Greentooth: Robust and Energy Efficient Wireless Networking for Batteryless Devices

SIMEON BABATUNDE, Clemson University, Clemson, USA
ARWA ALSUBHI, Clemson University, Clemson, USA
JOSIAH HESTER, Georgia Institute of Technology, Atlanta, USA
JACOB SORBER, Clemson University, Clemson, USA

Communication presents a critical challenge for emerging intermittently powered batteryless sensors. Batteryless devices that operate entirely on harvested energy often experience frequent, unpredictable power outages and have trouble keeping time accurately. Consequently, effective communication using today's low-power wireless network standards and protocols becomes difficult, particularly because existing standards are usually designed to support reliably powered devices with predictable node availability and accurate timekeeping capabilities for connection and congestion management.

In this article, we present Greentooth, a robust and energy-efficient wireless communication protocol for intermittently powered sensor networks. It enables reliable communication between a receiver and multiple batteryless sensors using Time Division Multiple Access–style scheduling and low-power wake-up radios for synchronization. Greentooth employs lightweight and energy-efficient connections that are resilient to transient power outages, while significantly improving network reliability, throughput, and energy efficiency of both the battery-free sensor nodes and the receiver—which could be untethered and energy constrained. We evaluate Greentooth using a custom-built batteryless sensor prototype on synthetic and real-world energy traces recorded from different locations in a garden across different times of the day. Results show that Greentooth achieves 73% and 283% more throughput compared to Asynchronous Wake-up on Demand MAC and Receiver-Initiated Consecutive Packet Transmission Wake-up Radios, respectively, under intermittent ambient solar energy and over 2× longer receiver lifetime.

CCS Concepts: • **Computer systems organization** → **Sensor networks**; **Embedded systems**; • **Hardware** → **Emerging architectures**; • **Networks** → **Mobile networks;**

Additional Key Words and Phrases: Intermittent networks, battery-free sensor networks, batteryless networking, low-power wireless protocols, energy harvesting, wake-up radios

ACM Reference Format:
Simeon Babatunde, Arwa Alsubhi, Josiah Hester, and Jacob Sorber. 2024. Greentooth: Robust and Energy Efficient Wireless Networking for Batteryless Devices. *ACM Trans. Sensor Netw.* 20, 3, Article 66 (April 2024), 31 pages. https://doi.org/10.1145/3649221

This research is supported by NSF award CNS-1453607. Any opinions, findings, conclusions, and recommendations expressed in these materials are those of the authors and do not necessarily reflect the views of the sponsors.

Authors' addresses: S. Babatunde, A. Alsubhi, and J. Sorber, Clemson University, School of Computing 821 McMillan Rd 100 McAdams Hall Clemson, SC, 29634-0001 United States of America; e-mails: sbabatu@clemson.edu, aalsubh@clemson.edu, jsorber@clemson.edu; J. Hester, Georgia Institute of Technology, Ka Moamoa Lab, Klaus Advanced Computing Building, 266 Ferst Drive, Atlanta, GA 30332-0765 United States of America; e-mail: josiah@gatech.edu.

ACM 1550-4859/2024/04-ART66
https://doi.org/10.1145/3649221

1 INTRODUCTION

Intermittently powered and battery-free sensing devices have recently emerged as a promising approach in building a more sustainable Internet of Things and enabling long-term and low-maintenance data gathering in smart farming [22], health and urban monitoring [49, 65], infrastructure sensing [3], industrial control [84], and other applications. These devices harvest energy from their surroundings—solar, thermal, kinetic, or RF—and store that energy in tiny capacitors to collect and process sensor data, then wirelessly transmit the results to other connected systems and services [57]. Harvested energy varies, often unpredictably, and when energy becomes scarce, these devices often operate intermittently with periods of active processing intermixed with frequent power outages of varying length.

Intermittent operation presents new challenges [33], and a range of language and runtime systems, tools, and techniques have been proposed [4, 12, 16, 35, 37, 46, 51, 58, 76] to address many of them; however, effective wireless communication remains a critical gap due to inherent challenges like unpredictable power failures, limited energy storage, and timing inaccuracies [33, 34]. The wireless technologies and protocols used in today's low-power networks have utilized techniques like duty-cycling, **Wake-up Radios (WuR)**, and energy harvesting to extend network lifetimes [73]. Simple duty-cycling (periodically turning off radios when not needed) reduces energy usage but suffers from latency and synchronization issues when nodes are intermittent. WuR-based **Medium Access Control (MAC)** protocols that use both a main radio and ultra-low power WuR, promise to eliminate idle listening, overhearing costs, and duty cycle induced latency [24, 68, 71]. However, these protocols are designed based on problematic assumptions that devices are reliably powered (using batteries or supercapacitors), can accurately keep time, or that a mains-powered base station compensates for the nodes' shortcomings. Other energy harvesting-oriented protocols aim for **energy neutral operation (ENO)** and adapt to avoid intermittence [8, 69, 70], but avoiding power failures is not always an option. Additionally, most WuR-based MAC protocols have been evaluated only in simulation, under simplifying assumptions that often differ significantly from real deployment conditions, specifically when operating entirely on harvested energy [8–10, 49, 54, 71, 72].

With few available options, many intermittent batteryless devices rely on best-effort transmission, allowing one-way data transmission without channel coordination or delivery guarantees, while requiring the sink node to listen constantly (at considerable energy cost) [3]. In many sensing applications—especially those relying on mobile, ad hoc, or solar-powered receivers [78, 80, 88, 92] or those located in remote locations [62, 91]—the energy efficiency of the receiver impacts the overall cost, size, and flexibility of the system and limits how these networks can be deployed. Overall, it is critical to address the following research questions to enable reliable and robust wireless networking for intermittent batteryless systems: (1) What are the novel changes in traditional PHY and MAC layer protocols required to support constrained batteryless networks with intermittent connectivity?, (2) What kinds of platform and radio support are needed to enable intermittent batteryless networks that rely entirely on harvested energy?, and (3) How can we effectively employ existing tools for empirical evaluation of new batteryless network protocols in real-world environments?

In this article, we explore a unique approach, called Greentooth—a receiver-initiated MAC and PHY layer protocol designed for efficient, active, connection-oriented communication among battery-free, intermittently powered sensing devices. Unlike other asymmetric WuR MACs that use expensive ID-based wake-up and control packets, Greentooth adopts a synchronous approach for reliable congestion-free communication using an energy-efficient connection mechanism that tolerates power outages and outage-induced timing inaccuracies. Greentooth utilizes connections for two reasons. First, a single broadcast of a **wake-up packet (WuPkt)** from an energy-constrained sink can synchronize multiple connected nodes simultaneously, eliminating the need to individually poll each node for data using costly addressed WuPkts. This approach minimizes latency and conserves energy at both intermittent nodes and the coordinating sink node. Second, employing **Time Division Multiple Access– (TDMA)** style communication, where dedicated time slots are allocated to each node, enhances reliability and energy efficiency by ensuring collision-free transmissions. In general, Greentooth is well suited for scheduled sensing applications with high reliability long-term and low-maintenance requirements, such as active volcano monitoring [91], smart agriculture [22], battlefield surveillance [90], infrastructure monitoring (protected archaeological site) [3], and animal tracking [19].

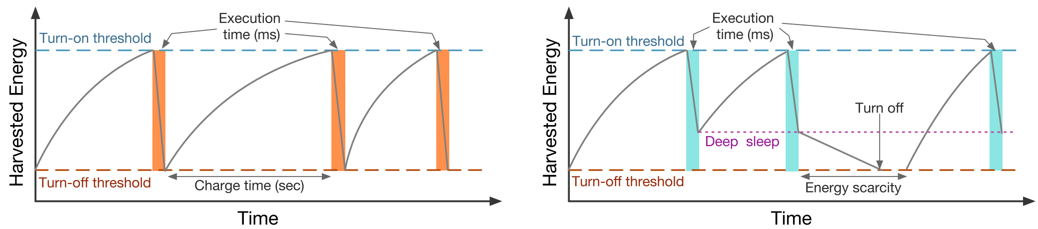
Specifically, Greentooth provides a robust and energy-efficient networking paradigm that supports emerging battery-free intermittent sensing applications requiring ad hoc deployment capabilities with untethered or mobile receivers that are energy constrained. A typical example is a livestock tracking application on a cattle farm where battery-free sensors are mounted on animals for sensing location and body temperature data, which are then collected using a mobile sink or smartphone by a farmer [56]. Greentooth addresses the challenges of batteryless networking by implementing the following intermittent-aware solutions: (i) **Sol1** (enabling lightweight synchronization and connections using a simplified WuPkt broadcast technique), (ii) **Sol2** (minimizing timing issues on batteryless nodes using a timekeeping solution that works across power failures alongside an ACK-based drift correction mechanism), (iii) **Sol3** (minimizing collisions, overhearing, and discovery latency through a TDMA-style MAC protocol with novel adaptive neighbor discovery and dynamic slot recycling capabilities), (iv) **Sol4** (minimizing reconnection overhead by preserving connection state across multiple power failures), and (v) **Sol5** (adopting a configurable WuPkt transmission recurrence to further reduce receiver energy consumption). While there has been notable progress in research related to WuR technology, low-power MAC protocols, and energy harvesting, this work presents the first implementation of an intermittent-resilient networking protocol that uses reliable connections for real intermittent batteryless networks. This is significant as effectively tackling the challenges posed by intermittent connectivity is crucial for realizing the long-term and low-maintenance potential of batteryless systems. Our contributions in this regard include the following:

(1) We present Greentooth, the first energy efficient MAC- and PHY-layer protocol for robust connection-oriented communication on intermittent battery-free sensor networks.
(2) We employ a dual-radio configuration, integrating a low-power WuR with a commodity radio transceiver, to efficiently synchronize batteryless sensors and manage device-to-receiver connections.
(3) We describe reference Greentooth hardware and software implementations, with the commitment to make these assets accessible to the research community upon the publication of our work.
(4) We evaluate and compare the performance of Greentooth against key representatives of **state-of-the-art (SoA)** receiver-initiated WuR MAC protocols, through both synthetic and real world energy traces for indoor temperature sensing and soil moisture monitoring applications.

2 BACKGROUND

2.1 Batteryless Networking Challenges

Batteryless systems capitalize on their external environment for energy harvesting, storing this harvested energy in small capacitors, and then strategically executing tasks once an adequate energy reserve has been accumulated [35]. As depicted in Figure 1(a) and (b), intermittent operation,



(a) Traditional (best-effort) intermittent execution **(b)** Soft intermittent execution used in Greentooth

Fig. 1. Intermittent execution models of battery-free systems operating entirely on harvested ambient energy. (a) The traditional (best-effort) intermittent execution model enables a battery-free node to power on and execute some code snippets as soon as it has harvested enough energy (beyond the turn-on threshold), it then powers off when the energy level depletes below the turn-off threshold. (b) For soft intermittent execution model, a battery-free node becomes active after reaching the turn-on threshold, consumes a fixed amount of energy for code execution, and voluntarily goes to sleep to prevent the storage capacitor from depleting below the turn-off threshold (thus minimizing power failures). However, it still loses power when harvested energy becomes scarce for extended period of time.

stemming from inconsistent energy harvesting and the presence of limited energy storage (orders of magnitude smaller than supercapacitors or batteries [33, 57]), leads to sporadic and inevitable power outages. This situation can give rise to unpredictable control-flow, uncertain timing, as well as the loss of all volatile memory (including stacks, program counters, and registers), which significantly complicates network communication [34, 35, 57] and introduces a variety of challenges discussed below.

Communication is expensive. Transmitting data packets or listening for messages from other sensors are among the most energy-intensive tasks performed by low-power sensors [77]. For applications that require infrequent transmissions, transmission costs can be amortized over time, but listening for messages remains prohibitively expensive. For instance, intermittent battery-free devices [32], using traditional low-power radios like the TI CC1101 [41], need to charge their small capacitors (often tens or hundreds of μF to encourage quick charging) for a few seconds depending on the harvesting situation, to listen for 1–50 ms before depleting their stored energy. Therefore, effective listening among batteryless nodes—a crucial component in relaying messages and acknowledgments—is impractical without knowing in advance when the message will arrive, which is challenging under unpredictable intermittent operation.

Uncertain timekeeping makes communication more expensive. Intermittent power outages can last anywhere from milliseconds to minutes and are often both unavoidable and hard to predict. During a power outage, system clocks stop working, and techniques like TARDIS [75], CusTARD [36], and CHRT [17] can provide rough estimates of outage duration but with errors of up to 10%. With less-accurate timekeeping, batteryless nodes must listen for longer periods to ensure they hear transmissions, while consuming more energy that could have been used for gathering, processing, and transmitting new data, and often causing new power outages, which further increase timing uncertainty. When faced with these timing and energy constraints, most batteryless intermittent networks have shifted the energy burden to the receiver—leaving the receiver on and listening continuously, so that intermittent nodes can transmit whenever they have data to send and the energy to do so.

What about the receiver? Shifting the energy costs to the receiver may not be an option for some applications. In smart buildings, autonomous vehicles, and other applications, receivers may have convenient access to wired power and can afford to listen continuously, as long as the owner does not mind running extra wires and paying the additional electricity costs. In other scenarios, like the cattle farming application [56] mentioned earlier, agricultural sensing [67, 87] and ecological monitoring [80], receivers are often battery-powered, untethered, mobile, or deployed in remote locations and extreme environments [62, 91]. Wired power may be expensive, inconvenient, or even impractical, and whether these receivers run on harvested energy or need periodic battery replacements, the cost of constant listening is significant. An energy-efficient receiver can be smaller, cost less, collect and relay more data and operate longer without human intervention.

2.2 Dual-radio Architecture

A WuR is an extremely low-power radio receiver (orders of magnitude lower consumption than existing main radio transceivers) that continuously monitors the wireless medium and generates an interrupt signal whenever it detects the presence of a WuPkt [23, 71]. A typical WuR supports two modes of operation: broadcast mode, in which a single WuPkt activates multiple WuRs, and addressable mode, in which each WuR is activated individually by a WuPkt containing its address pattern or ID [73]. Often, the WuR is used with a main radio (dual-radio architecture), because main radios are capable of generating WuPkts in addition to actual data transmission, and this obviates the need for a separate WuPkt transmitter. As shown in Figure 2, a WuR is configured to monitor network channel continuously, while the main radio is off or in a sleep mode. The main radio is then activated by an interrupt (for data communication) whenever the WuR receives an addressed (ID-based) or broadcast WuPkt, thereby eliminating idle listening overheads on the main radio.

A new class of asynchronous protocols, called WuR MACs have used the dual-radio architecture for improved performance. Like traditional MAC protocols, WuR MACs can be classified into transmitter initiated, receiver initiated, and bi-directional, based on which end initiates the communication. Transmitter-initiated WuR MACs like WuR-TICER [47], PTW [1], WuR-MAC [61], and DCW-MAC [63], though implemented and evaluated in simulation tools, often operate in an on-demand fashion, where a node initiates communication by sending a WuPkt to activate the sink, followed by an actual data transfer. Receiver-initiated protocols [19, 29, 49, 50, 79] however, place the task of initiating communication on the receiver or sink node. Because of the sink's capability to manage collisions, they are mostly suited for applications with high reliability requirements; coupled with scheduled and infrequent transmissions. Finally, in Bi-directional WuR MACs [5, 15, 44, 45, 64, 72], both transmitter and receiver can initiate communication as both have WuR capabilities, making them suitable for multihop communication.

Despite the low-power listening and low latency features of the dual-radio architecture, wake-up packet transmission can be energy consuming, particularly the ID-based WuPkts used for selective node awakening. Moreover, the bulk of existing receiver-initiated WuR MACs [15, 19] have only been tested in simulation, without any practical results based on real hardware or testbeds [73]. Evaluating the performance of these protocols using simulation-based testing is reasonable, but further evaluation on a real testbed (or in the wild) will provide more credibility to the performance claims. Furthermore, these protocols are designed for ENO—those that adapt to avoid power failures [8, 69, 70]. Unfortunately, when harvested energy becomes insufficient compared to the nodes' energy demands during their active periods, power failures become an inevitable challenge for numerous devices. When conventional WuR MAC protocols and methods are deployed in intermittent batteryless networks experiencing frequent and unpredictable power disruptions along with timing issues, it adversely affects the networks' performance, reliability, and overall energy efficiency.

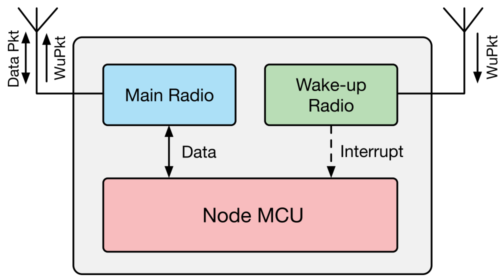


Fig. 2. Overview of a dual-radio architecture, consisting of an ultra low-power wake-up receiver used jointly with a main radio transceiver.

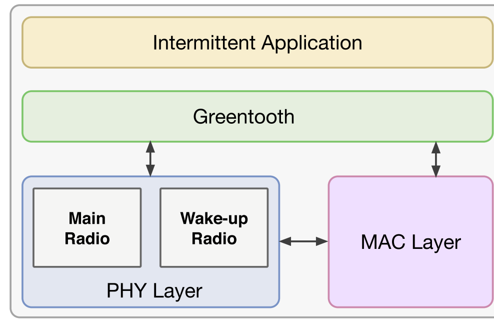


Fig. 3. Communication stack showing how Greentooth protocol interfaces with other layers of the stack on a battery-free node running intermittent sensing application.

3 GREENTOOTH

Greentooth is motivated by the need for a new networking paradigm capable of supporting long-lasting, battery-free intermittent sensing applications requiring untethered or mobile receivers (sinks) that are energy constrained. It enables reliable connection-oriented communication for intermittent energy-harvesting devices through a combination of MAC and PHY layer features that ensure scalable, robust, and energy-efficient batteryless networking. Greentooth uses lightweight connections initiated by a single WuPkt broadcast for synchronous, single-hop TDMA communication between intermittent sensor nodes and coordinating receiver, enabling the nodes to deliver reliable data to the sink without the concerns of packet collisions and poor quality of service. Using a single WuPkt broadcast to synchronize all nodes in the network eliminates the energy overhead and latency caused by continuous transmissions of expensive ID-based WuPkt for each individual node as in some prior protocols, such as **Asynchronous Wake-up on Demand MAC (AWD-MAC)** [49] and DoRa [50]. We design Greentooth as a cross-layer protocol over the medium access and physical layers of the communication stack presented in Figure 3, which provides a unified interface for the application layer at the top of the stack.

The communication cycle is central to how Greentooth manages synchronous connection-oriented communication between an energy-constrained receiver and many intermittent sensor nodes. In a similar way to BLE's connection interval [11], the cycle defines the protocol's communication pattern, which happens in three distinct phases: *Wake_and_Sync*, *Discovery*, and *Transmission*. These phases are repeated continuously throughout Greentooth's operation and are defined by a series of user-configurable parameters listed in Table 1. As shown in Figure 4, the communication cycle, T_c, is the sum of the duration for the three phases, T_w, T_d, and T_t. The value of T_c impacts network throughput, power consumption, and the total number of sensor nodes that the network can accommodate. For instance, a high T_c value will increase the number of nodes the network can support but will also decrease the throughput of each node.

3.1 Wake_and_Sync

The communication cycle begins with a WuPkt broadcast from the receiver that wakes and synchronizes all batteryless nodes in the network. The WuPkt—transmitted by the receiver's main radio—serves both as an advertisement packet for new nodes looking to pair and a synchronization packet for already paired nodes. The WuPkt transmission period, T_w, can also vary depending on whether pattern correlation is enabled (requires a longer wake-up pattern) or additional data

Table 1. Greentooth protocol model notation

Protocol parameters	
N_i	batteryless sensor node identifier
T_c	communication cycle period
T_d	discovery period
T_t	data transmission period
T_w	wake-up packet transmission time
t_s	time slot duration
t_0	start of the transmission phase
t_i	time slot for batteryless node i
d_{ss}	discovery step size
dd_t	random discovery delay threshold
n_{max}	max number of nodes that can be accommodated

The protocol parameters are configurable by a system designer or developer in the application code for both the receiver and sensor nodes. The deployment requirements and application needs will inform the choice of the right parameter values.

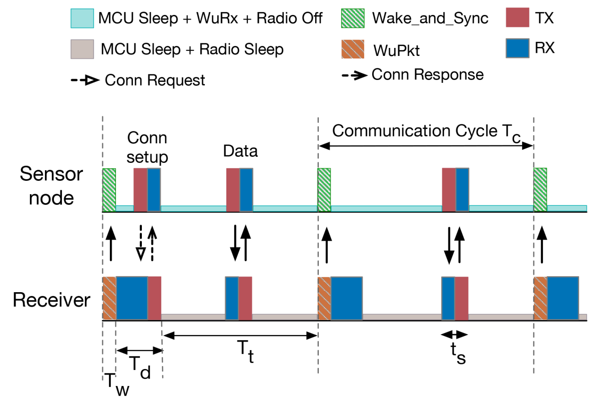


Fig. 4. Greentooth's communication cycle T_c consists of WuPkt period T_w, Discovery period T_d, and Transmission period T_t. Transmission period is divided into time slots with configurable duration (t_s). New nodes joining the network first establish a connection with the receiver by sending a *connection request* packet and then receive a *connection response* packet that contains time slot information.

is included in the WuPkt. Enabling pattern correlation is beneficial for asynchronous WuR MAC protocols that prioritize selective ID-based wake ups, as it reduces false wake-up events. After broadcasting the WuPkt, the receiver proceeds to the *Discovery* phase.

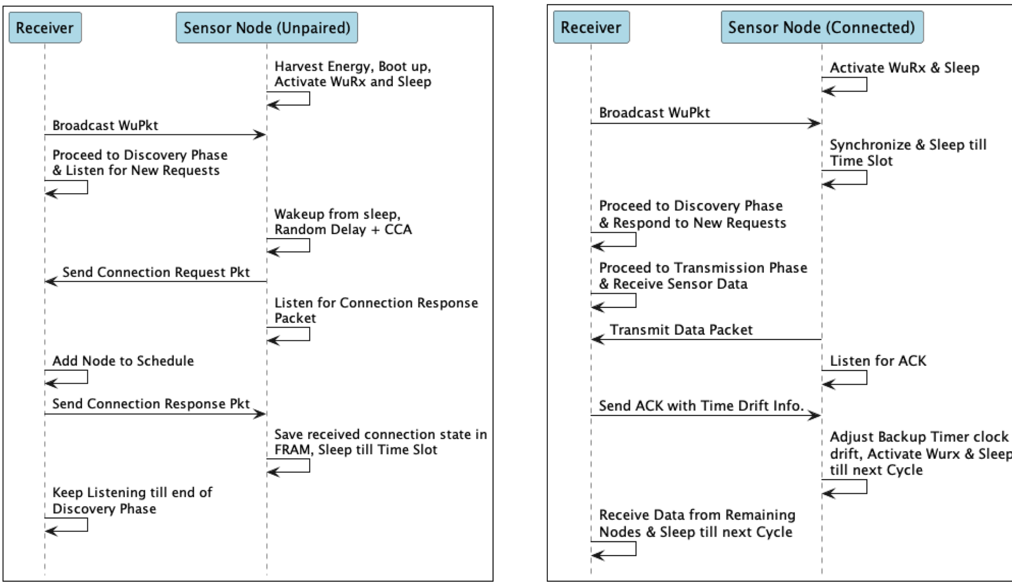
Prior to receiving a WuPkt, all nodes in the network are expected to either be in a sleep state or harvesting energy to startup (for nodes that just joined the network or have depleted their energy). After waking up, connected nodes synchronize their times with the receiver and return to sleep until the start of their time slots, while unconnected nodes proceed to the *Discovery* phase.

3.2 Discovery

The *Discovery* phase provides an opportunity, T_d, for unconnected nodes to connect. During this phase, the receiver actively listens (Rx) for new nodes. As described in Figure 5(a), a node that wants to connect and has enough energy will send a *Connection Request* packet to the receiver after a small random delay (configurable via dd_t). The random delay along with clear channel assessment helps avoid collisions when two or more nodes try to pair with the receiver at the same time.

After receiving a *Connection Request* packet, the receiver adds the new node to the schedule by assigning it a transmission slot t_i and then sends back a *Connection Response* packet with time slot if the pairing process was successful. The node saves the received connection information, in non-volatile memory (FRAM) for use across power failures, and returns to sleep until the start of its time slot. If pairing fails either due to a collision or a lost packet, then the node will try to rediscover in the next communication cycle—provided it has harvested enough energy.

The discovery period (T_d) on the receiver presents a tradeoff between energy efficiency and responsiveness. Longer discovery periods allow more nodes to joint the network more quickly. Shorter discovery periods improve the energy efficiency of the receiver. To help developers tune their systems to specific application conditions and needs, we support two discovery modes: *fixed* and *adaptive* discovery.



(a) Discovery flow for new/disconnected nodes. **(b)** Data transmission flow for connected nodes.

Fig. 5. Sequence diagrams showing key interaction flows between an energy constrained receiver and a batteryless sensor node during different phases of Greentooth protocol. Particularly, (a) when a newly deployed node or previously disconnected node wants to join the network, and (b) how connected nodes deliver their data to the receiver.

Fixed discovery mode, shown in Figure 6, uses a constant T_d value (which could be *High* or *Low*) every communication cycle throughout the entire deployment lifetime of the network. A *Low* T_d value shortens the active discovery listening time, thereby conserving energy on the receiver. But this allows only a small number of new nodes to connect each cycle, and may extend the time it takes to completely discover all participating nodes (discovery duration). A *Low* discovery duration is best when sensed data changes slowly, new nodes join the network less frequently, and the energy efficiency of the sink node is critical (i.e., a smart city air-quality sensing deployment with a mobile sink node or cellphone as the receiver). A *High* T_d value allows more nodes to connect more quickly due to the extended discovery listening time, but this leads to an increase in the receiver's power draw and reduces its battery life. This discovery mode is more suited for deployments where rapidly changing data must be collected as soon as new nodes join the network. Overall, the right settings for T_d will depend on deployment requirements and application needs, which may change over time and may be difficult to predict before the network is deployed.

Adaptive discovery mode automatically sets (T_d) to match current network conditions at runtime. In this mode, Greentooth dynamically scales T_d up or down depending on whether there are new nodes waiting to join the network or not respectively. This is aimed at increasing the likelihood of a node getting discovered in the first few discovery cycles. For instance, the value of T_d increases incrementally (in step size defined by d_{ss}) up to the max discovery period as long as new discoveries were made or when collisions happen during discovery. Conversely, T_d decreases gradually to the min discovery period (taken as d_{ss}) as soon as the nodes are completely discovered (no new discoveries). Increasing the T_d value helps in achieving a more responsive network through faster discovery, while decreasing it helps optimize for overall power savings. So adaptive discovery benefits from the strengths of both the *High* and *Low* discovery modes without significant

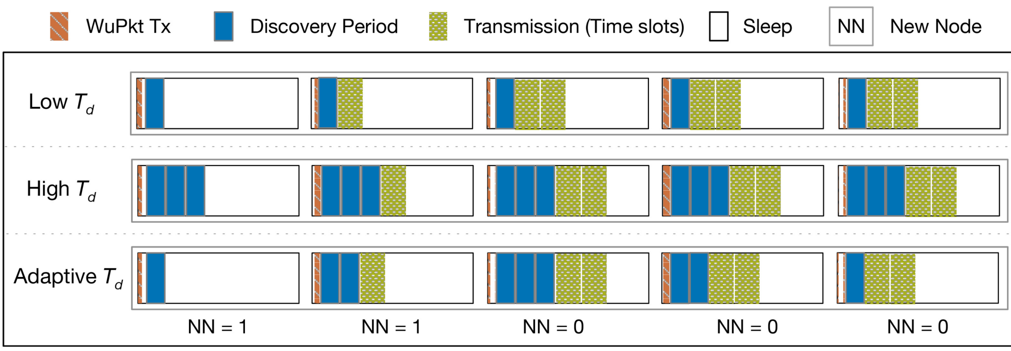


Fig. 6. Greentooth enables different discovery modes to ensure the best tradeoff between energy efficiency and responsiveness requirements of a network. *Low* discovery period conserves energy at the receiver at the expense of a longer discovery duration, while a *High* discovery period achieves a faster discovery at a higher receiver consumption. *Adaptive* discovery combines the benefits of both *Low* and *High* by dynamically adapting T_d at runtime based on the availability of unpaired nodes (NN).

effort on the part of the developer. To ensure that batteryless nodes adapts their random delay threshold dd_t in line with receiver's discovery period, the nodes first set their dd_t to d_{ss} and then increase it incrementally (by d_{ss}) whenever a node experiences packet collision while trying to pair with the receiver.

3.3 Transmission

During the *Transmission* phase, connected nodes have the opportunity to exchange messages with the receiver. This phase consists of n_{max} transmission slots, of time t_s, where $n_{max} = T_t/t_s$. Each connected node is assigned to a single transmission slot. The slot duration t_s and other parameters in Table 1 are customizable based on payload size and application needs.

As shown in Figure 5(b), if a connected node, assigned slot i, has data to send and enough energy to send it, then that node wakes up from sleep after receiving the broadcasted WuPkt, activates its main radio, and transmits its data at time, $t_i = t_0 + (i-1)t_s$, where t_0 is the start of the *transmission* period and t_i is the time delay before the start of each node's time slot. After receiving a data packet, the receiver responds with an ACK that contains the receiver's estimate of the node's current time drift (how far beyond t_i the packet was received) to allow the node to adjust its timing.

After communicating with all connected nodes, the receiver goes to sleep to conserve energy until the next communication cycle. Each sensor node also returns to sleep at the end of its time slot, turns off its main radio, and waits for the next communication cycle while listening for a WuPkt. If a WuPkt fails to arrive, then a backup timer wakes up the node at its assigned time slot and the communication cycle continues. If either the sensor node or receiver fails to communicate for a user-configurable amount of time (*CONNECTION_TIMEOUT*), then the connection is considered lost. Nodes can also specifically end a connection by sending a connection termination message during their assigned time slots.

While the scalability of Greentooth depends on the maximum communication cycle period achievable and the duration of each allocated time slot, Greentooth is capable of handling up to 10 times more concurrent connections than existing protocols.

3.4 Key Features of Greentooth

To enable robust, reliable, and energy efficient connection-oriented networking for intermittent batteryless devices, Greentooth implements the following set of features along with the *adaptive* *discovery mode* mechanism presented in Section 3.2. These key features are tailored distinctively to address the fundamental challenges of batteryless networking described in Section 2.1.

Lightweight synchronization and connections - Sol1: While existing synchronous protocols incur considerable energy overhead and complexity for neighbor discovery and synchronization [21, 25, 48], Greentooth uses a simple broadcast WuPkt (about 32 times less payload compared to ID-based WuPkts) in conjunction with the dual-radio architecture to achieve efficient synchronization and significant power savings on battery-free sensor nodes. The nodes in the network usually remain in sleep mode (like LPM3 on an MSP430 MCU [85]), with their main radios powered off, while using their WuR to listen for a WuPkt from the coordinating receiver—WuR listening costs 3–5 orders of magnitude less power than the main radio. When a WuPkt is received, each node wakes from sleep and aligns itself with the receiver's global time. This efficient synchronization mechanism helps Greentooth maintain inexpensive but reliable connections between the sink and nodes that are plagued with unpredictable failures and timing inaccuracies. The resulting connection-oriented communication method helps mitigate congestion and overhearing problems and is significantly more energy efficient and achieves better throughput compared to asynchronous WuR MAC protocols that poll individual nodes using ID-based WuPkt.

Backup timing and drift resolution - Sol2: The wake-up radio may occasionally miss WuPkts, perhaps due to physical barriers, RF-interference, or node mobility. In the absence of a WuPkt, a connected Greentooth node uses a local timer to continue using its time slot during subsequent *communication cycles* as long as it can. This local timer is configured to wake the node in a recurring fashion (every T_c seconds) as soon as the node receives a *Connection Response* packet at the end of the initial pairing process. So, WuPkts are always helpful, but only necessary during initial pairing or when a node completely drifts out of sync. The local timer on the batteryless nodes is prone to clock drift that can disrupt subsequent slot alignment with the receiver. Consequently, the receiver accounts for node clock drift by monitoring node timing errors at every connection event and provides corrective feedback containing the node ID and drift (*packet arrival time − slot start time*) in its ACKs. This allows sensor nodes to automatically compensate for time drift even when WuPkts are missed or transmitted less frequently. Power outages also introduce timing errors, as batteryless nodes lose track of global time right after reboot. Therefore, Greentooth nodes use a timekeeper to estimate outage duration, which is then used to stay on schedule if WuPkts are missed after reboots. The nodes can keep time with any remanence-based timekeeping method, but we use CuSTARD [36] in our implementation because of its simplicity and low-power benefits. In general, remanence-based timekeeping techniques, like CusTARD, tend to be less precise compared to their counterparts that require active power.

Adaptive neighbor discovery and dynamic slot recycling - Sol3: In addition to the detailed description of the novel adaptive neighbor discovery mechanism provided in Section 3.2, Greentooth's TDMA-style protocol also allows time slots to be managed dynamically, allowing sensor nodes to be added and removed from the network without disrupting existing connections or setup. For example, a node will be kicked off the network (disconnected) if it has been inactive for a period longer than *CONNECTION_TIMEOUT*. This helps maintain efficient use of allocated time slots and energy conservation at the receiver, as the receiver goes to sleep during unused time slots, including those belonging to recently evicted nodes. These unused slots are assigned to newly connected nodes, or reconnected ones that were previously disconnected.

Preserving connection state across outages - Sol4: Frequent and erratic power failures can significantly impact the operation and quality-of-service of intermittent networks. Greentooth addresses this fundamental issue by preserving connection state information across power failures.

Each node saves its connection state in persistent memory after successfully pairing with the receiver during the *discovery* phase. This checkpointing technique is lightweight as writing and reading operations in the MSP430 MCUs (with built-in persistent memory) [86] are very energy efficient. By preserving connection state in non-volatile memory (FRAM), a depleted node can resume communication with the receiver after each reboot without spending additional energy to establish a new connection.

Variable WuPkt transmission - Sol5: Sending a WuPkt every single communication cycle is often an overkill as the batteryless nodes can stay synchronized using their backup timers. Thus, Greentooth further reduces the energy consumption on both the batteryless nodes and the receiver by supporting two different WuPkt transmission modes: *fixed* mode and *dynamic* mode.

In fixed mode, *WKUP_TX_REPS* (a user defined variable) defines how often a WuPkt is broadcasted by the receiver. When *WKUP_TX_REPS* is set to 1, a WuPkt is transmitted at the start of every *communication cycle*, when set to 2, it is transmitted every other *communication cycle*, and so on. This flexibility allows programmers to define WuPkt transmission repetition that suits their application requirements. For example, a higher *WKUP_TX_REPS* value leads to exceptional power savings but may limit how quickly new nodes are discovered and added to the network. However, a small *WKUP_TX_REPS* value is crucial for poor harvesting conditions, where nodes are kicked off the network due to inactivity (exceeds *CONNECTION_TIMEOUT* period) or in conditions where nodes enter and leave the network frequently.

In dynamic mode, rather than defining the value of *WKUP_TX_REPS* statically, the receiver dynamically adjusts the value at runtime based on *node_coverage*—which tracks the percentage of connected nodes that actively interact with the receiver. This is critical as network conditions are subject to change over time because of the arbitrary nature of harvested energy. So, adapting *WKUP_TX_REPS* at runtime in response to changes in network environment provides the benefits of faster discovery and power savings on the receiver. We compute *node_coverage* every *communication cycle* using

$$node\_coverage = \frac{connected\_active\_nodes * 100}{connected\_nodes}.$$

The *connected_active_nodes* is computed by recording the number of connected Greentooth nodes that delivered data to the receiver during a given *communication cycle* period. We utilize a circular buffer to keep track of the last few values of *node_coverage*. So, before inserting the current *node_coverage* in to the buffer, the value is compared with the minimum value in the buffer. This is to confirm whether batteryless nodes are maintaining their interaction with the receiver or dropping out of the network at a significant rate due to poor harvesting conditions. If *node_coverage* is greater than or equal to buffer *min*, then the *WKUP_TX_REPS* is doubled to save power, else it is set to 1—which signifies that some of the nodes are either unavailable or disconnected, thus requiring continuous transmission of WuPkts.

3.5 Greentooth Analytical Model

To mathematically characterize and analyze the behavior of Greentooth, we adopt a formal approach by treating the operation of the batteryless nodes as a stochastic process in discrete time and state. This enables us to model the high-level behavior using a Markov process, with state transition probability from state i to state j defined as $p_{ij} = P[X_{n+1} = j \mid X_n = i]$, where $\{X_0, X_1, X_2, \ldots, X_n\}$ denotes a vector of discrete random variable at various timesteps. Figure 7 illustrates the representative Markov chain showing different states of the node and transition

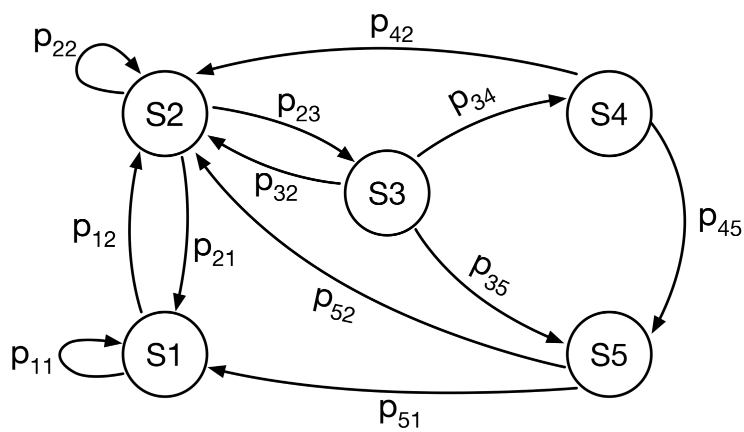


Fig. 7. Markov chain transition probability diagram representing the different states of Greentooth's batteryless nodes and the corresponding transition probabilities from one state to another.

probabilities among the states. The following are the key states that each batteryless sensor node can be in the following:

- Power failure – S_1: The sensor node losses power due to due to erratic and scarce harvesting conditions.
- **wake-up receiver (WuRx)** sleep – S_2: The sensor node remains in low-power mode with wake-up radio actively listening for WuPkts.
- Wake state – S_3: The sensor node is in active state after existing the sleep state due to WuPkt reception.
- Discovery – S_4: The sensor node establishes a connection with the receiver if time slot is yet to be allocated.
- Data transmission – S_5: The sensor node delivers data to the receiver during its allocated time slot.

Given the finite and countable nature of the Markov chain (Figure 7) with these five states, we consolidate the transition probabilities into a probability matrix denoted as P. The matrix P conveys the probabilities associated with transitioning from one state to another within a single timestep, and a transition probability of 0 denotes that a state is not accessible from another state.

$$P = \begin{bmatrix} p_{11} & p_{12} & 0 & 0 & 0 \\ p_{21} & p_{22} & p_{23} & 0 & 0 \\ 0 & p_{32} & 0 & p_{34} & p_{35} \\ 0 & p_{42} & 0 & 0 & p_{45} \\ p_{51} & p_{52} & 0 & 0 & 0 \end{bmatrix}.$$

Since the transition probabilities leaving a state are mutually exclusive and forms a universal set, each row of P represents the transition probabilities departing from a state and add up to 1. For instance, all transition probabilities departing from the $Power\ failure$ state S_1 equals $p_{11} + p_{12} = 1$. Conversely, each column represents the transition probabilities terminating at a state. Transitions terminating at the $WuRx\ sleep$ state S_2 is given as $p_{12} + p_{22} + p_{32} + p_{42} + p_{52}$. In general, the Chapman–Kolmogorov equation given as $P_{ij}(n + m) = \sum_{k \in K} p_{ik}(n) p_{kj}(m)$, allows us to recursively compute the transition probability of moving from state i to state j after n steps, where k is an intermediate state between i and j. Furthermore, the probability that the node is in state i

at timestep n is given as $\pi_i(n) = P(X_n = i)$, which can be grouped as a state probability vector $\boldsymbol{\pi}(n) = [\pi_0(n), \pi_1(n), \pi_2(n), \ldots]$. So, we can represent the state probability vector of the node at timestep n as $\boldsymbol{\pi}(n) = \boldsymbol{\pi}(n-1)\mathbf{P} = \boldsymbol{\pi}(0)\mathbf{P}^n$, where $\mathbf{P}^n$ is the n-step transition probability matrix for moving between different states and $\boldsymbol{\pi}(0)$ is the state probability vector at the initial state (timestep 0, signifying when the batteryless node is initially introduced into the network). As n continues to increase, the states of the nodes reaches a steady state i.e., $P_{\text{ss}} = \lim_{n\to\infty} P^n$, and we can compute the steady-state probability vector $\boldsymbol{\pi}$ using: $\boldsymbol{\pi} = \boldsymbol{\pi}(0)P_{\text{ss}}$ or $\boldsymbol{\pi} = \boldsymbol{\pi}P$, through a system of linear equations derived from

$$\boldsymbol{\pi} = [\pi_1\ \pi_2\ \pi_3\ \pi_4\ \pi_5] \begin{bmatrix} p_{11} & p_{12} & 0 & 0 & 0 \\ p_{21} & p_{22} & p_{23} & 0 & 0 \\ 0 & p_{32} & 0 & p_{34} & p_{35} \\ 0 & p_{42} & 0 & 0 & p_{45} \\ p_{51} & p_{52} & 0 & 0 & 0 \end{bmatrix}.$$

We can solve the following equations to obtain the steady-state probabilities π_1, π_2, π_3, π_4, π_5:

$$p_{11}\pi_1 + p_{21}\pi_2 + p_{51}\pi_5 = \pi_1, \tag{1}$$

$$p_{12}\pi_1 + p_{22}\pi_2 + p_{32}\pi_3 + p_{42}\pi_4 + p_{52}\pi_5 = \pi_2, \tag{2}$$

$$p_{23}\pi_2 = \pi_3, \tag{3}$$

$$p_{34}\pi_3 = \pi_4, \tag{4}$$

$$p_{35}\pi_3 + p_{45}\pi_4 = \pi_5, \tag{5}$$

$$\pi_1 + \pi_2 + \pi_3 + \pi_4 + \pi_5 = 1. \tag{6}$$

The computed steady-state probability vector $\boldsymbol{\pi} = [\pi_1\ \pi_2\ \pi_3\ \pi_4\ \pi_5]$ can be used to evaluate key network metrics like throughput, energy consumption, reliability, and so on.

Performance analysis using steady-state probability vector $\boldsymbol{\pi}$: We provide an analytical evaluation of the following network metrics using the derived steady-state probabilities π_1 to π_5:
Throughput. To evaluate network throughput, we first identify the relevant state (S_5) and transitions ($p_{12}, p_{35}, p_{45}, p_{52}$) in the Markov chain (Figure 7) that contribute to successful data transmission from each batteryless node to the receiver. For instance, a node will need to harvest enough energy to start up (p_{12}), be paired already (p_{35}) or just finish pairing (p_{45}) with the receiver, and be able to deliver its data to the receiver successfully (p_{52} and π_5). We compute the average successful transmissions per *communication cycle*, T_c, as the product of the steady-state probability of being in the relevant state and the associated state transitions, i.e., $(p_{12}+p_{35}+p_{45})\times\pi_5\times p_{52}$. The network throughput T_p is further computed as follows:

$$T_p = \frac{Average\ successful\ transmissions\ per\ communication\ cycle}{T_c}$$

$$T_p = \frac{(p_{12} + p_{35} + p_{45}) \times \pi_5 \times p_{52}}{T_c}.$$

The overall network throughput is then evaluated by averaging T_p with respect to the total number of batteryless nodes deployed in the network.

Energy consumption. For a batteryless node, the average energy consumption per successful transmission can be evaluated by considering the amount of energy consumed for data transmission given as E_{transmit} or E_{52} and the aggregate probability of successful transmission from all

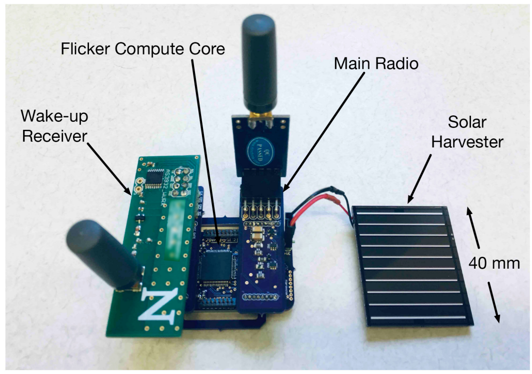


Fig. 8. Solar powered batteryless sensor node prototype built for evaluating Greentooth. It is an upgrade to the Flicker hardware platform [32], featuring the popular MSP430FR5994 MCU, CusTARD timekeeper [36], sensors, and radio peripherals (wake-up receiver and CC1101) in a dual-radio architecture.

relevant state transitions $P_{\text{tx}} = (p_{12} + p_{35} + p_{45} + p_{52})$. Hence,

$$\text{Average energy consumption per successful transmission} = \frac{E_{\text{transmit}}}{P_{\text{tx}}} = \frac{E_{52}}{p_{12} + p_{35} + p_{45} + p_{52}}.$$

This can also be extended to analyze the overall energy efficiency and lifetime of the receiver by considering the different states and transitions of the receiver Markov chain.

Reliability. The reliability (success probability) represents the likelihood of a successful data transmission from a batteryless node to the receiver, indicating network robustness. From the Markov chain, we consider the following states (S_1, S_3, S_4, S_5) and transitions (p_{12}, p_{35}, p_{45}, p_{52}) that contribute to the likelihood of a successful data transmission. Using the steady-state probabilities (π_1, π_3, π_4, π_5) corresponding to the identified states, we define the network reliability as

$$Reliability = \pi_1 \times p_{12} + \pi_3 \times p_{35} + \pi_4 \times p_{45} + \pi_5 \times p_{52}.$$

The overall network reliability can be generalized over the total number of sensor nodes deployed in the network.

4 IMPLEMENTATION

4.1 Hardware

To evaluate Greentooth's efficacy, we upgrade the Flicker hardware platform [32] with a MSP430FR5994 microcontroller, and build custom dual-radios boards (WuRx and main radio) as Flicker peripherals. Flicker's modular design allows several peripherals to be easily interfaced with the compute core—which consists of the TI MSP430FR5994 MCU, energy management unit, crystal oscillator, and CusTARD timekeeper, debugging, and programming interface.

Greentooth node prototype consists of the following subsystems as shown in Figure 8: energy harvesting and management, MCU, sensors, timekeepers, dual-radios, and programming interface.

Energy harvesting and management. Our prototype uses federated energy storage [31] where MCU and each peripheral (radio or sensor) on the board has a small dedicated capacitor for storing harvested energy from a small solar cell [20]. This energy storage approach helps harvest energy more efficiently, relaxes subsystem coupling, while simplifying task scheduling and peripheral prioritization. We use a 100-μF main storage capacitor to power the MCU via a 2V regulator, and two 220-μF ceramic capacitors connected in parallel for the main radio. The WuRx peripheral is powered directly from the main storage capacitor, since it requires only a few microamps to operate. We select these capacitor values, using empirical measurements, to balance two competing goals: having enough energy for communication opportunities and ensuring quick recovery (availability). For example, the 440-μF total capacitance selected for the radio peripheral is sufficient to execute the most expensive atomic operation performed by the main radio during the initial pairing process (TX + RX).

When the main capacitor stores enough energy, the MIC841 comparator chip [40]—which gates power to different parts of the system, turns on when the main capacitor voltage reaches 2.8 V, and off when it drops below 2.2 V to prevents the system from entering a brownout state. The MCU enters a *sleep* state (LPM3) after startup where it only draws 1 μA [85]. While sleeping, the WuRx listens for WuPkts, while other peripherals stay off but keep harvesting energy.

Sensors and Timekeepers. Our prototype includes two sensors—an internal temperature sensor and a soil moisture sensor [81], which we use in indoor temperature sensing and soil moisture monitoring applications. The prototype also features CusTARD [36], a remanence-based timekeeper that estimates power outage duration using capacitor voltage decay. The timer uses a 1-μF capacitor and a low leakage diode [14] that significantly extends timing duration by minimizing the reverse leakage current. This timer can estimate outage duration for up to 1 min.

Dual-radio communication. Greentooth uses a two-radio (main and wake-up radio) communication strategy. We use a common sub-1-GHz CC1101 radio [41] as our main radio and design a custom wake-up receiver board that uses a low-frequency wake-up radio (AS3932 [6]) for low-power listening while the main radio is turned off to conserve energy. The CC1101 radio operates at 433MHz and provides up to 500-m transmission range with configurable MAC and PHY layer properties, while the AS3932 operates at 125 kHz with a programmable 16-bit wake-up pattern. Multiple batteryless nodes running Greentooth communicate with a battery-powered receiver that is responsible for access control, coordination, and network synchronization. We implement our receiver using an MSP430FR5994 launchpad [42] connected to a CC1101 radio. We did not equip the receiver with a WuRx, because WuPkt transmission is expensive for battery-free nodes, consequently, we move key energy intensive operations to the receiver to enable the batteryless nodes to only perform useful sensing operations due to the scarce nature of harvested energy.

Although we use the CC1101 radio and MSP430 MCU, the Greentooth protocol, including WuPkt generation and detection is both radio and platform agnostic. The receiver generates a WuPkt using a two-stage modulation technique [24] that transmits a low-frequency WuPkt signal as a series of normal data packets at higher frequency using the CC1101 radio. First, the radio's 433-MHz carrier is turned **on and off (OOK)** creating an envelope signal with a period of 125 kHz. The 125-kHz signal is further OOK-modulated by a 40-bit wake-up signal containing 15 carrier bits, 1 separation bit, 8 preamble bits and 16 wake-up pattern bits when pattern correlation is enabled. This produces a 320-byte payload, which when transmitted by the receiver's main radio, will be detected by the wake-up receiver on the nodes.

Each node's wake-up receiver has a demodulation circuit, shown in Figure 9. Whenever a WuPkt arrives at the sensor node, that circuit filters out the high-frequency carrier and forwards the remaining 125-Hz signal to a low-power wake-up radio chip (AS3932). The received signal passes through an envelope detector and data correlator circuit and wakes the main MCU from sleep if the detected wake-up pattern is correct. When data correlation is disabled, the wake-up receiver wakes the MCU anytime it detects a broadcast WuPkt with a carrier frequency that lasts for more

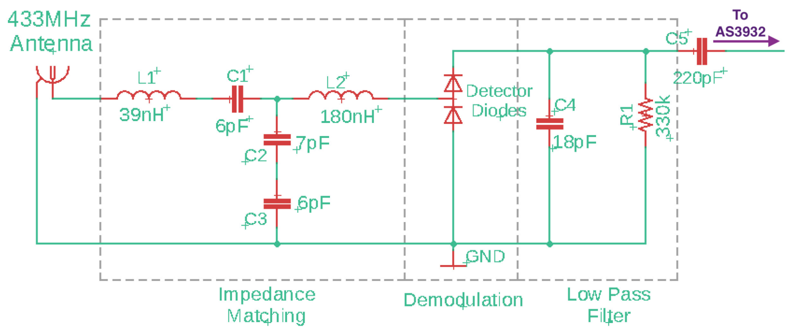


Fig. 9. Schematic design of our custom wake-up receiver peripheral.

than 550 µs. Greentooth's broadcast WuPkt is made up of a 10-byte payload with each byte containing 0xAA—whose binary representation (0b10101010) reflects the OOK carrier preamble expected by the receiver). A minimum 10-byte payload is needed to generate sufficient carrier burst and preamble that can wake a sleeping node. Our WuRx design is based on the dual-radio design presented in Reference [24], but we optimize the wake-up process for energy and latency by using a broadcast WuPkt with simplified payload (10 bytes) instead of the 320-byte payload needed for ID-based wake-up.

4.2 Greentooth Firmware

We develop Greentooth firmware in C, for both the receiver and sensor node. The firmware allows developers to configure MAC and PHY layer properties like data rate, modulation technique, transmit power, communication cycle duration, discovery duration, and other parameters listed in Table 1. The Greentooth protocol is built as an abstraction layer over low-level CC1101 and AS3932 libraries. All hardware designs, source code, and energy traces will be released open source at the time of publication.

4.3 Packets Structure

The Greentooth protocol makes use of the following key packets structure.

Wake-up Packet. The WuPkt, broadcast by the receiver at the beginning of a *communication cycle* is used for waking and synchronizing sleeping batteryless sensor nodes. While any amplitude modulation technique can be employed, we use OOK. The WuPkt simply carries the minimum 10 byte wake-up payload needed to activate the WuR on the wake-up receiver.

Connection Request and Response Packet. The *connection request* packet is sent to the receiver by a node that wants to join the network. It contains only two bytes, *Node Id* and *Pkt Type* (Discovery). The receiver replies with a *connection response* packet shown in Figure 10. This provides the node with specific time slot information and other connection state information used by the receiver in managing the network.

Data and ACK Packet. To prevent accidental wake ups of WuRs during the data transmission phase, we use Binary Frequency Shift Keying (2-FSK) modulation for data exchange between sensor nodes and the receiver. As shown in Figure 11, a total of 58 bytes of payload can be sent to the receiver in addition to *Node Id* and *Pkt Type* (Data). The lightweight ACK packet notifies the node of a successful transmission while delivering a time drift report. Finally, a node can also disconnect from the receiver by sending a special packet containing *Node Id* and *Pkt Type* (Disconnect).

Node ID (1 byte)	Pkt Type (1 byte)	Time Slot (1 byte)	Slot Duration (1 byte)	Discovery Period (2 bytes)	Comm_cycle Period (2 bytes)	Connection Timeout (1 byte)	Comm Offset (2 bytes)

Fig. 10. Greentooth's connection response packet.

Data Packet

Node ID (1 byte)	Pkt Type (1 byte)	Data Payload (58 bytes)

ACK Packet

Node ID (1 byte)	Pkt Type (1 byte)	Clock Drift (2 bytes)

Fig. 11. Overview of Greentooth's data and ACK packet structure.

5 EVALUATION

To carry out empirical evaluation of Greentooth on a real testbed, we assembled nine upgraded solar-powered Flicker batteryless nodes (shown in Figure 8) alongside a battery-powered receiver node. We performed all current measurements using a Teensy-based [74] custom power meter that features three amplification stages (providing up to 1,000× gain) for high-speed sampling of voltage drop across a small sense resistor using 12-bit ADC modules. The sense resistor (1 or 100Ω) allows us to measure a wide range of current (tens of nA to hundreds of mA). For the rest of this section, we first characterize the performance of our custom wake-up receiver, followed by the performance evaluation of key features of Greentooth, and, finally, we evaluate and compare Greentooth's performance against key SoA baselines under varying and intermittent harvested energy.

5.1 Wake-up Receiver Characterization

In addition to the system specifications of the AS3932 wake-up radio reported in the Datasheet [6], we perform additional experiments to evaluate the power consumption and wake-up range of our custom WuRx board and present the results summary in Table 2.

5.1.1 Wake-up Receiver Consumption. To measure power consumption of our WuRx board, we connect a power meter between the VCC pin of the WuRx board and the supply VCC pin on the Flicker board. We then record the average current draw for two key operations: (i) when the WuRx listens continuously for WuPkt and (ii) when it processes received WuPkt. Results show that our custom WuRx consumes 6.6 μW for continuous listening and 15 μW while processing received WuPkt.

5.1.2 Wake-up Packet Transmission Cost. Unlike other receive-initiated MAC protocols that use ID-based WuPkts to wake up individual sensor nodes, Greentooth uses simpler and shorter broadcast WuPkts. We evaluate and compare the energy overhead of transmitting an ID-based WuPkt against a broadcast WuPkt on the receiver by measuring the average transmission time of each packet at 10-dBm TX power—which draws 29.2 mA at 3.3 V. To accurately measure the transmission time, we probe a digital I/O pin on the receiver using a MSOX4034A mixed signal oscilloscope. Results show that ID-based WuPkt transmission takes 17 ms, and consumes 1.638 mJ, while the broadcast WuPkt transmissions takes 4 ms at 0.385 mJ—a *76.5%* reduction in energy consumption. These energy savings grow as the number of sensor nodes in the network increases, since ID-based WuPkt transmission consumption will increase linearly. Instead, Greentooth uses a single broadcast WuPkt to synchronize all nodes during a *communication cycle*, enabling better throughput and energy-efficiency for both the nodes and the receiver.

Table 2. Performance analysis summary of our custom wake-up receiver board

Performance metric	Value
WuRx listening consumption	6.6 µW
WuPkt processing consumption	15 µW
ID-based WuPkt transmission	1.638 mJ
Broadcast WuPkt transmission	0.385 mJ
Max wake-up range	55 m

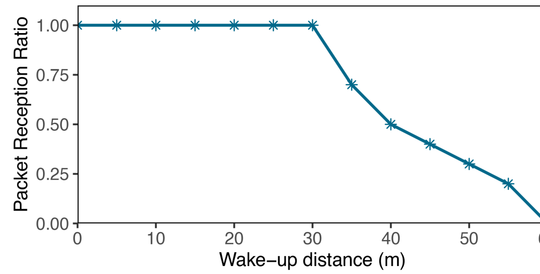


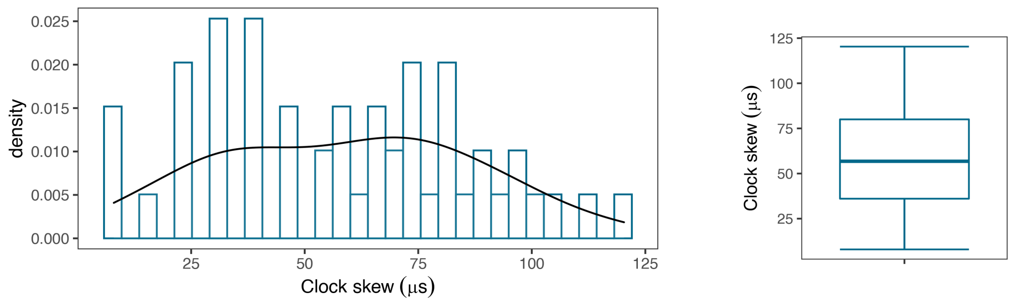
Fig. 12. Wake-up range based on packet reception ratio over distance.

5.1.3 Wake-up Range. We evaluate the wake-up range of our custom WuRx board by computing the packet reception ratio over distance from a receiver. The receiver transmits 10 WuPkts at 10 dBm and listens for 0.5 s after each transmission for a response (ACK) from a batteryless node placed certain distance away. The batteryless node—equipped with a 433-MHz quarter-wave monopole antenna [53]—replies with an ACK containing its ID whenever it receives a WuPkt. The receiver then records the total number of received packets (ACKs that contains the ID of the node), alongside the transmitted WuPkts. Initially, we place the node 5 m from the receiver, and then increment the distance in steps of 5 m until the node is completely out of wake-up range. We perform each experiment multiple times in a long hallway and obtain the average packet reception ratio (*received packets/transmitted packets*). Figure 12 shows a plot of the packet reception ratio over distance of our custom WuRx board—which is able to achieve a maximum line-of-sight wake-up range of 55 m.

5.2 Performance Evaluation of Key Greentooth Features under Varying and Intermittent Harvested Energy

To enable reliable and energy efficient networking of intermittent battery-free devices, Greentooth utilizes the key features (**Sol1** to **Sol5**) presented in Section 1 and described in Sections 3.4 and 3.2 to address critical challenges of intermittent networks. In this section, we evaluate these features and how they contribute to the overall performance of Greentooth.

*5.2.1 Network Synchronization - **Sol1**.* Efficient and accurate time synchronization is essential for effective operation of TDMA-based MAC protocols; specifically, it enables each sensor node to maintain its time slot, thereby minimizing congestion and collision. To evaluate the precision of Greentooth's synchronization mechanism enabled by the lightweight synchronization feature that uses simplified WuPkt broadcast, we measure the clock skew (relative difference between nodes wake-up times) among multiple batteryless nodes while they were being synchronized by a broadcast WuPkt from the receiver. Using the MSOX4034A mixed signal oscilloscope, we probe a digital I/O pin on three batteryless nodes as well as the receiver; to measure the relative difference between the wake-up times of the nodes. For each run of the experiment, the nodes remain in deep sleep after startup, and are awaken (indicated by the I/O pin going from LOW to HIGH) by a WuPkt from the receiver. We then collect the wake-up time of each node and compute the clock skew. We repeat this 50 times to ensure the results were reliable. Results (Figure 13(a) and (b)) show that Greentooth nodes can be synchronized at an average clock skew of 59 µs. This precise way of aligning sensor nodes with the beginning of the *communication cycle* period using the simplified WuPkt broadcast synchronization technique (**Sol1**) ensures effective use of time slots without requiring guard intervals that will incur additional latency and energy overhead.



(a) Clock skew distribution **(b)** Clock skew summary

Fig. 13. Network synchronization accuracy. Greentooth WuPkt broadcast mechanism synchronizes batteryless nodes with a mean clock skew of 59 μs.

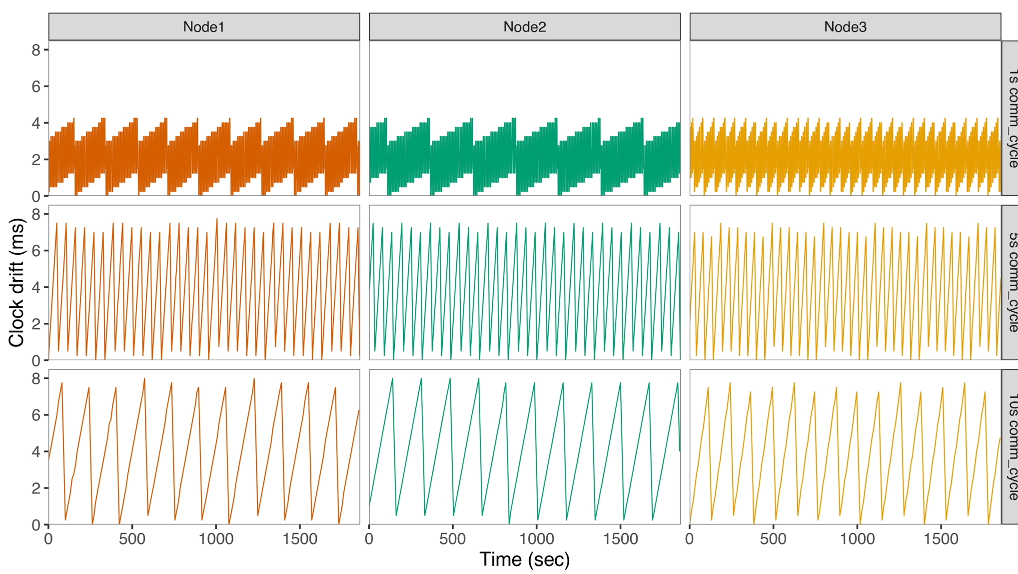
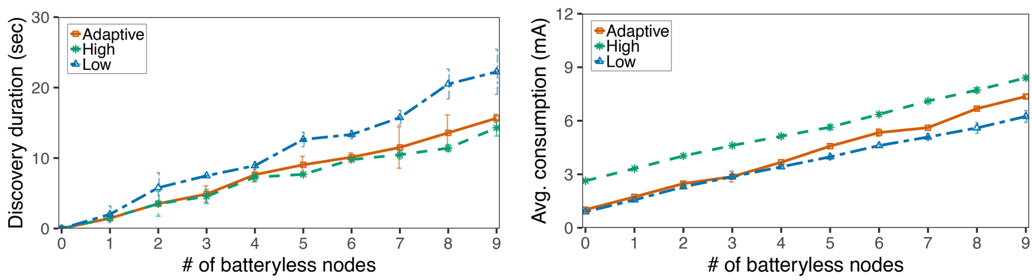


Fig. 14. Clock drift resolution over time on batteryless sensor nodes. The nodes maintain their time slots with the sink node in the absence of a WuPkt using a combination of backup timing and clock drift resolution feature.

5.2.2 Backup Timing and Clock Drift Resolution - ***Sol2***. When synchronizing WuPkts are missed, Greentooth uses a backup timer to keep all batteryless nodes in a network in sync with the receiver for as long as possible. This is critical for network conditions where WuPkt transmission is not guaranteed to reach the sensor nodes perhaps due to barriers, RF interference, or mobility-induced multipath fading.

Utilizing the backup timing feature enhances the network throughput and energy efficiency of both the node and the receiver as the node can maintain synchronization without requiring a WuPkt for an extended duration. Nevertheless, when a WuPkt is absent for an extended time, local clock drifts can cause nodes to drift completely out of their designated time slots. Consequently,



(a) Discovery duration **(b)** Consumption

Fig. 15. (a) Adaptive discovery mode achieves rapid discovery similar to High discovery mode, while Low exhibits prolonged discovery duration due to the spread of batteryless nodes across multiple discovery cycles. (b) Adapting the discovery duration enables significant savings in power consumption similar to Low discovery mode without the prolonged discovery duration.

the ACK-based clock drift resolution feature allows nodes to continue using their allocated time slots for an extended period.

Using a setup consisting of three batteryless nodes and a receiver, we evaluate how a batteryless node can employ the backup timer in conjunction with clock drift resolution to maintain its time slot. We configure each batteryless node to disable its WuRx after completing the initial paring process with the receiver. The nodes then report their local time drift to the receiver at every interaction. We run this experiment for 30 min and try different configurations of the *communication cycle* period—1 s, 5 s, and 10 s.

As shown in Figure 14, clock drifts are resolved consistently, using the backup timer to stay in sync with the receiver without a WuPkt. For short communication cycles (1 s) the nodes are able to resolve their clock drifts as soon as it grows to about 4.25 ms, while longer communication cycles (10 s) gets the drifts resolved around 8 ms, which is far below the 50 ms slot duration. The batteryless nodes' capacity to employ the backup timer alongside the Ack-based clock drift resolution feature (**Sol2**) to sustain communication with the receiver demonstrates that WuPkts are solely required during the initial pairing phase, and that the nodes are able to consistently resolve local clock drifts over a long period.

*5.2.3 Neighbor Discovery - **Sol3**.* Efficient and fast neighbor discovery is critical to achieving better network throughput, responsiveness, and optimal power usage on both the nodes and the receiver. Greentooth provides both fixed and adaptive neighbor discovery modes that can be configured by a developer with respect to application needs. We perform experiments to evaluate and compare the performance benefit of the following discovery modes: *Low*, *High*, and *Adaptive*. Using a setup comprising of a receiver and up to nine batteryless nodes, we record the *discovery duration* (how long it takes the receiver to recognize all participating nodes) and the average current draw of the receiver. We use the following discovery period configurations: Fifty milliseconds for *Low* discovery, 150 ms for *High* discovery, and 50–150 ms for *Adaptive* discovery. Also, we use 1-s *communication cycle* period and 50 ms *discovery step size* throughout the experiments. For each discovery mode, we perform multiple runs of the experiment, with each run lasting for 30 *communication cycles*.

As shown in Figure 15(a), as the number of nodes increases, it takes significantly longer for all the nodes to get discovered using the *Low* discovery mode. A Short discovery period requires sensor nodes to be distributed across multiple *communication cycles*, since only a limited number of nodes can be discovered in one cycle. Despite its prolonged discovery duration, *Low* discovery mode achieves better power consumption on the receiver (Figure 15(b)), since the radio only stays in active listening mode for a short duration. In contrast, *High* discovery mode enables faster discovery, due to the longer discovery period, and results in significantly higher power consumption and shorter battery life for an energy constrained sink node. Selecting *Low* discovery mode is beneficial for sensor deployments where the lifetime of an untethered or mobile receiver is critical, while *High* discovery mode is more suited for deployments where rapidly changing data must be collected as soon as the nodes join the network. *Adaptive* discovery clearly balances the strengths of both *High* and *Low* discovery modes through faster discovery times and optimal consumption similar to that of *Low* discovery mode. Adapting discovery duration (**Sol3**) based on network situation (availability of newly deployed or previously disconnected nodes) improves quality of service, responsiveness, and lifetime.

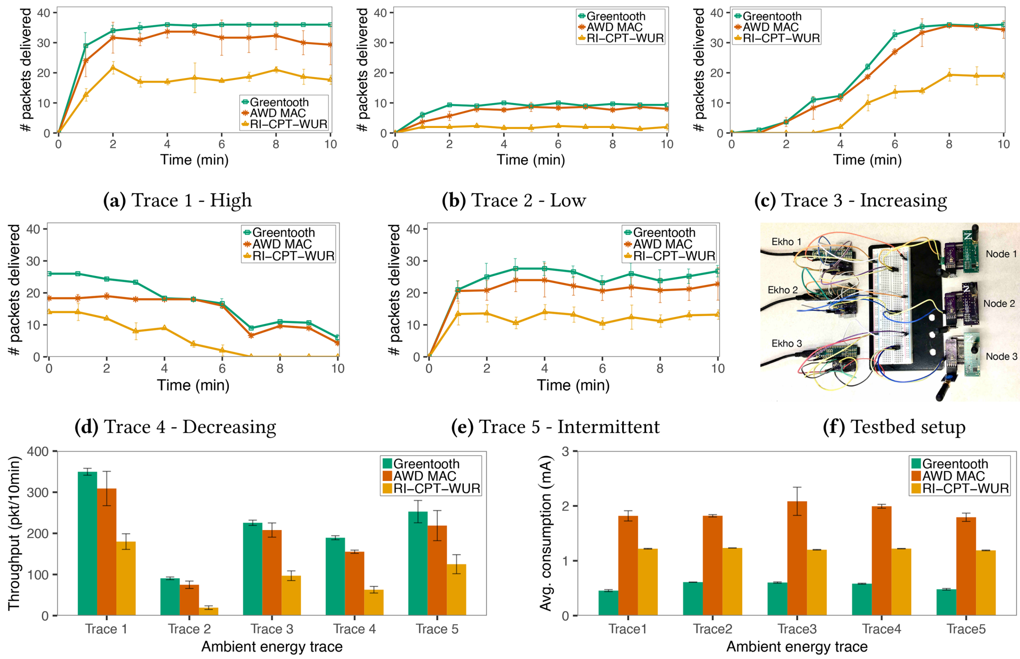
We further evaluate the impact of power failures and varying harvesting conditions on network throughput and overall power draw of Greentooth compared to two baselines—AWD-MAC [49] and **Receiver-Initiated Consecutive Packet Transmission Wake-up Radios (RI-CPT-WuR)** MAC [29]. AWD-MAC is an ID-based receiver-initiated WuR MAC that consists of a *neighbor discovery phase* and a *asynchronous communication phase*. During discovery, the receiver broadcasts wake-up beacons to activate neighboring nodes. Using time slots, each node registers its address and data rate with the receiver that stores it in a look-up table. During the *asynchronous communication phase*, the receiver then requests data from each node using an address beacon that contain the address of the node stored in the look-up table. It aims at reducing collisions, but makes use of expensive address beacons to query nodes individually, and was originally evaluated only in simulation.

RI-CPT-WuR MAC is a broadcast-based receiver initiated WuR MAC that uses communication cycles divided into time slots for data transmission. The receiver initiates data collection by broadcasting a wake-up message that starts channel access competition among the nodes. The node that wins the channel competition uses the next few slots (consisting of Data and Ack beacon transmissions) to communicate with the receiver until it sends all the data in its queue. Channel access competition then resumes to accommodate the remaining sensor nodes. RI-CPT-WuR MAC aims at achieving improved per-node throughput, but it was also evaluated in a custom simulator.

For our evaluation, we implement both AWD-MAC and RI-CPT-WuR MAC, and explore various performance metrics like network throughput, energy consumption, and receiver lifetime using both synthetic and real-world solar traces.

Testbed Setup. To evaluate and compare Greentooth with AWD MAC and RI-CPT-WuR on a real testbed powered by ambient harvested energy, it is critical that our experiments are repeatable and realistic considering the erratic and unpredictable nature of energy harvesting. We use *Ekho recorders* to record solar energy traces (both in the lab and in the wild) and *Ekho emulators* to replay the corresponding I–V surfaces of recorded traces for the batteryless nodes. Ekho [30] is a tool that records energy harvesting conditions so they can be accurately replayed in the lab in a realistic and repeatable fashion. Our testbed features a star network consisting of a receiver (sink) and three batteryless nodes powered by three Ekho emulators (Figure 16(f)).

*5.2.4 Performance under Intermittency - **Sol4**.* To assess the protocols under varying and intermittent indoor energy harvesting conditions, with a particular focus on how the connection state preservation feature (**Sol4**) of Greentooth impacts the overall network performance when power failures become unavoidable, we use an *Ekho recorder* to record five different types of solar traces generated in the lab from a programmable lightbox consisting of a small solar harvester [20] placed about 10 cm from an Arduino controlled light source [83] *Trace 1* captures periods of continuous energy abundance (High) with light level over 1,000 lux, similarly to the harvesting condition on



(a) Trace 1 - High **(b)** Trace 2 - Low **(c)** Trace 3 - Increasing

(d) Trace 4 - Decreasing **(e)** Trace 5 - Intermittent **(f)** Testbed setup

(g) Overall throughput **(h)** Receiver consumption

Fig. 16. Performance of Greentooth against AWD MAC and RI-CPT-WuR under various recorded harvesting conditions. Results from over 13 h of control experimentation shows that Greentooth outperforms both SoA representatives on both receiver consumption and throughput across all categories of the recorded harvesting traces.

a sunny day with clear sky. *Trace 2* captures periods of persistent energy scarcity (Low) where our batteryless nodes barely work (around 250 lux). This is comparable to the harvesting condition on a heavily overcast day. *Trace 3* captures periods of increasing energy availability (Increasing), similarly to the harvesting condition at dawn. *Trace 4* focuses on periods of decreasing energy availability (Decreasing) that happens at twilight. Finally, *Trace 5* characterizes periods of rapidly changing harvesting condition (Intermittent), similarly to the harvesting condition on a sunny day with intermittent clouds or when batteryless nodes are mobile or partially shaded by tree branches or other overhead objects. *Trace 5* is a combination of *Trace 1* (15 s), *Trace 2* (10 s), and off periods (5 s)—where nothing is harvested (around 0 lux), combined in a completely random fashion.

We replay the I–V surface of each trace for 10 min while the nodes sense and report indoor ambient temperature to the receiver. We configure the protocols to use similar configurations: 5-s communication cycle period, 50 ms for slot duration and discovery step size, 500-ms max discovery period, and connection timeout of 3. The short connection timeout value provides the opportunity to also assess the impact of the dynamic slot recycling and management feature on the overall performance of the network as nodes can easily be disconnected from the network as soon as the emulated I–V surface decreases significantly over three complete *communication cycles*. For *Trace* 1 to 4, we repeat each experiment 3 times, while *Trace 5* is repeated 5 times to capture the variations resulting from replaying different curves for each experiment run. For each run, we record the total number of data packets delivered to the receiver by each protocol, as well as the average current draw of the receiver using a power meter.

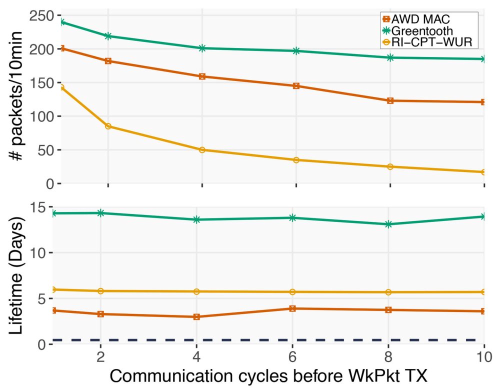


Fig. 17. Under varying WuPkt broadcast recurrence, Greentooth is able to achieve better throughput and lifetime compared to the baselines.

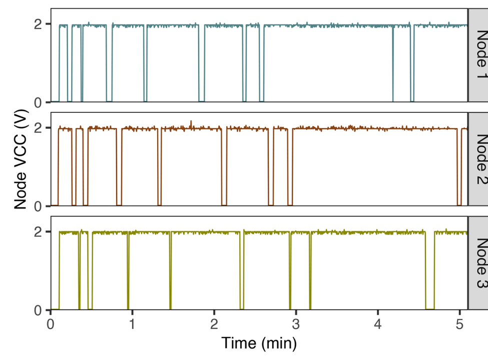


Fig. 18. Power failure distribution within the first 5 min of the real-world intermittent solar traces recorded from three different locations in a garden and replayed on three batteryless nodes.

From the results collected over 13 h of control experimentation, we compute the average current consumption of the receiver and network throughput—amount of temperature data packets successfully delivered to the receiver during the trace emulation period. Overall, Greentooth outperforms the baselines (as shown in Figure 16(g) and (h)) across all five categories of the recorded solar traces. Achieving 129% better throughput with 2.3× less consumption on the receiver compared to RI-CPT-WuR and a 15% improvement in throughput with 3.5× less receiver consumption compared to AWD MAC. The poor performance of RI-CPT-WuR over various harvesting conditions (Figure 16(a) to Figure 16(e)) stems from its use of expensive channel access competition every communication cycle, which depletes the limited amount of energy saved in the storage capacitor of a batteryless node. This continuous of use of saved energy for channel access competition wastes energy that could have been used for actual sensing operations. AWD MAC, however, makes use of expensive address beacons (which takes 325% longer to transmit compared to broadcast beacon) for requesting data individually from sensor nodes. The superior performance of Greentooth compared to the two state-of-the-art protocols results from its use of failure resilient connections that enable reliable communication across power failures through connection state preservation (which minimizes reconnection overhead) and simplified WuPkts (yielding significant energy savings on both the receiver and the nodes). Furthermore, the TDMA communication style of Greentooth coupled with the ability to manage and recycle time slot effectively further contribute to its overall performance benefits.

*5.2.5 Variable WuPkt Transmission - **Sol5**.* Receiver lifetime can be improved significantly by minimizing the number of WuPkt broadcast—one of the most expensive operations performed by the receiver. We conduct additional experiment using the same setup described above to empirically evaluate how variations in WuPkt broadcast rate, enabled by the configurable WuPkt transmission recurrence mechanism (**Sol5**), affect throughput, receiver consumption, and overall lifetime. We replay *Trace 5* I–V surface for the nodes and configure the receiver to broadcast WuPkt in steps of two communication cycles. For example, it broadcasts a WuPkt every cycle for the first run, every two cycles for the second run, every four cycles for the third run, and so on.

Results (Figure 17) show that Greentooth achieves significantly better throughput compared to the SoA baselines as WuPkts are transmitted less frequently. Greentooth also achieves over 2×

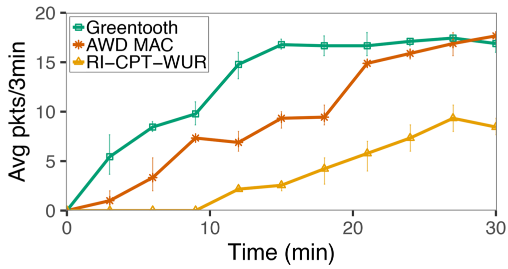
improvement in the expected lifetime of the receiver compared to the baselines when powered from a 3.7-V battery with 150 mA h capacity. This shows that even when the configurable WuPkt transmission recurrence feature is enabled to conserve energy on the receiver, which results in fewer WuPkt broadcasts, the Greentooth nodes keep communicating with the receiver using their backup timing and connection state preservation capabilities. The dashed line in Figure 17 shows the lifetime of a receiver in a Free-for-All network—where batteryless nodes deliver data to the receiver using best effort transmission without coordination. The lifetime of such receiver is very short (11 h), because it actively listens continuously for data from the nodes.

5.3 Performance Evaluation in the Wild

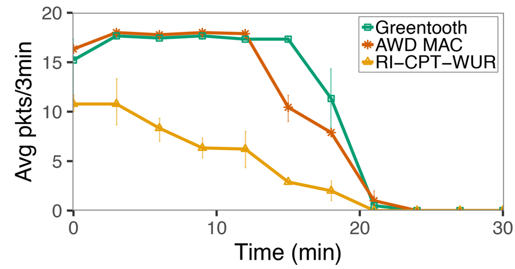
To assess the robustness of Greentooth in the wild, we select soil moisture monitoring application (smart agriculture) and evaluate the protocols under varying and intermittent real-world harvesting conditions. To ensure fair evaluation, batteryless nodes running different protocols must be deployed in close proximity—to minimize variations in harvested energy. However, deploying multiple nodes running different protocols in close proximity in a garden results in WuPkt interference. To prevent WuPkts from one protocol from interfering with nodes running another protocol, we record real-world solar energy traces using an Ekho recorder (Figure 19(f)) from different locations in the garden where the sensor nodes were meant to be deployed. This enables us to replay the recorded I–V surfaces in a repeatable manner across the competing protocols (Greentooth, AWD-MAC, and RI-CPT-WuR MAC). We capture different parts of the day by recording a solar trace at *sunrise*, *sunset*, and *midday*. To ensure our evaluation is carried out under real intermittent harvesting conditions, we also record three traces at different locations on a windy day, where nearby leaves intermittently shade the solar harvester.

Using a 33 min recorded trace, we replay the I–V surface of the traces on three separate Ehko emulators to capture the most critical section of the recorded energy traces and also to ensure the spatial differences in deployment locations are maintained. For the recorded intermittent solar traces, we explore the number of power failures present in the first 5 minutes (Figure 18) to confirm the presence and the arbitrary nature of power failures. We utilize the same testbed setup described earlier, but each sensor node is now equipped with a SEN-13322 soil moisture sensor [81] for soil wetness level measurement. Since soil moisture level changes less rapidly, we set up the protocols to use the following configurations: 10s *communication cycle* period, 50 ms *slot duration* and 500 ms max *discovery period*. We collect data for three experimental runs for the I–V surfaces replayed for each protocol. This is to ensure our experimental results are reliable.

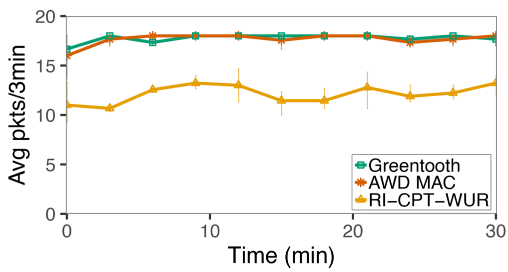
For the results presented in Figure 19(a) to (d), we group the number of packets delivered into 3 min buckets and compute the mean and standard deviation of each bucket to provide a clearer representation of the distribution. At sunrise (Figure 19(a)), Greentooth nodes begin transmission before AWD-MAC and RI-CPT-WuR. This is due to their capability to save connection state across power failures, which keeps communication going during periods of scarcity without reconnection overhead after reboots. Consequently, Greentooth achieves 32% and 230% better throughput compared to AWD-MAC and RI-CPT-WuR, respectively. Also at Sunset (Figure 19(b)), Greentooth persists a bit longer, thereby achieving up to 15% and 134% better throughput compared to AWD-MAC and RI-CPT-WuR, respectively, as harvested energy becomes increasingly scarce. Greentooth and AWD-MAC shows similar performance at midday (Figure 19(c)) when harvested solar energy is abundant. Despite the availability of surplus energy, RI-CPT-WuR still underperforms due to its channel access competition requirement, which degrades performance as collision becomes unavoidable. When harvested energy becomes intermittent (Figure 19(d)), Greentooth shows significantly improved performance as it achieves about *73%* and *283%* better throughput compared to AWD-MAC and RI-CPT-WuR, respectively. This shows that Greentooth is robust enough to



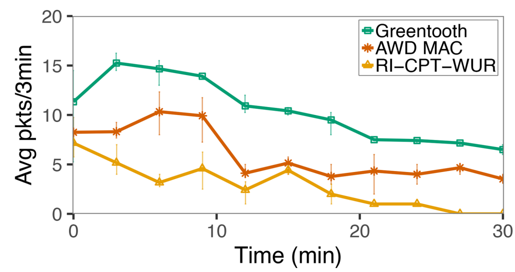
(a) Sunrise



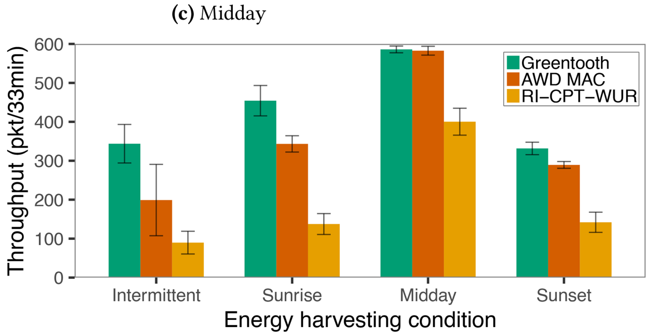
(b) Sunset



(c) Midday



(d) Intermittency



(e) Network throughput

(f) Real trace recording

Fig. 19. Field deployment. Greentooth shows superior performance compared to the SoA across different harvesting conditions (a), (b), and (d). At midday (c), when harvested solar energy becomes surplus, Greentooth and AWD-MAC shows similar results. As shown in (e), Greentooth achieves up to 73% and 283% better throughput compared to AWD-MAC and RI-CPT-WuR, respectively, when harvested energy becomes intermittent. (f) Real solar energy trace recording using Ekho recorder in a garden. We record traces at three different locations at sunrise, midday, sunset, and when harvesting condition is intermittent.

handle the complexities of intermittent harvesting conditions while enabling reliable communication between an energy constrained receiver and many battery-free sensors. In general, Greentooth is able to achieve better throughput (Figure 19(e)) under different real-world energy harvesting conditions.

6 RELATED WORK

Wireless sensing systems have revolutionized data acquisition and monitoring applications in many fields; nonetheless, they usually experience limited lifetime due to the expensive nature of radio communication. Several MAC techniques [2, 7, 66, 73] in the literature have been explored and used for extending network lifetime while optimizing for throughput, latency, and fairness.

First, duty-cycled MAC protocols [38, 52, 68, 82] have been proposed as better alternatives to always-on protocols due to their superior energy efficiency and channel utilization. In contrast to always-on protocols—in which sensor nodes continuously listen for or transmit data packets, they operate by systematically putting the node's main radio into sleep mode, which is later woken up briefly to either receive or transmit data. Synchronous duty-cycled MACs keep a common time reference among the nodes, which introduces time synchronization overhead and complexity, while the asynchronous counterparts utilize schemes like preamble sampling, random duty-cycling, or receiver-initiation to circumvent synchronization challenges [13]. Notwithstanding, they are still susceptible to idle listening—energy consumed listening for data packet during active period without success, and other major issues like latency—time spent waiting for sleeping nodes to wake-up, and overhearing—energy consumed receiving data meant for another node [68].

Advances in wake-up radios technology [28, 39, 43, 59, 60, 71] have provided ways to resolve most of the challenges faced by duty-cycled MAC protocols. Wake-up radios are ultra-low power receivers with orders of magnitude lower power consumption compared to existing low-power radio transceivers. They are mostly used alongside the main radios (dual-radio architecture) for continuous monitoring of the wireless medium for wake-up signals while the main radio is off or in a deep sleep [24, 69]. Unlike traditional MAC protocols with single radio transceivers, WuR MAC protocols utilize the dual-radio WuR architecture to minimize overhearing, idle listening, and latency issues. Transmitter initiated WuR MAC protocols [1, 27, 47, 61, 63] allow a node to initiate communication with a receiver in a single hop on-demand fashion. While bi-directional WuR MAC protocols [5, 15, 44, 45, 64, 72] enable both transmitters and receivers to initiate communication using WuPkt as they are both equipped with WuRs, thus making them suitable for multi-hop wireless communication. Receiver-initiated WuR MAC protocols [19, 29, 49, 50, 79] enable receiving nodes (sinks) to initiate single hop communication by announcing their readiness to collect data using WuPkts.

Receiver-initiated WuR MACs are classified as either broadcast-based—where a single WuPkt wakes several nodes equipped with WuR, or addressed-based (ID-based)—in which nodes are activated individually based on the address information in a WuPkt [79]. RI-LD-WuR MAC [79], RI-WuR MAC, and RI-CPT-WuR MAC [29] all utilize broadcast WuPkt transmission with CSMA/CA for asynchronous communication when multiple nodes compete for medium access utilization. However, channel access competition degrades performance as collision increases as the number of nodes increases. In contrast, AWD MAC [49] and DoRa [50] utilize ID-based WuPkt for polling individual node for data. This minimizes collisions and improves reliability, but also decreases the network throughput while increasing the overall energy consumption. Despite their potentials, these receiver-initiated WuR MAC protocols have only been designed and tested in simulation; without real-world empirical evaluations.

Unlike existing receiver-initiated WuR MACs whose operation is contingent on satisfying the ENO condition, Greentooth explores a broadcast-based synchronous mechanism for networking real battery-free intermittent sensor nodes with an energy constrained receiver. Time synchronization is crucial for neighbor discovery and TDMA communication in both battery-powered and batteryless networks. So, Find+Flync [25] have explored mechanisms for speeding up neighbor discovery among battery-free nodes. Find employs randomized delays to minimize discovery latency, while Flync uses harvested powerline-induced brightness variations in indoor lighting to further speed up neighbor discovery. Recently, Bonito [26] and FreeBie [18] have also explored the use of connections to maintain data exchange, however, Find and Bonito still incur the expensive beacon transmission and listening costs prior to achieving initial encounter and after connection is lost. Also, Flync is limited to indoor applications while FreeBie employs supercapitors for storing harvested energy, which makes it operate on the ENO condition with little to no intermittency.

Furthermore, Ambient backscatter [89] and Mesh [54, 55] networking techniques have been proposed for low-power intermittent systems. Despite their potentials, the modulated ambient RF signals used in Ambient backscatter are dynamic, unpredictable, and uncontrollable that complicates design and deployment, and limits network performance and reliability. Mesh networking method was only validated in MATLAB simulations. Greentooth however leverages the ultra-low power capability of WuR to provide a robust and energy efficient way of networking real intermittent batteryless systems that are prone to frequent and unpredictable power failures and timing inaccuracies.

7 CONCLUSIONS

We have presented Greentooth, a robust and energy efficient wireless communication protocol for intermittently powered sensor networks. It employs a TDMA-style communication scheduling and a dual-radio design to provide an energy-efficient connection mechanism that tolerates power outages and outage-induced timing inaccuracies, while improving network throughput and energy efficiency for both sensor nodes and an energy-constrained receiver. Results show that Greentooth achieves up to 73% and 283% better throughput when harvested energy becomes intermittent, and over 2× longer receiver lifetime compared to the SoA baselines (AWD MAC and RI-CPT-WuR, respectively).

REFERENCES

[1] Xue Yang and N. H. Vaidya. 2004. A wakeup scheme for sensor networks: Achieving balance between energy saving and end-to-end delay. In *Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'04)*. IEEE Computer Society, 19.

[2] Smriti Sachan, Rohit Sharma, and Amit Sehgal. 2021. Energy efficient scheme for better connectivity in sustainable mobile wireless sensor networks. *Sust. Comput.: Inf. Syst.* 30 (2021), 100504. https://doi.org/10.1016/j.suscom.2020.100504

[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. 2020. Battery-less zero-maintenance embedded sensing at the MithræUm of circus maximus. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems (SenSys'20)*. Association for Computing Machinery, New York, NY, 368–381. https://doi.org/10.1145/3384419.3430722

[4] Saad Ahmed, Naveed Anwar Bhatti, Muhammad Hamad Alizai, Junaid Haroon Siddiqui, and Luca Mottola. 2019. Efficient intermittent computing with differential checkpointing. In *Proceedings of the 20th ACM SIGPLAN/SIGBED International Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES'19)*. Association for Computing Machinery, New York, NY, USA, 70–81. https://doi.org/10.1145/3316482.3326357

[5] Moshaddique Al Ameen, Niamat Ullah, M. Sanaullah Chowdhury, and Kyungsup Kwak. 2011. A MAC protocol for body area networks using out-of-band radio. In *17th European Wireless 2011—Sustainable Wireless Technologies*. 1–6.

[6] ams. 2015. "3D Low Frequency Wakeup Receiver" AS3932 Datasheet. Retrieved from https://bit.ly/2XP0RXl

[7] Giuseppe Anastasi, Marco Conti, Mario Di Francesco, and Andrea Passarella. 2009. Energy conservation in wireless sensor networks: A survey. *Ad Hoc Netw.* 7, 3 (May 2009), 537–568. https://doi.org/10.1016/j.adhoc.2008.06.003

[8] Stefano Basagni, Valerio Di Valerio, Georgia Koutsandria, and Chiara Petrioli. 2017. Wake-up radio-enabled routing for green wireless sensor networks. In *Proceedings of the IEEE 86th Vehicular Technology Conference (VTC-Fall'17)*. 1–6. https://doi.org/10.1109/VTCFall.2017.8288130

[9] Stefano Basagni, Valerio Di Valerio, Georgia Koutsandria, Chiara Petrioli, and Dora Spenza. 2017. WHARP: A wake-up radio and harvesting-based forwarding strategy for green wireless networks. In *Proceedings of the IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems (MASS'17)*. 257–265. https://doi.org/10.1109/MASS.2017.18

[10] Stefano Basagni, Chiara Petrioli, and Dora Spenza. 2016. CTP-WUR: The collection tree protocol in wake-up radio WSNs for critical applications. In *Proceedings of the International Conference on Computing, Networking and Communications (ICNC'16)*. 1–6. https://doi.org/10.1109/ICCNC.2016.7440687

[11] Novel Bits. 2020. *Understand BLE Connection Intervals and Events in Under 5 Minutes*. Retrieved July 1, 2022 from https://www.novelbits.io/ble-connection-intervals/

[12] Adriano Branco, Luca Mottola, Muhammad Hamad Alizai, and Junaid Haroon Siddiqui. 2019. Intermittent asynchronous peripheral operations. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems (SenSys'19)*. Association for Computing Machinery, New York, NY, 55–67. https://doi.org/10.1145/3356250.3360033

[13] Ricardo C. Carrano, Diego Passos, Luiz C. S. Magalhaes, and Celio V. N. Albuquerque. 2014. Survey and taxonomy of duty cycling mechanisms in wireless sensor networks. *IEEE Commun. Surv. Tutor.* 16, 1 (2014), 181–194. https://doi.org/10.1109/SURV.2013.052213.00116

[14] 2011. SURFACE MOUNT ULTRA LOW LEAKAGE SILICON SWITCHING DIODE. CENTRAL SEMICONDUCTOR CORP. Retrieved from https://www.ablic.com/en/doc/datasheet/voltage_regulator/S1313_E.pdf

[15] Li Chen, Jeremy Warner, Wendi Heinzelman, and Ilker Demirkol. 2015. MH-REACH-mote: Supporting multi-hop passive radio wake-up for wireless sensor networks. In *Proceedings of the IEEE International Conference on Communications (ICC'15)*. 6512–6518. https://doi.org/10.1109/ICC.2015.7249362

[16] Alexei Colin and Brandon Lucia. 2016. Chain: Tasks and channels for reliable intermittent programs. *SIGPLAN Not.* 51, 10 (Oct. 2016), 514–530. https://doi.org/10.1145/3022671.2983995

[17] Jasper de Winkel, Carlo Delle Donne, Kasim Sinan Yildirim, Przemysław Pawełczak, and Josiah Hester. 2020. Reliable timekeeping for intermittent computing. In *Proceedings of the 25th International Conference on Architectural Support for Programming Languages and Operating Systems*. 53–67.

[18] Jasper de Winkel, Haozhe Tang, and Przemysław Pawełczak. 2022. Intermittently-powered bluetooth that works. In *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services (MobiSys'22)*. Association for Computing Machinery, New York, NY, 287–301. https://doi.org/10.1145/3498361.3538934

[19] Falko Dressler, Bastian Bloessl, Martin Hierold, Chia-Yu Hsieh, Thorsten Nowak, Robert Weigel, and Alexander Koelpin. 2015. Protocol design for ultra-low power wake-up systems for tracking bats in the wild. In *Proceedings of the IEEE International Conference on Communications (ICC'15)*. 6345–6350. https://doi.org/10.1109/ICC.2015.7249335

[20] Digi-Key Electronics. 2020. AM-5904CAR: Amourphous Solar Cell 49.5mW 7.7V. Retrieved September 1, 2020 from https://www.digikey.com/en/products/detail/panasonic-bsg/AM-5904CAR-DGK-T/2165197

[21] Zhi Ang Eu, Hwee-Pink Tan, and Winston K. G. Seah. 2011. Design and performance analysis of MAC schemes for wireless sensor networks powered by ambient energy harvesting. *Ad Hoc Netw.* 9, 3 (May 2011), 300–323. https://doi.org/10.1016/j.adhoc.2010.07.014

[22] Nick Flaherty. 2020. Energy Harvesting Enables Batteryless Smart Farming. Retrieved from https://www.eenewspower.com/news/energy-harvesting-enables-batteryless-smart-farming

[23] G. U. Gamm, M. Sippel, M. Kostic, and L. M. Reindl. 2010. Low power wake-up receiver for wireless sensor nodes. In *Proceedings of the 6th International Conference on Intelligent Sensors, Sensor Networks and Information Processing*. 121–126. https://doi.org/10.1109/ISSNIP.2010.5706778

[24] G. U. Gamm, S. Stoecklin, and L. M. Reindl. 2014. Wake-up receiver operating at 433 MHz. In *Proceedings of the IEEE 11th International Multi-Conference on Systems, Signals Devices (SSD'14)*. 1–4. https://doi.org/10.1109/SSD.2014.6808754

[25] Kai Geissdoerfer and Marco Zimmerling. 2021. 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'21)*.

[26] Kai Geissdoerfer and Marco Zimmerling. 2022. Learning to communicate effectively between battery-free devices. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI'22)*. USENIX Association, Berkeley, CA, 419–435.

[27] Debasish Ghose and Frank Y. Li. 2017. Enabling backoff for scm wake-up radio: Protocol and modeling. *IEEE Commun. Lett.* 21, 5 (2017), 1031–1034. https://doi.org/10.1109/LCOMM.2017.2653779

[28] Lin Gu and J.A. Stankovic. 2004. Radio-triggered wake-up capability for sensor networks. In *Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'04)*. 27–36. https://doi.org/10.1109/RTTAS.2004.1317246

[29] Lakshmikanth Guntupalli, Debasish Ghose, Frank Y. Li, and Mikael Gidlund. 2018. Energy efficient consecutive packet transmissions in receiver-initiated wake-up radio enabled WSNs. *IEEE Sens. J.* 18, 11 (2018), 4733–4745. https://doi.org/10.1109/JSEN.2018.2825540

[30] Josiah Hester, Timothy Scott, and Jacob Sorber. 2014. Ekho: Realistic and repeatable experimentation for tiny energy-harvesting sensors. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems (SenSys'14)*. ACM, New York, NY, 1–15.

[31] Josiah Hester, Lanny Sitanayah, and Jacob Sorber. 2015. Tragedy of the coulombs: Federating energy storage for tiny, intermittently-powered sensors. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems (SenSys'15)*. ACM, New York, NY, 5–16. https://doi.org/10.1145/2809695.2809707

[32] Josiah Hester and Jacob Sorber. 2017. Flicker: Rapid prototyping for the batteryless internet-of-things. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems (SenSys'17)*. Association for Computing Machinery, New York, NY, Article 19, 13 pages. https://doi.org/10.1145/3131672.3131674

[33] Josiah Hester and Jacob Sorber. 2017. The future of sensing is batteryless, intermittent, and awesome. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems (SenSys'17)*. ACM, New York, NY, Article 21, 6 pages. https://doi.org/10.1145/3131672.3131699

[34] Josiah Hester and Jacob Sorber. 2019. Batteries not included. *XRDS* 26, 1 (Sep. 2019), 23–27. https://doi.org/10.1145/3351474

[35] Josiah Hester, Kevin Storer, and Jacob Sorber. 2017. Timely execution on intermittently powered batteryless sensors. In *Proceedings of the 15th ACM Conference on Embedded Networked Sensor Systems (SenSys'17)*. ACM, New York, NY, USA.

[36] Josiah Hester, Nicole Tobias, Amir Rahmati, Lanny Sitanayah, Daniel Holcomb, Kevin Fu, Wayne P. Burleson, and Jacob Sorber. 2016. Persistent clocks for batteryless sensing devices. *ACM Trans. Embed. Comput. Syst.* 15, 4 (2016).

[37] Matthew Hicks. 2017. Clank: Architectural support for intermittent computation. In *Proceedings of the 44th Annual International Symposium on Computer Architecture (ISCA'17)*. Association for Computing Machinery, New York, NY, 228–240. https://doi.org/10.1145/3079856.3080238

[38] Pei Huang, Li Xiao, Soroor Soltani, Matt W. Mutka, and Ning Xi. 2013. The evolution of mac protocols in wireless sensor networks: A survey. *IEEE Commun. Surv. Tutor.* 15, 1 (2013), 101–120. https://doi.org/10.1109/SURV.2012.040412.00105

[39] Jaeho Im, Jacob Breiholz, Shuo Li, Benton Calhoun, and David D. Wenzloff. 2020. A fully integrated 0.2V 802.11ba wake-up receiver with -91.5dBm sensitivity. In *Proceedings of the IEEE Radio Frequency Integrated Circuits Symposium (RFIC'20)*. 339–342. https://doi.org/10.1109/RFIC49505.2020.9218383

[40] Microchip Technology Inc. 2017. AMIC841 Comparator with 1.25% Reference and Adjustable Hysteresis. Retrieved October 22, 2020 from http://ww1.microchip.com/downloads/en/DeviceDoc/20005758A.pdf

[41] Texas Instruments. 2013. CC1101: Low-Power Sub-1 GHz RF Transceiver. Retrieved from http://www.ti.com/lit/ds/symlink/cc1101.pdf

[42] Texas Instruments. 2020. MSP430FR5994 LaunchPad Development Kit. Retrieved from https://www.ti.com/tool/MSP-EXP430FR5994

[43] Vana Jelicic, Michele Magno, Davide Brunelli, Vedran Bilas, and Luca Benini. 2012. Analytic comparison of wake-up receivers for WSNs and benefits over the wake-on radio scheme. In *Proceedings of the 7th ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks (PM2HW2N'12)*. Association for Computing Machinery, New York, NY, 99–106. https://doi.org/10.1145/2387191.2387206

[44] Heikki Karvonen, Juha Petäjäjärvi, Jari Iinatti, Matti Hämäläinen, and Carlos Pomalaza-Ráez. 2014. A generic wake-up radio based MAC protocol for energy efficient short range communication. In *Proceedings of the IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC'14)*. 2173–2177. https://doi.org/10.1109/PIMRC.2014.7136533

[45] Issa Khalil, Saurabh Bagchi, and Ness B. Shroff. 2007. SLAM: Sleep-wake aware local monitoring in sensor networks. In *Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07)*. 565–574. https://doi.org/10.1109/DSN.2007.88

[46] Vito Kortbeek, Kasim Sinan Yildirim, Abu Bakar, Jacob Sorber, Josiah Hester, and Przemysław Pawełczak. 2020. Time-sensitive intermittent computing meets legacy software. In *Proceedings of the 25th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'20)*. Association for Computing Machinery, New York, NY, 85–99. https://doi.org/10.1145/3373376.3378476

[47] Trong Nhan Le, Michele Magno, Alain Pegatoquet, Olivier Berder, Olivier Sentieys, and Emanuel Popovici. 2013. Ultra low power asynchronous MAC protocol using wake-up radio for energy neutral WSN. In *Proceedings of the 1st International Workshop on Energy Neutral Sensing Systems (ENSSys'13)*. Association for Computing Machinery, New York, NY, Article 10, 6 pages. https://doi.org/10.1145/2534208.2534221

[48] Trong Nhan Le, Alain Pegatoquet, Olivier Berder, and Olivier Sentieys. 2015. Energy-efficient power manager and MAC protocol for multi-hop wireless sensor networks powered by periodic energy harvesting sources. *IEEE Sens. J.* 15, 12 (2015), 7208–7220.

[49] Trong Nhan Le, Alain Pegatoquet, and Michele Magno. 2015. Asynchronous on demand MAC protocol using wake-up radio in wireless body area network. In *Proceedings of the 6th International Workshop on Advances in Sensors and Interfaces (IWASI'15)*. 228–233. https://doi.org/10.1109/IWASI.2015.7184942

[50] Jean Lebreton and Nour Murad. 2015. Implementation of a wake-up radio cross-layer protocol in OMNeT++/MiXiM. In *Proceedings of the 2nd OMNeT++ Community Summit (September 2015)*. Retrieved from https://summit.omnetpp.org/2015/assets/pdf/OMNET-2015-14-SP.pdf

[51] Seulki Lee, Bashima Islam, Yubo Luo, and Shahriar Nirjon. 2019. Intermittent learning: On-device machine learning on intermittently powered system. *Proc. ACM Interact. Mob. Wear. Ubiq. Technol.* 3, 4, Article 141 (Dec. 2019), 30 pages. https://doi.org/10.1145/3369837

[52] E.-Y. A. Lin, J. M. Rabaey, S. Wiethoelter, and A. Wolisz. 2005. Receiver initiated rendezvous schemes for sensor networks. In *IEEE Global Telecommunications Conference (GLOBECOM'05)*. Vol. 5. https://doi.org/10.1109/GLOCOM.2005.1578331

[53] Linx Technologies Inc. 2014. *ANT-433-CW-QW Data Sheet*. Linx Technologies Inc.

[54] Edward Longman, Oktay Cetinkaya, Mohammed El-Hajjar, and Geoff V. Merrett. 2021. Wake-up radio-enabled intermittently-powered devices for mesh networking: A power analysis. In *Proceedings of the IEEE 18th Annual Consumer Communications & Networking Conference (CCNC'21)*. 1–6. https://doi.org/10.1109/CCNC49032.2021.9369557

[55] Edward Longman, Oktay Cetinkaya, Mohammed El-Hajjar, and Geoff V. Merrett. 2022. Mesh networking for intermittently powered devices: Architecture and challenges. *Netwrk. Mag. Global Internetwkg.* 36, 3 (may 2022), 122–128. https://doi.org/10.1109/MNET.105.2000782

[56] Elena Lopez-Aguilera, Ilker Demirkol, Eduard Garcia-Villegas, and Josep Paradells. 2019. IEEE 802.11-enabled wake-up radio: Use cases and applications. *Sensors* 20 (12 2019), 66. https://doi.org/10.3390/s20010066

[57] Brandon Lucia, Vignesh Balaji, Alexei Colin, Kiwan Maeng, and Emily Ruppel. 2017. Intermittent computing: Challenges and opportunities. In *Proceedings of the 2nd Summit on Advances in Programming Languages (SNAPL'17), Leibniz International Proceedings in Informatics (LIPIcs), Vol. 71*, Benjamin S. Lerner, Rastislav Bodík, and Shriram Krishnamurthi (Eds.). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 8:1–8:14. https://doi.org/10.4230/LIPIcs.SNAPL.2017.8

[58] Kiwan Maeng and Brandon Lucia. 2018. Adaptive dynamic checkpointing for safe efficient intermittent computing. In *Proceedings of the 13th USENIX Conference on Operating Systems Design and Implementation (OSDI'18)*. USENIX Association, Berkeley, CA, 129–144.

[59] Michele Magno and Luca Benini. 2014. An ultra low power high sensitivity wake-up radio receiver with addressing capability. In *Proceedings of the IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob'14)*. 92–99. https://doi.org/10.1109/WiMOB.2014.6962155

[60] Michele Magno, Vana Jelicic, Bruno Srbinovski, Vedran Bilas, Emanuel Popovici, and Luca Benini. 2016. Design, implementation, and performance evaluation of a flexible low-latency nanowatt wake-up radio receiver. *IEEE Trans. Industr. Inf.* 12, 2 (2016), 633–644. https://doi.org/10.1109/TII.2016.2524982

[61] S. Mahlknecht and M. Spinola Durante. 2009. WUR-MAC: Energy efficient wakeup receiver based MAC protocol. *IFAC Proc. Vol.* 42, 3 (2009), 79–83. https://doi.org/10.3182/20090520-3-KR-3006.00012

[62] K. Martinez, P. Padhy, A. Elsaify, G. Zou, A. Riddoch, J. K. Hart, and H. L. R. Ong. 2006. Deploying a sensor network in an extreme environment. In *Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'06)*, Vol. 1. 8 pp.–. https://doi.org/10.1109/SUTC.2006.1636175

[63] Nafiseh Seyed Mazloum and Ove Edfors. 2014. Performance analysis and energy optimization of wake-up receiver schemes for wireless low-power applications. *IEEE Trans. Wireless Commun.* 13, 12 (2014), 7050–7061. https://doi.org/10.1109/TWC.2014.2334658

[64] M. J. Miller and N. H. Vaidya. 2005. A MAC protocol to reduce sensor network energy consumption using a wakeup radio. *IEEE Trans. Mobile Comput.* 4, 3 (2005), 228–242. https://doi.org/10.1109/TMC.2005.31

[65] Masateru MINAMI, Takashi MORITO, Hiroyuki Morikawa, and Tomonori Aoyama. 2005. Solar Biscuit: A Battery-less Wireless Sensor Network System for Environmental Monitoring Applications. *INSS (June 2005)*. Retrieved from https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=fa24833315cbaef543ac2180b8261c9b5ec103f2

[66] Yong Niu, Liren Yu, Yong Li, Zhangdui Zhong, and Bo Ai. 2019. Device-to-device communications enabled multicast scheduling for mmwave small cells using multi-level codebooks. *IEEE Trans. Vehic. Technol.* 68, 3 (2019), 2724–2738. https://doi.org/10.1109/TVT.2018.2883641

[67] Tamoghna Ojha, Sudip Misra, and Narendra Singh Raghuwanshi. 2015. Wireless sensor networks for agriculture: The state-of-the-art in practice and future challenges. *Comput. Electr. Agric.* 118 (2015), 66–84. https://doi.org/10.1016/j.compag.2015.08.011

[68] Joaquim Oller, Ilker Demirkol, Jordi Casademont, Josep Paradells, Gerd Ulrich Gamm, and Leonhard Reindl. 2016. Has time come to switch from duty-cycled mac protocols to wake-up radio for wireless sensor networks? *IEEE/ACM Trans. Netw.* 24, 2 (Apr. 2016), 674–687. https://doi.org/10.1109/TNET.2014.2387314

[69] Mario Paoli, Dora Spenza, Chiara Petrioli, Michele Magno, and Luca Benini. 2016. MagoNode++: A wake-up-radio-enabled wireless sensor mote for energy-neutral applications: Poster abstract. In *Proceedings of the 15th International Conference on Information Processing in Sensor Networks (IPSN'16)*. IEEE Press, Article 57, 2 pages.

[70] Alain Pegatoquet, Trong Nhan Le, and Michele Magno. 2019. A wake-up radio-based mac protocol for autonomous wireless sensor networks. *IEEE/ACM Trans. Netw.* 27, 1 (Feb. 2019), 56–70. https://doi.org/10.1109/TNET.2018.2880797

[71] Chiara Petrioli, Dora Spenza, Pasquale Tommasino, and Alessandro Trifiletti. 2014. A novel wake-up receiver with addressing capability for wireless sensor nodes. In *Proceedings of the IEEE International Conference on Distributed Computing in Sensor Systems*. 18–25. https://doi.org/10.1109/DCOSS.2014.9

[72] Rajeev Piyare, Timofei Istomin, and Amy L. Murphy. 2017. WaCo: A wake-up radio COOJA extension for simulating ultra low power radios. In *Proceedings of the International Conference on Embedded Wireless Systems and Networks (EWSN'17)*. 48–53.

[73] Rajeev Piyare, Amy L. Murphy, Csaba Kiraly, Pietro Tosato, and Davide Brunelli. 2017. Ultra low power wake-up radios: A hardware and networking survey. *IEEE Commun. Surv. Tutor.* 19, 4 (2017), 2117–2157. https://doi.org/10.1109/COMST.2017.2728092

[74] PJRC. 2022. Teensy 3.6 Development Board. Retrieved from https://www.pjrc.com/store/teensy36.html
[75] Amir Rahmati, Mastooreh Salajegheh, Dan Holcomb, Jacob Sorber, Wayne P. Burleson, and Kevin Fu. 2012. TARDIS: Time and remanence decay in SRAM to implement secure protocols on embedded devices without clocks. In *Proceedings of the 21st USENIX conference on Security symposium*. USENIX Association, 36–36.
[76] Benjamin Ransford, Jacob Sorber, and Kevin Fu. 2011. Mementos: System support for long-running computation on rfid-scale devices. In *Proceedings of the 16th International Conference Architectural Support for Programming Languages and Operating Systems (ASPLOS'11)*. ACM, New York, NY, 159–170.
[77] Faisal Karim Shaikh and Sherali Zeadally. 2016. Energy harvesting in wireless sensor networks: A comprehensive review. *Renew. Sust. Energy Rev.* 55 (2016), 1041–1054. https://doi.org/10.1016/j.rser.2015.11.010
[78] Lukas Sigrist, Rehan Ahmed, Andres Gomez, and Lothar Thiele. 2020. Harvesting-aware optimal communication scheme for infrastructure-less sensing. *ACM Trans. Internet Things* 1, 4, Article 22 (Jun. 2020), 26 pages. https://doi.org/10.1145/3395928
[79] Ripudaman Singh and Biplab Sikdar. 2020. A receiver initiated low delay MAC protocol for wake-up radio enabled wireless sensor networks. *IEEE Sens. J.* 20, 22 (2020), 13796–13807. https://doi.org/10.1109/JSEN.2020.3003929
[80] Jacob Sorber, Alexander Kostadinov, Matthew Garber, Matthew Brennan, Mark D. Corner, and Emery D. Berger. 2007. Eon: A language and runtime system for perpetual systems. In *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems (SenSys'07)*. Association for Computing Machinery, New York, NY, 161–174. https://doi.org/10.1145/1322263.1322279
[81] Sparkfun. 2020. SparkFun Soil Moisture Sensor SEN-13322. Retrieved from https://www.sparkfun.com/products/13322
[82] Yanjun Sun, Omer Gurewitz, and David B. Johnson. 2008. RI-MAC: A receiver-initiated asynchronous duty cycle mac protocol for dynamic traffic loads in wireless sensor networks. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys'08)*. Association for Computing Machinery, New York, NY, 1–14. https://doi.org/10.1145/1460412.1460414
[83] Osram Sylvania. 2022. H6054STBX Headlight Bulb, Part Detail Information. Retrieved from https://www.autoecat.com/pies/0002774999OSRAMH6054STBX.html
[84] Massachusetts Institute of Technology. 2020. Breaking Free From Batteries: Industrial "Internet of Things" Platform Built on Battery-Free Sensors. Retreved from https://scitechdaily.com/breaking-free-from-batteries-industrial-internet-of-things-platform-built-on-battery-free-sensors/
[85] Texas Instrument 2021. *MSP430FR599x, MSP430FR596x Mixed-Signal Microcontrollers datasheet (Rev. D)*. Texas Instrument. Rev. D.
[86] Texas Instrument 2021. *MSP430 FRAM Technology—How To and Best Practices (Rev. B)*. Texas Instrument. Rev. B.
[87] Divyansh Thakur, Yugal Kumar, Arvind Kumar, and Pradeep Kumar Singh. 2019. Applicability of wireless sensor networks in precision agriculture: A review. *Wireless Pers. Commun.* 107, 1 (2019), 471–512. https://doi.org/10.1007/s11277-019-06285-2
[88] Ricklef Wohlers, Vladimir Dyo, Stephen A. Ellwood, David W. Macdonald, Andrew Markham, Cecilia Mascolo, Bence Pasztor, Salvatore Scellato, Niki Trigoni, and Kharsim Yousef. 2012. WILDSENSING: Design and deployment of a sustainable sensor network for wildlife monitoring. *ACM Trans. Sens. Netw.* (2012).
[89] Nguyen Van Huynh, Dinh Thai Hoang, Xiao Lu, Dusit Niyato, Ping Wang, and Dong In Kim. 2018. Ambient backscatter communications: A contemporary survey. *IEEE Commun. Surv. Tutor.* 20, 4 (2018), 2889–2922. https://doi.org/10.1109/COMST.2018.2841964
[90] Bang Wang. 2011. Coverage problems in sensor networks: A survey. *ACM Comput. Surv.* 43, 4, Article 32 (Oct. 2011), 53 pages. https://doi.org/10.1145/1978802.1978811
[91] Geoffrey Werner-Allen, Konrad Lorincz, Matt Welsh, Omar Marcillo, Jeff Johnson, Mario Ruiz, and Jonathan Lees. 2006. Deploying a wireless sensor network on an active volcano. *IEEE Internet Comput.* 10, 2 (Mar. 2006), 18–25. https://doi.org/10.1109/MIC.2006.26
[92] Pei Zhang, Christopher M. Sadler, Stephen A. Lyon, and Margaret Martonosi. 2004. Hardware design experiences in ZebraNet. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*. ACM, 227–238.

Received 23 January 2023; revised 28 September 2023; accepted 19 February 2024