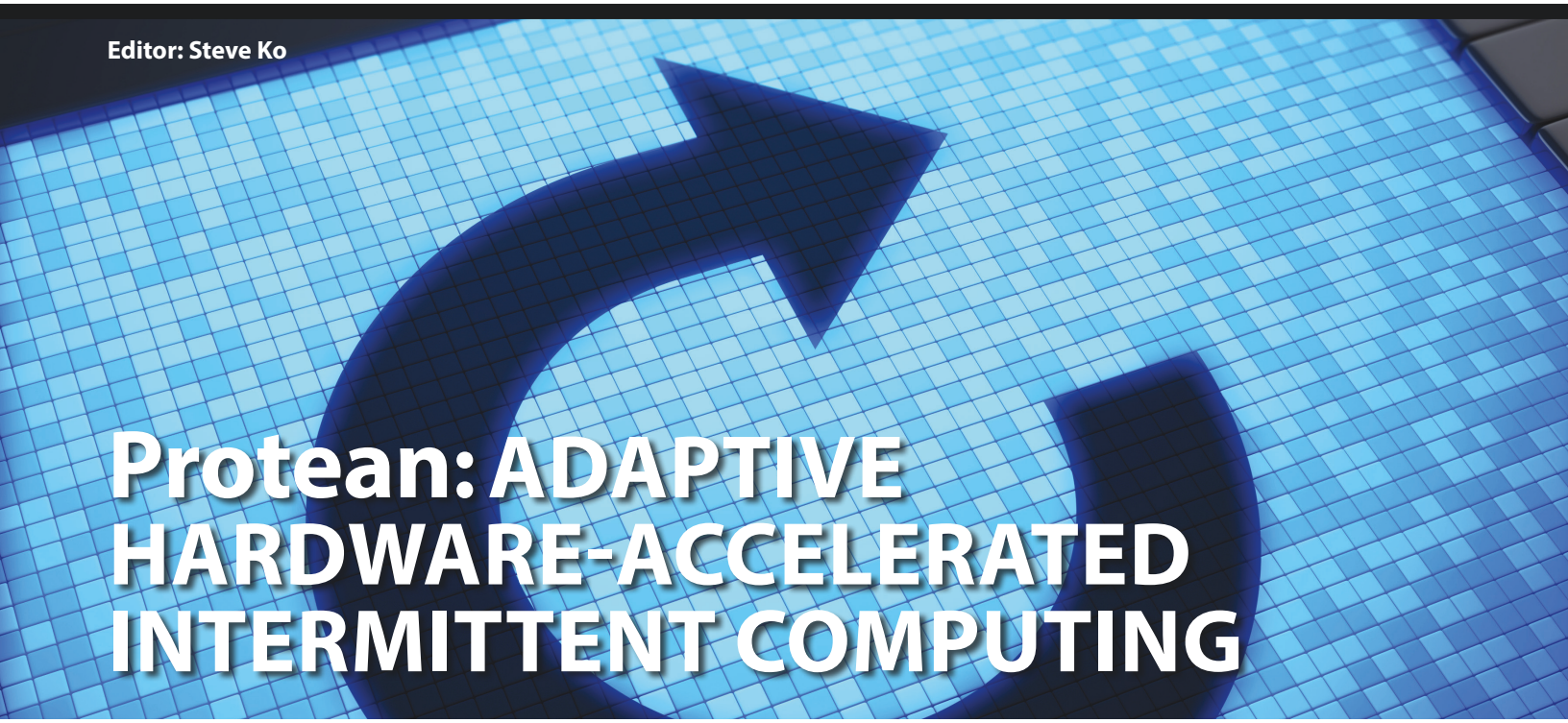
Abu Bakar, Rishabh Goel *Georgia Institute of Technology, Atlanta, GA*
Jasper de Winkel *Delft Delft University of Technology, Delft, Netherlands* **Jason Huang** *Northwestern University Evanston, IL*
Saad Ahmed *Georgia Institute of Technology, Atlanta, GA* **Bashima Islam** *Worcester Polytechnic Institute, Worcester, MA*
Przemysław Pawełczak *Delft University of Technology Delft, Netherlands* **Kasım Sinan** *Yıldırım University of Trento, Trento, Italy*
Josiah Hester *Georgia Institute of Technology, Atlanta, GA*

Editor: Steve Ko



Protean: ADAPTIVE HARDWARE-ACCELERATED INTERMITTENT COMPUTING

Excerpted from "Protean: An Energy-Efficient and Heterogeneous Platform for Adaptive and Hardware-Accelerated Battery-free Computing," from *SenSys 2022: Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems* with permission. https://dl.acm.org/doi/10.1145/3560905.3568561 ©ACM 2022

Today's smart devices have short battery lifetimes, high installation and maintenance costs, and rapid obsolescence — all leading to the explosion of electronic waste in the past two decades. These problems will worsen as the number of connected devices grows to one trillion by 2035. Energy harvesting, battery-free devices offer an alternative. Getting rid of the battery reduces e-waste, promises long lifetimes, and enables deployment in new applications and environments. Unfortunately, developing sophisticated inference-capable applications is still challenging. The lack of platform support for advanced (32-bit) microprocessors and specialized accelerators, which can execute data-intensive machine-learning tasks, has held back batteryless devices.

This article details the design of the *Protean* platform, which bridges the gap for inference-capable battery-free sensors. Protean includes a modular "plug-and-play" hardware design with a 32-bit ARM-based microcontroller with a convolutional neural network accelerator. An adaptive task-based runtime system provides intermittency-proof execution of machine learning tasks across heterogeneous processing elements. The runtime accounts for dynamic and intermittent power, automatically scaling and dispatching compute tasks based on incoming energy, current state, and programmer annotations. Protean is the first general-purpose, hardware-accelerated, adaptive battery-free platform, enabling new applications with data-intensive audio and visual workloads.

BATTERY-FREE DEVICES AND INTERMITTENT COMPUTING

For at least the past decade, researchers have explored battery-less, energy-harvesting computing devices as a sustainable alternative to a battery-powered Internet of Things. Ambient energy from sunlight, motion, thermal gradients, and even microbes is stored in capacitors to power computation, sensing, actuation, and communication. These battery-free devices compute intermittently due to the dynamic and unpredictable nature of available energy, causing power failures to occur multiple times a second, at which point volatile state (stack, registers, time) of programs is lost. Recovering gracefully and efficiently from those interruptions [2] has been the theme for a decade of intermittent computing research across the stack. These advances have yielded significant progress: batteryless devices have been shot into space, played Nintendo Game Boy games [3], programmed in Python [4], Rust, and

Illustration, istockphoto.com

JavaScript, and conducted simple vision tasks. However, expert programmers still find it challenging to quickly build useful things with these devices, while novices find them confounding. Furthermore, constrained and weak hardware makes machine learning or signal processing workloads challenging to execute. The most impressive demonstrations of intermittent computing mentioned above are all highly tuned, bespoke solutions that do not offer foundations for general approaches. The average batteryless devices are passive, low capability, unreliable, and less valuable for applications where data-intensive operations and inference require reactive, interactive, or highly dynamic systems.

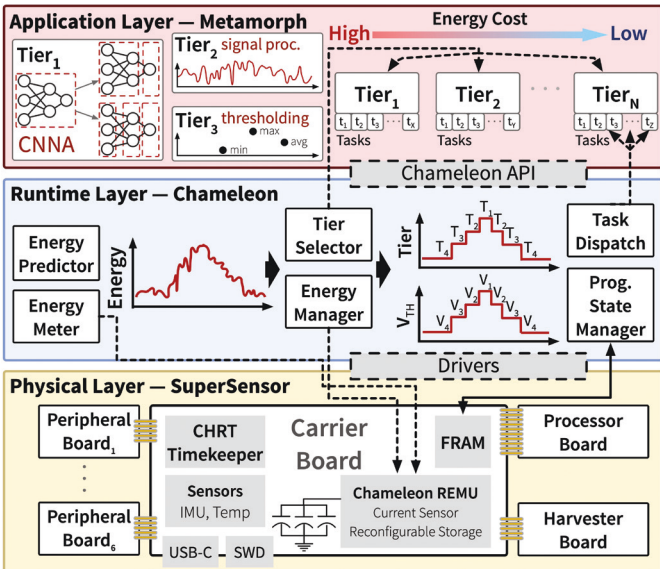


FIGURE 1. The three main components of Protean and how they interact to support adaptive accelerated applications.

What should a modern battery-free platform look like?

The maturity of the maker movement and stability of longtime hardware manufacturers AdaFruit and Sparkfun has led to standardization across hardware platforms in many ways, from communication protocols to interconnect, like Adafruit's Feather specification and Sparkfun's MicroMod platform, [5] and sensor breakouts. Beyond hobbyists, students, and makers, research labs and industry experts regularly rely on these vendors and their platforms to rapidly prototype high-performing inference-capable sensing applications. These community-supporting platforms are modular and standardized and include support for diverse sensors and computational resources. They provide a blueprint for any platform for intermittent computing and a backbone of components and tools for hardware prototyping. We distill four key requirements from this broader context and trends to guide platform development for intermittent computing.

1. Inference capable. Modern applications demand sophisticated hardware to conduct machine learning on-device. Machine learning in battery-free platforms in the past relied on application-tailored software

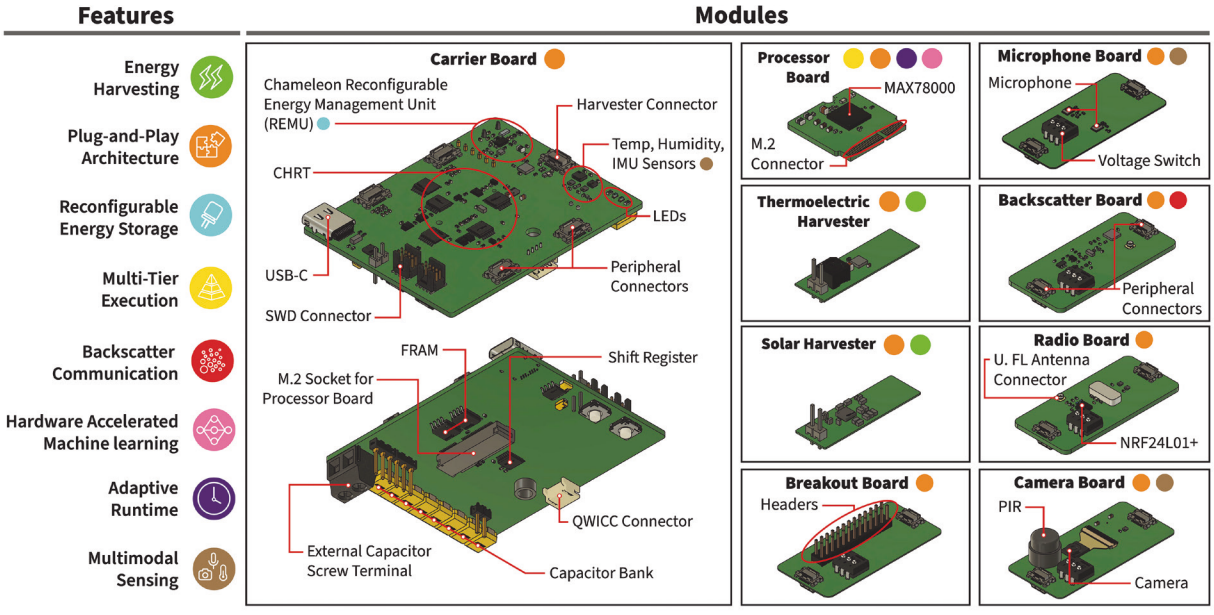


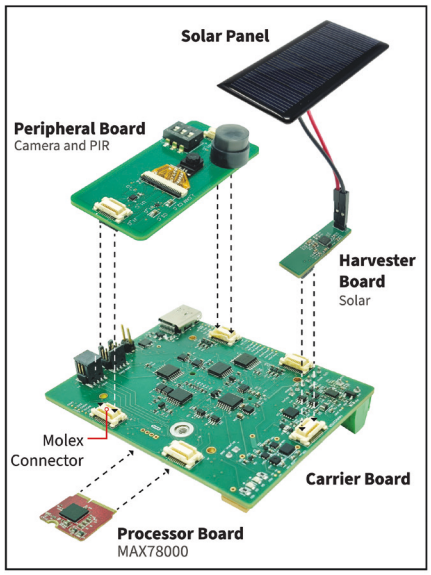
FIGURE 2. Overview of the capabilities, modules, and important features of the hardware/software platform. Interchangeable compute modules plug into the carrier board, and stacks of peripheral boards can be attached to a common bus to enable rapid prototyping for sophisticated batteryless devices.

support or custom hardware. These tailored solutions are not scalable to general-purpose applications.

2. Energy-Aware Adaptation and Scalability. Energy is dynamic and intermittent. Any platform must be able to do useful things when energy is scarce, as well as when it is abundant. From reconfigurable hardware to runtime systems, enabling malleable computation in the face of dynamic energy is key to success.

3. Modularity and Standardization. Expandability has been critical to the success of hardware platforms due to the high cost (in expertise, time, and money) of building hardware from scratch. In the past few years, major platforms have been introduced that embrace modularity and have broad maker industry support, including the Sparkfun MicroMod platform, which allows modularity up to and including the processor itself (via the standardized on the M.2 interconnect). While we cannot predict the future, this level of community buy-in and existing infrastructure makes us stronger in our belief that we can build devices and frameworks that can hang around for a longer time, supporting a research community.

Plug-and-Play Example



4. Programmability. Future platforms, like ours, must strongly consider the developers at the other end, who often struggle to compose a batteryless program. Simplifying the workflow is key to an effective platform.

Protean OVERVIEW

We design and build Protean around the four requirements established above, intended for developers who want to build inference-focused, adaptive, robust battery-free applications. Our goal is to (i) provide multiple hardware options in terms of computing modalities, peripherals, and harvesting technologies, (ii) enable energy-efficient inference applications, (iii) provide resilient runtime support for managing program state and memory across power failures, and (iv) allow rapid development and testing of different configurations of machine learning models.

We achieve these goals with a cross-stack approach (see Figure 1), building (i) SuperSensor, a modular hardware platform inspired by Sparkfun's MicroMod interconnect method, with a dynamically reconfigurable energy storage circuit, (ii) and Chameleon, an adaptive task-based runtime system that provides intermittency-proof execution of adaptive machine learning tasks across heterogeneous processing elements (in our prototype, a 32-bit ARM core, and a CNN accelerator). The runtime dynamically dispatches these tasks based on incoming energy and program state and arbitrates data movement for greater energy efficiency; and (iii) *Metamorph*, a code generator for transforming ML models developed in state-of-the-art frameworks (TensorFlow, PyTorch) into intermittence-safe C programs with little to no user intervention. The following sections present the high-level design of Protean's components (see Figures 1 and 2).

SuperSensor: Modular Platform Design

SuperSensor is a modular plug-and-play hardware design that supports four distinct modules — harvesters boards, sensors and radio peripheral boards, processors boards, and the carrier/main board. The boards, functions, and an example of how they work together are shown in Figure 2. The design is partly influenced by Sparkfun's MicroMod ecosystem, which separates carrier and processor boards. We discuss specific functionalities per board below.

Carrier Board. The brawn of SuperSensor, the intention of the carrier board is to fit all the absolute necessities for successful intermittent computing into one place, encompassing the lessons and designs of the last decade. Each of these functions must exist outside of the main processing unit. Those essential functions are checkpoint memory, energy management, timekeeping, debugging, and expansion interconnects. To allow for greater flexibility in developing adaptive runtimes, we include power measurement circuitry on the carrier board as part of the energy management unit. Finally, the Carrier board includes dedicated interconnects for all other boards, peripherals, processors, and harvesters. A block diagram is shown at the bottom of Figure 1.

Processor Board. The brain of SuperSensor consists of a microcontroller (MCU) and minimum supporting circuitry in a standardized M.2 connector for MicroMod compatibility. By separating the processor and carrier boards, SuperSensor remains agile to new developments in MCUs, allowing for upgrades and alternate builds without significant disruption to the ecosystem. We used the MAX78000, which consists of an ARM Cortex M4 core, RISC-V core, and a CNN accelerator. The processor board is programmed by the developer and hosts the runtime that maintains the forward progress and memory consistency of intermittently running applications. It manages peripheral control, energy, adaptation, etc.

Peripheral Boards connect to the carrier board to add functionality via sensors, actuators, radios, and other breakout modules. All peripheral boards use a common peripheral bus of our design that provides analog IOs, digital IOs, and digital bus lines, including QSPI, SPI, I2C, UART, I2S, and a parallel camera interface (PCIF). This shared bus supports the vast majority of available sensors and peripherals and enables SuperSensor to be used in various applications as almost all off-the-shelf sensing and communication components use one of the interfaces that our peripheral bus supports.

Harvester Boards harvest energy from different environmental sources. Many energy sources — solar, kinetic, vibration, radio frequency (RF), thermal, and microbial — all provide energy differently. Some provide direct current (DC), while others generate alternating current (AC), all at a variety of different voltages and currents. Finally, every harvester has different internal characteristics that require different circuitry to pull out maximum power in a particular context. The harvester boards are meant to support all these operations without requiring any change on the rest of the circuitry (carrier, processor, and peripheral boards.)

ADAPTIVE RECONFIGURABLE ENERGY STORAGE

Within SuperSensor, energy management and storage must be paid special attention, just as in previous platforms for intermittent computing. The key focus of the energy management system, called Reconfigurable Energy Management Unit (REMU), is to scale, as in to be dynamically configurable to provide energy for different tiers of execution, with tiers taking increasingly more energy (see top of Figure 1). The main challenge is in balancing capacitor size and program responsiveness. A bigger capacitor will keep the device operating for longer, but will also take a long time to charge (and even longer when ambient energy is scarce). In a scenario with a single static capacitor with a constant energy input, the capacitor needs to be able to sustain the system for the longest uninterruptible system task. For smaller uninterruptible tasks, the system still must wait until a full charge, introducing a penalty in the form of latency [6].

SuperSensor addresses this need with a novel reconfigurable energy storage architecture built around a single supercapacitor and single control unit. The architecture dynamically adapts the charging threshold of a single supercapacitor, effectively modifying the amount of energy stored in the system. Lowering the threshold when less energy is available increases responsiveness and increasing the threshold when more energy is available leads to a longer on-time.

Multi-tiered Tasks Runtime: Chameleon

SuperSensor provides a number of useful tools: energy monitoring and management, timekeeping, access to peripherals, and access to multiple computational elements. But hardware support alone cannot ensure the best configuration for the platform under variable energy conditions and unavoidable power failures. Moreover, supporting adaptation across heterogeneous computing platforms is nontrivial, as the same task might be drastically different depending on where it is executed. Put simply, a signal processing routine on an FPGA would be written in Verilog, a CNN would require trained weights, while an MCU would execute instructions. Furthermore, a CNN might host various implementations of the same routine that may trade off latency for performance. The central question is then: *how can a programmer design and manage tasks that can be dispatched by a runtime system to various computing elements, and at various quality levels, depending on available energy?*

We developed Chameleon to leverage the capability of the hardware for scalable, inference-focused intermittent computing. The core idea of Chameleon is to embrace scalability in hardware as well as software, providing a seamless way to degrade or upgrade tasks across diverse computational units. The basic idea is shown in Figure 1: as the rate of energy decreases (shown in the lower slope of the stored energy line), Chameleon changes the threshold for starting computation, and switches to a lower tier, which means switching the main inference task from being hosted on the CNN accelerator, to host on the MCU (in a lighter form) in this case.

Basically, a *tier* is a set of tasks that together form a control flow graph, whereas tasks are atomic code blocks that perform sensing, computations, communication, etc. Chameleon allows the programmer to write multiple tiers of the same application with lower computational complexity — which can execute on different computational units (e.g., MCU, accelerators, or both) — that can help maintain the latency and deadlines requirements under changing energy conditions. Each tier is a complete application with potentially different approaches to solving the same inference problem (i.e., deep learning, signal processing) and is computationally independent of all other tiers. A lower tier must require less energy than a higher tier. Chameleon's scheduler (tier selector) can automatically adapt to the best tier under given energy conditions.

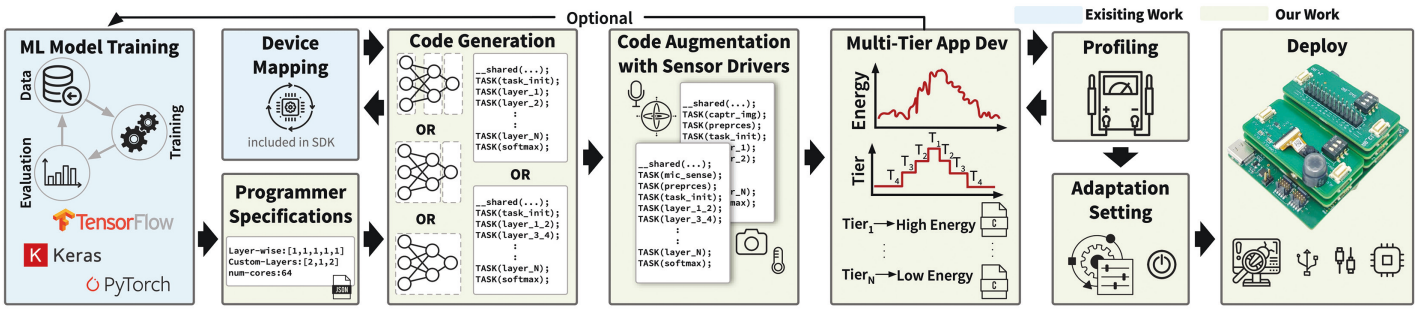


FIGURE 3. Typical workflow for programmers developing inference-focused applications with Protean. Models that are designed and trained using state-of-the-art frameworks are too computationally intensive to execute in one power cycle. Metamorph transforms large models into smaller chunks for execution across multi-tier compute elements with low programmer burden.

TABLE 1. Comparison of energy usage of Protean.

Dataset	Protean		State-of-the-art		Improvement
	# Classes	Energy (mJ)	# Classes	Energy (mJ)	
MNIST	10	0.06	10 [7]	40, 27	666.7×, 450×
CIFAR	10	2.12	2 [8]	17	8.02×
KWS	20	1.23	10 [9]	313	254.47×
Face ID	30	2.32	-	-	-

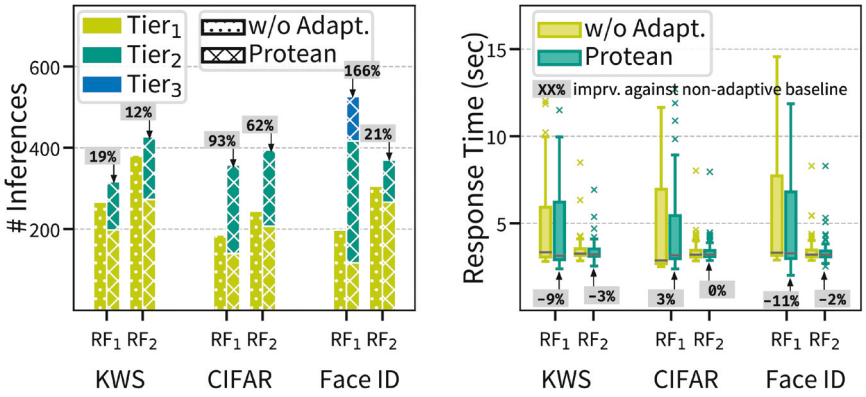


FIGURE 4. Protean has higher throughput with less recovery time when powered using real-world RF energy trace.

Threshold and tier selection is assisted by an energy prediction model, which leverages energy measurements and other heuristics for estimating current and future energy availability and choosing which tier to dispatch.

Metamorph: Intermittent-Safe Code Generation

Runtime systems cannot do everything and developing multiple tiers of a single application can be challenging. Using standard tools for TinyML in an intermittent computing workflow becomes challenging as these tools have no conception of how to persist state across power failures. Metamorph is a developer-facing code generation tool that bridges the gap between existing ML tools like PyTorch and TensorFlow, and intermittent computing. We built Metamorph, as the glue holding the runtime system (Chameleon) and heterogeneous hardware platform (SuperSensor) together. Metamorph wraps existing workflows to provide an automated way of generating intermittence-safe application code. Utilizing Metamorph, a developer can iteratively explore different design points of the application without dealing with underlying code generation frameworks or worrying about power failures.

Figure 3 shows a typical development workflow of Protean. It shows how a neural network developed in TensorsFlow and PyTorch can be converted to an intermittence-safe program that uses Chameleon's APIs through different programmer specified parameters using Metamorph.

EVALUATION

We evaluate Protean using three different acoustic and vision applications/benchmarks, each having three tiers that utilize different processing units on the SuperSensor. We use standard datasets to train the networks. For speech recognition, referred to as Keyword Spotting (KWS), we use a subset of Google's Speech Command dataset, where we have 20 different words. For image recognition, we used the CIFAR-10 dataset consisting of ten classes of 32×32 pixel color images of automobiles and animals. For face identification, referred to as Face ID, we use MaximCeleb, a dataset created by Maxim containing the faces of 30 celebrities. We test these benchmarks or RF energy traces collected under two different settings. With the dipole antenna, the power is variable because of external factors like people walking between transmitter and receiver (RF1), and with the patch antenna, the power is rather consistent with minor variability (RF2).

Protean is Energy-Efficient

We compare our Protean with state-of-the-art systems and show Protean performs better in terms of energy consumption when using the same benchmarks used by existing approaches. Table 1 lists the comparison. We observe that Protean outperforms the existing state-of-the-art system by showing an improvement of 666× for MNIST, 82× for the KWS application, and the trend holds for CIFAR. It must be noted here that this is the least we can achieve with our platform as the number of classes supported by existing systems are very small. With the higher number of classes, the complexity of the machine learning model increases, and so does its computational complexity and energy consumption at run-time. Due to Protean enabling us to use accelerators on intermittent power, making a one-to-one comparison with existing systems will always favor Protean in terms of performance (by multiple orders of magnitude). In a sense, this is important, as Protean is the first to allow access to these powerful computational resources for intermittent computing.

Protean has higher throughput with less recovery time

With dynamic voltage thresholding configured and tier-switching ON, Figure 4 reflects the throughput (number of inferences) and response time of Protean for two RF energy traces. We observe 12% to 166% improvement in inference throughput and up to 11% faster recovery from non-adaptive baselines across all applications. As SuperSensor supports dynamic turn-on voltage based on incoming energy, it facilitates tier adaptation to make the best use of the variable available energy. Higher variability in the energy results in more adaptation by Chameleon to ensure the system provides timely output.

The full paper [1] includes an in-depth evaluation of all the components of Protean and offers more insights into memory overhead, developer overhead, and adaptation based on multi-tier execution and dynamic thresholding.

CONCLUSION

This short article describes Protean, a unified platform for robust and adaptive execution of inference-driven applications on a battery-free platform. Protean contributes to the entire development stack with SuperSensor, a modular plug-and-play hardware; Chameleon, an adaptive task-based runtime system for heterogeneous intermittent systems; and Metamorph, a code generator to convert traditional CNN models to intermittent-safe task-based CNN models. SuperSensor pushes the boundary of battery-free computing systems by supporting more efficient ARM-based MCUs and CNN accelerators that achieve 666× more energy-efficient inferences than traditional battery-free platforms. Chameleon reduces the recovery time from power failure by 11% while achieving 166% higher throughput than non-adaptive counterparts in real-world settings. Finally, Metamorph speeds up the development process while reducing programmer burden. ■

Funding Acknowledgement

This research is based upon work supported by the National Science Foundation (NSF) under award numbers, CNS-2145584, CNS-2038853, and CNS-2030251. It was also supported by the Netherlands Organisation for Scientific Research (NWO), and partly funded by the Dutch Ministry of Economic Affairs (EZK), through TTW Perspective program ZERO (P15-06) within Project P1. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF, NWO, or EZK.

Abu Bakar is a PhD student at the School of Interactive Computing at Georgia Institute of Technology. His research interests are in low-power battery-free intermittent computing systems. He explores new hardware designs and runtime systems to make on-device inference with self-adaptation possible in extreme energy harvesting conditions. http://abubakar.info/.

Rishabh Goel is a PhD student at the School of Interactive Computing at Georgia Institute of Technology. His research interests are in energy harvesting, battery-free computing and robotics. He designs sustainable robots that can accomplish their mission without relying on batteries.

Jasper de Winkel is a PhD candidate at Delft University of Technology, and received his MSc (Cum Laude) in Embedded Systems from the same university. His research focuses on the intersection of batteryless devices and wireless networking, furthering the feasibility of networked batteryless devices through novel device architectures, ultra low power wireless networking and supporting software. https://jasperdewinkel.com/

Jason Huang is an undergraduate student studying computer science at Northwestern University. His research interests include low power embedded systems and their applications towards areas such as health monitoring. https://jjh9397.github.io/portfolio/

Saad Ahmed is a postdoctoral scholar at the School of Interactive Computing at Georgia Institute of Technology. His research interests include compilers, architecture, and runtimes for low-power batteryless embedded sensing devices that have applications in mobile health and education. He works towards developing system support for batteryless devices to ensure reliable application execution despite frequent power failures. https://www.saadahmedch.com

Bashima Islam is an assistant professor in the Department of Electrical and Computer Engineering and the Department of Computer Science at Worcester Polytechnic Institute. Her research interests are in machine learning for mobile and ubiquitous computing systems, and her speciality is on AI for low-power and batter-free computing systems. She received her PhD from University of North Carolina at Chapel Hill and was a visiting postdoctoral research associate at University of Illinois at Urbana-Champaign.

Kasım Sinan Yildirim is an associate professor at the Department of Information Engineering and Computer Science, University of Trento, Italy. He works on various topics on low-power and networked embedded systems, including intermittent computing, operating systems and runtimes, computer architectures, tiny machine learning, and low-power wireless protocols. http://sinanyil81.github.io/.

Przemysław Pawełczak is an associate professor at Delft University of Technology, leading Sustainable Systems Lab (https://github.com/tudssl). He completed his PhD at the same university in the area of Cognitive Radio. His current research focuses on low power embedded wireless systems, and batteryless systems in particular.

Josiah Hester is the Allchin Chair and associate professor of Interactive Computing and Computer Science at the Georgia Institute of Technology. His research interests span operating systems, architecture, and human-computer interaction, focusing on intermittent and ultra-low-power computing applied to health wearables, interactive devices, and large-scale sensing for sustainability and conservation. He works toward a sustainable future for computing informed by his Native Hawaiian heritage. http://josiahhester.com

REFERENCES

[1] Abu Bakar, Rishabh Goel, Jasper de Winkel, Jason Huang, Saad Ahmed, Bashima Islam, Przemysław Pawełczak, Kasım Sinan Yıldırım, Josiah Hester. 2022. Protean: An energy-efficient and heterogeneous platform for adaptive and hardware-accelerated battery-free computing. *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems.*

[2] Josiah Hester, Kevin Storer, and Jacob Sorber. Timely execution on intermittently powered batteryless sensors. 2017. *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, 1-13.

[3] Jasper De Winkel, Vito Kortbeek, Josiah Hester, and Przemysław Pawełczak. 2020. Battery-free game boy. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, no. 3, 1-34.

[4] Vito Kortbeek, Abu Bakar, Stefany Cruz, Kasım Sinan Yildirim, Przemysław Pawełczak, and Josiah Hester. Bfree: Enabling battery-free sensor prototyping with python. 2020. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4, no. 4, 1-39.

[5] Sparkfun. What is MicroMod? Retrieved Feb 03, 2023 from https://www.sparkfun.com/micromod

[6] Alexei Colin, Emily Ruppel, and Brandon Lucia. A reconfigurable energy storage architecture for energy-harvesting devices. 2018. *Proceedings of the 23rd International Conference on Architectural Support for Programming Languages and Operating Systems*, 767-781.

[7] Graham Gobieski, Brandon Lucia, and Nathan Beckmann. Intelligence beyond the edge: Inference on intermittent embedded systems. 2019. *Proceedings of the 24th International Conference on Architectural Support for Programming Languages and Operating Systems*, 199-213.

[8] Abu Bakar, Tousif Rahman, Alessandro Montanari, Jie Lei, Rishad Shafik, and Fahim Kawsar. 2022. Logic-based intelligence for batteryless sensors. *Proceedings of the 23rd Annual International Workshop on Mobile Computing Systems and Applications*, 22-28.

[9] Alessandro Montanari, Manuja Sharma, Dainius Jenkus, Mohammed Alloulah, Lorena Qendro, and Fahim Kawsar. 2020. ePerceptive: energy reactive embedded intelligence for batteryless sensors. *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, 382-394.