Self-Sustainable Cyber-Physical Systems with Collaborative Intermittent Computing

Gaosheng Liu
Vrije Universiteit Amsterdam
g.s.liu@vu.nl

Lin Wang
Vrije Universiteit Amsterdam and TU Darmstadt
lin.wang@vu.nl

ABSTRACT

Cyber-physical systems have become a main technology driver for our intelligent society. However, almost all cyber-physical systems rely on battery-powered devices to function, which suffer from high maintenance cost for recharging/replacing the batteries and bring in negative environmental impacts due to the hazardous chemicals used in the batteries. To address this challenge, a new computing paradigm called intermittent computing (IC) was proposed which advocates a battery-free design where cyber-physical devices can be completely powered by energy scavenged from ambient sources such as sunlight, radio waves, and vibrations. Since its advent, many efforts have been made on addressing the challenges in IC, from the hardware to the software stack. In this vision paper, we make a brief summary of existing works on IC and discuss a more realistic setup where, instead of focusing on one IC node as done in most existing works, we propose to build a self-sustainable cyber-physical system through the collaboration of distributed IC devices—collaborative intermittent computing (CIC). We discuss the challenges in CIC and provide a vision for the future cyber-physical systems.

CCS CONCEPTS

• **Computer systems organization → Embedded and cyber-physical systems**; **Dependable and fault-tolerant systems and networks**.

KEYWORDS

Intermittent computing, cyber-physical systems, energy harvesting, low-power sensing and computing

ACM Reference Format:
Gaosheng Liu and Lin Wang. 2021. Self-Sustainable Cyber-Physical Systems with Collaborative Intermittent Computing. In *The Twelfth ACM International Conference on Future Energy Systems (e-Energy '21), June 28-July 2, 2021, Virtual Event, Italy*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3447555.3465324

1 INTRODUCTION

Cyber-physical system has shown its great potential in enabling an intelligent society with applications in numerous domains, including environment monitoring [36], healthcare [17, 31, 37], smart buildings [44], and even earth observation in Outerspace [5]. In general, cyber-physical applications follow a paradigm called sensing-processing-actuation [15]. More specifically, these applications deploy sensors to collect information further processed by a processing unit on-site or in a remote cloud. Based on the processing results, embedded devices perform actuation to react to the sensed information. In large-scale cyber-physical system deployments, sensors/actuators typically disperse geographically, powered by batteries.

With the rapid adoption of the cyber-physical system in many domains, it starts to show clear limitations due to its dependence on the battery [8]. The reason is mainly three-fold: (1) Batteries are typically bulky, hampering sensor miniaturization and increasing deployment difficulties of cyber-physical systems. In most cyber-physical scenarios, portability is the key to adoption, which is bottlenecked by the battery's size. (2) Batteries need to be recharged when dead and replaced when worn out, leading to escalated maintenance costs. In large-scale cyber-physical system deployment, it takes a considerable effort to access all the cyber-physical devices once deployed, e.g., sensors for monitoring large infrastructures—like buildings or bridges—built with concrete. (3) Batteries contain hazardous chemicals, and using batteries in every cyber-physical system deployment ubiquitously leads to adverse environmental effects.

Researchers have tried to leverage energy harvesting technologies to overcome these battery-induced limitations, i.e., scavenging electricity from ambient energy sources such as sunlight, electromagnetic waves, heat, or vibrations [32]. The main advantage of energy harvesting is that the considered energy sources are clean and cost-efficient. Cyber-physical devices collect energy from the aforementioned ambient sources and store the energy in a capacitor with energy harvesting. Once the stored energy in the capacitor reaches a certain threshold, the cyber-physical device will be woken up and operate using the stored energy. As a result, the collected energy drains gradually. When the collected energy is exhausted, the operation pauses, and the capacitor starts recharging. Without the battery's involvement, cyber-physical devices can last indefinitely, except for hardware failures in theory.

One of the main challenges in cyber-physical systems based on energy harvesting is intermittency, i.e., ambient energy sources can only provide limited electricity and are not always stable. As described above, cyber-physical devices are only active when enough energy has been scavenged and are completely off when the energy is exhausted before the capacitor is recharged to a sufficient level. Nevertheless, these cyber-physical devices are expected to perform normal sensing-processing-actuation activities. This phenomenon leads to the so-called "Sisyphean" phenomenon, resulting in a set of unique challenges to the cyber-physical system design.

e-Energy '21, June 28-July 2, 2021, Virtual Event, Italy
© 2021 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-8333-2/21/06.
https://doi.org/10.1145/3447555.3465324

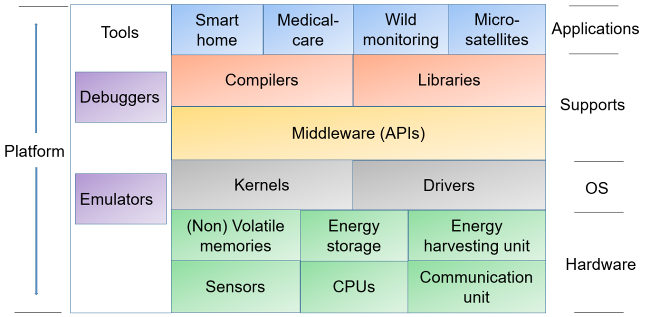

Figure 1: Overview of the intermittent computing landscape.

The cyber-physical paradigm under such intermittency is typically referred to as "intermittent computing (IC)" where computing on cyber-physical devices happens intermittently to complete a given cyber-physical task [6, 7, 25, 35]. IC devices are fundamentally different from traditional wireless sensor networks since they typically rely on unreliable ambient energy to function. Intermittent computing has been studied in various aspects, ranging from hardware [26], over software [23, 45, 49], to applications. However, existing efforts mainly focus on a single cyber-physical device that fits only limited scenarios. We argue that "intermittent computing" is a crucial enabler to building future *self-sustainable* cyber-physical systems. Such self-sustainable cyber-physical systems, once deployed, can run indefinitely without the need for any human maintenance and are completely environment-friendly.

To this end, we propose the concept of collaborative intermittent computing (CIC) for cyber-physical systems. CIC builds on top of recent advancements in intermittent computing, including support from hardware, operating systems, programming languages (PL), and runtime environments. Most importantly, CIC goes beyond a single cyber-physical device and advocates a *distributed system* design where multiple intermittent cyber-physical devices collaborate to perform non-interruptive, reliable computations transparently. Such a distributed system requires coordinating the energy-harvesting, computation, and communication activities on all the involved cyber-physical devices by reconciling the duty cycles of these devices. To the best of our knowledge, this is still an open research challenge.

In this vision paper, we present a systematic study of intermittent computing (Section 2) and discuss the main open challenges towards the CIC's full vision (Section 3).

2 STATE-OF-THE-ART

Recently, intermittent computing has attracted tremendous interest. So far, most of the existing works focus on hardware and software support for a single intermittent cyber-physical device. A high-level overview of the current landscape of intermittent computing is shown in Figure 1. In the following, we summarize these developments across the landscape.

2.1 Hardware

The hardware of an intermittent computing (IC) system, which is essential to support an IC system's functionalities with energy efficiency, typically includes energy-harvesting devices, microcontroller units (MCUs) with memory and clocks, communication units, and other peripherals like sensors. Energy-harvesting devices are responsible for energy collection and storage for the IC system, which then supply power to the whole system. For example, Flicker is a batteryless platform that supports energy harvesting from multiple energy sources including radio waves, sunlight, and kinesis [22]. Energy storage in IC systems is typically achieved by employing capacitors. Instead of using fixed-size capacitors, Capybara provides an interface for programmers to reconfigure the hardware energy capacitor to reconcile the energy demand of capacity- and temporally-constrained tasks in the same application [14].

MCUs provide computing capabilities for coordination and data processing in the IC system and they are more energy-efficient than regular CPUs in general. To mitigate the limited processing capability of MCUs, some IC systems, e.g., SONIC [19], are also equipped with hardware accelerators for performing complex computations like deep neural network (DNN) inference. MCUs used in IC systems need to be equipped with non-volatile memories like ferroelectric random access memories (FRAMs) to persist system states. While being more energy-efficient, FRAMs typically have lower access speeds than SRAMs, leading to worse performance. Therefore, IC systems typically adopt a hybrid design by combining FRAMs and SRAMs to balance energy efficiency and performance [26]. MANIC further optimizes memory efficiency by leveraging caches thus reducing the time and energy used for memory accesses [20]. Another critical hardware challenge for IC systems is robust timekeeping during power outages, which is essential for tasks such as synchronization and real-time operations. In an intermittent environment, on-chip digital timers are not capable of surviving power outages. Special designs are needed, e.g., CHRT proposes a multi-tier timekeeper architecture by employing a series of coupled capacitor-resistor circuits [16].

Communication units are required for an IC system to exchange information and perform coordination with others IC systems or platforms. However, communication is an energy-consuming task with radio signals. Alternative approaches include ambient backscatter which modulates ambient electromagnetic signals to encode information for communication [33] and visible light communication which leverages visible light instead of electromagnetic signals for communication. Depending on the application scenario, IC systems can also be equipped with other peripherals like temperature/humidity sensors. One critical challenge in supporting these peripherals is to retrieve the state of these peripherals after power outages and ensure state consistency. Samoyed targets this problem and introduces Just-In-Time checkpointing for the execution of peripheral operations [39]. Overall, hardware support for IC systems has been extensively explored in the literature and this lays a solid foundation for our vision of CIC—building a self-sustainable distributed computing platform out of IC systems.

2.2 Operating Systems

The operating system (OS) serves as a bridge between the hardware and application software. The OS for an IC system is responsible for providing functionalities including memory management, power management, task scheduling, and providing APIs for application development. Many of these tasks are similar to the ones supported by other wireless sensor OSes such as TinyOS and Contiki, but the intermittent nature of IC systems brings some additional challenges which manifest in the following aspects: (1) Memory management for IC systems should handle the memory allocation for applications on both the volatile and non-volatile memories at runtime. The coordination between the two types of memories is particularly important for avoiding data inconsistency and loss during power failures. DINO [34] provides a management solution for IC systems that aims to guarantee memory consistency between the two types of memories. Momentos enables automatic, energy-aware state checkpointing to non-volatile memories for transiently powered devices such as RFID tags [45]. SONIC targets deep neural network inference and proposes a new technique called loop continuation which can guarantee program correctness at power failures with low overhead for inference workloads. (2) To support peripherals, the OS of an IC system needs to preserve the state of these peripherals in non-volatile memories as done in Sytare [9]. The state can be recovered later when enough energy has been harvested and the activities on the peripherals can continue as if there were no interruptions [3, 11]. (3) Another important feature that the OS of an IC system has to support is to ensure responsiveness to time-sensitive tasks. This is typically achieved via new programming models and runtime, which will be further discussed in the next section [40, 51]. So far, all the existing OS solutions are focused on single IC nodes. OS support for CIC in a distributed setting is limited.

2.3 Programming Environments and Tools

Ideally, IC systems should be able to run legacy code developed for battery-powered sensor systems directly, simplifying the development and deployment costs. This requires support from programming languages, programming models, compilers, and libraries. The de facto programming language for IC systems is the C language (and its variants) mainly due to its support for low-level operations and high efficiency. For example, Mayfly is a language and runtime built on top of embedded-C for timely execution of sensing code in IC environments [23]. BFree provides a power failure resilience version of Python to simplify the programming of IC systems [27]. New programming models for IC systems have also been explored, e.g., to support timely execution of time-sensitive code. InK introduces an event-driven programming model for IC and proposes a reactive runtime system featuring preemptive scheduling to ensure power failure resilience [51]. Coati adopts a task-based interface for synchronous computations and an event-based interface for asynchronous interrupts [46]. CatNap is a similar system that uses a new programming model to allow programmers to flag partial code as time-critical and reserves energy for the reliable execution of such code while deferring the rest of the code [40]. To ensure forward progress and correctness of program execution, it is critical to recovering the program state (including registers and memory space) after power failures. This is typically done with the compilers and there are generally two approaches: (1) *checkpointing-based* where the compiler can automatically insert checkpoints in the program [10, 28, 38, 49] and (2) *task-based* where the compiler automatically divides the executable code into idempotent tasks that are free from consistency concerns [34]. Various reusable libraries have also been developed for simplifying the development of IC applications. For example, Chain [13] provides libraries for data encryption and compression. An analysis of the theoretical foundation for IC systems can be found in [47].

Development tools, like debuggers and emulators, have also been explored for IC systems. For example, EKHO [21] supports to simulate realistic energy conditions, which helps researchers or developers observe system behaviors under controlled environments. SIREN [18] simulates the runtime energy conditions at the instruction level, which is useful for debugging the code. An ideal IC system debugging tool, like Colin [12], should be able to support the check-off program error status without affecting the energy state of the IC system. ScEpTic adopts program analysis to find bugs and hidden anomalies in IC programs [41, 42]. Nevertheless, these tools are usually still limited to specific conditions, e.g., specific OS and libraries.

2.4 Applications

The application landscape of IC systems is overall limited so far. The following applications have been considered in the literature: communication, monitoring, medical care, and some modern machine learning-based application like voice recognition. Battery-free cellphones harvest energy from light and radio frequency signals and enable cellular communications [48]. IC systems have also been used in monitoring the health condition of buildings or other infrastructures. One example is the application for monitoring UNESCO-protected underground archaeological sites with IC devices collecting energy from thermal and kinetic sources [1]. With implanted IC devices that harvest energy from body movement or body thermal, the health condition of patients can be monitored [31]. Modern applications such as voice control have also been explored which leverage machine learning inference [19, 28, 50]. So far, almost all these applications are limited to one single IC device, without considering the collaboration among multiple IC devices. This can be attributed to the fact that communication primitives for IC systems are not mature yet. We will discuss further details in the next section.

3 VISION

In this section, we present our vision for future cyber-physical systems—achieving self-sustainability based on collaborative intermittent computing (CIC). Figure 2 provides an overview of CIC. We discuss the challenges in achieving the CIC vision from three aspects: hardware, systems software, and applications. These aspects need to be fully integrated in order to build a self-sustainable cyber-physical system.

3.1 Hardware

While tremendous efforts have been made in hardware support for IC systems, current IC systems are still capability limited with respect to both computing and communication. To support more (sophisticated) applications, it is desirable to have more powerful yet energy-efficient MCUs and memories, as well as capacitors with higher capacities to preserve more energy. Given the same physical situation, more energy-efficient MCUs can last longer, thus providing more opportunities for reliable computing in a CIC

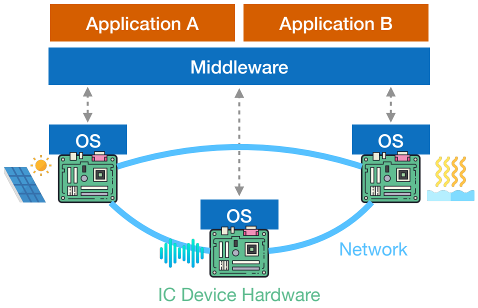

Figure 2: An overview of a self-sustainable cyber-physical system based on collaborative intermittent computing.

system. We expect more developments will be carried out in these areas in the coming years so that more capable IC devices can be built with modern hardware. Eventually, these IC devices are powerful enough for us to build a distributed CIC system on top of them. Yet, it is important to answer the question: How to make the CIC software system be able to adapt to the continuous evolvement of hardware devices?

3.2 Systems Software

The essential support for CIC comes from the systems software including both the OS and distributed middleware coordinating a set of IC devices. On top of the CIC middleware, cyber-physical applications can be built without the concern of intermittency in the system.

Operating system. Apart from all the support required for single IC devices as discussed in Section 2.2, nodes in a CIC system have also to deal with inter-node communication so that multiple nodes can exchange information and collaborate on sensing or computing. However, achieving efficient communication is challenging due to the fact that communication is an expensive task on IC devices. In addition, current support for communication in IC systems is rather limited where the implementation of a power failure resilient networking stack on IC systems is still completely missing. In a CIC system, communication can happen with different paradigms including unicast, multicast, or even broadcast, each of which has different energy requirements. The CIC system will need to figure out the best communication paradigm for each communication task and make a decision at runtime based also on the energy status on the IC device. There could also be different communication media available on IC devices, e.g., Bluetooth, visible light, or even backscatter [33]. Each of them provides different tradeoffs between performance and energy efficiency. The research questions include: (1) How to achieve reliable communication among IC devices? (2) How to select the most energy-efficient communication paradigm for a target application?

CIC middleware. With communication and networking enabled on individual IC devices, we can push for a middleware on top of a set of IC devices so that we can build a CIC distributed systems. This middleware takes care of coordinating the IC devices within such a system and ensuring the correct execution of distributed cyber-physical applications deployed on such a system. The IC devices in such a system may have different roles—some are responsible for sensing, while others are responsible for storing or processing the data. It can also be that all the IC devices play the same role and the CIC system achieves better reliability by leveraging these devices collaboratively. A recent work has explored asynchronous duty cycles for reliable sensing with multiple IC devices, but unlike our CIC vision this is done without any proactive coordination of the IC devices [43]. On top of the CIC middleware, high-level distributed programming models are required to program such a CIC system. The following research questions need to be examined carefully: (1) How to achieve reliable computing with unreliable IC devices? (2) What programming models are suitable for programming cyber-physical applications in the CIC environment?

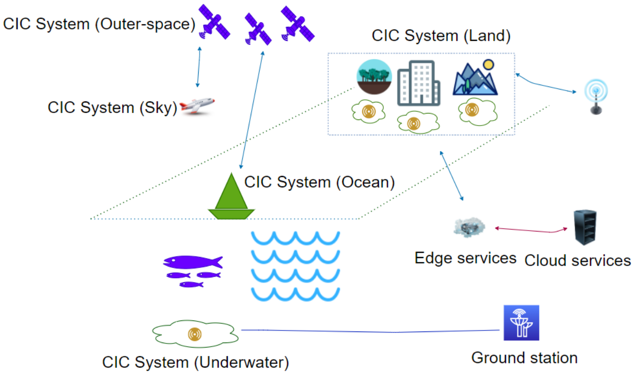

Figure 3: An overview of CIC applications covering underwater, ocean, land, sky, and outer space scenarios.

Security. A durable system must guarantee data privacy and security. In wild environments, data is easier to be spoofed, erased, and modified [2, 24, 29]. It is thus desirable to find a way to apply modern security technologies to CIC systems since data consistency and confidentiality are the premises of deploying sensitive applications. The main challenge is that security measures are typically expensive to implement on IC devices [4] while understanding the new attack tactics in the CIC system is also not easy. The following questions need to be explored: (1) What are the possible security vulnerabilities in CIC systems? (2) How to develop lightweight mechanisms to mitigate possible attacks in a CIC system?

3.3 Applications

Overall, CIC has a promising application perspective. Figure 3 depicts an overview of potential CIC applications, covering the following scenarios: underwater, ocean, land, sky, and outer space. There are application-specific challenges that come from the environmental condition of the application. For example, it is difficult to collect energy and communicate in an underwater CIC system since water has a negative effect on the propagation of electromagnetic waves due to increased attenuation. Furthermore, CIC systems may fulfill the following functionalities: learning and inference, interconnection to other systems, and interaction with humans.

Future IC systems require every IC device to have learning and inference abilities. To this end, machine learning models need to be deployed on IC devices. This requires us to solve the following main issues: (1) How to design lightweight machine learning (or deep learning) algorithms that can adapt to CIC environments. It is known that machine learning algorithms typically require intensive computation. Several attempts have been made towards this direction [19, 30, 50]. (2) How to store the training data and machine learning model with limited memory on IC devices? Machine learning algorithms need abundant data to train their model to achieve high accuracy. Achieving such training activities on a CIC system is nontrivial since the data and the machine learning model has to be stored on multiple IC devices to ensure availability, but this introduces issues on data consistency over the involved IC devices.

4 CONCLUSION

To summarize, this paper provides a brief survey on IC systems, illustrating the recent research progress in the IC field, the research challenges, and its significance in cyber-physical systems. This paper also depicts a vision of self-sustainable cyber-physical systems based on collaborative intermittent computing and discusses the challenges in achieving such a vision.

ACKNOWLEDGMENTS

This work has been partially funded by the German Research Foundation (DFG) grant 392046569. Gaosheng Liu is funded by the China Scholarship Council (CSC) fellowship.

REFERENCES

[1] 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 *SenSys '20: The 18th ACM Conference on Embedded Networked Sensor Systems, Virtual Event, Japan, November 16-19, 2020*, Jin Nakazawa and Polly Huang (Eds.). ACM, 368–381. https://doi.org/10.1145/3384419.3430722

[2] Omar Alfandi, Salam Ismail Rasheed Khanji, Liza Ahmad, and Asad Masood Khattak. 2021. A survey on boosting IoT security and privacy through blockchain. *Clust. Comput.* 24, 1, 37–55. https://doi.org/10.1007/s10586-020-03137-8

[3] Alberto Rodriguez Arreola, Domenico Balsamo, Geoff V. Merrett, and Alex S. Weddell. 2018. RESTOP: Retaining External Peripheral State in Intermittently-Powered Sensor Systems. *Sensors* 18, 1, 172. https://doi.org/10.3390/s18010172

[4] Hafiz Areeb Asad, Erik Henricus Wouters, Naveed Anwar Bhatti, Luca Mottola, and Thiemo Voigt. 2020. On Securing Persistent State in Intermittent Computing. In *Proceedings of the 8th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems (ENSsys '20)*. Association for Computing Machinery, New York, NY, USA, 8–14. https://doi.org/10.1145/3417308.3430267

[5] Justin Atchison and Mason Peck. 2007. A millimeter-scale lorentz-propelled spacecraft. In *AIAA Guidance, Navigation and Control Conference and Exhibit*. 6847.

[6] Domenico Balsamo, Alex S. Weddell, Anup Das, Alberto Rodriguez Arreola, Davide Brunelli, Bashir M. Al-Hashimi, Geoff V. Merrett, and Luca Benini. 2016. Hibernus++: A Self-Calibrating and Adaptive System for Transiently-Powered Embedded Devices. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 35, 12, 1968–1980. https://doi.org/10.1109/TCAD.2016.2547919

[7] Domenico Balsamo, Alex S. Weddell, Geoff V. Merrett, Bashir M. Al-Hashimi, Davide Brunelli, and Luca Benini. 2015. Hibernus: Sustaining Computation During Intermittent Supply for Energy-Harvesting Systems. *IEEE Embed. Syst. Lett.* 7, 1, 15–18. https://doi.org/10.1109/LES.2014.2371494

[8] Batteryproblems 2019. Overcoming the Battery Obstacle to Ubiquitous Sensing-Finally. https://everactive-media.s3.amazonaws.com/content/uploads/2019/06/17095844/Overcoming-the-Battery-Problem-to-Ubiquitous-Sensing_Everactive_June-2019.pdf

[9] Gautier Berthou, Tristan Delizy, Kevin Marquet, Tanguy Risset, and Guillaume Salagnac. 2019. Sytare: A Lightweight Kernel for NVRAM-Based Transiently-Powered Systems. *IEEE Trans. Computers* 68, 9, 1390–1403. https://doi.org/10.1109/TC.2018.2889080

[10] Naveed Anwar Bhatti and Luca 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 2017, Pittsburgh, PA, USA, April 18-21, 2017*, Pei Zhang, Prabal Dutta, and Guoliang Xing (Eds.). ACM, 209–219. https://doi.org/10.1145/3055031.3055082

[11] Adriano Branco, Luca Mottola, Muhammad Hamad Alizai, and Junaid Haroon Siddiqui. 2019. Intermittent asynchronous peripheral operations. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems, SenSys 2019, New York, NY, USA, November 10-13, 2019*, Raghu K. Ganti, Xiaofan Fred Jiang, Gian Pietro Picco, and Xia Zhou (Eds.). ACM, 55–67. https://doi.org/10.1145/3356250.3360033

[12] Alexei Colin, Graham Harvey, Alanson P. Sample, and Brandon Lucia. 2017. An Energy-Aware Debugger for Intermittently Powered Systems. *IEEE Micro* 37, 3, 116–125. https://doi.org/10.1109/MM.2017.48

[13] Alexei Colin and Brandon Lucia. 2016. Chain: tasks and channels for reliable intermittent programs. In *Proceedings of the 2016 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications, OOPSLA 2016, part of SPLASH 2016, Amsterdam, The Netherlands, October 30 - November 4, 2016*, Eelco Visser and Yannis Smaragdakis (Eds.). ACM, 514–530. https://doi.org/10.1145/2983990.2983995

[14] Alexei Colin, Emily Ruppel, and Brandon 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 2018, Williamsburg, VA, USA, March 24-28, 2018*, Xipeng Shen, James Tuck, Ricardo Bianchini, and Vivek Sarkar (Eds.). ACM, 767–781. https://doi.org/10.1145/3173162.3173210

[15] Roshan Bharath Das, Marc X. Makkes, Alexandru Uta, Lin Wang, and Henri E. Bal. 2019. Aves: A Decision Engine for Energy-efficient Stream Analytics across Low-power Devices. In *2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, December 9-12, 2019*. IEEE, 441–448. https://doi.org/10.1109/BigData47090.2019.9005607

[16] Jasper de Winkel, Carlo Delle Donne, Kasim Sinan Yildirim, Przemyslaw Pawelczak, and Josiah D. Hester. 2020. Reliable Timekeeping for Intermittent Computing. In *ASPLOS '20: Architectural Support for Programming Languages and Operating Systems, Lausanne, Switzerland, March 16-20, 2020*, James R. Larus, Luis Ceze, and Karin Strauss (Eds.). ACM, 53–67. https://doi.org/10.1145/3373376.3378464

[17] Saul Rodriguez Duenas, Stig Ollmar, Muhammad Waqar, and Ana Rusu. 2016. A Batteryless Sensor ASIC for Implantable Bio-Impedance Applications. *IEEE Trans. Biomed. Circuits Syst.* 10, 3, 533–544. https://doi.org/10.1109/TBCAS.2015.2456242

[18] Matthew Furlong, Josiah Hester, Kevin Storer, and Jacob Sorber. 2016. Realistic simulation for tiny batteryless sensors. In *Proceedings of the 4th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*. Association for Computing Machinery, 23–26. https://doi.org/10.1145/2996884.2996889

[19] Graham Gobieski, Brandon Lucia, and Nathan Beckmann. 2019. Intelligence Beyond the Edge: Inference on Intermittent Embedded Systems. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2019, Providence, RI, USA, April 13-17, 2019*, Iris Bahar, Maurice Herlihy, Emmett Witchel, and Alvin R. Lebeck (Eds.). ACM, 199–213. https://doi.org/10.1145/3297858.3304011

[20] Graham Gobieski, Amolak Nagi, Nathan Serafin, Mehmet Meric Isgenc, Nathan Beckmann, and Brandon Lucia. 2019. MANIC: A Vector-Dataflow Architecture for Ultra-Low-Power Embedded Systems. In *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture, MICRO 2019, Columbus, OH, USA, October 12-16, 2019*. ACM, 670–684. https://doi.org/10.1145/3352460.3358277

[21] Josiah D. Hester, Timothy Scott, and Jacob Sorber. 2014. Ekho: realistic and repeatable experimentation for tiny energy-harvesting sensors. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems, SenSys '14, Memphis, Tennessee, USA, November 3-6, 2014*, Ákos Lédeczi, Prabal Dutta, and Chenyang Lu (Eds.). ACM, 1–15. https://doi.org/10.1145/2668332.2668336

[22] Josiah D. Hester and Jacob Sorber. 2017. Flicker: Rapid Prototyping for the Batteryless Internet-of-Things. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems, SenSys 2017, Delft, Netherlands, November 06-08, 2017*, M. Rasit Eskicioglu (Ed.). ACM, 19:1–19:13. https://doi.org/10.1145/3131672.3131674

[23] Josiah D. Hester, Kevin M. Storer, and Jacob Sorber. 2017. Timely Execution on Intermittently Powered Batteryless Sensors. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems, SenSys 2017, Delft, Netherlands, November 06-08, 2017*, M. Rasit Eskicioglu (Ed.). ACM, 17:1–17:13. https://doi.org/10.1145/3131672.3131673

[24] Niccolò Izzo, Alessandro Barenghi, Luca Breveglieri, Gerardo Pelosi, and Paolo Amato. 2019. A secure and authenticated host-to-memory communication interface. In *Proceedings of the 16th ACM International Conference on Computing Frontiers, CF 2019, Alghero, Italy, April 30 - May 2, 2019*, Francesca Palumbo, Michela Becchi, Martin Schulz, and Kento Sato (Eds.). ACM, 386–391. https://doi.org/10.1145/3310273.3323401

[25] Hrishikesh Jayakumar, Arnab Raha, and Vijay Raghunathan. 2014. QUICKRECALL: A Low Overhead HW/SW Approach for Enabling Computations across

Power Cycles in Transiently Powered Computers. In *2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems, Mumbai, India, January 5-9, 2014.* IEEE Computer Society, 330–335. https://doi.org/10.1109/VLSID.2014.63

[26] Hrishikesh Jayakumar, Arnab Raha, Jacob R. Stevens, and Vijay Raghunathan. 2017. Energy-Aware Memory Mapping for Hybrid FRAM-SRAM MCUs in Intermittently-Powered IoT Devices. *ACM Trans. Embed. Comput. Syst.* 16, 3, 65:1–65:23. https://doi.org/10.1145/2983628

[27] Vito Kortbeek, Abu Bakar, Stefany Cruz, Kasim Sinan Yildirim, Przemyslaw Pawelczak, and Josiah D. Hester. 2020. BFree: Enabling Battery-free Sensor Prototyping with Python. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 4, 135:1–135:39. https://doi.org/10.1145/3432191

[28] Vito Kortbeek, Kasim Sinan Yildirim, Abu Bakar, Jacob Sorber, Josiah D. Hester, and Przemyslaw Pawelczak. 2020. Time-sensitive Intermittent Computing Meets Legacy Software. In *ASPLOS '20: Architectural Support for Programming Languages and Operating Systems, Lausanne, Switzerland, March 16-20, 2020*, James R. Larus, Luis Ceze, and Karin Strauss (Eds.). ACM, 85–99. https://doi.org/10.1145/3373376.3378476

[29] Archanaa S. Krishnan and Patrick Schaumont. 2018. Exploiting Security Vulnerabilities in Intermittent Computing. In *Security, Privacy, and Applied Cryptography Engineering - 8th International Conference, SPACE 2018, Kanpur, India, December 15-19, 2018, Proceedings (Lecture Notes in Computer Science)*, Anupam Chattopadhyay, Chester Rebeiro, and Yuval Yarom (Eds.), Vol. 11348. Springer, 104–124. https://doi.org/10.1007/978-3-030-05072-6_7

[30] Seulki Lee, Bashima Islam, Yubo Luo, and Shahriar Nirjon. 2019. Intermittent Learning: On-Device Machine Learning on Intermittently Powered System. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 4, 141:1–141:30. https://doi.org/10.1145/3369837

[31] Yoonmyung Lee, Gyouho Kim, Suyoung Bang, Yejoong Kim, Inhee Lee, Prabal Dutta, Dennis Sylvester, and David T. Blaauw. 2012. A modular 1mm^3 die-stacked sensing platform with optical communication and multi-modal energy harvesting. In *2012 IEEE International Solid-State Circuits Conference, ISSCC 2012, San Francisco, CA, USA, February 19-23, 2012.* IEEE, 402–404. https://doi.org/10.1109/ISSCC.2012.6177065

[32] Huicong Liu, Hailing Fu, Lining Sun, Chengkuo Lee, and Eric M Yeatman. [n.d.]. Hybrid energy harvesting technology: From materials, structural design, system integration to applications. *Renewable and Sustainable Energy Reviews*, 110473.

[33] Vincent Liu, Aaron N. Parks, Vamsi Talla, Shyamnath Gollakota, David Wetherall, and Joshua R. Smith. 2013. Ambient backscatter: wireless communication out of thin air. In *ACM SIGCOMM 2013 Conference, SIGCOMM 2013, Hong Kong, August 12-16, 2013*, Dah Ming Chiu, Jia Wang, Paul Barford, and Srinivasan Seshan (Eds.). ACM, 39–50. https://doi.org/10.1145/2486001.2486015

[34] Brandon Lucia and Benjamin 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, Portland, OR, USA, June 15-17, 2015*, David Grove and Stephen M. Blackburn (Eds.). ACM, 575–585. https://doi.org/10.1145/2737924.2737978

[35] Dong Ma, Guohao Lan, Mahbub Hassan, Wen Hu, and Sajal K. Das. 2020. Sensing, Computing, and Communications for Energy Harvesting IoTs: A Survey. *IEEE Commun. Surv. Tutorials* 22, 2, 1222–1250. https://doi.org/10.1109/COMST.2019.2962526

[36] Jun Ma, Hongzhi Yu, Yan Xu, and Kaiying Deng. 2020. CDAM: Conservative data analytical model for dynamic climate information evaluation using intelligent IoT environment - An application perspective. *Comput. Commun.* 150, 177–184. https://doi.org/10.1016/j.comcom.2019.11.014

[37] Yunfei Ma, Zhihong Luo, Christoph Steiger, Giovanni Traverso, and Fadel Adib. 2018. Enabling deep-tissue networking for miniature medical devices. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication, SIGCOMM 2018, Budapest, Hungary, August 20-25, 2018*, Sergey Gorinsky and János Tapolcai (Eds.). ACM, 417–431. https://doi.org/10.1145/3230543.3230566

[38] Kiwan Maeng and Brandon Lucia. 2018. Adaptive Dynamic Checkpointing for Safe Efficient Intermittent Computing. In *13th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2018, Carlsbad, CA, USA, October 8-10, 2018*, Andrea C. Arpaci-Dusseau and Geoff Voelker (Eds.). USENIX Association, 129–144. https://www.usenix.org/conference/osdi18/presentation/maeng

[39] Kiwan Maeng and Brandon Lucia. 2019. Supporting peripherals in intermittent systems with just-in-time checkpoints. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2019, Phoenix, AZ, USA, June 22-26, 2019*, Kathryn S. McKinley and Kathleen Fisher (Eds.). ACM, 1101–1116. https://doi.org/10.1145/3314221.3314613

[40] Kiwan Maeng and Brandon Lucia. 2020. Adaptive low-overhead scheduling for periodic and reactive intermittent execution. In *Proceedings of the 41st ACM SIGPLAN International Conference on Programming Language Design and Implementation, PLDI 2020, London, UK, June 15-20, 2020*, Alastair F. Donaldson and Emina Torlak (Eds.). ACM, 1005–1021. https://doi.org/10.1145/3385412.3385998

[41] Andrea Maioli, Luca Mottola, Muhammad Hamad Alizai, and Junaid Haroon Siddiqui. 2019. On intermittence bugs in the battery-less internet of things (WIP paper). In *Proceedings of the 20th ACM SIGPLAN/SIGBED International Conference on Languages, Compilers, and Tools for Embedded Systems, LCTES 2019, Phoenix, AZ, USA, June 23-23, 2019*, Jian-Jia Chen and Aviral Shrivastava (Eds.). ACM, 203–207. https://doi.org/10.1145/3316482.3326346

[42] Andrea Maioli, Luca Mottola, Muhammad Hamad Alizai, and Junaid Haroon Siddiqui. 2021. Discovering the Hidden Anomalies of Intermittent Computing. https://mottola.faculty.polimi.it/papers/maioli21discovering.pdf

[43] Amjad Yousef Majid, Patrick Schilder, and Koen Langendoen. 2020. Continuous Sensing on Intermittent Power. In *19th ACM/IEEE International Conference on Information Processing in Sensor Networks, IPSN 2020, Sydney, Australia, April 21-24, 2020.* IEEE, 181–192. https://doi.org/10.1109/IPSN48710.2020.00-36

[44] George M. Milis, Christos G. Panayiotou, and Marios M. Polycarpou. 2021. IoT-Enabled Automatic Synthesis of Distributed Feedback Control Schemes in Smart Buildings. *IEEE Internet Things J.* 8, 4, 2615–2626. https://doi.org/10.1109/JIOT.2020.3019662

[45] Benjamin Ransford, Jacob Sorber, and Kevin Fu. 2011. Mementos: system support for long-running computation on RFID-scale devices. In *Proceedings of the 16th International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2011, Newport Beach, CA, USA, March 5-11, 2011*, Rajiv Gupta and Todd C. Mowry (Eds.). ACM, 159–170. https://doi.org/10.1145/1950365.1950386

[46] Emily Ruppel and Brandon 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 2019, Phoenix, AZ, USA, June 22-26, 2019*, Kathryn S. McKinley and Kathleen Fisher (Eds.). ACM, 1085–1100. https://doi.org/10.1145/3314221.3314583

[47] Milijana Surbatovich, Brandon Lucia, and Limin Jia. 2020. Towards a formal foundation of intermittent computing. *Proc. ACM Program. Lang.* 4, OOPSLA, 163:1–163:31. https://doi.org/10.1145/3428231

[48] Vamsi Talla, Bryce Kellogg, Shyamnath Gollakota, and Joshua R. Smith. 2017. Battery-Free Cellphone. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 2, 25:1–25:20. https://doi.org/10.1145/3090090

[49] Joel van der Woude and Matthew Hicks. 2016. Intermittent Computation without Hardware Support or Programmer Intervention. In *12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016, Savannah, GA, USA, November 2-4, 2016*, Kimberly Keeton and Timothy Roscoe (Eds.). USENIX Association, 17–32. https://www.usenix.org/conference/osdi16/technical-sessions/presentation/vanderwoude

[50] Yawen Wu, Zhepeng Wang, Zhenge Jia, Yiyu Shi, and Jingtong Hu. 2020. Intermittent Inference with Nonuniformly Compressed Multi-Exit Neural Network for Energy Harvesting Powered Devices. In *57th ACM/IEEE Design Automation Conference, DAC 2020, San Francisco, CA, USA, July 20-24, 2020.* IEEE, 1–6. https://doi.org/10.1109/DAC18072.2020.9218526

[51] Kasim Sinan Yildirim, Amjad Yousef Majid, Dimitris Patoukas, Koen Schaper, Przemyslaw Pawelczak, and Josiah D. Hester. 2018. InK: Reactive Kernel for Tiny Batteryless Sensors. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems, SenSys 2018, Shenzhen, China, November 4-7, 2018*, Gowri Sankar Ramachandran and Bhaskar Krishnamachari (Eds.). ACM, 41–53. https://doi.org/10.1145/3274783.3274837