Realistic Simulation for Tiny Batteryless Sensors

Matthew Furlong, Josiah Hester, Kevin Storer, Jacob Sorber
School of Computing
Clemson University
{mfurlon, jhester, kstorer, jsorber}@clemson.edu

ABSTRACT

Battery-free sensing promises to revolutionize the scientific and industrial communities by enabling long term, maintenance free deployments in tough to reach places. However, developing applications for these intermittently-powered, batteryless devices is notoriously demanding. Each device's performance is closely tied to its environment at runtime, and developers are often unable to predict how their system will be behave upon deployment. In this paper we present an instruction level simulator based off of MSPsim for intermittently-powered devices that can accurately emulate real-world energy harvesting conditions, taking into account power models of common hardware peripherals like a radio, and accelerometer. These harvester conditions are represented by IV surfaces recorded by the Ekho hardware emulator We have provided this simulator as an open source tool for the benefit of the community.

CCS Concepts

•**Computer systems organization** → **Architectures; Embedded and cyber-physical systems;** •**Computing methodologies** → *Modeling and simulation;* •**Hardware** → *Power and energy;*

Keywords

Energy Harvesting, Ekho, MSPsim, Simulation

1. INTRODUCTION

In the near future, we expect to see a world covered with billions, or even trillions, of connected heterogeneous computing devices that sense and share data with each other, and people. This network of physical objects–the Internet of Things (IoT)–has inspired decades of computing research and will continue to motivate technological innovation for years to come.

Traditional sensing platforms have assumed the presence of a stable power supply. While the use of energy harvesters in these devices has increased, applications for these devices are predicated on the constraints of large batteries. In these systems harvested energy is viewed as surplus, not as the primary source of energy. Batteries size, weight, material costs, and environmental impact, limit their feasibility for sensing, especially at large scale. Furthermore, the manual replacement (and eventual recycling) of billions of dead batteries is not practical. These limitations are unlikely to change in the near future; improvements in battery technology have historically been gradual and are unable to keep pace with the advances within the sensing community.

Battery-free sensing has emerged as a promising path toward long-term sustainable sensing at massive scale. These devices can operate perpetually for years without the need for human intervention. However, using small energy stores, like capacitors, and volatile energy supplies present unique challenges when working with transiently-powered devices. Harvested ambient energy is unreliable, and computational tasks are often interrupted by power failures. Consequently, the operating behavior of energy-harvesting devices is difficult to predict

Software simulations allow application developers and research scientists to verify multiple schemes and applications for wireless sensor networks with reduced cost and in less time. While there are various simulators for sensor networks such as MSPsim[4] and TOSsim[6], currently there is no simulator for batteryless, intermittently-powered devices that simulates the actual energy harvesting environment from recorded traces. This is crucial to the developers of sensor networks who require the ability to obtain reliable power consumption figures to predict the runtime behavior of their programs before deployment[7].

The primary objective of this work is to provide the community a simulator for batteryless, intermittently-powered devices. SIREN (**S**imulation of **I**ntermittently-powered devices using **R**ealistic **EN**vironments) accurately simulates a realistic energy harvesting environment using energy harvesting environments recorded at the source by Ekho[5], powering FRAM enabled MSP430's, and low-power hardware peripherals including the CC1101, a common radio model.[1]

2. BACKGROUND

The behavior of transiently-powered devices is heavily dependent upon the ambient energy available to harvest in deployment. Additionally, the efficiency and amount of energy harvested by intermittently powered devices depends

ENSsys 2016 November 16, 2016, Stanford, CA, USA
© 2016 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-4532-3/16/11.
DOI: http://dx.doi.org/10.1145/2996884.2996889

[1]SIREN is open-source and can be found at: `https://github.com/PERSISTLab/BatterylessSim`

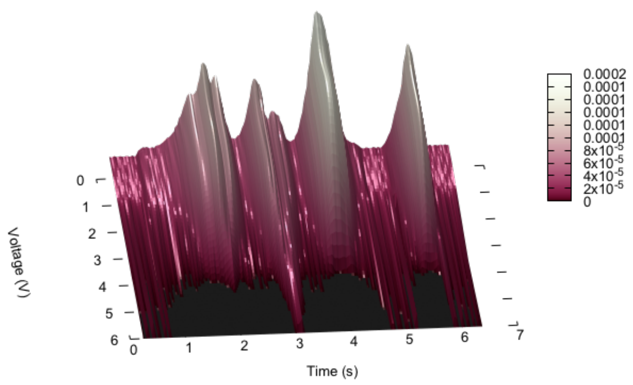


Figure 1: A solar IV surface generated from an IXYS solar cell exposed to a lightbox. An IV surface captures all possible harvesting scenarios for a harvester. Each possible harvesting current (I) for the supply voltage (V) over time are shown.

on the device's behavior, As such, it is difficult for developers and researchers to predict how their applications will perform prior to deployment.

Ekho[5] was designed to make the testing process more rigorous. Ekho is a tool that allows users to record, and then emulate energy harvesting conditions, in-lab, using the general abstraction of I-V surfaces. I-V surfaces are made up of I-V curves, which relate harvesting current (I) to the supply voltage (V). An I-V surface is a sequence of I-V curves over time that represents how an energy harvester will behave for every possible load. Different device behaviors (such as reading the temperature, broadcasting on the radio, or blinking an LED) will change the current draw of the batteryless device. This current (I) change will cause a change in supply voltage on the harvester, which changes the amount of energy harvested. Since I-V surfaces provide a digital representation of harvesting conditions, they provide a general energy harvester abstraction that can be used by a simulator. A solar I-V surface is shown in Figure 1.

Ekho and other hardware based approaches are an essential part of the batteryless sensor developer's toolbox. By emulating the environmental conditions of deployment scenarios, Ekho provides realistic insight into runtime device behavior that is otherwise unavailable. Using Ekho, developers can observe and adjust their applications appropriately, without waiting until deployment for feedback, saving both time and money. However, the real time constraints of Ekho and other real time hardware based solutions are not suitable for some types of testing. For some experimentation and validation, simulation is preferred.

We envision multiple use cases where energy environment simulation would be preferred to emulation and debugging in hardware: 1) understanding the performance and behavior of sensor networks requires simulation tools that can scale to very large numbers of nodes, 2) comparing different scheduling paradigms for a variety of task loads quickly and realistically, 3) comparing different harvesters, over multiple environmental conditions, for multiple firmwares quickly and easily. Essentially, an energy harvesting environment simulator allows for agile testing and refactoring of batteryless sensor programs. However, current simulators lack the ability to simulate realistic energy harvesting environments that were gathered at the source, like those recorded with Ekho. SIREN seeks to fill this technology gap.

3. SIREN

SIREN is based on MSPsim, an instruction level simulator for MSP430 processors that can run unmodified target platform firmware. MSPsim is written in Java, and was created as an extensible, instruction-level, timing accurate simulator that is easily configurable for new sensor boards and hardware peripherals. MSPsim accurately simulates most of the peripheral modules of the MSP430 chipset, including Timers, the Clock system, and IO peripherals. In this section we provide a system overview with implementation details, as well as common use cases for SIREN when working with batteryless, energy harvesting devices.

3.1 Siren Implementation

A system-level overview of SIREN is shown in Figure 2. SIREN is composed of five main parts, the Ekho surface manager, Ekho emulator, the Capacitor model, the instruction level simulator, and the peripheral simulator. Additionally, SIREN is complemented by an I-V surface generator, and custom graphing tools. We detail each of these below:

Ekho Surface Manager: The surface manager loads I-V surfaces from digital files stored on the host computer. These can be generated by an Ekho recorder, or the I-V surface generator. Multiple surfaces can be stored, allowing firmware to be tried on many different energy environments with minimal developer effort. The surface manager feeds I-V curves to the emulator for further processing.

Ekho Emulator: The emulator is a software implementation of the actual hardware Ekho emulator. It takes I-V curves from the surface manager, reads the harvesting current of the capacitor, and then attempts to set the harvester voltage.

Capacitor Model: The capacitor model is based off code from the simulator used to verify Mementos[8], also based off MSPsim. The capacitor model simulates the physics of an arbitrary sized capacitor and handles the behavior of a MSP430 rest and power cycle.

Instruction Level Simulator: The instruction level simulator is MSPsim with a few critical enhancements. We added support for modern processors, including FRAM enabled MSP430s. Additionally, MSPsim was augmented with code hooks that allow the firmware to interface with the simulator, but at no energy or time cost, giving a zero overhead printf and logging function. MSPsim was also modified to track supply voltage and operating current.

Peripheral Simulator: The peripheral simulator handles the interface between the MSP430 core, and hardware peripherals connected over GPIO pins, or the SPI / I2C bus. Importantly, the peripheral simulator manages the instantaneous power draw of each peripheral. SIREN can simulate the power draw and power states of the CC1101 radio, a very common transciever used in low power wireless networks. Two common hardware peripherals used in batteryless devices were added as I/O and energy models to MSPsim— the CC1101 low frequency radio transceiver, and the ADXL362,

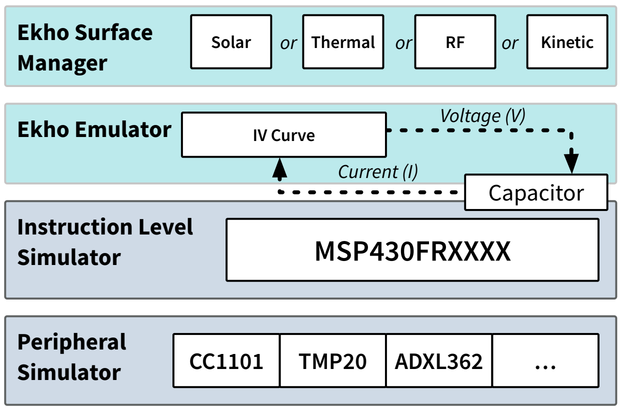


Figure 2: An overview of SIREN is shown. MSPsim was complemented with a capacitor model, an Ekho surface manager, and multiple IC implementations including the FRAM enabled MSP430FR6989 MCU, ADXl362 accelerometer, and CC1101 radio.

an ultra low power accelerometer.

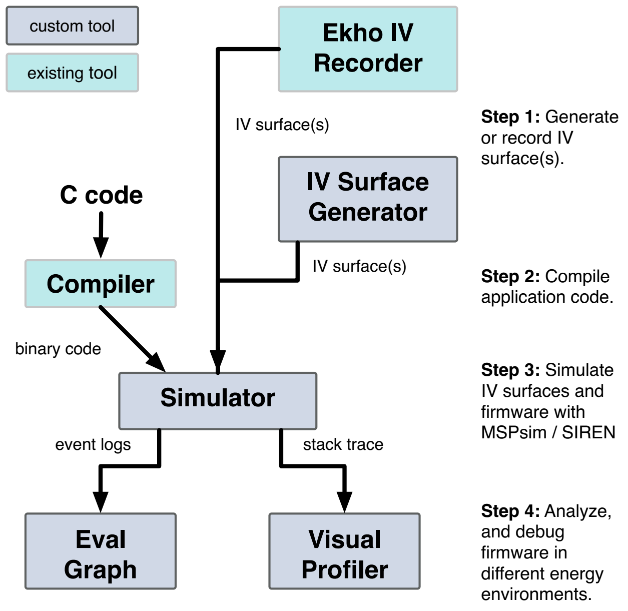
I-V Surface Generator: Often developers do not have access to an Ekho device, or the environment where a deployment will happen. For this reason we developed an I-V surface generator that allows developers to use datasheets and domain knowledge to estimate energy environments. The surface generator takes developer input to generate pseudo-random I-V surfaces for testing. The developer inputs (I,V) pairs defining an archetypal I-V curve, the length in seconds of the surface, and the time resolution. The generator uses 1-dimensional Perlin noise as a scaling factor for the archetypal I-V curve, creating a realistic looking, smoothly random I-V surface of any arbitrary length and resolution.

3.2 Debugging with SIREN

SIREN allows users to to easily test applications with its suite of debugging tools, including a built-in profiler non-invasive printf, and event metrics. The typical user workflow of SIREN is shown in Figure 3. SIREN uses a custom C function, `siren_command`, that writes to a section of memory outside the bounds of the MSP430's memory. Our custom function expects input to be a pre-defined event, such as "PRINTF:" or "GRAPH-EVENT:", followed by the programmer's statement. SIREN commands also allows users to gather metrics on how often programmer defined code events were encountered during a program. This enables the developer to identify which sections of the code are most important to the application and ensure that they are using the proper energy harvester to achieve the desired number of events. As events are encountered, they are written to a file that can be graphed using a provided script.

4. RELATED WORK

In response to the difficulty of working with transiently-powered devices, the community has developed tools to emulate energy- harvesting environments[5], model energy harvesters[10], and to enable energy-interference-free debugging[3]. These systems inform our work, however, they are orthogonal to system level simulation of a batteryless, MSP430



custom tool
existing tool
Ekho IV Recorder
IV surface(s)
Step 1: Generate or record IV surface(s).
C code
IV Surface Generator
IV surface(s)
Compiler
Step 2: Compile application code.
binary code
Simulator
Step 3: Simulate IV surfaces and firmware with MSPsim / SIREN
event logs
stack trace
Eval Graph
Visual Profiler
Step 4: Analyze, and debug firmware in different energy environments.

Figure 3: The typical user workflow with batteryless sensors and SIREN is shown. Users record or generate I-V surfaces, and input firmware into SIREN. SIREN then generates metrics with which to evaluate, test, and profile firmware(s) in multiple energy environments.

energy harvesting sensor device. In fact, we envision all of them being used with SIREN in the future.

The community has provided numerous simulators for microcontrollers that simplify the development and debugging process for battery-powered devices. MSPsim, a Java-based "instruction level simulator for the MSP430 microcontroller", allows users to test programs without specific hardware and provides configurable simulations of hardware peripherals. MSPsim was extended by including voltage traces that regulate power availability for testing Mementos[8], a tool for placing checkpoints into non-volatile memory. However, neither of these tools enable the simulation of realistic energy environments for batteryless sensing devices, specifically simulating how the amount of energy harvested is dependant on the behavior of the device.

Some simulators have been extended to specifically deal with multi-source energy harvesting. GreenCastalia [2], and others [9, 1] provide energy harvesting and super capacitor models for wireless sensor networks. Users can also provide an efficiency metric to simulate things like shadows or cloud cover. These approaches are not as realistic as using Ekho traces, that have been recorded from the actual physical energy harvester, instead of modeled. Moreover, no current simulator models the changes in harvesting efficiency caused by changes in application behavior.

5. DISCUSSION & FUTURE WORK

SIREN is designed to provide application developers a realistic energy environment with the added benefits of a simulator, such as non-invasive profiling, printf debugging, and code event metrics. Based on the results described in the previous section, SIREN assures users can accurately replicate energy conditions in simulation. However, our current implementation of SIREN is limited by a number of important factors.

Limitations: The majority of limitations stem from the I-V surfaces emulated within SIREN. If these I-V surfaces are not accurate representations of the energy environment during deployment, runtime behavior in simulation will not match behavior in deployment. While the supplied Surface Maker tool provides developers with the ability to create their own I-V surfaces, these surfaces are unlikely to match energy environments in deployment. With the Surface Maker tool, variations in I-V curves are created using perlin noise, but variations in the the physical environment can be more volatile. Without the proper I-V surfaces, SIREN can not properly simulate realistic energy environments.

Future Work: Our immediate future work includes a thorough evaluation of SIREN against real Ekho IV surfaces, comparing sensing behaviors, active time, sleep times, and dead times, across multiple applications. We also envision step debugging with GDB, code coverage, and program visualization in simulation. Code coverage metrics are commonly used to identify which areas of source code are never executed, and which are used most frequently. These measures allow developers to focus attention on critical sections and ensure sections of code are run as expected. Traditional methods for code coverage operate under the assumption that the system will not be interrupted by a power failure. However, to accurately provide code coverage metrics for an energy-harvesting device, developers must be able to gather this data while devices are subjected to realistic environmental changes.

6. CONCLUSIONS

In this paper, we have described the design of SIREN, an instruction level simulator for intermittently-powered devices that replays energy harvesting conditions recorded in the actual deployment environment with Ekho.[2] SIREN tracks power states of peripherals, such as a radio or accelerometer, and the MSP430 microcontroller, per instruction. SIREN additionally provides a collection of debugging tools that allows the developer access to energy aware code coverage and real-time visualization of program flow. SIREN enables developers to test new programming models, checkpointing techniques, and new energy harvesters on existing applications. We have provided SIREN as an open source tool for the benefit of the community.

Battery-free sensing has the capacity to dramatically transform the industrial and scientific communities by providing long term, maintenance-free deployments in difficult to access locations. While current tools for working with transient power ease the development process, they do not satisfy developers' every need. SIREN enhances toolchain support by supplying simulation tools for profiling, printf debugging, and code event metrics.

[2]SIREN is open-source and can be found at:
`https://github.com/PERSISTLab/BatterylessSim`

7. REFERENCES

[1] M. H. Alizai, Q. Raza, Y. Chandio, A. A. Syed, and T. M. Jadoon. Simulating intermittently powered embedded networks. In *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks*, pages 35–40. Junction Publishing, 2016.

[2] D. Benedetti, C. Petrioli, and D. Spenza. Greencastalia: an energy-harvesting-enabled framework for the castalia simulator. In *Proceedings of the 1st International Workshop on Energy Neutral Sensing Systems*, page 7. ACM, 2013.

[3] A. Colin, G. Harvey, B. Lucia, and A. P. Sample. An energy-interference-free hardware-software debugger for intermittent energy-harvesting systems. In *Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '16, pages 577–589, New York, NY, USA, 2016. ACM.

[4] J. Eriksson, A. Dunkels, N. Finne, F. Osterlind, and T. Voigt. Mspsim–an extensible simulator for msp430-equipped sensor boards. In *Proceedings of the European Conference on Wireless Sensor Networks (EWSN), Poster/Demo session.*

[5] J. Hester, T. Scott, and J. Sorber. Ekho: Realistic and repeatable experimentation for tiny energy-harvesting sensors. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, SenSys '14, pages 1–15, New York, NY, USA, 2014. ACM.

[6] P. Levis, N. Lee, M. Welsh, and D. Culler. Tossim: Accurate and scalable simulation of entire tinyos applications. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 126–137. ACM, 2003.

[7] E. Perla, A. Ó. Catháin, R. S. Carbajo, M. Huggard, and C. Mc Goldrick. Powertossim z: realistic energy modelling for wireless sensor network environments. In *Proceedings of the 3nd ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks*, pages 35–42. ACM, 2008.

[8] B. Ransford, J. Sorber, and K. Fu. Mementos: System support for long-running computation on rfid-scale devices. *SIGPLAN Not.*, 46(3):159–170, Mar. 2011.

[9] C. Tapparello, H. Ayatollahi, and W. Heinzelman. Energy harvesting framework for network simulator 3 (ns-3). In *Proceedings of the 2nd International Workshop on Energy Neutral Sensing Systems*, pages 37–42. ACM, 2014.

[10] N. F. Tinsley, S. T. Witts, J. M. Ansell, E. Barnes, S. M. Jenkins, D. Raveendran, G. V. Merrett, and A. S. Weddell. Enspect: A complete tool using modeling and real data to assist the design of energy harvesting systems. In *Proceedings of the 3rd International Workshop on Energy Harvesting &#38; Energy Neutral Sensing Systems*, ENSsys '15, pages 27–32, New York, NY, USA, 2015. ACM.