E-Cube: Towards a First Benchmarking Facility for Battery-Free Systems

Markus Schuß
markus.schuss@tugraz.at
Graz University of Technology
Graz, Austria

Carlo Alberto Boano
cboano@tugraz.at
Graz University of Technology
Graz, Austria

ABSTRACT

IoT devices are commonly powered by batteries: even rechargeable ones wear out and must be replaced. Hence, in a future with billions of IoT devices, the disposal of their batteries represents a looming environmental disaster. *Battery-free* systems have the potential to address this key sustainability issue: by relying on energy harvested from ambient sources, IoT devices could, in theory, operate in perpetuity, require zero maintenance, and produce less waste. However, even though research on battery-free systems has bloomed in recent years, the community still lacks public testbeds and a well-defined yardstick to benchmark the performance of various solutions. As a result, battery-free solutions have rarely been compared under the same conditions, which hinders a comprehensive understanding of the best-performing approach in specific settings, hampering industrial adoption. To fill this gap, we move our first steps towards the design of *E-Cube*: the first fully-automated, open, and low-cost benchmarking facility for battery-free IoT systems. We present E-Cube's design and architecture, showing how it can be used to facilitate a *competition* evaluating the performance of solutions running on devices powered by intermittent sources of energy.

CCS CONCEPTS

• **Computer systems organization → Embedded systems**.

KEYWORDS

Benchmark, Energy harvesting, E-Cube, Intermittent computing, Internet of Things, Sensor nodes, Testbed infrastructure, Wireless.

ACM Reference Format:
Markus Schuß and Carlo Alberto Boano. 2024. E-Cube: Towards a First Benchmarking Facility for Battery-Free Systems. In *International Conference on Information Technology for Social Good (GoodIT '24), September 04–06, 2024, Bremen, Germany*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3677525.3678688

1 INTRODUCTION

Up to 30 billion connected devices will be deployed by 2030, many of which powered by batteries [29]. Replacing and disposing batteries at this volume is impractical and expensive; moreover, modern battery cells include a wide range of toxic chemicals including heavy metals [18], which represents a key sustainability issue [4].

GoodIT '24, September 04–06, 2024, Bremen, Germany

ACM ISBN 979-8-4007-1094-0/24/09
https://doi.org/10.1145/3677525.3678688

To alleviate this problem, the development of *battery-free* systems has gained traction [2]. While, in theory, such systems are powered perpetually by energy harvested from ambient sources such as light, temperature, and vibration, the intermittent nature of such power sources means that devices rely exclusively on virtually maintenance-free capacitors for energy storage [11, 22]. Battery-free systems are typically designed using two approaches: (i) using large capacitors to buffer energy to let the system operate continuously over a longer period of time, or (ii) accepting the intermittently-powered state of the device by relying solely on a small capacitor and operating only when energy is available. While the former seems to be the obvious path forward (as existing applications can be migrated seemingly with little effort), the low energy density of capacitors compared to conventional batteries means that such devices will often be larger than their battery-equipped counterpart. The alternative solution of only operating when power is plenty has drawbacks in terms of responsiveness, as it cannot be guaranteed that an external event is correctly detected should the device not be operational at the time in which the event occurs.

Problem statement. Several solutions have been proposed to improve the performance and usability of battery-free systems [9, 28]. However, due to the lack of a unified testing methodology, an agreed-upon hardware platform, and readily-available tools, such solutions have rarely been compared under the same conditions. Even worse, device-to-device differences in key parts such as capacitors can cause as much as 48% variation in system lifetime [7], necessitating experiments to be run on the exact same hardware (HW) to be truly comparable. To enable the quantitative comparison of solutions under the same conditions (i.e., to facilitate *benchmarking*), one requires an open testbed providing a low barrier of entry.

Lack of an open and low-cost testbed. While some testbeds have been set up by research groups [12], their often closed nature and the relative complexity of their specialized feature set hinder their use for quantitative comparisons. As shown in other fields, e.g., in the context of low-power wireless (LPW) systems, open testbeds [1, 20, 24, 31] are invaluable not only for development, but also for comparing approaches. Some tools [15, 23] tailored to battery-free systems have been made available as open-source software and HW, but, alone, they only solve a small part of the benchmarking problem – just the recording and replaying of energy traces.

Lack of automation. A benchmark for battery-free systems requires more than readily-available HW: it needs an agreed-upon methodology. However, this entails the specification of an application (i.e., the task that needs to be implemented) and performance metrics (to determine how well the solution performs). The benchmarking facility should have the ability to configure its HW (e.g., connected peripherals such as sensors, memory, or actuators) and software

(e.g., the scripts computing the metrics) as a function of the provided specifications dynamically without manual intervention. Moreover, the nature and amount of energy available must be accurately and repeatably replayed from a recorded trace to evaluate the performance of solutions for different real-world conditions.

Our contributions. We present E-Cube, an open benchmark facility encompassing (i) a testbed infrastructure and (ii) a fully-automated workflow easily accessible via a public web interface. Built on top of D-Cube [13], an open-source benchmark that we created to evaluate the dependability of LPW protocols [26, 27], E-Cube is a solution tailored to evaluate the performance of battery-free systems. Similar to our previous effort leveraging D-Cube to organize a series of competitions in which we benchmarked the reliability and efficiency of LPW protocols [6], our goal is now to evaluate the performance of battery-free solutions, so to understand which ones meet best the requirements of a given IoT application. After describing E-Cube's low-cost testbed infrastructure, we implement a proof-of-concept benchmark using E-Cube to automatically evaluate the performance of a given solution. A user just needs to upload a binary firmware via an intuitive web interface, and can seamlessly obtain both the raw measurements and the derived performance metrics. We argue that E-Cube's high degree of automation will facilitate the creation of a competition to compare the performance of battery-free solutions under the same settings.

2 RELATED WORK

Until now, battery-free systems have been developed and tested on a wide range of HW platforms with the aid of various purpose-built tools; however, no benchmark for battery-free systems exists yet. We argue that efforts similar to those recently undertaken in the LPW networking field [5, 27] need to be pursued to tackle this issue.

Testing LPW systems. In order to test the performance of various LPW protocols, several open testbeds [1, 20, 31] have provided the means to compare a plethora of solutions running on the same experimental setup. However, using the same testbed does not imply that multiple test runs are conducted in the same way: this requires an agreed-upon methodology (e.g., users may select different traffic loads or extract performance metrics differently [5, 17]).

Benchmarking LPW systems. To bring in a common methodology, we have created D-Cube, the first benchmark [27] for LPW systems featuring a fully-automated testbed infrastructure. Now publicly available [13], D-Cube has been the foundation for running the EWSN dependability competition series [6], which provided – for the first time – a level playing field for researchers and practitioners developing dependable LPW protocols. The benefit of such competitions is that experts in the field are actively motivated in pushing the performance of their solutions to the limit: this can trigger major enhancements to existing approaches [16] or even brand new ideas [19, 21]. In this regard, D-Cube's ability to measure key performance metrics in HW (e.g., power consumption, reliability, and latency) by accurately detecting and time-stamping GPIO events using network-wide time synchronisation played a crucial role in guaranteeing the objectivity of the results [26]. D-Cube also featured an asynchronous I2C mailbox with an interrupt line (to enable communication between the testbed infrastructure and the target node at run-time), as well as the ability of building and applying patches to binary files [24, 27]. These features allow not only to automatically change traffic patterns and loads, node identities, as well as message payloads, but to also change user-defined protocol parameters on a per-node and per-experiment basis. Finally, D-Cube also allowed to automatically return a performance report containing raw measurements collected in HW, custom performance metrics, and the relative performance compared to other solutions for specific settings (i.e., a leaderboard). Unfortunately, D-Cube cannot be directly used to benchmark battery-free systems, as it is unable to control the supply voltage (currently, voltage is being monitored only) and as it relies on runtime communication via the I2C mailbox (which leaks current and hence biases results).

Testing battery-free systems. A few tools for testing battery-free and intermittent computing systems exist. One of the most advanced is Shepherd [12], a testbed allowing to record energy traces in the field and to replay them using the same HW. Multiple nodes can synchronize their operation to also run experiments with networked systems. Unfortunately – to the best of our knowledge – no instance of Shepherd is currently open to the public or intended for any automated form of benchmarking. Shepherd's design is centered around custom add-ons that have to be manually assembled for each experiment. Moreover, Shepherd does not provide a user interface and expects users to manually orchestrate experiments by running scripts on one or more BeagleBone single-board computers (SBC). Another popular tool for battery-free systems is Ekho [15], which provides a USB device capable of logging or replaying an energy trace. While this seems to be a perfect solution to be utilized as part of a benchmarking infrastructure, the device is built on older and hard-to-get components. Moreover, for benchmarking purposes, the ability to capture energy traces using the same HW is not necessary: thus, in principle, one can opt for a simpler design.

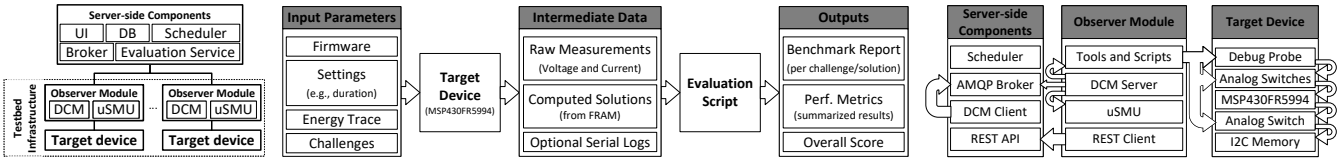
In summary, existing tools for battery-free systems are insufficient to benchmark performance. In this paper, we fill this gap by conducting a similar effort to D-Cube, but tailored to battery-free systems.

3 REQUIREMENTS & CHALLENGES

We discuss next the requirements of a benchmarking facility for battery-free systems, together with the associated challenges that need to be tackled to come up with a concrete design instance.

R1: Low-cost. To evaluate the performance of battery-free systems, a source measurement unit (SMU) is necessary to provide a voltage curve while concurrently measuring the current consumption of the system. Unfortunately, such SMUs – while very accurate – can cost up to 9000€ for a single device. Tools such as Ekho and testbeds such as Shepherd are cheaper, but their cost is still relatively high, as they are meant to not only emulate a power source by replaying an energy trace, but also to record such traces in the field. A key challenge is hence to find a device with the versatility and accuracy akin to an SMU, but at a significantly lower cost – without adding costly features that are not needed for a benchmarking use case.

R2: Repeatability. While a low-cost solution facilitates replication across multiple sites, this must not come at the price of lower repeatability. For example, sampling the current consumption at a high speed is desirable for the emulation of a feedback system, but is secondary to the accurate replaying of voltage traces. Hence, the chosen tool replaying energy traces should offer an accurate control of the supplied voltage, and not rely on any proprietary or complex format of the traces to simplify reuse on different HW.



(a) Architecture (b) Workflow of an experiment (c) Components and their relationships

Figure 1: E-Cube's high-level architecture split between the server-side components and multiple observer modules (a), workflow for the execution and evaluation of an experiment (b), and the connectivity between components during an experiment (c).

R3: Generality and HW-agnostic design. The benchmarking facility should be able to emulate any energy source (e.g., kinetic, RF, solar, temperature) and support any target platform (especially popular ones in the community, such as the MSP430FR series). Moreover, the facility should be able to support and remotely configure any application-specific peripheral (e.g., sensors, memory, or actuators) without need for manual intervention.

R4: Automation. To enable an objective comparison of different solutions, the benchmarking facility should use a unified testing methodology. That is, one should be able to specify a concrete application (e.g., the task performed by the system, the employed HW, and the supplied energy trace), as well as performance metric(s). Once the benchmark is defined, the facility should be able to automatically set up a test run and all involved HW (to minimize unwanted mistakes and differences in the setup), as well as autonomously compute the defined performance metric(s). That is, the defined benchmark should be executed on the testbed facility with a high degree of automation w.r.t. both setup and evaluation.

R5: Public availability and exposure. The creation of a benchmark facility tackling the previous challenges does not guarantee a wide adoption. Merely using the facility to re-run existing solutions from prior publications is also not an option, as many solutions need to be adapted to both the benchmarked application and employed HW, which requires expert knowledge. As learned from our past D-Cube effort, it is important to make the facility open to the public and to also organize initiatives (e.g., competitions) involving leading researchers and practitioners from academia and industry.

4 E-CUBE: DESIGN

We present next the design of E-Cube, a benchmarking facility for battery-free systems that meets the requirements listed in Sect. 3. Before explaining E-Cube's architecture (Sect. 4.3), we define a sample benchmark to outline the role of each component (Sect. 4.1), and discuss how we base the facility's design on D-Cube (Sect. 4.2).

4.1 Benchmark Example: Hashcash

While E-Cube is designed to support a wide range of applications and to be agnostic to the energy supply and HW platform, we use a simple benchmark as running example to explain its inner working. Specifically, we define a benchmark to assess the computational overhead of intermittent computing solutions. We leverage *hashcash* [3], a proof-of-work algorithm to prove that a device (an MSP-EXP430FR5994 Launchpad) has solved a computationally-expensive task given a configurable difficulty – which we refer to as an hashcash *challenge*. The challenge's difficulty is configurable to evaluate the ability of the solution to deal with different task lengths despite interruptions in the power supply. To support this benchmark, E-Cube needs to be able to interact with the target device to provide it with the challenges to be solved, as well as to record any solutions provided by the device. In general, there are two ways to tackle this problem: (i) at runtime via a shared interface with the device such as UART or I2C, or (ii) before and after the execution of the experiment. For example, D-Cube utilizes the first approach via a shared mailbox. However, any connection between the target device and testbed infrastructure may result in a small leakage current. We hence design E-Cube following the second approach. As shown in Fig. 1b, each experiment involves a set of input parameters such as the firmware, energy trace, as well as the number and difficulty of the challenges. Before an experiment, E-Cube stores a configurable amount of challenges of varying difficulty into FRAM, and hands over exclusive control to the target device during an experiment. During execution, E-Cube collects the energy trace's set voltage and measured current. After the experiment, E-Cube automatically retrieves the solutions computed by the target device from FRAM along with optionally-enabled serial logs from the target device (i.e., any `printf` output). Moreover, after the completion of the experiment, a script on the server (run by the evaluation service) takes the collected data and automatically verifies all solutions to determine if they are correct. It then creates a human-readable PDF report breaking down each solution and matching challenge, as well as highlighting any incorrect solution. In addition, a simpler breakdown of the number of correct and incorrect challenges per difficulty is provided and, finally, all correct and incorrect solutions are weighted with their difficulty to provide a single overall score that can be used for ranking the performance.

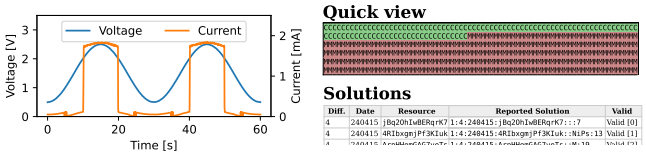
4.2 From D-Cube to E-Cube

E-Cube is built on top of the open-source D-Cube [25], which allows us to re-purpose its automation features while adapting the benchmarking process to the requirements of battery-free systems. While D-Cube is built using custom HW facilitating the measurement of the energy consumed by a device, it cannot change the supply voltage to replay energy traces. As such, one still needs to find a low-cost SMU-like device to reproduce energy traces.

By design, in D-Cube [27] it is relatively simple to replace individual steps of a benchmark. We hence adapt D-Cube's scheduler and evaluation scripts to support the type of battery-free benchmarks E-Cube is designed for. However, the functionality responsible for executing the experiment in HW had to be rebuilt mostly from the ground up. The latter encompasses changes such as a completely new energy supply and profiling unit built on top of the existing open-source μSMU [30], as well as the switch from D-Cube's mailbox system to an exclusive-access I2C FRAM memory.

4.3 Architecture

Fig. 1a shows the architecture of E-Cube split into (i) *testbed infrastructure* and (ii) *server-side components*. The former consists of



(a) Energy Trace & Measurement (b) Benchmark Report

Figure 2: Example outputs of E-Cube for the hashcash benchmark. (a) Replayed voltage curve from an energy trace and measured current consumption of the target device. (b) Human-readable PDF report (cropped).

observer modules which execute the experiment on a target device. The latter are hosted on a central server and handle the interaction with the user, as well as the communication with the observers.

Testbed infrastructure. Each observer module consists of an off-the-shelf Raspberry Pi (RPI) SBC, an open-source μSMU [30], and a few analog switches. To ensure that E-Cube can be used with a wide variety of target devices, we rely on USB for programming. Most development boards already come with an onboard debug probe, including the MSP-EXP430FR5994 Lauchpad used as target devices. Jumpers on the device allow to electrically disconnect parts such as the debug probe. Instead of manually opening and closing jumpers, E-Cube uses analog switches directly controlled by the GPIOs of the RPI. For example, one can connect the debug probe only during the programming phase, while disconnecting it during the execution of the experiment to avoid leakage. This would otherwise cause significant errors in the replay of energy traces, and it is the reason why E-Cube – unlike D-Cube – forgoes runtime communication.

Server-side components. E-Cube's server-side components consist of: (i) a user interface (UI), (ii) two databases (DB) to store the experiments and collected measurements, (iii) a message broker handling communication, (iv) a scheduler responsible of executing experiments, and (v) an evaluation service computing the performance metrics. Most communication in E-Cube utilizes a variant of D-Cube's messaging protocol (DCM), which utilizes the broker to relay messages implementing remote procedure calls with a server exposing functionality and a client remotely calling these functions. As shown in Fig. 1c, each observer module starts a DCM server which handles commands from the scheduler such as turning the device on or off. Commands requiring larger payloads use a REST API to download the required data, such as the `csv`-formatted energy trace (to be replayed via the μSMU). DCM implements a strict separation of concerns pattern, allowing the scheduler to implement the high-level flow of an experiment (e.g., flashing the provided firmware). The observer module thus does not need to know the order of steps required for a given experiment, but only how to execute each step on a low level (e.g., the location, name, and arguments of the executable needed to program the target device).

5 E-CUBE: PROOF-OF-CONCEPT

We next implement the benchmark example presented in Sect. 4.1 using E-Cube as well as a simple solution. This way, we show that E-Cube indeed satisfies the requirements listed in Sect. 3.

Automation. E-Cube allows us to upload a firmware and to specify the experiment's duration and energy trace. It then automatically returns not only the raw measurements (see an excerpt of these in Fig. 2a), but also a human-readable report (see Fig. 2b) and an overall score to rank the solution against all others.

Repeatability. As for any benchmark, it is vital that results are repeatable, i.e., that given the same inputs (firmware, energy trace, etc.) E-Cube derives the same result. However, due to variability in the components caused by environmental changes, there remains an inherent variance between experiments. To quantify it, we run a series of ten consecutive experiments with the aforementioned firmware, energy trace, and challenges for 60 seconds, and obtain identical results for each run (in terms of number of solved challenges). We repeat the experiment over several days to verify that the environment did not affect the outcome. Due to the coarse resolution of hashcash – one challenge takes several milliseconds to seconds depending on the difficulty – we also run a task that allows for a higher granularity. We hence add `printf` output to our firmware, and also collect the serial output throughout the 60 seconds experiment. We can see that the final line printed is off by merely a few characters at a baud rate of 9600 (i.e., a few hundred microseconds), which confirms that our variance is minimal.

Generality. The employed μSMU can power devices with a wide variety of energy traces, and building on D-Cube's target-agnostic design only requires a USB connection for programming as well as a few pins for communication and power. This enables E-Cube to benchmark a wide range of target devices in the future.

Low-cost. The most expensive part of E-Cube is the μSMU: depending on component availability it costs ≈60€. As such, E-Cube can be considered a versatile, yet low-cost solution for benchmarking battery-free systems. Depending on the computational power desired, the other expensive part is a RPI SBC. We decided to use an older RPI 4B as our hashcash-based benchmark requires nearly no CPU resources, i.e., almost any RPI can be used as is, or with minor modifications to the GPIO code. In theory, any other SBC could be used, as long as it has GPIO pins. In summary, an E-Cube observer module can be built for ≈200€, including the target node.

Public availability and exposure. Following our experience with D-Cube, we have opened the preliminary E-Cube testbed to the community [14]. To date, 17 international teams have obtained an account and are actively using our benchmarking facility. Our ultimate goal is to organize a sustainability competition resembling the format used in the EWSN dependability competition series [6]. We have recently published a call for competitors [10] with the goal of hosting a contest within the end of the year.

Next steps. While we did verify that the benchmarked results obtained on the same device are repeatable, we are still in the process of scaling E-Cube to multiple devices and to evaluate the impact of different instances of the target HW and μSMU on the obtained results. With the recent advances of easy-to-use simulation tools for battery-free systems [8], E-Cube could further be used to help deriving and verifying the models for such systems.

6 CONCLUSIONS

In this paper, we have moved our first steps towards the design of E-Cube, a fully-automated, open, and low-cost benchmarking facility for battery-free IoT systems. After presenting E-Cube's design and architecture, we have showcased a proof-of-concept implementation of a benchmark. We have also opened our preliminary implementation of E-Cube to the community, and plan to use it as a basis to run a sustainability competition quantitatively comparing the performance of state-of-the-art battery-free solutions.

REFERENCES

[1] Cedric Adjih, Emmanuel Baccelli, Eric Fleury, Gaetan Harter, Nathalie Mitton, Thomas Noel, Roger Pissard-Gibollet, Frederic Saint-Marcel, Guillaume Schreiner, Julien Vandaele, and Thomas Watteyne. 2015. FIT IoT-LAB: A Large Scale Open Experimental IoT Testbed. In *Proc. of the 2nd World Forum on the Internet of Things (WF-IoT)*. IEEE, 459–464.

[2] Saad Ahmed, Bashima Islam, Kasim Sinan Yildirim, Marco Zimmerling, Przemysław Pawełczak, Muhammad Hamad Alizai, Brandon Lucia, Luca Mottola, Jacob Sorber, and Josiah Hester. 2024. The Internet of Batteryless Things. *Commun. ACM* 67, 3 (Feb. 2024), 64–73.

[3] Adam Back. 2002. Hashcash - A Denial of Service Counter-Measure. http://www.hashcash.org/papers/hashcash.pdf.

[4] Carlo Alberto Boano. 2021. Enabling Support of Legacy Devices for a more Sustainable Internet of Things. In *Proc. of the 1st International Conference on Information Technology for Social Good (GoodIT)*. ACM, 97–102.

[5] Carlo Alberto Boano, Simon Duquennoy, Anna Förster, Omprakash Gnawali, Romain Jacob, Hyung-Sin Kim, Olaf Landsiedel, Ramona Marfievici, Luca Mottola, Gian Pietro Picco, Xavier Vilajosana, Thomas Watteyne, and Marco Zimmerling. 2018. IoTBench: Towards a Benchmark for Low-power Wireless Networking. In *Proc. of the 1st International Workshop on Benchmarking Cyber-Physical Networks and Systems (CPSBench)*. IEEE, 36–41.

[6] Carlo Alberto Boano, Markus Schuß, and Kay Römer. 2017. EWSN Dependability Competition: Experiences and Lessons Learned. *IEEE Internet of Things Newsletter* (March 2017).

[7] Hannah Brunner, Carlo Alberto Boano, and Kay Römer. 2022. Leakage-Aware Lifetime Estimation of Battery-Free Sensor Nodes powered by Supercapacitors. In *Proc. of the 10th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems (ENSsys)*. ACM, 892—-898.

[8] Hannah Brunner, Jasper de Winkel, Carlo Alberto Boano, Przemysław Pawełczak, and Kay Römer. 2024. Simba: A Unified Framework to Explore and Facilitate the Design of Battery-Free Systems. In *Proc. of the 23rd International Conference on Information Processing in Sensor Networks (IPSN)*. ACM.

[9] Jasper de Winkel, Carlo Delle Donne, Kasim Sinan Yildirim, Przemysław Pawełczak, and Josiah Hester. 2020. Reliable Timekeeping for Intermittent Computing. In *Proc. of the 25th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. ACM, 53–67.

[10] EWSN 2024 Sustainability Competition. 2024. Call for Competitors. https://iti-ecube.tugraz.at/wiki/index.php/Call.

[11] Francesco Fraternali, Bharathan Balaji, Yuvraj Agarwal, Luca Benini, and Rajesh Gupta. 2018. Pible: Battery-Free Mote for Perpetual Indoor BLE Applications. In *Proc. of the 5th Conference on Systems for Built Environments (BuildSys)*. ACM, 184–185.

[12] Kai Geissdoerfer, Mikołaj Chwalisz, and Marco Zimmerling. 2019. Shepherd: A Portable Testbed for the Batteryless IoT. In *Proc. of the 17th International Conference on Embedded Networked Sensor Systems (SenSys)*. ACM, 83–95.

[13] Graz University of Technology. 2024. D-Cube: A Low-Power Wireless Networking Benchmark. https://iti.tugraz.at/d-cube.

[14] Graz University of Technology. 2024. E-Cube. https://iti-ecube.tugraz.at.

[15] Josiah Hester, Timothy Scott, and Jacob Sorber. 2014. Ekho: Realistic and Repeatable Experimentation for Tiny Energy-Harvesting Sensors. In *Proc. of the 12th Conference on Embedded Network Sensor Systems (SenSys)*. ACM, 330–331.

[16] Timofei Istomin, Matteo Trobinger, Amy Lynn Murphy, and Gian Pietro Picco. 2018. Interference-Resilient Ultra-Low Power Aperiodic Data Collection. In *Proc. of the 17th International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 84–95.

[17] Romain Jacob, Marco Zimmerling, Carlo Alberto Boano, Laurent Vanbever, and Lothar Thiele. 2021. Designing Replicable Networking Experiments With Triscale. *Journal of Systems Research (JSys)* 1, 1 (Nov. 2021).

[18] Dominique Larcher and Jean-Marie Tarascon. 2014. Towards Greener and more Sustainable Batteries for Electrical Energy Storage. *Nature Chemistry* 7, 1 (Nov. 2014), 19–29.

[19] Roman Lim, Reto Da Forno, Felix Sutton, and Lothar Thiele. 2017. Competition: Robust Flooding using Back-to-Back Synchronous Transmissions with Channel-Hopping. In *Proc. of the 14th International Conference on Embedded Wireless Systems and Networks (EWSN)*. ACM, 270–271.

[20] Roman Lim, Federico Ferrari, Marco Zimmerling, Christoph Walser, Philipp Sommer, and Jan Beutel. 2013. FlockLab: A Testbed for Distributed, Synchronized Tracing and Profiling of Wireless Embedded Systems. In *Proc. of the 12th International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 153–166.

[21] Xiaoyuan Ma, Peilin Zhang, Ye Liu, Carlo Alberto Boano, Hyung-Sin Kim, Jianming Wei, and Jun Huang. 2020. Harmony: Saving Concurrent Transmissions from Harsh RF Interference. In *Proc. of the International Conference on Computer Communications (INFOCOM)*. IEEE, 1024–1033.

[22] Sayedsepehr Mosavat, Matteo Zella, Marcus Handte, Alexander Julian Golkowski, and Pedro José Marrón. 2023. Experience: ARISTOTLE: wAke-up ReceIver-based, STar tOpology baTteryLEss sensor network. In *Proc. of the 22nd International Conference on Information Processing in Sensor Networks (ISPN)*. ACM, 177–190.

[23] Sayedsepehr Mosavat, Matteo Zella, and Pedro José Marrón. 2021. SOCRAETES: SOlar Cells Recorded And EmulaTed EaSily. In *Proc. of the 18th International Conference on Embedded Wireless Systems and Networks (EWSN)*. Junction Publishing, 183–184.

[24] Markus Schuß, Carlo Alberto Boano, and Kay Römer. 2018. Moving Beyond Competitions: Extending D-Cube to Seamlessly Benchmark Low-Power Wireless Systems. In *Proc. of the 1st International Workshop on Benchmarking Cyber-Physical Networks and Systems (CPSBench)*. IEEE, 30–35.

[25] Markus Schuß, Carlo Alberto Boano, and Kay Römer. 2020. Making D-Cube an Open Low-Power Wireless Networking Benchmark. In *Proc. of the 17th International Conference on Embedded Wireless Systems and Networks (EWSN), poster session* (Lyon, France). Junction Publishing, 164–165.

[26] Markus Schuß, Carlo Alberto Boano, Manuel Weber, and Kay Römer. 2017. A Competition to Push the Dependability of Low-Power Wireless Protocols to the Edge. In *Proc. of the 14th International Conference on Embedded Wireless Systems and Networks (EWSN)*. Junction Publishing, 54–65.

[27] Markus Schuß. 2022. *Benchmarking Low-Power Wireless Networking Systems*. Ph. D. Dissertation. Graz University of Technology, Austria.

[28] Sivert T Sliper, Domenico Balsamo, Nikos Nikoleris, William Wang, Alex S Weddell, and Geoff V Merrett. 2019. Efficient State Retention through Paged Memory Management for Reactive Transient Computing. In *Proc. of the 56th Design Automation Conference (DAC)*. IEEE, 1–6.

[29] Statista. 2024. Number of Internet of Things (IoT) Connected Devices Worldwide from 2019 to 2023, with Forecasts from 2022 to 2030. https://tinyurl.com/msza2dtr.

[30] Joel Troughton. 2024. μSMU. https://github.com/joeltroughton/uSMU.

[31] Roman Trüb, Reto Da Forno, Lukas Sigrist, Lorin Mühlebach, Andreas Biri, Jan Beutel, and Lothar Thiele. 2020. FlockLab 2: Multi-Modal Testing and Validation for Wireless IoT. In *Proc. of the 3rd International Workshop on Benchmarking Cyber-Physical Systems and Internet of Things (CPS-IoTBench)*. OpenReview.net, 1–7.