Battery-less Zero-maintenance Embedded Sensing at the Mithræum of Circus Maximus

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[*†]

[*]Politecnico di Milano (Italy), [†]RI.SE Sweden, [+]LUMS (Pakistan), [‡]Air University (Pakistan), [#]Microsoft Italy, [△]KU Leuven (Belgium), [∠]Credit Suisse (Poland)

ABSTRACT

We present the design and evaluation of a 3.5-year embedded sensing deployment at the *Mithræum of Circus Maximus*, a *UNESCO-protected underground* archaeological site in Rome (Italy). Unique to our work is the use of energy harvesting through thermal and kinetic energy sources. The extreme scarcity and erratic availability of energy, however, pose great challenges in system software, embedded hardware, and energy management. We tackle them by testing, for the first time in a multi-year deployment, existing solutions in intermittent computing, low-power hardware, and energy harvesting. Through three major design iterations, we find that these solutions operate as isolated silos and lack integration into a complete system, performing suboptimally. In contrast, we demonstrate the efficient performance of a hardware/software co-design featuring accurate energy management and capturing the coupling between energy sources and sensed quantities. Installing a battery-operated system alongside also allows us to perform a comparative study of energy harvesting in a demanding setting. Albeit the latter reduces energy availability and thus lowers the *data yield* to about 22% of that provided by batteries, our system provides a comparable *level of insight* into environmental conditions and structural health of the site. Further, unlike existing energy-harvesting deployments that are limited to a few months of operation in the best cases, our system runs with *zero maintenance* since almost 2 years, including 3 months of site inaccessibility due to a COVID19 lockdown.

CCS CONCEPTS

• **Computer systems organization** → **Sensor networks**; *Embedded software.*

KEYWORDS

Intermittent computing, low-power hardware, energy harvesting.

ACM Reference Format:

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 *The 18th ACM Conference on Embedded Networked Sensor Systems (SenSys '20), November 16–19, 2020, Virtual Event, Japan.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3384419.3430722

SenSys '20, November 16–19, 2020, Virtual Event, Japan

ACM ISBN 978-1-4503-7590-0/20/11…$15.00
https://doi.org/10.1145/3384419.3430722

1 INTRODUCTION

Ambient energy harvesting is progressively enabling battery-less embedded sensing. A variety of harvesting techniques exist that apply to, for example, light, vibrations, and thermal phenomena [14]. These technologies are naturally attractive wherever replacing batteries is unfeasible or impractical, and represent a foundation to achieve *zero-maintenance* embedded sensing [53].

Real-world deployments. Besides systems that use solar radiation as energy source, few examples exist of long-term deployments demonstrating energy-harvesting zero-maintenance systems [20, 21, 45], as we discuss in Sec. 2. The longest-running such deployment is reported to be operational for 3 months [20]. Further, very few of these deployments serve the needs of actual end users; rather, they are most often instrumental to demonstrate isolated software, hardware, or energy harvesting techniques. We argue that the limited span and scope of such real-world experiences is a sign that current technology is not ready for prime time, as a complete-system perspective is sorely missing.

This paper is about our first-hand experience of such state of affairs, specific to a 3.5-year embedded sensing deployment at the *Mithræum of Circus Maximus*, a UNESCO-protected archaeological site in Rome (Italy). Such an effort is prompted by the municipality of Rome, motivated by the need to understand environmental and structural conditions of the site, as we illustrate in Sec. 3. The site, shown in Fig. 1, is generally closed to the public, completely *underground*, and only accessible through spiral staircases and provisional ladders. Access to the site is strictly regulated to avoid gatherings that may create detrimental environmental conditions and requires authorization from the municipality to assign an accompanying officer. Artificial lighting is temporary, as it is deployed impromptu by archaeologists and restorers only for the duration of their visits.

Our work. Our deployment unfolds through three distinct phases, shown in Fig. 2 and summarized in Fig. 3.

The first design iteration, called Kingdom and illustrated in Sec. 4, is largely based on off-the-shelf components and operates with batteries. We use a commercial platform coupled with acceleration, inclination, temperature, and relative humidity sensors, along with a sub-GHz radio. Despite its satisfactory performance

(a) Mitra altar. (b) Concrete columns.

Figure 1: *Mithræum of Circus Maximus* in Rome, Italy. *The site is underground and only accessible through spiral staircases and provisional ladders, along with proper authorizations.*

during operational times, its reliability is limited, mainly because of batteries. Due to the difficulties to access the site to replace them, this renders the system impractical. After 1.5 years of operation, we eventually turn to energy harvesting. Besides making battery replacement a hurdle, however, the site characteristics rule out most of the energy-rich sources, notably including light.

The second design iteration, called Republic and described in Sec. 5, starts out from the overly optimistic belief—somehow fueled by the lack of experiences akin to ours—that relying on ambient energy is as simple as replacing batteries with a suitable harvester. Due to the site characteristics, we rely on thermal and kinetic sources, harvesting energy from temperature gradients and structural vibrations. We do not expect to achieve energy-neutral operation [8, 65], and design the system as an intermittently-executing one [41]. Intermittent executions interleave periods of active operation with periods of solely recharging energy buffers. We use existing programming techniques [15, 60, 72] to implement sensing, data processing, and communication. The system now operates with essentially zero maintenance, but lower energy availability causes data yield to degrade compared to the Kingdom, which we keep in place (and continue to maintain) as a baseline.

Based on the lessons learned from the earlier designs, the third iteration, called Empire and discussed in Sec. 6, is rooted in two key observations, namely *i)* a hardware/software co-design is required to efficiently manage the little available energy, and *ii)* in our deployment, a form of coupling exists between energy sources and sensed quantities [21, 64]. We make the former concrete through dedicated hardware designs that tightly integrate with program structure and execution model. As for the latter, we capitalize on structural vibrations representing both the energy source and the data we sense. As a result, while not remedying the decreased data yield, Empire greatly improves the *level of insight* into environment conditions and structural health of the site, provably bringing it back to the same level as the battery-operated system.

Outcomes. We report on site-specific insights from sensed data and on system performance in Sec. 7. We consider Kingdom as a baseline for our evaluation, as similar technology demonstrates remarkable measurement accuracy in previous embedded sensing deployments [12, 17–19, 27, 32, 44, 51, 66].

We illustrate the novel understanding of the *Mithræum* conditions we offer to the end users, and how that influences restoration and preservation activities. We show, for example, that relative humidity levels easily cross 90% in a 21C°-25C° temperature range, motivating the need of dedicated preservation procedures. We also analyze the performance trade-offs through the three design iterations and compare energy harvesting to battery-powered operation. We specifically show that in the same conditions, energy harvesting reduces energy availability and thus lowers the system's data yield to about 22% of that provided by batteries, but our design in Empire retains quality of collected data. For example, the conclusions drawn on the site's structural conditions remain unaltered using energy harvesting in Empire as compared to batteries in Kingdom.

Still in Sec. 7, the account of our experience culminates in demonstrating the *zero-maintenance* operation of Empire. Amidst the COVID19 lockdown in Rome, Kingdom goes down as batteries are exhausted while we are prevented from accessing the site, and yet Empire makes the most of the little energy available by promptly recording the occurrence of a moderate earthquake on May 11th, 2020. Analysis of our acceleration data result in a 3.14 estimate of Richter magnitude, close to the (3.2, 3.7) interval officially reported using professional seismographs [29].

In Sec. 8, we discuss key take-aways and design choices that apply more generally to zero-maintenance embedded sensing systems in contrast to the ones that are specific to our deployment. Sec. 9 ends the paper with brief concluding remarks.

2 BACKGROUND AND RELATED WORK

Our work touches upon different areas. We discuss next the relation to those works we deem closer to ours.

2.1 Deployments

A rich body of literature exists on deploying battery-powered embedded sensing systems at different scales and in various environments [12, 17–19, 27, 32, 44, 51, 63, 66, 68, 80]. Common to these efforts are the many sources of unreliable operation and the hectic experience with frequent battery replacements. Lessons from these works help us swiftly set up a fully functional Kingdom, but despite decades of research, limited and unpredictable battery lifetime remains the root cause of malfunction.

Various works demonstrate prolonged lifetime using rechargeable batteries backed by solar [1, 21, 24, 28, 46, 64, 67] or sometimes kinetic and thermal energy harvesting [21, 64]. The longest such deployment is understandably based on solar, and demonstrates a 2-year uninterrupted operation [24]. In contrast, deployments based on thermal [64] and kinetic [21] energy harvesting are limited in lifespan, extending up to four weeks [64]. Our deployment location is void of solar energy, mandating the use of lower-energy sources like thermal and kinetic, yet the lifetime performance of Empire matches the one of the longest deployment using solar energy.

Fewer examples exist replacing rechargeable batteries with environment-friendly super-capacitors [20, 34, 46, 55, 70] or regular capacitors [45, 73, 75–77] to buffer energy and smoothen harvesting fluctuations. Again, only a fraction of these works consider energy sources other than solar [55, 73, 75, 77], let alone real-world deployments [20, 45]. The longest such deployment uses microbial fuel cells to power nodes for water quality monitoring for three months [20]. Although these efforts communicate invaluable

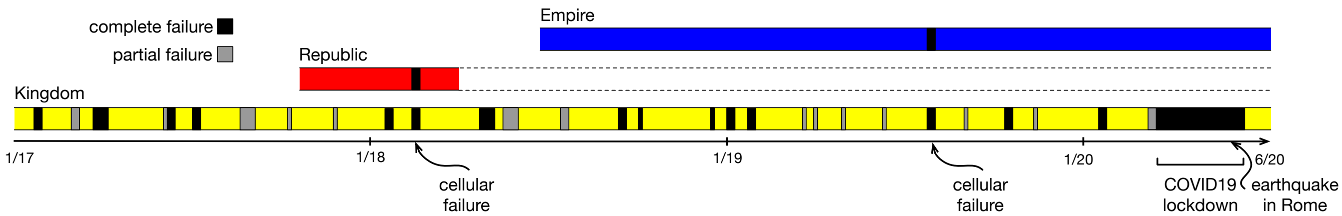


Figure 2: Time evolution of deployments at *Mithræum of Circus Maximus.* *Kingdom is battery-operated and covers the entire deployment duration, representing a baseline for the other systems. Republic and Empire use energy harvesting. A partial (total) failure occurs when at least one (all) device(s) stop operating, and lasts until it is resolved through one or multiple site visits.*

Phase	Time span	Energy source	Hardware
Kingdom	January 2017 - time of writing	Batteries	Off the shelf
Republic	November 2017 - March 2018	Thermal and kinetic	Custom
Empire	June 2018 - time of writing	Thermal and kinetic	Custom

Figure 3: Design iterations at *Mithræum.* *We name them after the three major ages of ancient Rome. These names, however, have no relation to the legacy of the* Mithræum.

lessons on specific techniques, they provide no evidence of a complete system design. Similarly, only a few of them concretely fulfills the requirements of real end users [20, 45], unlike what we do here.

2.2 System Support

Limited form factors impose restrictions on the harvesting unit, limiting power supply to tens of *mW* [20, 34, 46]. This creates a demand-supply gap, which is tackled through two possible approaches: *energy-neutral system design* and *intermittent computing*.

Energy-neutral systems. The idea is to aggressively tune system performance to achieve a demand-supply balance, thus enabling continuous operation [8, 9, 33, 69, 78, 79]. A range of hardware and software optimizations exist to improve energy generation or reduce its consumption, such as maximum power point tracking (MPPT) [9, 75], variable duty-cycling [69, 78, 79] and dynamic voltage and frequency scaling (DVFS) [8].

Techniques for energy neutrality, however, tend to cap the system performance, squeezing the set of feasible applications. Energy neutrality, moreover, may simply not be feasible whenever the average input power is lower than a minimum requirement. This is precisely our setting, where the thermal and kinetic sources offer an insufficient energy content to even conceive continuous operation.

Intermittent computing. Unlike energy-neutral system design, intermittent computing allows energy to buffer for performing operations whose power consumption may exceed the maximum harvesting capabilities. Executions thus become intermittent [41]: periods of active operation are interspersed with periods for recharging energy buffers, while the rest of the system is quiescent.

Intermittent systems typically employ techniques such as checkpointing [4, 10, 11, 15, 47, 60, 61, 72, 85] or task-based programming abstractions [22, 57, 59, 62, 74, 89] to recover from power failures. The former consist in replicating the application state on non-volatile memory, where it is retrieved back once the system resumes with sufficient energy. The latter target mixed-volatile platforms and offer abstractions that programmers use to define and manage persistent state, while taking care of data consistency in case of repeated executions of non-idempotent code [85].

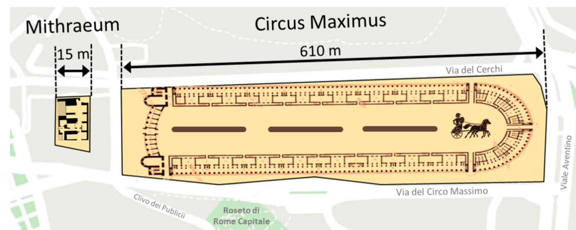


Figure 4: *Mithræum* location relative to the *Circus Maximus*.

Most existing solutions in intermittent computing, again, operate in isolation and lack integration into a complete system. Our work uses a hardware/software co-design for higher efficiency in a complete system and ultimately represents one of the few examples of intermittent computing long-term deployment.

3 MOTIVATION

The *Mithræum of Circus Maximus* is an archaeological site in Rome (Italy). It is largely considered one of the "hidden gems" of its age [81] and is part of the larger UNESCO heritage site in Rome [84].

Site. The *Mithræum* was accidentally discovered in 1931 while performing construction works to build a workshop for the local Opera Theater. Historians conjecture that the location was originally used to host horses and carriages (*carceres*) before entering the nearby *Circus Maximus* for the traditional chariot races. Fig. 4 shows the location of the *Mithræum* relative to the *Circus Maximus*. In the third century *d.C.*, a place of worship to god Mitra was created.

The site unfolds as a series of small communicating rooms, covered by barrel vaults whose remains are shown in Fig. 1. Of unique historical and artistic value are the plaster layers on the walls and the Mitra altar, shown in Fig. 1(a). The workshop of the Opera Theater currently sits right above the *Mithræum* and hosts large machinery and equipment for building theatrical backdrops and sceneries. A set of concrete columns support the ground level of the workshop, reaching into several of the rooms of *Mithræum* or standing on top of the barrel vaults, as shown in Fig. 1(b).

Goal and requirements. The *Mithræum* belongs to a set of archeological sites the municipality of Rome plans to open up to the larger public. Before doing so, an intense process of preservation and restoration is to be carried out. These activities must be planned and executed based on a thorough understanding of the current conditions. Two environmental aspect are key at the *Mithræum*:

[R1] Temperature and relative humidity of plasters: The integrity of the plaster layers may be affected by specific patterns of temperature and relative humidity. Given a certain

Measurement	Sensor	Accuracy	Relevance	Device
Temperature	SHT85	±0.1C°	Plasters	I/A and T/H
Relative humidity	SHT85	±1.5%	Plasters	I/A and T/H
Acceleration	ADIS16210	±1mg	Vaults	T/H
Inclination	ADIS16210	±0.1°	Vaults	T/H

Figure 5: Sensed physical quantities, corresponding sensing equipment, and device configuration.

temperature, a threshold exists in humidity where hygroscopic salts start forming on the surfaces. The salts absorb water from vapor in the air, causing a corrosion process to happen on the surface. In a site with no external ventilation like the *Mithræum*, this process may only be prevented using specific chemicals whose type, quantity, and method of deposition depend on temperature and humidity [56].

[R2] **Vibrations around the barrel vaults:** Vibrations originating from surrounding vehicular traffic and from the activities at the workshop above may affect the structural stability of the site, and be especially detrimental to the integrity of the barrel vaults [38]. No studies currently exist on the structural conditions of the site and no evidence is available motivating the need for specific interventions, such as installing auxiliary reinforcements of the barrel vaults or deploying dedicated damping mechanisms [38].

Collecting data to support a quantitative investigation on these aspects at the *Mithræum* must co-exist with specific constraints:

[C1] *Placing devices* to record vibrations is difficult, as it requires installing accelerometers *on the columns* supporting the Opera Theater workshop. This literally necessitates climbing up the barrel vaults to access the device, putting at risk the operator safety and the integrity of the vaults. This kind of maintenance operations are to be reduced to a minimum.

[C2] *Form factors* must be reduced, because of the visual impact on historical and artistic pieces. Since the very beginning of our effort, this aspect limits the size of deployed batteries. Such a constraint is not unique to our experience and many embedded sensing deployments, especially in heritage buildings, share similar limitations [12].

[C3] *Commercial chemical batteries* are considered dangerous by the restorers. With average relative humidity values in excess of 90% at the *Mithræum*, as discussed in Sec. 7, the chances that batteries start leaking greatly increase [90]. This is, of course, not welcome in such a sensitive environment.

Lowering the *maintenance effort* is thus key, as it determines how practical is the system and, thus, beneficial for end users.

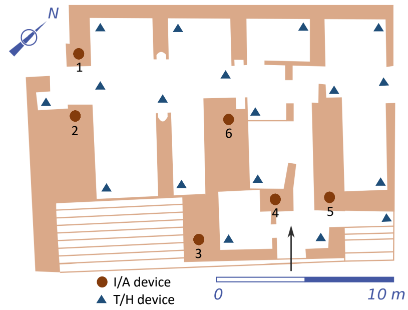
4 BATTERIES → KINGDOM

We set off by using commercial off-the-shelf components. As such, KINGDOM represents a baseline based on established solutions.

4.1 Design and Deployment

We describe next the hardware we use for KINGDOM, the software we implement, and the initial deployment at *Mithræum*.

Hardware. We use Libelium Waspmotes [54] as the computation and communication core. We couple the computing core with an XBee 868LP sub-GHz radio for communication to a data sink.



(a) Deployment map.

(b) Paper authors installing I/A devices.

(c) I/A device in place.

Figure 6: Deployment at *Mithræum*. *We install 18 T/H devices with temperature and humidity sensors and 6 I/A devices with temperature, humidity, inclination, and acceleration sensors.*

Fig. 5 summarizes the deployed hardware. To read temperature and humidity, we use a Sensirion SHT85 digital sensor through I^2C because of the low-power operation and temperature accuracy, which is sufficient to enable the analysis sought by the restorers [38, 56]. It also features a PTFE membrane for protection against liquids and dust as per IP67 specifications, without affecting the response time. The nodes equipped with this sensor are termed T/H nodes.

Acceleration readings are obtained through an Analog Devices ADIS16210 combined inclinometer and accelerometer, connected through SPI on a subset of the deployed devices. High accuracy of acceleration sensing and availability of the on-board inclinometer motivate this choice; the latter may be used to detect permanent changes in the structure [43]. We calibrate each sensor using a shake table and piezoelectric accelerometers for seismic vibrations as a reference [19]. The nodes equipped with this sensor *in addition* to the temperature/humidity one are termed I/A devices.

Software. We implement a periodic procedure to sense temperature and humidity every 20 minutes and to locally store the readings. At every hour, average and standard deviation of these quantities are computed and reported via radio to the sink. Such a sensing period is deemed to provide sufficient granularity [38, 56].

Onboard I/A nodes, every 20 minutes we additionally record a one minute burst of acceleration readings at 400Hz and sample the inclinometer, according to the guidelines of the structural engineers. At every hour, we process acceleration data by computing the Fast Fourier transform and determining the fundamental frequency as well as spectral density. These information are compressed and also reported to the sink. Such a form of periodic acceleration sensing is common to many deployments for structural analysis [43].

Upon reception, the sink timestamps the data along a global time reference. Between every sensing period, the radio is switched off and the system is placed in low-power mode.

Deployment. Fig. 6 illustrates the deployment. We install a total of 24 devices; 18 devices of type T/H, and 6 devices of type I/A, laid down as shown in Fig. 6(a). For the latter, we use industry-grade epoxy resins to attach the inclinometer/accelerometer sensor to the structure, as shown in Fig. 6(b) and Fig. 6(c) during and after installation. The devices are powered with six type-C batteries.

We deployed a data sink using a Raspberry Pi 3 computer, not shown in the picture, connected to the Internet via 4G. The sink is powered from the grid and, due to the availability of cellular connectivity, could only be installed in a different building at about 250 meters from the *Mithræum*. This motivates the choice of a sub-GHz radio, as the signal needs to penetrate two layers of concrete to reach the sink. Using this radio, no multi-hopping is necessary.

4.2 Lessons Learned

Sec. 7 provides a quantitative account of the performance of Kingdom. We anticipate the fundamental lessons learned, which are input to the following design iteration.

> **Lesson 1:** *Whenever there is sufficient energy, embedded sensing runs like a charm.*

Whenever energy is available, the system provides substantial data yield. Compared to the earlier efforts discussed in Sec. 2, we also note that the effort required to go from zero to a fully-working embedded sensing deployment also drastically reduced. We quantify this effort from one to two person-months.

> **Lesson 2:** *Batteries are the one and only aspect that makes Kingdom unreliable.*

As shown in Fig. 2, Kingdom experiences a number of failures. Batteries are ultimately accountable for all such occurrences, but for two cases of cellular failure. The latter, however, is no significant problem as the sink locally caches sensor data. Our experience contrasts the literature discussed in Sec. 2, where earlier deployment experiences resulted in a number of failures due to a variety of factors, including hardware failures and software bugs [12, 44].

Through a series of ad-hoc experiments, we try and improve the energy figure. The first attempt is based on multi-hop networking [7]. Based on lab experiments in a setting akin to *Mithræum*, we quickly realize the use of such protocols to be detrimental to energy consumption, due to control traffic overhead. In a short-term deployment at *Mithræum* alongside Kingdom, we integrate a transmission power control protocol [86]. Over a 10-day span, we measure the energy performance to improve by a mere 1.7%.

The peculiar conditions at the *Mithræum* makes predicting the system lifetime extremely difficult. High ambient humidity and temperature fluctuations cause the alkaline batteries we use to fail unpredictably. This complicates planning the maintenance visits and the associated logistics, causing the periods of down to prolong. Using different battery technology, such as industry-grade alkaline, pro-alkaline, or lithium make essentially no difference.

Despite our best efforts, maintenance represents a hampering factor regardless of the value of the data. We have two options to proceed. One possibility is to apply iterative improvements to lower the system energy consumption and extend the maintenance cycle. The unpredictability of failures would, however, remain. The other option is to tackle the root of the problem, namely, to seek energy sources other than batteries. We choose the latter.

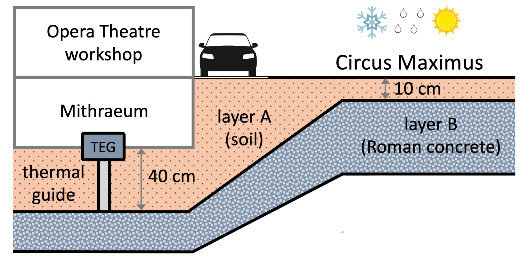


Figure 7: TEG harvester installation. *We exploit thermoelectric generation between air inside the* Mithræum *and layer B, made of Roman concrete and found 10cm below the surface at* Circus Maximus.

5 ENERGY HARVESTING → REPUBLIC

Energy harvesting is often advertised as a direct alternative to battery-powered operation [14]. Because of this, we set off by merely swapping batteries for a suitable harvester, performing the *minimum* effort to make the system work on harvested energy rather than batteries. As a result, the degree of hardware/software co-design here is limited to adapting the software to work with the chosen hardware, that is, the design process starts with the selection of hardware components and ends with the implementation of the necessary software functionality. We eventually recognize that this approach is, in fact, naive.

5.1 Design and Deployment

The opportunities for energy harvesting at the *Mithræum* are minimal. As described in Sec. 3, the site is underground and is not illuminated besides when someone is there. Moreover, the nature of the site requires minimally-invasive solutions. T/H and I/A devices thus use different energy harvesting mechanisms because of their different deployment configuration; T/H being placed next to the ground, whereas I/A being attached to the structure.

Thermoelectric energy harvesting. Fig. 7 shows the setup for T/H devices. At about 40cm below the soil at *Mithræum*, a layer of debris is found largely composed of what is called *Roman concrete* [52]. The same layer is found at the nearby *Circus Maximus* at about 10cm below the surface. Scholars conjecture that the two layers are, in fact, the same [52, 81]. This means that *Circus Maximus* potentially acts as a ≈73,000m^2 thermal surface linking the *Mithræum* to the outside. The heat flux generated because of the thermal transfer between air, layer A, and layer B creates an opportunity to employ a thermoelectric energy generator (TEG).

A broad range of commercial TEGs exists. Based on the air temperature values collected during Kingdom and the outdoor seasonal trends in Rome, we expect the thermal deltas between air and layer B to be of some K°. We thus choose a Thermalforce 254-150-36 TEG [82], offering a 30mm by 60mm harvesting surface, connected to layer B through a thermal guide, as shown in Fig. 7.

Available harvesting management circuits usually combine battery charge functionality and output voltage regulation. Solutions for TEG may be passively controlled converters or actively controlled single inductor circuits [83]. The latter offer a dynamic conversion ratio and maximum power point tracking (MPTT) [9], but require a higher minimal input voltage. Despite this, we use a BQ25570 due to its high efficiency for the range of input voltages that most likely correspond to the TEG output in our conditions.

Because the output voltage of the TEG depends on the direction of heat transfer, depending on time of the day, its output may be positive or negative. However, the BQ25570 does not support negative input voltages and hence the TEG output needs to be rectified before being input to the harvesting circuit. We build an ultra low-power rectifier using SiR404DP switches, based on the observation that the TEG output only switches twice a day [37].

Piezoelectric energy harvesting. Thermoelectric generation is not available for I/A devices, as they are too far from the ground. We absorb energy from structural vibrations to power them, taking advantage of the piezoelectric effect. The limited vibrations of the structure, however, require a careful dimensioning of the harvester and of the energy management circuitry, as we discuss next.

We employ a ReVibe modelD energy harvester [30]. The device can be customized by the manufacturer for highest efficiency at a given resonance frequency. We do this based on vibration data gathered with Kingdom. We choose this specific harvester over alternatives, for example, the modelQ [31] of the same manufacturer, because of the higher power output at the target frequencies. The harvester is attached to the columns of Fig. 1(b) using the same epoxy resins used for attaching the accelerometer/inclinometer.

Based on similar considerations as for T/H devices, we use a BQ25505 here as well. No rectifier circuit is needed.

Computing and communication. As discussed in Sec. 2.2, due to the limited energy availability, it is not conceivable to achieve energy-neutral operation [8, 65]. Therefore, we design the system to work in an intermittent fashion [41].

The Libelium Waspmote we use for Kingdom is not designed to work in such a setting. We opt to build our own computing and communication platform, using an MSP430FR5989 MCU coupled to a CC1101 transceiver. The choice of an MCU from the FR series is motivated by the need of non-volatile memory to manage persistent state. The radio chip retains the advantages of sub-GHz transmissions described in Sec. 4, with comparable energy consumption. The sensors we use are the same as in Kingdom.

We configure the output voltage of the BQ25505 buck converter to 2.2V, which represents the worst-case energy need including sensing, local processing, and data transmission. This means that the device is activated as soon as the capacitor voltage is at or above 2.2V. We also configure the BQ25505 to operate in pass-through mode whenever the capacitor voltage falls below this value, to prolong the execution for as long as possible.

We use a 20μF capacitor as energy buffer. We determine its size through a mixed analytical and experimental approach [83], striking a balance between charging times and available energy to guarantee eventual progress. A too large capacitor may take long to charge to a sufficient level, yielding large periods of no system operation when interesting environmental events might be missed. A too small capacitor may not suffice to supply enough energy to complete energy-intensive operations, such as transmitting data.

An external Abracon AB18X5 real-time clock (RTC) keeps track of the passing of time while the MCU is off, connected via I^2C. We choose this over remanence timekeepers [26, 42] because of the lower power consumption in the setting we consider. As we only require minute granularity, using the internal RC oscillator on the AB18X5 requires a mere 14nA current. Should the capacitor voltage fall below the RTC supply voltage, causing the latter to reset, we post-process the data at the sink to re-align the timestamps to the global time reference [87]. According to our logs, this happens roughly twice a year in our deployment.

Programming. As described in Sec. 2.2, system supports exist for intermittent computing [41]. In Republic, we use a static checkpoint approach [15, 72], which inlines calls to a checkpoint library to copy the complete system state onto FRAM. To place checkpoints, we profile the energy consumption of different parts of the code [3] and accordingly inline checkpoint calls. At each call, a checkpoint takes place if the capacitor voltage drops below a threshold that barely guarantees the energy to dump the state on FRAM.

We opt for static as opposed to dynamic checkpoints [10, 11, 47, 48], as we cannot afford additional hardware. Compared to task-based programming abstractions [22, 42, 57, 59] that require significant restructuring of the program [50], we wish to leverage the earlier codebase used in Kingdom.

Sensing. As the device activates depending on energy intake, the periodicity of sensing can no longer be guaranteed. Depending on harvesting performance, we may simply not have sufficient energy to activate the device every 20 minutes. As a result, we modify the local processing and data transmission functionality to execute only when the same amount of data as in Kingdom locally accumulates.

It may also happen that the required operation complete with some energy left. To avoid unnecessarily performing a checkpoint at this time, we enter a sleep state, which is a technique borrowed from Lukosevicius et al.[58]. This includes switching the radio off, putting the MCU in the lowest power mode, and setting a timer to trigger another round of sensing in 20 minutes. This also ensures that, at least in the cases where some consecutive rounds may be achieved, this happens with the same period as in Kingdom.

5.2 Lessons Learned

Republic represents the *minimum* of design and implementation effort to turn a battery-operated system into an energy-harvesting one. Similar to Sec. 4.2, we discuss here the main learned lessons and postpone the performance discussion to Sec. 7.

> **Lesson 3:** *When executions are intermittent, peripherals become markedly decisive.*

The workload at *Mithræum* is peripheral-bound. Peripherals execute asynchronously with respect to the computing unit. Their functioning is characterized by own states, frequently updated due to the execution of I/O instructions. Information on peripheral states is not automatically reflected in main memory, neither it may be simply queried or restored [16]. System support for intermittent computing often only provides support for the computing unit and expect developers to take care of peripherals [22, 57, 89]. Similarly, the few systems addressing the intermittent peripheral problem are not integrated with those for the computing unit [6, 13, 16].

To address this issue in Republic, we manually replicate the initialization procedures of all peripherals, including sensors and radio, at every point in the code where execution can possibly resume after a power failure. This is necessary as we cannot anticipate for how long an execution proceeds after resuming and thus what peripherals are used when. The profiling data we use to place checkpoint calls indicates, however, that re-initializing peripherals this way accounts for about 28% of the overall energy consumption, opening up avenues for energy savings with a better solution.

We also crucially realize how the use of radio and sensors vastly determines how far the computation can progress. We observe that the first checkpoint call right a packet transmission is systematically triggering a checkpoint, as radio operations are sufficient to cause the capacitor voltage to fall below the checkpoint threshold. However, handling peripherals is not the only source of inefficiency.

> **Lesson 4:** *When energy is scarce, sleeping may not be a smart choice.*

The technique we use in case some energy is left after completing the required workload ultimately represents a waste of energy. Based on the logs we collect, after setting a timer to expire in 20 minutes, in about 89% of the cases the node dies before the timer fires. This means that the energy invested in keeping the system in sleep mode is wasted, as another round of sensing cannot happen in the majority of the cases. In Sec. 7, we further quantify the performance impact of this design choice.

To some extent, this is again an effect of how peripherals impact the energy figure. As every time the device activates at least one peripheral is used, the chances that some energy is left that could power the sleep state for another 20 minutes are slim. This problem aggravates if the radio is also used. If we only consider the cases where we set the 20-minute timer after a packet transmission, in 98% of the cases the node dies before the timer fires.

> **Lesson 5:** *Energy availability may not necessarily overlap with events of interest.*

We expect the data yield to be affected, due to the lower availability of energy. As reported in Sec. 7, Republic can only provide about 22% of the net amount of data Kingdom provides on a monthly basis. Worse is that the information gain obtained from Republic is comparatively way below the reduction in data yield.

This observation particularly applies to I/A devices. In Kingdom, the relative abundance of acceleration data forgives that acceleration sensing is not necessarily synchronized with events of interest, such as activities at the Opera Theatre workshop or vehicular traffic. In Republic, I/A devices activate only depending on capacitor voltage levels, which might cross 2.2V merely because of vibration noise of no interest [43]. The structural engineers state that, by only using the acceleration data from Republic, *no structural analysis is possible*, due to the signal information being too poor for modal analysis, irrespective of the amount of collected data [43].

6 BETTER ENERGY HARVESTING → EMPIRE

We eventually choose to co-designing the hardware and software. This is primarily based on the experience and insights gained from Republic, but also on a number of (failed) attempts at remedying the deficiencies of Republic by only working at software level.

For example, alongside Republic, we eventually deploy two additional I/A devices with a customized software implementation that postpones acceleration sensing to the next active cycle in case the initial 1-sec burst of data indicates no specific event of interest. In a sense, we bet on the fact that we may have better chances to capture something interesting at the next time around. Over a two-week span, we realize that system performance stays the same in terms of enabling structural analysis, as the hardware may keep activating the device because of vibration noise and events of interests are completely uncorrelated with that. In terms of data yield, the performance even degrades, because of the additional processing required to decide on the possible postponement.

In contrast to Republic, therefore, we attack the problem by looking at hardware and software together. As a result, for example, we discover and exploit further opportunities by capturing the coupling between energy sources and sensed data.

6.1 Design and Deployment

We realize different designs for T/H and I/A devices. Their key elements are described next, whereas attached sensors and the timer subsystem remain the same as in Republic.

Programmable activation threshold. Fig. 8 shows the block diagram of the 2nd generation T/H device. It offers two fundamental features: *i)* it allows the MCU to dynamically configure the amount of energy available at the next device activation, and *ii)* it provides a software-controlled shutdown functionality, which the MCU uses once the required operations are completed.

To achieve these functionality, we place three voltage comparators in parallel; each corresponding to a different activation threshold. We select comparators from the BU49xx series corresponding to voltage levels matching the energy required for *i)* sensing (V_s^{th}), *ii)* sensing and local processing (V_{sp}^{th}), and *iii)* all application functionality including data transmission (V_{spt}^{th}), where $V_s^{th} < V_{sp}^{th} < V_{spt}^{th}$. Every threshold also includes the energy required to dump the state on FRAM once the necessary operations complete. An ADG704 digital multiplexer selects the comparator to use based on the input of a two-bit memory the MCU can program by manipulating two GPIO pins. The choice of components is dictated by both their low energy consumption and their matching with the voltage threshold we require, given the same capacitor size as in Republic.

We implement the two-bit memory using two SN74AUP1G74 flip-flops in a cascading configuration. These feature both an extremely limited quiescent current and a low reset voltage. Below 0.8V, however, they lose their state. We have evidence that this was the case for only eleven times in almost two years. If the flip-flops reset, the circuit causes the multiplexer to select the lowest threshold V_s^{th}. This ensures that some progress is eventually achieved.

We deploy a TPS62736 buck converter, which is optimized for the target range of currents. A further voltage detector turns the "power good" signal up to make the buck converter activate the device whenever the selected input comparator switches its output. As device activation is now separately controlled, we configure the output of the buck converter exactly to 2.1V, which represents an energy-efficient regime for both the MCU [3] and the radio [25] As in Republic, the converter operates in pass-through mode whenever the capacitor voltage falls below this value. The TPS62736 also features an independent "enable" signal that can be used by

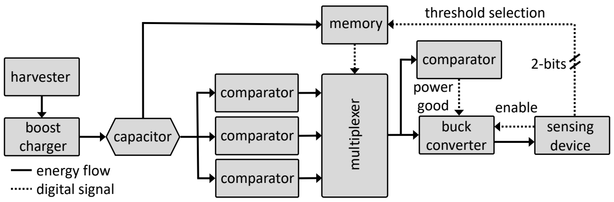


Figure 8: Second generation T/H device in EMPIRE. *The MCU can programmatically configure the amount of energy available at the next device activation and shutdown the system via software.*

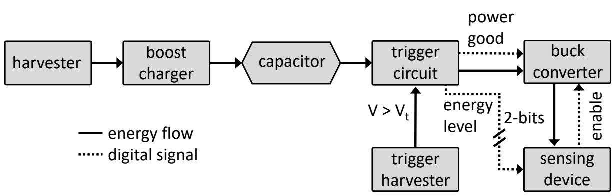


Figure 9: Second generation I/A device used in EMPIRE. *A secondary piezo element triggers device activation. A 2-bit input line informs the MCU of the energy available at activation time.*

the MCU to disconnect from the power sub-system, effectively implementing a software-controlled shutdown.

Our concept of programmable activation shares similarities with Capybara [23] and Dynamic Energy Burst Scaling (DEBS) [36], but we trade generality for a lower energy overhead. The whole power sub-system, in fact, only consumes 5.35μA of quiescent current.

Vibration-triggered activation. Fig. 9 depicts the block diagram of the 2nd generation I/A device. It features two key elements: *i)* a second piezo element that operates as a trigger, activating the device only when vibrations above a certain frequency are detected, and *ii)* a 2-bit input line that informs the MCU of the energy available at activation time, which the application uses to determine what operations may be completed in the active cycle.

We use a Piezo.com Q220-H4BR-2513YB bending transducer [71] as trigger, enclosed in a PPA-500x clamping base with a 13mg tip mass. A custom trigger circuit turns up the "power good" line of the buck converter whenever the trigger piezo generates an output voltage above a threshold V_t and the capacitor voltage is above a threshold V_s^{ia} sufficient for acceleration sensing. The buck converter then activates the device. We select both piezoelectric element and tip mass in a way that V_t can be accurately detected and corresponds to vibrations of interest [43].

A set of TLV369x comparators and SiR404DP switches control the "power good" line of the buck converter and the 2-bit "energy level" input line connected via GPIO to the MCU. Upon activating the device, the latter informs the MCU of the amount of energy available, based on whether three additional voltage thresholds are crossed. These correspond to the energy for *i)* sensing and local processing (V_{sp}^{ia}), *ii)* sensing and data transmission (V_{st}^{ia}), and *iii)* all application functionality (V_{spt}^{ia}), where $V_s^{ia} < V_{sp}^{ia} < V_{st}^{ia} < V_{spt}^{ia}$. Depending on this input, the application schedules the operations it can perform given a certain energy budget.

The roles and connections of the remaining components are similar to the T/H devices. In this case, the power sub-system only consumes 4.98μA of quiescent current.

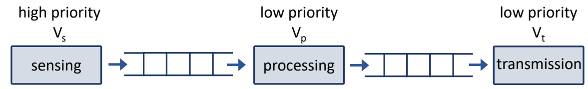


Figure 10: Task-based program structure. *Every task demands a different amount of energy. Tasks have different priorities and are connected through non-volatile data pipelines.*

Programming. Both designs aim to exert a higher control on erratic energy patterns. T/H devices achieve that by giving the MCU the ability to decide the energy available for the next iteration. I/A devices proactively provide the MCU with information on available energy at the time of activation. Both designs also give the MCU a means to shutdown the device whenever required.

Taking advantage of these features requires to co-design the software in ways to *i)* precisely isolate and decouple the functionality corresponding to different voltage thresholds, *ii)* abandon the strictly-sequential execution semantics, so different functionality can execute independent of each other, depending on available energy. In doing so, we must come to terms with the need to refactor the codebase created in KINGDOM, which is unavoidable now.

We opt for a task-based structuring of the code, shown in Fig. 10. A task is an atomic piece of functionality that executes in a transactional manner [22, 57, 59, 62, 89]. If energy suffices and a task completes, its output are committed onto a non-volatile data pipeline. If a power failure happens before the task completes, the effects of a partial execution are lost and the task restarts from the beginning. Unlike existing solutions, our design enables a form of energy-aware scheduling that simplifies system operation, while reducing overhead. Upon device activation, the sensing task is enabled and sensors are (re-)initialized. The power sub-system ensures that sufficient energy is available for this when activating the device, as V_s^{ia} is certainly crossed. We additionally enable any other task with input data and whose energy demands match the available energy and (re-)initialize (only) the necessary peripherals.

We set higher priority for the sensing task to make sure we do not miss any environment data. Among enabled tasks, we therefore run the sensing task first and commit its results on FRAM. We proceed to run the other enabled tasks and similarly commit the results on FRAM. For I/A devices, as long as sufficient energy is available not to starve the transmission task, no data buffers overflow. For T/H devices, we can proactively ensure this by configuring the activation threshold to provide the transmission task with sufficient energy.

In Sec. 8, we discuss the limitations of our work and cast our design rationale in the larger context of battery-less systems.

6.2 Lessons Learned

The efficient operation of EMPIRE, reported in Sec. 7, leads us to additional lessons learned.

> **Lesson 6:** *Determine how much energy you need, when, and for which operation.*

In EMPIRE, knowledge of energy demands and is key in our hardware/software co-design. The inefficient operation of REPUBLIC, instead, stems from the application whimsically unfolding through three distinct phases with different energy demands and periods. Sensing is moderately energy consuming and happens most often.

Local processing is the least energy consuming, but comparatively happens more rarely, as it needs a batch of sensed data to operate on. Data transmission is the most energy hungry, and must happen as frequently as local processing, as it relays the results to the sink.

Energy management in Republic was totally unaware of these aspects and only operated based on the net energy inputs from the harvesters. Existing literature, besides a few exceptions [3, 35, 39], offers little support for gaining or exploiting this information, and almost never includes peripherals in the picture.

> **Lesson 7:** *Only perform an operation when you have the energy required; consume no more, no less.*

Empire performs efficiently because it is provided with just the right amount of energy for the required operation and for committing the results on FRAM. This means, for example, that sufficient energy to perform local computation and data transmission is at disposal whenever the required amount of data is available. The device then completely switches off to avoid wasting energy doing nothing in sleep state, as in Republic.

> **Lesson 8:** *Selectively activate peripherals, and only when you need them.*

The execution pattern we enforce in Empire also allows us to shave off some of the significant energy overhead for re-initializing peripherals, discussed in Sec. 5.2. Knowing what operations are going to be performed within the given energy budget, only the required peripherals are initialized.

Republic is unable to guarantee this; for example, because we cannot anticipate for how long an execution proceeds after a power failure, as explained in Sec. 5.1. Again available solutions provide a limited foundation to build upon these observations [16, 23, 40].

> **Lesson 9:** *Capturing the coupling between energy sources and sensed quantities is key, if one exists.*

In Empire, we make use of vibrations both as energy source and as a quantity to sense. Significant vibrations are used for both harvesting energy and for triggering the sensing process, if the accumulated energy suffices. On the other hand, crossing the activation threshold merely because of vibration noise does not lead to activating a device, and we rather keep accumulating energy. Only a few solutions currently exist in this direction [21, 64].

7 EVALUATION

We study multiple complementary dimensions. In Sec. 7.1 we evaluate our deployment as a scientific instrument to fulfill the requirements in Sec. 3. We compare the system performance of the three design iterations in Sec. 7.2. The overarching question we seek to answer is whether accurate *zero-maintenance* embedded sensing is possible in our setting. We elaborate on this in Sec. 7.3.

7.1 Application

We separate the discussion of the environmental information we gather, as per requirement **R1** in Sec. 3, from the structural analysis of the site, as per requirement **R2** in Sec. 3. The differences in sensed values between the three design iterations are discussed in Sec. 7.2.

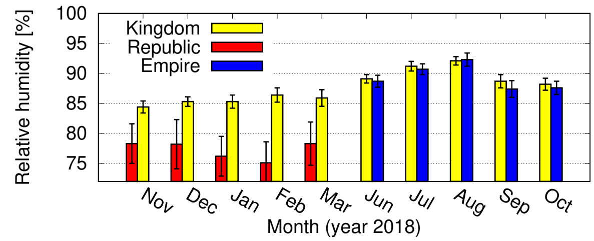


Figure 11: Monthly average of relative humidity recorded by Republic and Empire, compared to Kingdom in the same time interval. *High relative humidity and ambient temperature in the 21C° -25C° range may cause the creation of hygroscopic salts.*

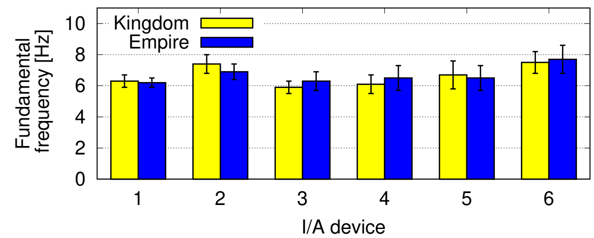


Figure 12: Fundamental frequencies using Kingdom and Empire when both systems are operational. *The structures at* Mithræum *have different fundamental frequencies than possible external excitating phenomena, ruling out resonance behaviors.*

Environment. Fig. 11 shows the monthly average of relative humidity recorded by the three systems in 2018. We obtain comparable trends also in other periods and for temperature data, making the following conclusions applicable throughout the deployment duration. The differences in sensed values between the three systems are due to different energy efficiency, as discussed in Sec. 7.2.

Fig. 11 shows that relative humidity at the *Mithræum* is markedly higher than in a regular environment, with distinct seasonal trends and peaks in the summer months. This may be attributed to the nature of the soil combined with the lack of external ventilation. Together with our recording of ambient temperature in the 21C°-25C° range, the situation corresponds to roughly 15 grams of vapor per kilogram of air, with peaks of 18 grams in the summer months. This is way above the threshold for the creation of hygroscopic salts that possibly cause corrosion processes to occur on the surfaces [56], as explained in Sec. 3, and prompts immediate action by the restorers.

This information is also crucial for a public opening of the site. Existing standards for thermal comfort indicate a maximum of 6 grams of vapor per kilogram of air [5]. This value is less than half of what we record[1]. Ensuring thermal comfort for the general public at the *Mithræum* requires the installation of an auxiliary ventilation system, at least during the opening times. In turn, this would likely change the general environment conditions at the site, making them more variable depending on the operation of the ventilation system. A permanent installation of a minimally-invasive *zero-maintenance* sensing system thus becomes even more fundamental.

Structure. Fig. 12 shows a sample output of the analysis on acceleration data, plotting the average fundamental frequencies recorded throughout the deployment at every I/A device. As explained in

[1]We also experience the thermal discomfort at the *Mithræum*, as we can barely work at the site continuously for more than a couple of hours without reaching the outside.

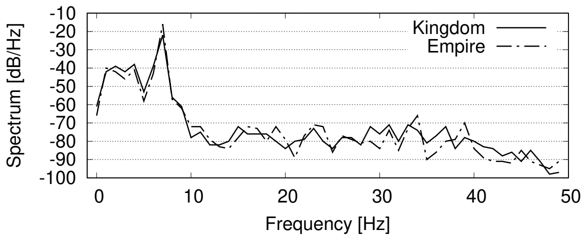


Figure 13: Spectral density at I/A device #6 using Kingdom or Empire when both systems are operational. *Only one dominant frequency exists and most of the energy is concentrated below 10Hz, supporting general validity of the structural analysis.*

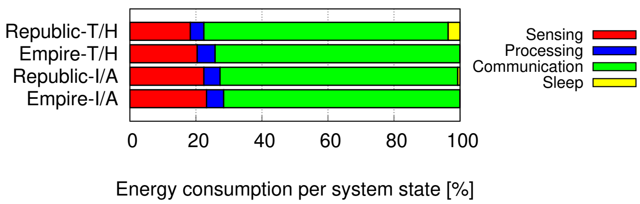


Figure 14: Breakdown of percentage energy consumption depending on functionality. *Peripherals bear an impact on energy, as per Lesson 3 in Sec. 5.2. Accurate energy management makes better use of the energy spent in sleep mode at T/H devices, as per Lesson 4 in Sec. 5.2, as well as Lesson 6, 7 and 8 in Sec. 6.2.*

Sec. 5.2, Republic provides no usable data. This information is valuable in that, if the fundamental frequency of an external excitating phenomenon match those of the structure, then the motion of the structure is amplified, resulting in resonance behavior [43].

The external phenomena may be, in our case, activities at the Opera Theater workshop or vehicular traffic. The values in Fig. 12, however, indicate that the fundamental frequencies of the structures at the *Mithræum* are relatively far from those possibly characterizing the aforementioned phenomena, which are thought to lie above 10Hz [38]. Resonance behaviors may thus be safely ruled out.

This reasoning is confirmed by the information on spectral density, shown in Fig. 13 for node #6 as an example. Only one dominant fundamental frequency exists and most of the signal energy is concentrated below 10Hz. As every fundamental frequency follows a specific deflection shape, usually referred to as vibrational mode, we can argue only one such mode exists for the structure at the sampling points. The analysis on the dominant fundamental frequency thus bears general validity [43].

7.2 System

We assess the performance of the two energy harvesting systems compared to the battery-operated one. We take Kingdom as a baseline hereafter, as the sensing equipment is the same across the three systems and only the power source and associated designs differ.

T/H devices. We return to Fig. 11 to analyze the compare the three systems. The plot shows how Republic constantly underestimates the relative humidity at the site, whereas Empire provides values closer to those of Kingdom. The variability of data around the average is also much higher for Republic than for Empire. This is an effect of Republic's inability to accurately manage the available energy, which is used opportunistically and partly wasted.

Based on detailed logs we collect at a subset of the devices, Fig. 14 quantifies this aspect by showing the breakdown of energy consumption across the four main system states in Republic and Empire. The plot demonstrates the impact of the peripherals on the energy figure, supporting our claims in Lesson 3 in Sec. 5.2. For Republic, it also shows the contribution of entering a sleep state when the required operations complete with energy left, as described in Sec. 5.1. Crucially, the latter accounts for almost the same fraction of energy consumption as local processing, thus providing a quantitative indication for Lesson 4 of Sec. 5.2. Empire shifts this energy budget to other functionality, as our hardware design offers a way for the software to shutdown the device when the current workload completes. This testifies that Lesson 6, 7 and 8 in Sec. 6.2 are key to achieve better energy management.

Fig. 15 examines data yield, plotting the amount of sensed data that reaches the end user using either of the energy-harvesting systems, normalized to the monthly performance of Kingdom throughout the deployment. Even though the data yield is slightly higher in Empire due to better energy management of T/H devices, the improved design in Empire is not meant to increase data yield, but to repurpose available energy to capture more data that describes events of interest. In contrast, inefficient energy management in Republic makes it unable to capture humidity and temperature readings faithfully, as less data reported to the sink means certain trends in humidity or temperature are missed.

Fig. 16 corroborates this reasoning by showing how Republic provides values close to Kingdom at sunrise or sunset, that is, when the heat flux from *Circus Maximus* to/from *Mithræum* is larger and thus the TEG provides more energy. We find that the number of times a T/H device activates in these periods is roughly equal for Republic and Empire. At times when the heat flux is lower, the measure obtained with Republic appear to deviate from Kingdom. Our logs indicate that the T/H devices in Republic activate about 32% fewer times in these periods compared to Empire. The latter makes better use of the lower energy available at these times, reporting as valuable information as the battery-operated system.

This performance is enabled by accurate energy management at T/H devices in Empire by *i)* programmatically configuring the energy available at the next device activation, and *ii)* enforcing a device shutdown whenever the current workload is accomplished.

I/A devices. Different from T/H devices, Fig. 14 shows that the impact of using low-power modes at I/A devices is minimal. This is consistent with Fig. 15, which shows that the monthly data yield for I/A devices is generally the same for Republic and Empire, and anyways about one fourth of what the battery-operated system can collect. In this case, the nature of the data matters.

In Kingdom, the amount of data collected is sufficient to comprehensively describe the signal features. As anticipated, no structural analysis is possible using Republic, in that acceleration data is unsuitable for modal analysis [43]. The acceleration data we gather with Republic, while being the same as Empire in net amount, is of a different nature: it poorly describes the signal features. Unlike the functioning of Empire, I/A devices in Republic may activate at times where no specific vibration of interest takes place. This substantiates our claims in Lesson 5 of Sec. 5.2.

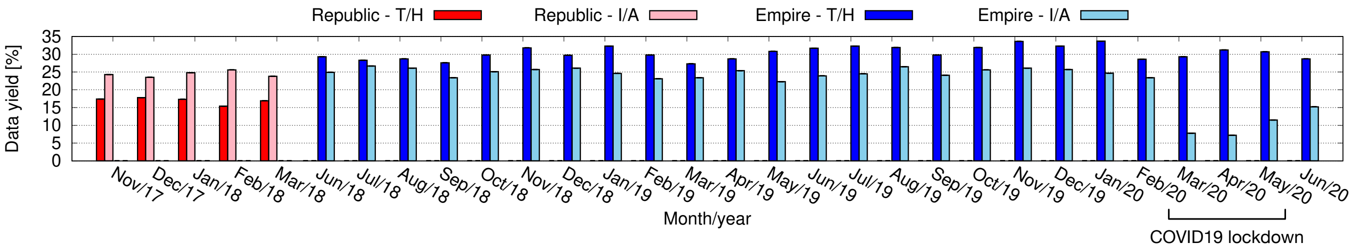


Figure 15: Monthly data yield of Republic or Empire normalized to that of Kingdom throughout the deployment duration. *For T/H devices, Republic provides a lower data yield than Empire because of less accurate management of energy. Data yield for I/A devices in Empire lowers during the COVID19 lockdown because of reduced vibrations useful for energy harvesting.*

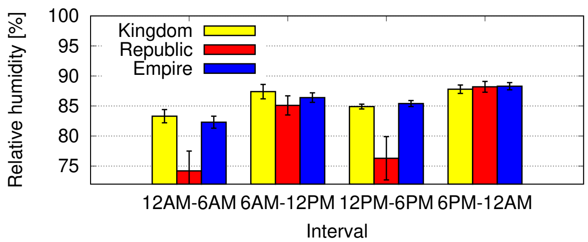


Figure 16: Average relative humidity at different times of a day, recorded by Republic and Empire compared to Kingdom throughout the system lifetime. *Less accurate energy management in Republic becomes apparent in periods of energy scarcity.*

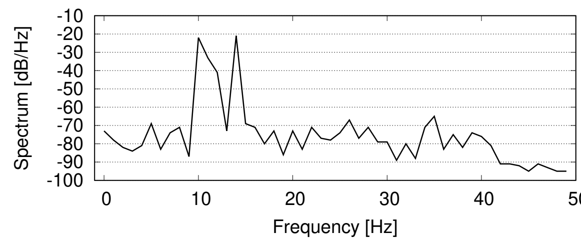


Figure 17: Spectral density at I/A device #6 using Empire during May 11th earthquake in Rome. *The data comes in a period where Empire experiences very little energy intake, due to the ongoing COVID19 lockdown. The Richter magnitude from the acceleration data we obtain is* 3.14*, close to the official* (3.2, 3.7) *estimates [29].*

Conversely, the ability of Empire to activate when a relevant phenomenon occurs counterbalances the smaller amount of collected data. Fig. 12 and Fig. 13, for example, demonstrate that the structural analysis obtained using Kingdom or Empire is largely equivalent, as the outputs are quite similar in absolute value and variability. This performance is enabled by our design of I/A devices, including *i)* the use of a secondary piezo element to activate the device upon detecting vibrations of interest, crucially based on Lesson 9 in Sec. 6.2, and *ii)* the 2-bit "energy level" input line that enables energy-aware scheduling of tasks.

7.3 Maintenance

We record a partial (total) failure of Kingdom as the point in time when at least one (all) device(s) stop operating. The duration of a failure is the time between when the failure is recorded first, until it is resolved through one or multiple site visits.

Batteries. By subtracting the time of partial or total failures from the deployment duration, we find that the uptime of Kingdom is roughly 71%. This notably includes a period of almost three months, shown on the right in Fig. 2, where a partial failure on March 3rd, 2020—eventually turned into a total failure—was impossible to resolve promptly as the city of Rome, as much as the entire country, is under a lockdown due to COVID19 [88]. During this period, citizens mobility was limited to the bare essential for one's sustenance and access to health services. As restrictions are progressively lifted in late May 2020, we access the site for the required maintenance.

Besides two failures in the cellular connection at the sink, shown in Fig. 2, all failures in Kingdom are due to battery problems. Each failure requires accessing the site for maintenance, including obtaining authorizations from the municipality, scheduling an appointment with the accompanying officer based on her availability and not overlapping with other people's visits, accessing the site for the time required, performing the maintenance work, and rebooting the system. We estimate this effort to be around six person-month until the time of writing. On a yearly basis, this equals the effort for the initial development of the system, discussed in Sec. 4.2; throughout the duration of the deployment, the maintenance effort is right now more than twice the development one.

Zero maintenance. We cannot similarly indicate a measure of system uptime for the energy-harvesting ones, due to the lack of ground truth on the availability of ambient energy. However, we can offer ultimate evidence of the *zero-maintenance* operation of Empire. We not only do not touch the system other than the initial installation, but during the lockdown we are *prevented* to do so. The lockdown is, nonetheless, apparent in the data yield of I/A devices in Fig. 15. As activities at the Opera Theater workshop come to a halt and vehicular traffic greatly lowers, vibrations useful for energy harvesting sharply reduce. The system then provides less acceleration data to end users during March, April, and May 2020.

Nonetheless, capturing the coupling between energy sources and sensed data, as per Lesson 9 in Sec. 6.2, allows the I/A devices in Empire to become operational when needed, even during the lockdown. On May 11th, 2020 at 3.03AM UTC, an earthquake of moderate intensity hits the area north of Rome [29]. The I/A devices in Empire are activated by the trigger piezo, while they accumulate enough energy to execute at least the sensing task. Fig. 17 shows the spectral density of the acceleration signal we eventually receive. The sharp difference compared to Fig. 13 testifies the different nature of the vibration, with two peaks at frequencies much higher than those in normal circumstances, as explained in Sec. 7.1. Kingdom is, on the other hand, hopelessly down since the beginning of March and thus unable to provide any data.

Using existing computational methods [49], we estimate the Richter magnitude of the earthquake from spectral density and Fast Fourier transform of the signal we collect. We obtain a value of 3.14, close to the official estimates [29] of the Italian Institutes of Geophysics that report a (3.2, 3.7) interval, obtained using numerous professional seismographs around Rome. Our estimate, in contrast, is obtained using an energy-harvesting embedded sensing device that operates with *zero maintenance* since almost two years.

8 KEY TAKE-AWAYS

We articulate how the insights we gain through our specific experience may serve to other system builders and seed new directions. Our primary message is that, in situations of energy scarcity like ours, generality in concrete implementations is a luxury one cannot afford. Different than existing literature that seeks generality in *both* concepts and concrete implementations, our experience motivates developing *general concepts* supported by *application-* or even *deployment-specific implementations*.

Evidence of our reasoning is found on the hardware side, where existing works that focus on accurate energy management [23, 36, 40] largely trade generality for overhead. The generic implementation of the federated energy architecture concept in the Flicker platform [40], for example, costs 10.24μA in device quiescent current: almost twice what we have for T/H devices in Empire. Similar observations apply to Capybara [23] and DEBS [36], both proposing useful concepts coupled with general-purpose implementations whose overhead, in settings akin to ours, are hardly tolerable.

Existing programming techniques largely seek independence from energy patterns and hardware platforms. Most task-based solutions, in particular, adopt a pure software approach [22, 57, 59, 62, 89]. In contrast, our design of Empire fundamentally builds upon Lesson 6, 7, and 8 in Sec. 6.2, as

1) the decision on what task to execute is taken not just based on the availability of input data [22, 57, 59], but also on whether sufficient energy is available; this information is known beforehand in Empire, as it is proactively provided by the power sub-system (I/A devices) or the MCU configures the activation threshold at end of the previous activation cycle (T/H devices).
2) available energy at the start of an active cycle matches the energy demands of a defined subset of tasks, and little to no energy is harvested during an active cycle; as a result, techniques such as two-phase commit of task outputs [59], run-time energy events [89], or task splitting [62] are an unnecessary overhead: if we schedule a task to start, we know it completes successfully.
3) tasks are decoupled and only connected by variable-sized data pipelines; therefore, there is no strict ordering of task executions to be guaranteed [57], neither there are relative timing constraints on their execution [42], as long as the transmission task does not starve, no buffer overflows occur.
4) partitioning the application in tasks explicates the relation between functionality and required peripherals; as a consequence, general solutions for intermittent peripheral operations become unnecessary [6, 13, 16, 23], as every task knows what peripherals it needs and only (re-)initializes those.

In general, we argue that programming techniques must not be oblivious to everything outside software, as long as a proper hardware abstraction layer is defined. Core to this is energy management, both for informing the computing core on its availability and to give the latter the knobs to exert some control on it.

Our arguments do not entail that work in this area is necessarily bound to a narrow scope. One may argue, for example, that our design in Empire is enabled by a priori knowledge of energy demands, which is generally not available and may change at run-time. Tools and techniques to accurately gain this information are, however, emerging [3]. Moreover, as we learn from Lesson 3 in Sec. 5.2, peripherals makes the case of varying run-time energy demands a rare, and often remediable issue, as they dominate the energy figure in our deployment as well as in many others [20, 21, 45, 83]. Should peripherals be used based on run-time information, our design is applicable by scaling down the granularity of individual functionality to the level of single peripheral operation [36].

We thus advocate that our experience be a basis to develop general concepts, backed by *(semi-)automatic methods* to synthesize application-specific implementations *across hardware and software*. For example, the concept of energy buffering for T/H devices in Empire, while similar to Capybara [23] and DEBS [36] that only offer generic implementations, currently has no way to be instantiated with little effort for a different application. Enabling a form of (semi-)automatic generation of hardware/software designs may reap the best of both general concepts and efficient implementations.

9 CONCLUSION

We presented the design and evaluation of a 3.5-year embedded sensing deployment at the UNESCO-protected *Mithræum of Circus Maximus* in Rome, Italy. Besides serving the end users, the effort was an opportunity to assess the state of energy harvesting embedded sensing. We did so through three design iterations.

In our Kingdom design, we find that battery-powered embedded sensing still suffers from the hectic performance of batteries. In our Republic design, we realize that using energy harvesting as a replacement for batteries is not as easy in an energy-scarce setting, mainly due to the lack of complete system solutions. In contrast, a dedicated hardware/software co-design in Empire achieves better utility for data, bringing it back to the level of a battery-powered system. Empire also shows that accurate *zero-maintenance* embedded sensing is possible in a demanding setting. While Kingdom is down since 2+ months due to battery depletion during a COVID19 lockdown, Empire accurately captures a earthquake after almost 2 years of unattended operation. Our 3.14 estimates of Richter magnitude, obtained from acceleration data collected by Empire, is remarkably close to the official (3.2, 3.7) estimate [29].

The hardware schematics and application code for the three design iterations are available [2] for the community to build on.

Acknowledgments. We thank the shepherd and reviewers for the feedback received on the initial submission. This work was supported partly by the Google Faculty Award programme and by the Swedish Foundation for Strategic Research (SSF).

REFERENCES

[1] J. Adkins, B. Ghena, N. Jackson, P. Pannuto, S. Rohrer, B. Campbell, and P. Dutta. 2018. The Signpost Platform for City-Scale Sensing. In *Proceedings of the 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*.

[2] M. Afanasov, N. A. Bhatti, D. Campagna, G. Caslini, F. M. Centonze, K. Dolui, A. Maioli, E. Barone, M. H. Alizai, J. H. Siddiqui, and L. Mottola. [n.d.]. Battery-less Zero-maintenance Embedded Sensing at the Mithræum of Circus Maximus: Hardware Schematics and Source Code. https://www.neslab.it/mitreo

[3] S. Ahmed, A. Bakar, N. A. Bhatti, M. H. Alizai, J. H. Siddiqui, and L. Mottola. 2019. The Betrayal of Constant Power × Time: Finding the Missing Joules of Transiently-powered Computers. In *Proceedings of the 20th ACM SIGPLAN/SIGBED International Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES)*.

[4] S. Ahmed, N. A. Bhatti, M. H. Alizai, J. H. Siddiqui, and L. 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)*.

[5] ANSI/ASHRAE. [n.d.]. Standard 55 - Thermal Conditions for Human Comfort. Retrieved July 10th, 2020 from https://www.ashrae.org/technical-resources/55

[6] A. R. Arreola, D. Balsamo, G. V. Merrett, and A. S. Weddell. 2018. RESTOP: Retaining External Peripheral State in Intermittently-Powered Sensor Systems. *Sensors* (2018).

[7] N. Baccour, A. Koubâa, L. Mottola, M. Zúñiga, H. Youssef, C. Boano, and M. Alves. 2012. Radio Link Quality Estimation in Wireless Sensor Networks: A Survey. *ACM Transactions on Sensor Networks (TOSN)* 8, 4 (2012).

[8] D. Balsamo, A. Das, A. S. Weddell, D. Brunelli, B. M. Al-Hashimi, G. V. Merrett, and L. Benini. 2016. Graceful Performance Modulation for Power-Neutral Transient Computing Systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2016).

[9] D. Balsamo, B. J. Fletcher, A. S. Weddell, G. Karatziolas, B. M. Al-Hashimi, and G. V. Merrett. 2019. Momentum: Power-Neutral Performance Scaling with Intrinsic MPPT for Energy Harvesting Computing Systems. *ACM Transactions on Embedded Computing Systems* (2019).

[10] D. Balsamo, A. S. Weddell, A. Das, A. R. Arreola, D. Brunelli, B. M. Al-Hashimi, G. V. Merrett, and L. Benini. 2016. Hibernus++: A Self-Calibrating and Adaptive System for Transiently-Powered Embedded Devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2016).

[11] D. Balsamo, A. S. Weddell, G. V. Merrett, B. M. Al-Hashimi, D. Brunelli, and L. Benini. 2015. Hibernus: Sustaining Computation During Intermittent Supply for Energy-Harvesting Systems. *IEEE Embedded Systems Letters* (2015).

[12] G. Barrenetxea, F. Ingelrest, G. Schaefer, and M. Vetterli. 2008. The Hitchhiker's Guide to Successful Wireless Sensor Network Deployments. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SENSYS)*.

[13] G. Berthou, T. Delizy, K. Marquet, T. Risset, and G. Salagnac. 2018. Sytare: a Lightweight Kernel for NVRAM-Based Transiently-Powered Systems. *IEEE Trans. Comput.* (2018).

[14] N. A. Bhatti, M. H. Alizai, A. A. Syed, and L. Mottola. 2016. Energy Harvesting and Wireless Transfer in Sensor Network Applications: Concepts and Experiences. *ACM Transactions on Sensor Networks* (2016).

[15] N. A. Bhatti and L. Mottola. 2017. HarvOS: Efficient Code Instrumentation for Transiently-powered Embedded Sensing. In *Proceedings of the 16th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*.

[16] A. Branco, L. Mottola, M. H. Alizai, and J. H. Siddiqui. 2019. Intermittent Asynchronous Peripheral Operations. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems (SENSYS)*.

[17] D. Carlson, J. Gupchup, R. Fatland, and A. Terzis. 2010. K2: A System for Campaign Deployments of Wireless Sensor Networks. (2010).

[18] M. Ceriotti, M. Corrà, L. D'Orazio, R. Doriguzzi, D. Facchin, G. P. Jesi, R. L. Cigno, L. Mottola, A. L. Murphy, M. Pescalli, et al. 2011. Is there light at the ends of the tunnel? Wireless sensor networks for adaptive lighting in road tunnels. In *Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN)*.

[19] M. Ceriotti, L. Mottola, G. P. Picco, A. L. Murphy, S. Guna, M. Corrà, M. Pozzi, D. Zonta, and P. Zanon. 2009. Monitoring Heritage Buildings with Wireless Sensor Networks: The Torre Aquila Deployment. In *Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN)*.

[20] Q. Chen, Y. Liu, G. Liu, Q. Yang, X. Shi, H. Gao, L. Su, and Q. Li. 2017. Harvest Energy from the Water: A Self-Sustained Wireless Water Quality Sensing System. *ACM Transactions on Embedded Computing Systems* (2017).

[21] H. Chiang, J. Hong, K. Kiningham, L. Riliskis, P. Levis, and M. Horowitz. 2018. Tethys: Collecting Sensor Data without Infrastracture or Trust. In *Proceedings of the 3rd IEEE/ACM International Conference on Internet-of-Things Design and Implementation (IoTDI)*.

[22] A. Colin and B. Lucia. 2016. Chain: Tasks and Channels for Reliable Intermittent Programs. In *Proceedings of the ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*.

[23] A. Colin, E. Ruppel, and B. Lucia. 2018. A Reconfigurable Energy Storage Architecture for Energy-Harvesting Devices. In *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*.

[24] P. Corke, P. Valencia, P. Sikka, T. Wark, and L. Overs. 2007. Long-Duration Solar-Powered Wireless Sensor Networks. In *Proceedings of the 4th Workshop on Embedded Networked Sensors (EMNETS)*.

[25] Datasheet. [n.d.]. ChipCon 1101. Retrieved July 10th, 2020 from https://www.ti.com/lit/ds/symlink/cc1101.pdf

[26] J. de Winkel, C. Delle Donne, K. S. Yildirim, P. Pawelczak, and J. Hester. 2020. Reliable Timekeeping for Intermittent Computing. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*.

[27] P. Dutta, M. Grimmer, A. Arora, S. Bibyk, and D. Culler. 2005. Design of a Wireless Sensor Network Platform for Detecting Rare, Random, and Ephemeral Events. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN)*.

[28] P. Dutta, J. Hui, J. Jeong, S. Kim, C. Sharp, J. Taneja, G. Tolle, K. Whitehouse, and D. Culler. 2006. Trio: enabling sustainable and scalable outdoor wireless sensor network deployments. In *Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN)*.

[29] Istituto Nazionale Geofisica e Vulcanologia. [n.d.]. Earthquake Data in Italy. Retrieved July 10th, 2020 from http://cnt.rm.ingv.it

[30] ReVibe Energy. [n.d.]. modelD Piezoelectric Energy Harvester. Retrieved July 8th, 2020 from https://revibeenergy.com/modeld/

[31] ReVibe Energy. [n.d.]. modelQ Piezoelectric Energy Harvester. Retrieved July 8th, 2020 from https://revibeenergy.com/modelq/

[32] V. L. Erickson, S. Achleitner, and A. E. Cerpa. 2013. POEM: Power-Efficient Occupancy-Based Energy Management System. In *Proceedings of the 12th International Conference on Information Processing in Sensor Networks (IPSN)*.

[33] B. J. Fletcher, D. Balsamo, and G. V. Merrett. 2017. Power Neutral Performance Scaling for Energy Harvesting MP-SoCs. In *Proceedings of the Conference on Design, Automation & Test in Europe (DATE)*.

[34] F. Fraternali, B. Balaji, Y. Agarwal, L. Benini, and R. Gupta. 2018. Pible: Battery-Free Mote for Perpetual Indoor BLE Applications. In *Proceedings of the 5th Conference on Systems for Built Environments (BUILDSYS)*.

[35] M. Furlong, J. Hester, K. Storer, and J. Sorber. 2016. Realistic Simulation for Tiny Batteryless Sensors. In *Proceedings of the 4th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems (ENSSYS)*.

[36] A. Gomez, L. Sigrist, M. Magno, L. Benini, and L. Thiele. 2016. Dynamic Energy Burst Scaling for Transiently Powered Systems. In *Proceedings of the 2016 Conference on Design, Automation & Test in Europe (DATE)*.

[37] A. Gomez, L. Sigrist, T. Schalch, L. Benini, and L. Thiele. 2017. Efficient, Long-Term Logging of Rich Data Sensors Using Transient Sensor Nodes. *ACM Transactions on Embeddded Computing Systems* (2017).

[38] M. Guarducci. 2015. Ricordo della Magia in un Graffito del Mitreo del Circo Massimo. In *Mysteria Mithrae*. In Italian.

[39] J. Hester, T. Scott, and J. 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)*.

[40] J. Hester and J. Sorber. 2017. Flicker: Rapid Prototyping for the Batteryless Internet-of-Things. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems (SENSYS)*.

[41] J. Hester and J. Sorber. 2017. The Future of Sensing is Batteryless, Intermittent, and Awesome. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems (SENSYS)*.

[42] J. Hester, K. Storer, and J. Sorber. 2017. Timely Execution on Intermittently Powered Batteryless Sensors. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems (SENSYS)*.

[43] R. C. Hibbeler and T. Kiang. 2015. *Structural analysis*. Pearson Prentice Hall Upper Saddle River.

[44] T. W. Hnat, V. Srinivasan, J. Lu, T. I. Sookoor, R. Dawson, J. Stankovic, and K. Whitehouse. 2011. The Hitchhiker's Guide to Successful Residential Sensing Deployments. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems (SENSYS)*.

[45] N. Ikeda, R. Shigeta, J. Shiomi, and Y. Kawahara. 2020. Soil-Monitoring Sensor Powered by Temperature Difference between Air and Shallow Underground Soil. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)* (2020).

[46] N. Jackson, J. Adkins, and P. Dutta. 2019. Capacity over Capacitance for Reliable Energy Harvesting Sensors. In *Proceedings of the 18th International Conference on Information Processing in Sensor Networks (IPSN)*.

[47] H. Jayakumar, A. Raha, W. S. Lee, and V. Raghunathan. 2015. QuickRecall: A HW/SW Approach for Computing Across Power Cycles in Transiently Powered Computers. *ACM Journal on Emerging Technologies in Computing Systems* (2015).

[48] H. Jayakumar, A. Raha, J. R. Stevens, and V. Raghunathan. 2017. Energy-Aware Memory Mapping for Hybrid FRAM-SRAM MCUs in Intermittently-Powered IoT Devices. *ACM Transactions on Embedded Computing Systems* (2017).

[49] C. Kircher, A. Nassar, O. Kustu, and W. Holmes. 1997. Development of building damage functions for earthquake loss estimation. *Earthquake spectra* 13, 4 (1997).

[50] V. Kortbeek, K. S. Yildirim, A. Bakar, J. Sorber, J. Hester, and P. Pawelczak. 2020. Time-Sensitive Intermittent Computing Meets Legacy Software. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*.

[51] T. T. Lai, W. Chen, K. Li, P. Huang, and H. Chu. 2012. TriopusNet: Automating wireless sensor network deployment and replacement in pipeline monitoring. In *Proceedings of the 11th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*.

[52] H. N. Lechtman and L. W. Hobbs. 1987. Roman concrete and the Roman architectural revolution. In *High-Technology Ceramics: Past, Present, and Future-The Nature of Innovation and Change in Ceramic Technology*.

[53] E. A. Lee and S. A Seshia. 2016. *Introduction to embedded systems: A cyber-physical systems approach*. Mit Press.

[54] Libelium. [n.d.]. Waspmote. Retrieved July 10th, 2020 from http://www.libelium.com/products/waspmote/

[55] G. Loubet, A. Takacs, and D. Dragomirescu. 2019. Implementation of a Battery-Free Wireless Sensor for Cyber-Physical Systems Dedicated to Structural Health Monitoring Applications. *IEEE Access* (2019).

[56] B. Lubelli, R.P.J. Van Hees, and C.J.W.P. Groot. 2006. Sodium chloride crystallization in a salt-transporting restoration plaster. *Cement and concrete research* (2006).

[57] B. Lucia and B. Ransford. 2015. A Simpler, Safer Programming and Execution Model for Intermittent Systems. In *Proceedings of the 36th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*.

[58] G. Lukosevicius, A. R. Arreola, and A. S. Weddell. 2017. Using Sleep States to Maximize the Active Time of Transient Computing Systems. In *Proceedings of the ACM International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems (ENSSYS)*.

[59] K. Maeng, A. Colin, and B. Lucia. 2017. Alpaca: Intermittent Execution Without Checkpoints. *Proceedings of the ACM Programming Languages* (2017).

[60] K. Maeng and B. Lucia. 2018. Adaptive dynamic checkpointing for safe efficient intermittent computing. In *Proceedings of the 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*.

[61] K. Maeng and B. Lucia. 2019. Supporting Peripherals in Intermittent Systems with Just-in-Time Checkpoints. In *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI) (PLDI)*.

[62] A. Y. Majid, C. Delle Donne, K. Maeng, A. Colin, K. S. Yildirim, B. Lucia, and P. Pawelczak. 2020. Dynamic Task-Based Intermittent Execution for Energy-Harvesting Devices. *ACM Transactions on Sensor Networks* (2020).

[63] R. Marfievici, P. Corbalán, D. Rojas, A. McGibney, S. Rea, and D. Pesch. 2017. Tales from the C130 Horror Room: A Wireless Sensor Network Story in a Data Center. In *Proceedings of the First ACM International Workshop on the Engineering of Reliable, Robust, and Secure Embedded Wireless Sensing Systems (FAILSAFE)*.

[64] P. Martin, Z. Charbiwala, and M. Srivastava. 2012. DoubleDip: Leveraging Thermoelectric Harvesting for Low Power Monitoring of Sporadic Water Use. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems (SENSYS)*.

[65] G. V. Merrett and B. M. Al-Hashimi. 2017. Energy-Driven Computing: Rethinking the Design of Energy Harvesting Systems. In *Proceedings of the Conference on Design, Automation & Test in Europe (DATE)*.

[66] L. Mottola, G. P. Picco, M. Ceriotti, S. Guna, and A. L. Murphy. 2010. Not All Wireless Sensor Networks Are Created Equal: A Comparative Study on Tunnels. *ACM Transactions on Sensor Networks* (2010).

[67] F. E. Murphy, E. Popovici, P. Whelan, and M. Magno. 2015. Development of an heterogeneous wireless sensor network for instrumentation and analysis of beehives. In *Proceedings of the IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*.

[68] M. Navarro, T. W. Davis, Y. Liang, and X. Liang. 2013. A study of long-term WSN deployment for environmental monitoring. In *Proceedings of the 24th IEEE Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*.

[69] S. Peng and C. P. Low. 2012. Throughput optimal energy neutral management for energy harvesting wireless sensor networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*.

[70] A. I. Petrariu, A. Lavric, and E. Coca. 2019. Renewable Energy Powered LoRa-based IoT Multi Sensor Node. In *Proceedings of the 25th IEEE International Symposium for Design and Technology in Electronic Packaging (SIITME)*.

[71] Piezo.com. [n.d.]. Q220-H4BR-2513YB piezoelectric bending transducer. Retrieved July 8th, 2020 from https://piezo.com/products/piezoelectric-bending-transducer-q220-h4br-2513yb

[72] B. Ransford, J. Sorber, and K. Fu. 2011. Mementos: System Support for Long-running Computation on RFID-scale Devices. *ACM SIGARCH Computer Architecture News* (2011).

[73] A. Rodriguez, D. Balsamo, Z. Luo, S. P. Beeby, G. V. Merrett, and A. S. Weddell. 2017. Intermittently-powered energy harvesting step counter for fitness tracking. In *Proceedings of the IEEE Sensors Applications Symposium (SAS)*.

[74] E. Ruppel and B. Lucia. 2019. Transactional Concurrency Control for Intermittent, Energy-harvesting Computing Systems. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*.

[75] M. M. Sandhu, K. Geissdoerfer, S. Khalifa, R. Jurdak, M. Portmann, and B. Kusy. 2020. Towards Optimal Kinetic Energy Harvesting for the Batteryless IoT. *arXiv preprint arXiv:2002.08887* (2020).

[76] N. Saoda and B. Campbell. 2019. No Batteries Needed: Providing Physical Context with Energy-Harvesting Beacons. In *Proceedings of the 7th International Workshop on Energy Harvesting & Energy-Neutral Sensing Systems (ENSSYS)*.

[77] U. Senkans, D. Balsamo, T. D. Verykios, and G. V. Merrett. 2017. Applications of Energy-Driven Computing: A Transiently-Powered Wireless Cycle Computer. In *Proceedings of the 5th ACM International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems (ENSSYS)*.

[78] V. Sharma, U. Mukherji, V. Joseph, and S. Gupta. 2010. Optimal energy management policies for energy harvesting sensor nodes. *IEEE Transactions on Wireless Communications* (2010).

[79] L. Spadaro, M. Magno, and L. Benini. 2016. Poster Abstract: KinetiSee - A Perpetual Wearable Camera Acquisition System with a Kinetic Harvester. In *Proceedings of the 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*.

[80] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, and D. Culler. 2004. An Analysis of a Large Scale Habitat Monitoring Application. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SENSYS)*.

[81] C. Tavolieri and P. Ciafardoni. 2010. Mithra. Un viaggio dall'Oriente a Roma: l'esempio del Mitreo del Circo Massimo. *Archaeology Archives, BA* (2010). In Italian.

[82] Thermalforce. [n.d.]. 254-150-36 TEG. Retrieved July 10th, 2020 from https://www.dropbox.com/s/4xx1z2gwddntc42/TG254-150-36l.pdf?dl=0

[83] M. Thielen, L. Sigrist, M. Magno, C. Hierold, and L. Benini. 2017. Human body heat for powering wearable devices: From thermal energy to application. *Energy conversion and management* (2017).

[84] UNESCO. [n.d.]. Heritage Site Rome. Retrieved July 10th, 2020 from https://whc.unesco.org/en/list/91/

[85] J. Van Der Woude and M. Hicks. 2016. Intermittent Computation Without Hardware Support or Programmer Intervention. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation (OSDI)*.

[86] Y. Wang. 2008. Topology Control for Wireless Sensor Networks. In *Wireless sensor networks and applications*. Springer.

[87] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh. 2006. Fidelity and Yield in a Volcano Monitoring Sensor Network. In *Proceedings of the Symposium on Operating Systems Design and Implementation (OSDI)*.

[88] Wikipedia. [n.d.]. COVID-19 pandemic lockdown in Italy. Retrieved July 10th, 2020 from https://en.wikipedia.org/wiki/COVID-19_pandemic_lockdown_in_Italy

[89] K. S. Yildirim, A. Y. Majid, D. Patoukas, K. Schaper, P. Pawelczak, and J. Hester. 2018. InK: Reactive Kernel for Tiny Batteryless Sensors. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems (SENSYS)*.

[90] J. Zhang, C. Chen, X. Zhang, and S. Liu. 2016. Study on the environmental risk assessment of batteries. *Procedia Environmental Sciences* (2016).