical-layer issues, such as improving range and throughput, purposely considering high-energy environments, batteries, or cables to continuously power the devices in the experiments to avoid intermittency [144, 111, 92, 83]. However, when running off ambient energy, duty cycling of the backscatter transceivers becomes necessary [83, 126, 30]—and, without a battery, the intermittency problem occurs.

Contribution. This paper presents Bonito, the first connection protocol for battery-free wireless networks. Bonito provides reliable and efficient bi-directional communication despite the time-varying intermittency of battery-free devices.

The real-world trace in Figure 4.2b illustrates the high-level protocol operation. Unlike the state of the art, Bonito enables two battery-free nodes, after an initial encounter, to maintain a *connection* across multiple consecutive encounters. To this end, Bonito continually adapts the *connection interval*, which is the time between the end of an encounter and the beginning of the next encounter. A shorter connection interval provides more communication opportunities in the long run. However, a connection interval that is shorter than any of the nodes' charging times breaks the connection and requires the nodes to wait for a long time until they encounter each other again. Thus, the challenge is to keep the connection interval as short as possible without losing

the connection, which is difficult in the face of time-varying charging times.

One of our key insights is that, depending on the scenario and energy-harvesting modality, the charging time of a battery-free node approximately follows well-known probability distributions. We leverage this insight in Bonito by letting each node *continuously learn and track* the parameters of a model that approximates the distribution of locally observed charging times against non-stationary effects (*e.g.*, changes in mean or variance). Then, to maintain an efficient and reliable connection, the nodes exchange at every encounter their current model parameters and jointly adapt the connection interval.

We implement Bonito on a custom-designed ultra low-power battery-free node. Our prototype is built from off-the-shelf components, including an ARM Cortex-M4 microcontroller unit and a 2.4 GHz BLE radio. The node harvests energy from a solar panel or a piezoelectric harvester, using a 47 µF capacitor as energy storage.

To evaluate Bonito through testbed experiments and fairly compare it against two baselines, we use up to 6 Shepherd observers [44] to record and replay real-world energy-harvesting traces from 5 diverse scenarios. Our results show, for example, that Bonito maintains connections for hundreds of consecutive encounters, and that it outperforms the state of the art by 10–80× in terms of throughput. We also conduct a case study that demonstrates the utility of Bonito for accurate and timely occupancy monitoring in homes and commercial buildings.

Overall, this paper contributes the following:

- We collect 32 h of energy-harvesting traces from 5 different scenarios. Our analysis of these traces provides new insights into spatio-temporal intermittency patterns.
- We design the Bonito protocol. Bonito enables, for the first time, reliable and efficient communication between intermittently powered battery-free devices.
- We demonstrate an efficient implementation of Bonito on a prototype node with a 3.1 mm^3 ceramic capacitor.

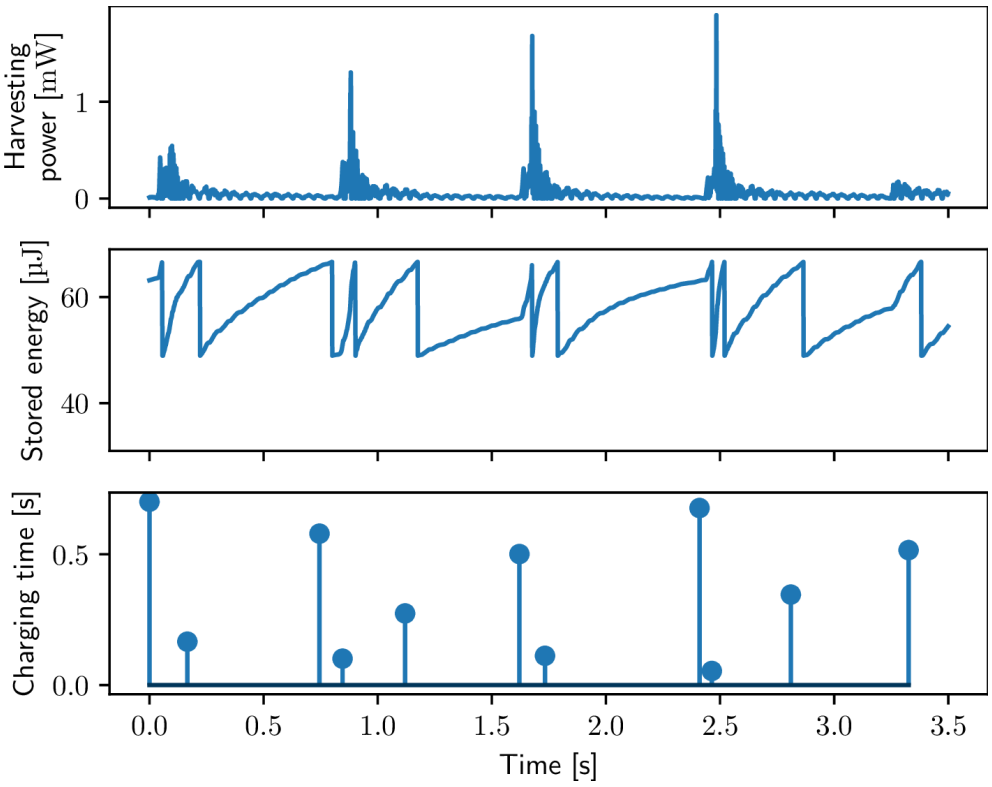
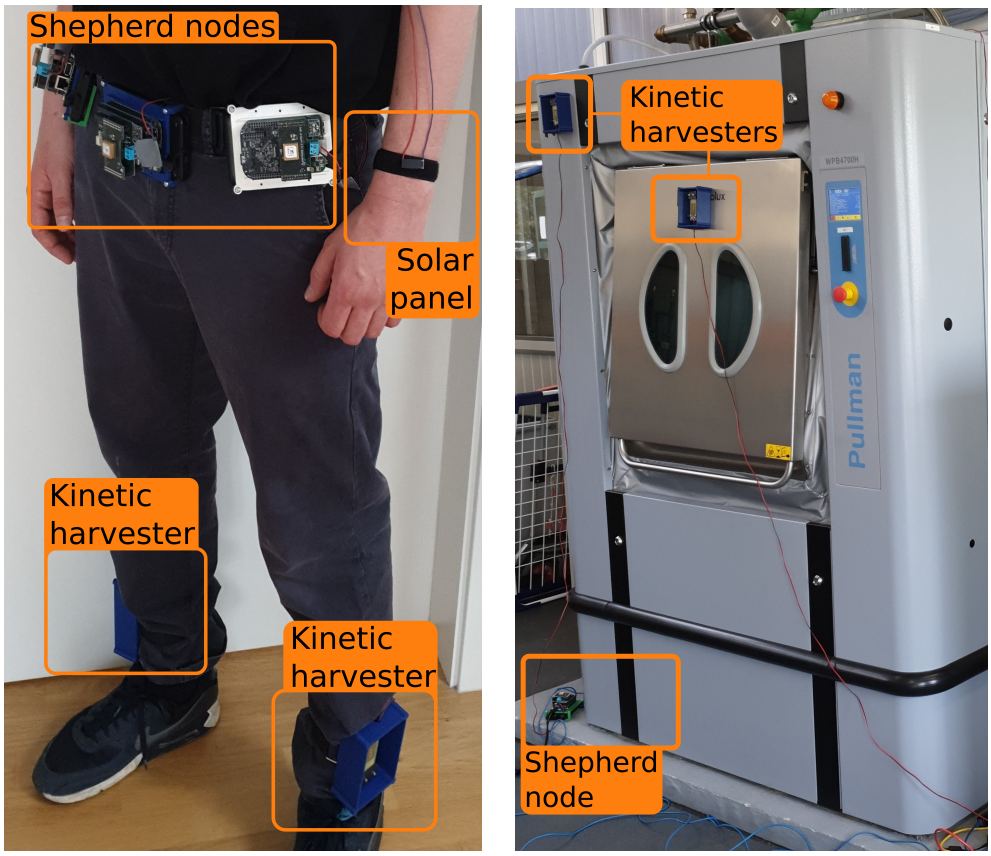


FIGURE 4.3: The top plot shows an example trace of real kinetic harvesting power during jogging (see picture in Figure 4.4a). The middle and bottom plots show the corresponding energy stored in the capacitor and the resulting charging times of a simulated battery-free device.

- Results from testbed experiments and an occupancy monitoring case study provide evidence that Bonito performs well under a diverse range of real-world conditions.

4.2 Motivation

While previous work on intermittency has focused on individual battery-free devices [88, 16, 27] or discovery of neighboring devices [46], reliable and efficient device-to-device communication is still an open challenge. By *device-to-device communication* we mean the regular exchange of application data between two battery-free devices after they have successfully discovered each other through a first encounter [46].



(A) Runner with full measurement setup for the *jogging* dataset.

(B) Washing machine with partial setup for the *washer* dataset.

FIGURE 4.4: Pictures from two of the five scenarios in which we use synchronized Shepherd nodes [44] to record energy-harvesting traces.

Dataset	Energy Source	Harvester Part #	Duration	#Devices	#Links	#Wake-ups	Model
Jogging	Human motion	S128-J1FR-1808YB	1 h	3	10	13252	Exponential
	Outdoor solar	KXOB25-05X3F		2		119127	Normal
Stairs	Outdoor solar	KXOB25-05X3F	1 h	6	15	359002	Normal
Office	Indoor light	SM141K06L	1 h	5	10	98324	GMM
Cars	Car vibrations	S128-J1FR-1808YB	2 h	6	15	8517	Exponential
Washer	Machine vibrations	S128-J1FR-1808YB	45 min	5	10	22224	Normal

TABLE 4.1: Overview of energy-harvesting datasets we record in a variety of scenarios.

To motivate the need for our work, we consider the scenario of battery-free wearables. Figure 4.3 shows real-world data from a piezoelectric energy harvester that is attached to the ankle of a person (see Figure 4.4a). The upper plot shows the harvesting power while the person is jogging, recorded by a Shepherd node. Shepherd is a measurement tool that records time-synchronized voltage and current traces from one or more energy-harvesting nodes with high rate and resolution [44]. The power spikes correspond to when the foot strikes the ground, with significantly lower harvesting power during the rest of the stride cycle. Based on trace-driven simulations, the middle plot shows the corresponding amount of harvested energy stored in an ideal 17 µF capacitor powering a battery-free device that turns on when the capacitor voltage exceeds 3 V and turns off when the capacitor voltage falls below 2 V. We see that when the device powers up, the stored energy is quickly consumed, forcing it to turn off already after about 1 ms. While powered off the harvesting power exceeds the standby power, so energy is accumulated and the capacitor voltage rises again. Compared to the short activity phases, the time needed to charge the capacitor, shown in the bottom plot of Figure 4.3, is much longer and varies significantly over time.

The variability of a node's charging time is a function of its location and the associated energy environment, that is, how much power the harvester delivers at any given time. Thus, two battery-free devices, even when they are physically close to each other, have a different energy environment and therefore experience different charging times.

As an example, Figure 4.5 plots the charging times of two devices during jogging over one hour. One device is powered by a piezoelectric harvester attached to the left ankle of a person, while the other device is powered by the same type of harvester attached to the right ankle of the person (see Figure 4.4a). Each point in Figure 4.5 indicates the charging times of both devices when they begin to charge their 17 µF capacitors at the same time from the same initial charge. We observe that in many instances the two nodes have vastly different charging times. This means that if nodes become active as soon as they reach

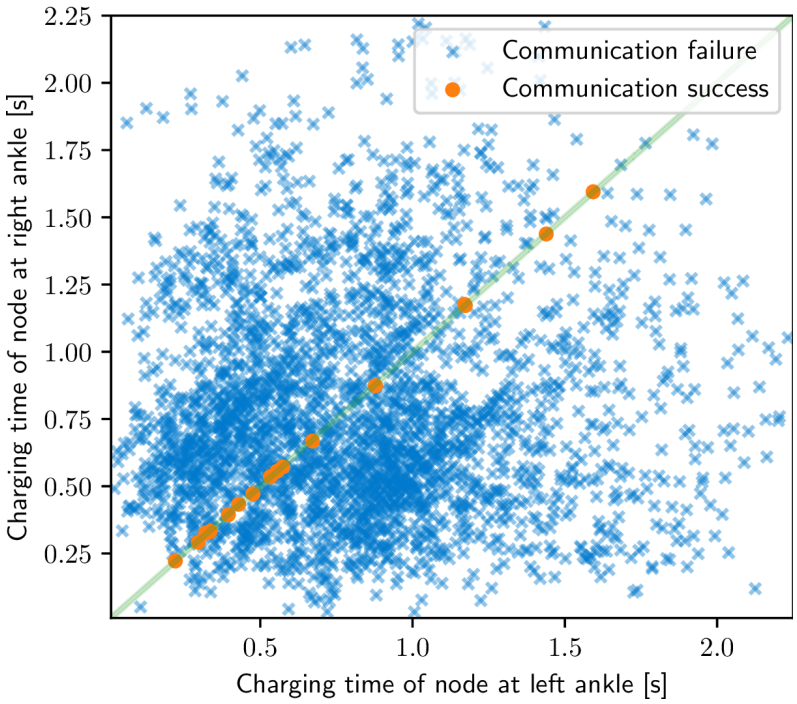


FIGURE 4.5: Charging times of two battery-free devices powered by kinetic harvesters attached to a jogger's ankles (see Figure 4.4a). Using the greedy approach, the devices communicate successfully only in 0.04 % of the cases in which the charging times are almost identical.

the turn-on threshold, which is the state-of-the-art approach, called *greedy* and illustrated in Figure 4.2a, the nodes often wake up with an offset that prevents communication, despite a successful encounter at the previous wake-up. Indeed, the success rate for the two nodes in Figure 4.5 is less than 0.04 %. This leads to poor communication reliability and efficiency as the nodes more often than not fail to exchange their data.

To assess the generality of these observations, we record distributed energy-harvesting traces in diverse scenarios using multiple Shepherd nodes [44]. Table 4.1 lists the main characteristics of the five datasets we collected:

- The full *jogging* dataset comprises traces from two participants, each equipped with two piezoelectric harvester at the ankles and a solar panel at the left wrist (see Figure 4.4a). The two participants run together for an hour in a public park, including short

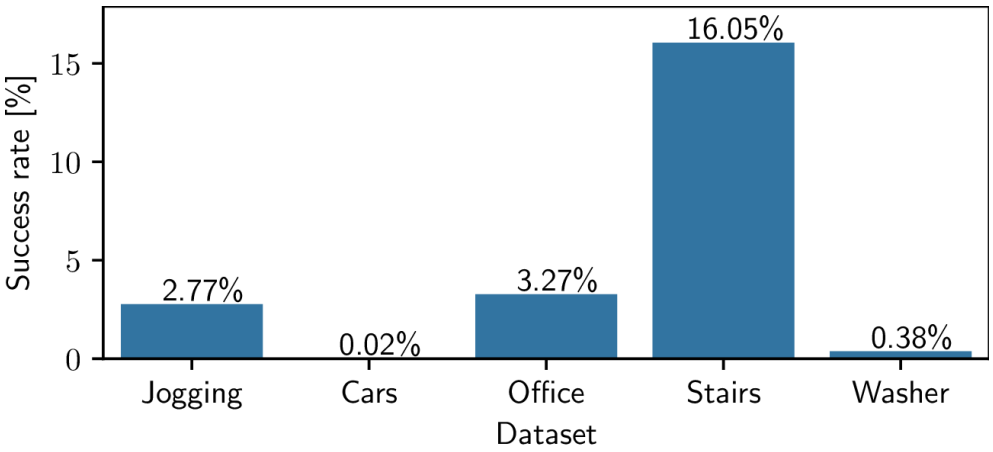


FIGURE 4.6: Success rate of greedy approach in trace-driven simulations, averaged across all pairs of devices (*i.e.*, links) in a given dataset.

walking and standing breaks.

- For the *stairs* dataset, we recorded traces from six solar panels that are embedded into the surface of an outdoor stair in front of a lecture hall. Over the course of one hour, numerous students pass the stairs, leading to temporary shadowing effects on some or all of the solar panels.
- The *office* dataset comprises traces from five solar panels mounted on the doorframe and walls of an office with fluorescent lights. During the one-hour recording, people enter and leave the office and operate the lights.
- The *cars* dataset contains traces from two cars. Each car is equipped with three piezoelectric harvesters mounted on the windshield, the dashboard, and in the trunk. The cars drive for two hours in convoy over a variety of roads.
- The *washer* dataset includes five traces from piezoelectric harvesters mounted on a WPB4700H industrial washing machine, as shown in Figure 4.4b, while the machine runs a washing program with maximum load for 45 min.

Figure 4.6 plots for each dataset the average success rate across all pairs of devices (*i.e.*, communication links) in the scenario. Even in the most favorable scenario, *stairs*, where the solar panels receive a fairly constant and similar energy input from natural sunlight, we find that

the greedy approach succeeds in only 16 % of the cases. In all other scenarios, the success rate ranges below 3.5 %. Our experiments on real battery-free nodes in Section 4.5 confirm these trace-driven simulation results.

4.3 The Bonito Protocol

This section describes the Bonito protocol. The Bonito protocol enables two battery-free devices to stay connected after a first encounter, which can happen either coincidentally or with the support of a neighbor discovery protocol [46].

4.3.1 Overview

Bonito aims to make nodes repeatedly encounter each other so they can exchange application data reliably and efficiently, as shown in Figure 4.2b. To ensure that nodes wake-up with a time offset small enough for a successful encounter, they agree at every encounter on a new *connection interval* T_C. This is the time between the end of the current encounter and the beginning of the next (*i.e.*, planned) encounter.

Main idea and approach. For two nodes i and j with known charging times c_i and c_j, the shortest possible connection interval T_C^* is simply the maximum of their charging times

$$T_C^* = max(c_i, c_j) \tag{4.1}$$

If a shorter connection interval $T_C < T_C^*$ is used, then one node does not reach the required energy level to become active by T_C. Thus, the encounter fails, preventing the nodes from agreeing on the next connection interval—the connection is *lost*. A lost connection entails that the nodes often need to wait for a long time until they encounter each other again to resume communication. However, choosing a longer connection interval $T_C > T_C^*$ to mitigate the risk of a lost connection adds unnecessary delay as nodes, after having reached the required energy level, are forced to wait before they wake up at T_C.

The key challenge is to determine the connection interval T_C such that both nodes have enough energy while introducing only minimal delay. This is difficult as the charging times c_i and c_j are unknown and time-varying, as discussed in Section 4.2.

Using a probabilistic approach, we address this problem as follows. Let p be the probability that nodes i and j have sufficient energy to become active after a connection interval T_C. This corresponds to the probability that the nodes' charging times, c_i and c_j, are shorter than the connection interval T_C. Modeling c_i and c_j as random variables with a strictly monotonically increasing joint cdf $F_{i,j}$, this translates into

$$p = F_{i,j}(c_i = T_C, c_j = T_C) \tag{4.2}$$

Solving for T_C yields the minimum connection interval that guarantees, with a user-defined probability p, a successful encounter of the two nodes at their next wake-up

$$T_C = F_{i,j}^{-1}(p) \tag{4.3}$$

where $F_{i,j}^{-1}$ is the inverse joint cdf of c_i and c_j.

Base protocol. In practice, the joint cdf $F_{i,j}$ is rarely known a priori. Moreover, $F_{i,j}$ can only be estimated online by the nodes based on full knowledge of each other's charging times. Unfortunately, this requires frequent communication between battery-free nodes—precisely what Bonito intends to enable.

To circumvent this chicken-and-egg problem, we assume that the charging times, c_i and c_j, are statistically independent. In this case, the joint cdf $F_{i,j}$ is the product of the marginal cdfs F_i and F_j. The marginal cdfs can be estimated locally by each node from observations of their own charging times.

Based on these insights, we propose the following main steps of the Bonito protocol:

1. Each node i continuously estimates the marginal cdf F_i of its charging time based on local measurements.
2. When two nodes i and j encounter each other, they exchange their current estimates of F_i and F_j.
3. Using the same inputs (*i.e.*, the marginal cdfs F_i and F_j and the user-defined probability p), both nodes compute the same new connection interval T_C according to (4.3).
4. Both nodes become active and communicate after the new connection interval T_C, and continue with step 2.

In this way, Bonito adapts the connection interval to changes in the energy environment, effectively enabling battery-free nodes to stay connected across several hundreds of subsequent encounters, as demonstrated by our experiments in Section 4.5.

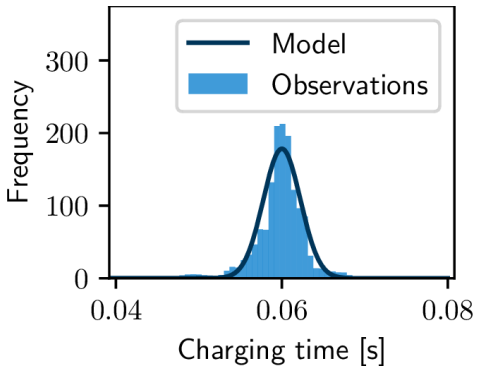
To achieve this performance, we first need to answer the following key questions in our design of Bonito:

- How to compactly represent and exchange the marginal cdfs F_i and F_j in the face of limited energy (Section 4.3.2)?
- How to learn and track online an accurate estimate of F_i against a changing energy environment? (Section 4.3.3)
- How to efficiently compute the inverse joint cdf $F_{i,j}^{-1}(p)$ to obtain the connection interval T_C? (Section 4.3.4)

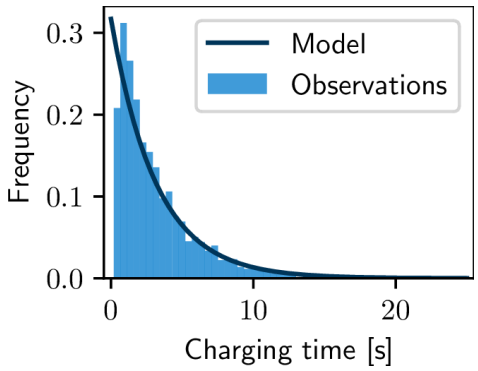
4.3.2 Modeling Charging Time Distributions

Because of the small energy storage, battery-free devices can only exchange a limited amount of data during an encounter. Thus, the marginal cdfs F_i and F_j must be represented in a compact form in order to be able to exchange them.

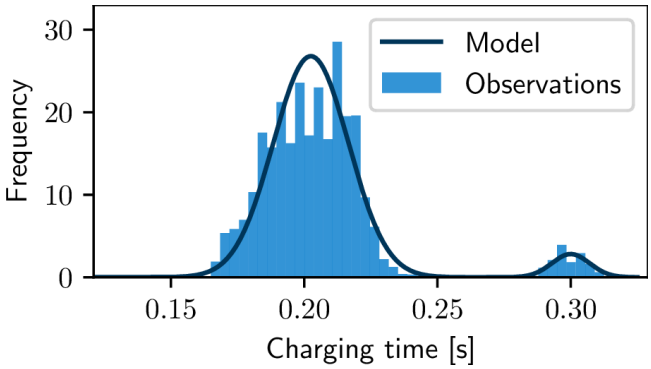
Unlike the common belief that the duration of a recharge is completely random [85, 27], we make the empirical observation that, in the scenarios we considered, the nodes' charging times can be faithfully modeled by well-known distributions. The rightmost column of Table 4.1 lists the models we use for each dataset. To illustrate, Figure 4.7 plots representative charging time distributions and the corresponding models



(A) Stairs dataset, normal distribution.



(B) Cars dataset, exponential dist.



(C) Office dataset, Gaussian mixture model.

FIGURE 4.7: Charging time distributions of individual nodes. The nodes' charging times can be modeled by well-known distributions.

for the stairs, cars, and office datasets. Non-stationary effects like a time-varying mean are removed in the plots as these are effectively handled by our online learning approach detailed in Section 4.3.3.

We observe in Figure 4.7a that when harvesting energy from outdoor solar with a constant harvesting voltage, the charging time can be modeled by a normally distributed random variable. The intuition is that temporary environmental effects, such as shadowing and change in incidence angle, let the charging time vary around a certain value. Figure 4.7b shows that an exponential distribution is often a good fit when harvesting kinetic energy. This can be explained by the decaying response of a piezoelectric harvester to the distinct impulses of a car during driving (*e.g.*, acceleration, breaking, bumps) or a person during jogging (see Figure 4.3). In the washer scenario, instead, we

find that the continuous shaking of the industrial washing machine over long periods induces approximately normally distributed charging times. Looking at Figure 4.7c, we see that in the office scenario the charging times are mostly distributed around a certain value. However, the MPPT of the DC-DC converter used in this scenario, which periodically disconnects the charger for a short time, leads to a second peak. We approximate this distribution with a Gaussian mixture model (GMM).

These observations motivate us to model the marginal cdf F_i of a node's charging time in the scenarios we considered through the parameters of a normal distribution (2 parameters), an exponential distribution (1 parameter), or a GMM (6 parameters for two Gaussians and two weights). The last column of Table 4.1 lists the corresponding model for each of the datasets. The jogging dataset contains traces from different types of harvesters: We use an exponential distribution to model the charging times of kinetic harvesting nodes and a normal distribution for the solar harvesting nodes. During an encounter, a node only needs to share the type of model and the current estimates of the model parameters.

4.3.3 Learning Distribution Parameters Online

We now turn to the problem of estimating the parameters of a given charging time distribution based on local observations. Given a sample of n independent and identically distributed observations, the log-likelihood $\mathcal{L}(\theta \mid x)$ and the corresponding maximum likelihood estimator $\widehat{\theta}$ are given by

$$\mathcal{L}(\theta \mid x) = \ln\left(\prod_{i=1}^{n} f_\theta(x_i)\right) = \sum_{i=1}^{n} \ln f_\theta(x_i) \tag{4.4}$$

$$\widehat{\theta} = \arg\max_{\theta} \mathcal{L}(\theta \mid x) \tag{4.5}$$

where $f_\theta(x_i)$ is the conditional probability to observe x_i if the underlying distribution is parameterized with θ.

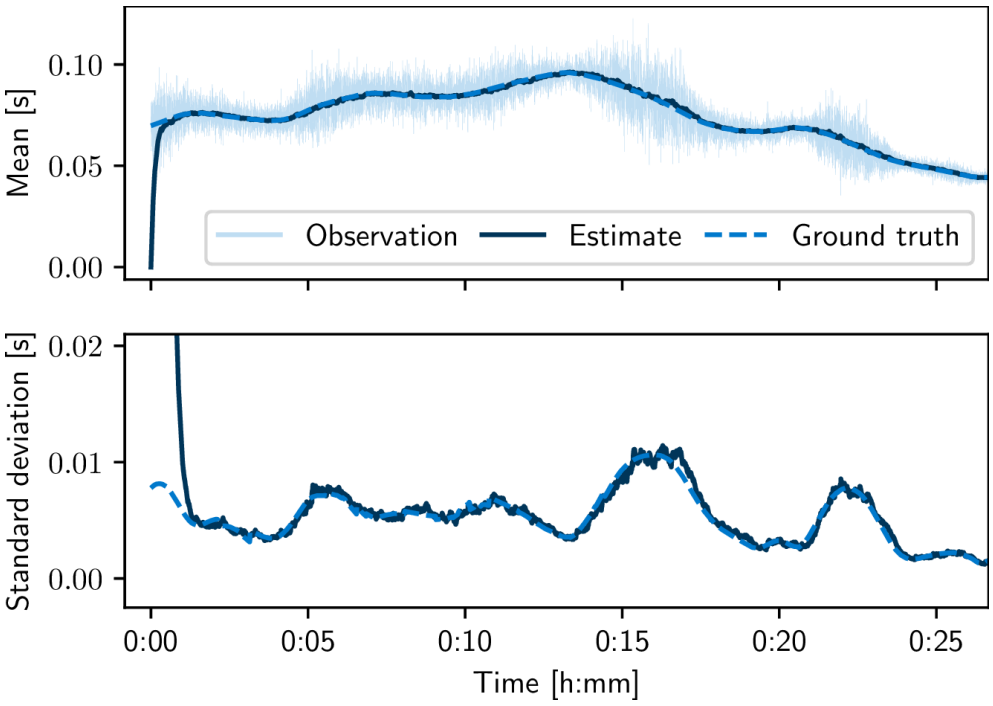


FIGURE 4.8: Varying mean and standard deviation over a moving window of one of the trace from the stairs dataset reveal non-stationarity. Using SGD, the changing distribution parameters are tracked online.

Unfortunately, vanilla maximum likelihood estimation is not viable in our setting. First, the observations of the charging time become available only one by one at runtime, yet the nodes do not have enough memory and energy to recompute the estimator with every new observation. Further, the charging time distributions are non-stationary. For instance, the dashed lines in Figure 4.8 reveal trends in the mean and standard deviation of a node's charging time from the stairs dataset. Thus, an approach is needed that dynamically adjusts the parameter estimates to changing energy harvesting conditions.

To address these problems, Bonito learns the distribution parameters online using stochastic gradient descent (SGD), which has become a popular method for training a wide range of machine learning models [15]. Compared to a sliding window based approach, SGD is less computationally demanding as the parameter update is only computed for the current observation rather than for a set of past observations that also have to be kept in memory. If the gradient of $\mathcal{L}(\theta \mid x)$ is

known, one solution to (4.5) is to iteratively adjust $\widehat{\theta}$ along the gradient, known as gradient descent

$$\nabla \mathcal{L}(\theta \mid x) = \nabla \sum_{i=1}^{n} \ln f_\theta(x_i) \tag{4.6}$$

$$\widehat{\theta} = \widehat{\theta} + \eta \cdot \nabla \mathcal{L}(\widehat{\theta} \mid x) \tag{4.7}$$

By pulling the ∇ operator in (4.6) into the sum, the update step in (4.7) can be split into a series of updates for every individual observation x_i. This yields the update equation of SGD

$$\widehat{\theta}_i = \widehat{\theta}_{i-1} + \eta \cdot \nabla \mathcal{L}(\widehat{\theta} \mid x_i) \tag{4.8}$$

Section 4.9 derives the gradient equations required to solve (4.8) for the normal, exponential, and Gaussian mixture models. By keeping the learning rate η constant, Bonito implicitly reduces the weight of old observations relative to more recent observations. This way, devices dynamically learn changing properties of the charging time distribution locally, without information exchange with other devices.

Example. Figure 4.8 illustrates how Bonito learns and tracks mean and standard deviation of a non-stationary normal distribution. To obtain ground truth, we sample charging times (*i.e.*, observations) from a known normal distribution, whose mean and variance change dynamically over time. We extract these changes from one of the traces in the stairs dataset using a 2 min moving average filter. We can see in Figure 4.8 that the parameter estimates of Bonito converge from their initial values (zero mean and unit standard deviation) to the true ground truth parameters within less than a minute. Then the estimates closely follow the changes of the underlying distribution.

4.3.4 Computing Inverse Joint CDF Efficiently

Having shared the type of model and the current estimates of the model parameters during an encounter, Bonito needs to compute the

new connection interval T_C from the inverse joint cdf $F_{i,j}^{-1}$ for a user-defined probability p. This is difficult since there exists no closed-form solution for most bivariate distributions, let alone for joint cdfs of different distribution families (*e.g.*, when a solar and a kinetic energy harvesting node in the jogging scenario want to communicate). Instead, we have to solve (4.3) numerically, while taking into account the energy and compute constraints of battery-free devices.

We are interested in the connection interval T_C where the joint cdf is equal to the user-defined target probability, that is, $F_{i,j}(T_C) = p$. This yields the following objective function

$$f(T_C) = F_{i,j}(T_C) - p = F_i(T_C) \cdot F_j(T_C) - p = 0 \tag{4.9}$$

Note that $f(T_C)$ has a single root—the sought solution—as $F_{i,j}$ is strictly monotonically increasing. Bonito solves this problem using the well-known bisection method, which iteratively finds the root of any continuous function that has its root inside a bracket (*i.e.*, search interval). Indeed, we can derive such a bracket based on the inverse cdfs of our marginal distributions, which either have a closed form solution (exponential and normal) or are easy to approximate (GMM).

To derive a lower bracket, we first note that $F(x) < 1$ for any cdf F. It follows that $F_{i,j}(x = z, y = z) = F_i(x = z) \cdot F_j(y = z) < \min(F_i(x = z), F_j(y = z))$ and therefore the lower bracket

$$F_{i,j}^{-1}(p) > \max\left(F_i^{-1}(p), F_j^{-1}(p)\right) \tag{4.10}$$

Next, to obtain an upper bracket, we introduce $q = \sqrt{p}$ and $c = \max(F_i^{-1}(q), F_j^{-1}(q))$. Let F_m be the marginal cdf (*i.e.*, either F_i or F_j) such that $F_m^{-1}(q) = c$, that is, the marginal cdf that reaches q *later*. Let F_n be the other marginal cdf that reaches q *sooner*. From $F_n(c) \geq F_m(c)$ follows $F_{i,j}(c) = F_m(c) \cdot F_n(c) \geq F_m(c) \cdot F_m(c) = q^2 = p$. Finally, because $F_{i,j}(c)$ is monotonically increasing, we obtain the upper bracket

$$F_{i,j}^{-1}(p) \leq max(F_i^{-1}(q), F_j^{-1}(q)) \tag{4.11}$$

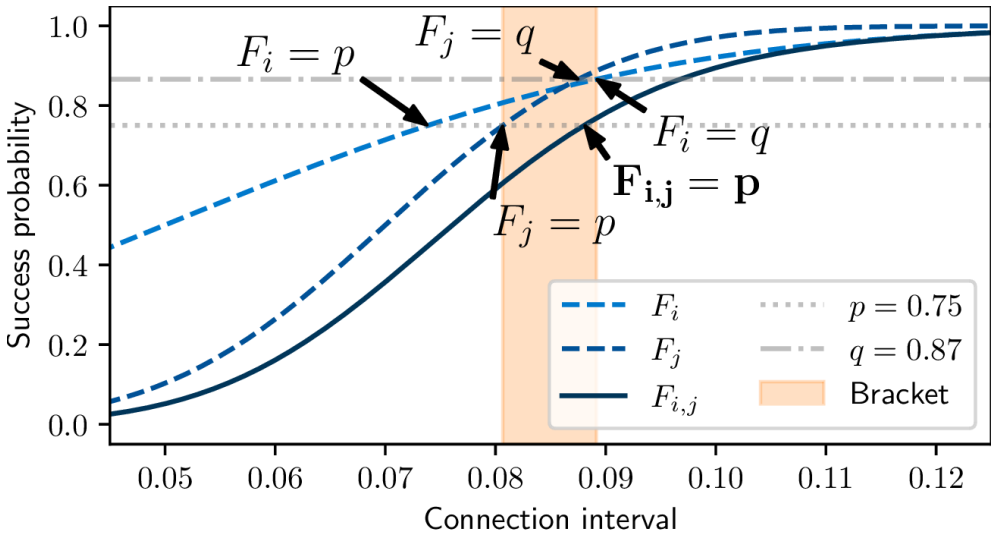


FIGURE 4.9: Bracketing the inverse joint cdf based on the inverse cdf of the marginal distributions enables efficient computation of the connection interval on resource-constrained battery-free devices.

Example. Figure 4.9 shows an example, where (4.10) and (4.11) are used to determine an initial bracket for $F_{i,j}^{-1}(p = 0.75)$. The resulting bracket $[0.61, 0.77]$ is already relatively tight, and therefore we find the solution $F_{i,j}^{-1}(p = 0.75) = 0.88$ with a tolerance of 0.01 after only three bisection steps.

4.3.5 Impact of Target Probability

The target probability p is a key parameter of the Bonito protocol that must be set by the user. It specifies the probability that both devices have accumulated enough energy in their capacitors to become active after a connection interval T_C. A high target probability requires a long connection interval T_C, increasing communication delay and lowering throughput.

To illustrate how the choice of p impacts communication reliability and efficiency, we run trace-driven simulations as detailed in Section 4.2 on the traces from the datasets in Table 4.1. We use two metrics to quantify the performance of Bonito: As a proxy for communication reliability, we define the *success rate* as the ratio of successful encounters with Bonito to the total number of wake-ups. As a proxy for communication

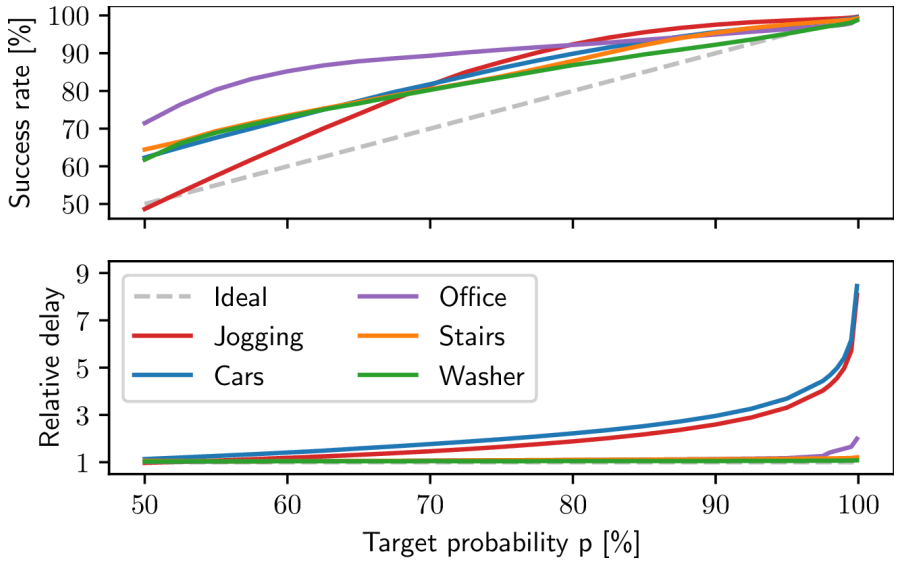


FIGURE 4.10: Trace-driven simulations reveal that the rate of successfully arranged encounters matches the user-defined target probability. The price to pay in terms of latency depends on the model of the underlying charging time distribution.

efficiency, we consider the *relative delay* as the median of all successful connection intervals with Bonito divided by the median of the optimal clairvoyant solutions according to (4.1). Figure 4.10 plots for each dataset success rate and relative delay averaged across all links. We can observe the following:

- A higher target probability p leads to a higher success rate, which demonstrates the plausibility of our approach. In most cases, the success rate is even slightly higher than requested, presumably due to small model errors.
- Since connection losses are costly, a higher target probability is preferable in practice. Figure 4.10 shows that the price to pay in terms of a higher relative delay depends on the scenario. For the cars and jogging datasets, where most or all links include at least one node with approximately exponentially distributed charging times, the relative delay increases exponentially with p, due to the heavy tail of the distribution. For GMM (office), the increase is moderate, whereas it is hardly noticeable for the normal distribution (washer and stairs).

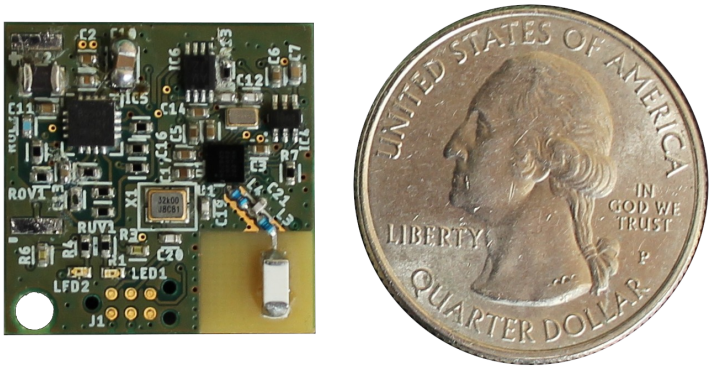


FIGURE 4.11: Prototype battery-free node based on the nRF52805 MCU. A sustainable 3.1 mm^3 ceramic capacitor is used as energy storage.

4.4 Implementation

In this section, we describe the hardware and software components of our prototype implementation.

4.4.1 Hardware

We design an ultra low-power battery-free node based on the popular Nordic Semiconductor nRF52805 MCU. This particular MCU features a 2.4 GHz BLE radio and a state-of-the-art 32-bit 64 MHz ARM Cortex-M4, which is powerful enough to complete also more demanding computations in a short time, benefitting overall system efficiency. To enable low-power timekeeping between wake-ups, the MCU is equipped with a 32 kHz crystal with $\pm$20 ppm frequency tolerance. A TI BQ25504 step-up converter charges a 2 mm $\times$ 1.25 mm $\times$ 1.25 mm 47 µF MLCC from a connected solar panel or a piezoelectric energy harvester. Once the capacitor voltage reaches a hardware-programmable threshold of 3.3 V, the BQ25504 sets one of its pins high. This pin is wired to a TI TS5A23166 analog switch that connects the MCU to the capacitor-buffered supply voltage.

Due to its DC bias characteristics, the capacitor has an effective capacitance of only 17 µF at 3.3 V. This allows for a maximum active time of around 1 ms per wake-up. A larger capacitance would increase the

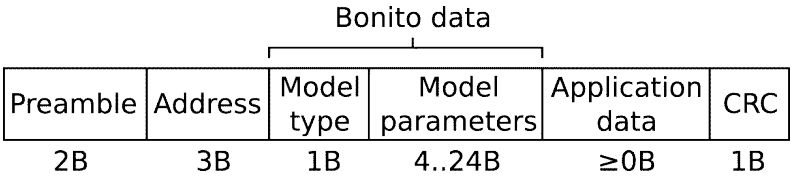


FIGURE 4.12: Packet format. Using Bonito, nodes exchange between 5 B and 25 B carrying model type and parameters during an encounter.

active time per wake-up and the charging time between wake-ups. To minimize the physical dimensions and the price of the node, we choose the minimum capacitance that allows the nodes to remain active for long enough to compensate for clock drift accumulated over a connection interval of 5 s (see Section 4.4.2).

The node also integrates a circuit to measure the current flow from the harvester, which can be used as a sensing signal [114]. The two-layer PCB shown in Figure 4.11 measures 20 mm × 20 mm. The total cost of all components is $8.73.

4.4.2 Software

We implement Bonito and the Find neighbor discovery protocol [46] on our battery-free nodes. Find is used to establish an initial encounter after a connection loss or a power failure.

Bonito protocol settings. We use the 2 Mbit s^{-1} BLE mode and the frame structure depicted in Figure 4.12. Depending on the model type, encoded by one byte, a packet carries 1, 2, or 6 model parameters represented by 32-bit floating point values.

To jointly agree on the next connection interval, Bonito requires nodes to exchange messages bi-directionally during an encounter. The exact sequence of packet exchanges is subject to application requirements and can be flexibly configured. We implement the packet sequence shown in Figure 4.13. When two nodes encounter each other using Find, one of the nodes receives the first beacon and replies with an acknowledgment. At all following encounters, the node that received the first beacon starts to listen at the time agreed on using Bonito.

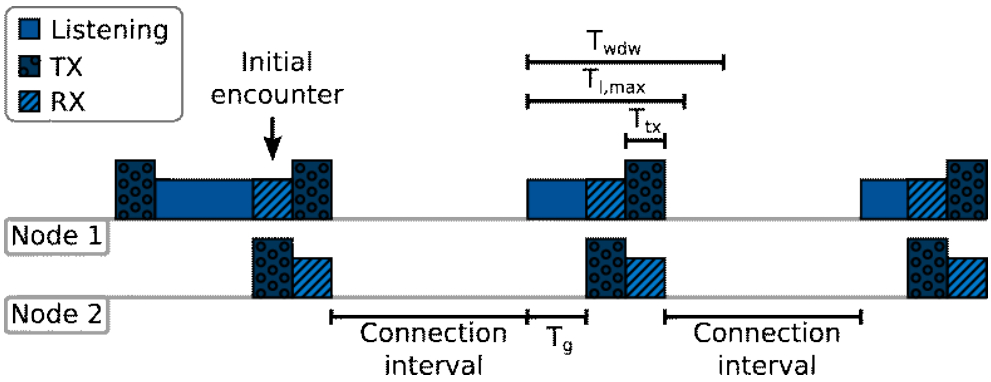


FIGURE 4.13: After the initial encounter, nodes use Bonito to agree on a connection interval. At the next encounter, one of the nodes starts to listen and the other node transmits its packet after a grace period to account for clock drift due to the long charging times. After receiving this packet, the listening nodes replies with its own packet.

Due to the small energy buffer, a node can keep the radio on for at most $T_{wdw} = 1\,\text{ms}$. Thus, the maximum listening time is $T_{l,max} = T_{wdw} - T_{tx} - T_{ta} = 820\,\mu\text{s}$, where $T_{ta} \approx 40\,\mu\text{s}$ is the time it takes to switch from receive to transmit mode and $T_{tx} = 140\,\mu\text{s}$ is the airtime of a packet with 6 model parameters and 4 B of application data. To increase the robustness to clock drift in the face of long charging times and hence long connection intervals, we let the node that sends first transmit its packet after a grace period of $T_g = 0.5 \cdot T_{l,max} - 1.5 \cdot T_{tx} = 200\,\mu\text{s}$. We can thus tolerate an offset of up to $\pm 200\,\mu\text{s}$ between the clocks of the two nodes, which corresponds to a maximum connection interval of 5 s when taking into account the frequency tolerance of the 32 kHz crystal oscillator. Upon receiving the packet, the other node switches to transmit mode and sends its own packet.

In our current implementation of Bonito, the devices consider a connection as lost whenever a planned packet exchange fails, for example, due to fading, external interference, or when one of the two devices does not reach the turn-on threshold by the end of the connection interval. In this case, they return to discovery mode and use Find to re-establish the connection.

Runtime support. In addition to Bonito and Find, we implement an

efficient *soft intermittency* runtime, where the MCU is gracefully suspended to an ultra low-power mode before an impeding power failure [46]. This reduces the costs associated with a cold start after a hardware reset and allows to keep track of time between consecutive wake-up events using the built-in RTC. To this end, a node periodically samples the capacitor voltage during charging with the built-in ADC until the capacitor voltage reaches a software-defined turn-on threshold. Then the node executes protocol and application code until it is interrupted by the power-fail comparator, upon which it immediately transitions back into low-power mode to replenish its energy buffer.

Although our runtime tries to prevent hardware resets, after multiple seconds without any energy input, the sleep current drains the remaining charge from the capacitor and the node eventually powers off. While powered off, the on-board SRAM is subject to decay, that is, bits that were set to one may flip and become zero after some time. To still retain the trained model of a node's charging time distribution across short power failures, we store it in a dedicated section of the SRAM. After every model update, we compute a checksum over this section and store it next to the model parameters. If the recomputed checksum after a hardware reset does not match the checksum stored in memory, we conclude that the memory is corrupted and restart training the model with the initial parameters.

4.5 Evaluation

This section uses testbed experiments to evaluate Bonito on real battery-free nodes under realistic, repeatable conditions. We start by showing in Section 4.5.2 how Bonito dynamically adjusts the connection interval to changes in the nodes' charging times to maintain long-running connections. Next, in Section 4.5.3, we compare Bonito against two baseline approaches. Finally, in Section 4.5.4, we quantify the runtime overhead of Bonito. Our experiments reveal the following key findings:

- Bonito establishes connections that outlast on average hundreds of consecutive encounters even between nodes that harvest from different types of energy sources.
- Bonito improves the throughput by 10–80× compared with the current state of the art. It achieves this by consciously keeping the connection interval as short as possible while maintaining a high success rate that agrees to within 1 % of the requested target probability.
- Depending on the distribution model, Bonito consumes between 4 % and 25 % of the energy available per wake-up on our nodes. The energy cost of losing a connection is 1000× higher than the energy overhead of Bonito.

4.5.1 Testbed and Settings

We connect two battery-free nodes (see Figure 4.11) to two Shepherd observers [44]. In addition to recording spatio-temporal harvesting traces (see Section 4.2), Shepherd can also replay previously recorded traces and monitor the behavior of connected battery-free devices. The observers synchronously replay for all 60 links in our datasets (see Table 4.1) the two corresponding energy-harvesting traces. At the same time, the observers log the serial output and GPIO events of the attached nodes, which we use to compute performance metrics. In total, we collect measurements from 218 hours of testbed experiments.

For the stairs, office, and washer scenarios, we replay the recorded energy-harvesting traces as is. When using the original traces from the cars and jogging scenarios, however, we were not able to collect sufficient data points. The reason is that the piezoelectric harvesters were selected and tuned for the frequency and amplitude of the washer scenario, which led to a relatively low harvesting power in the cars and jogging scenarios, as evident from the small number of wake-ups in Table 4.1. Because it can take thousands of wake-ups until two nodes encounter each other, we had to scale the cars and jogging traces by a factor of five to allow for a meaningful evaluation. Note that this does not change the dynamics and shape of the charging time distributions,

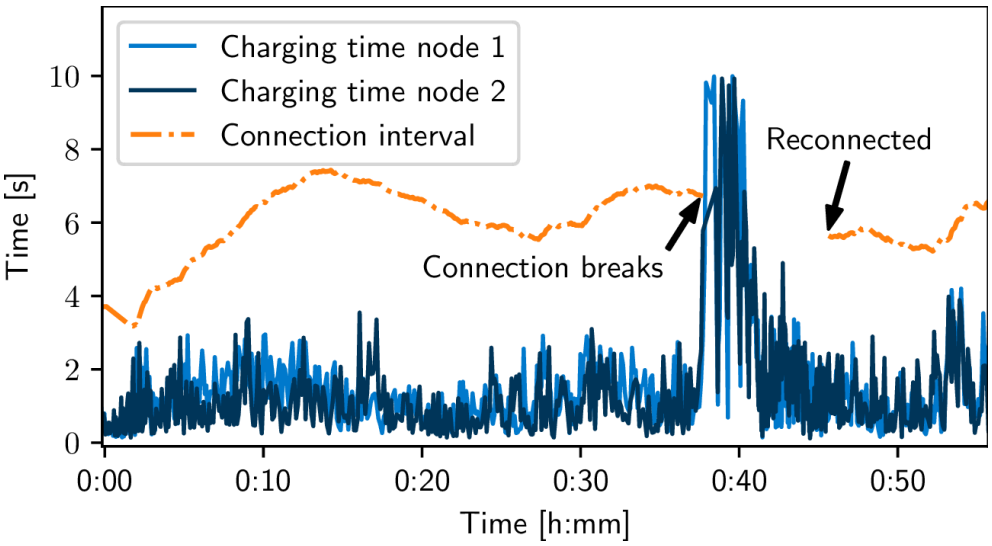


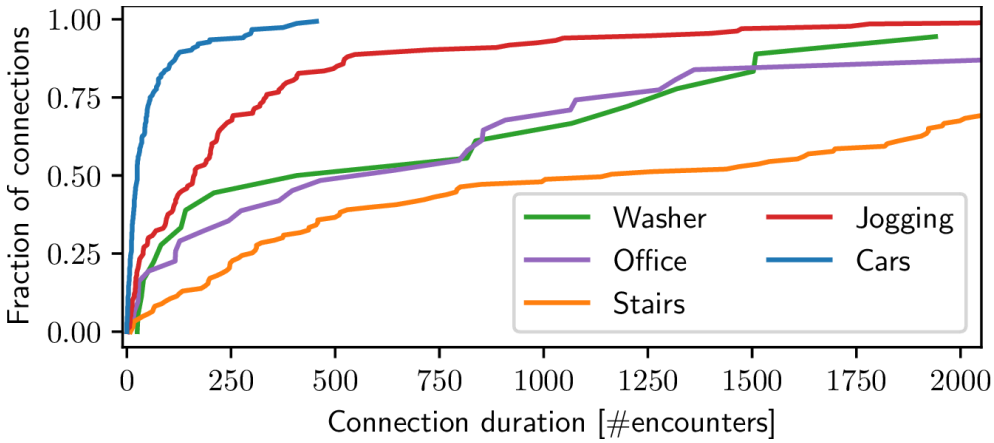
FIGURE 4.14: Real-world trace from testbed experiments showing the charging times of two nodes from the cars dataset. By dynamically adjusting the connection interval, Bonito maintains a connection for 37 min until the cars leave the highway and enter stop-and-go traffic; the charging times increase dramatically and the connection breaks.

nor does it affect relative performance when comparing different approaches.

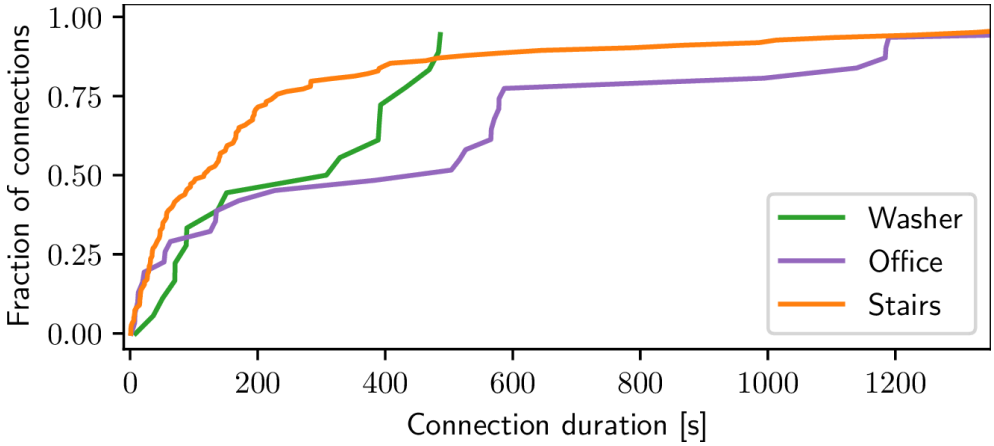
In all experiments, we configure Bonito with a target probability of $p = 0.99$. We use a learning rate of $\eta = 0.01$ for the normal and exponential models and $\eta = 0.001$ for GMM, which we found to perform well in a wide range of scenarios.

4.5.2 Maintaining Long-running Connections

We begin by looking at how well Bonito can maintain a connection between battery-free devices. As an illustrative example, Figure 4.14 shows the charging times of two nodes from the cars dataset and the connection interval determined by Bonito over the course of 55 min. Bonito successfully maintains the connection for more than half an hour by dynamically adjusting the connection interval based on the continuously updated models of the nodes' charging time distributions. Then, after around 37 min, the two cars driving in convoy exit



(A) cdf of connection duration in terms of number of encounters.



(B) cdf of connection duration in terms of time.

FIGURE 4.15: Bonito maintains connections over hundreds of encounters even in challenging scenarios with different types of energy sources.

the highway and enter stop-and-go traffic. As a result, the charging times increase suddenly and exceed the connection interval—the connection is lost. At this point, the nodes switch over to executing the Find neighbor discovery protocol and successfully reconnect after roughly 10 min. Afterward, Bonito takes over and again maintains a connection for several minutes.

Figure 4.15a plots for all datasets the cdf of the connection duration in terms of the number of encounters, while Figure 4.15b plots it in terms of time for the unscaled datasets (see Section 4.5.1). Overall, we find that in 90 % of the cases, the nodes stay connected for at least 30 consecutive encounters, and 40 % of the connections last for 800

encounters or more. This demonstrates that Bonito enables, for the first time, reliable and efficient communication between intermittently powered nodes.

4.5.3 Bonito versus Baseline Approaches

We now compare Bonito against two baseline approaches:

- Greedy: This is the current state of the art. Using Greedy, nodes wake up and attempt to communicate as soon as they reach the minimum required energy level. Greedy is the prevalent execution model in the intermittent computing literature [23, 88] as it maximizes the effective duty cycle of a battery-free device.
- Modest: As a complementary approach to Greedy, we design Modest. Using Modest, each node keeps track of the maximum observed charging time c_{max}. During an encounter, two nodes i and j share their current maximum charging times $c_{max,i}$ and $c_{max,j}$, and agree to meet again after a common connection interval of $T_C = \max(c_{max,i}, c_{max,j})$.

Our comparison uses two end-to-end metrics that also account for periods where Find runs to establish a first encounter after a connection loss or power failure. *Throughput* is the number of packets delivered from one node to another node per time unit. Note that traffic is always bi-directional, that is, the same number of packets is also delivered in the other direction (see Figure 4.13). *Latency* is the time between two consecutive packet exchanges. We also consider *success rate*, which is the ratio of successfully arranged encounters to the total number of trials when using Greedy, Modest, or Bonito.

Figure 4.16 plots for each dataset the throughput gains of Bonito and Modest over Greedy. We see that Bonito improves the throughput by 10–80×. For example, for the stairs dataset, Bonito achieves a throughput of $15.18\,\mathrm{pkt\,s^{-1}}$ versus $0.33\,\mathrm{pkt\,s^{-1}}$ with Greedy. Modest outperforms Greedy across the board, too, but often falls far short of Bonito's throughput.

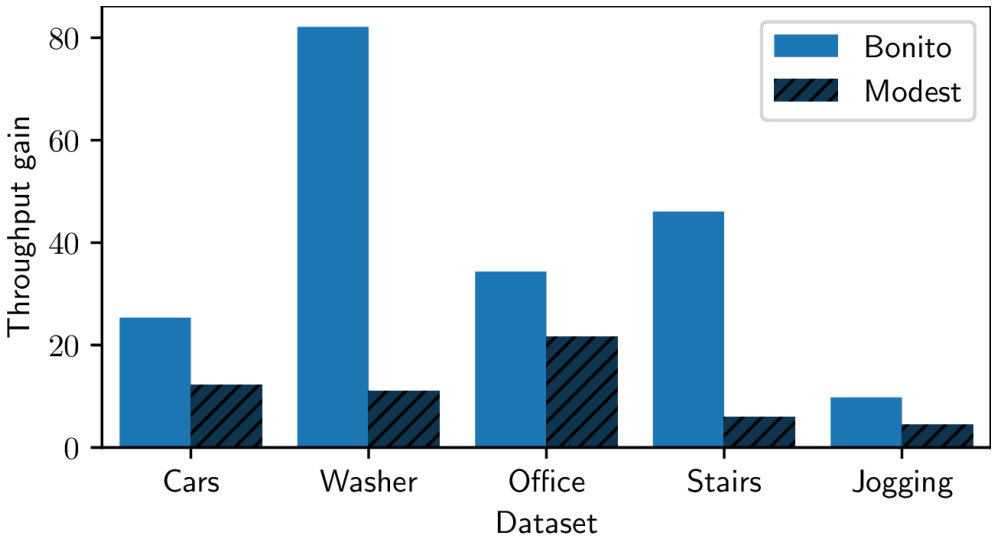
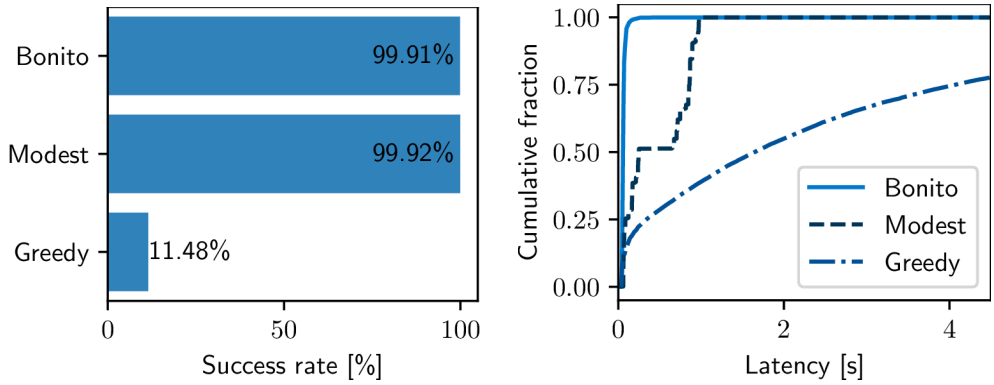


FIGURE 4.16: Throughput improvement over Greedy. By maintaining connections over many wake-ups, the average number of encounters with Bonito is at least an order of magnitude higher than with Greedy.



(A) Success rate (B) Latency

FIGURE 4.17: Detailed comparison of performance metrics for the stairs scenario. Bonito achieves a high success rate that is on a par with the Modest approach, while providing a significantly lower latency.

To understand the reasons for the significant performance differences among the different approaches, we plot in Figure 4.17 success rate and latency for the stairs dataset. As the charging times vary across time and space, Greedy achieves a low success rate of only 11.48 % (see Figure 4.17a). This means that in 9 out of 10 cases the nodes lose the connection right after the first encounter. Every time the connection is lost, the nodes cannot communicate until they reconnect, causing excessively long latencies as visible in Figure 4.17b. Instead, Modest chooses the connection interval highly conservatively, which leads to a high success rate of 99.92 % but also long latencies. Bonito provides much shorter latencies at almost the same high success rate, which agrees to within 1 % of the requested target probability. By aiming to keep the connection interval short and to avoid the latency associated with reconnecting after a connection loss, Bonito significantly increases the end-to-end throughput compared with the two baseline approaches.

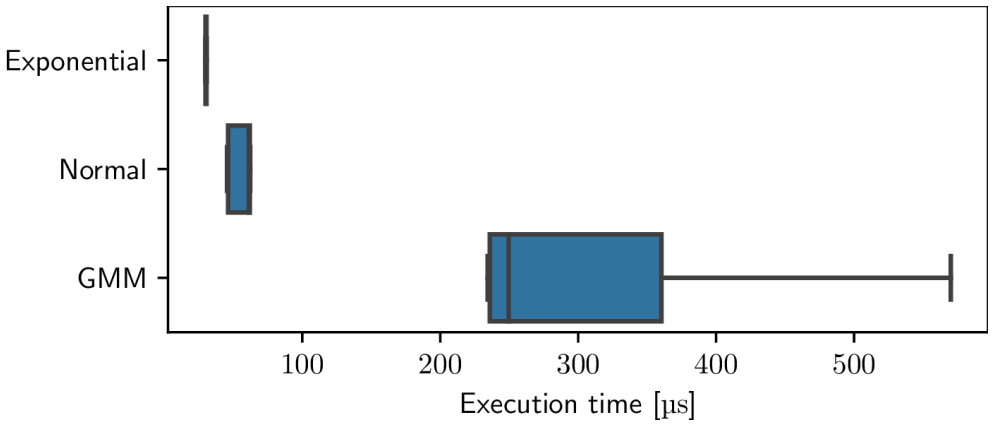


FIGURE 4.18: Distribution of execution times on our battery-free node when computing the inverse joint cdf. The execution time depends on the number of model parameters and varies with the number of bisection steps needed to satisfy the required tolerance.

4.5.4 Bonito's Runtime Overhead

Next, we evaluate the runtime overhead of Bonito based on the logs from the testbed experiments. The overhead can be broken down into three components: (*i*) updating the model parameters using SGD, (*ii*)

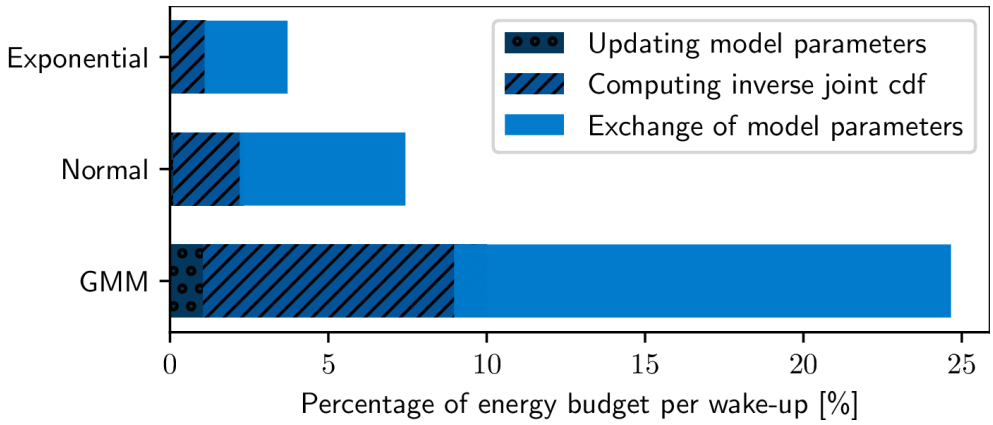
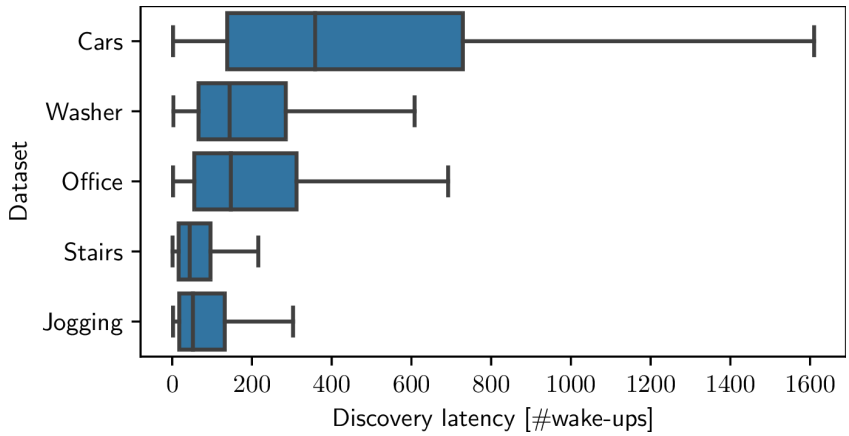


FIGURE 4.19: The energy overhead of Bonito ranges between 4 % and 25 % of the energy available per wake-up on our nodes. In absolute terms, the cost to recover from a lost connection is 1000× higher.

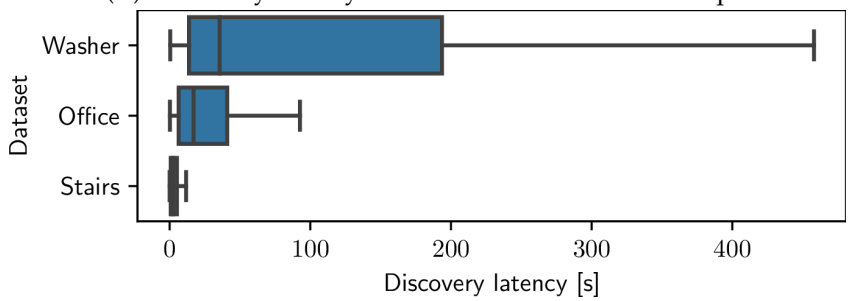
exchange of the model parameters over wireless during an encounter, and (*iii*) computing the inverse joint cdf to obtain the connection interval.

The time required to update the model is constant: 1.3 µs for exponential, 3.2 µs for normal, and 28.8 µs for GMM. This constitutes up to 2.8 % of the around 1 ms active time per wake-up. Similarly, the airtime to exchange 4, 8, or 24 bytes of model parameters is fixed and determined by the bitrate of the BLE radio. By contrast, Figure 4.18 shows that the time to compute the inverse joint cdf varies depending on the number of bisection steps required to reach the desired tolerance.

In terms of energy, our battery-free nodes have an energy budget of 27.5 µJ per wake-up. Figure 4.19 shows for each model the median percentage of energy budget spent by Bonito. We can see that the required energy mainly depends on the number of model parameters and the computational complexity of evaluating the inverse joint cdf. In the worst case, for GMM, Bonito consumes 7.1 µJ, which amounts to about 25 % of the available energy per wake-up. To set this into perspective, Figure 4.20a plots for all datasets the time it takes for two nodes to synchronize with the Find neighbor discovery protocol [46] in terms of the number of wake-ups, while Figure 4.20b plots it in terms of time



(A) Discovery latency in terms of number of wake-ups.



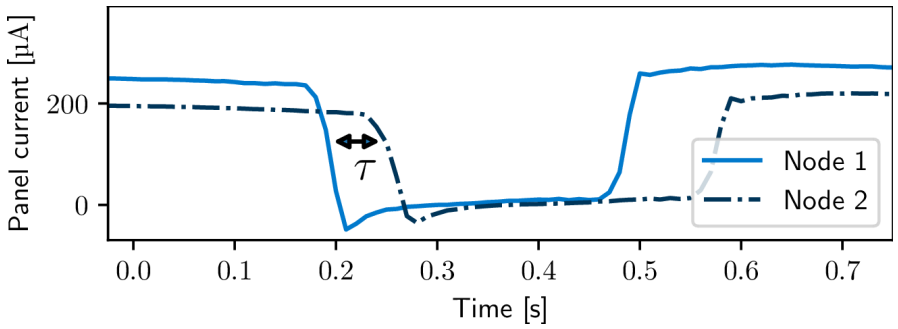
(B) Discovery latency in terms of time.

FIGURE 4.20: Synchronizing two devices with the Find neighbor discovery protocol takes a long time and consumes significant energy. Using Bonito, devices can establish long-running connections to periodically exchange data without the need to resynchronize.

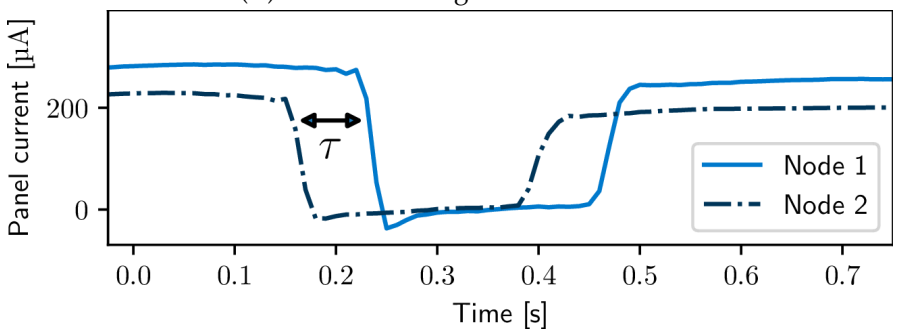
for the unscaled datasets (see Section 4.5.1). On average it takes 283 wake-ups, or 7782.5 µJ, to synchronize after a lost connection—1000× more than the energy required by Bonito to maintain a connection. This demonstrates that, overall, the absolute energy costs of Bonito are well spent.

4.6 Case Study: Occupancy Monitoring

Occupancy monitoring is essential to save energy in homes and commercial buildings [26, 38]. Recently, it has also become an important tool to manage the spread of infectious diseases, such as SARS-CoV2 [101]. To assess the potential of Bonito for real-world battery-free applications, we conduct an occupancy monitoring case study with our prototype nodes.



(A) Person entering the room: $\tau > 0$



(B) Person leaving the room: $\tau < 0$

FIGURE 4.21: The shadow of a person passing ambient light harvesting devices on a doorframe causes a distinct temporal pattern in the solar panel current. By comparing the times of the onset of the shadowing on the two nodes, we can determine the direction of movement.

Occupancy sensor. To efficiently count the number of people in a room, we use the solar panel as a sensor [66, 114] to detect when a person enters or leaves the room. Figure 4.21 shows the solar panel current of two nodes mounted next to each other on a doorframe (see Figure 4.22), when a person enters the room in Figure 4.21a and when a person leaves the room in Figure 4.21b. To detect the direction of movement, the nodes record the time when they detect the onset of the shadowing by the person. Then the nodes exchange the recorded times and compute the time difference τ. The sign of τ indicates the direction.

Setup. We mount two battery-free nodes, each equipped with one IXYS SM141K06L solar panel next to each other on the doorframe at the entrance of an office room, as shown in Figure 4.22. The nodes sample the solar panel current with a sampling rate of 1 kHz, and record

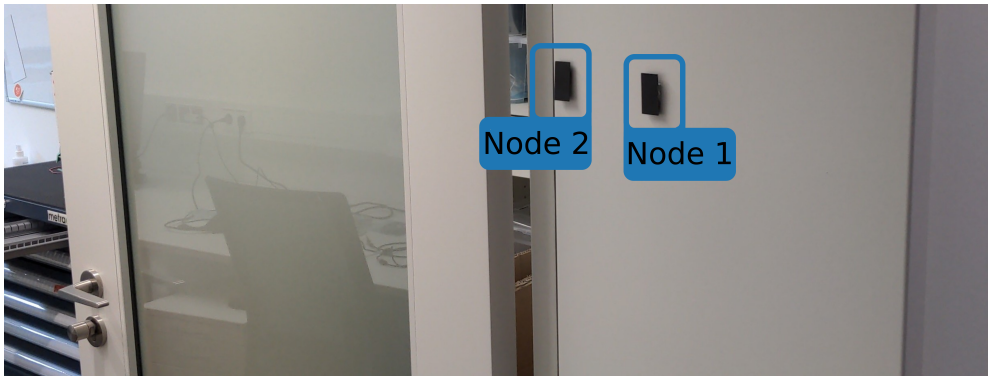


FIGURE 4.22: Two of our battery-free nodes are attached to the doorframe and harvest energy from ambient light. Thanks to Bonito, the nodes can communicate in a timely and reliable fashion, allowing them to count the number of people entering and leaving the room.

the time when the solar panel current falls below 87.5 % of its average value. The nodes run Bonito and insert the timestamp of detected events into the packets. Together with logging information (charging time, connection interval, etc.) every packet carries 26 B of application data.

Because the clocks of the two nodes are not synchronized, timestamps are transmitted relative to the start of the corresponding packet. To this end, nodes measure the time between the detected event and the start of the transmission and insert the result into the packet. The receiving node timestamps the reception of the packet and converts the contained relative timestamp to its local clock. Finally, by relating a received timestamp to the timestamp of the corresponding event that was recorded locally, the nodes compute the time difference τ.

The nodes transmit the result over wireless to an nRF52840 development board that serves as a base station. We configure the base station to timestamp the reception of packets containing a detected event and button presses of two on-board push buttons, one for each direction. Four participants randomly enter and leave the room one by one. Another person records ground truth by pressing the corresponding button on the nRF52840 board precisely when a person passes through the doorframe.

		Ground truth		
		In	Out	No event
Recorded	In	30	0	1
	Out	0	31	0
	No event	1	0	0

TABLE 4.2: By collaborating, the battery-free nodes classified people entering and leaving the room with an average accuracy of 96.83 %.

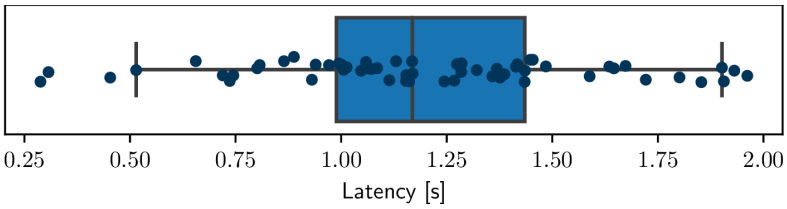


FIGURE 4.23: Due to the low communication latency provided by Bonito, the system reported detected events both timely and accurately.

Results. The confusion matrix in Table 4.2 shows that the system correctly classified 60 out of 61 events, corresponding to an accuracy of 96 %. It missed just one in-event, and falsely reported an in-event and an out-event for a single in-event.

Figure 4.23 plots the latency in terms of the time between a button press and the reception of the detected event at the base station. The median latency was 1.2 s and all events were reported within less than 2 s. Over the course of the experiment, the two nodes successfully exchanged 10.56 kB of application data for an application-level throughput of 28.38 B s^{-1}.

Figure 4.24 shows a ten-second excerpt from the experiment. The markers indicate the charging times of the nodes. Solid vertical lines indicate button presses (ground truth); dashed vertical lines indicate when an event was received at the base station. We can observe that, right after the received out-event, node 1 reports an exceptionally high charging time of 210 ms. This happens when the shadowing by a person occurs while a node charges its capacitor: The shadowing reduces the energy input for a short time, which prolongs the recharge. Nevertheless, by keeping the connection interval at around 700 ms, Bonito provides a stable connection despite such dynamics.

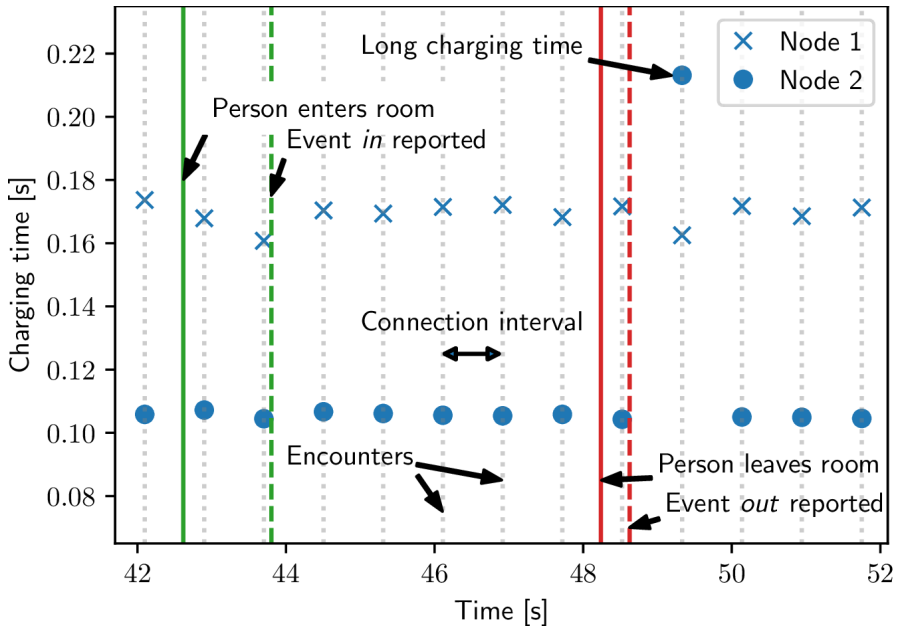


FIGURE 4.24: Example trace from the occupancy monitoring case study. The system correctly classifies and reports events to the base station. With Bonito, the connection interval is chosen large enough to sustain outliers of the charging time in response to transient shadowing.

4.7 Discussion

Bonito is the first connection protocol for battery-free devices. It enables two devices to communicate efficiently and reliably by dynamically adapting the connection interval to changes in the devices' energy availability. In this section, we discuss limitations and opportunities for extending Bonito.

From connections to networks. The ability to efficiently and reliably exchange data between two devices is the fundamental building block required to form large wireless networks consisting of multiple battery-free devices. A number of trade-offs and challenges arise from each of the possible approaches to move from the two-node setting to larger networks, which could be explored by future work. For example, devices may sequentially connect with their neighbors or devices may try to establish Bonito connections with one common connection interval between multiple devices.

Communication with battery-powered devices. While we focus on communication between two battery-free devices, Bonito is also useful for effective communication from battery-free to battery-powered devices. For example, a battery-free tag may want to transmit data to a user's smartphone or to a battery-powered gateway in a wireless sensor network. Because battery-powered devices are in control of their wake-up times, any connection interval works for them. Thus, instead of computing the inverse joint cdf of the charging time distribution of both devices, it is sufficient to compute the inverse cdf of the charging time of the battery-free device in order to determine a connection interval that works for both devices.

Model accuracy. The goodness-of-fit of the learned charging time model critically affects the performance of Bonito. With perfect knowledge of the underlying distribution, Bonito would compute the minimum connection interval feasible for the requested target probability. Overestimating the real distribution leads to increased delay, while underestimation reduces reliability. If the distribution is so complex that a large number of model parameters or a non-parametric model (e.g., a deep neural network) would be required to accurately capture this complexity, then the limited resources on a battery-free device may not be sufficient to learn the model online.

Exploiting statistical dependence. In the current implementation, Bonito assumes statistical independence of the nodes' charging time distributions to compute a connection interval without prior knowledge of the statistical properties of the joint charging time distribution. After establishing a connection, the devices can record observations of the joint distribution and could attempt to exploit statistical dependence between their charging times, possibly improving communication efficiency and reliability.

4.8 Related Work

Intermittent computing. The thriving research area of intermittent computing has made great strides in recent years, including the first real deployments of battery-free sensors [3]. This achievement rests upon techniques that ensure forward progress [91], consistent peripheral state [16], and a reliable notion of time [27] despite frequent and random power failures. This line of research is highly relevant but completely orthogonal to our work as it deals exclusively with intermittency issues on *individual* devices and, if at all, considers communication with continuously powered base stations [120].

Battery-free device-to-device communication. Prior work on battery-free wireless device-to-device communication is mainly theoretical [70, 145, 84], studying the energy trade-offs for different scheduling, transmission, and decoding policies. Recent work discusses middleware and applications for networks of intermittently powered devices, yet explicitly leaves the question of how to communicate between the devices as an open problem [142, 82]. A simulative study also acknowledges the sheer difficulty of sychronizing the wake-up times of intermittently powered devices and proposes to communicate an energy state via an always-on backscatter radio, without demonstrating a real implementation or experiments [130]. Similar to Bonito, a recent theoretical work proposes to let nodes agree on a future point in time when they become active to increase communication throughput [137]. This time is computed based on a moving average of previous charging times, whereas Bonito lets the user explicitly trade reliability against delay by taking into account the charging time distributions.

In terms of practical work, tag-to-tag backscatter communication has mainly focused on physical-layer issues and considers intermittency an orthogonal problem [144, 111, 92, 83]. Instead, the Find neighbor discovery protocol explicitly addresses the intermittency problem and shows that by delaying wake-ups by a random time battery-free nodes

can encounter each other faster [46]. We use Find to bootstrap efficient and reliable device-to-device communication with Bonito. Concurrently to our work, a protocol was proposed and implemented that lets devices "die early" when no packet is received to preserve energy and maximize the number of wake-ups [29].

Delay-tolerant networking (DTN). DTN studies networks that are only intermittently connected because of, for example, node failures, mobile users, and power outages [14, 40]. Both DTN and Bonito have the same high-level goal: effective communication in intermittently connected networks. However, DTN and Bonito address orthogonal problems toward the same end goal. While DTN is concerned with forwarding, routing, naming, in-network storage, and optimization of node trajectories to generate encounters in the spatial domain, Bonito aims to generate encounters in the time domain between nodes that are spatially close to each other. Whether concepts from the DTN literature could be applied on top of Bonito is an interesting question for future research.

Energy-aware MAC protocols. Numerous MAC protocols have been proposed for ad-hoc and sensor networks [32]. These protocols turn the radio off most of the time, and power it up only to send or receive a packet. The goal is to achieve a desired network lifetime by maintaining a certain average duty cycle. A fundamental assumption of these protocols is that the radio can be powered up *at any point in time*, which is exploited to reduce idle listening by flexibly scheduling communication among nodes. This is, however, not possible in a battery-free system, where devices are unavailable whenever the capacitor voltage is below a certain threshold, which renders existing energy-aware MAC protocols ineffective.

4.9 Conclusions

We have presented Bonito, a connection protocol for wireless battery-free devices. By adapting the connection interval to the different and

time-varying charging times of intermittently powered nodes, Bonito maintains long-running connections that provide significantly better throughput, latency, and reliability than the state of the art. We have evaluated Bonito by implementing it on a battery-free prototype, conducting testbed experiments with real energy-harvesting traces from diverse scenarios, and demonstrating its utility in an occupancy monitoring case study. With Bonito, we contribute a prime communication primitive, device-to-device unicast, that brings the capabilities of battery-free systems one step closer to those known from today's battery-supported systems.

Availability

The data described in Section 4.2 and a Python implementation of the Bonito protocol from Section 4.3 are available under a permissive MIT license at `https://bonito.nes-lab.org/`.

Acknowledgments

We thank Sarah Nollau, Ingmar Splitt, Friedrich Schmidt, Justus Paulick, and Lebenshilfe Altenburg e. V. for supporting the data collection campaign, and all participants of the occupancy monitoring case study. Thanks also to the anonymous reviewers, and to our shepherd, Shyam Gollakota. This work was supported by the German Research Foundation (DFG) within the Emmy Noether project NextIoT (grant ZI 1635/2-1) and the Center for Advancing Electronics Dresden (cfaed).

Appendix: Gradient Equations

Exponential distribution. The derivative of the log-likelihood function is given by:

$$\mathcal{L}(\lambda) = \log\left(\lambda \cdot \exp(-\lambda x)\right) = \log\lambda - \lambda x_i \tag{4.12}$$

$$\nabla\mathcal{L}(\lambda) = \frac{1}{\lambda} - x_i \tag{4.13}$$

Calculating the *natural gradient* by defining the step size in terms of the Kullback-Leibler divergence in the distribution space has been shown to speed up convergence in many cases [4]. We obtain the natural gradient by multiplying the regular gradient from (4.13) with the inverse of the Fisher Information Matrix of the exponential distribution M_{exp}:

$$M_{exp} = \lambda^{-2} \tag{4.14}$$

$$\frac{\partial\mathcal{L}}{\partial\lambda} = \left[M_{exp}\right]^{-1} \cdot \frac{1}{\lambda} - x_i = \lambda - \lambda^2 \cdot x_i \tag{4.15}$$

Gaussian mixture model. We adopt the gradient equations from [129]: Let $f(x_i, \mu, \sigma^2)$ be the probability density function of the standard normal distribution. The responsibility function $r(x_i, k)$ quantifies the contribution of the k-th component to the model:

$$r(x_i, k) = \frac{\rho_k \cdot f(x_i, \mu_k, \sigma_k^2)}{\sum_l^K \left(\rho_l \cdot f(x_i, \mu_l, \sigma_l^2)\right)} \tag{4.16}$$

The update equations for the model parameters for the k-th component are then:

$$\frac{\partial\mathcal{L}}{\partial\rho_k} = r(x_i, k) - \rho_k \tag{4.17}$$

$$\frac{\partial\mathcal{L}}{\partial\mu_k} = \frac{1}{\rho_k} \cdot r(x_i, k) \cdot (x_i - \mu_k) \tag{4.18}$$

$$\frac{\partial\mathcal{L}}{\partial\sigma_k^2} = \frac{1}{\rho_k} \cdot r(x_i, k) \cdot (x_i - \mu_k)^2 - \sigma_k^2 \tag{4.19}$$

Normal distribution. We consider the special case of a gaussian mixture model with a single component and also use the equations from [129]:

$$\frac{\partial \mathcal{L}}{\partial \mu} = (x_i - \mu) \tag{4.20}$$

$$\frac{\partial \mathcal{L}}{\partial \sigma^2} = (x_i - \mu)^2 - \sigma^2 \tag{4.21}$$

Postscript. The methods presented in Chapter 3 are suitable to establish communication without prior synchronization, but the low throughput prevents their use as a data transport layer. With significantly increased throughput, Bonito instead allows efficient exchange of application data between two intermittently powered devices. Preliminary simulations confirm that Bonito can indeed be used as the communication primitive for large-scale battery-free mesh networks.

5

Conclusions

Wireless sensors yield valuable insights into physical processes that were previously impossible to observe. From building automation to habitat monitoring these insights help to optimize energy consumption, improve human well-being and deepen our understanding of the environment. Powering billions of such devices with batteries, however, is unsustainable and severely limits the duration and scale of deployments. Replacing batteries with energy harvesters and tiny capacitors reduces the environmental impact of the devices and enables large-scale deployments that have the potential to operate maintenance-free for decades. Establishing and operating such vast infrastructure requires coordination and communication among the devices. This is extremely challenging, because battery-free devices operate intermittently, i.e., they are forced to remain inactive for long, varying times before becoming active for short durations.

5.1 Contribution

As a result, the challenges and opportunities of operating groups of battery-free devices have so far remained largely unexplored, despite being frequently mentioned as one of the most urging gaps in the

space of battery-free systems. There were no dedicated tools and few practical approaches to connecting battery-free devices with each other. The contributions presented in this thesis have radically changed this and have enabled the end-to-end evaluation of first applications that rely on battery-free device-to-device communication.

Testbed. One of our key hypotheses was that understanding real-world spatio-temporal energy availability is key to developing practical methods for battery-free networks. This hypothesis was confirmed once Shepherd enabled us to record and analyze high resolution spatio-temporal energy harvesting traces. The ability to test different approaches against realistic, repeatable energy conditions has significantly accelerated the design, implementation and evaluation of the protocols presented in this thesis. By making hardware and software open-source, we provide a tool for the community that helps to advance the field of battery-free communication beyond our own work.

Neighbor Discovery. In contrast to traditional battery-powered networks, where neighbor discovery is done once at the beginning of a deployment or after the rare event of a node failure, battery-free devices need to resynchronize and rediscover after every extended period without energy input. Thus, to optimize this critical aspect, we have presented Find, the first neighbor discovery protocol for battery-free networks. Find adds random wake-up delays before devices become active to break up interleaving patterns between the wake-up times of two devices. It is agnostic to the energy harvesting modality and the type of wireless communication. To further accelerate neighbor discovery in indoor light harvesting scenarios, we have developed Flync, the first solution extracting a stable clock from solar harvesting current, whose amplitude changes due to powerline-induced flicker of state-of-the-art lamps. While we combine Flync with Find to speed up neighbor discovery, Flync is useful for applications beyond neighbor discovery and battery-free networking.

Connection protocol. While Find and Flync provide devices with an initial encounter, they are not efficient when used as the sole means of communication. To improve on this aspect we have designed Bonito, the first connection protocol that exploits the opportunity of the first encounter to enable two battery-free nodes to establish a *connection* that lasts across multiple consecutive encounters. We collected 32 h of energy-harvesting traces from 5 different scenarios and our analysis revealed that, depending on the scenario and energy-harvesting modality, the charging time of a battery-free device approximately follows well-known probability distributions. With Bonito, devices *continuously learn and track* the parameters of a model that approximates the distribution of locally observed charging times against non-stationary effects (*e.g.*, changes in mean or variance) and exchange at every encounter their current model parameters to jointly adapt the connection interval. Bonito enables, for the first time, reliable and efficient communication between intermittently powered battery-free devices. By sharing the extensive datasets recorded with Shepherd, we encourage the community to build upon our work.

Our central findings are that ...

- ...contrary to popular believe, direct wireless communication between battery-free, intermittently powered devices is possible even with tiny energy storage capacity and highly limited energy availability.
- ...efficient communication requires understanding and incorporating spatio-temporal energy availability in the design of network protocols.
- ...introducing dynamic wake-up delays is an effective method to control and synchronize the wake-up times of battery-free devices.

5.2 Future directions

Our research has not only provided answers to some of the most urgent issues on the path to battery-free networks, but has also raised exciting new research questions.

Deeper understanding of energy environments. With Shepherd we developed the necessary tool to collect and analyze real-world harvesting traces. Our data analysis was driven by network protocol design and has focused on the charging times of devices which are a derivative of the harvested energy. Analyzing statistical properties of the actual harvesting data may yield additional starting points for networking battery-free devices. For example, in Bonito, we assume that charging times of devices are independent, but preliminary evaluation suggests that energy harvested on close-by devices often correlates. Periodic data exchange allows devices to track and exploit such spatio-temporal correlations online and may lead to more informed decisions, improving communication efficiency and reliability. Furthermore, we are convinced that data collected with Shepherd presents an invaluable source of information beyond the specific problems that we considered. For example, devices placed in a common energy environment share access to a secret that may be used for securing data and communication.

Battery-free mesh networks. With Find, Flync and Bonito we have contributed the communication primitives to efficiently and reliably exchange data between two devices. This is the fundamental building block required to form large wireless networks consisting of multiple battery-free devices. A number of trade-offs and challenges arise from each of the possible approaches to move from the two-node setting to larger networks, which could be explored by future work. For example, devices may use Find to sequentially connect with their neighbors and establish pair-wise Bonito connections. By relaying packets from other nodes, they could form mesh networks that would allow sending messages far beyond the communication range of individual nodes. A different approach to larger scale networks could use Flync

to establish a common time-grid where devices wake up randomly in each slot or according to a global schedule in order to exchange data with their neighbors. Owing to the statistical analysis in the context of Bonito, we now have a basic understanding of the protocol performance in the two node case. The effectiveness of forwarding realistic data traffic across multiple hops instead is not easily derived, yet highly relevant for practical applications.

Integration with application requirements. When communicating with continuously powered basestations, previous work on battery-free devices had little timing constraints for scheduling communication tasks. When communicating between battery-free devices instead, timing becomes critical. In our work, we put communication first and assigned whatever energy was left to the application. Clearly, this isn't feasible in many practical settings, where application tasks might have strict timing or energy constraints. Co-scheduling communication and application tasks under the severe constraints imposed by energy availability and storage capacity is a challenge and will open up the field to exciting new research.

Bibliography

[1] Henko Aantjes, Amjad Y. Majid, and Przemysław Pawełczak. A testbed for transiently powered computers. *arXiv:1606.07623*, 2016.

[2] Kofi Sarpong Adu-Manu, Nadir Adam, Cristiano Tapparello, Hoda Ayatollahi, and Wendi Heinzelman. Energy-Harvesting Wireless Sensor Networks (EH-WSNs): A Review. *ACM Transactions on Sensor Networks*, 14(2), 2018.

[3] Mikhail Afanasov, Naveed Anwar Bhatti, Dennis Campagna, Giacomo Caslini, Fabio Massimo Centonze, Koustabh Dolui, Andrea Maioli, Erica Barone, Muhammad Hamad Alizai, Junaid Haroon Siddiqui, and Luca Mottola. Battery-less zero-maintenance embedded sensing at the mithræum of circus maximus. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems (SenSys)*, 2020.

[4] S. Amari and S.C. Douglas. Why natural gradient? In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1998.

[5] D'mello Anasia. Energy harvesting brings 'battery that never dies' claim for iot applications. Online at https://www.iot-now.com/2020/10/28/105718-energy-harvesting-brings-battery-that-never-dies-claim-for-iot-applications/, 2020.

[6] Science Communication Unit at University of the West of England. Towards the battery of the future. *Science for Environment Policy*, 2018.

[7] Abu Bakar and Josiah Hester. Making sense of intermittent energy harvesting. In *Proceedings of the 6th ACM International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems (ENSSys)*, 2018.

[8] Mehedi Bakht, Matt Trower, and Robin Hilary Kravets. Searchlight: Won't you be my neighbor? In *Proceedings of the 18th ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2012.

[9] Domenico Balsamo, Alex S. Weddell, Geoff V. Merrett, Bashir M. Al-Hashimi, Davide Brunelli, and Luca Benini. Hibernus: Sustaining Computation During Intermittent Supply for Energy-Harvesting Systems. *IEEE Embedded Systems Letters*, 7(1), 2015.

[10] Célestin Banza Lubaba Nkulu, Lidia Casas, Vincent Haufroid, Thierry De Putter, Nelly D. Saenen, Tony Kayembe-Kitenge, Paul Musa Obadia, Daniel Kyanika Wa Mukoma, Jean-Marie Lunda Ilunga, Tim S. Nawrot, Oscar Luboya Numbi, Erik Smolders, and Benoit Nemery. Sustainability of artisanal mining of cobalt in DR Congo. *Nature Sustainability*, 1(9), September 2018.

[11] James Bates, Marc Beaulieu, Michael Miller, and Joseph Paulus. Reaching the highest reliability for tantalum capacitors. Technical report, AVX Corporation, 2013.

[12] Naveed Anwar Bhatti, Muhammad Hamad Alizai, Affan A. Syed, and Luca Mottola. Energy harvesting and wireless transfer in sensor network applications: Concepts and experiences. *ACM Transactions on Sensor Networks*, 12(3), 2016.

[13] Naveed Anwar Bhatti and Luca Mottola. HarvOS: Efficient code instrumentation for transiently-powered embedded sensing. In *Proceedings of the 16th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2017.

[14] Sanjit Biswas and Robert Morris. ExOR: opportunistic multi-hop routing for wireless networks. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication*

on the applications, technologies, architectures, and protocols for computer communication (SIGCOMM), 2005.

[15] Léon Bottou. Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade: Second Edition*. Springer Berlin Heidelberg, 2012.

[16] Adriano Branco, Luca Mottola, Muhammad Hamad Alizai, and Junaid Haroon Siddiqui. Intermittent asynchronous peripheral operations. In *Proceedings of the 17th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2019.

[17] R. P. Brent. An algorithm with guaranteed convergence for finding a zero of a function. *The Computer Journal*, 14(4), 1971.

[18] Bernhard Buchli, Felix Sutton, Jan Beutel, and Lothar Thiele. Towards Enabling Uninterrupted Long-Term Operation of Solar Energy Harvesting Embedded Systems. In *Proceedings of the International Conference on Embedded Wireless Systems and Networks (EWSN)*, 2014.

[19] Lin Chen and Kaigui Bian. Neighbor discovery in mobile sensing applications: A comprehensive survey. *Ad Hoc Networks*, 48, 2016.

[20] Albert Cohen, Xipeng Shen, Josep Torrellas, James Tuck, Yuanyuan Zhou, Sarita Adve, Ismail Akturk, Saurabh Bagchi, Rajeev Balasubramonian, Rajkishore Barik, Micah Beck, Ras Bodik, Ali Butt, Luis Ceze, Haibo Chen, Yiran Chen, Trishul Chilimbi, Mihai Christodorescu, John Criswell, Chen Ding, Yufei Ding, Sandhya Dwarkadas, Erik Elmroth, Phil Gibbons, Xiaochen Guo, Rajesh Gupta, Gernot Heiser, Hank Hoffman, Jian Huang, Hillery Hunter, John Kim, Sam King, James Larus, Chen Liu, Shan Lu, Brandon Lucia, Saeed Maleki, Somnath Mazumdar, Iulian Neamtiu, Keshav Pingali, Paolo Rech, Michael Scott, Yan Solihin, Dawn Song, Jakub Szefer, Dan Tsafrir, Bhuvan Urgaonkar, Marilyn Wolf, Yuan Xie, Jishen Zhao,

Lin Zhong, and Yuhao Zhu. Inter-disciplinary research challenges in computer systems for the 2020s. Technical report, USA, 2018.

[21] Alexei Colin, Graham Harvey, Brandon Lucia, and Alanson P. Sample. An Energy-interference-free Hardware-Software Debugger for Intermittent Energy-harvesting Systems. In *Proceedings of the 21st International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2016.

[22] Alexei Colin and Brandon Lucia. Chain: Tasks and channels for reliable intermittent programs. In *Proceedings of the 2016 ACM International Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*, 2016.

[23] Alexei Colin, Emily Ruppel, and Brandon Lucia. A reconfigurable energy storage architecture for energy-harvesting devices. In *Proceedings of the 23rd International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2018.

[24] Pablo Corbalán and Gian Pietro Picco. Concurrent ranging in ultra-wideband radios: Experimental evidence, challenges, and opportunities. In *Proceedings of the International Conference on Embedded Wireless Systems and Networks (EWSN)*, 2018.

[25] KEMET Electronics Corporation. T52x/t530 polymer electrolytic capacitors, 2018.

[26] Stephen Dawson-Haggerty, Andrew Krioukov, Jay Taneja, Sagar Karandikar, Gabe Fierro, Nikita Kitaev, and David Culler. BOSS: Building operating system services. In *10th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2013.

[27] Jasper de Winkel, Carlo Delle Donne, Kasim Sinan Yildirim, Przemysław Pawełczak, and Josiah Hester. Reliable timekeeping for intermittent computing. In *Proceedings of the 25th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2020.

[28] Jasper de Winkel, Vito Kortbeek, Josiah Hester, and Przemysław Pawełczak. Battery-free game boy. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(3), 2020.

[29] Vishal Deep, Mathew L. Wymore, Alexis A. Aurandt, Vishak Narayanan, Shen Fu, Henry Duwe, and Daji Qiao. Experimental Study of Lifecycle Management Protocols for Batteryless Intermittent Communication. In *Proceedings of the 18th IEEE International Conference on Mobile Ad Hoc and Smart Systems (MASS)*, 2021.

[30] Farzan Dehbashi, Ali Abedi, Tim Brecht, and Omid Abari. Verification: can wifi backscatter replace RFID? In *Proceedings of the 27th ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2021.

[31] A. R. Dehghani-Sanij, E. Tharumalingam, M. B. Dusseault, and R. Fraser. Study of energy storage systems and environmental challenges of batteries. *Renewable and Sustainable Energy Reviews*, 104, 2019.

[32] I. Demirkol, C. Ersoy, and F. Alagoz. Mac protocols for wireless sensor networks: a survey. *IEEE Communications Magazine*, 44(4), 2006.

[33] Bradley Denby and Brandon Lucia. Orbital Edge Computing: Nanosatellite Constellations as a New Class of Computer System. In *Proceedings of the 25th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2020.

[34] Patricia Derler, Edward A. Lee, and Alberto Sangiovanni Vincentelli. Modeling cyber-physical systems. *Proceedings of the IEEE*, 100(1), 2012.

[35] Harsh Desai, Matteo Nardello, Davide Brunelli, and Brandon Lucia. Camaroptera: A Long-Range Image Sensor with Local Inference for Remote Sensing Applications. *ACM Transactions on Embedded Computing Systems*, 2022.

[36] Mun Choon Doddavenkatappa, Manjunath Chan and Ananda A. L. Indriya: A low-cost, 3d wireless sensor network testbed. In *Proceedings of the ICST Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom)*, 2011.

[37] Prabal Dutta and David Culler. Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications. In *Proceedings of the 6th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2008.

[38] Varick L. Erickson, Miguel Á. Carreira-Perpiñán, and Alberto E. Cerpa. Occupancy Modeling and Prediction for Building Energy Management. *ACM Transactions on Sensor Networks*, 10(3), 2014.

[39] Xenofon Fafoutis and Nicola Dragoni. ODMAC: an on-demand MAC protocol for energy harvesting - wireless sensor networks. In *Proceedings of the 8th ACM Symposium on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*, 2011.

[40] Kevin Fall. A delay-tolerant network architecture for challenged internets. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication (SIGCOMM)*, 2003.

[41] Muhammad Shoaib Farooq, Shamyla Riaz, Adnan Abid, Kamran Abid, and Muhammad Azhar Naeem. A Survey on the Role of IoT in Agriculture for the Implementation of Smart Farming. *IEEE Access*, 7, 2019.

[42] Francesco Fraternali, Bharathan Balaji, Yuvraj Agarwal, Luca Benini, and Rajesh Gupta. Pible: Battery-Free Mote for Perpetual Indoor BLE Applications. In *Proceedings of the 5th ACM Conference on Systems for Built Environments (BuildSys)*, 2018.

[43] Gerd Ulrich Gamm, Matthias Sippel, Milos Kostic, and Leonhard M. Reindl. Low power wake-up receiver for wireless sensor nodes. In *Proceedings of the 6th International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2010.

[44] Kai Geissdoerfer, Mikołaj Chwalisz, and Marco Zimmerling. Shepherd: A portable testbed for the batteryless IoT. In *Proceedings of the 17th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, pages 83–95. Association for Computing Machinery, 2019.

[45] Kai Geissdoerfer, Raja Jurdak, Brano Kusy, and Marco Zimmerling. Getting more out of energy-harvesting systems: energy management under time-varying utility with preact. In *Proceedings of the 18th ACM International Conference on Information Processing in Sensor Networks (IPSN)*, 2019.

[46] Kai Geissdoerfer and Marco Zimmerling. Bootstrapping battery-free wireless networks: Efficient neighbor discovery and synchronization in the face of intermittency. In *Proceedings of the 18th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 439–455. USENIX Association, 2021.

[47] Kai Geissdoerfer and Marco Zimmerling. Learning to communicate effectively between battery-free devices. In *Proceedings of the 19th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 419–435. USENIX Association, 2022.

[48] Graham Gobieski, Brandon Lucia, and Nathan Beckmann. Intelligence beyond the edge: Inference on intermittent embedded systems. In *Proceedings of the 24th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2019.

[49] A. Gomez, L. Sigrist, T. Schalch, L. Benini, and L. Thiele. Wearable, energy-opportunistic vision sensing for walking speed estimation. In *Proceedings of the IEEE Sensors Applications Symposium*

(SAS), 2017.

[50] Bernhard Großwindhager, Michael Stocker, Michael Rath, Carlo Alberto Boano, and Kay Römer. Snaploc: An ultra-fast uwb-based indoor localization system for an unlimited number of tags. In *Proceedings of the 18th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2019.

[51] Zhaoquan Gu, Yuexuan Wang, Qiang-Sheng Hua, and FC Lau. *Rendezvous in Distributed Systems*. Springer, 2017.

[52] Lars Hanschke, Christian Renner, Jannick Brockmann, Tobias Hamann, Jannes Peschel, Alexander Schell, and Alexander Sowarka. Light in the Box: Reproducible Lighting Conditions for Solar-Powered Sensor Nodes. In *Proceedings of the 15th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2017.

[53] Josiah Hester, Lanny Sitanayah, Timothy Scott, and Jacob Sorber. Realistic and Repeatable Emulation of Energy Harvesting Environments. *ACM Transactions on Sensor Networks*, 13(2), 2017.

[54] Josiah Hester and Jacob Sorber. Flicker: Rapid prototyping for the batteryless internet-of-things. In *Proceedings of the 15th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2017.

[55] Josiah Hester and Jacob Sorber. The Future of Sensing is Batteryless, Intermittent, and Awesome. In *Proceedings of the 15th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2017.

[56] Josiah Hester, Nicole Tobias, Amir Rahmati, Lanny Sitanayah, Daniel Holcomb, Kevin Fu, Wayne P. Burleson, and Jacob Sorber. Persistent clocks for batteryless sensing devices. *ACM Transactions on Embedded Computing Systems*, 15(4), 2016.

[57] Tingpei Huang, Haiming Chen, Li Cui, and Yuqing Zhang. EasiND: Neighbor discovery in duty-cycled asynchronous multichannel mobile WSNs. *International Journal of Distributed Sensor Networks*, 9(7), 2013.

[58] Vikram Iyer, Rajalakshmi Nandakumar, Anran Wang, Sawyer B. Fuller, and Shyamnath Gollakota. Living IoT: A Flying Wireless Platform on Live Insects. In *Proceedings of the 25th ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2019.

[59] Vikram Iyer, Vamsi Talla, Bryce Kellogg, Shyamnath Gollakota, and Joshua Smith. Inter-technology backscatter: Towards internet connectivity for implanted devices. In *Proceedings of the ACM SIGCOMM Conference*, 2016.

[60] Neal Jackson, Joshua Adkins, and Prabal Dutta. Capacity over capacitance for reliable energy harvesting sensors. In *Proceedings of the 18th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2019.

[61] Hrishikesh Jayakumar, Arnab Raha, Woo Suk Lee, and Vijay Raghunathan. Quickrecall: A hw/sw approach for computing across power cycles in transiently powered computers. *ACM Journal on Emerging Technologies in Computer Systems*, 12(1), 2015.

[62] Kang Eun Jeon, James She, Jason Xue, Sang-Ha Kim, and Soochang Park. luXbeacon—A batteryless beacon for green IoT: Design, modeling, and field tests. *IEEE Internet of Things Journal*, 6(3), 2019.

[63] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li Shiuan Peh, and Daniel Rubenstein. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet. In *Proceedings of the 10th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2002.

[64] Arvind Kandhalu, Karthik Lakshmanan, and Ragunathan (Raj) Rajkumar. U-connect: A low-latency energy-efficient asynchronous neighbor discovery protocol. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2010.

[65] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava. Power management in energy harvesting sensor networks. *ACM Transactions on Embedded Computing Systems (TECS)*, 6(4), 2007.

[66] Sara Khalifa, Mahbub Hassan, Aruna Seneviratne, and Sajal K. Das. Energy-Harvesting Wearables for Activity-Aware Services. *IEEE Internet Computing*, 19(5), 2015.

[67] Philipp H. Kindt and Samarjit Chakraborty. On optimal neighbor discovery. In *Proceedings of the ACM SIGCOMM Conference*, 2019.

[68] Toshihiko Komine and Masao Nakagawa. Fundamental analysis for visible-light communication system using LED lights. *IEEE Transactions on Consumer Electronics*, 50(1), 2004.

[69] Christopher Kraemer, Amy Guo, Saad Ahmed, and Josiah Hester. Battery-free MakeCode: Accessible Programming for Intermittent Computing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(1), 2022.

[70] Meng-Lin Ku, Wei Li, Yan Chen, and K. J. Ray Liu. Advances in energy harvesting communications: Past, present, and future challenges. *IEEE Communications Surveys & Tutorials*, 18(2), 2016.

[71] Piyush Kuchhal and Umesh Chandra Sharma. Battery waste management. *Environmental science and engineering*, 5, 2019.

[72] Ye-Sheng Kuo, Pat Pannuto, Ko-Jen Hsiao, and Prabal Dutta. Luxapose: Indoor positioning with mobile phones and visible light. In *Proceedings of the 20th ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2014.

[73] Branislav Kusy, Christian Richter, Siddartha Bhandari, Raja Jurdak, Victor J. Neldner, and Michael R. Ngugi. Evidence-based landscape rehabilitation through microclimate sensing. In *2015 12th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, 2015.

[74] Shouwen Lai, Binoy Ravindran, and Hyeonjoong Cho. Heterogenous Quorum-Based Wake-Up Scheduling in Wireless Sensor Networks. *IEEE Transactions on Computers*, 59(11), 2010.

[75] Shouwen Lai, Bo Zhang, Binoy Ravindran, and Hyeonjoong Cho. CQS-Pair: Cyclic quorum system pair for wakeup scheduling in wireless sensor networks. In *Proceedings of the International Conference on Principles of Distributed Systems (OPODIS)*. Springer, 2008.

[76] J.N. Laneman, D.N.C. Tse, and G.W. Wornell. Cooperative diversity in wireless networks: Efficient protocols and outage behavior. *IEEE Transactions on Information Theory*, 50(12), 2004.

[77] Christoph Lenzen, Philipp Sommer, and Roger Wattenhofer. Optimal clock synchronization in networks. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2009.

[78] Jiangtao Li, Angli Liu, Guobin Shen, Liqun Li, Chao Sun, and Feng Zhao. Retro-VLC: Enabling battery-free duplex visible light communication for mobile and IoT applications. In *Proceedings of the 16th ACM International Workshop on Mobile Computing Systems and Applications (HotMobile)*, 2015.

[79] Tianxing Li and Xia Zhou. Battery-free eye tracker on glasses. In *Proceedings of the 24th ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2018.

[80] Zhenjiang Li, Wenwei Chen, Cheng Li, Mo Li, Xiang-Yang Li, and Yunhao Liu. FLIGHT: Clock calibration using fluorescent lighting. In *Proceedings of the 18th ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2012.

[81] Roman Lim, Federico Ferrari, Marco Zimmerling, Christoph Walser, Philipp Sommer, and Jan Beutel. FlockLab: A Testbed for Distributed, Synchronized Tracing and Profiling of Wireless

Embedded Systems. In *Proceedings of the 12th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2013.

[82] Gaosheng Liu and Lin Wang. Self-Sustainable Cyber-Physical Systems with Collaborative Intermittent Computing. In *Proceedings of the 12th ACM International Conference on Future Energy Systems*, 2021.

[83] Vincent Liu, Aaron Parks, Vamsi Talla, Shyamnath Gollakota, David Wetherall, and Joshua R. Smith. Ambient backscatter: Wireless communication out of thin air. In *Proceedings of the ACM SIGCOMM Conference*, 2013.

[84] Edward Longman, Oktay Cetinkaya, Mohammed El-Hajjar, and Geoff V. Merrett. Wake-up Radio-enabled Intermittently-powered Devices for Mesh Networking: A Power Analysis. In *Proceedings of the 18th IEEE Annual Consumer Communications Networking Conference (CCNC)*, 2021.

[85] Brandon Lucia, Vignesh Balaj, Alexei Colin, Kiwan Maeng, and Emily Ruppel. Intermittent computing: Challenges and opportunities. In *Proceedings of the 2nd Summit on Advances in Programming Languages (SNAPL)*, 2017.

[86] Brandon Lucia, Brad Denby, Zachary Manchester, Harsh Desai, Emily Ruppel, and Alexei Colin. Computational nanosatellite constellations: Opportunities and challenges. *ACM GetMobile: Mobile Computing and Communications*, 25(1), 2021.

[87] Dong Ma, Guohao Lan, Weitao Xu, Mahbub Hassan, and Wen Hu. SEHS: Simultaneous energy harvesting and sensing using piezoelectric energy harvester. In *Proceedings of the 3rd IEEE/ACM International Conference on Internet-of-Things Design and Implementation (IoTDI)*, 2018.

[88] Kiwan Maeng and Brandon Lucia. Adaptive dynamic checkpointing for safe efficient intermittent computing. In *Proceedings*

of the 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI), 2018.

[89] Kiwan Maeng and Brandon Lucia. Supporting peripherals in intermittent systems with just-in-time checkpoints. In *Proceedings of the 40th ACM Conference on Programming Language Design and Implementation (PLDI)*, 2019.

[90] Fabian Mager, Dominik Baumann, Romain Jacob, Lothar Thiele, Sebastian Trimpe, and Marco Zimmerling. Feedback control goes wireless: guaranteed stability over low-power multi-hop networks. In *Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*, New York, NY, USA, 2019.

[91] Amjad Yousef Majid, Carlo Delle Donne, Kiwan Maeng, Alexei Colin, Kasim Sinan Yildirim, Brandon Lucia, and Przemysław Pawełczak. Dynamic Task-based Intermittent Execution for Energy-harvesting Devices. *ACM Transactions on Sensor Networks*, 16(1), 2020.

[92] Amjad Yousef Majid, Michel Jansen, Guillermo Ortas Delgado, Kasim Sinan Yildirim, and Przemysław Pawełczak. Multi-hop backscatter tag-to-tag networks. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2019.

[93] Amjad Yousef Majid, Patrick Schilder, and Koen Langendoen. Continuous sensing on intermittent power. In *Proceedings of the 19th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2020.

[94] Robert Margolies, Maria Gorlatova, John Sarik, Gerald Stanje, Jianxun Zhu, Paul Miller, Marcin Szczodrak, Baradwaj Vigraham, Luca Carloni, Peter Kinget, Ioannis Kymissis, and Gil Zussman. Energy-Harvesting Active Networked Tags (EnHANTs): Prototyping and Experimentation. *ACM Transactions on Sensor Networks*, 11(4), 2015.

[95] Michael J. McGlynn and Steven A. Borbash. Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks. In *Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2001.

[96] Geoff V. Merrett and Bashir M. Al-Hashimi. Energy-driven computing: Rethinking the design of energy harvesting systems. In *Proceedings of the EDAA Conference on Design, Automation & Test in Europe (DATE)*, 2017.

[97] mOOvement. moovement cattle tracking and management. Online at: https://www.moovement.com.au/, 2021.

[98] Sujay Narayana, R. Muralishankar, R. Venkatesha Prasad, and Vijay S. Rao. Recovering bits from thin air: Demodulation of bandpass sampled noisy signals for space iot. In *Proceedings of the 18th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2019.

[99] Gang Ning and Branko N. Popov. Cycle Life Modeling of Lithium-Ion Batteries. *Journal of The Electrochemical Society*, 151(10), 2004.

[100] Sung Y. Park and Anil K. Bera. Maximum entropy autoregressive conditional heteroskedasticity model. *Journal of Econometrics*, 150(2), 2009.

[101] Giulia Pazzaglia, Marco Mameli, Luca Rossi, Marina Paolanti, Adriano Mancini, Primo Zingaretti, and Emanuele Frontoni. People Counting on Low Cost Embedded Hardware During the SARS-CoV-2 Pandemic. In *Proceedings of Pattern Recognition. ICPR International Workshops and Challenges*, 2021.

[102] Rajeev Piyare, Amy L. Murphy, Csaba Kiraly, Pietro Tosato, and Davide Brunelli. Ultra Low Power Wake-Up Radios: A Hardware and Networking Survey. *IEEE Communications Surveys & Tutorials*, 19(4), 2017.

[103] Andrea Polenta, Pietro Rignanese, Paolo Sernani, Nicola Falcionelli, Dagmawi Neway Mekuria, Selene Tomassini, and Aldo Franco Dragoni. An Internet of Things Approach to Contact Tracing - The BubbleBox System. *MDPI Information*, 11(7), 2020.

[104] Carlos Pérez-Penichet, Claro Noda, Ambuj Varshney, and Thiemo Voigt. Battery-free 802.15.4 receiver. In *Proceedings of the 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2018.

[105] Vijay Raghunathan, A. Kansal, J. Hsu, J. Friedman, and Mani Srivastava. Design considerations for solar energy harvesting wireless embedded systems. In *Proceedings of the 4th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2005.

[106] Niranjini Rajagopal, Patrick Lazik, and Anthony Rowe. Visual light landmarks for mobile devices. In *Proceedings of the 13th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2014.

[107] Julian Randall, Oliver Amft, Jürgen Bohn, and Martin Burri. LuxTrace: Indoor positioning using building illumination. *Personal and Ubiquitous Computing*, 11(6), 2007.

[108] Rui Rocha, Jorge Dias, and Adriano Carvalho. Cooperative multi-robot systems: A study of vision-based 3-d mapping using information theory. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA)*, 2005.

[109] Guoxin Rong, Simon R. Corrie, and Heather A. Clark. In Vivo Biosensing: Progress and Perspectives. *ACS Sensors*, 2(3), 2017.

[110] Anthony Rowe, Vikram Gupta, and Ragunathan (Raj) Rajkumar. Low-power clock synchronization using electromagnetic energy radiating from AC power lines. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2009.

[111] Jihoon Ryoo, Yasha Karimi, Akshay Athalye, Milutin Stanaćević, Samir R. Das, and Petar Djurić. Barnet: Towards activity recognition using passive backscattering tag-to-tag network. In *Proceedings of the 16th ACM Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2018.

[112] Alanson P. Sample, Daniel J. Yeager, Pauline S. Powledge, Alexander V. Mamishev, and Joshua R. Smith. Design of an RFID-Based Battery-Free Programmable Sensing Platform. *IEEE Transactions on Instrumentation and Measurement*, 57(11), 2008.

[113] Muhammad Moid Sandhu, Kai Geissdoerfer, Sara Khalifa, Raja Jurdak, Marius Portmann, and Brano Kusy. Towards energy positive sensing using kinetic energy harvesters. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2020.

[114] Muhammad Moid Sandhu, Sara Khalifa, Kai Geissdoerfer, Raja Jurdak, and Marius Portmann. SolAR: Energy positive human activity recognition using solar cells. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2021.

[115] Mahadev Satyanarayanan, Wei Gao, and Brandon Lucia. The computing landscape of the 21st century. In *Proceedings of the 20th ACM International Workshop on Mobile Computing Systems and Applications (HotMobile)*, 2019.

[116] Olga Saukh, David Hasenfratz, and Lothar Thiele. Reducing multi-hop calibration errors in large-scale mobile sensor networks. In *Proceedings of the 14th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2015.

[117] Fred B. Schneider. Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Computing Surveys*, 22(4), 1990.

[118] Markus Schuß, Carlo Alberto Boano, Manuel Weber, Matthias

Schulz, Matthias Hollick, and Kay Römer. JamLab-NG: Benchmarking low-power wireless protocols under controlable and repeatable Wi-Fi interference. In *Proceedings of the 16th International Conference on Embedded Wireless Systems and Networks (EWSN)*, 2019.

[119] Yiran Shen, Reza Arablouei, Frank de Hoog, Jaques Malan, James Sharp, Sara Shouri, Timothy D. Clark, Carine Lefevre, Frederieke Kroon, Andrea Severati, and Brano Kusy. Estimating Heart Rate and Detecting Feeding Events of Fish Using an Implantable Biologger. In *Proceedings of the 19th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2020.

[120] Lukas Sigrist, Rehan Ahmed, Andres Gomez, and Lothar Thiele. Harvesting-Aware Optimal Communication Scheme for Infrastructure-Less Sensing. *ACM Transactions on Internet of Things*, 1(4), 2020.

[121] Lukas Sigrist, Andres Gomez, Roman Lim, Stefan Lippuner, Matthias Leubin, and Lothar Thiele. Measurement and Validation of Energy Harvesting IoT Devices. In *Proceedings of the EDAA Conference on Design, Automation & Test in Europe (DATE)*, 2017.

[122] Philipp Sommer, Kai Geissdoerfer, Raja Jurdak, Branislav Kusy, Jiajun Liu, Kun Zhao, Adam McKeown, and David Westcott. Energy- and Mobility-Aware Scheduling for Perpetual Trajectory Tracking. *IEEE Transactions on Mobile Computing*, 19(3), 2020.

[123] Philipp Sommer and Branislav Kusy. Minerva: Distributed Tracing and Debugging in Wireless Sensor Networks. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2013.

[124] Philip Sparks. The route to a trillion devices: The outlook for iot investment to 2035. Technical report, 2017.

[125] Orr Spiegel, Wayne M. Getz, and Ran Nathan. Factors influencing foraging search efficiency: why do scarce lappet-faced vultures outperform ubiquitous white-backed vultures? *The American Naturalist*, 181(5), 2013.

[126] Vamsi Talla, Joshua Smith, and Shyamnath Gollakota. Advances and Open Problems in Backscatter Networking. *ACM GetMobile: Mobile Computing and Communications*, 24(4), 2021.

[127] Wilson M. Tan, Paul Sullivan, Hamish Watson, Joanna Slota-Newson, and Stephen A. Jarvis. An indoor test methodology for solar-powered wireless sensor networks. *ACM Transactions on Embedded Computing Systems*, 16(3), 2017.

[128] Ashok Samraj Thangarajan, Fan Yang, Wouter Joosen, and Danny Hughes. Real-time Distributed In-Situ Benchmarking of Energy Harvesting IoT Devices. In *Proceedings of the 5th ACM Workshop on Middleware and Applications for the Internet of Things (M4IoT)*, 2018.

[129] D. M. Titterington. Recursive Parameter Estimation Using Incomplete Data. *Journal of the Royal Statistical Society: Series B (Methodological)*, 46(2), 1984.

[130] Alessandro Torrisi, Kasim Sinan Yildirim, and Davide Brunelli. Enabling Transiently-Powered Communication via Backscattering Energy State Information. In *Applications in Electronics Pervading Industry, Environment and Society*. Springer International Publishing, 2021.

[131] Nicolas Tsurukawa, Siddharth Prakash, and Andreas Manhart. Social impacts of artisanal cobalt mining in katanga, democratic republic of congo, 2011.

[132] Ambuj Varshney, Andreas Soleiman, Luca Mottola, and Thiemo Voigt. Battery-free visible light sensing. In *Proceedings of the 4th ACM Workshop on Visible Light Communication Systems (VLCS)*, 2017.

[133] Sudarshan Vasudevan, Micah Adler, Dennis Goeckel, and Don Towsley. Efficient algorithms for neighbor discovery in wireless networks. *IEEE/ACM Transactions on Networking*, 21(1), 2013.

[134] Sudarshan Vasudevan, Donald Towsley, Dennis Goeckel, and Ramin Khalili. Neighbor discovery in wireless networks and the coupon collector's problem. In *Proceedings of the 15th ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2009.

[135] Keyu Wang, Xufei Mao, and Yunhao Liu. BlindDate: A neighbor discovery protocol. In *Proceedings of the 42nd International Conference on Parallel Processing (ICPP)*, 2013.

[136] Thomas Cherico Wanger. The Lithium future - resources, recycling, and the environment. *Conservation Letters*, 4(3), 2011.

[137] Mathew L. Wymore, Vishal Deep, Vishak Narayanan, Henry Duwe, and Daji Qiao. Lifecycle Management Protocols for Batteryless, Intermittent Sensor Nodes. In *Proceedings of the 39th IEEE International Performance Computing and Communications Conference (IPCCC)*, 2020.

[138] Xun Xian, Xinran Wang, Jie Ding, and Reza Ghanadan. Assisted learning: A framework for multi-organization learning. In *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[139] Liguang Xie, Yi Shi, Y. Thomas Hou, and Andwenjing Lou. Wireless power transfer and applications to sensor networks. *IEEE Wireless Communications*, 20(4), 2013.

[140] Wei Ye, J. Heidemann, and D. Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, 2002.

[141] Kasim Sinan Yildirim, Amjad Yousef Majid, Dimitris Patoukas, Koen Schaper, Przemyslaw Pawelczak, and Josiah Hester. InK: Reactive Kernel for Tiny Batteryless Sensors. In *Proceedings of*

the 16th ACM Conference on Embedded Networked Sensor Systems (SenSys), 2018.

[142] Kasim Sinan Yildirim and Przemyslaw Pawelczak. On Distributed Sensor Fusion in Batteryless Intermittent Networks. In *Proceedings of the 15th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2019.

[143] Jie Zhan, Geoff V. Merrett, and Alex S. Weddell. Exploring the Effect of Energy Storage Sizing on Intermittent Computing System Performance. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(3), 2022.

[144] Jia Zhao, Wei Gong, and Jiangchuan Liu. X-tandem: Towards multi-hop backscatter communication with commodity wifi. In *Proceedings of the 24th ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2018.

[145] Tongxin Zhu, Jianzhong Li, Hong Gao, and Yingshu Li. Broadcast scheduling in battery-free wireless sensor networks. *ACM Transactions on Sensor Networks*, 15(4), 2019.