UC San Diego
UC San Diego Electronic Theses and Dissertations

Title

Intermittent Computing But Persistent State: Leveraging Low-Power Energy Harvesting and Extremely Efficient Electronics to Simplify State Management

Permalink

https://escholarship.org/uc/item/0vk8m45c

ISBN

9798241646187

Author

Marcano, Gabriel Eugenio

Publication Date

2026-04-10

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA SAN DIEGO

Intermittent Computing But Persistent State: Leveraging Low-Power Energy Harvesting and Extremely Efficient Electronics to Simplify State Management

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Computer Science (Computer Engineering)

by

Gabriel Eugenio Marcano

Committee in charge:

Professor Pat Pannuto, Chair
Professor Josiah Hester
Professor Ryan Kastner
Professor Patrick Mercier
Professor Tajana Rosing

2026

Copyright

Gabriel Eugenio Marcano, 2026

All rights reserved.

The Dissertation of Gabriel Eugenio Marcano is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2026

DEDICATION

To my wife, Elizabeth Jane Marcano, and mi hijita, Ana María Marcano.

EPIGRAPH

SAVING... DON'T TURN OFF THE POWER.

Save Text Box, Pokémon Crystal, Game Freak

Energy is the currency of the future.

CEO Nabbubu, Sid Meier's Alpha Centauri, Firaxis Games

TABLE OF CONTENTS

Dissertation Approval Page iii

Dedication iv

Epigraph v

Table of Contents vi

List of Figures x

List of Tables xiii

Acknowledgements xiv

Vita xvi

Abstract of the Dissertation xviii

Introduction 1
1.1 The Origin and Evolution of Intermittent Computing 3
1.1.1 Tracking Time for Intermittent Computing 5
1.1.2 Power Management 5
1.1.3 Software Runtimes 6
1.2 Key Challenges 7
1.3 Thesis 8
1.4 Insights 8
1.5 Key Contributions 9

Chapter 2 A Survey on Maximum Power Point Tracking Algorithms, Comparing Their Efficiency, Complexity, and Convergence Across Diverse Power Sources . . 11
2.1 Introduction 12
2.1.1 Background 13
2.1.2 Maximum Power Transfer Theorem 15
2.1.3 A Note on Efficiency 17
2.1.4 Switching Voltage Regulator Design 18
2.1.5 Characteristics of Energy Sources 21
2.1.6 MPPT Ontology 30
2.1.7 Functional Methods 31
2.1.8 Incremental Conductance 32
2.1.9 Open Circuit Voltage 34
2.1.10 Closed Circuit Current 34
2.1.11 Model-based 37
2.1.12 Computational Methods 37

2.1.13 Artificial Neural Network 38
2.1.14 Artificial Intelligence Algorithms 39
2.1.15 Other Algorithms 39
2.2 Hybrid Methods 39
2.3 MPPT Publication Sampling Methodology 40
2.4 Results and Discussion 40
2.4.1 Solar 42
2.4.2 Wind 43
2.4.3 Tidal 44
2.4.4 Hydroelectric 46
2.4.5 Thermoelectric 47
2.4.6 Radio Frequency 48
2.4.7 Piezoelectric 50
2.4.8 Proton-Exchange Membrane Fuel Cell 51
2.4.9 Microbial Fuel Cell 52
2.5 Research Trends and Future Work 53
2.6 Conclusion 54
2.7 Acknowledgments 54

Chapter 3 Artemia: Persistent Wall-Clock Time for Intermittent Computing Applications 56
3.1 Introduction 57
3.1.1 Contributions 60
3.2 Background & Related Works 60
3.2.1 Timekeeping for Intermittent Systems 61
3.2.2 Energy Harvesting and Management 62
3.3 Design 63
3.3.1 Design Requirements 63
3.3.2 Artemia Overall Design 65
3.3.3 Energy Harvesting Subsystem 66
3.3.4 Time Subsystem 66
3.3.5 Electrical Isolation Subsystem 67
3.3.6 Application Support Subsystem 68
3.3.7 External Applications 68
3.4 System Implementation 69
3.4.1 Component Selection 71
3.4.2 Test Software & Task Scheduling 73
3.4.3 Tasks 74
3.5 Evaluation 74
3.5.1 Artemia Quiescent Power Draw 76
3.5.2 Energy Utilization of Environmental Sensing Tasks 76
3.5.3 Full System Feasibility 77
3.6 Case Studies 79
3.6.1 Indoor Solar as a Constant Power Source 79

3.6.2 Enabling More Sampling Over Time 80
3.6.3 Enabling New Applications 81
3.7 Discussion 83
3.7.1 Electrical Isolation is Evolving 83
3.7.2 Extracting Energy from Low Power Sources is Difficult 84
3.7.3 Electronics and Software Implementation Improvements 84
3.7.4 Supercapacitors and High ESR 85
3.8 Conclusion 85
3.9 Acknowledgments 86

Chapter 4 Blinded by the MSP: Rethinking Intermittent Execution for the 32-bit, SoC Era and Beyond 87
4.1 Introduction 88
4.2 The Intermittent System Landscape 91
4.2.1 Case Study: ENGAGE 92
4.3 Modeling and Comparing Intermittent Policies 93
4.3.1 A Glance at Intermittent Computing 93
4.3.2 The Devil is In the Details 95
4.3.3 A Detail-Oriented Model 96
4.3.4 A Metric for Cross-Policy and Cross-Platform Comparisons 98
4.4 Modeling Results 99
4.4.1 Comparing Platforms to Themselves 100
4.4.2 Is the Latency Cost Worth It? 101
4.4.3 Cross-Platform Comparisons 101
4.5 Overcoming Model Limitations 102
4.5.1 System Power and Energy Simulator 102
4.6 Implementation 103
4.6.1 Microcontroller Platforms 103
4.6.2 Power Control 104
4.6.3 Emulating Low Input Power 105
4.6.4 Application Software 105
4.7 Evaluation 106
4.7.1 Comparing Policies On Hardware 106
4.8 Discussion 107
4.8.1 Intermittent Platforms & Policy Co-Evolution 107
4.8.2 Decoupling Capacitors Are Not Free 110
4.8.3 Metric Limitations 111
4.8.4 Future Optimizations 111
4.9 Conclusion 112
4.10 Acknowledgments 112

Chapter 5 Future Work & Conclusion 120
5.1 Enhancing Networking 120
5.1.1 Upgrading to a Class B Device 121

5.2 Designing a New Intermittent Computing Platform 122
5.2.1 Brine: An Early Design Exploration for a Modular Mainboard for Intermittent Computing 122
5.2.2 Revisiting Batteries: The Inadvertent 20+ Year Experiment 124
5.2.3 An Example Design for a Battery-backed Intermittent Computing System 126
5.3 Future Work 130
5.3.1 We Need to Rally Around an Operating System 130
5.3.2 Dealing with Clock Drift 131
5.3.3 We Need Better Energy Harvesters and a Better Way to Prototype Control Logic 133
5.4 Conclusion 134

Bibliography 135

LIST OF FIGURES

Figure 1.1. Architecture of the Prometheus system [91], with the major intermittent computing subsystems highlighted by different colors, as follows: Energy harvesting, energy store, power management, and application computing. . 3

Figure 2.1. Schematic of a boost converter. Reproduced from [55]. 19

Figure 2.2. NASA Nimbus switching voltage regulator design. This is a buck converter, used to reduce the source voltage. Reproduced from [115]. 19

Figure 2.3. Current vs. voltage and power vs. voltage curves of a solar arrays, with sample points showing the effect of load resistance on the operating point. Reproduced from [12]. 22

Figure 2.4. Wind generator speed vs. turbine power, showing the MPPs at a few different wind speeds. Reproduced from [1]. 23

Figure 2.5. Tidal power generation vs. generator angular velocity, given at different tide velocities. The MPP trend is highlighted by the dotted line. Reproduced from [58]. 24

Figure 2.6. Power surface given a fixed pressure head and turbine diameter. Reproduced from [72]. 25

Figure 2.7. Thermoelectric voltage vs current and power curves at different thermal gradients ($\Delta 100°C$, $\Delta 150°C$, $\Delta 200°C$). Reproduced from [121] 26

Figure 2.8. RF power vs voltage at specific amounts of signal power. Reproduced from [139]. 27

Figure 2.9. Piezoelectric power vs voltage at specific frequency, with curves at different vibration amplitudes. Reproduced from [112]. 28

Figure 2.10. PEMFC Power vs Membrane Water Content. Reproduced from [132]. . . . 29

Figure 2.11. MFC IV and power curves. Specifically, this is data from an MFC after recovering from a voltage overshoot condition. Reproduced from [5]. 30

Figure 2.12. P&O logic flow. Reproduced from [23]. 33

Figure 2.13. Flow diagram showing how the incremental conductance method operates. Reproduced from [85]. 35

Figure 2.14. Control flow for the OCV algorithm. 36

Figure 2.15. Sample fuzzy logic table, with outputs mapped based on E and CE inputs. Reproduced from [164]. 38

Figure 3.1. Overview of Artemia, showing the four main Artemia subsystems connected to an external application. 65

Figure 3.2. Picture of the complete Artemia prototype, configured for the experiment in section 3.5.3. 69

Figure 3.3. Power traces of the intermittent tasks evaluated for Artemia. 74

Figure 3.4. Example trace, from the 3.3V temperature task test, used to determine feasibility of running a task at a given starting voltage level. 75

Figure 3.5. Voltage traces captured over a 24 hour period of Artemia running our test application being powered completely from 4 sMFCs. 78

Figure 3.6. Voltage traces captured over a day and a half measuring the power dissipated through 271 kΩ resistors. A diurnal pattern is visible. The average voltage and power recorded at night are 1.8 and 12 µW, accordingly. 80

Figure 3.7. Excerpt of voltage traces captured over a 9 hour period of Artemia running our test application powered completely from a voltage power source set to 2 V through a 6.8 kΩ resistor as input to the ADP5091. 82

Figure 4.1. The voltage of the primary energy store across multiple intermittent computing execution cycles. 93

Figure 4.2. A single intermittent computing execution cycle. The violet is the charging period, the green is the boot or wake period, and the orange is the run period of the intermittent computing cycle. 94

Figure 4.3. Analytical results showing when one policy is better than the other, based on the input power and the energy store size. 113

Figure 4.4. Analytical results comparing the Apollo3 and MSP430FR5994 metrics based on their datasheet values. 114

Figure 4.5. Schematic of the testing setup. A 3.6 V source with a 12 kΩ resistor approximates a solar energy harvester in an office setting, providing approximately an average of 200 µW to the 100 µF energy store capacitor. 115

Figure 4.6. The testing setup for an Apollo3 microcontroller. 116

Figure 4.7. Picture of the modified RedBoard Artemis ATP. 116

Figure 4.8. Zoomed in portions of both the Apollo3 and MSP430FR5994 executing in both the power-off policy and the sleep policy, showing the difference in latency from one cycle to the next. 117

Figure 4.9. Zoomed in portion of the Apollo3 in the power-off policy to show the effect of decoupling capacitors on the main energy store at start-up time. 118

Figure 4.10. This shows the theoretical time over energy used per cycle for the Apollo3 and MSP430FR5994 (without performance). This shows that even in this scenario, where compute does not matter, it usually makes more sense to use the Apollo3. 119

Figure 5.1. Brine mainboard, a prototype for testing modular intermittent system configurations. 123

Figure 5.2. The populated PCB of a Super Nintendo Entertainment System Super Mario All-Stars game cartridge. 125

Figure 5.3. High level overview of the proposed battery-backed intermittent computing system. There are three power rails, high power for the more “traditional” energy harvesting sources, low power for trickle but constant energy harvesting sources, and then a backup primary battery. 127

Figure 5.4. The second version of Artemia, on its own custom PCB. This version of Artemia integrates all of the peripherals the original work tests, and adds a LoRa radio on a second SPI bus. 132

LIST OF TABLES

Table 1.1. Sampling of constant sources of low power. †Where only power estimates were available, we estimate the active area of the harvester as best as possible to give an estimate for power density. 2

Table 2.1. Categorization and comparison of the surveyed MPPT publications. 41

Table 3.1. Constant sources of low power. 64

Table 3.2. Major components used in our prototype implementation of Artemia, broken down by subsystem. 70

Table 4.1. Parameters from High-Impact Intermittent Systems Over Time. Where possible, empirical numbers are sourced from the original paper or their supporting artifacts. 91

Table 4.2. Microcontroller parameters extracted from datasheets. 99

Table 4.3. Major components used in our evaluation, broken down by subsystem. 104

Table 4.4. Metrics from the simulator and the experiments for both microcontrollers, under both policies. There is close agreement between the simulated metrics and the actual ones, with agreement that the sleep policy yields a higher metric. 106

Table 5.1. Describes the sleep states of the proposed intermittent system. S0, S1, and S2 require that the high power rail is good, S3 requires that the low power rail is good, S4 requires that the battery rail is good, and OFF is the effect of no rails being good. 128

Table 5.2. Describes the control signals and source of power for the different power domains based on the current state. 129

Table 5.3. State transition table for sleep state management. X means that that field does not matter. 129

ACKNOWLEDGEMENTS

I would like to acknowledge and thank my advisor Pat Pannuto for all of the guidance and help for these five and a half years. From figuring out soil microbial fuel cell energy harvesting, and pivoting to intermittent computing, Pat has provided a lot of support, sometimes to the point of overextending themselves. It has been a pleasure also to collaborate with someone that also sees the value of engineering in science. As chance would have it, us both also became parents within months of each other, so it has been a tremendous help to have an advisor that not only knows but is actively feeling the demands of parenthood on their time.

I want to thank Jen Switzer for the amazing opportunity to collaborate on her smartphone recycling project. She has been a fantastic collaborator, and is a very close friend of me and my family. Additionally, I have her project to thank for the doors it opened, as I would not be going to work for Google if not for the connections her project provided!

I also want to thank Olivia Weng and Andy Meza for their friendship and collaboration. Alexander Redding and Chris Crutchfield have been something of kindred spirits, and it has been a blast to geek out over old using and hacking old computer systems, including those three Super Famicoms we repaired, and talking about home servers and networking. I would like also to thank Alex Yen, Wenshan Luo, Alex Bellon, Zhenghua Ma, Aba Gnaneswaran, Raymond Duenas, Tyler Potyondy, Janet Vorobyeva, and all of the students in PATLab and in the Kastner Group, for all of the collaborations we've done, and all the conversations we've had over the years.

I also want to thank my family for all of their support. My parents, Ramona Gil and Eugenio Marcano, for sacrificing so much so that I could have made it this far. I also have to thank my younger brother, David Marcano, for pushing me to do my PhD. Upon hearing that my younger brother was leaving his industry job to pursue a PhD, I decided to do the same and seek out my own PhD. I thank him for helping me with my understanding and mechanics of mathematics, it was extremely helpful, especially early on in my PhD.

And lastly, I want to thank my wonderful wife, Elizabeth Marcano, for always being by

my side and providing all kinds of support. We left all of our family on the east coast during the middle of the COVID-19 pandemic, at the same time I left a nice and comfortable engineering job. Her constant presence at home, and all the friends she has brought into our lives, have been a great source of happiness for me, and I hope for her as well. And in the span of my PhD, we also welcomed our wonderful daughter, Ana María, who is now a very talkative toddler that loves books, just like her mother. I cannot wait to see what the next chapter of our lives looks like as we transition to Chicago.

Chapter 2, in full, is my pre-qualifying exam, or research exam, as part of my PhD program at UC San Diego. The work is my own, but I want to acknowledge Dr. George Porter, who chaired the exam, and Dr. Ryan Kastner and Dr. Aaron Schulman, as committee members and great faculty I've had the privilege of being around.

Chapter 3, in full, is unpublished work coauthored with Melody Gill, Kristin Ebuengan, Sophia Gomez, Dr. Josiah Hester, Dr. Colleen Josephson, and Dr. Pat Pannuto. I am the primary author of this material.

Chapter 4, in full, is my own unpublished work. Dr. Pat Pannuto is my sole collaborator on this work.

VITA

2015	Bachelor of Science in Software Engineering, Rochester Institute of Technology
2015–2017	Computer Science Engineer, The MITRE Corporation
2017–2020	Computer Science Engineer Sr., The MITRE Corporation
2023	Master of Science in Computer Science (Computer Engineering), University of California San Diego
2026	Doctor of Philosophy in Computer Science (Computer Engineering), University of California San Diego

PUBLICATIONS

"Junkyard Computing: Repurposing Discarded Smartphones to Minimize Carbon" Jennifer Switzer, Gabriel Marcano, Ryan Kastner, and Pat Pannuto. 2023. In Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS 2023). Association for Computing Machinery, New York, NY, USA, 400–412.

"Early Characterization of Soil Microbial Fuel Cells" Gabriel Marcano, Colleen Josephson and Pat Pannuto. 2022. IEEE International Symposium on Circuits and Systems (ISCAS), Austin, TX, USA, 2022, pp. 1362-1366.

"Hardware to Enable Large-Scale Deployment and Observation of Soil Microbial Fuel Cells" John Madden, Gabriel Marcano, Stephen Taylor, Pat Pannuto, and Colleen Josephson. 2023. In Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems (SenSys '22). Association for Computing Machinery, New York, NY, USA, 906–912.

"Soil Power? Can Microbial Fuel Cells Power Non-Trivial Sensors?" Gabriel Marcano and Pat Pannuto. 2021. In Proceedings of the 1st ACM Workshop on No Power and Low Power Internet-of-Things (LP-IoT'21). Association for Computing Machinery, New York, NY, USA, 8–13.

"Tailor: Altering Skip Connections for Resource-Efficient Inference" Olivia Weng, Gabriel Marcano, Vladimir Loncar, Alireza Khodamoradi, Abarajithan G, Nojan Sheybani, Andres Meza, Farinaz Koushanfar, Kristof Denolf, Javier Mauricio Duarte, and Ryan Kastner. 2024. ACM Trans. Reconfigurable Technol. Syst. 17, 1, Article 11 (March 2024), 23 pages.

"The Future of Clean Computing May Be Dirty" Colleen Josephson, Weitao Shuai, Gabriel Marcano, Pat Pannuto, Josiah Hester, and George Wells. 2022. GetMobile: Mobile Comp. and

Comm. 26, 3 (September 2022), 9–15.

"Adapting Skip Connections for Resource-Efficient FPGA Inference" Olivia Weng, Gabriel Marcano, Vladimir Loncar, Alireza Khodamoradi, Nojan Sheybani, Farinaz Koushanfar, Kristof Denolf, Javier Mauricio Duarte, and Ryan Kastner. 2023. In Proceedings of the 2023 ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA '23). Association for Computing Machinery, New York, NY, USA, 229.

"Powering an E-Ink Display from Soil Bacteria" Gabriel Marcano and Pat Pannuto. 2021. In Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems (SenSys '21). Association for Computing Machinery, New York, NY, USA, 590–591.

ABSTRACT OF THE DISSERTATION

Intermittent Computing But Persistent State: Leveraging Low-Power Energy Harvesting and Extremely Efficient Electronics to Simplify State Management

by

Gabriel Eugenio Marcano

Doctor of Philosophy in Computer Science (Computer Engineering)

University of California San Diego, 2026

Professor Pat Pannuto, Chair

Intermittent computing systems harvest limited energy from the environment to power their sensing and computing work. Due to low and irregular power harvesting, these systems contend with unpredictable interruptions. Research has focused on ways to cope with the intermittent energy, leveraging soft and hard intermittency, and driving advances in intermittent computing hardware architecture, time-keeping, and software runtimes. However, intermittent computing systems are still difficult to design, program, and deploy. This work explores ways to simplify intermittent computing state management by exploring low power energy sources and analyzing boot and sleep performance. I present a module for real wall-clock timekeeping,

leveraging extremely low power sources to maintain time indefinitely. I then analyze the true costs of hard intermittency for intermittent computing hardware platforms, and develop a model and analysis demonstrating that in most cases, especially where computing is required, soft intermittency is actually the ideal mode of operation. Wall-clock time retention enables the use of more complex networking protocols, and opens the door to easier synchronization and collaboration across networks of intermittent computing devices. And lastly, I argue that batteries, coupled with soft intermittent systems, can last decades, and should be considered for intermittent computing designs moving forward.

Introduction

Mark Weiser coined the concept of ubiquitous computing while working as a Chief Scientist at Xerox. His 1991 Scientific American article [161] describes ubiquitous computing as being composed of systems that "weave themselves into the fabric of everyday life until they are indistinguishable from it." The required technology "comes in three parts: cheap, low-power computers ... , a network that ties them all together, and software systems implementing ubiquitous applications." This is the domain of embedded systems, of computing devices that embed themselves into every day objects to give or enhance function.I estimate that in an apartment could have over 80 embedded systems, in the form of smart light bulbs, window and door sensors, electric sockets, pens, TVs, tablets, cellphones, and even utility meters, but current commercial offerings all require charging, batteries, or infrastructure for energy. To reach the goal of ubiquitous computing, we need to detach embedded systems from energy infrastructure, while still networking them together. Batteries are the traditional way to power devices that are disconnected from the energy grid, but changing the battery or charging hundreds of devices in a home every few days is time and cost prohibitive.

Energy scavenging systems address this particular issue by deriving most or all of their operating energy from ambient sources. The most common is solar, as it is widely accessible and very power dense. However, its power output is very sensitive to changes in illumination, which introduces uncertainty. And this is true of many other environmental sources, including wind, tidal, hydroelectric, thermoelectric, radio frequency, and piezoelectric energy harvesting. Systems that draw their power from these sources inherit their intermittency, creating intermittent computing systems.

Table 1.1. Sampling of constant sources of low power. †Where only power estimates were available, we estimate the active area of the harvester as best as possible to give an estimate for power density.

Traditional Energy Source	Power Density (mW/m^2)
Photovoltaic (Solar) [30]	$310 - 209,000$
Micro Wind Turbine [4]	$65,000$
Tidal Turbine [135]	$40 - 4,000,0000$
Hydrogen Fuel Cell [35]	$8,000,000$

Trickle Energy Source	Power Density (mW/m^2)
Cathodic protection repurposing [89]	100†
Aqueous microbial fuel cells [71]	$10 - 300$
Soil microbial fuel cells [118, 119]	$0.18 - 18$
Betavoltaic batteries [106, 148]	1–50 $mW\,g^{-1}$
Tree trunk thermoelectric harvesting [88, 129]	$0.3 - 3$†
Thermoelectric from solar panels [11]	50
Air-gen [111]	~$0.0001 - 0.05$
Amorphous indoor solar cells [section 3.6.1]	30

There are also environmental sources that are relatively constant, such as from galvanic corrosion, microbial fuel cells, and even tree trunk thermoelectric harvesting! These, however, only provide power in the order of microwatts, and thus have generally been sidelined as energy sources, in lieu of solar. In this work, we leverage soil microbial fuel cells for a proof-of-concept that these low power sources are useful in building intermittent computing primitives.

1.1 The Origin and Evolution of Intermittent Computing

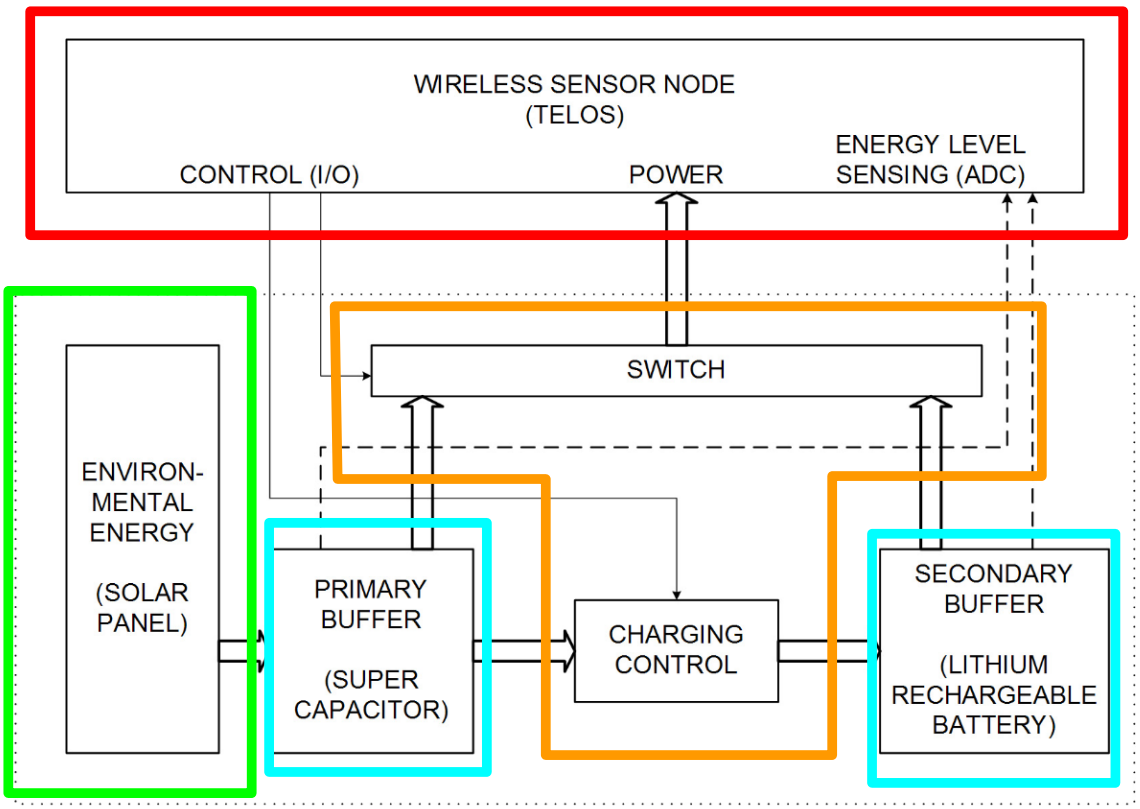


Figure 1.1. Architecture of the Prometheus system [91], with the major intermittent computing subsystems highlighted by different colors, as follows: Energy harvesting, energy store, power management, and application computing.

Prometheus is one of the first "modern" intermittent computing platforms, described in 2005, and it set the basic architecture for intermittent computing systems. Prior systems were radio beacons [137], or used rechargeable batteries with limited lifespans [94]. Critically,

Prometheus is one of the first systems designed to last "indefinitely", or per their estimation at least 40 years (due to the degradation of its rechargeable battery). Figure 1.1 shows the Prometheus architecture, highlighting the four primary subsystems: 1. environment energy harvesting (specifically solar), 2. energy stockpiles (they use super capacitors for primary storage, and rechargeable batteries for secondary storage), 3. power control circuitry (controlling the charging of the rechargeable batteries, as well as which buffer provides power to the mote compute core), and 4. the Telos mote [127]. The Telos mote used in Prometheus is also one of the first in the literature to use the relatively new TI MSP430F microcontrollers (the first MSP430 microcontrollers used ROM or EPROM and came out in 1992, and in 1999 TI introduced the MSP430F with flash for program storage). To reduce average power, Prometheus has the Telos board duty cycle its operation by sleeping every so often, to reduce the average power draw over intermittent cycles. This approach, of sleeping and waking up the system without running out of power completely, is referred to today as soft intermittency.

At approximately the same time, the Intel WISP [145], in 2006, demonstrated that it is possible to do batteryless computing on NFC energy harvesting. It harvests NFC power to respond to a Verify ID command, also leveraging duty cycling the MCU to maintain an average power draw below the amount being harvested when in use. Unlike Prometheus, WISP does not use rechargeable batteries, relying completely on a relatively small 10 μF capacitor to store energy [140]. One major limitation of duty cycling is that its period needs to be determined a priori, which assumes knowing the input power average. For WISP this is not completely unreasonable, as it responds to RFID reader requests and thus harvests energy just when it is needed, but even then it is possible that being far enough from the reader leads to less average input power than the original duty cycle calculation allows, leading to system failure.

Notably, intermittent computing systems have moved away from batteries (with few exceptions [31, 86, 87]), leading to them becoming practically synonymous with batteryless computing. As a result, intermittent computing research has focused on coping with unpredictable energy harvesting. To that end, most of the work focuses on addressing time-keeping,

ensuring forward progress through software runtime support, energy control, and more recently networking, primarily leveraging MSP430 microcontrollers.

1.1.1 Tracking Time for Intermittent Computing

On the timing front, TARDIS [130] in 2012 was the first to use the remanence effect in SRAM on an MSP430 platform. Specifically, the contents of SRAM on the MSP430 do not decay instantly on power-loss, but it degrades stochastically over time. They estimate the time the platform spent off by estimating the rate of decay, and then checking the state of SRAM upon power back on. This estimate is very coarse and system dependent, with the experimental setup for the paper only being able to estimate time between 175 and 225 seconds, or indicating that time was below or above those thresholds. CusTARD [79] introduces using resistor-capacitor (RC) circuits to track time, and achieves less than a 10% error after 40 seconds. CHRT and its variants [46, 170] remain the state-of-the-art for intermittent systems time tracking, and builds on CusTARD by using a cascading array of RC circuits to reduce the overall energy cost and increase accuracy, reaching 1 ms accuracy over 100 s. CHRT is particularly attractive because it is a relatively straight forward circuit that can be implemented with discrete components. Going beyond CHRT, I show in chapter 3 that low power sources enable running modern RTCs indefinitely, retaining timing accuracy well beyond the 100 s range of CHRT while consuming less energy.

1.1.2 Power Management

On the power management front, Prometheus [91] harvests solar energy and stores it in both super capacitors, and rechargeable batteries. They themselves acknowledge the limited number of charge cycles of batteries, however. WISP [145] harvests RFID energy and stores it in a capacitor, and once the capacitor reaches a high enough voltage, it then duty-cycles its microcontroller to reduce average power draw below the power available over RFID energy harvesting. This is the simplest case for energy harvesting for intermittent computing: harvesting

energy into a capacitor or supercapacitor and turning the system on once there is sufficient energy stockpiled. United Federation of Peripherals (UFoP) [75] enhances this idea and gives each peripheral in a system its own capacitor store, in addition to having a primary one that provides energy for the control circuitry and the main microcontroller. Capybara [38] takes the reconfigurability a step further and adds state to the control switches so they do not need a microcontroller to hold their state. Culpeo [138] addresses the problem high equivalent series resistance (ESR) on supercapacitors brings, namely a large sudden voltage drop once the capacitor is loaded, in part by leveraging efficient DC-DC voltage conversion to provide clean power to the microcontroller. CapDYN [54] uses a dynamically and automatically reconfigurable capacitor bank array so that it can adapt to changes in power available. All of these works rely on existing chips for doing the actual energy harvesting. I survey in chapter 2 maximum power point tracking (MPPT) algorithms that maximize the energy extracted from sources and compensate for changing environmental parameters.

1.1.3 Software Runtimes

On the software runtime front, the simplest implementations run as long as the system has power, or until a task is done. WISP [145] figures out empirically how much it needed to duty cycle itself to keep its power draw below the level of power harvested in average. Dewdrop [28] takes a WISP node, and implements a scheduler that runs tasks based on the energy stored, trying to ensure a task does not start unless it is able to complete its work. If a task does fail, however, there is no recovery mechanism, it must start again. Mementos [133] implements checkpointing, where it stores state into flash memory every so often, allowing it to resume tasks from those checkpoints if they are interrupted. Chain [36] uses a task-based system to track state, storing only the data that flows from one task to another, channels, into nonvolatile memory. InK [172] follows in the vein of Chain, but enhances tasks by leveraging some PL concepts to aid in describing them to the kernel, and introduces a preemptive task scheduler to improve reactivity. CatNap [117] goes further by modifying LLVM and allowing more control over task priority and

how tasks should behave in a degraded state. Ocelot [149] follows in InK's idea of freshness, and leverage PL modifications to LLVM to enforce these properties in Rust at compile time. ENGAGE [47] implements a differential checkpointing algorithm to store only the emulator state that changes. AsTAR [167] implements a self-adaptive scheduler, not that dissimilar from Dewdrop, but it does this by working with an RTOS, which enables multiple tasks to execute side by side. And more recently, IntOS [165] implements an RTOS-like system that manages checkpointing for tasks, by undoing interrupted transactions and then redoing them. Notably, the grand majority of runtimes leverage MSP430FR microcontrollers, due to their on-chip FRAM for nonvolatile storage. This choice also led the change from soft to hard intermittency, where systems run until their energy stores run out. However, I show in chapter 4 that soft intermittency generally increases the amount of computation achievable for a given amount of energy.

1.2 Key Challenges

Intermittent computing shows promise for enabling extremely wide-area and remote computing and sensing, but it is still held back by the difficulty in dealing with its intermittency. In time-keeping, for instance, state-of-the-art systems can only keep modest accuracy for a few minutes [46]. This is not sufficient for applications that need to schedule tasks further out or need to have wall clock time. For intermittent systems that require wall clock time, most systems either do some sort of network synchronization, or if the reason is to timestamp data, they transmit the data to a server for that purpose.

On the software runtime front, most of the research focuses on the MSP430FR platforms, as they include on-board non-volatile FRAM able to store data and working memory that automatically persists on power loss. However, the MSP430FR platforms can be slow (typical clock speed range between 1 and 16 MHz).And while the sleep power draw of the MSP430FR family can be fantastic, down to 100s of nanowatts, its runtime power draw is worse than that of more modern microcontrollers running at much faster speeds. And finally, the limited SRAM

(typically 8 KiB) and the 16-bit architecture of the MSP430 also makes porting applications to it particularly painful. In recent years, there have been more papers using ARM-based microcontrollers, but the lack of built-in and/or fast non-volatile memory prevents the use of a lot of the MSP430FR based software runtimes.

Energy management is still critically important, because every harvested microjoule of energy matters. Ever since the release of the MSP430FR MCU line, with built-in FRAM, the community has assumed hard intermittency is better than soft intermittency. Hard intermittency does not incur (much of) a cost while the system is offline and the main store is charging. But modern microcontrollers have non-negligible boot times as they do a lot more in bootrom than the old MSP430s, including clearing SRAM and doing cryptographic checks for OTA updates in flash.

1.3 Thesis

The thesis I defend in this work is this: The power efficiency improvements of components and microcontrollers, along with improving energy harvesting algorithms and ICs, enable the use of low power sources to retain state, and makes soft intermittency more compute efficient than hard intermittency.

This thesis relies on three core components: 1. a clear understanding of energy use and harvesting, 2. simplifying time keeping, and 3. revisiting old assumptions about hard and soft intermittency, and developing a model for evaluating their tradeoffs.

1.4 Insights

I identify the following insights that support and guide my thesis. First, modern off-the-shelf electronics are incredibly efficient, compared even to the mid 2010s. In my experience, each individual chip and/or component today typically draws just 1 µA of current, or even as low as 100 nA. In prior work [49], making discrete control systems with capacitors, resistors,

and transistors was the best way to reduce overall power consumption. In my work, I find that in many cases discrete logic ICs and even non-trivial modern PMIC chips can draw the same or less current than the discrete designs of 10 years ago. This opens an opportunity to reduce part counts on many simple intermittent computing designs, as well a reduce overall quiescent current draw. In some cases, this actually expands the power budget that can be allocated to control circuitry, allowing more complex decision making for energy management.

I spent a lot of my early research working with soil microbial fuel cells, and through them became acquainted with the concept of low but constant power sources [88]. Among these are soil microbial fuel cells [119], galvanic corrosion batteries [89], tree trunk batteries [147], and betavoltaic batteries. Traditionally, these power sources only provide in the order of 100 µW or less of power, but they do so for long time spans, reaching into decades if not indefinitely. Due to their low power output, their use is limited, but it turns out that the control systems for intermittent computing, and even external SRAM and RTCs, operate within this power range! Thus, it should be possible to leverage constant but low power sources to drastically simplify some of the more troublesome aspects of intermittent computing, namely keeping state and time!

Finally, designing low power systems is difficult, and abstractions and rules-of-thumb help speed up designs. Unfortunately, sometimes the low level details do matter. Specifically, decoupling capacitors are sometimes necessary as they help manage noisy and unstable power rails, and in the context of intermittent computing they can even hold onto a small amount of charge we can use. Unfortunately, they incur a cost every time they energize, something I have not seen anyone in the literature address. Intermittent systems relying on hard intermittency, as a result, unconditionally burn as much as 17% of their energy budget just energizing their decoupling capacitors!

1.5 Key Contributions

These insights form the background for the chapters of this dissertation.

Chapter 2 surveys maximum power point tracking algorithms for maximizing energy harvesting across seven different environmental energy sources. The most common of all of these algorithms in energy harvesting chips is fixed point maximum power point tracking (MPPT). The most advanced MPPT research involves solar energy harvesting, but I find that many techniques from solar can be adjusted for other energy sources. Consequently, most commercially available ICs with smart energy harvesting algorithms target solar energy sources. However, I find that it should be possible to manufacture ASICs with specialized MPPT implementations for different power sources.

Chapter 3 describes Artemia, a module that leverages constant but low power energy sources to provide wall clock time for intermittent computing. This goes a step beyond CHRT [46], in that it provides wall-clock time with performance comparable to battery-backed systems. The RTC can be programmed to send pulses at specific time intervals, or even at specific times and dates, to be used to wake up the system, or for any other reason.

Chapter 4 analyzes hard and soft intermittency in depth. The majority of contemporary work defaults to hard intermittency. I present a metric that describes the amount of useful work per total energy harvested.Using this metric, I demonstrate that for most applications that perform even modest computing work, soft intermittency is superior, for power incomes as low as 50 µW to 100 µW. Additionally, I show that in the landscape of modern MCUs, the MSP430FR platform is simply no longer a good fit for intermittent *computing*.

And lastly, chapter 5 describes applications enabled by this work, as well as directions for future work.

Chapter 2

A Survey on Maximum Power Point Tracking Algorithms, Comparing Their Efficiency, Complexity, and Convergence Across Diverse Power Sources

I spent the first three years of my PhD trying to optimize energy harvesting from soil microbial fuel cells. To that end, I sought algorithms that could be used to extract the most amount of power from energy sources. For most energy sources, there exists a load on them that extracts the largest amount of power, and that is what Maximum power point tracking algorithms try to find. They do this by adjusting the load presented by energy harvesters to maximize the power drawn from energy sources. This chapter is the result of surveying the complexity, efficiency, and convergence of maximum power point tracking algorithms across nine distinct energy sources from 2000 to 2022. I also provide a table summarizing the results of the survey.

Amusingly enough, soil microbial fuel cells are particularly resistant to maximum power point tracking algorithms, as their power output is negatively affected if too much load is placed on them, even if briefly. However, other energy sources, particularly ones relying on more physical phenomena, do lend themselves well to maximum power point tracking algorithms, and are mostly limited by the lack of hardware that implements these algorithms for them (most hardware implementations target solar energy harvesting).

2.1 Introduction

Power is a constant challenge for real-world electronic deployments. Additionally, there is a growing demand for power in remote areas, away from existing infrastructure. Renewable energy sources are a possible solution to powering sensors and other applications in remote areas. However, the amount of energy available from many renewable energy sources differs over time due to changing environmental factors. As a result, the maximum amount of power that may be drawn from the source changes as well. Maximum power point tracking (MPPT) algorithms identify the maximum amount of power that can be drawn from a source, and adjust the energy harvesting apparatus accordingly to draw that much power. We examine the trends of MPPT algorithm research for the past 20 years, focusing on the tradeoffs between efficiency, complexity, and convergence.

MPPT research is a well established field, with papers being published on implementations for photovoltaic (PV) cell applications in the 1960s. This early MPPT research focused on increasing power harvesting for solar or PV cells for satellites in orbit [25, 115]. As of 2022, most MPPT research still mostly focuses on solar energy. Additionally, many commercial solar energy deployments leverage MPPT to maximize the amount of power drawn from the cells. Researchers applied MPPT to other energy sources in the mid 1980s, starting with wind turbines. More recently, there is research in applying MPPT for optimizing energy harvesting from ambient energy harvesting, such as thermoelectric and radio frequency (RF) harvesting, and other extremely low power sources, such as microbial fuel cells (MFCs). As there is a lot of history in MPPT research, this survey focuses only on the past 20 years of research, from the early 2000s to 2022.

We focus only on nine energy sources for this survey, namely: solar or PV, wind, tidal, (micro/pico) hydroelectric, piezoelectric, RF harvesting, thermoelectric, proton-exchange membrane fuel cells (PEMFCs), and MFCs. In general, solar, wind, tidal, micro/pico hydroelectric, and PEMFC are high power sources, while piezoelectric, RF, thermoelectric, and MFCs are

low to very low power sources. High power energy sources have the advantage that the MPPT circuitry consumes a very small fraction of the total power being extracted. Very low power sources, on the other hand, have extremely little power available in general, severely limiting the complexity of the MPPT control circuitry if it self-powered. However, recent advances in energy harvesting semiconductor technology are making harvesting energy from such sources increasingly feasible [89, 119].

This survey provides an analysis of the differences in efficiency, complexity, and convergence of the implementations across sources, summarized in table 2.1. We identify and discuss research trends for each energy source, and compare their similarities and differences.

2.1.1 Background

The Nimbus program was run by NASA in the 1960s to develop, among other objectives, a satellite system that could support various atmospheric sensors. The first generation of these designs, used for the Nimbus 1 and Nimbus 2 satellites, used simple power dissipation and battery protection circuitry to manage power. A major drawback of the dissipative regulation implementation is the approximately 75 percent efficiency it achieved. A second generation power system was developed using a switching voltage regulator, improving efficiency to approximately 90 percent. Insights gleaned from the second generation inspired research and development on MPPT [115].

The problem with the first generation of the power management circuitry on the Nimbus satellites was that the solar cells on the Nimbus satellites would, at specific times, produce much more power than the power system could manage. Additionally, as the batteries were fully charged, the drop in the load means something else must dissipate the extra power. This could lead to thermal managements problems on the satellite. The second generation power management system alleviated those problems, but it did not draw the maximum amount of power available from the solar cells as power production decreased during orbit. MPPT and its associated control circuitry addressed all of the previous problems with more control over the

extremes, and being able to extract power maximally during the day as needed.

First generation MPPT systems were slow, and could not handle some load transients that managed to confuse the maximum power point (MPP) detection circuitry. Second generation Nimbus MPPT implementations sped up the sampling frequency to speed up convergence and to deal with load transients more effectively. However, this new implementation traded off the increased ability to deal with transient for higher tracking error due to more noise in the measurements. Overall, the second generation system was better than the original, notwithstanding the tradeoffs endured. [25].

Additionally, NASA also studied applying MPPT to radioisotope thermoelectric generators for off-world power, but determined that the current implementations were too inefficient to apply, compared to a linear shunt regulator.

In the mid 1970s, Costogue and Lindena compared multiple different maximum power utilization approaches for solar arrays. They determined that a dynamic impedance approach, or MPPT, yielded the best results. Additionally, they mentioned that these techniques could be applied for terrestrial solar arrays as well [42]. A confluence of factors, including the energy crisis in the 1970s, increased the interest in researching solar energy production around this time. The first papers beginning to outline the differences in MPPT algorithms appear, comparing closed and open loop implementations (or hill-climbing and model based). Also, a wider variety of implementations are developed, reducing sampling inefficiencies [62, 124]. Software simulators were also developed to help explore different system configurations [69]. In 1978 the MIT Lincoln Labs developed a new MPPT algorithm that could be completely implemented using digital logic [107]. This was achieved by maximizing the current or voltage to the load, instead of searching for the MPP of the source.

Research in the 1980s continued making improvements to solar MPPT designs. Most MPPT implementations until then had used a method called perturb and observe (P&O), the load is adjusted and then the effect monitored to identify the MPP. New studies began to identify problems with P&O, such as its slow ability to adapt to sudden changes in the MPP due to

environmental factors and suggested alternatives, such as the incremental conductance (INC) and open circuit voltage (OCV) methods, discussed later [141, 160]. Of note, Schoeman and van Wyk are the first to identify that for PV cells the MPP is located at a voltage approximately 75% of the solar array open circuit voltage [141]. There was also growing body of literature focusing on incorporating PV systems with MPPT into the regular electric grid [9, 27]. Additionally, for the first time, MPPT is applied to wind energy sources [105].

In the 1990s there is a focus on improving existing MPPT implementations by tackling assumed simplifications from previous systems [26]. Some work also worked to reduce the power requirements for MPPT implementations to make them applicable for lower power sources than before [24]. MPPT is also, for the first time, applied to tidal energy harvesting [151]. There is also the first applications of artificial neural networks to MPPT [82, 142]. In a similar vein, software simulations are also becoming more common as evidence that the improvements work as intended [85, 156].

By the 2000s, more difficult problems were being addressed, such as non-idealities in implementation and environmental conditions [136]. After decades of little activity, there was a renewed interest in applying MPPT algorithms to thermoelectric sources for terrestrial use [171].

The work from the 2010s to 2022 focuses to hybrid MPPT approaches, where more than one MPPT implementation is used at a given time to compensate for the drawbacks in each [21]. Additionally, there is interest in investigating MPPT algorithms with more energy sources, such as MFCs, PEMFCs, RF energy harvesting, and more. We sample publications from this time period for this survey, and present our findings in section 2.4.

2.1.2 Maximum Power Transfer Theorem

MPPT relies on the maximum power transfer theorem. The maximum power transfer theorem states that the maximum power is transferred from a source to a load when the source and load resistances match. The following derivation is for resistive loads, but a similar approach can be taken to derive a similar theorem for reactive loads [100].

For a circuit with a voltage source, a source resistance, and a load resistance, the total current flowing through the system is given by

$$I = \frac{V}{R_S + R_L} \tag{2.1}$$

The power dissipated by the load is given by the equations

$$P_L = V_L I = I^2 R_L \tag{2.2}$$

Substituting the equation for I into the equation for the power dissipated yields

$$P_L = \frac{V}{R_S + R_L}^2 R_L = \frac{V^2}{\frac{R_S^2}{R_L} + 2R_S + R_L} \tag{2.3}$$

As V is constant, in order to maximize the power at the load, we just need to determine the minimum of the denominator of the previous equation

$$\frac{d}{dR_L}\frac{R_S^2}{R_L} + 2R_S + R_L = \frac{-R_S^2}{R_L^2} + 1 \tag{2.4}$$

Setting the first derivative to zero and solving for R_L gives us a minimum or maximum point

$$\frac{-R_S^2}{R_L^2} + 1 = 0 \tag{2.5}$$

$$R_L = R_S \tag{2.6}$$

R_L and R_S cannot be negative, so only when both are equal and positive do we have a minimum or maximum. Looking at the second derivative

$$\frac{d^2}{dR_L^2}\frac{R_S^2}{R_L} + 2R_S + R_L = \frac{2R_S^2}{R_L^3} \tag{2.7}$$

we see that it is positive for all positive values of R_L, indicating that the conditions determined previous is for a minimum. Thus, only when the source and load resistances are matched is the power transferred maximized.

A similar derivation can be shown for reactive loads as well, with similar results. Some converters designed to work with extremely low power sources do not operate in continuous mode, greatly complicating the impedance calculations. However, these can still be performed and managed in order to modify the impedance the source experiences.

2.1.3 A Note on Efficiency

The power dissipated by each component when the source and load resistances match is given by the following equations:

$$P_L = V_L * I = \frac{I^2}{R} \tag{2.8}$$

$$P_S = V_S * I = \frac{I^2}{R} \tag{2.9}$$

$$P_S = P_L \tag{2.10}$$

As the current and resistances equal, each is dissipating an equal amount of power. The efficiency of power transferred from the source to the load is given by

$$E = \frac{P_L}{P_S + P_L} \tag{2.11}$$

and as the source and load power equal per eq. (2.11)

$$E = \frac{1}{2} \tag{2.12}$$

In other words, the efficiency of power transfer from the source to the load, at the maximum power transfer point, is 50%. The power transfer efficiency increases as the load

resistance increases, or as the source resistance decreases. However, a mismatch between the source and load resistances will yield a non-maximal power transfer.

Most MPPT literature focuses on a different, implicit definition of efficiency, however, given by

$$E = \frac{P_{L_{actual}}}{P_{L_{max}}} \tag{2.13}$$

which states that efficiency is defined as the ratio of the actual measured power dissipated to the maximum theoretical power that can be transferred to the load.

An interesting observation from this discussion is that there are instances where maximum power transfer is not ideal. For instance, per the 50% efficiency of power transfer at the MPP, a power source capable of transferring 500MW to a load at its MPP means that it itself is also dissipating 500MW (assuming the source can be modeled by a Thevenin equivalent circuit)! Depending on the application, this might not be a concern, but if total system efficiency is required, the system must be moved off its MPP. Generally, though, the real interest for all of the energy sources reviewed for this survey is to maximize the power that can be extracted.

2.1.4 Switching Voltage Regulator Design

All MPPT designs rely on a switching voltage regulator to adjust the load the source is exposed to. In general, all implementations leverage a switch connected to an inductor and a capacitor that is actuated with some sort of pulse-width modulation (PWM) signal. The actual circuit depends on whether the input voltage is higher or lower than the output voltage. If the input voltage is not high enough for the load, a boost converter is usually used [123].

Should the input voltage be higher than what the load requires, a buck converter is used, as is traditionally used for large solar arrays like for the Nimbus satellites [12, 115]. Figure 2.2 shows a circuit diagram for a buck converter.

The Nimbus report explains well how the PWM duty cycle affects the perceived load the energy sources sees for a buck converter. The derivation for boost and buck converters is as

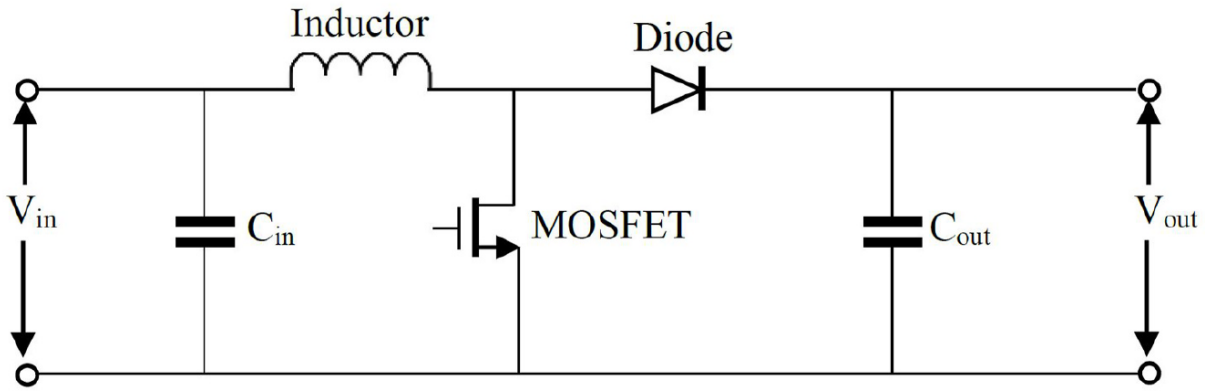


Figure 2.1. Schematic of a boost converter. Reproduced from [55].

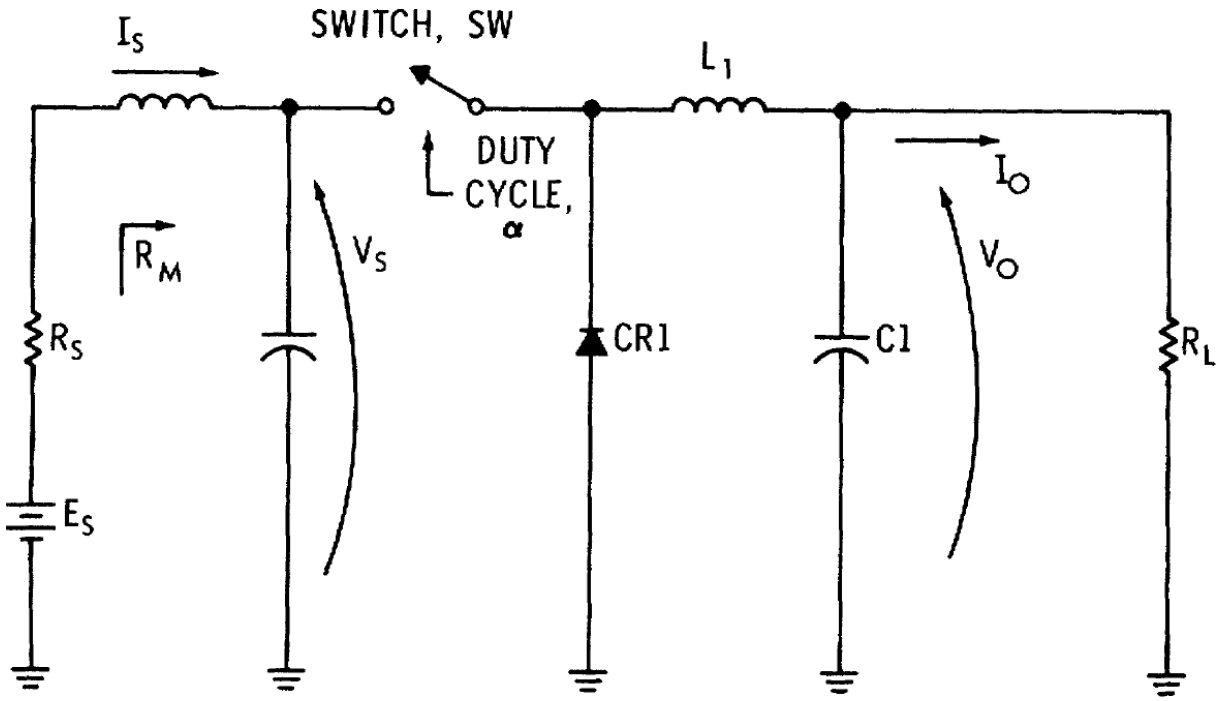


Figure 2.2. NASA Nimbus switching voltage regulator design. This is a buck converter, used to reduce the source voltage. Reproduced from [115].

follows:

$$P_S = \frac{V_S^2}{R_M} \tag{2.14}$$

$$R_M = \frac{V_S^2}{P_S} \tag{2.15}$$

and assuming the switching network is essentially non-dissipative, so, there is no power loss in the transfer itself

$$P_S = P_O = \frac{V_O^2}{P_L} \tag{2.16}$$

then, substituting P_S into R_M yields

$$R_M = \frac{V_S^2}{V_O^2} R_L \tag{2.17}$$

For a boost converter operating in continuous mode (meaning the inductor is never saturated or drained), the switching components act as a sort of multiplier

$$V_O = \frac{V_S}{1-\alpha} \tag{2.18}$$

or

$$V_S = V_O(1-\alpha) \tag{2.19}$$

and substituting this into R_M

$$R_M = (1-\alpha)^2 R_L \tag{2.20}$$

Thus, the resistance the sources sees due to the switching regulator is a function of the duty cycle.

For a buck converter in continuous operating mode, if the converter components are effectively an averaging filter with the switch [12], we can derive the following:

$$V_O = \alpha V_S V_S = \frac{V_O}{\alpha} \tag{2.21}$$

and substituting V_S into R_M

$$R_M = \frac{R_L}{\alpha^2} \tag{2.22}$$

Thus, altering the duty cycle also affects the load perceived by the source.

Similar derivations can be done with Cuk and Buck-Boost converters, all demonstrating that the resistance exposed by the switching regulator is a function of the duty cycle, so long as the regulators are operating in their continuous or steady state mode.

It is interesting that since its inception in the 1960s, the switching regulators have remained an integral and mostly unchanging part of an MPPT implementation. These voltage regulators are also relatively straight forward to implement, with some studies having prototyped them using breadboard [44, 156]. However, some papers since 2010 to 2022 use alternative DC-DC converter topologies, like LLC [153], zeta [128], and flipping-capacitor rectifier [34] which are more difficult to model, but may yield improvements in efficiency. Almost every paper reviewed for this survey uses a more typical converter.

2.1.5 Characteristics of Energy Sources

Solar

MPPT has been extensively studied for solar, or photovoltaic, applications. Photovoltaic cells produce electricity primarily as a factor of insolation (or how much sunlight they are receiving) and temperature. Solar cells are non-linear power sources. Basically, solar cells behave like current sources for a lot of their operating range. As the voltage on the source increases to near the MPP, the voltage of the cells begins to drop. Past the MPP, the current drops rapidly as voltage increases. Figure 2.3 shows this behavior in an IV curve (current versus voltage, also referred to as a polarization curve in some fields).

Wind

For wind generators, the MPP is usually described with graphs comparing the power available versus the generator turning speed. The MPP is affected by the wind speed turning the

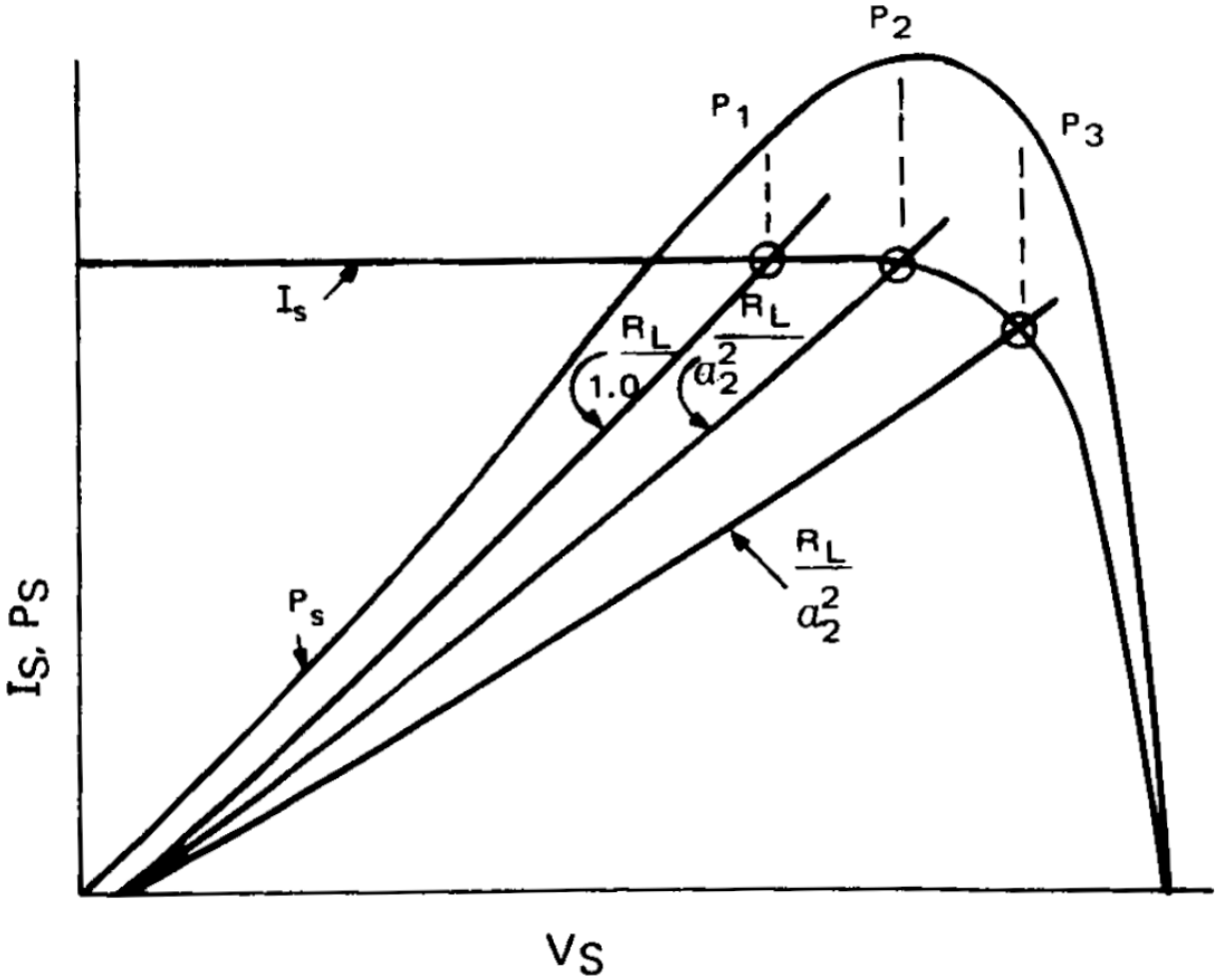


Figure 2.3. Current vs. voltage and power vs. voltage curves of a solar arrays, with sample points showing the effect of load resistance on the operating point. Reproduced from [12].

blades. As shown in fig. 2.4, usually as wind speed increases, the MPP shifts, requiring a higher generator speed. Generator speed is controlled via gearboxes, or by the electrical load providing resistance to the turning generator. Usually there are additional controls preventing the generator from spinning faster than it is designed to turn.

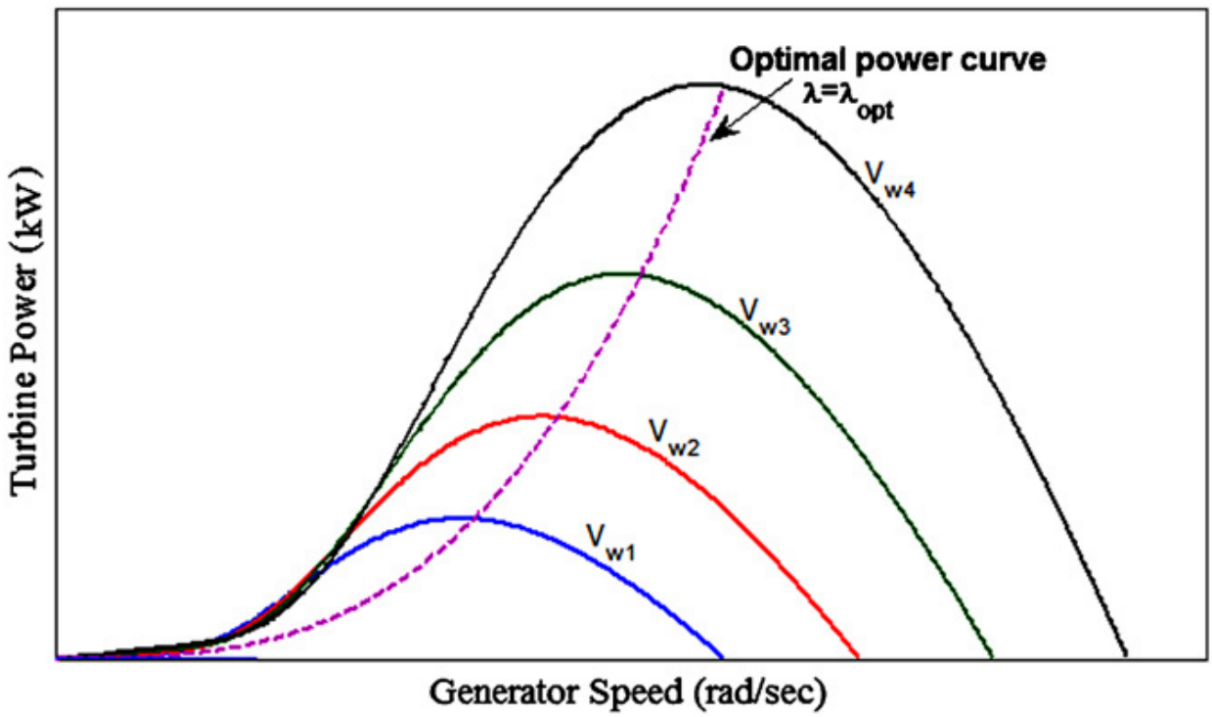


Figure 2.4. Wind generator speed vs. turbine power, showing the MPPs at a few different wind speeds. Reproduced from [1].

Tidal

Similar to wind, the MPP is also described with graphs comparing the power available versus the generator turning speed. As shown in fig. 2.5, usually as water speed increases, the MPP shifts, requiring a higher generator speed. There are also limits to the speed the generator can handle. In a way, tidal and wind generators extract power the same way, by capturing the energy of a medium as it flows past its blades.

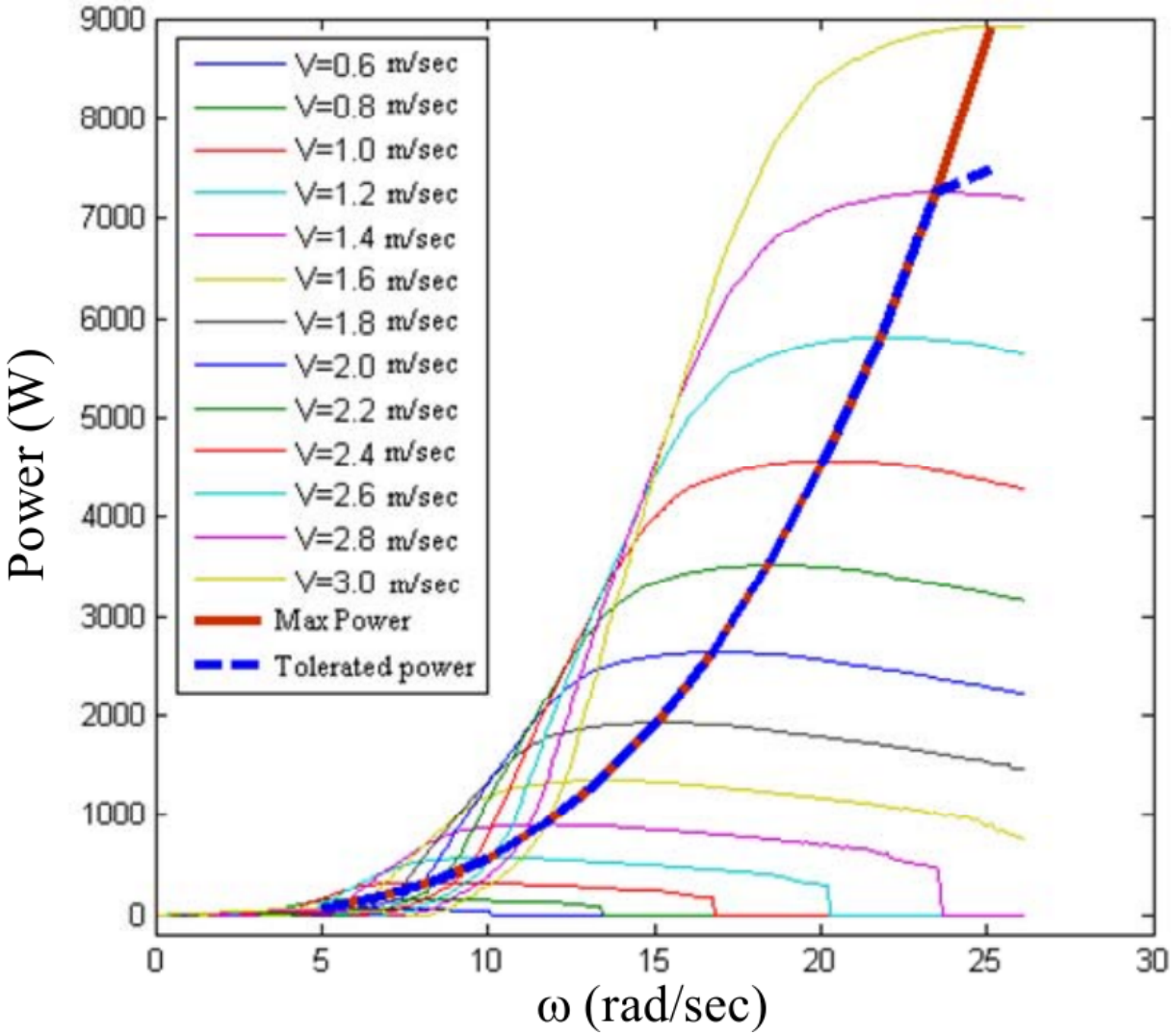


Figure 2.5. Tidal power generation vs. generator angular velocity, given at different tide velocities. The MPP trend is highlighted by the dotted line. Reproduced from [58].

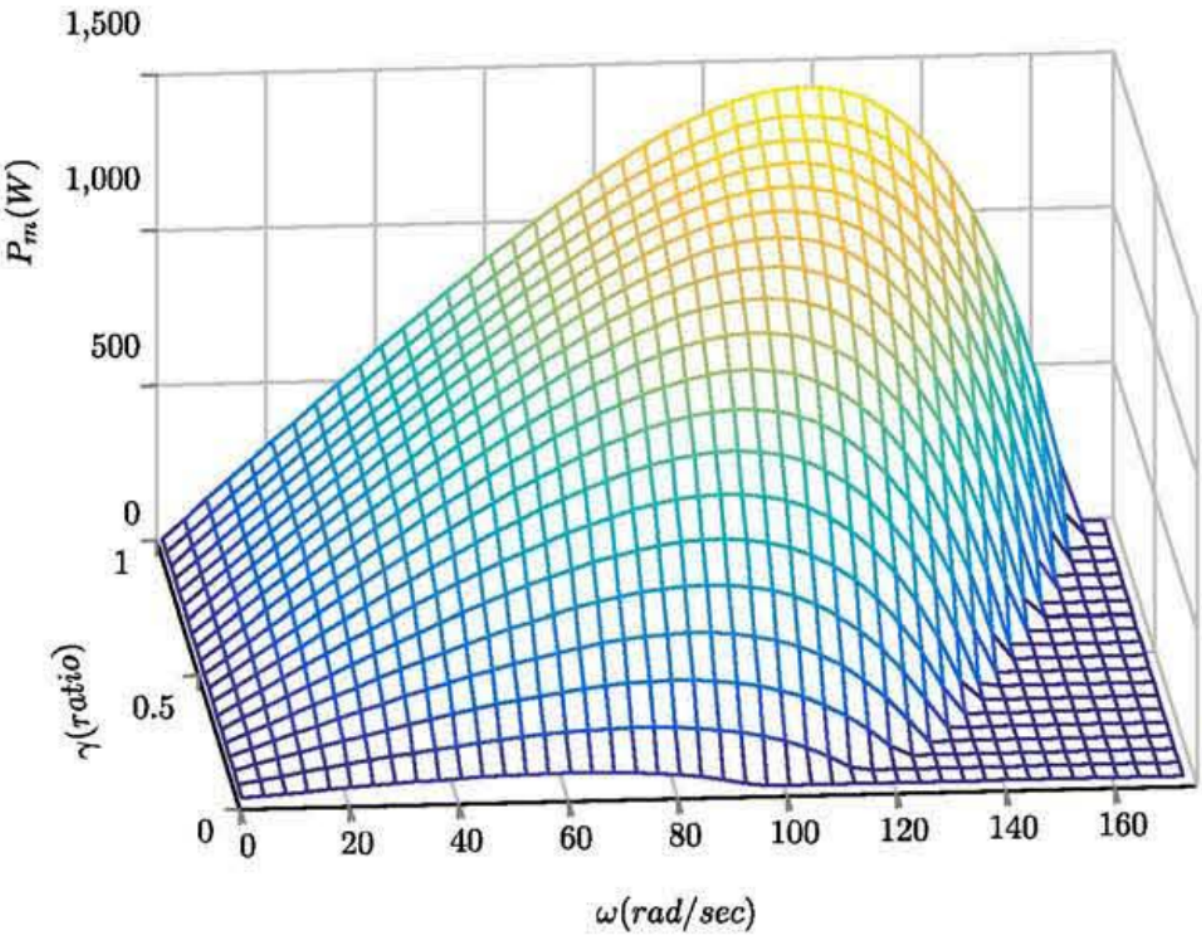


Figure 2.6. Power surface given a fixed pressure head and turbine diameter. Reproduced from [72].

Micro/Pico Hydroelectric

Micro/Pico hydroelectric generators are difficult to analyze. Modeling these generators analytically is a significant problem. Typically, hydraulic turbines behavior is described by Hill charts [72]. Hill charts detail the relationship between the rotational velocity of the turbine, the guide vanes opening ratio, the pressure head (or a description of how much pressure the water is under), the volumetric flow rate, and the turbine's mechanical torque [72]. Figure 2.6 shows a 3D plot comparing the power production versus both the the rotation of the generator and the ratio describing how open the valves controlling water flow are.

Thermoelectric

For thermoelectric generators, the MPP depends on the difference in temperature between the metal conductors being used to produce power. Figure 2.7 shows a typical IV and power plot for a thermoelectric source. Of note, the IV curve of the source is linear, which is leveraged by some researchers to implement simple model-based MPPT algorithms.

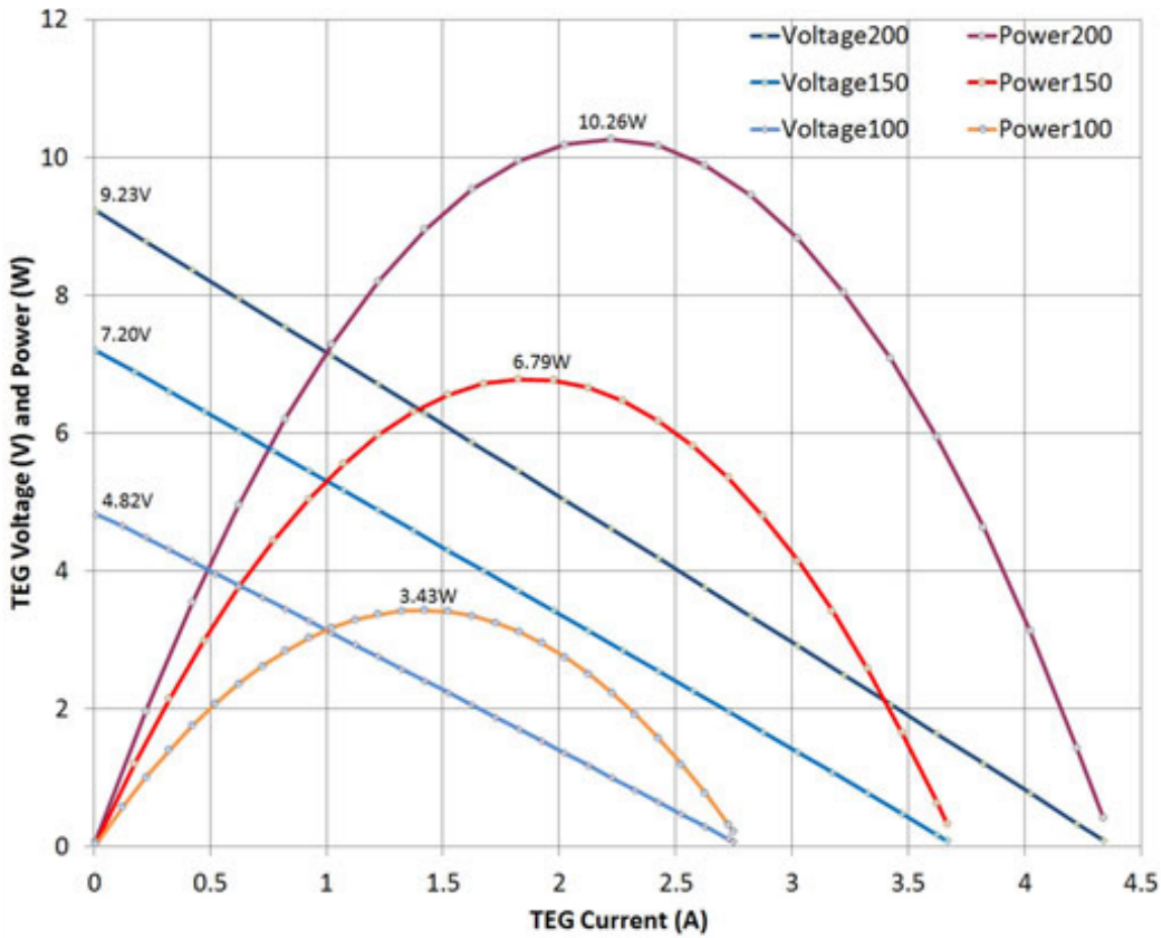


Figure 2.7. Thermoelectric voltage vs current and power curves at different thermal gradients (Δ100°C, Δ150°C, Δ200°C). Reproduced from [121]

Radio Frequency

The MPP for radio frequency harvesting depends on the signal strength received by the rectenna doing the harvesting. Figure 2.8 shows an example of this.

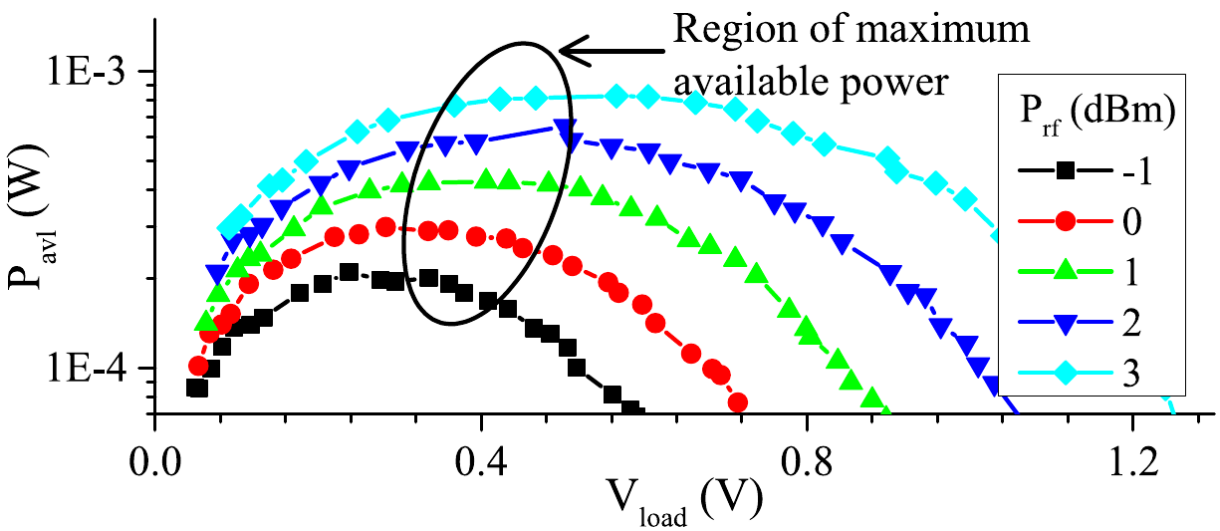


Figure 2.8. RF power vs voltage at specific amounts of signal power. Reproduced from [139].

Piezoelectric

The MPP of piezoelectric sources depends on the frequency and the amplitude of the vibration of the piezoelectric element. Figure 2.9 shows an example of this, with a fixed frequency, but varying amplitudes.

Proton-Exchange-Membrane Fuel Cell

Also known as polymer electrolyte membrane fuel cells. The net power produced depends on the flux of hydrogen to the anode, flux of oxygen to the cathode, the power consumption of the systems providing air to the cell, the membrane water content and the temperature of the system. However, research has demonstrated that for a given set of parameters, there appears to be an MPP. An example is given by fig. 2.10.

Microbial Fuel Cell

There are different kinds of MFCs. In general, power is produced by the metabolic activity of the bacteria as they process fuel. The exact modeling of what happens is an active area of research. Figure 2.11 shows an IV curve for an MFC, showing that the voltage versus current relationship in an MFC cannot be described by a function! However, there still appears

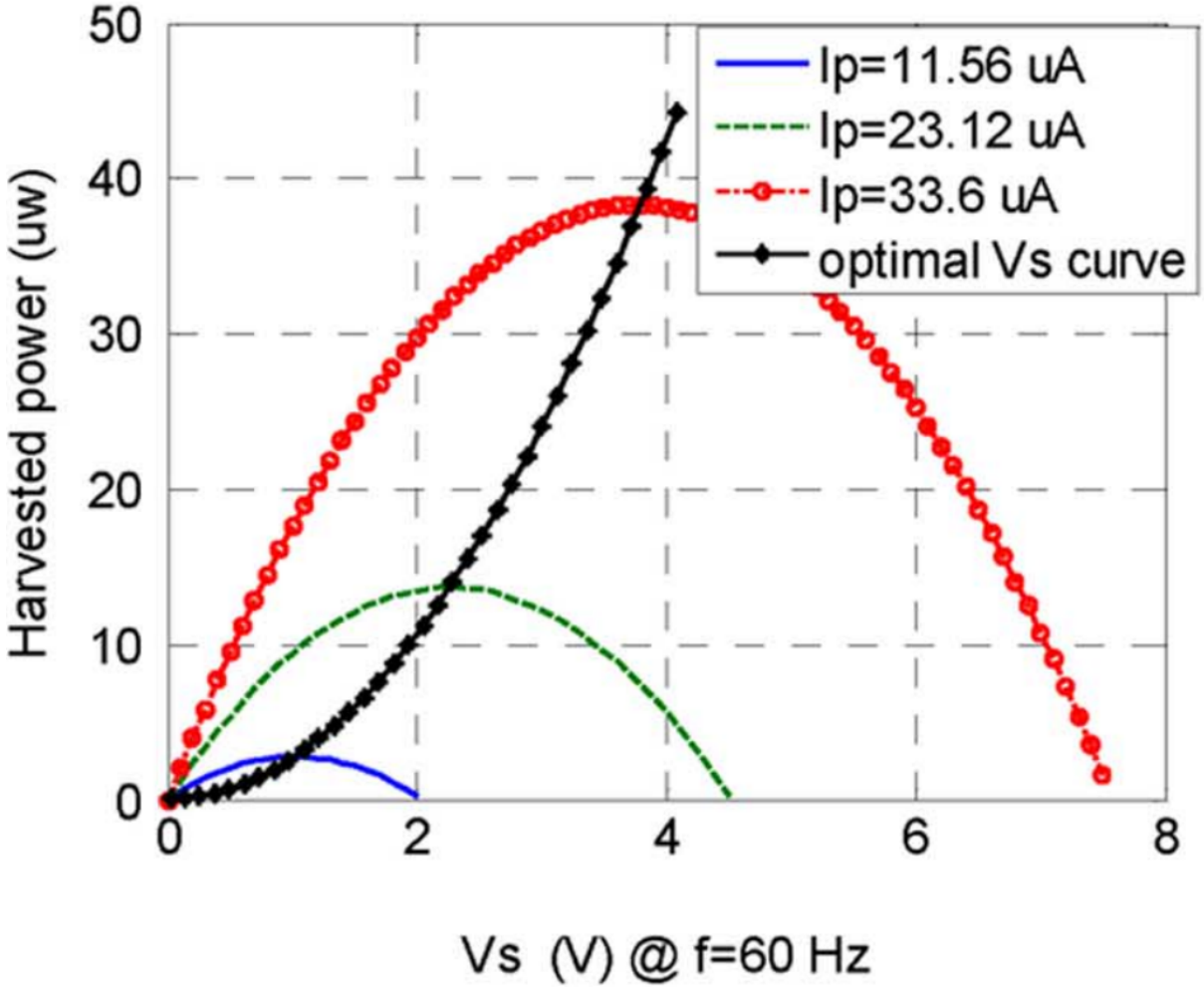


Figure 2.9. Piezoelectric power vs voltage at specific frequency, with curves at different vibration amplitudes. Reproduced from [112].

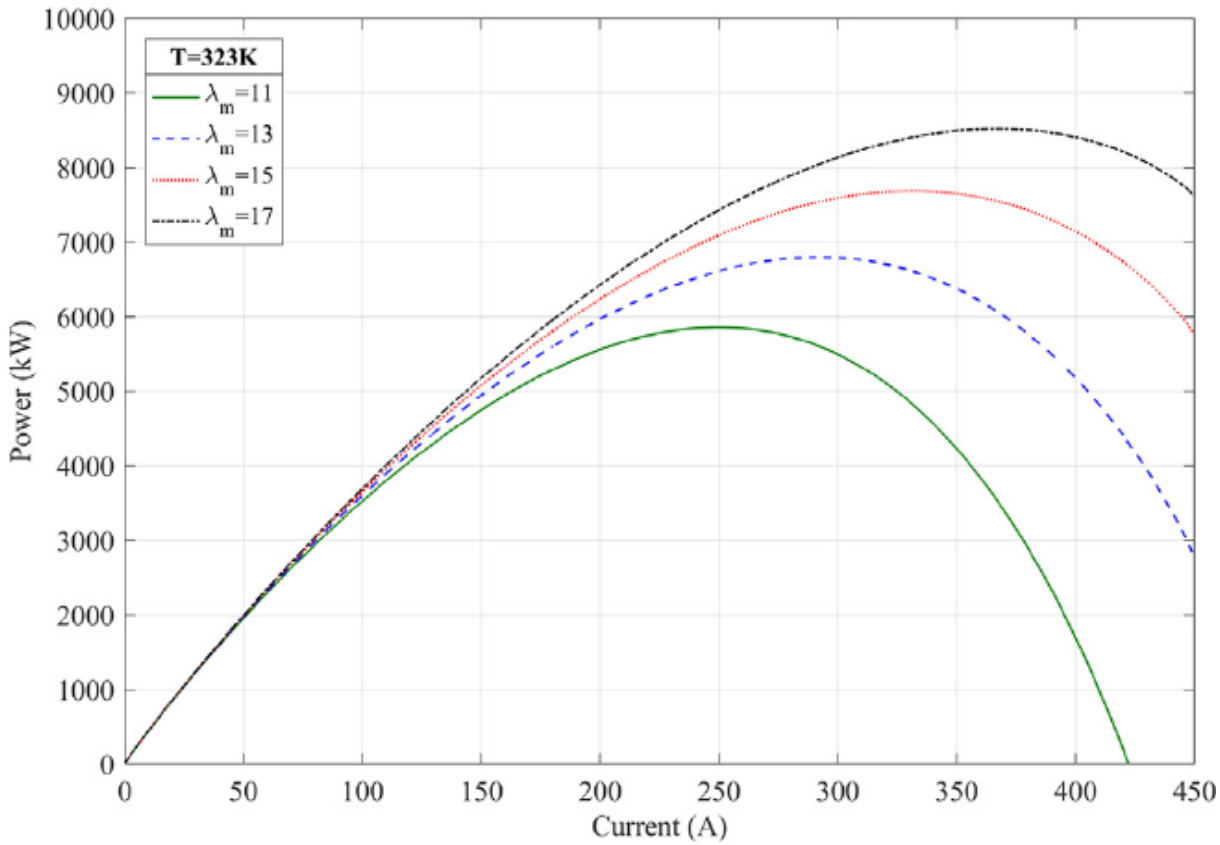


Figure 2.10. PEMFC Power vs Membrane Water Content. Reproduced from [132].

to be an MPP.

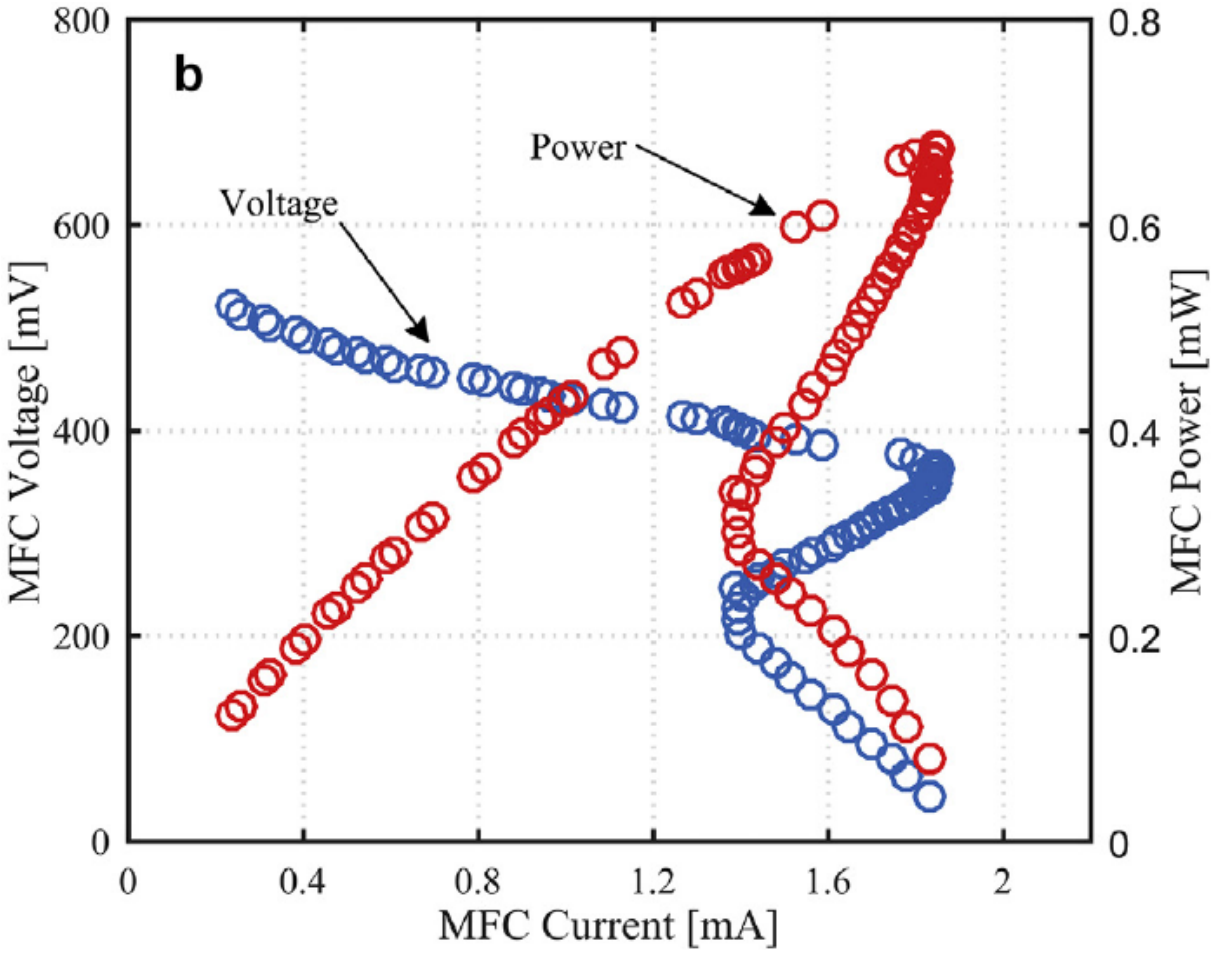


Figure 2.11. MFC IV and power curves. Specifically, this is data from an MFC after recovering from a voltage overshoot condition. Reproduced from [5].

2.1.6 MPPT Ontology

For this survey, we categorize the algorithms into formulaic, computational, and hybrid methods. We take inspiration from a previous survey which categorized the algorithms as conventional, soft computing, and hybrid methods [15]. We believe that the labels used in our survey describe the algorithms more directly. Other surveys categorize MPPT algorithms into categories such as offline and online [23, 134], based on the source of the parameters used to determine the MPP. Offline methods determine the MPP using models of the source a priori,

using simple signals to control the MPPT algorithm. Online methods, on the other hand, rely strongly on feedback parameters to identify MPPT behavior and locate the true MPP. We find these categorizations lacking, as in actuality all of these methods (except for some model-based methods) rely in some form of feedback to find the MPP. The primary difference we find between these offline and online methods is that the offline methods make strong assumptions about where the MPP is located (and thus are really just estimates), while online methods make no such assumptions. Some other surveys also classify some approaches as hybrid methods, which we borrow [134].

Formulaic methods were the first methods to be developed. These methods usually involve one to a small set of equations or formulas to test the operations of the system and make decisions on what to adjust. P&O and the open circuit voltage method are examples of these approaches. Due to their relative simplicity, most formulaic methods are implementable using analog components. Also, due to their reliance on formulas and equations, model-based methods are also in the formulaic category. Computational methods leverage computationally intensive algorithms such as artificial neural networks (ANNs) and fuzzy logic. These methods, while more complex to implement, and more energy intensive, tend to have better MPP tracking behavior than the formulaic methods. Hybrid methods combine one or more of the previous methods to cover for the deficiencies of one with the others.

One general observations is that many algorithms include slight variations from the base/original method, usually to compensate or deal with system limitations, or trading off convergence speed for stability, as an example.

2.1.7 Functional Methods

Perturb and Observe

The oldest, and most well-known method used for MPPT is known as perturb and observe (P&O). At a high level, P&O relies on modifying, or perturbing, the duty cycle of the PMW signal used to drive the switching voltage regulator, and tracking some parameter to identify

whether the modification moved the current power point towards or away the MPP. The process is illustrated by fig. 2.12. The Nimbus MPPT design implements P&O, tracking the current and identifying where it is maximized [12]. So long as there is only one MPP, as this is a hill-climbing algorithm, it will settle around the region of the MPP.

Simple implementations of P&O suffer from instability around the MPP. Specifically, because this approach traditionally perturbs the load to the source by a set step size, it tends to oscillate around the actual MPP. This can be alleviated by reducing the perturbation step size, but then it makes the implementation less robust to sudden changes in the system (e.g. a cloud obstructing solar panels). Another possible approach is to increase the perturbation rate, but this could make the implementation more sensitive to transients and mischaracterize the MPP in steady state operation, reducing efficiency [61].

2.1.8 Incremental Conductance

Due to the weaknesses of P&O, an alternative method referred to incremental conductance was invented in 1983 by Wasynezuk [160]. Incremental conductance leverages the observation that at the MPP

$$\frac{dP}{dV} = 0$$

and additionally

$$\frac{dP}{dV} = \frac{d(VI)}{dV} = \frac{d(VI)}{dV} = V\frac{dI}{dV} + I$$

With these equations, Hussein et al. developed the flowchart in fig. 2.13 demonstrating the logic of the method [85]. Effectively, if $\frac{dP}{dV} < 0$, the true MPPT is to the left of the current set point, and if $\frac{dP}{dV} > 0$, the true MPPT is to the right of the current set point. Figure 2.13 shows the flow of the logic.

Of note, like P&O this algorithm only works if there is only one MPP with no additional local maxima or minima. This algorithm shows better stability compared to P&O as it actually has a termination condition when $\frac{dP}{dV} = 0$, although it is usually implemented so the stop condition

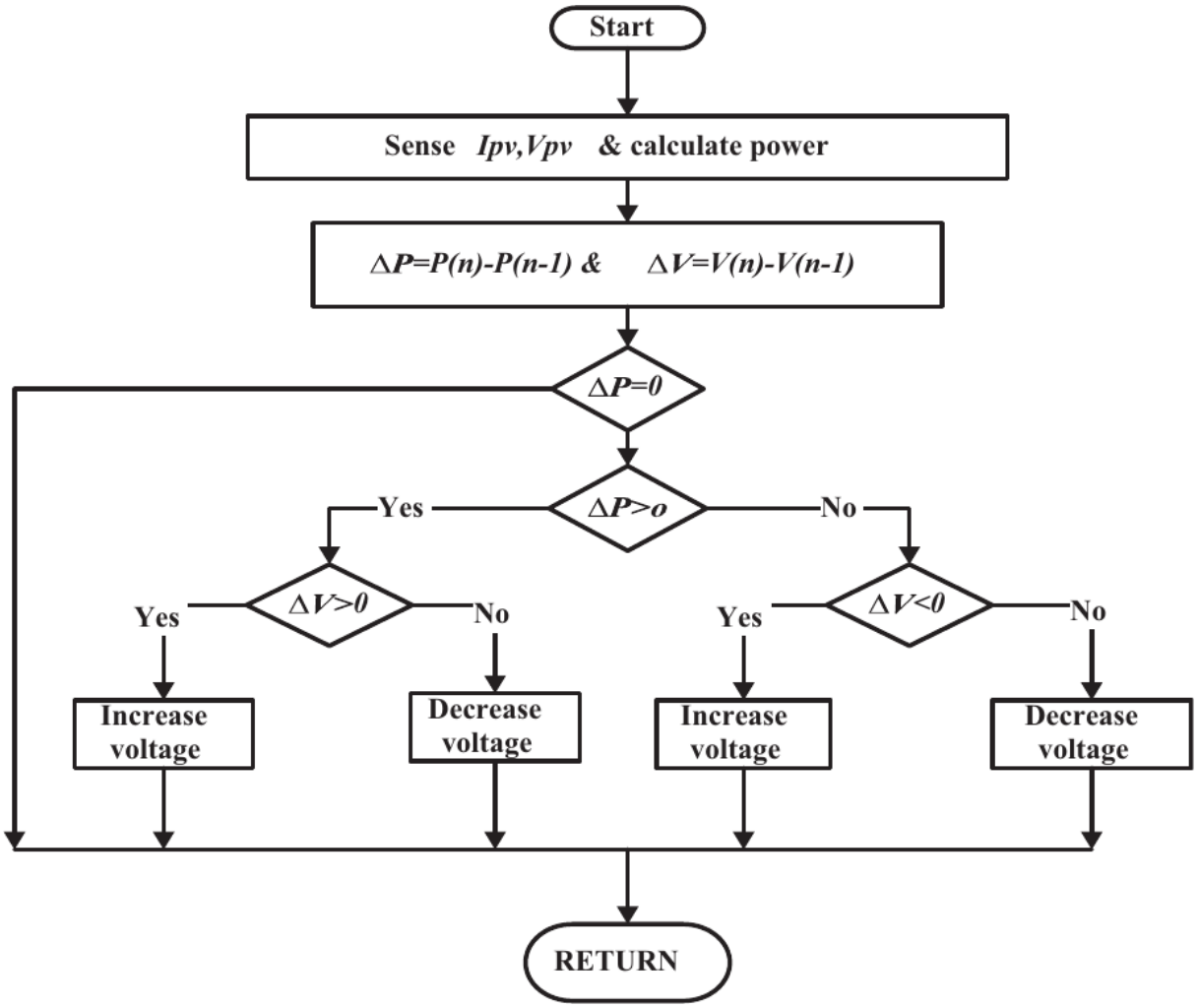


Figure 2.12. P&O logic flow. Reproduced from [23].

is met when $\frac{dP}{dV}$ gets close enough to 0.

2.1.9 Open Circuit Voltage

The open circuit voltage method, unlike the previous functional methods, is one of simplest in its design and implementation [141]. However, its simplicity is due to the major assumption it makes that the MPP is located at a fixed percentage of the open circuit voltage. Figure 2.14 outlines the algorithm. The control circuitry temporarily disconnects the load to sample the open circuit voltage of the source, and then compares said voltage against a predetermined voltage at some percentage of the open circuit voltage. For solar arrays, this percentage has been empirically determined to be between 75% [141] and 80% [84]. The result of this comparison is then used to control the duty cycle of the switch, with the goal of keeping the regulated voltage output at the specified percentage of the open circuit voltage.

The major downside of this method is that the control circuitry must disconnect the source temporarily to sample its open voltage, reducing the efficiency of the power transfer [83].

2.1.10 Closed Circuit Current

The closed circuit current method is very similar in principle to the open circuit voltage approach. Alghuwainem determined that the MPP was attained when the load current is around 85% of the short circuit current [6]. The implementation by Alghuwainem eliminates the need to interrupt the source to measure the closed circuit current by using a second, reference solar cell co-located with the primary. The reference closed circuit voltage is compared against the current actively flowing through the primary circuit. This difference is used to adjust the duty cycle, and thus the effective resistance of the switching regulator, to track the real MPP.

The choice by Alghuwainem to leverage a secondary cell to not affect the power being harvested by the primary cells is a design trade-off. While the secondary cell does allow for a clean reference signal, the real MPP for the primary cells might be slightly different than for the secondary cell due to the difference in their location. A cell from the main array can be allocated

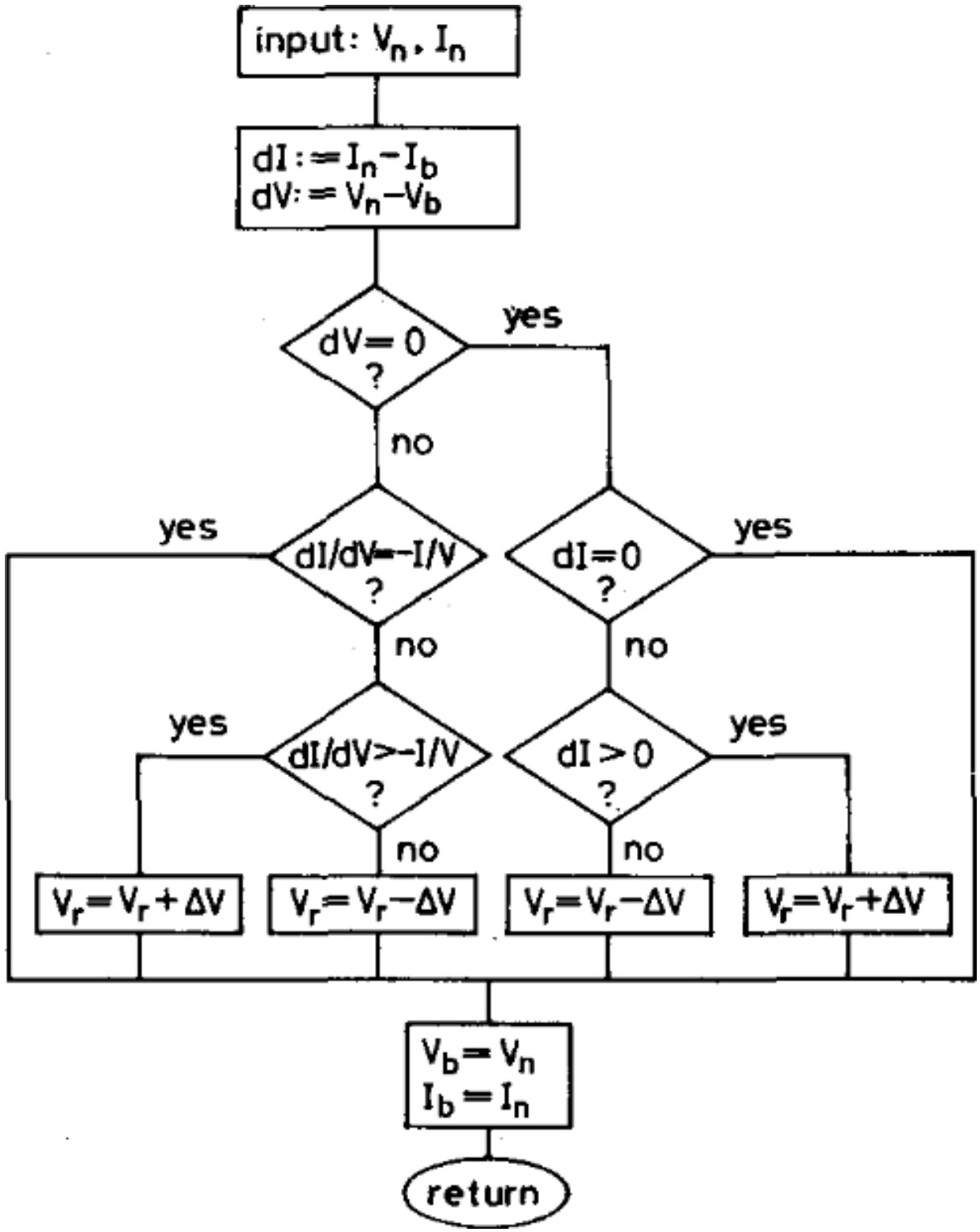


Figure 2.13. Flow diagram showing how the incremental conductance method operates. Reproduced from [85].

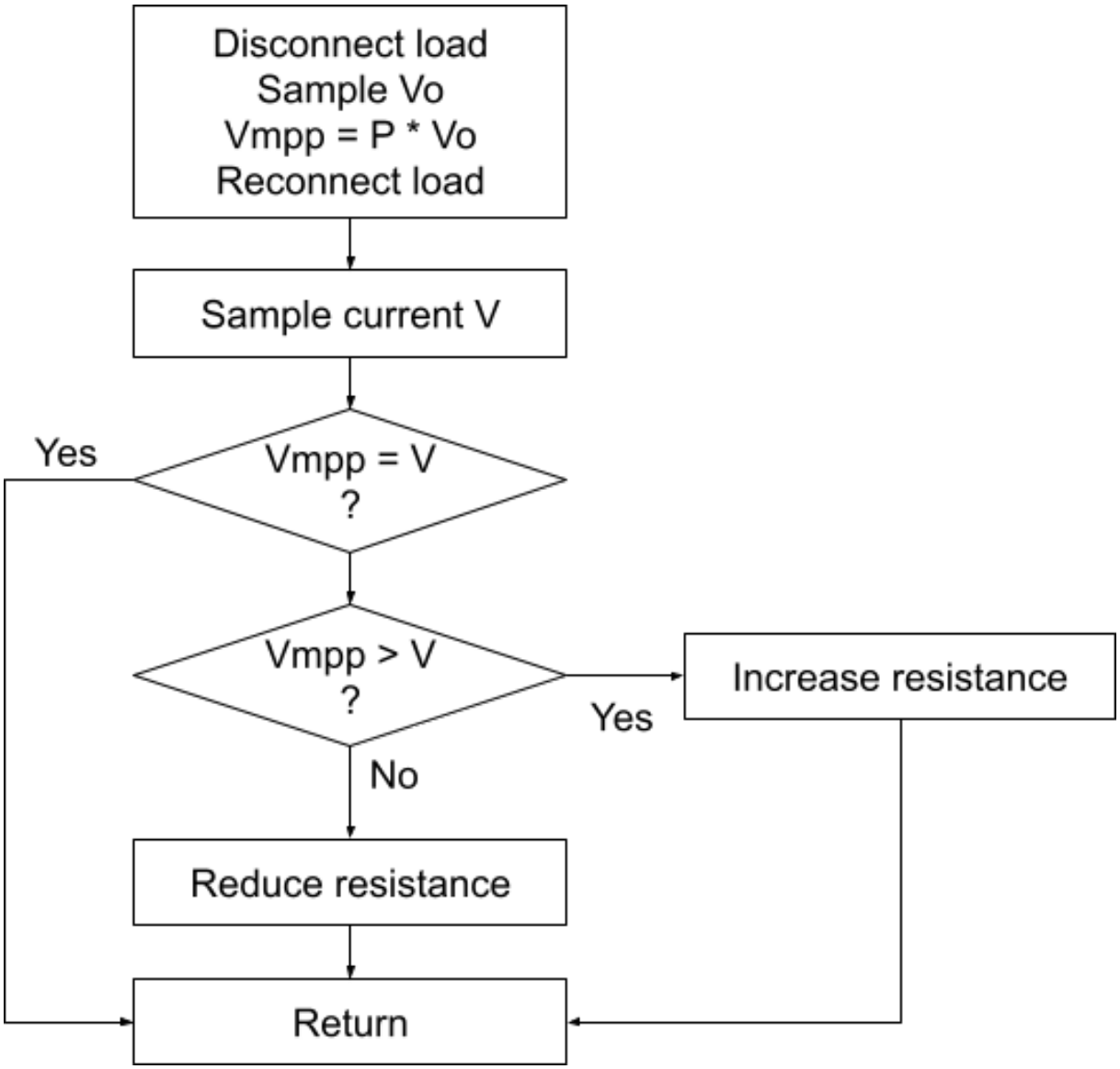


Figure 2.14. Control flow for the OCV algorithm.

to be the reference cell, but then this means that some of the capacity of the primary array is reduced as a result.

2.1.11 Model-based

We refer to any method that relies solely on precomputed models to determine the control signal for MPPT as a model-based MPPT implementation. These methods are among the simplest to implement, as they require very little control hardware. However, at best they yield an approximation of the MPP, although it may be a very good one. Some implementations rely on some form of feedback to update parameters the model requires. Others merely require a constant signal reference that fixes the impedance of the DC-DC converter to a predetermined value.

2.1.12 Computational Methods

Fuzzy Logic

MPPT algorithms based on fuzzy logic leverage its lack of reliance on a precise mathematical model of the system [164]. For the first implementation of such a system, Won et al. expressed input variables terms of linguistic variables like PB (positive big), PS (positive small), ZO (zero), NS (negative small), and NB (negative big). The input variables are error, which is defined as

$$E(k) = \frac{P(k) - P(k-1)}{i(k) - i(k-1)}$$

and change of error which is defined as

$$CE(k) = E(k) - E(k-1)$$

Where P is the power output, i is the current, and k represents a given time. The output variable is the change of duty ratio (dD). Each input and output is then mapped into the membership classes. A table is filled out where the axes correspond to the input variables, and the output is

the table entry. The output goes through a defuzzification process, where it is converted from fuzzy logic to something that the PWM controller can manage.

CE / E	NB	NS	ZO	PS	PB
NB	ZO	ZO	NB	NB	NB
NS	ZO	ZO	NS	NS	NS
ZO	NS	ZO	ZO	ZO	PS
PS	PS	PS	PS	ZO	ZO
PB	PB	PB	PB	ZO	ZO

Figure 2.15. Sample fuzzy logic table, with outputs mapped based on *E* and *CE* inputs. Reproduced from [164].

Fuzzy logic MPPT implementations converge faster than typical P&O implementations and do not exhibit oscillatory behavior near the MPP [164]. One downside of fuzzy logic is that each application requires determining how to best map values to fuzzy logic variables, and back [110], which is difficult.

2.1.13 Artificial Neural Network

Artificial neural networks (ANNs) leverage layers of artificial neurons to model systems [81, 82]. Training happens offline, generating a set of weights that can then be applied to the network that will be connected online to perform MPPT. ANN implementations track the MPPT faster and more accurately than P&O under changing conditions, with the tradeoff of being significantly more complex to implement. Additionally, training must be performed for each system with different energy source parameters, which requires a lot of data collection.

One particular ANN we wish to highlight from our review is the adaptive neuro fuzzy inference system (ANFIS) ANN, designed to perform like a fuzzy logic control system. ANFIS ANNs require training, but via that training they can design the membership classes of its inputs, facilitating the use of fuzzy logic for MPPT purposes.

2.1.14 Artificial Intelligence Algorithms

Some MPPT implementations leverage advances in AI algorithms to perform random searches that converge quickly on maximums. Among the algorithms used in some of the papers are particle swarm optimization (PSO) [96, 128], greedy search [175], and grey wolf optimization (GWO) [90].

2.1.15 Other Algorithms

We find at least one paper that implemented a form of linear search and also binary search as part of their hybrid MPPT implementation [109]. A more thorough discussion of this particular implementation can be found in section 2.4.6, but it is important to note that this hybrid algorithmic approach is designed to be implementable in silicon hardware. There may also be other MPPT based algorithm approaches, but these appear to be a small minority of all papers.

2.2 Hybrid Methods

Hybrid MPPT implementations leverage the strengths of one or more algorithms to compensate for the weaknesses of others to achieve better performance overall. For instance, some algorithms may be very fast in arriving at an approximation of the MPP, but on their own do not reach the true MPP. Other methods, such as hill-climbing algorithm like P&O, might be slower to converge to an MPP but will converge at the true maximum (assuming a single power source so there are no local maxima to contend with). When combined, the approximating algorithm finds a close starting point for the hill-climbing algorithm, which can then find the proper MPP with little hill-climbing.

There are many possible combinations of methods that could work in tandem to improve their individual MPPT performance. Refer to section 2.4 for our discussion of these hybrid methods [90, 128, 153, 175].

2.3 MPPT Publication Sampling Methodology

We sample five MPPT publications per energy source discussed in this survey published since 2000 to 2022 understand the research trajectory for MPPT across the studied energy sources. We limit the quantity of publications to five per source, yielding a total of 45 across all nine energy sources. More specifically, we seek to understand what are the prevalent methods applied within and across all energy sources, and how the methods applied have evolved over time.

We select two of the top cited non-survey papers since 2000, one of the top cited papers since 2020, and two papers from either 2021 or 2022. We analyze the selected papers and classify the methods presented per our ontology. We then compare to see which specific methods are applied, and whether any modifications were made and for what reason. We identify whether the algorithms presented could be used for more than one energy source. We discuss our findings, and summarize our results in table 2.1.

2.4 Results and Discussion

We construct table 2.1 by estimating the different properties of each algorithm we survey. It is difficult to provide specific numbers for the different qualities of the methods explored and shown in table 2.1, as many values are not directly comparable. For instance, the same algorithm as applied to solar might have 98% efficiency, but that might be 50% for MFCs, although that might be high for both. Another difficult to describe parameter is whether each algorithm tracks the true MPP. Some algorithms fail to track the MPP in the presence of many local maxima. To distinguish these from other methods that can find the global maximum, we mark them with an

Table 2.1. Categorization and comparison of the surveyed MPPT publications.

Method	Type	Complexity	Efficiency	Convergence Speed	Convergence Accuracy	Sources Applicable	Tracks True MPP	Vulnerable to Oscillation
P&O [61, 68, 95, 104] [13, 14, 72, 101] [116, 150]	Functional	Low	Good	Moderate-Fast	Moderate-Good	Many	Yes*	Yes
OCV [33, 119, 158] [34, 108, 154] [126, 173]	Functional	Low	Moderate	Fast	Approx.	Many	No	No
Model [103, 112, 123] [20, 131, 176] [50, 59, 155, 177] [18, 41, 139]	Functional	Low	Variable	Fast	Moderate-Good	Many	No	No
FMIM, P&O [153]	Hybrid	Moderate	High	Fast	Good	Solar	Yes*	Yes
ANN [92]	Computational	High	High	Fast	Good	Many	Yes*	No
Fuzzy Logic [7, 125]	Computational	High	High	Fast	Good	Many	Yes*	No
AI [60, 152]	Computational	High	High	Moderate	Good	Many	Yes*	No
ANN, AI [128, 175]	Hybrid	High	High%	Fast	Good	Many	Yes	No
AI, Linear Extrapolation [90]	Hybrid	High	High	Moderate	Good	Many	Yes	No
Linear Search, Binary Search [109]	Hybrid	Moderate	Moderate-High	Moderate	Moderate-Good	Many	Yes*	No

asterisk (*) in the table.

2.4.1 Solar

Of all the papers read for this survey, we identified only one which used an LLC converter [153]. The proposed MPPT system uses a two step process to identify the MPP: 1. using a frequency-modulated impedance-matching (FMIM), and 2. P&O. This FMIM approach approximates an MPP by determining the intersection of the IV curve and the conductance (G) curve $G_p v = \frac{I_s c}{V_o c}$. Final adjustments to the MPP are made by P&O, with a small modification that if the difference between the current and prior power readings exceeds a threshold, the FMIM process is re-executed to converge faster on the new MPPT. This hybrid approach maintained low error rates (less than 2%) even under rapidly changing conditions. It also converges in low light conditions where conventional algorithms fail due to tradeoffs between tracking speed and operating frequency. One caveat is that this algorithm will only converge on the MPP of a single cell, as it makes no attempt to filter out any possible local maxima that occur under non-uniform shading conditions on solar arrays with multiple cells.

Another hybrid approach uses an ANFIS ANN and the PSO AI algorithm, in addition to using a zeta DC-DC converter instead of a more typical boost-buck converter [128]. The ANFIS network is trained to behave like a fuzzy-logic controller, providing an estimate of the duty cycle required to match the global MPP. As it behaves like fuzzy-logic, the ANFIS model incurs its weaknesses, including its impreciseness. The PSO algorithm is used to refine the ANFIS output to find the true global MPP.

The two hybrid MPPT implementations demonstrate that it is possible to improve MPPT performance if the application can tolerate the increased complexity (and thus energy demands). For large solar applications, the energy consumption of modern microcontrollers capable of executing complex algorithms is low compared to the array power output. More importantly, the additional power harvested by leveraging these hybrid MPPT algorithms may be more than the additional power required to execute the more complex implementations, resulting in a net

increase in power harvested.

The other three works we survey use P&O or variants of P&O. P&O has well known problems, but nonetheless it sees continued use due to its simplicity, enabling smaller and/or cheaper deployments [150]. The oldest of these publications uses a microcontroller (MCU) to run a modified P&O algorithm [104]. The primary adjustment is that the algorithm has some threshold/error around the MPP, that once the derivative of the power is close enough to zero, it will stop running until the threshold is crossed again. This eliminates the P&O oscillation problem, but then makes it overall less accurate and slightly less responsive to subtle changes. Another paper concludes that the reason P&O underperforms is because the parameters used to control the algorithm are not usually optimized for a given application and solar array [61]. The most recent of the P&O papers introduces an interesting modification involving adding zones to the power vs voltage curve and adjusting the perturbation period based in the zone [150]. Each zone is defined based on the slopes of the power vs voltage curves of two curves at a given minimum and maximum irradiance and temperature. The step size of the P&O algorithm is larger for zones far from the MPP, and small for the zone with the MPP. This implementation still suffers from oscillations at steady state, although these are kept limited due to the small step size in the MPP zone. One important observation is that besides the known oscillatory problems of general P&O implementations, all of these algorithms also fail to converge to a global MPP if there are any local minima, as is the case under partial shading conditions on an array with multiple solar cells.

2.4.2 Wind

Many of the MPPT implementations for wind generators rely on wind-generator-specific models [18, 41, 103]. The principal reason for this is that for wind (and tidal) generators the maximum power extracted from the generator is more easily described and controlled as a function of the rotor speed. Rotor speed can be controlled via gearboxes or directly by adjusting the load resistance experienced by the generator. Two of the methods investigated require

individual system calibration, after which control can be maintained with the help of a constant reference current signal [18] or identifying the tip speed ratio (TSR) and using more model parameters to estimate the wind speed based on the angular speed of the generator [41]. The other method investigated implements a method similar to INC where it tracks $\frac{dP}{d\Omega}$ where Ω is the rotor speed, instead of tracking $\frac{dP}{dV}$.

As with many other energy sources, at least one paper investigated the use of P&O [95]. An interesting deviation from typical P&O implementation is that they also use some modeling knowledge to extract a constant once the MPP is reached for the first time, and then use that to adapt the perturbation size for faster MPP tracking under changing wind conditions. However, this process merely yields an approximation to the new real MPP, which is then reached using a more typical P&O. This method still requires wind generator specific information, such as the number of rotor poles in the generator, as it uses this information to estimate the wind speed.

The last method investigated uses fuzzy logic to help manage generator inertia and interface well with the electric grid [125]. The problem being addressed is that wind turbines that use MPPT decouple their generator from the electric grid, and thus do not support the grid frequency well. Fuzzy logic is used to help the MPPT system to recover quickly from disturbances in generation introduced by grid frequency support conditions.

An interesting observation is that none of the papers surveyed for wind generators mention combining multiple generators in parallel and having them all controlled by a single MPPT algorithm. We hypothesize that this is because it is difficult to co-locate multiple wind turbines in close enough proximity to allow multiple to be managed by a single MPPT controller. As a result, none of the algorithms explored contend with the possibility of encountering local maxima instead of the global MPP.

2.4.3 Tidal

We cannot find sufficient papers for tidal energy sources published between 2020 and 2022. As an illustration of the difficulties in finding recent papers on tidal MPPT design, the

top-cited paper in that time span mentioning tidal and MPPT describes a system combining solar, wind, and tidal energy harvesting, but only offers an MPPT implementation for the solar component [146].

Tidal energy harvesting is very similar to wind energy harvesting, as both extract energy from a medium flowing through blades that turn a generator. As such, some similar model-based approaches are used in some of the implementations investigated. However, unlike the wind MPPT implementations leveraging TSR models, the tidal implementations do not attempt to measure the current velocity of the water currents via a proxy, but rely on direct measurements [176], or do not mention what they do [177].

One of the other papers explores the impact a speed controller based on a proportional-integral (PI) controller has on P&O for tidal energy harvesting using an underwater kite [116]. The same MPPT P&O implementation was used with and without the PI controller, and the results indicate that a well-tuned PI controller managing turbine speed improves the MPPT performance, drawing 4% more power than without the controller. However, the PI controller adds additional complexity to the solution.

The newest paper examined implements an MPPT algorithm utilizing the cuckoo search with levy flight AI algorithm [152]. One of the advantages of this algorithm is that, given enough iterations, it will converge on the maximum, and it does not oscillate like P&O. The paper begun comparing its proposed implementation with P&O, but stopped short of providing experimental comparisons. Notably, direct comparisons between the amount of time to converge would have been interesting.

It is interesting to see the similarities in the calculations and the MPPT implementations between tidal and wind. We hypothesize that this is because both (broadly) work by having some medium push blades and cause a torque at the generator. More generally, we think that any energy harvesting working with a generator that produces power by rotating may all be similar enough to employ similar MPPT algorithms. Hypothetically, this means that hydroelectric generation might also be similar, but the algorithms used for hydroelectric generation are more

complex (likely due to the more complex interaction between different environmental parameters and turbine velocity).

2.4.4 Hydroelectric

Similar to tidal energy MPPT, there is a surprisingly low number of publications studying MPPT with hydroelectric power generation. As a result, we select three relevant papers between 2011 and 2021 instead of following the methodology proposed earlier. We found other papers discussing hydroelectric MPPT, but details on the MPPT implementation used were either scant or ignored.

Interestingly, all three papers we survey all describe P&O implementations. One of the papers uses an extremum seeking control (ESC) and claims that it is better than P&O, but as far as we understand it can be classified as a variant of P&O, as ESC relies on introducing a perturbation into a control signal and observing its effect [13]. This investigation into ESC does not present data backing up assertions made in the paper that this ESC approach is better than P&O– in fact, there is data also showing an appreciable amount of oscillation in the control, similar to what is expected of P&O.

Another paper tests whether P&O can be used successfully under changing conditions for hydroelectric power MPPT [14]. There was nothing particularly surprising about the results, as they are in line with other P&O outcomes from other papers and other energy sources. The investigation does confirm that P&O can identify the MPP for hydroelectric applications.

The final paper we survey presents considerably more discussion over the modeling complexities that face hydroelectric generation [72]. In particular, they provide a framework to aid in modeling future micro-hydro turbines based on analyzing known parameters of the turbine, data processing and regression, and then computing flow rates and efficiencies given speed and valve opening ratios. The MPPT algorithm in use is a typical P&O implementation, as the paper is less focused on inventing a new MPPT for hydroelectric power MPPT and more in testing their model framework.

It is not clear why there an apparent lack of interest in MPPT for hydroelectric energy harvesting. There are a lot of research opportunities in this particular field, especially given all of the other MPPT alternatives that have yet to be tried and tested. In particular, some of the more generic hybrid approaches invented for solar applications, such as the one presented by Tang et al. [153], may yield good results.

2.4.5 Thermoelectric

Up until now we have surveyed energy sources that can provide in the kilowatts of power, depending on generator/panel sizes. Thermoelectric energy sources typically produce far less power, depending on the size of the installation and the temperature differential between the contacts of the harvester. For smaller deployments, the generated power is in the order of 100s of microwatts [20, 131] to around a milliwatt [154]. Larger deployments can produce power in the hundreds of watts, however [90].

One observation made by several papers is that thermoelectric harvesters can be modeled very simply as a voltage source with a series resistor, very much like a Thevenin equivalent circuit, and that this series resistor is approximately constant over the harvester's operating range [20, 131]. As a result, it is possible to predetermine-determine the impedance the DC-DC converter driven by the MPPT system should present to the thermoelectric harvester to maximize the power harvested. This leads to simple implementations, which are ideal for harvesting scenarios where every microwatt matters.

While a single thermoelectric harvester has a single MPP, an array of harvesters, much like an array of solar cells, may have many local MPP. Two of the newer papers we survey use hybrid approaches to identify this global MPP. Zhang et al. leverage a radial basis function meshwork, a three layer ANN that models an estimate of the power curve of the thermoelectric harvester deployment, and then use a greedy search algorithm to identify the proper global MPP, using the model output to estimate the global MPP [175]. The ANN is trained every time a large enough change in temperature occurs, and then the complete algorithm is re-run to re-compute

the global MPP. Results show that this hybrid approach does converge to a solution faster than most other implementations (but not P&O), while also achieving a higher amount of power harvested, usually. The paper identifies that under some circumstances, due to overfitting, the algorithm may converge to a non global MPP. An interesting aside in the paper is that their comparisons with P&O also illustrate that P&O is faster at converging to a local MPP, and shows a tighter spread of voltages sensed when operating.

The other hybrid approach combines a GWO algorithm and linear extrapolation to identify the global MPP [90]. The GWO algorithm finds an approximation of the global MPP based on random search. With an approximation, a linear extrapolation approach leverages the linear IV characteristics of thermoelectric harvesters to identify the global MPP, which should already be close by. The new hybrid algorithm is compared against plain GWO and P&O, and as expected, P&O tends to find local maxima, and GWO on its own does eventually converge to similar results as the novel algorithm, but it takes a longer amount of time.

The last paper we study uses an open circuit voltage MPPT implementation [154]. This paper targets extremely low power energy harvesting, which explains the MPPT algorithm selection. The proposed system cold-starts at 83 mV and can continue operating after a cold start down to just under 7 mV! Prior work [114] indicates that the MPP of these thermoelectric systems can be found at 50% of the open circuit voltage, in line with observations drawn from applying the maximum power transfer theorem to a linear IV curve.

2.4.6 Radio Frequency

Most algorithms surveyed are simpler in implementation, but there was one that was more complex than the others. Specifically, Liang et al. introduce a hybrid method combining a linear searching algorithm with a binary search algorithm [109]. The key to the measurement mechanism is how much a capacitor charges over a fixed time window. The level of charge is a proxy for how much power is being harvested. The linear searching algorithm is similar to P&O, except that it is trying to identify bounds that can be used to initialize the binary search

algorithm. The system is simulated using a virtual 180 nm TSMC CMOS process, indicating that the design is possibly realizable in a chip. This implementation appears to be limited in input power between -24 and 6 dBm, which appears problematic as an RF survey from London in 2013 indicated that there were few signals with that much power available for harvesting [126]. Also, while this implementation claims to support two input sources, it does not address the problems introduced by such, specifically, that there might not be a singular MPP maximum.

The other implementations are more typical of very low power harvesting, leveraging model-based, open circuit voltage, and constant voltage MPPT implementations. One of the papers proposes using a boost converter as a resistor emulator source [123]. In this specific case, they tune the boost converter to yield a constant 750 Ω resistance, a value determined by plotting the power output of an actual antenna exposed to RF versus a variable load resistance. As with other model-based implementations, the required hardware is minimal, but the resulting power is merely an estimate of the actual MPP.

Two of the papers surveyed use open circuit voltage to track the MPP. One of the papers surveyed London for RF energy at ground level near stations of the London Underground network [126]. They designed four different antennas to target 3G, GSM900, GSM1800, and DTV signals, which their survey identified as being the most available in their areas of testing. To extract maximum power, they opted for an off-the-shelf TI BQ25504 chip, which performs open circuit voltage MPPT with a low operating power requirement (330 mV cold-start, with 80 mV hot-start or minimum once started). They determined the ideal open circuit voltage ratio that would yield an approximate MPP is for their application is between 0.48 to 0.53. They also discuss that because they do use multiple antennas, having a single MPPT controller may not yield the true MPP. They run additional experiments by giving each rectenna its own MPPT harvester, and observed that while the individual behavior improved, due to how the array was connected, some active array components provided power to others that were not actively contributing power to keep them in the hot-start region.

The other open circuit voltage paper identifies that while the impedance of a rectenna is

non-linear in general, over a narrow working range it can be approximated as a resistor [173]. They arrived to the same conclusion as Piñuela et al. [126] that the open circuit voltage ratio for the MPP is approximately 0.50. The primary contribution of the paper is one of a reconfigurable rectifier design, and not so much focusing on MPPT.

The final paper we survey uses a modified constant voltage approach [139]. They introduce an auxiliary antenna, identical to the primary energy harvesting antenna, that senses the availability of power, and based on how much voltage it detects, it selects one of three reference voltages to forward to the MPPT voltage tracker. This improves the flexibility of the constant voltage MPPT while keeping the required circuitry relatively simple.

2.4.7 Piezoelectric

Similar to RF harvesting, the MPPT methods applied lean towards simpler to implement to reduce energy requirements. One of the most cited papers uses P&O to find the MPP [101]. This is one of the first designs to perform cold-start for piezoelectric harvesting without the aid of some kind of pre-charged energy store. The system does not use MPPT while cold-starting, but once enough energy is harvested to start up the rest of the circuitry, the MPPT system starts in order to boost the amount of power collected.

Two of the papers leverage model-based MPPT algorithms. One of these models the piezoelectric source and configure a tracking system to identify the peak voltage during an oscillation period of the piezoelectric element, which, through some fine tuning of components ensures that this voltage is $\frac{1}{4}$ the optimal voltage [112]. The system then adjusts the impedance of the load by varying the duty cycle of the load DC-DC converter to bring the harvester's output voltage to match the aforementioned optimal voltage.

The second model-based paper leverages that the IV curve of piezoelectric elements is a line [50]. The system senses the voltage of the rectifier's smoothing capacitor to extract information to be used to determine the current IV curve, and from there computes the MPP. One advantage of this system is that energy is still harvestable while tracking is going on, unlike

open circuit voltage implementations.

The remaining two papers leverage open circuit voltage MPPT implementations. The first of these papers analyzes the model of their split-phase flipping-capacitor rectifier to determine that there is a constant ratio describing measurable voltages in the system and the voltage at the MPP. Empirically, for their application this is measured to be around 2.3. MPPT is not the key part of the paper (the design of the rectifier is), so discussion on how it functions is limited [34].

The final publication states that it uses an open circuit voltage MPPT, but details are scant on details on its implementation [108]. The system allow for multiple forms of inputs and multiple loads. The most notable point of the paper is the claimed 32 nA quiescent current. However, there is no discussion on whether there are multiple MPPT systems, or if there is just one shared for all, and if so, how it deals, if at all, with local MPPs.

2.4.8 Proton-Exchange Membrane Fuel Cell

One of the paper uses P&O due to the relative simplicity to implement digitally compared to other alternatives [68]. One design consideration for the P&O implementation is that the perturbation and sampling period needs to be long enough to allow the system to settle. One important consideration is that oscillations caused by power electronics can lead to safety problems. As a result, they design an analog control network to manage the duty cycle of the DC-DC converter to reduce ripple, allowing for enough margin to implement P&O.

In a completely different approach, another paper uses a radial basis function network, an ANN, to implement MPPT for a fuel cell based electric vehicle [92]. The ANN has 3 layers, with the middle hidden layer implementing Gaussian functions. However, the paper does not describe the kind of training data required for the network. They do show some comparisons against a fuzzy logic based MPPT, but once again, details are scant.

Another paper uses fuzzy logic to implement MPPT [7]. Of note, they use a differential evolution optimization algorithm to fine-tune the fuzzy logic membership functions during the design stage of implementation. Compared to many other MPPT implementations, such as P&O,

INC, and a more typical fuzzy logic system, the proposed MPPT has smaller fluctuations and tracks changes quickly.

Another paper leverages a forensic-based investigation (FBI) algorithm, along with an FPI-fractional order PID, to implement an MPPT algorithm [60]. Specifically, the FBI algorithm optimizes the parameters for the FOPID component, which in turn then controls the DC-DC converter duty cycle. The paper explains the FBI algorithm generally, but does not expand on how it is applied to the specific implementation. It is not clear to us whether this algorithm is meant to be used as a design tool, or if it is something that is always updating the FOPID, although we posit it is the latter.

The last paper we survey uses a model-based MPPT, which they refer to as a predictive MPPT implementation [59]. Effectively, they characterize the behavior of fuel cells as functions of measurable values and then use these to predict what control behavior will maximize power. As with other model-based approaches, this implementation converges quickly to the MPP.

2.4.9 Microbial Fuel Cell

As with hydroelectric and tidal energy sources, there are not many MFC publications focusing on MPPT.

One of the papers uses a model-based implementation, where they estimate that the resistance the DC-DC converter needs to be to keep the source at the MPP is a constant known at design time [155]. With that information, and some additional simplifications, all that is needed it to control the DC-DC switch duty cycle and maintain the effective resistance of the converter fixed. This is certainly a very simple implementation, which is required for the extremely low power harvestable from small thermoelectric sources and many kinds of MFCs. However, this is a somewhat rough approximation of the MPP, as shown by the results, where other implementations yield better power output.

There are three papers focusing on using open circuit voltage. One of the papers claims that MFCs behave like a voltage source, and through some additional simplifications, conclude

that the MPPT setpoint for the open circuit voltage should be 0.5 [33]. We are unconvinced that this is accurate enough for MFCs, considering that MFCs are not linear in their IV response (even if they sometimes are close [155]).

Another paper leverages an off-the-shelf TI BQ25505 chip, which implements an open circuit voltage MPPT [158]. However, no information is given on what the sampling pin on the chip is set to, so it is impossible to analyze their decision on the set point in use.

The last paper is one we published as our own early exploration of soil MFCs [119]. We used an Analog Devices ADP5091 energy harvester, which uses an open circuit voltage MPPT implementation. We did not find an MPPT setpoint that would successfully and stably harvest from a soil MFC. Specifically, soil MFCs are very sensitive to being set to a power point past their MPP, and behave pathologically, reducing both current and voltage at the same time. For comparison, most other sources trade off voltage and current, or at the very least neither move in the same direction.

2.5 Research Trends and Future Work

It is clear that most MPPT research happens with PV systems. We expect this trend to continue, as there continues to be a growing demand for solar for consumer applications, and for power generation both at residential and commercial levels.

MPPT research for tidal, hydroelectric, and MFCs, is effectively nascent, although there are techniques from other sources that can be borrowed. In particular, tidal, hydro, and wind have a lot in common due to their reliance on a spinning generator, so many techniques that apply to one apply to the others. Also, as these sources can generate power in levels comparable to solar, it may be worth investigating some of the more advanced solar MPPT algorithms, as there should be sufficient energy to power the required computational hardware.

The growing, but yet small body of MPPT literature for radio frequency, thermoelectric, piezoelectric, and MFCs are mostly using functional MPPT algorithms as these are simpler

to implement, and generally require less power. However, due to advances in semiconductor technology, more and more complex algorithms can now be implemented in low-power silicon ICs, opening the door to more complex MPPT implementations for ultra-low power sources.

We expected P&O to be the most commonly used algorithm, yet in our survey model-based approaches were more common, per table 2.1. There is also a growing number of hybrid approaches developed for multiple sources. A major driving force behind hybrid approaches is tracking the true MPP when combining multiple energy sources, as their power curves combine to create many local power maxima.

2.6 Conclusion

We survey 41 MPPT publications between 2000 and 2020 across nine energy sources, and identify patterns in the use of MPPT across these sources. Solar is the most studied, and has the most advanced algorithms available, yet most of these can be adapted for other energy sources. Wind, tidal, and hydroelectric sources are similar enough that advances in one should also benefit the others. Advances in semi-conductor technology should make more complex algorithms feasible for the really low power sources.

2.7 Acknowledgments

Pat Pannuto, my advisor, for pushing his students, myself included, to schedule and complete the research exam. Pat also helped with basic paper structural advice, and basic discussion on the contents of my survey.

My fellow lab mates, for helping each other with suggestions during lab meetings and encouraging each other to finish our research exams.

Ramona Gil, my mother, for coming to visit from Massachusetts and helping me make it through the research exam process, as well as some basic grammar and spelling proofreading.

Elizabeth Marcano, my spouse, for putting up with my 12-hour work days leading up to

the research exam, and for emotional support and basic grammar and spelling proofreading.

Chapter 3

Artemia: Persistent Wall-Clock Time for Intermittent Computing Applications

After trying to characterize soil microbial fuel cells and optimize energy harvesting from them for a couple of years, we search for an application of low but constant power sources [88]. In that process, we find that intermittent computing has a timekeeping problem. State-of-the-art intermittent computing time keeping solutions can track up to a few minutes at most. As a result, runtimes surrender efficiency to timing uncertainty, and applications that require accurate wall-clock time are out of reach for intermittent computing. Artemia enables battery-free, energy-scavenging systems to maintain a highly accurate time-of-day measurement indefinitely. Artemia uses constant ultra-low-voltage/power ambient energy sources to maintain state in real-time clocks (RTCs) through power failures. We build a prototype of Artemia, power it from energy harvested from soil microbial fuel cells, and deploy it in two configurations running four different sensor collecting tasks. We find that the mere microwatts of harvested power harvested from soil microbial fuel cells sustains our timekeeping subsystem indefinitely. Beyond demonstrating functionality with low overhead, we show that longer decision time horizons of Artemia enable efficiency improvements in energy use for wall-clock-centric applications—providing up to 25% more data for collection applications—and enable new classes of application for intermittent computing in deployments with no supporting infrastructure.

3.1 Introduction

Timekeeping is essential for all computing systems. It forms the basis of runtime decision-making, creates useful abstractions for programmers, and allows for robust security. The notion of a trustworthy clock is so fundamental to computer systems that many sub-systems will refuse to work if aberrant behavior is detected. Battery-free, energy harvesting devices have had to make sacrifices to operate without this essential ingredient for robust computing, which limits the classes of applications available to intermittent systems. Specifically, intermittently powered computing devices have never had a reliable local wall-clock that can give them time of day.

Battery-free embedded computing and sensing devices harvest energy from various sources (Solar, RF, Thermal, etc.) and lose power frequently (up to multiple times a second [133]) due to highly variable energy harvesting environments. Runtime systems attempt to piece together execution fragments to complete program goals promptly [77, 113]. Despite the frustration of power failures, battery-free devices represent an important emerging area for embedded systems and the Internet of Things (IoT), where billions or even trillions of new devices are expected to be deployed in the next century [3]. Battery-free operation provides relief from the hassle of changing batteries and offers a potential solution to some of our growing e-Waste problems.

Many computing systems, from the smallest 8-bit microcontrollers to large-scale servers, have an internal Real Time Clock (RTC) that maps precisely tuned crystal oscillations to a human-readable time, giving milliseconds, seconds, days, months, years, and often things like weekdays and leap years. These RTCs function as both a precise internal calendar and an alarm clock for the microcontroller. The wall-clock (or RTC) functions are essential for real-time systems: it helps manage tasks with time-series data by creating meaningful timestamps, it provides synchronization for network activities, and it allows the scheduling of longer-term events or computing tasks. Additionally, some security protocols rely on time for their security guarantees [130].

This paper presents the first wall-clock primitive for intermittent computing, Artemia.

Previous local timekeeping techniques for intermittent computing only express the time since the last (few) power failures at best. As energy runs out in a battery-free device's main capacitor, power begins to fail across all sub-systems, with larger memories and peripherals often the first to go and timing to follow soon after.

A line of work dating from some of the first battery-free platforms (Computational RFID, i.e. Moo/Wisp [29, 174]) to as recently as 2024 use "remanence timekeepers" to estimate elapsed time between power failures [16]—this is, in essence, a "stopwatch" function, only useful for near term scheduling, and always just one long shadow or long power outage away from being reset. Indeed, the Mayfly authors make this limitation explicit: "This 17 minute timing ability is more than adequate for our applications, however, for applications with longer timing requirements, new advances in zero power timekeeping are required" [78]—this work aims to fulfill this ask.

Some recent work integrates a low power RTC with intermittent systems, but only as a more precise "stopwatch" [48, 89]. In contrast, we provide a robust and reliable timekeeping module that provides true wall-clock time and can be integrated seamlessly with most intermittent systems.

Our work is built around three key ideas:

1. Reconsidering energy-scavenging sources previously ignored as "too low power"—those with orders of magnitude lower instantaneous power capability, but also orders of magnitude higher reliability in the availability of energy over long periods of time.

2. A hierarchical power tree rooted in these ultra-low power sources that (a) isolates the circuitry and power rail of the wall-clock from the rest of the device so that the clock can persist even when the rest of the device cannot, and (b) delegates control of main system power to this always-on domain via simple, runtime-configurable rules based on time and voltage.

3. Revisiting "unreliable" traditional high power sources such as solar to find that they can

also function as low power constant sources in limited circumstances.

Our first key idea relies on constant low power energy sources. Primarily, these constant sources tend to belong to a specific class of electrochemical cells, ones which generate a trivially small amount of power, but do so incredibly consistently even when environmental factors change. Recent examples include "ambient galvanic cells", which siphon energy from the galvanic corrosion of a sacrificial metal in cathodic protection systems to generate electricity (for years), and microbial fuel cells (MFCs), which convert chemical energy (in soil or water) to electrical energy by the action of microorganisms already present in the medium [89, 118]. Each of these generates 10s of microwatts, at often less than 1 volt, not quite providing enough energy to power a microcontroller continuously, but just enough (potentially) to support an ultra-low-power external RTC.

Our second key idea adds *time* as a new control channel (in addition to voltage) for an always-on supervisor. "Federating energy" is a foundational mechanism for preventing power failures [75] and apportioning energy to peripherals [38, 76]. We view Artemia akin to the charge controller and/or supervisor elements of these federated designs. Artemia adds real-world wall-clock time as an available control source to this always-on domain, which results in a platform controller that can newly use both voltage and time to manage system operation.[1]

Our third key insight is that traditional "unreliable" intermittent energy sources are not all simply feast or famine. We revisit solar energy harvesting and demonstrate that solar cells, under specific conditions, behave like constant power sources at the levels required to sustain our platform. We show how this insight enables a prior energy-harvesting "but-for-the-RTC-battery" deployment to become truly battery-free.

[1] Indeed, we view Artemia as complementary to the variety of voltage-based controllers. As we focus on the time facet of control, our implementation includes only single-threshold voltage supervision of one energy storage domain.

3.1.1 Contributions

Artemia demonstrates the viability and utility of constant but "trivial" ambient energy sources, i.e., those with mere microwatts of instantaneous power, which have largely been ignored to-date. We also demonstrate that some "intermittent" sources, like solar energy harvesting, can actually also provide this same baseline, constant power during time periods historically viewed as 'off.' We show how such constant power sources enable richer platform supervisors for intermittent designs.

In particular, this work shows how battery-free systems can, for the first time, have longitudinal, real-world, wall-clock time as a system primitive. We implement an example software runtime which includes a task and time based scheduler to enable new functions for intermittent computing, including alarms, long-term scheduling, and time-of-day events.

Finally, we test Artemia in real-world conditions. We demonstrate how wall-clock time can be used to improve the efficiency of traditional intermittent applications, achieving 25% more samples with local timestamping compared to prior infrastructure-reliant designs. Lastly, we show via a series of case studies how Artemia enables new classes of applications, and evaluate its suitability to newly enable completely battery-free monitoring in remote sensing applications with no supporting infrastructure.

3.2 Background & Related Works

Many intermittent designs aspire to provide "battery-like" operation atop more sustainable, zero-maintenance battery-free hardware platforms. Artemia gets one step closer to this dream with the addition of a reliable system clock which provides, for the first time, wall-clock time for battery-free intermittent systems.

3.2.1 Timekeeping for Intermittent Systems

In the face of unpredictable energy income, a long-standing challenge faced by intermittent systems is answering, "how long has it been since I last ran?" One early answer came from TARDIS, which observed that without time, it is hard to implement cryptographic protocols properly. To address this, TARDIS uses SRAM decay to determine the time period between intermittent operations [130]. The more general idea of using RC decay to measure intervals are today called "remanence timers" [79]. While reasonably accurate over seconds—or many minutes with careful design [46]—, RC decay becomes progressively less viable for accurate timekeeping over larger intervals. Furthermore, any decay-based timer necessarily has some maximum interval, which, if exceeded, will cause the system to lose all sense of time. With a true RTC, Artemia can provide accurate time over spans of hours or even years.

For this reason, RTCs have begun to overtake remanence as interval timers in intermittent designs. Indeed, in 2021 Jagtap et. al first demonstrated the viability of an RTC in an intermittent system to address the challenge of "how to send a message once a day?" [89]. More recently, RTCs were used for high-accuracy, short-term interval timing with fine-grained timesteps to support network synchronization for BLE [48]. None of these systems go past interval timing, however. The BLE peripheral expects to disconnect and rejoin from scratch in power droughts. Jagtap's deployed system ignores the RTC and simply sends as often as energy is available, and relies on the collecting infrastructure to timestamp samples. Artemia draws inspiration from these designs, but provides true wall-clock time rather than simple intervals.

A more recent alternative to keeping track of time locally is for networks of intermittent devices to share time. Deep et. al use simulation to show that for reasonably predictable harvesting environments, nodes can learn harvesting patterns of neighbors to maintain synchronization [51]. Find assumes a less coordinated network, with less long-term stability of energy income, and emphasizes efficient discovery among intermittent nodes, where it achieves a median discovery (and thus sync) latency of 141 s [66]. With an RTC, however, Artemia-enabled nodes do not

require any nearby partner nodes or infrastructure for timekeeping support.

3.2.2 Energy Harvesting and Management

Traditional, battery-backed systems duty cycle operation to save power. Intermittent systems and intermittent operation follow the same principle, but differ in that an energy-harvesting system may not have energy available to realize its target duty cycle, to execute off-cycle operations on-demand, and/or to preserve state between execution periods.

Software and Hardware to Handle Non-Determinism. Dealing with non-deterministic system uptime typically falls into two general approaches: hiding intermittency from software (e.g., transparent checkpointing) or forcing the burden of deep, system-specific knowledge onto developers (while giving the system greater runtime decision-making capacity) [17, 47, 78, 102, 117, 149, 172]. Mostly, these designs hide or handle execution-time non-determinism of conceptually atomic events, but none can fully hide the non-determinism of when events happen, nor in cases of extended energy loss, can they accurately infer temporal context. When the energy source is uncontrollably intermittent, so too is the system.

Avoiding Non-Determinism in Energy Income. Other designs find ways to avoid or mask the unpredictability of energy-availability. Jackson's Permamote platform adds a primary cell (a conventional battery) as an 'emergency reserve' to provide stable operation [87]. While effective, this comes with the drawbacks, risks, and eventually-finite lifetime of any battery-backed system.

Jagtap et. al argue for the use of hither-to ignored non-intermittent energy sources [88]. Table 3.1 details some of these constant-yet-small powers sources, which have historically not been utilized due to their very low power output. They advocate for these as a foundation for 'reliable intermittency', wherein events are much rarer, but are able to be deterministic in their execution (i.e., equivalent to a battery-backed duty-cycling). Follow-on work proves the concept with a corrosion-powered corrosion health monitoring system, which is able to guarantee at-least-once-daily operation [89]. While such infrequent operation may be suitable for applications such as infrastructure health monitoring, many others require a duty cycle faster than 12 μHz.

Embracing Heterogeneity with Artemia. Artemia's key innovation is that subsystems of an intermittent device can be partitioned separately across the continuous operation, reliably intermittent, and unpredictably intermittent spectrum, and that time is of particular value for continuous operation. Even with the suboptimal commercially-available energy harvesters currently available, most of table 3.1's low power sources can provide enough average power to sustain a real time clock indefinitely, and thus provide intermittent computing systems a solid source of time and long-term scheduling. While surplus energy from these low power sources can power whole systems (a design point we evaluate in section 3.6.3), the real strength of Artemia is when coupled with more traditional energy sources, where availability of wall-clock time allows for more efficient use of captured solar energy or similar.

Aside: Why not just buffer a high-power source? In principle, a monocrystaline solar cell powered system could charge a dedicated storage capacitor to, e.g., power an RTC peripheral overnight. In practice, achieving robustness falters in the face of increasingly frequent major natural events (e.g. wildfires covering the skies with smoke for prolonged periods of time). Low-but-constant power sources guarantee function in the face of adverse/unexpected events.

3.3 Design

3.3.1 Design Requirements

Artemia is a battery-free system primitive that provides wall-clock time. To provide a more concrete foundation for the design, we design Artemia to interface with a classical battery-free environmental sensing application [91].

We base Artemia's time keeping around newer RTCs that operate using mere microwatts of power [89]. To sustain the RTCs indefinitely, we rely on constant, but extremely low power sources, a sampling shown in Table 3.1. Artemia must draw no more power than the power harvested from these sources to guarantee operation.

We empirically determine a minimum power requirement of 10 µW for a constant power

Table 3.1. Constant sources of low power. We compute the minimum surface area required based on a minimum power of 10 μW as detailed in section 3.3.1. †Where only power estimates were available, we estimate the active area of the harvester as best as possible to give an estimate for power density.

Energy Source	Power Density (mW/m^2)	Surface Area Sufficient for Artemia (cm^2)
Cathodic protection repurposing [89]	100†	1
Aqueous microbial fuel cells [71]	$10-300$	$0.3-10$
Soil microbial fuel cells [118, 119]	$0.18-18$	$5.7-57$
Betavoltaic batteries [106, 148]	1–50 $mW\,g^{-1}$	$0.0002-0.01$ g
Tree trunk thermoelectric harvesting [88, 129]	$0.3-3$†	$30-300$
Thermoelectric from solar panels [11]	50	2
Air-gen [111]	~$0.0001-0.05$	$2000-\sim10^6$
Amorphous indoor solar cells [section 3.6.1]	30	3

source based on our measurement of the quiescent power draw of our prototype in section 3.5.1. Artemia must also take care to electrically isolate itself to prevent its limited power from being accidentally drained by external systems.

3.3.2 Artemia Overall Design

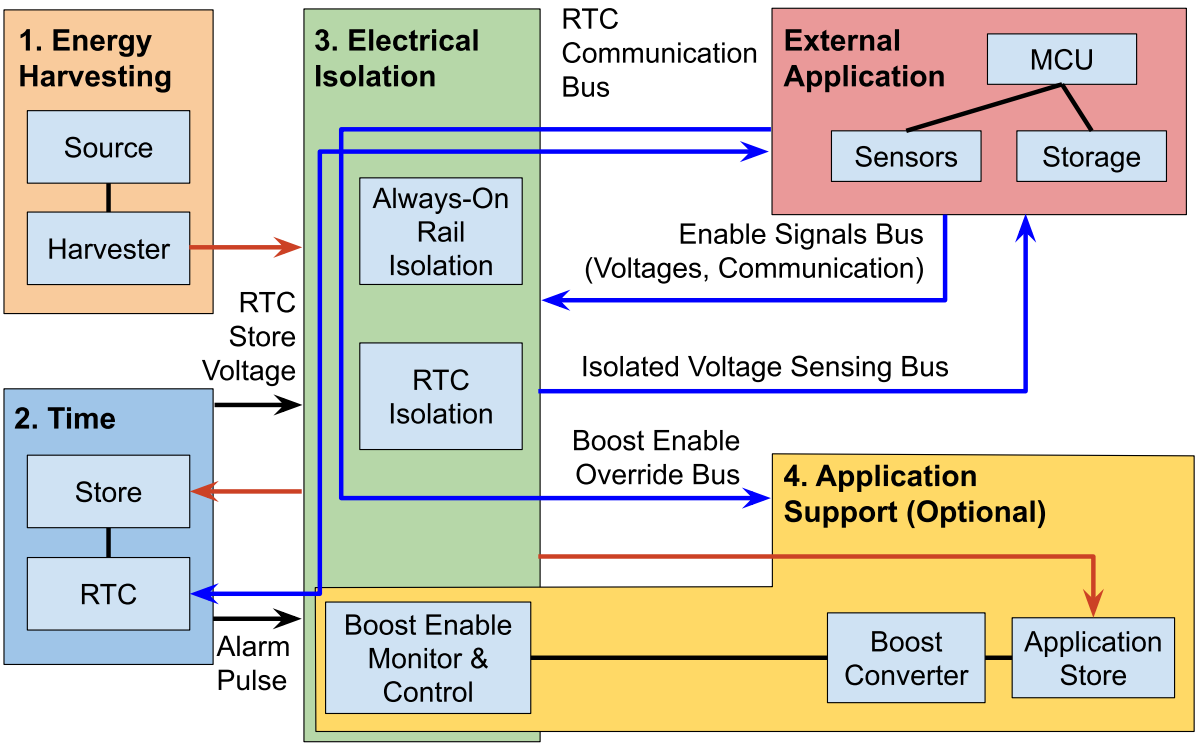


Figure 3.1. Overview of Artemia, showing the four main Artemia subsystems connected to an external application. Red arrows represent the always-on power rail, and blue arrows represent signal busses. All signals between the external application and Artemia are isolated by the electrical isolation subsystem to prevent excess power drain.

Artemia consists of four core subsystems, shown in fig. 3.1:

1. An energy harvesting subsystem to sustain system operation.

2. A wall-clock time subsystem with a crystal oscillator based real-time clock (RTC).

3. An electrical isolation subsystem to control access to the low power rail and RTC signals.

4. An optional application power support subsystem that stores energy and boosts voltage for use by an external application.

Artemia by default provides power only to the time subsystem, but with the optional application support subsystem it can provide intermittent power to an external application using surplus energy from its trickle source. However, any application drawing power from Artemia's optional application support subsystem must tolerate long periods between intermittent execution windows due to the very low power generated by the harvesting subsystem.

3.3.3 Energy Harvesting Subsystem

Most of the sources in Table 3.1 have voltages that are typically lower than most applications require, usually under 1 V. The RTC at the heart of the Artemia design (discussed in section 3.3.4) requires higher voltage than what can be provided directly by most of these sources, so we must use energy harvesting electronics to boost the input voltage. An additional and critical requirement is that the average power harvested over a window of time must exceed the average quiescent power draw of all Artemia subsystems in the same window. Many applications use the BQ25570 energy harvester [17, 47, 172]. However, its high cold-startup voltage excludes some of the low power constant sources in table 3.1. Particularly of note, the ADP5091 [89] and the Matrix Mercury [53] have both relatively low cold-start voltages, and very low cold-start power requirements. The energy harvester provides an always-on power rail for the rest of Artemia. Nominally, this rail maintains an operating voltage level at all times, but we design Artemia to only require an average power over a fixed window of time, as some power sources can be non-ideal [119].

3.3.4 Time Subsystem

The core of Artemia is the time subsystem. We base the design of this subsystem around Ambiq's Artasie RTCs, as these are the lowest power RTCs we find commercially available [89], drawing a theoretical maximum of 1 μW at 3.0 V. At typical operating conditions

(room temperature), the expected power draw is closer to 170 nW at 3.0 V. These power requirements are low enough to be sustained by many constant energy harvesting sources! As an additional safeguard, the Artasie RTCs support running from a backup energy store with built-in trickle charging of the backup. For communication and programming, variants of the RTC family support either SPI or I^2C. Additionally, the RTCs also support alarms based on calendar and time and can generate pulses when these occur. There are more features these Artasie RTCs support, but these are the key ones Artemia leverages.

The power produced by the Artemia energy harvesting subsystem should ensure the RTC has enough power to persist indefinitely. Should the always-on power rail voltage drop below the RTC's operating range, due to an intermittent power source or the application power support subsystem, the backup capacitor sustains the RTC temporarily, until the always-on power rail recovers.

We connect the RTC alarm line to the application support subsystem to make it update its control state. An application can reprogram the RTC's alarm function to schedule a wake up signal at a specific time, should there be enough energy stored to execute at that point.

Finally, the time subsystem exposes to the electrical isolation subsystem a tap into the RTC capacitor's voltage. The electrical isolation subsystem can expose this voltage to downstream applications upon request for monitoring purposes.

3.3.5 Electrical Isolation Subsystem

The electrical isolation subsystem routes and isolates power rails and other signals. The isolator components control access to critical Artemia voltage lines, namely the RTC capacitor and always-on power rail voltages. By default these signals are gated off to prevent leakage, and require active signaling from the downstream application to enable.

The main requirement on the electrical isolation subsystem is to have extremely low quiescent power draw, as it and the time subsystem are always active. Previous systems seeking to minimize quiescent draw leverage ultra-low power discrete components [49]. We originally

designed an ultra-low power latch based on existing schematics [49], but found that there are now new basic ICs, such as flip-flops, with extremely low quiescent current that can be used to implement more complex designs than with discrete COTS components. We describe our exact part selection in table 3.2 with our implementation in section 3.4.

3.3.6 Application Support Subsystem

This subsystem aggregates excess trickle power from the energy harvesting subsystem, and controls a boost converter to supply power to an external application. Its boost enable monitor and control component plugs into the electrical isolation subsystem so that it can monitor and control Artemia's power rails.

An internal voltage monitor generates a voltage-good signal once a threshold is passed. However, that alone is not enough to enable the boost converter. The boost enable monitor and control component waits for an alarm pulse from the time subsystem before enabling the boost converter, if the stored voltage is high enough.

Two control signals are exposed to the external application: a keep-alive signal, and an immediate shutdown signal. If the application support and time subsystems' energy stores are designed appropriately, an application can use the keep-alive signal to force the boost converter to remain active, even if the voltage on the storage capacitor (and thus the always-on power rail) is no longer at a safe level. This use of the boost converter allows for the extraction of more energy from the supercapacitor than would otherwise be possible due to ESR [138]. An external application can use the immediate shutdown signal to force the boost converter to turn off immediately.

3.3.7 External Applications

External applications can provide their own power, and/or they can use the optional application support subsystem to leverage the trickle power from Artemia's energy harvesting. In order to make better decisions about when to shut down or when to hold power-good high,

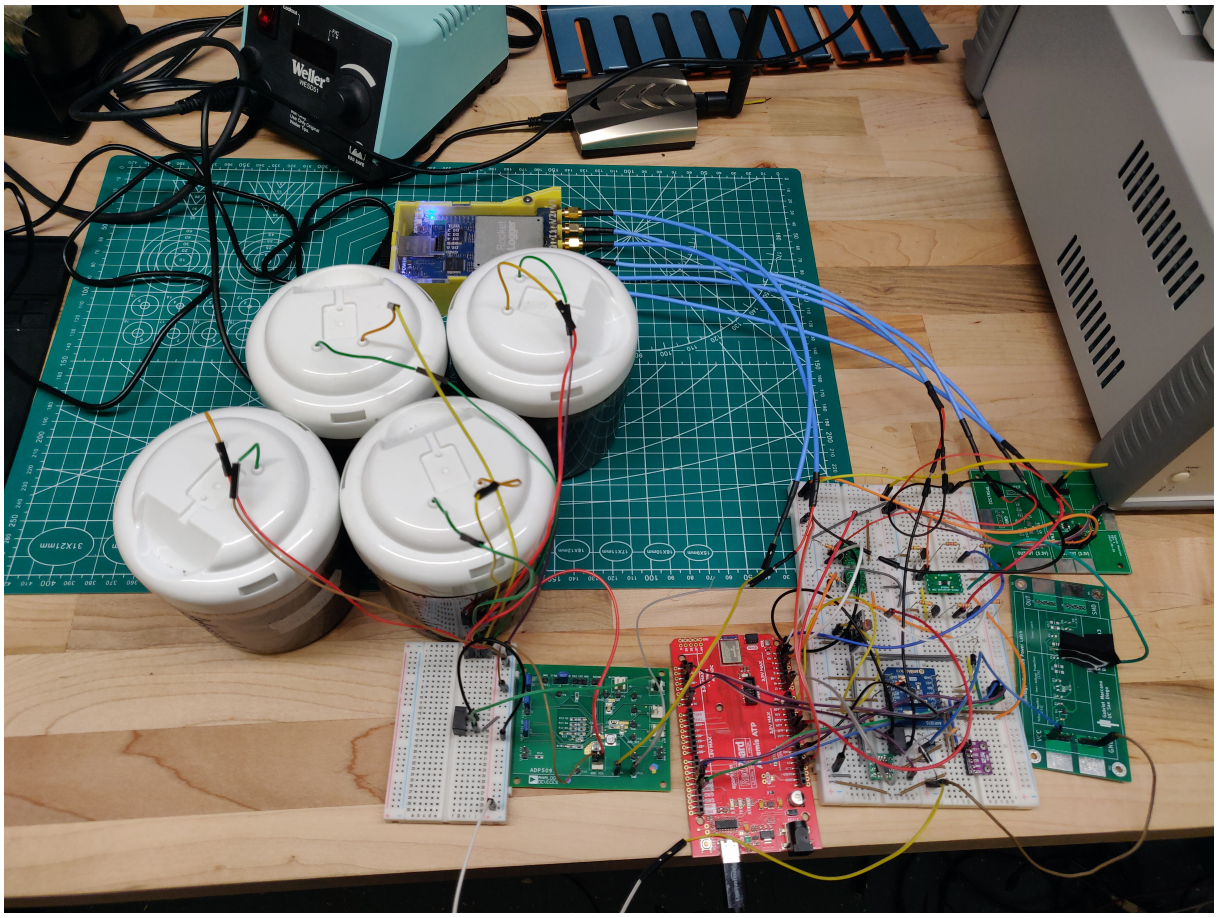


Figure 3.2. Picture of the complete Artemia prototype, configured for the experiment in section 3.5.3. The four containers on the left comprise the 300 cm^2 sMFC power source, which we connect to the Artemia prototype. The yellow device above the sMFCs is the RocketLogger we use to record voltages and current throughout the prototype.

an application may request access from the electrical isolation subsystem to the RTC capacitor voltage and the always-on power rail for monitoring purposes *only*. External applications can use the RTC communication enable line to enable communication with the RTC, allowing the applications to reprogram the RTC alarm signals and get the current time.

3.4 System Implementation

The design files and source code for this work are hosted online.[2]

We implement Artemia in two configurations, with and without the optional application

[2]Redacted for anonymous peer review.

Table 3.2. Major components used in our prototype implementation of Artemia, broken down by subsystem.

Component Type	Part
Energy Harvesting Subsystem	
Power trickle source	MudWatt
Energy harvester	ADP5091
Electrical Isolation Subsystem	
Voltage isolation	2x TPS22860
Voltage isolation	2x N MOSFETs
Voltage isolation	2x Schottky diodes
Time Subsystem	
Real time clock	AM1815SPIEVB
Backup energy store	Kemet 47 mF supercapacitor
Application Support Subsystem	
Energy store	Kemet 47 mF Supercapacitor
Voltage supervisor	TLV803EA24DCKR
Boost state control	SN74AUP1G74
Secondary boost converter	TLV61070ADBVR
External Application	
Microcontroller	RedBoard Artemis ATP
Atmospheric sensor	BMP280
Microphone	SPH0641LM4H-1 (on RedBoard)
Photoresistor	KLS6-3537
Flash	MX25V16066M2I02

support subsystem. Our prototype is shown in fig. 3.2 and component summary in table 3.2. We prepare an example software implementation that schedules four distinct tasks depending on the voltage, and thus energy, of the capacitor in the application support subsystem (or an external independent capacitor).

3.4.1 Component Selection

We build Artemia using the lowest power COTS components we find, which we list in table 3.2. The most important is the Ambiq Artasie AM1815 RTC, as described in Section 3.3.4. For the low, constant-power source, we select soil microbial fuel cells (sMFCs) due to their low cost and easy availability [56, 118, 119]; in particular, we use MudWatts, which are commercially-available, turnkey, ~57 cm^2 sMFCs [120].

Energy Harvesting Subsystem

Due to the relatively low range of voltage available from sMFCs, we select the Analog Devices ADP5091 to harvest energy. However, the ADP5091 is optimized for solar, and prior work shows that while it can harvest energy from sMFCs, it does so inefficiently (in practice, it is the 'least-bad' option) [119]. We combine four sMFCs in parallel (~300 cm^2 total cell area) to boost the current they can provide to support our application load demands [118].

Time Subsystem

We use an AM1815 evaluation board for the RTC as it is fully populated with a clock crystal for more accurate time keeping, and modify it to expose the battery pin. We attach a 47 mF^3 supercapacitor for C_{RTC} to the battery pin to hold a charge for the RTC for a long time, to allow for more flexibility when running applications from the always-on power rail.

Electrical Isolation Subsystem

We implement power domain isolation with low power switches and NFETs to prevent leakage into the external application from the RTC data interface and power rail voltage observa-

tion lines. We also use diodes to prevent anything from draining power from the time subsystem, while allowing the always-on rail or the external application to power and charge the time energy store. We also leverage a unique feature of the AM1815 to self-isolate its communication interface when running completely off of its backup battery. We expose the AM1815's input voltage as a communication enable line, so that only when an external application requests communication explicitly it will become enabled.

Application Support Subsystem

We use a voltage supervisor IC with a typical supply current of 250 nA to detect when there is sufficient charge on the main supercapacitor. We connect this voltage-good signal to the application support subsystem management component, a D flip-flop with equally low supply current and both synchronous and asynchronous inputs. We design this iteration of Artemia to require an RTC alarm pulse to trigger the application support boost converter, so we connect the RTC alarm pin to the clock of the flip-flop. The flip-flop latches the voltage-good signal when the RTC signals an alarm. The flip-flop output is then the enable signal for the application support boost converter. We expose the asynchronous clear and set pins on the flip-flop to the external application as shutdown and keep-alive signals, respectively.

We use a 47 mF supercapacitor for C_{STORE} to store enough energy to power an external application from Artemia. The trade-off is that a larger capacitor like this one takes longer to charge up as well, leading to longer periods between bursts of application activity.[3] Prior work also shows that supercapacitors have tolerable leakage, unlike large tantalums [138].

External Application

We use a RedBoard Artemis ATP board from SparkFun for our compute core, which features an Ambiq Apollo3 with an ARM Cortex-M4 MCU. The Apollo3 is one of the lowest-

[3] Trade-offs in sizing of energy stores, application flexibility, and first-run latency have been discussed at length in prior literature [38, 87, 166] and are not a focus of this design; we simply choose reasonable values that work well for our applications in empirical testing.

power MCUs commercially available, drawing around 500 µA when active, and as low as 2 µA when in deep sleep. We implement four environmental sensing tasks: We use the PDM microphone built into the RedBoard for acoustic sensing and add a BMP280 atmospheric sensor for temperature and atmospheric pressure and a photoresistor to track changes in light intensity. For state that must survive intermittent operation, we use a 2 MB Macronix flash chip.

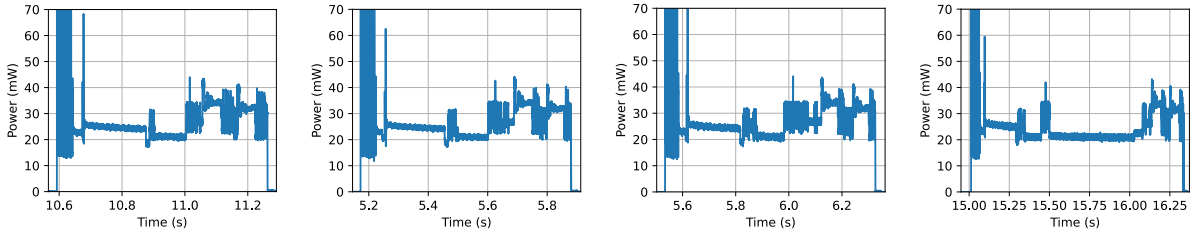
3.4.2 Test Software & Task Scheduling

The test software task scheduler implements a facility similar to the `cron` application, which we dub `scron` (i.e., 'simple `cron`'). Tasks are specified statically at compile time. A task structure consists of a task name, which must be unique, a minimum voltage below which the software will not schedule the task, a `scron` schedule, and the function to call to execute the task. Schedules are described by a simple 3-tuple of (hours, minutes, seconds). As with `cron`, these values correspond to a time when the task should run, not an interval.

The scheduler application begins by loading from storage the last time all registered task executed. It then requests the current time from the RTC and the voltage level from the always-on power rail. The algorithm next selects the task that ran last the furthest in the past and checks to see if, per `scron` rules, its next execution time has passed. If so, and the always-on power rail voltage is high enough,[4] then the scheduler executes the task's associated function. Once a task finishes, the scheduler records the time the last-executed, and then it fetches the next task and repeats the voltage and time checks. This continues until either no more tasks need to be executed per `scron`, or until the voltage level drops too low for all tasks.[5] Finally, the application scans through all tasks to determine which one is the closest one requesting execution, and an alarm is programmed with the RTC for that time. The time each task was last run is stored to

[4]We determine the safe operating voltage empirically, as shown in section 3.5.2 and similarly to prior work [138].

[5]Note that this scheduling policy could lead to starvation of other tasks if a short-lived (lower stored energy requirement) task requests an aggressive interval. The `scron` scheduler is pluggable to allow for other fairness policies if-desired. The focus of this work is not scheduler design, but rather providing the core primitive necessary to enable temporal scheduling on intermittent systems. Optimal scheduling policy is left to future work and likely is specific to application scenarios.



(a) Temperature sensing task. **(b)** Atmospheric pressure task. **(c)** Light sensing task. **(d)** Sound frequency sensing task.

Figure 3.3. Power traces of the intermittent tasks evaluated in this work. On startup, in-rush current causes a short spike to 700˜800 mW, which is clipped here for presentation. Graphs for the temperature, atmospheric pressure, and light sensing task look very similar as they take a very short amount of time to execute. Sound frequency sensing takes a longer time to sample data and process it. The burst of activity on all four graphs towards the end is Artemia writing data to flash storage.

flash, and then the application asserts the shutdown signal.

3.4.3 Tasks

We implement four environmental sensing tasks and a scheduling policy inspired by an informal sampling of typical environmental monitoring applications. We leverage the BMP280 to sense both temperature and pressure, but schedule the tasks separately from each other. The SparkFun RedBoard ATP includes a PDM microphone, which we task to capture about half a second worth of audio data which is then processed with an FFT to extract the most prominent frequency in the sampling period. Lastly, we track light intensity by reading a voltage from the voltage divider formed between a resistor and a photoresistor. Every task saves to flash the time the sampling took place and the data collected.

3.5 Evaluation

Our evaluation aims to demonstrate the viability of real-world, *always*-on battery-free timekeeping, to validate the resource requirements laid out in our design, and to show the utility of wall-clock time for real-world intermittent computing applications—in particular, that

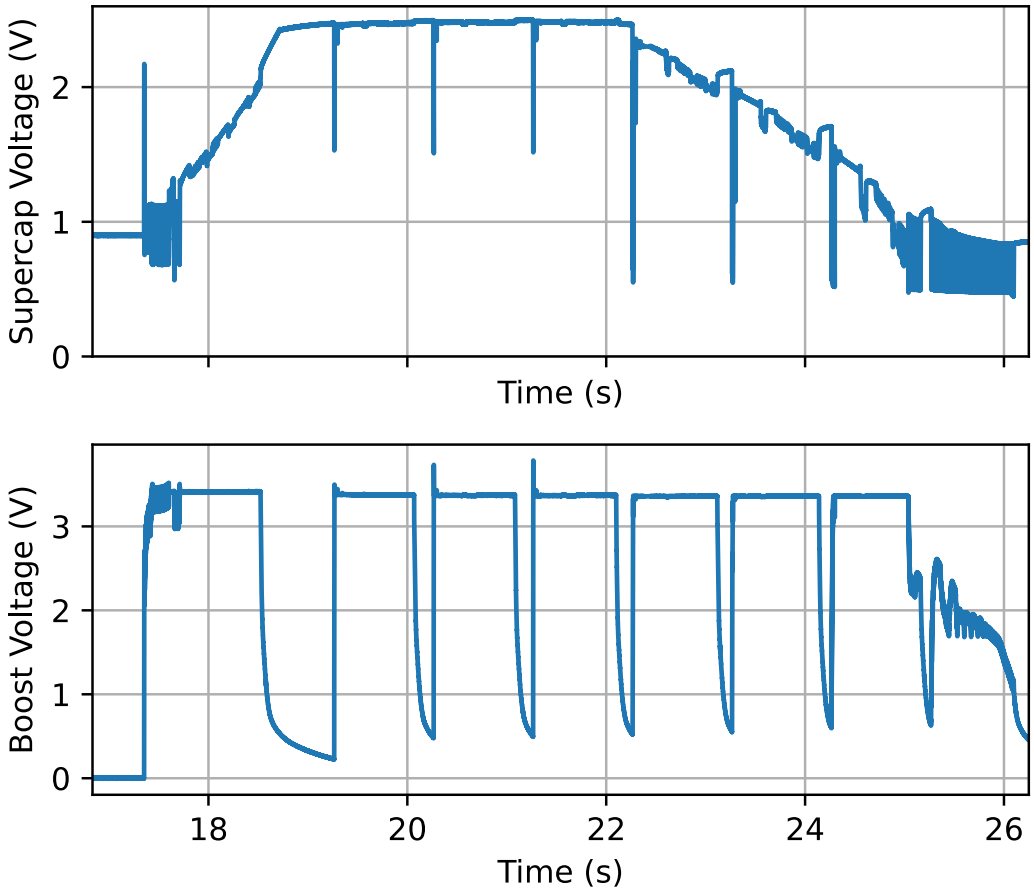


Figure 3.4. Example trace, from the 3.3V temperature task test, used to determine feasibility of running a task at a given starting voltage level. The top graph is the voltage of the supercapacitor, and the bottom graph is the voltage of the boost converter.

wall-clock time both improves the efficiency of existing applications and that it enables new application classes previously unavailable to intermittent computing systems. Unless otherwise noted, we use the implementation as-described in the previous section. For high-fidelity tracking of the very low power signals, we use a RocketLogger [144] to measure voltages and currents.

3.5.1 Artemia Quiescent Power Draw

As Artemia is always-on, it is critical for it to minimize quiescent power draw. We sweep the input voltage to Artemia to measure the quiescent power of the system before the application support subsystem turns on. We replace the energy harvesting subsystem with a benchtop power supply, and ensure that the application support subsystem maintains the boost converter enable signal as false. We sweep the benchtop power supply from 0 V to 2.4 V.

The maximum quiescent power draw occurs at 2.4 V and is 6.7 µW, which is below the 10 µW design target.

3.5.2 Energy Utilization of Environmental Sensing Tasks

In fig. 3.3, we profile the energy use of each of the tasks that run on the test application scheduler. We find that the power profile of the light, microphone, and pressure tasks are similar, each using between 20˜22 mJ, with an average power of 26˜28 mW. The microphone task uses more energy at around 32 mJ but it also takes just around half a second longer, which coincides with the sampling of around half a second of audio. Interestingly, the average power of the microphone task is around 24 mW, lower than the others, likely because it takes longer and thus that amortizes the high power startup at the beginning of execution.

To empirically derive the minimum safe voltage level for each task, we charge the energy storage supercapacitor with a benchtop voltage source to different initial voltages, and execute each task at candidate voltage levels. Using fig. 3.4 as an example, we use the voltage of the supercapacitor to identify when the voltage source is removed. At the same time, we inspect the voltage output of the application support subsystem boost converter. A successful execution

constitutes of steady boost converter output from beginning to end of a task run, which is demarcated in the top graph by the deep dips due to the boost converter acting on the ESR of the energy harvesting supercapacitor. We look through the different voltage levels until we find one where no successful executions took place once we remove the voltage source. We deem the previous voltage level as the minimum one.

We derive a minimum voltage of 1.40 V for the temperature task, 1.85 V for the pressure task, 1.69 V for the light task, and 2.08 V for the microphone task. When adding this to the Artemia software, we try to add some margin above these value to be safe, but we are limited by the Apollo 3's ADC reference voltage, which is unable to sense voltage above 2 V.

3.5.3 Full System Feasibility

Artemia relies on the availability of real-world, continuous power sources. Jagtap's corrosion harvester would have ample power for Artemia [89], but is limited to deployment only where cathodic protection systems are present. We are particularly interested in the viability of sMFCs, as, in principle, they could work anywhere where there is available soil (including underground).

We connect the full Artemia system to a 300 cm^2 sMFCs and monitor the RTC voltage for a full day, to confirm that the system can be powered from sMFCs and retain RTC state. Figure 3.5 shows the voltages of the sMFCs, the RTC capacitor, the main system capacitor, and a signal indicating when the application support subsystem is active. We show that Artemia is able to provide power to the application support subsystem, and that the RTC capacitor maintains a working level the entire duration of the test. Artemia keeps time even when the time between events is over 120 minutes. In this test the activation frequency of the application support boost converter is high enough to increase the RTC capacitor voltage above the always-on rail voltage.

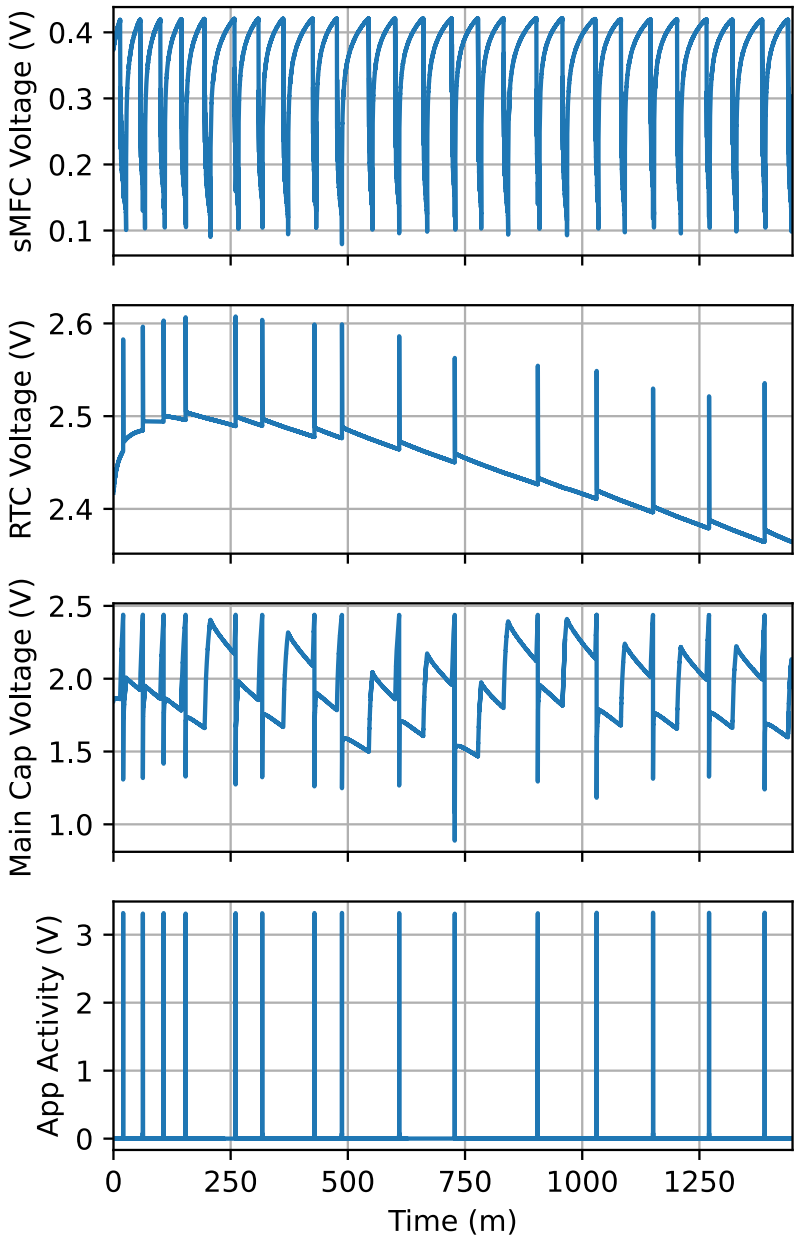


Figure 3.5. Voltage traces captured over a 24 hour period of Artemia running our test application being powered completely from 4 sMFCs. The top graph is the voltage of the sMFC array, the second graph is the voltage of the RTC backup capacitor, the third graph is the voltage of the main backup capacitor, and the final graph shows the voltage of a pin driven by the MCU when the application is active. The activation period varies, but exceeds 120 minutes at times during the experiment.

3.6 Case Studies

We examine other real applications and adapt our prototype to empirically study Artemia's feasibility and impact. We show that Artemia enables removal of a battery required in a prior, otherwise battery-free solution, Artemia improves the performance of an existing intermittent sensing system, and Artemia opens a new, historically "battery-only" application domain to intermittent systems.

3.6.1 Indoor Solar as a Constant Power Source

Prior work on battery-free environmental sensors demonstrates that if time-keeping is necessary, sometimes a battery is needed [31]. Specifically, the BigBen sensor is a light/occupancy sensor that only needs to track when a light goes on or off. To do so, it uses an RTC and a battery to retain RTC state. We posit, however, that in non-residential buildings, where some lights must remain on for safety or emergency reasons, these always-on lights may provide enough predictable power to enable Artemia to provide battery-free, always-on time.

Building codes provide a means to estimate available 'emergency lighting' power a priori, but they also vary highly by jurisdiction, building age, occupant function, and other parameters. As example, for general commercial space in NFPA 101 (a US national standard), stairwells require a minimum of 108 lux and pathways require a minimum *average* of 10.8 lux with no less than 1.1 lux at any specific point (all measured at the walking surface). Devices mounted at the wall or ceiling near emergency fixtures could thus have a surprisingly plentiful source of continuous energy in much of the modern built environment.

To test this, we connect an AM-1417CA-DGK-E amorphous solar panel (with dimensions 35.0×13.9×1.1 in millimeters) to resistors with a total resistance of 271 kΩ. We place the solar panel in a residential room with overhead lighting, approximately 8 feet away from the overhead light to match 'staircase ground' or 'hallway near-light' conditions, and leave this light on for a day and a half. Figure 3.6 contains our results. We record an average power overnight of 12 μW

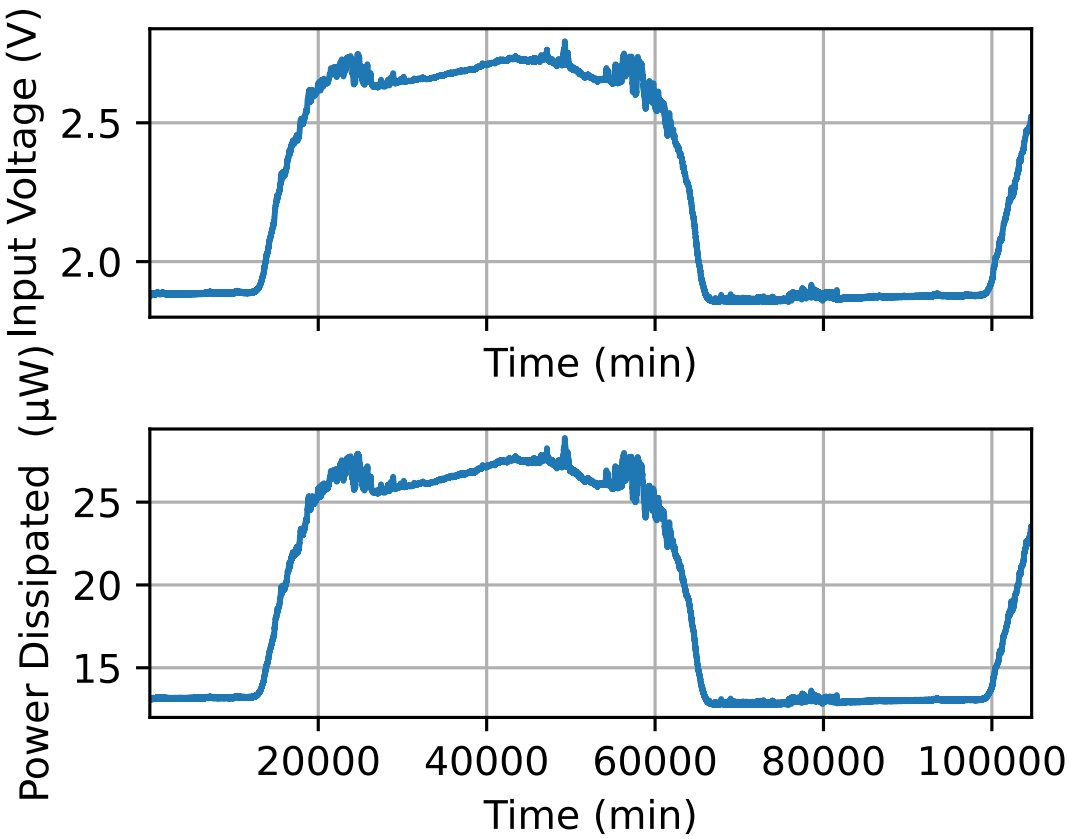


Figure 3.6. Voltage traces captured over a day and a half measuring the power dissipated through 271 kΩ resistors. A diurnal pattern is visible. The average voltage and power recorded at night are 1.8 and 12 µW, accordingly.

at 1.8 V, which is enough to power Artemia and to achieve battery-free timekeeping for select building sensors.

3.6.2 Enabling More Sampling Over Time

For many sensing applications, the time a sample is collected is a critical piece of information. We are motivated by the SONYC city-scale sensing network [22], which among other applications, measures the frequency of 'loud noise' events in urban settings to understand impact on human health. The Signpost project took this SONYC application and demonstrated that it could operate on scavenged solar power in real-world settings [2]. However, the implementation immediately transmitted each acoustic sample result to the cloud backend, relying on

the infrastructure to timestamp samples.[6]

With Artemia, we no longer need to collect data and transmit it immediately if timestamping is required, as we can timestamp, save locally, and transmit later at a more opportune time or in a larger batch. We test to see how many more packets of information we can store versus if we needed to transmit every time. We measure that it takes 6 mJ to transmit a LoRa packet at the lowest transmission power using a HopeRF RFM95W LoRa module.

We use two voltage sources and resistors to emulate power sources of different power densities, and connect them to two ADP5091s. We provide 1.5V through a 6.8 kΩ resistor to the ADP5091 connected to Artemia. We set the input of the second ADP5091, connected only to the external application circuit, to 1.5 V and a current limit of 7 mA, to emulate a higher source of power for the external application only.

We run this experiment for two hours, and measure 298 Artemia executions. From section 3.5.2, we estimate the average energy Artemia expends per execution to be 23.75 mJ. Thus, Artemia uses approximately 7.1 J over the course of two hours collecting 298 samples. If Artemia were to send every sample over LoRa to be timestamped by an external system, the LoRa system would use 1.8 J, leaving 5.3 J for sampling data. This translates to 223 samples, a 25% reduction in samples.[7]

3.6.3 Enabling New Applications

Often sensing applications look to collect data in remote environments, far from any infrastructure support. For this experiment, our application follows the operation of an AudioMoth device [80], which was first designed to be deployed in deep jungle contexts, record audio samples for several months, and later be collected by researchers. Traditional intermittent computing cannot support such application scenarios, as unknown off-time between execution

[6]The Signpost platform does also provide an option to applications for GPS time, but in practice the reliability of GPS coverage proved challenging for a few stations, so data-offloading was the preferred approach.

[7]N.b., while Artemia still needs to backhaul this data, with simple delta-encoding our system can fit all of these samples in a single SF7 LoRa packet.

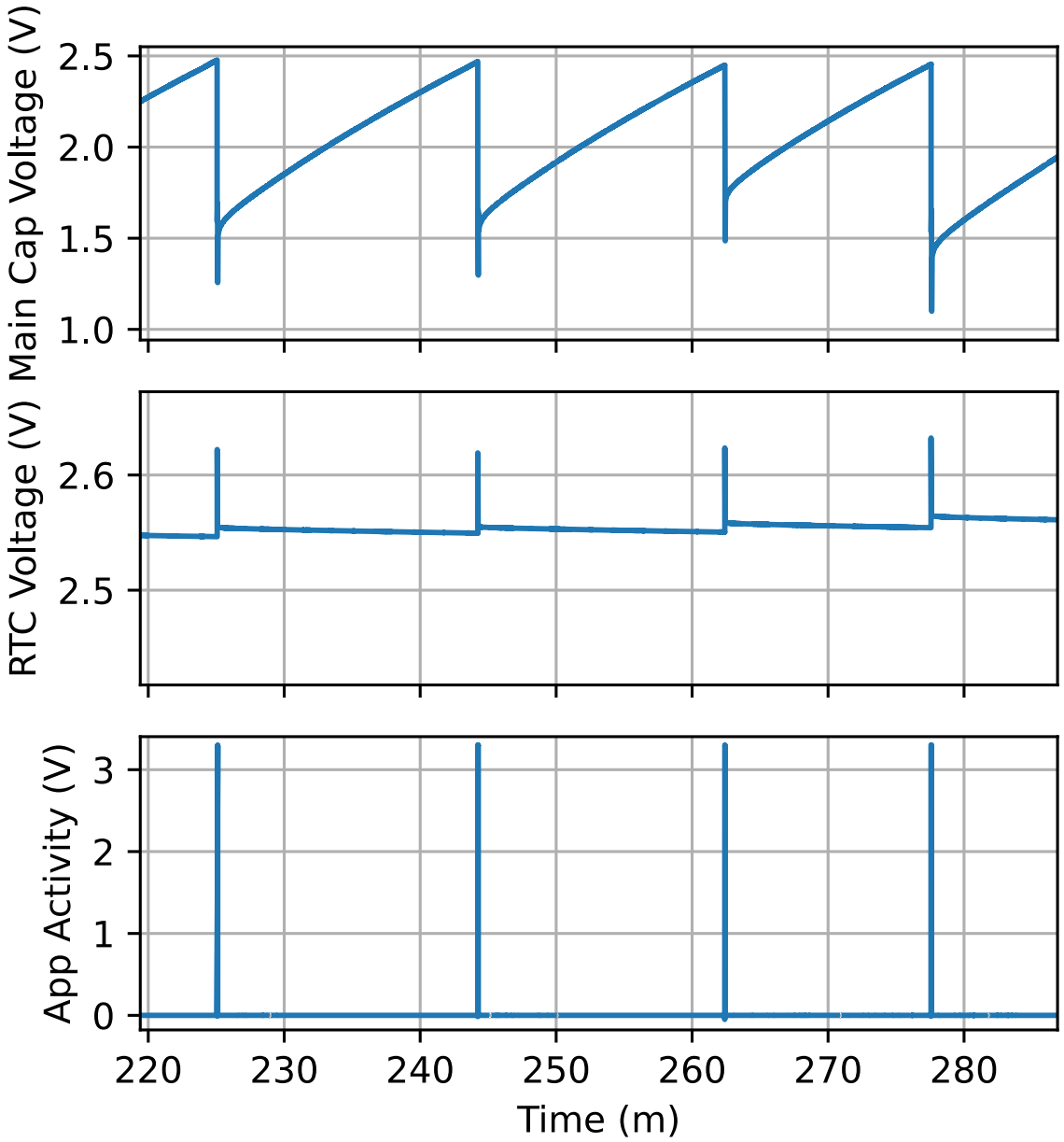


Figure 3.7. Excerpt of voltage traces captured over a 9 hour period of Artemia running our test application powered completely from a voltage power source set to 2 V through a 6.8 kΩ resistor as input to the ADP5091. The top graph is the voltage of the main storage capacitor, the second graph is the voltage of the RTC backup capacitor, and the final graph shows the voltage of a pin driven by the microcontroller when the application is active. This excerpt shows Artemia activating the external application approximately every 20 minutes.

intervals loses the context for when audio events occurred over months-long deployment.

We replicate in-lab the conditions of Artemia being deployed somewhere remote without wireless data exfiltration capabilities and limited power income—i.e., a whole system powered solely from the continuous trickle source. We connect Artemia to a 2 V source through a 6.8 kΩ resistor to limit current.

We run the experiment for 9 hours. We see that the interval between executions is 16 and 19 minutes, as shown in Figure 3.7. We confirm that the RTC time had advanced uninterrupted by comparing the initial and final RTC time values. This indicates that the time subsystem did not lose power for the duration of the experiment, and that Artemia opens the door to a new suite of long-running, unattended sensing applications previously available only to the battery-powered domain.

3.7 Discussion

We are excited to demonstrate the real-world viability of real-world wall-clock time for battery-free systems with this work. As an initial prototype, and the first to support wall-clock time for intermittent systems, there is much opportunity for future improvements.

3.7.1 Electrical Isolation is Evolving

Electrical isolation approaches are changing quickly. Prior work leverages discrete components for power control systems [49], but now we show that newer COTS ICs allow for more complex control schemes while reducing quiescent power. Additionally, some manufacturers are beginning to support electrical isolation within their offerings. We recommend checking the datasheets of ICs as it is possible that the vendor may have integrated isolation support.

As an example, the Ambiq AM1815 can completely disconnect its communication bus when powered from its backup battery, a feature we use to implement electrical isolation for the RTC. From our experiments with Artemia, each low power IC used to implement isolation adds approximately 1 μW of quiescent power draw. Thus, reducing the number of components

matters in order to limit quiescent power, especially when incoming power is around the Artemia design point of 10 µW.

Extreme care is needed when connecting different systems together, as it is very easy for a powered off domain to accidentally draw appreciable power. We observe this in an earlier prototype of Artemia, where the RedBoard Artemis draws power from the RTC through the communication lines. With microcontrollers specifically, we suspect pin protection diodes are a major cause of this leakage.

3.7.2 Extracting Energy from Low Power Sources is Difficult

Most energy harvesters available on the market are optimized for more traditional power sources, such as solar, piezoelectric, and thermoelectric. Many of the low power sources have unique characteristics making available harvesters non-ideal [119]. For instance, we use an ADP5091 solar energy harvesting controller to extract power from our sMFC array. We measure a maximum extraction efficiency of around 40%, but it could be as low as 7%! Additionally, as sMFCs do not behave like solar panels, the MPPT algorithm implementation in the ADP5091 actually over-harvests the cells, leading to the oscillatory behavior shown in the top graph of fig. 3.5. Still, the low start voltage and power of the ADP5091 make it one of the few harvesting ICs available that can extract power from sMFCs. Improvements in energy harvesters will allow Artemia to to use even lower power sources.

3.7.3 Electronics and Software Implementation Improvements

Our current implementation uses an alarm from the RTC to generate the power-good signal. Future implementations can combine external signals with voltage supervisors to make Artemia more responsive to events, instead of merely relying on the RTC alarm.

The energy use of the example application we present in this work use can be reduced by simple changes to the software stack. The test software task scheduler is not optimized or profiled to minimize runtime. The Apollo3 can sleep the ARM core while still performing I/O

with its peripherals to save power. We do not do this, and generally spinloop to check for I/O completion. For ease of programming, we did not replace the SparkFun-provided bootloader, which from empirical testing appears to take about 100 ms to finish execution, all during which we are effectively wasting power. By analyzing the application activity traces from fig. 3.7, which the software triggers after the bootloader is done, we see that the Artemia tasks execute in under 40 ms (except for the microphone task, which samples for an additional half second). Thus, the bootloader takes up an inordinate amount of the overall software time.

3.7.4 Supercapacitors and High ESR

Our use of supercapacitors with extremely low leakage exposes us to the negative effects of high ESR. As a point of reference ceramic capacitors tend to have ESRs below 0.1 Ω, if not much lower. Supercapacitors, on the other hand, range from one to hundreds of Ohms. This limits the applications that can use supercapacitors, as ESR limits the maximum current that can be drawn from them instantaneously. We can see the effects of our supercapacitor's ESR in fig. 3.4, where the voltage drops from 2.4V to at least 1.5V once the boost converter turns on, before rising back up to around 2.2V. Another issue the ESR introduces is an efficiency loss in extracting power from the supercapacitors. The supercapacitor ends up dissipating some power as heat, reducing the amount of power available to use for the application. A future design could incorporate additional tiers of capacitor banking, similar to some other prior work [38, 75], to improve the efficiency of the power transfer.

3.8 Conclusion

Control and scheduling in intermittent systems has historically been limited to only instantaneous (or very recent) voltage measures and coarse-grained estimates of immediate-past time intervals. If we embrace non-traditional, low power energy income sources, we can design mixed-operation systems that enable true always-on domains, and in-turn, true wall-clock time for intermittent designs. The availability of wall-clock time is an opportunity to reimagine the

operation of intermittent systems, and to open battery-free computing to an even richer and more capable suite of applications.

3.9 Acknowledgments

Chapter 3, in full, is unpublished work coauthored with Melody Gill, Kristin Ebuengan, Sophia Gomez, Dr. Josiah Hester, Dr. Colleen Josephson, and Dr. Pat Pannuto. I am the primary author of this material.

Chapter 4

Blinded by the MSP: Rethinking Intermittent Execution for the 32-bit, SoC Era and Beyond

We planned to investigate the benefits of memory retention through soft intermittency. As we investigate, we find that there are more benefits of soft intermittency than we originally considered. With this work, we find that the traditional intermittent execution pattern of *charge/run/die* simply does not make sense for anything beyond the legacy 16-bit MSP platforms that dominate the intermittent landscape today. The reasons are simple: cold-boot costs are going up, runtime is getting (much) more efficient, and deep sleep power draw is going down. Instead of shutting off, systems should simply enter deep sleep instead—and they can do so without sacrificing reliability or integrity.

We develop a compute-per-energy-used metric to evaluate intermittent system performance, and use it to compare sleep versus power-down execution policies across the design space—varying energy income, energy stores, compute hardware, and workloads. We then do an in-depth analysis focused on the MSP430FR5994 and Apollo3 microcontrollers to corroborate our analytical results and validate the simulations for these platforms by running actual tests on real hardware. We demonstrate that for real systems, sleeping instead of powering off can realize as much as 92% more compute per energy used, and 136% more computing time for the same input energy.

4.1 Introduction

Intermittent computing refers to the class of systems that *necessarily* operate intermittently, i.e., those where the average runtime power draw exceeds the average, steady-state system power available. Intermittency may be required to extend the lifetime of a finite energy reserve [93], to allow sufficient accumulation of charge from trickle sources [88], or to accommodate unpredictable rates of incoming energy [77]; modern designs are often a mix of all of these factors [87].

As pressures to improve both the sustainability and maintainability of electronics rise, we see a greater demand for *battery-free* architectures, and intermittent systems are growing up from a niche technical novelty to a fundamental technical necessity [3]. Simple, proof-of-possible sense-and-send applications are no longer sufficient. Today's intermittent systems are tasked with far more capable applications, such as activity recognition [19], image analyses [52], and other ML tasks [43]. As a consequence, intermittent platforms are evolving to use newer and more powerful microcontrollers.

More demanding microcontrollers running more demanding applications put an even tighter squeeze on an already-limited energy budget. Since intermittent systems are, by definition, often idle, it is critical to their performance to minimize power draw while idle—and nothing draws less power than turning off. This, coupled with the rise of new classes of non-volatile storage technologies [45], has led to a wealth of interest in 'turning off well'—ever-more-efficient checkpointing schemes [47, 162, 163], minimalist and adaptive execution frameworks [49, 78, 117], and runtimes to handle it all [17, 102, 172]. However, all this emphasis on performantly powering off has ignored the surprisingly high energy cost of turning physical hardware back on (i.e., cold-booting).

This paper investigates whether the fundamental system architecture and execution pattern that has dominated the last decade-plus of intermittent system design—charge/run/die—may be the wrong design point for the next, compute-forward, generation of intermittent computing

platforms. Indeed, much to our surprise, we find that powering off is not even the most efficient design for the current generation of systems! Many intermittent systems *should* 'fear the reaper' and should instead simply enter deep sleep between runs, like any other low-power platform.

To understand why brings the first core contribution of this paper: **We are the first to provide a detailed model and framework for optimizing compute when comparing power-off versus sleep policies for an intermittent computing system.** Section 4.3 spends significant time developing a model, and section 4.5 describes a simulation tool to accurately capture the end-to-end performance of real-world intermittent systems.

Developing this model leads to a second key contribution, **a reminder that with low-power design, the devil really is in the details.** We find that oft-ignored factors such as microcontroller cold boot time and the discharging and recharging of decoupling capacitors dwarf the energy demands of simply preserving SRAM between intermittent executions. The sum of these previously-ignored details is significant enough to indicate that *intermittent systems can get more work done for the same amount of energy if they simply suspended instead of powering off*. Notably, this result is independent of any checkpointing strategy to non-volatile storage (for robustness, which in the common-case they never need to bother recovering from; see section 4.8.4).

We perform a retrospective analysis, summarized in table 4.1 in section 4.2, that looks at fourteen systems from the last fourteen years of intermittent systems research. Our analysis yields two key results. First, of the four systems that publish sufficient detail to analyze, all but one are better served by never powering off. Second, and of some concern, the majority of works do not publish sufficient detail to accurately reason about optimal execution strategies. Systems and their evaluations focus on improvement of a specific subsystem, but lack detailed, end-to-end analysis of system performance *even as critical variables such as the main platform microcontroller are updated*. We suspect that this missing holistic analysis may in part explain why many intermittent computing systems have chosen sub-optimal execution strategies.

As experienced practitioners in low-power systems, we cast a skeptical eye on models

and simulations, especially as ours seem to buck conventional wisdom. Thus, **our third key contribution is a rigorous empirical evaluation of the accuracy of our simulations and the superiority of sleep over a power-off policy.** Our proposed model in section 4.3 enables broad analysis of system behavior while varying input power and energy stores. Our simulator in section 4.5 better addresses more complex relationships between current and voltage. Our implementation in section 4.6 and evaluation section 4.7 find that our simulations show no more than 2% error relative to empirical results, and that sleeping instead of dying consistently allows both our MSP430FR5994- and Apollo3-based platforms to perform more useful compute work given the same energy scavenging profile.

Frankly, our results are surprising. If it is really more efficient to just sleep, why do most state-of-the-art intermittent computing implementations instead charge, use all their stored energy, shut down, and then repeat the cycle? One answer may be latency, which we address in depth in sections 4.4.2 and 4.8.3. We are motivated by the increasing demand for non-trivial compute tasks on intermittent systems, and our metrics optimize for useful work done given total energy available. A sleep policy sometimes demands longer idle periods before the system can run again compared to powering off, especially for applications with little to no compute operation. Simple sense-and-send or event detection applications do not require intermittent *computers*, however. New "nano-power" analog tags provide the same functionality using orders of magnitude less energy at orders of magnitude lower economic cost [10, 73]. We believe **the future of intermittent computers is *computing*, and computing well requires re-thinking intermittent execution**.

There is one final factor that we suspect may be skewing intermittent execution policy decisions—platform inertia.[1] At the outset of the modern intermittent renaissance, i.e., the post-WISP era [29], the MSP family of microcontrollers were the only low-enough power option to realize real-world intermittent systems [98]. The release of the MSP430FR family in the mid

[1]Quite similar to that of the broader wireless sensor networking community, who stuck with MSP-based platforms such as TelosB long past their prime, limiting advancement in the field, especially low-power MAC research [97].

Table 4.1. Parameters from High-Impact Intermittent Systems Over Time. Where possible, empirical numbers are sourced from the original paper or their supporting artifacts. Shaded cells indicate those where we estimate or infer a value based on extrapolation of incomplete data from the original source, datasheet estimates based on platform components, or similar best-effort approaches. ✗'s mark where were we unable to derive a confident estimate based on available data.

	Parameters		2012 Gecko [168]	2013 Monjolo [49]	2015 UFoP [75]	2017 MayFly [78]	2018 Ink [172]	2018 Capybara [38]	2020 TotalRecall [163]	2020 ENGAGE [47]	2020 Kortbeek2020 [102]	2020 CatNap [117]	2021 Cathodic Prot. [89]	2022 Protean [17]	2022 FreeBie [48]	2024 Camel [162]
Boot	P_{avg}	(µW)	1500	1500	810	630	630	630	990	1711	630	675	1564	2250	10982	1050
	time	(ms)	2	2	1.6	1	1	1	2	78	1	0.025	78	5	✗	2
Wake	P_{avg}	(µW)	1500	1500	810	630	630	630	990	1711	630	675	1564	2250	10982	1050
	time	(ms)	0.006	0.006	0.078	0.007	0.007	0.007	0.0016	0.025	0.007	0.0067	0.025	0.035	✗	0.0016
Run Time	P_{avg}	(µW)	86000	66000	✗	✗	✗	✗	✗	11500	✗	✗	30700	✗	16000	✗
	time	(ms)	6.1	58	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Off	P_{avg}	(µW)	✗	✗	8.91	0.178	✗	✗	✗	✗	✗	✗	✗	✗	0.8352	✗
	time	(ms)	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Sleep	P_{avg}	(µW)	0.6	0.6	20	✗	✗	✗	0.3	15	✗	✗	15	✗	2410	✗
	time	(s)	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Energy Config	V_{on}	(V)	2.5	5.1	3.1	✗	✗	✗	3.3	3.97	✗	✗	3.74	✗	2.6	✗
	V_{off}	(V)	1.0	3.8	2.0	✗	✗	✗	1.5	3.3	✗	✗	2.9	✗	2.05	✗
	C_{store}	(µF)	200	500	✗	✗	22000	730	11	3300	10	1000	500000	✗	15000	47
Policy Analysis	Chosen Policy		`off`	`off`						`off`			`off`			
	$P_{harvest}$	(µW)	85[§]	23[†]						8330[*]			324			
	$P_{zzz_{better}}$	(µW)	≥64	≥16	✗	✗	✗	✗	✗	≥380	✗	✗	Never	✗	✗	✗
	Optimal Policy		zzz	zzz[‡]						zzz			`off`			

[§] The reported IV curve for the XOB17-4X3 IV (x3) photovoltaic cells used in the study show a max power of 85 µW at indoor office levels of illumination.
[†] We present $P_{harvest}$ from the lowest viable operating point, i.e., 0.008 'Wakeups/second' with 2.9 mJ events.
[‡] N.b., Monjolo's operating principle *requires* an `off` policy, even if it is not the most efficient design point (to ensure that run events occur when a precise, consistent quanta of energy has been harvested). Nonetheless, it is interesting here to observe how much is 'left on the table' by this design.
[*] The lowest input power discussed (i.e., the 20 klux scenario and assuming no button presses); other configurations even more strongly favor a sleep policy.

2010s introduced FRAM as on-chip non-volatile RAM, which a lot of work then leveraged for maintaining state across cycles [37, 172]. For compute-heavy workloads, the MSP architecture is inefficient and dated compared to modern 32-bit options, and its edge in other facets of low-power design has faded; even compiler support is becoming stale (e.g., the MSP430 GCC target has not been updated to use the Local Register Allocator in GCC, which has been preferred since 2018 [64, 65]). We suggest that it is time for the community to explore additional platforms, to allow for a new era of richer and more capable intermittent systems.

4.2 The Intermittent System Landscape

To understand the history and state of intermittent computing systems, we sample 14 intermittent computing papers from the past 14 years and extract parameters that describe the performance of the systems, including power used and harvested, and present them in table 4.1. We find that surprisingly few intermittent computing papers include sufficient information to

perform an in-depth analysis of their intermittent computing cycles. Such analysis at minimum requires: the components used (microcontroller, peripherals, and a basic idea of the power distribution system), the boot energy cost and boot time, wake energy cost and wake time, the average power used during runtime, any quiescent draw while the system is off, the power used sleeping, and the start, stop voltages of the energy store (it also helps to know what the voltage fed to the system is, if it is regulated as the power use of the system is voltage dependent). We estimate what parameters we are able to (boot energy use, boot time, wake time, etc.) by referencing the datasheets for the components. Most papers are missing key parameters, most commonly information about the expected or required minimum input power.

We do find four papers we can analyze fully: Gecko [169], Monjolo [49], ENGAGE [47], and Repurposing Cathodic Protection [89]. After applying our model, which we describe in section 4.3, we note that three of the four systems from the papers would yield better intermittent computing energy efficiency if they sleep instead of powering down, *with no changes to the software checkpointing or memory management strategy!* The exception, the Repurposing Cathodic Protection platform, uses an *extremely* large energy store, and per our model findings in section 4.4.1, causes it to not benefit from sleeping between cycles.

4.2.1 Case Study: ENGAGE

ENGAGE [47] presents a battery-free Game Boy that harvests solar and kinetic energy. It uses a 3.3 mF supercapacitor to stockpile energy, and uses a buck converter to step down the voltage to a usable 3 V. At its heart is an Ambiq Apollo3 microcontroller, configured to run at its highest speed, with a current draw of 2.5 mA. In its default configuration it powers off when it runs out of power. If instead it were to sleep, we find that the intermittent computing energy efficiency by sleeping with 1.3 mW input power equals the efficiency with the default configuration with an input power of 8.33 mW! Or in other words, sleeping requires an input power 15% the one required by the default configuration to have equal measures of intermittent computing energy efficiency. We also find that sleeping continues to yield better intermittent

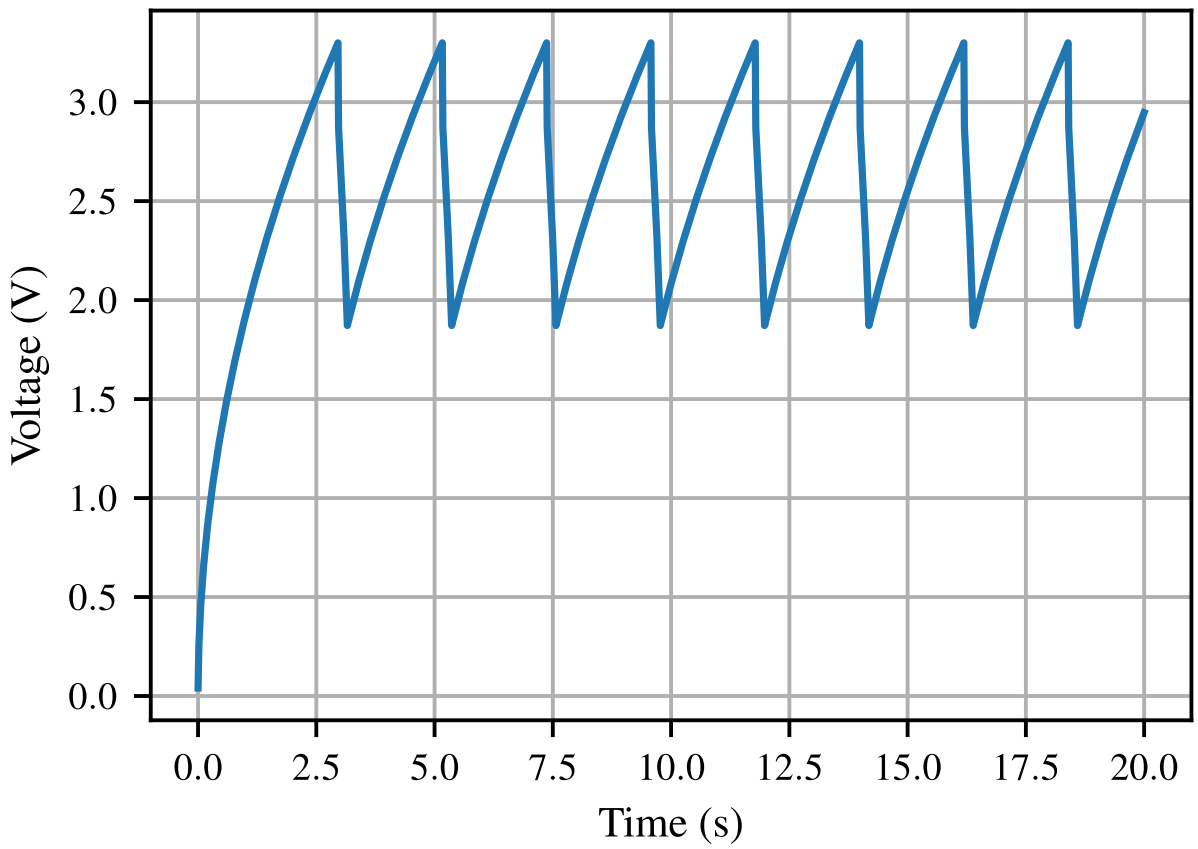


Figure 4.1. The voltage of the primary energy store across multiple intermittent computing execution cycles. The primary energy store charges to the start-up voltage (0-2.7 s), then the microcontroller powers on and uses energy until it powers off (2.7-2.8 s), then the energy store recharges (2.8-5.1 s), entering a repeating cycle.

computing energy efficiency until the input power drops below 380 µW!

4.3 Modeling and Comparing Intermittent Policies

4.3.1 A Glance at Intermittent Computing

Intermittent computing execution is cyclical (fig. 4.1). Typically, when the energy store reaches some predetermined threshold, execution begins; it stops once the energy store can no longer sustain operation. The system then harvests energy until the energy store reaches the execution threshold, and the cycle repeats. We model this cyclical pattern as two general time periods, while the microcontroller is active, and when it is inactive. We further subdivide these

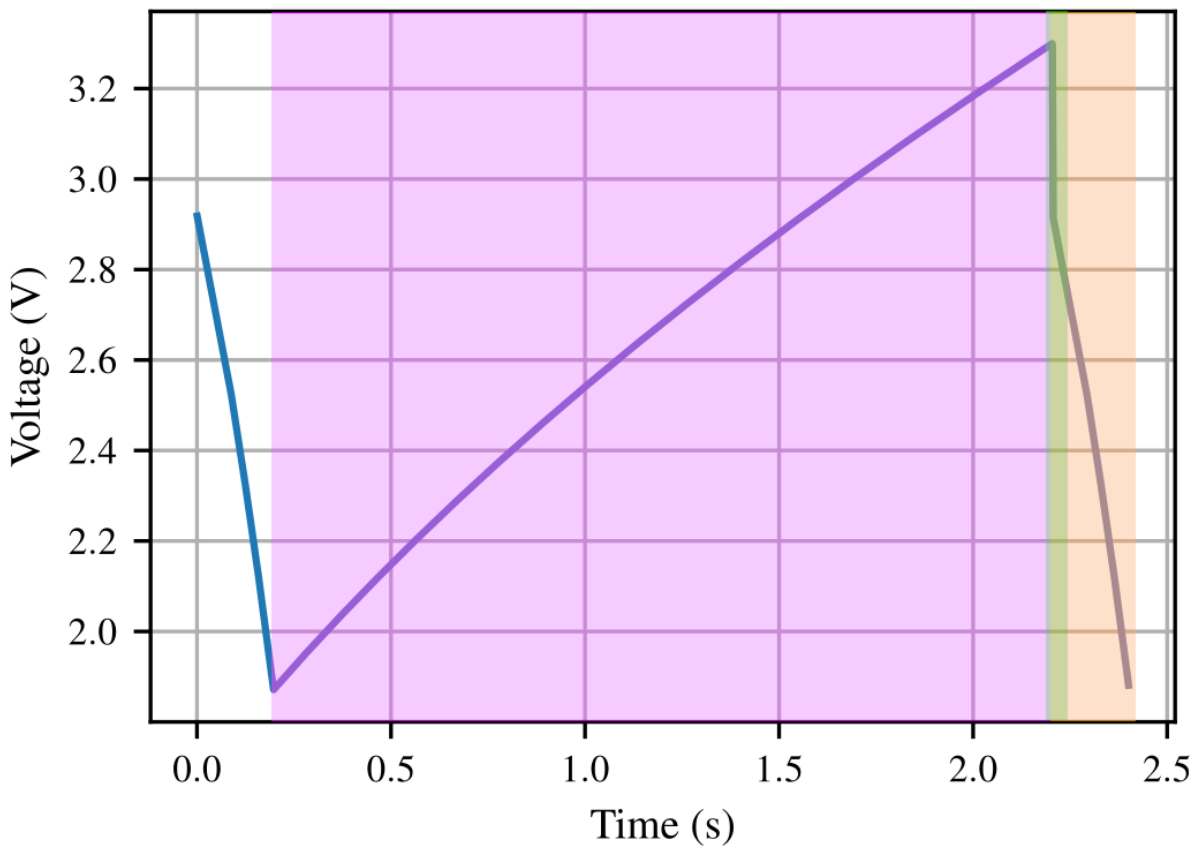


Figure 4.2. A single intermittent computing execution cycle. The violet is the charging period, the green is the boot or wake period, and the orange is the run period of the intermittent computing cycle.

periods based on system state, such as when the system is booting or waking, when peripherals are active, when the system is asleep, etc. The goal is to enable comparison of intermittent computing cycles where the system is completely off in the charging half (we refer to this as the **power-off policy**) or asleep in the charging half (we refer to this as the **sleep policy**).

4.3.2 The Devil is In the Details

The primary benefit of the power-off policy is its lack of power draw while charging the main energy store, allowing all of the harvested power to charge the energy store. In practice, the power-off policy suffers from two problems. The first problem is that cold-booting may take a non-negligible amount of time, and thus energy, reducing the amount of time spent doing useful work. The second problem is subtle, in that every time the system turns on, its decoupling capacitors need to re-energize, and run into the two capacitor paradox [178]—summarized, it observes that connecting two capacitors with different voltages together leads to a loss of up to half of the energy of the original capacitor, due to dissipation from non-ideal resistance and inductance.

In contrast, the primary benefit of the sleep policy is that it is significantly faster to wake up from sleep than to cold-boot (section 4.4.1 shows that the Apollo3 wakes up *3120 times faster* than it boots). A second benefit is that since the system does not de-energize, waking up does not incur an energy loss due to the decoupling capacitors. But, as expected, the primary drawback of the sleep policy is the additional power draw while sleeping, which decreases the net power harvested, thus increasing the charging time.

Existing models do not capture all of these details. They also often make simplifying assumptions, such as ignoring energy income during runtime (since charging periods are 'much longer'). However, as compute demand grows and increases runtime, such simplifications can introduce meaningful errors in estimations of performance. To understand the tradeoffs between execution policies, then, we need a model which captures all the fine-grained behaviors of real systems. We develop such a model next.

4.3.3 A Detail-Oriented Model

To accurately model intermittent systems, we break down the execution cycles into precise phases and account for all of the energy income and outflow during each phase. These are the three general subdivisions: 1. Boot or wake, 2. Runtime, and 3. Charging, shown in Figure 4.2. Our model considers two execution policies, power-off and sleep.

Terminology

For consistency, we explicitly define the following terms:

ΔE is the energy budgeted for use in the energy store between system start and stop.

T_1 is the set of event time subdivisions on the first half of the intermittent computing cycle.

T_2 is the set of event time subdivisions on the second half of the intermittent computing cycle.

t_i is the amount of time at time subdivision i.

E_{used} is the set of energy used on each time subdivision on both halves of the intermittent computing cycle.

P_i is the power used at time subdivision i.

P_h is the average power harvested.

E_i is the energy used at time subdivision i.

E_{h_i} is the energy harvested at time subdivision i.

For each term, we then add subscripts that indicate subdivision and policy:

b index for the boot time subdivision.

w index for the wake time subdivision.

r index for the runtime time subdivision.

q index for the quiescent time subdivision (while system is inactive).

o suffix indicating that the index is for the power-off policy.

s suffix indicating that the index is for the sleep policy.

E.g., P_{qs} describes power draw during the quiescent time subdivision for the sleep policy.

Putting It All Together

To generate our model, we derive equations for one intermittent computing cycle. To simplify the model, we assume that we know or can estimate the average power use by the intermittent computing system for each one of the four stages in a cycle. We model the two halves of the intermittent computing cycle from the energy used or charged into the energy store:

$$\Delta E = \sum_{i \in T_1} (E_i - E_{h_i}) \tag{4.1}$$

$$\Delta E = \sum_{i \in T_2} (E_{h_i} - E_i) \tag{4.2}$$

We then extract the time spent doing useful work:

$$\begin{aligned} \Delta E &= P_r t_r - P_h t_r + \sum_{i \in T_1 \setminus \{r\}} (E_i - E_{h_i}) \\ t_r &= \frac{\Delta E - \sum_{i \in T_i \setminus \{r\}} (E_i - E_{h_i})}{P_r - P_h} \end{aligned} \tag{4.3}$$

And the length of the charging period:

$$\begin{aligned} \Delta E &= P_h t_q - P_q t_q + \sum_{i \in T_2 \setminus \{q\}} (E_{h_i} - E_i) \\ t_q &= \frac{\Delta E - \sum_{i \in T_2 \setminus \{q\}} (E_{h_i} - E_i)}{P_h - P_q} \end{aligned} \tag{4.4}$$

Applying these general equations to the power-off policy:

$$t_{ro} = \frac{\Delta E - E_b + E_{h_b} - \sum_{i \in T_1 \setminus \{ro,b\}} (E_i - E_{h_i})}{P_r - P_h} \tag{4.5}$$

$$t_{qo} = \frac{\Delta E - \sum_{i \in T_2 \setminus \{q\}} (E_{h_i} - E_i)}{P_h - P_{qo}} \tag{4.6}$$

And if there is no quiescent draw while the system is off, eq. (4.6) simplifies to:

$$t_{qo} = \frac{\Delta E}{P_h - P_{qo}} \tag{4.7}$$

And for the sleep policy:

$$t_{rs} = \frac{\Delta E - E_w + E_{h_w} - \sum_{i \in T_1 \setminus \{rs,w\}} (E_i - E_{h_i})}{P_r - P_h} \tag{4.8}$$

$$t_{qs} = \frac{\Delta E - \sum_{i \in T_2 \setminus \{q\}} (E_{h_i} - E_i)}{P_h - P_{qs}} \tag{4.9}$$

We now have enough to model arbitrary intermittent computing systems.

4.3.4 A Metric for Cross-Policy and Cross-Platform Comparisons

Now that we can model intermittent computing systems, we need a way to compare them. We could use energy efficiency to compare a platform to itself, but unfortunately, it is not enough to compare different platforms. Different platforms may be equally efficient, but one may be able to do ten times more work with that energy. We propose eq. (4.10), **a metric of work per energy used in a cycle, or intermittent computing energy efficiency**, to help maximize the amount of useful work that intermittent computing systems perform in a single cycle relative to the energy used. For eq. (4.10), any measure of work done over time can be used, so long as it is consistent across platforms being compared. For this work we leverage CoreMark [63], which

Table 4.2. Microcontroller parameters extracted from datasheets. The Apollo3 datasheet does not actually have information about how long until user code starts executing, it only mentions that the system is held in reset for 30 ms on power on. From our own data collection, it seems like the Apollo3 boot ROM takes an additional 48 ms, including some time spent clearing all of SRAM.

Parameter	Apollo3	MSP430FR5994
P_{wake} mW	1.6	0.74
$P_{boot\|run}$ (mW)	1.6	2.6
P_{sleep} (mW)	0.014	0.002
t_{wake} (µs)	25	7
t_{boot} (ms)	78*	0.5
CoreMark	109.089	11.099 [57]

provides a number equal to CoreMark iterations per second.

$$\begin{array}{c}\textbf{Intermittent Computing}\\ \textbf{Energy Efficiency}\end{array} = \frac{\text{Perf} * t_r}{\sum_{i \in E_{used}} E_i} \tag{4.10}$$

We use eq. (4.10) to determine, given the amount of energy available to use and the power harvested, whether sleeping or turning off intermittent computing platforms between active times is a better use of energy for computation.

4.4 Modeling Results

Now it is time to put these models and metrics to work and answer the question of which execution policy is better for real-world microcontrollers. We select two representative microcontrollers, the venerable TI MSP430FR5994 [157], used in myriad intermittent platforms, and the Ambiq Apollo3 [8], a performant, compute-capable core optimized for very low-power operation, used in a handful of recent intermittent designs that sought to push computational boundaries. We extract parameters from the datasheets for both microcontrollers for comparison and list them in table 4.2. We empirically measure the boot time for the Apollo3 as its datasheet does not describe its boot ROM function. We find a CoreMark value for the MSP430FR5994 online [57]. We run the CoreMark benchmark on the Apollo3 microcontroller to compute

its value. As with other examples we find in the CoreMark database, we build the Apollo3 CoreMark program with full optimizations enabled (`-O3`) and link-time optimizations (`-flto`). We use the parameters in table 4.2 for our modeling.

4.4.1 Comparing Platforms to Themselves

The Apollo3 MCU takes approximately 78 ms to boot, versus a relatively short 25 µs to wake up. **waking up the Apollo3 is 3120 times faster than cold-booting!** The MSP430FR5994, in contrast, only needs 500 µs to boot (typical, with a maximum of 1 ms), and 7 µs to wake up. Even though it boots quickly, **waking up the MSP430FR5994 is still 71 times faster waking up than cold-booting**. Thus, we expect for both microcontrollers to experience some benefit from the sleep policy over the power-off policy, especially when the total active time in the cycle is short.

We use our model and metric to determine where both policies are equal, and explore the surrounding regions in fig. 4.3. Our modeling reveals that for these microcontrollers, the higher the input power and the smaller the energy store, the more likely it is for the sleep policy to be better than the power-off policy. Effectively, the shorter the time spent in active execution (due to low input power or a small energy store), the larger the percentage of active time is spent booting or waking up instead of doing useful work with the power-off policy. The slope of the intersection of the policies in fig. 4.3 shows that the Apollo3, with a less steep slope, benefits more from sleeping, compared to the MSP430FR5994. Notably, the Apollo3 needs significantly less input power to reach its always-on condition, which is due to its lower current draw while active compared to the MSP430FR5994.

As an example of a possible intermittent system energy store, a 100 µF capacitor, with a start-up voltage of 3.3 V and an off/sleep voltage of 1.82 V, holds 379 µJ usable by an intermittent computing system. Per the models, **the Apollo3 needs at least 25 µW input power** over the cycle for the sleep policy to be better. **For the MSP430FR5994, the minimum input power is 36 µW.**

4.4.2 Is the Latency Cost Worth It?

We use our metric specifically **to maximize the amount of work done per energy used in an intermittent computing cycle**. *This is not equal to nor translates to low execution latency!* This could mean that in extreme cases, per our metric, the better policy may have significantly worse latency! For example, our modeling indicates that the sleep policy is better for our Apollo3 model with a 100 µF store, at 30 µW input power. However, with these parameters, the Apollo3 has a latency of 23 s vs. 13 s for the off policy, which is 81% worse! Thus, for a system that cares more about latency, the power-off policy would be better in this scenario, at the expense of being able to do less work per energy. However, per data from the Bonito [67] traces for office space energy harvesting, they were able to harvest an average of 200 µW. At this amount of input power, the Apollo3 sleep policy latency is 2.03 s vs. 1.97 s for the off policy, or only 3% worse.

4.4.3 Cross-Platform Comparisons

We find that the easiest way to compare different microcontrollers is to compare all policies simultaneously graphically. We graph eq. (4.10) as a surface for both policies and both microcontrollers in fig. 4.4. The best policy at any combination of input power and energy store size is whichever has the highest metric at that point, or graphically the surface visible from above the 3 dimensional plot. The plot indicates that for almost all combinations of input power and energy store size, the Apollo3 outperforms the MSP430FR5994, regardless of which policy is used. The MSP430FR5994 only outperforms the Apollo3 when the energy store is extremely small and/or the power harvested is extremely small, solely because the Apollo3 will not boot under such circumstances. However, there are even more niche cases where even if the Apollo3 will not boot, if it has already booted (due to more power being available in a previous cycle), it can actually perform work and charge using the sleep policy! In summary, our modeling shows that **the Apollo3 performs between 20 and 40 times more work per energy used than the MSP430FR5994**.

4.5 Overcoming Model Limitations

section 4.3 provides a closed-form means for estimating the performance of an intermittent computing system. One critical requirement of this model-based approach, however, is that all required power values must be accurate averages over their respective time windows. Obtaining accurate averages is non-trivial, as, for example, microcontroller current draw is voltage dependent. Worse, many intermittent system designs power their microcontroller directly from the main energy storage capacitor, which results in a wide range of operating voltage (and thus operating current) during an execution event.

Fortunately, we can address this concern by simply shrinking the time window. As our model already breaks out each phase of execution, we can consider them iteratively in small time slices, i.e., building a simulator on top of this model, which we do next.

4.5.1 System Power and Energy Simulator

We find that accurately tracking voltage-dependent current draw is critical in practice to accurately capture performance. Thus, we build a simulator that leverages Riemann integration with small time intervals to estimate energy use and charging more accurately, especially in the face of changing voltage at the main energy store. The simulator estimates the voltage dependence of the current of microcontrollers to adjust the power draw applied per interval. It also updates its state with a configurable step size, but for our analysis we use (10 ns). At each time step, it computes the input and output energy, and updates the voltage of the energy store. It also simulates the effect of decoupling capacitors charging and being disconnected on the total energy use.

For identical energy store and input power parameters, the simulator yields metric values between 10 and 50 percent the values estimated by the model; i.e., the model alone can give a sense of things, but accurate projections require attending to the differences in power estimates due to the changing current as voltage changes. Errors in the estimated average values we use

for the model are the likely cause of the difference between the simulator and the model. For example, our model assumes 3.3 V but empirical testing shows the actual average voltage is closer to 2.7 V (this also depends on the system configuration!). However, when comparing the simulator to actual hardware in section 4.7, **our evaluation shows that the simulator's estimated metrics are within 2% of the values we extract from our physical systems.**

For our Apollo3 and MSP430FR5994 platforms, we analyze their current-voltage relationship and perform polynomial regression to create a model that provides "close-enough" current values given the current system voltage. We find that the most reliable way to collect the current-voltage data from the microcontrollers is to test them with a low impedance voltage source and sweep through the working voltage range. The sweep needs to be slow and the voltage source low impedance enough so that the decoupling capacitors on board equalize with the voltage source, otherwise the collected data will be tainted by additional current flowing through the decoupling capacitors.

4.6 Implementation

To confirm the modeling and simulation, we test the power off and sleep policies on actual hardware. **We open-source all design files, source code, and data for this work.**

Table 4.3 shows all the components that we use, and fig. 4.5 shows their setup.

4.6.1 Microcontroller Platforms

We compare a stripped-down RedBoard Artemis ATP with an Apollo3 microcontroller, shown in fig. 4.7, and an unmodified MSP-EXP430FR5994, which has an MSP430FR5994 microcontroller. The RedBoard modifications reduce the power draw of the board to only the microcontroller complex itself. Fortunately, the MSP-EXP430FR5994 development kit includes jumpers that isolate the MSP430FR5994 from the debug and flashing circuitry, so we do not need to modify it.

Table 4.3. Major components used in our evaluation, broken down by subsystem.

Component Type	Part
Energy Harvesting Subsystem	
Power trickle source	3.6 V source with a 11.9 kΩ resistor in-series
Power Control	
Power isolation	LM358 to buffer energy store voltage
Voltage Supervisor	RedBoard Artemis ATP
Power Switch	TPS22860DBVR
Signal conditioning	15 kΩ, 18 kΩ voltage divider
Energy store	100 µF electrolytic capacitor
Filtering	2x 100 nF ceramic capacitors
	100 µF electrolytic capacitor
Microcontroller Platforms	
Microcontroller	RedBoard Artemis ATP (modified)
Microcontroller	MSP-EXP430FR5994 devkit

4.6.2 Power Control

For simplicity, we implement a software solution for the voltage supervisor on another unmodified RedBoard Artemis ATP. This supervisor RedBoard monitors the voltage on the energy store capacitor (with the help of a voltage buffer to prevent loading the actual energy store), controls a switch connecting the energy store to the microcontroller, and sends sleep/wake control signals to the microcontrollers. Specifically, for the power-off policy, it connects the capacitor to the microcontroller when the voltage is greater than approximately 3.3 V and disconnects it when the voltage is approximately 1.82 V. For the sleep policy, it sends a wake-up pulse to the microcontroller when the voltage is greater than approximately 3.3 V, and a sleep pulse when the voltage is approximately 1.82 V. Just in case, with the sleep policy, if the voltage drops below 1.75 V, the monitor disconnects the energy store capacitor from the microcontroller and reconnects it once the voltage rises above 3.3 V.

4.6.3 Emulating Low Input Power

We use a benchtop voltage source set to 3.6 V in series with a 12 kΩ to form an RC circuit with our 100 µF capacitor. In this configuration, we estimate an average power between 150 to 200 µW, in line with the power harvested in an office setting by the Bonito paper [67].

4.6.4 Application Software

We configure the software running on the test microcontrollers to set a GPIO high as soon as possible after booting. We use this signal to identify the end of the boot stage and the start of the runtime stage. For the sleep policy tests we configure the same GPIO to toggle when the MCU enters and exits sleep mode. Both the Apollo3 and the MSP430FR5994 can retain their GPIO state while they sleep (with a small current cost under 1 µA).

We want to run non-trivial computations on both microcontroller boards, to approximate real-world compute workloads. We struggle to find a compute intensive application that can run on both the MSP430FR5994 and the Apollo3, with the MSP430FR5994 and its architecture being the primary limiting factor. We briefly looked at Fast-Inf [43], as it promises fast neural network inference using binary tree-based neural networks on MSP430 MCUs, but we fail to find any code to run inference on any microcontroller platform in the paper or code repository. Inspired by the Battery-Free Game Boy paper [47], we also briefly looked at emulating a Super Nintendo audio processing unit using the extremely accurate emulator by Shay ”Blargg” Green [70]. The emulator runs on the Apollo3, but it assumes 32-bit wide integers and the Super Nintendo audio processing unit needs 64 kilobytes of RAM, of which the MSP430FR5994 only has 8 kilobytes. Finally, we settle and use the tiny-aes-c library [99] to run some AES block encryption routines on both microcontrollers. We borrow NIST test vectors [122] for the ECB block cipher, to confirm correct execution, and execute ECB block encryption in a loop. We configure a GPIO to toggle every 10 ECB blocks so we can track how many are processed per cycle.

Table 4.4. Metrics from the simulator and the experiments for both microcontrollers, under both policies. There is close agreement between the simulated metrics and the actual ones, with agreement that the sleep policy yields a higher metric.

MCU & Policy	Simulated	Actual	Percent Error
Apollo3 Power-Off	29232	29282	-0.2%
Apollo3 Sleep	56650	57488	-1.5%
MSP430FR5994 Power-Off	2461	2510	-2.0%
MSP430FR5994 Sleep	2501	2539	-1.5%

4.7 Evaluation

We conduct four experiments to confirm the predicted performance of the simulations, based on the implementations we describe in section 4.6. For high-fidelity tracking of the very low power signals, we use a RocketLogger [144] to measure voltages and currents. For each experiment, we configure the RocketLogger to poll at 16000 samples per second to minimize ADC noise [39]. We collect data for multiple intermittent computing cycles. We record the energy store voltage, the microcontroller voltage, the boot-complete GPIO and ECB block activity signal, and the microcontroller low-side current. We extract a metric value for multiple cycles and average them together. We extract electrical parameters from the experimental results, such as the actual energy store capacitance and effective decoupling capacitance, and program models of these into the simulator, to test the accuracy of the simulator. Based on the data, we then configure the simulator to simulate a 105 µF capacitor and use an average input power of 165 µW to approximate the actual average power we observe from our experiments.

4.7.1 Comparing Policies On Hardware

Table 4.4 shows the simulated and actual metrics for the Apollo3 and the MSP430FR5994. Notably, the simulated results are all within 2 percent of the actual measured data. The simulator and actual hardware also agree that the sleep policy is best for both microcontrollers with the same setup and electrical parameters. And as predicted by both fig. 4.4 and the simulator, the Apollo3 in either policy outperforms the MSP430FR5994 by at least a factor of 10!

The extremely long time it takes the Apollo3 to boot vs. to wake up causes the large difference between the off and sleep policies for the Apollo3. The Apollo3 in the off policy spends almost as long booting (78 ms) as it does running (120 ms). With the sleep, policy, the wake time is fairly insignificant (75 µs) compared to the run time (280 ms). The MSP430FR5994 shows a much smaller difference due to how close the boot vs. wake times are.

For the off policy, the loss of energy due to the decoupling capacitors is non-trivial, and manifests as a sudden drop in voltage, shown in fig. 4.9. For the Apollo3, this drop accounts for 30% and for the MSP430FR5994, the drop accounts for 20% of the entire energy energy budget! We can see the effects of this energy loss the on total active time (the time spent booting and running combined). The Apollo3 spends 200 ms active with the off policy, while with the sleep policy, where it is not affected by the decoupling capacitor induced energy loss, it is active for 280 ms, or 40% longer! For comparison, the MSP430FR5994 spends approximately 90 ms active with the off policy, and 100 ms with the sleep policy, or 11% longer.

Figure 4.8 shows a couple of execution cycles for the Apollo3 and MSP430FR5994 under both policies. The primary difference between the graphs is that the Apollo3 is more efficient during execution, drawing an average of 1.7 mW, versus 3.8 mW for the MSP430FR5994. This translates to a longer active time for the Apollo3 when compared to the MSP430FR5994. The MSP430FR5994 is significantly more efficient sleeping, however, with an average of 1.5 µW, while the Apollo3 draws an average of 10 µW, almost 10x more power than the MSP430FR5994. As a result, a sleeping Apollo3 spends approximately 13% more time recharging compared to the MSP430FR5994.

4.8 Discussion

4.8.1 Intermittent Platforms & Policy Co-Evolution

Intermittent systems have always operated on (and often pushed) the boundary of hardware capability. But as new hardware capabilities emerge, so do new possibilities for intermittent

systems.

Gen 0: The Pre-Compute Era (1970-2006)

Arguably, the first intermittent 'computing' systems were early (passive) RFID tags. RFID, whose roots trace back to the first identify friend or foe systems from World War II [159], differentiated itself by 'adding programmable memory and operating from the transmitted code signal' [32]. Primarily used for tagging and tracking applications, early RFID devices inherently followed a power-off policy as they were only capable of operation in the presence of an excitation signal.

Defining Technical Milestones: Battery-free operation from RF scavenging; portable (small) non-volatile memory.

Gen 1: General-Purpose Intermittent Computing (2006-2015)

In the mid-2000s, the Intel WISP demonstrated that rising capability of energy scavenging circuitry was close enough to the falling energy demands of low-power microcontrollers that a battery-free, general-purpose computing platform was feasible—so long as it ran no more than 12% of the time [145]. Later in this time window others investigated building sensor platforms with intermittent computing systems and explored better ways to handle power [31, 49, 74, 75, 169].

Defining Technical Milestones: Communicate arbitrary, multi-bit data in response to a single RFID reader poll event; proof-of-concept sensor platforms.

Gen 2: The NVRAM Era and Powering Off (2016-2025)

Earlier systems required slower off-chip memory to retain state, or to sleep between active states. The introduction of the MSP430FR family, which included integrated NVRAM FRAM on-chip, led to an explosion of works leveraging it [78, 78, 102, 117, 163, 172]. One result is that most of these systems no longer needed to sleep to retain state, so pretty much all of them ran until their energy stores were depleted.

Defining Technical Milestones: Integrated fast and low power NVRAM; proliferation of frameworks and runtimes improving checkpointing using NVRAM; Able to run down energy stores due to not having to worry about losing all state.

Gen 3: The AI Era (2025+)

While there is some work bringing more advanced applications to MSP430FR platforms [43], many modern applications and frameworks assume at least a 32-bit processor, as we mention in section 4.6.4. Also, the low clock speed and low SRAM on the MSP430FRs further limits the type of applications that can run on these platforms. That said, there are now newer microcontrollers with efficiency rivaling or exceeding the MSP430FR family, including Ambiq's Apollo line (including the new Apollo510B specifically geared for AI edge computing), Onsemi's RSL15, and a handful of nRF microcontrollers with BLE functionality (such as the NRF52840 used in [48]). Most of these do not have integrated NVRAM (although, the Apollo4 and Apollo510B both also have some!), but they can leverage their faster clocks to access off-chip NVRAM at faster speeds than before. By analyzing the MSP430FR5994 datasheet we estimate that it takes approximately 4.6 nC to transfer a 16-bit word. Looking at external FRAM chips, we find some that, when clocked fast enough, require a comparable charge per 16-bit word (MB85RS64VY @ 24MHz uses 3.83 nC, CY15B128Q @ 24MHz uses 4.17 nC, and S3A1004V0M @ 48MHz uses 2.75 nC to transfer one 16-bit word). This is to say that the primary advantages of the MSP430FR family are not as unique as they once were.

In this work we find that the power-off policy is not the only option that can maximize the amount of work we can do with the energy we harvest. To be clear, there are still cases where the power-off policy is better, generally when input power is so low that charging while sleeping takes a very long time or is infeasible. But there are many other cases where we can make better use of the scant energy we harvest merely by sleeping between active cycles. This makes using a lot of the newer microcontrollers, many which have non-trivial boot ROMs, and thus take a long time to boot, more feasible.

Defining Technical Milestones: New energy-efficient and compute-capable microcontrollers; fast external NVRAM comparable to MSP430FR performance; Increased availability of BLE and other wireless technologies for connectivity at low power; sleep policy facilitates using newer microcontrollers with more complex boot ROMs.

4.8.2 Decoupling Capacitors Are Not Free

Papers like Culpeo [138] highlight that the ESR of capacitors causes dips in the capacitor voltage that can confuse voltage regulators. However, we fail to find any intermittent computing papers discussing the cost of energizing decoupling capacitors. One way to compute the energy lost due to the two capacitor paradox is to assume the conservation of charge across capacitors. We can use one of our systems as an example, and use this fig. 4.9 for analysis. We find that our Apollo3 board has approximately a decoupling capacitance of 15 µF. If we connect a 100 µF capacitor with a voltage of 3.3 , the shuffling of charge should lead to a new voltage of 2.9 V, or a 0.4 V drop which equals 60 µJ. If the 100 µF capacitor has a maximum and minimum voltage of 3.3 to 1.82 V, **the energy loss is equal to 15% of the entire energy budget!** With the sleep policy, our Apollo3 board would have to sleep for approximately 5.5 s in order to use the same amount of energy lost by charging the decoupling capacitors.

Another point of nuance, we find that the decoupling capacitors do not discharge completely when we disconnect the microcontrollers. Empirically, we see the voltage on the decoupling capacitors drop quickly until approximately 300 mV below what we believe is the internal core voltage, at which point the leakage drops appreciably. This helps reduce the energy loss on re-energizing the decoupling capacitors. The voltages are very different for each microcontroller system, however. The voltage on decoupling capacitors on the MSP430FR5994 drops to around 1.3 V, while on the Apollo3 it drops to 0.3 V.

4.8.3 Metric Limitations

In section 4.4.1 we mention that our metric optimizes for computation per energy used in a cycle, and not latency. A a simpler metric may suffice for systems that only care about latency and efficiency. Instead of using a measure of compute per time as in eq. (4.10), dropping that variable yields a metric closer to a measure of efficiency:

$$\text{Runtime per Energy} = \frac{t_r}{\sum_{i \in E_{used}} E_i} \tag{4.11}$$

Figure 4.10 graphs eq. (4.11) for MSP430FR5994 and Apollo3 microcontrollers, according to their datasheet. The Apollo3 is more efficient at runtime than the MSP430FR5994, which is reflected by the graph showing the Apollo3 being better than the MSP430FR5994 regardless of the policy used even without accounting for how much computing work is done.

If we focus solely on latency, our modeling identifies that for identical configurations using the same amount of energy, the MSP430FR5994 will always yield lower latency than the Apollo3. The Apollo3 in its off policy spends too much time booting, extending latency appreciably, and the higher sleep power draw of the Apollo3 lengthens the charging period. We emphasize, however, that if there is *any* computation required, the Apollo3 will complete it approximately 10 times faster than the MSP430FR5994, and that could bring the Apollo3 latency below that of the MSP430FR5994 if the system is allowed to power down or sleep as soon as the task is completed.

4.8.4 Future Optimizations

Determining Policies at Runtime

For a given system configuration that is already instantiated, the only variable that could change that affects our metric is input power. If a system can estimate input power, and even better, if it can predict it, it may be able to determine at runtime which policy is best for the next execution cycle and adjust for it.

An Option to Skip Checkpointing!

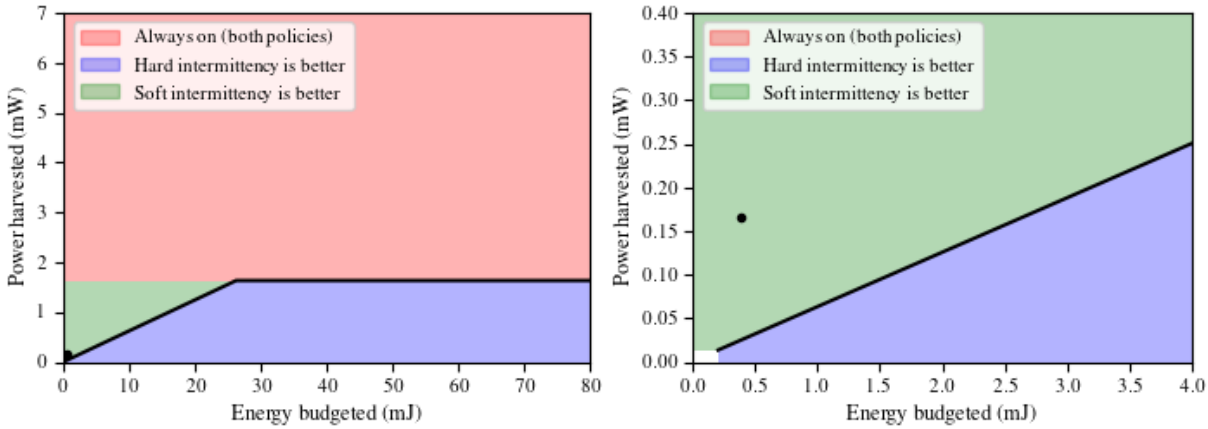
If a system supports retaining SRAM while sleeping, the sleep policy has an additional advantage over the power-off policy. In such a system, checkpointing to FRAM or some other non-volatile memory can be avoided, and resumed once input power becomes too low to sustain sleep.

4.9 Conclusion

The MSP430FR family has done a lot for intermittent computing, but per our modeling and analysis, we recommend exploring other options for newer intermittent computing applications first. The sleep policy for intermittent computing systems is viable for many cases, opening the door to microcontrollers that require a lot of power to boot, but otherwise are energy-efficient. Sleeping also brings the opportunity to explore it as a way to retain state and improve intermittent computing continuation from cycle to cycle.

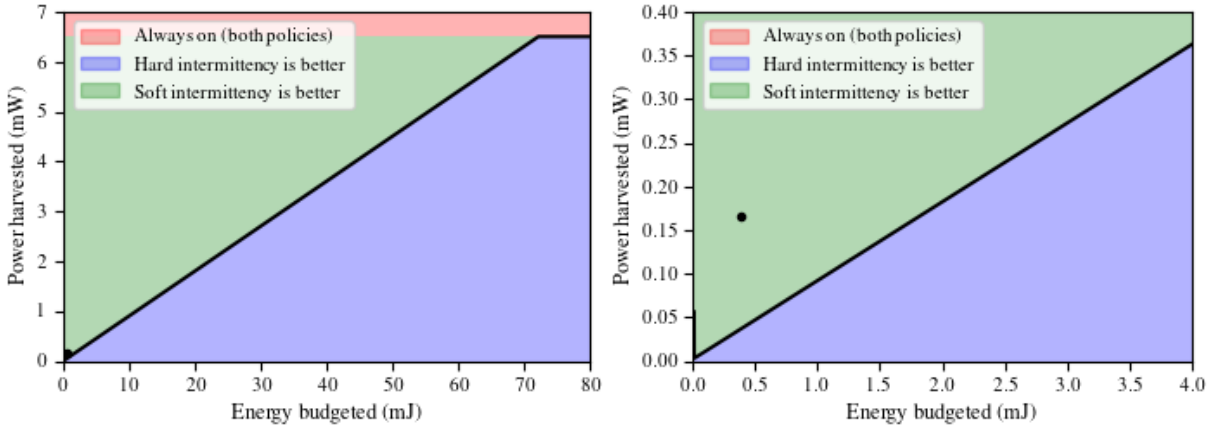
4.10 Acknowledgments

Chapter 4, in full, is my own unpublished work. Dr. Pat Pannuto is my sole collaborator on this work.



(a) Comparing sleep and power-off policies for the Apollo3.

(b) Zoomed in view of fig. 4.3a, showing the location of our Apollo3 test setup on the graph.



(c) Comparing sleep and power-off policies for the MSP430FR5994.

(d) Zoomed in view of fig. 4.3c, showing the location of our MSP430FR5994 test setup on the graph

Figure 4.3. Analytical results showing when one policy is better than the other, based on the input power and the energy store size. Generally, the smaller the amount of energy budgeted or stored for use in an energy store, and the higher the input power, the more likely the sleep policy out-performs the power off policy. After a certain input power, neither policy matters as the microcontroller is able to power on indefinitely. For the Apollo3 that is approximately at 1.6 mW (making it significantly more efficient at runtime than the MSP430FR5994), while for the MSP430FR5994 it is approximately at 6.5 mW. The Apollo3 benefits from the sleep policy due to its large boot energy cost. The MSP430FR5994 has extremely low current deep-sleep, which helps to decrease the difference between policies.

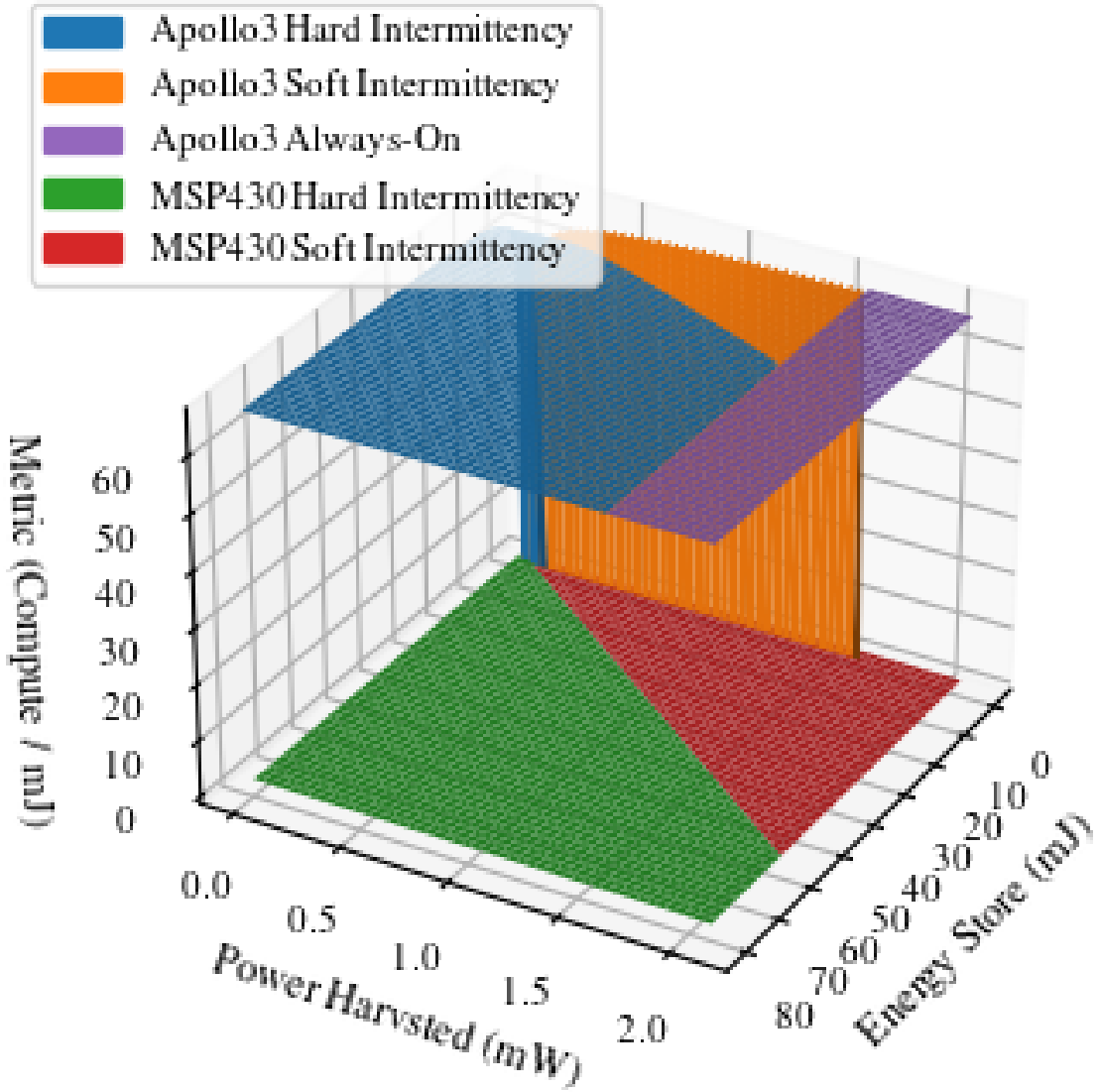


Figure 4.4. Analytical results comparing the Apollo3 and MSP430FR5994 metrics based on their datasheet values. The blue and orange surfaces are for the Apollo3 power off and sleep policies, respectively. The green and red surfaces as for the MSP430FR5994 power off and sleep policies, respectively. The MSP430FR5994 outperforms the Apollo3 when the energy store is small, such that the Apollo3 fails to boot while the MSP430 manages to do so, and when the input power is so low that the Apollo3 cannot sustain its sleep state, but the MSP430FR5994 can. There are also interesting corner cases where, for example, a combination of a small energy store and low input power means Apollo3 cannot boot, but if it is already booted, it is actually able to sleep and run!

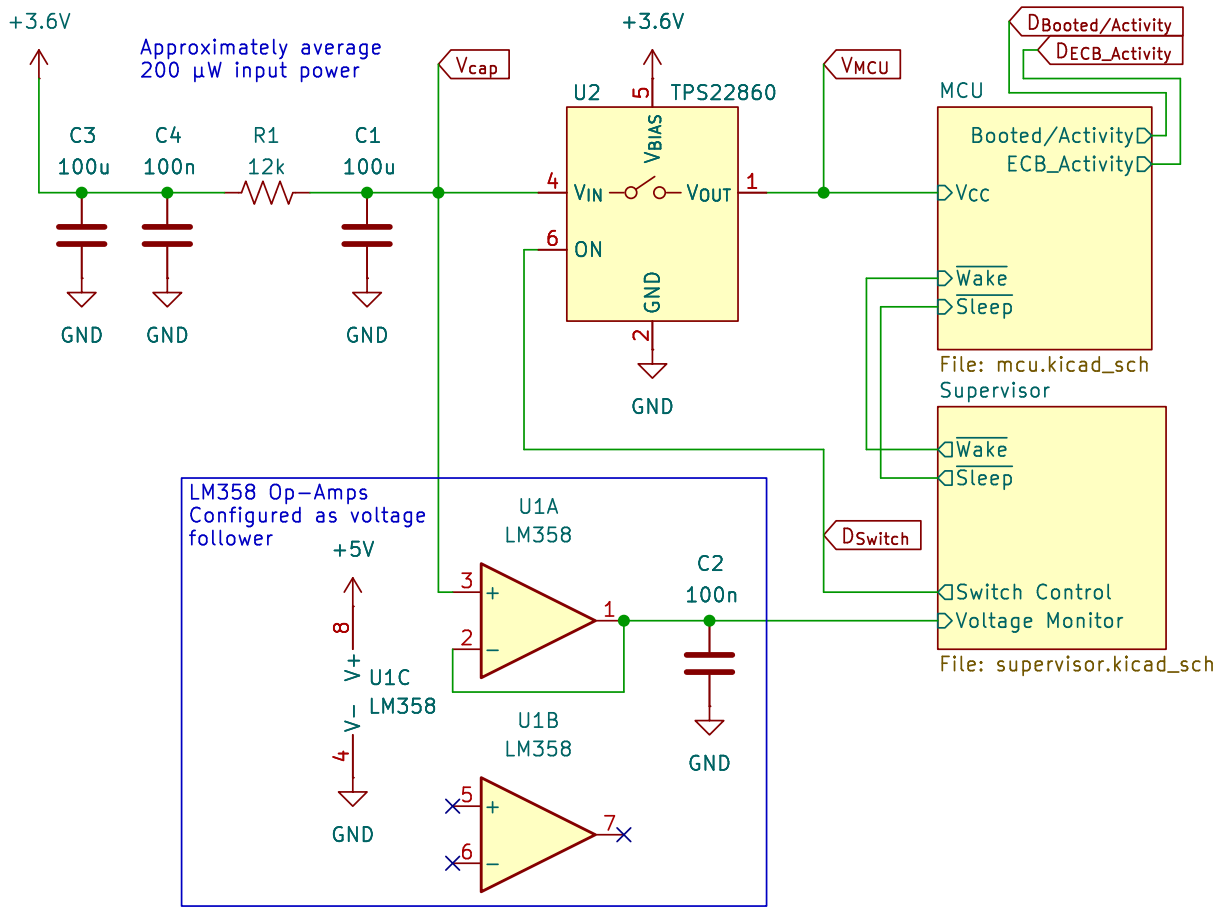


Figure 4.5. Schematic of the testing setup. A 3.6 V source with a 12 kΩ resistor approximates a solar energy harvester in an office setting, providing approximately an average of 200 µW to the 100 µF energy store capacitor. An LM358 op-amp buffers the energy store capacitor voltage, which the supervisor uses to control the switch between the energy store and the microcontroller. For the sleep policy tests, the microcontroller receives low-active pulses from the supervisor indicating when to wake up and when to sleep. The microcontroller exposes status GPIOs indicating when it is done booting, and logging application activity, which we connect to a RocketLogger for measurement as shown in fig. 4.6.

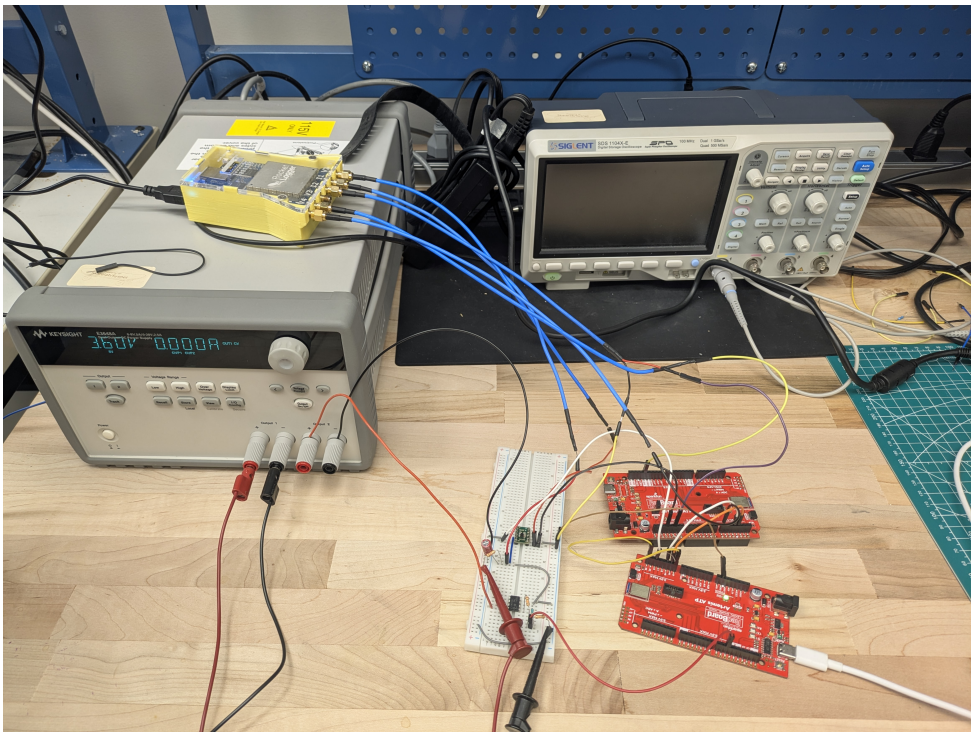


Figure 4.6. The testing setup for an Apollo3 microcontroller. The benchtop power supply provides a 3.6 V rail for the energy harvesting portion of the circuit, and 5 V to the LM358 op-amp due to its lack of rail-to-rail function. A computer provides the power to the supervisor RedBoard, and logs the supervisor activity over USB.

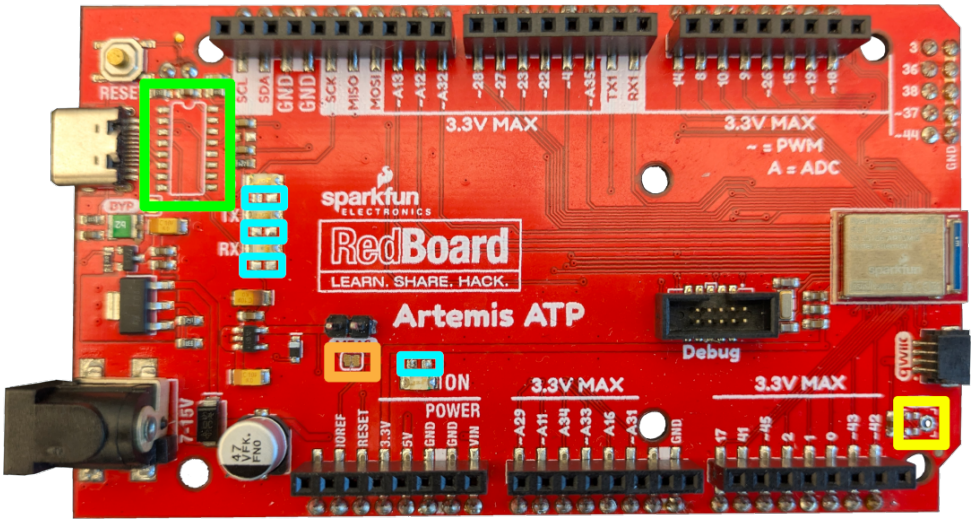
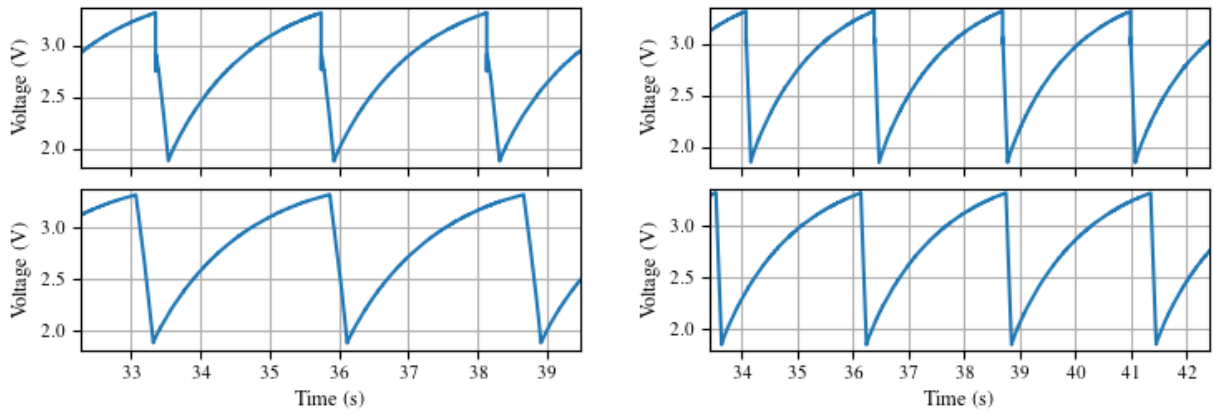


Figure 4.7. Picture of the modified RedBoard Artemis ATP. We remove the USB to UART converter (area highlighted in light green), four LED resistors (area highlighted in light blue), the MEMS microphone (area highlighted in yellow), and cut the jumper connecting the Apollo3 main voltage rail to the voltage regulator used with USB power (area highlighted in orange).



(a) Zoomed in portion of the Apollo3 executing in both the power-off policy (top) and the sleep policy (bottom).

(b) Zoomed in portion of the MSP430FR5994 executing in both the power-off policy (top) and the sleep policy (bottom).

Figure 4.8. Zoomed in portions of both the Apollo3 and MSP430FR5994 executing in both the power-off policy and the sleep policy, showing the difference in latency from one cycle to the next. Per table 4.4, the sleep policy out-performs the power-off policy. However, the charging time of the power-off policy is certainly shorter than the charging time of the sleep policy. A sharp drop in voltage is notable in the power-off policy execution, due to the energizing of the decoupling capacitors. This sudden drop in voltage is absent in the sleep policy execution.

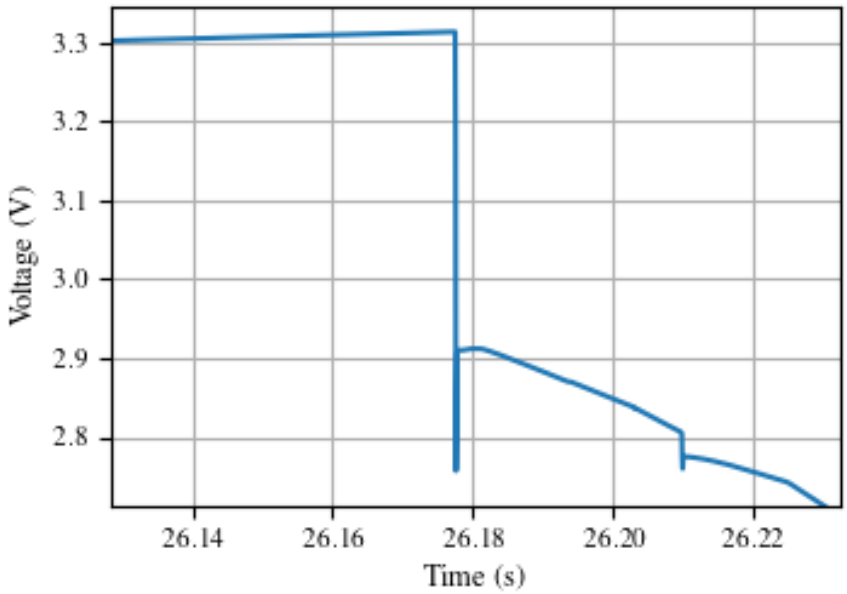


Figure 4.9. Zoomed in portion of the Apollo3 in the power-off policy to show the effect of decoupling capacitors on the main energy store at start-up time. We actually see two effects: 1. an overshoot in voltage when the system is energized, which fits the descriptions in [138] regarding the effects of capacitor ESR, which is expected as we use an electrolytic aluminum electrolytic capacitor, which are known to have more ESR than MLCCs or tantalum capacitors. And 2. voltage stabilizes around 2.9 V before being drained down by the Apollo3 booting. This voltage drop of 400 mV equals to a loss of approximately 60 µJ, which is 15 percent of all the energy the capacitor holds between the operating voltages of 3.3 and 1.82 V!

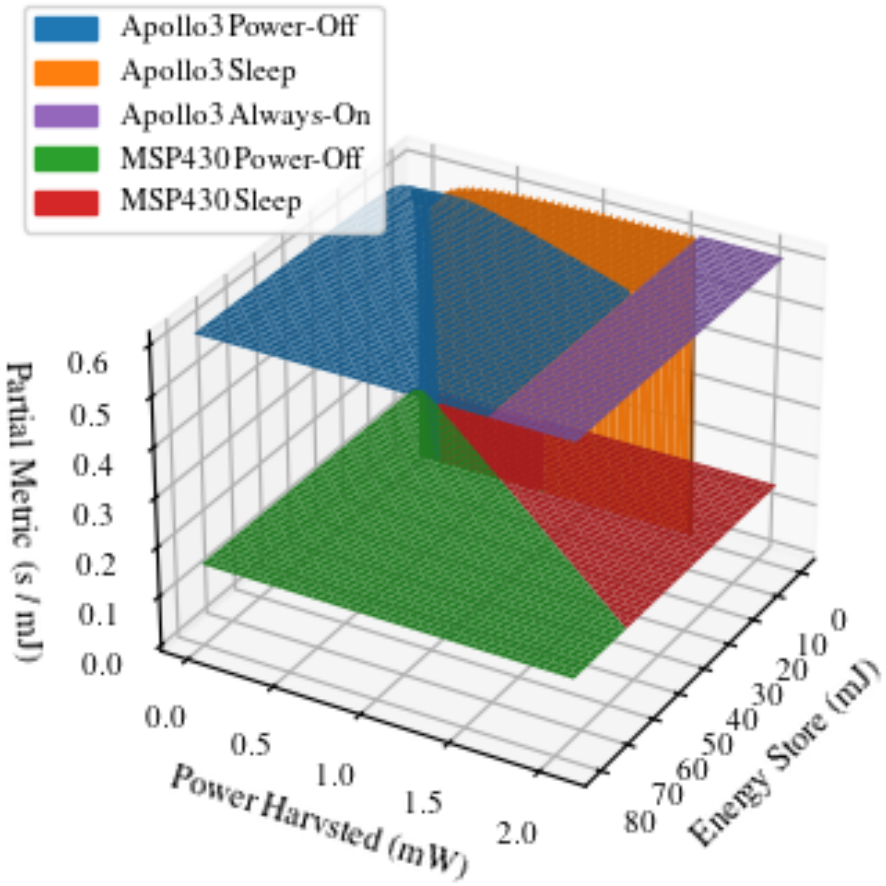


Figure 4.10. This shows the theoretical time over energy used per cycle for the Apollo3 and MSP430FR5994 (without performance). This shows that even in this scenario, where compute does not matter, it usually makes more sense to use the Apollo3.

Chapter 5

Future Work & Conclusion

Intermittent computing systems are ideal for large-scale deployments, so long as they can power themselves and have ways to manage their inherent intermittency. Chapter 3 introduces Artemia, a wall-clock primitive that we can use for intermittent systems, giving us a good sense of time and the ability to schedule tasks or events into the future. We can use a good wall-clock time source to synchronize with other devices, and engage in more complex network protocols, as I propose in section 5.1.

Chapter 4 shows that soft intermittency is a feasible intermittent computing policy, especially in the presence of a known minimal average input power. Low power sources can provide this minimal average input power, but although they are more consistent than, say, solar energy harvesting, they may be restricted to specific locales, or may suffer changes in output over time. This makes me consider batteries again, but this time in the context of using them as low power trickle sources, which I describe in section 5.2.2. I then present as future work a tentative intermittent computing system design, which I describe in section 5.2, that combines my thesis work with batteries to further reduce the complexity of managing memory and time.

5.1 Enhancing Networking

CHRT [170] is effectively the state-of-the-art for time keeping for intermittent systems, but it only provides a relative time from when power loss happened. FreeBie [48] takes it a step

closer to Artemia and shows that an RTC can enable an intermittent computing system to retain a Bluetooth connection even in the face of intermittency. With Artemia, we can guarantee that the RTC always retains power. We can then extend the concepts introduced with FreeBie to other protocols. For example, LoRaWAN has three classes:

- Class A – fully intermittent, only listens for downlink right after an uplink
- Class B – Same as class A, but with additional known downlink windows
- Class C – Fully online, always listening for downlink when not transmitting

Additionally, Class B devices continuously keep their clocks in sync with the network by having to listen for time-synchronized beacons every so often. Thus, by having wall-clock time, we can upgrade a class A device to class B by managing stored energy so we can wake up to synchronize with beacons, and then handle possible downlink transmissions.

5.1.1 Upgrading to a Class B Device

A class B LoRaWAN device needs to listen to beacons, which are transmitted every 128 seconds by the network, to help keep its clock in sync. Then, every 128 second beacon period is broken up into 4096 ping slots. The network and device negotiate the ping period for the device, and from then on the network and device follow an algorithm to pick a slot randomly, to try to avoid collisions with other devices. This requires the device to have a notion of time, which Artemia provides. A microcontroller can then register with Artemia to be woken up right before the ping slots.

A class B device does incur additional overhead, so it may need an improved energy management system. Notably, LoRa radios draw more current than many modern low-power microcontrollers (10× an Apollo3 MCU, for example), so a energy management system like UFoP [75] or Capybara [38] would help ensure there's enough energy scheduled for use by the radio. Additionally, class B function does impose a minimum amount of power harvested. The

amount of power depends on the LoRaWAN network configuration and frequency, how many ping slots are configured, the data rate for the ping slots, and how much data is transmitted down to the device. The Semtech Corporation, which supplies the LoRa radio chips, estimates (for SF9, one 30 byte downlink per hour, and 10 mA RX current) an average Class B additional average current of 23.33 µA over a beacon period, or an additional average power of 77 µW [40] (at 3.3 V).

5.2 Designing a New Intermittent Computing Platform

I did some design exploration for an intermittent computing platform that uses both high and low but constant power sources, specifically in terms of what kind of power management this needs. This power management subsystem needs to remain powered on constantly so it can react to changes in energy availability, and also react to signals from Artemia. I describe this exploration in section 5.2.1, but then I propose going beyond and using batteries as low but constant power sources in section 5.2.2. Finally, I present my new design in section 5.2.

5.2.1 Brine: An Early Design Exploration for a Modular Mainboard for Intermittent Computing

Brine is a modular mainboard for intermittent computing, to which microcontrollers can be plugged into through an adapter board with a PCI-E express connector. Brine exposes two SPI buses, two UART busses, and a PDM bus that can expose microcontroller busses. Additionally, it provides inputs for low and high power rails. Brine relies on status and control signals, described in eqs. (5.1) and (5.2), which track the voltage rail states and also internal events to determine when to turn on the microcontroller. One of the SPI peripherals is further enhanced to support Artemia control signals and notifications. Of note, Brine implements power management logic on-board, by using discrete NAND chips for combinatorial logic and flip-flops as S-R latches.

Equation (5.1) describes the primary logic brine uses to turn on the primary boost converter, which we use to deal with ESR effects from the the main super capacitor store, like

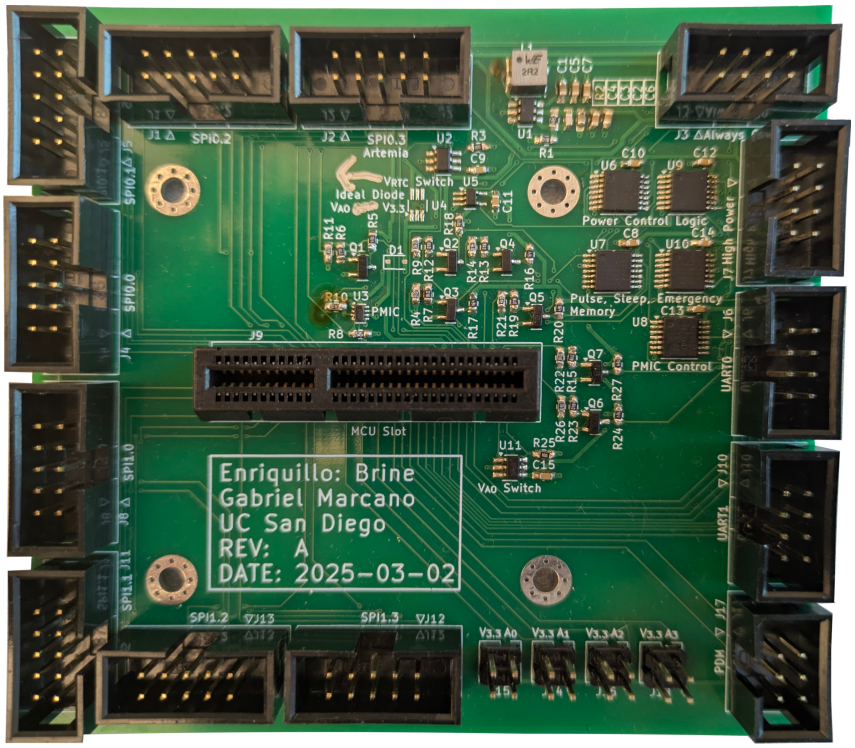


Figure 5.1. Brine mainboard, a prototype for testing modular intermittent system configurations. Notably, the mainboard conditions power through a boost converter, and also manages the sleep states of the system by implementing control logic with discrete NAND gates and some flip-flops to track state. The mainboard pins out two SPI busses and two UART channels that any microcontroller plugged into the board can use. One of the SPI ports is designed to be used with Artemia, which provides signals the mainboard control logic uses to manage sleep states.

Culpeo [138].

$$R_{off}(R_{on} + V_{ao_good}(L_M + L_{RTC})) + L_{EMERGENCY} = TURN_ON \tag{5.1}$$

Signal Name	Description
R_{off}	Request to turn off circuit (this one has priority)
R_{on}	Request to keep circuit on
V_{ao_good}	Always-on rail voltage level good signal
L_M	Latch memory output, want to use RTC latch signal (software controls)
L_{RTC}	RTC latch, pulled high once the RTC finally pulses (software must reset)
$L_{EMERGENCY}$	Emergency low voltage latch, pulled high once the voltage on the always-on rail drops to critical levels (software must reset)
$TURN_ON$	Output signal, indicating that boost converter should turn on

Brine uses flip-flops as SR latches (through exposed asynchronous pins) to hold onto state (such as L_M) or to hold information about an event that occurred ($L_{EMERGENCY}$ and L_{RTC}).

Additionally, eq. (5.2) describes the logic Brine uses to control a PMIC, which it uses to switch between the high power and low power rails, depending on which is higher. Notably, eq. (5.2) turns off the PMIC completely if the low power rail is bad, or if the high power rail is bad and the microcontroller is sleeping. The net effect is that the PMIC remains on so long as the low power rail is good and the microcontroller is either sleeping or the high power rail is good. The microcontroller has the ability to clear any latches, and it also has the ability to override the control logic and force the system to stay on or off (at least until all power rails fail).

$$V_{AO_GOOD} + (\ V_{H_GOOD})(\ Sleeping) = PMIC_OFF \tag{5.2}$$

Signal Name	Description
V_{AO_GOOD}	Status from always-on rail indicating voltage is good
V_{H_GOOD}	Status from high power rail indicating voltage is good
$Sleeping$	Status signal indicating that the microcontroller is asleep
$PMIC_OFF$	Signal to turn off PMIC (puts it in high impedance)

Lastly, I measure the quiescent current draw of the logic control components of Brine to be 2.5 µA. Turning on the boost converter, even without a load, increases the quiescent draw to 200 µA, which is not ideal. Fortunately, this large quiescent draw due to poor component selection, as there are other boost converters that provide better unloaded current draw. This increased quiescent load is only necessary when the boost converter is turned on, which it is not required if the always-on rail has power and the microcontroller is sleeping.

Brine demonstrates that it is possible to have some enhanced PMIC function while keeping quiescent current draw low. However, it is still at the mercy of both the high and low but constant energy harvesting external modules. If both of these fail, eventually Artemia and the sleeping microcontroller will fail.

5.2.2 Revisiting Batteries: The Inadvertent 20+ Year Experiment

Artemia requires, and soft intermittency prefers, a low but constant power source. Historically, the intermittent computing field prefers batteryless systems. The general rationale given is that batteries require changing, and may be toxic, expensive, etc. It turns out, however, I have

Figure 5.2. The populated PCB of a Super Nintendo Entertainment System Super Mario All-Stars game cartridge. The right top chip is the 64KB of SRAM, the bottom right chip is the game ROM, the middle chip is the MAD-1 memory mapper, and the left chip is the CIC copy protection chip (it is a curious little device, with a 4-bit SHARP microcontroller and a non-linear PC). In the top left is the CR2032 battery used to power the SRAM chip when the cartridge is disconnected from power, and this particular one is the original battery from at least 1998, when I approximately received this game as a gift.

electronics at home, using batteries, that have been maintaining their own state since the 1990s, or for *decades*. Figure 5.2 shows the PCB of a Super Nintendo Entertainment System (SNES) cartridge that has been in my collections since at least 1998. These cartridges, and those of the Nintendo Entertainment System (NES) and the Nintendo Game Boy all use CR2032 batteries to sustain save game data in an SRAM chip while they are disconnected from their console's power rails. These chips support a deep-sleep mode that, even in the 1990s, dropped current draw to around and below 6 µA [143]! The included battery then ensures these SRAM chips retain their state. Some of my games have had their battery fail, like Super Mario World from 1991, and Donkey Kong Country from 1994, but others, like Pokémon Red from 1998, *still had valid and playable data in its SRAM as of December 2024!* This means that for these few games, the CR2032 battery has maintained SRAM state for over *20 years!* Notably, the current draw of these old SRAM chips is pretty much identical to the sleep performance of microcontrollers like the Ambiq Apollo3. This implies that, so long as a CR2032 battery is only used when the Apollo3

is sleeping, it should last for over a decade of continuous use! This gives us non-volatile-like functionality while the microcontroller sleeps for the cost of the battery for at least a decade!

5.2.3 An Example Design for a Battery-backed Intermittent Computing System

BigBen [31] uses a battery to maintain an RTC, not too differently from how Artemia uses low power constant sources for power. Further, Permamote [87] actually argues for a system design that uses both a primary cell battery and a rechargeable battery as its energy stores, switching to the backup primary cell only when the rechargeable battery is unable to provide enough power. I take this a step further, in that we can use low power constant sources in addition to batteries to simplify soft intermittency management, and lengthen the lifespan of the batteries. Unlike Permamote, which runs applications from the primary cell battery when there is no other power available, I argue using the battery only as a low power constant source, in order to extend its lifespan from months or years into decades. Figure 5.3 shows the high-level architecture of such a system. Notably, it harvests energy from multiple sources, and uses a battery only to retain critical state. This system needs at least three separate power domains, specifically:

- Power Management
- Microcontroller
- Peripherals

And then 3 energy sources, in order of priority:

1. High power (e.g. solar energy harvesting)
2. Constant low power
3. Primary battery backup

The high power rail, when it is available, powers all power domains. When the high power source is insufficient, the microcontroller switches to a sleep state, and the peripheral subsystem powers down to save energy. At that point, the low power source, or the primary battery backup, provide power to the sleeping system. The control logic domain is always powered on, so it needs to be as efficient as possible.

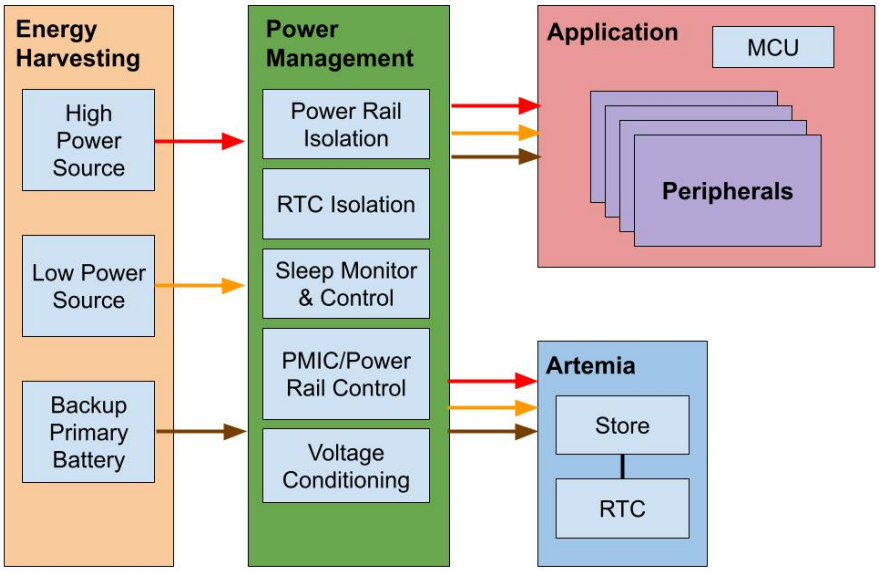


Figure 5.3. High level overview of the proposed battery-backed intermittent computing system. There are three power rails, high power for the more "traditional" energy harvesting sources, low power for trickle but constant energy harvesting sources, and then a backup primary battery. All power rails connect to the power management subsystem, which determines which rail to use to power the application domain and Artemia (optional). The power management subsystem tracks sleeps states and sends signals to the microcontroller to force it to sleep when power rails fail.

Energy Harvesting and Power Management

Each energy source should have its own energy harvesting hardware, and it must provide a signal that the power management subsystem can use to determine if it is ready to be used. The actual power rails, as well as the status signals, go to the power management subsystem, where routes the rails based on the system's current sleep state. A typical system would probably use solar energy for its high energy source, and depending on where it is, a secondary smaller amorphous solar cell for its low power source, and a primary cell battery for its backup energy

Table 5.1. Describes the sleep states of the proposed intermittent system. S0, S1, and S2 require that the high power rail is good, S3 requires that the low power rail is good, S4 requires that the battery rail is good, and OFF is the effect of no rails being good.

State	Description
S0	**Online**, microcontroller is active, can control peripherals
S1	**Sleep 1**, microcontroller is sleeping, but peripherals are active
S2	**Sleep 2**, microcontroller and peripherals are sleeping
S3	**Sleep 3**, Same as Sleep 2, but high power rail is down
S4	**Sleep 4**, Same as Sleep 3, but high and low power rails are down
OFF	**Offline**, battery failed, all rails are down, system is offline

source.

The power management subsystem manages the sleep state machine, and controls the power distribution to the power domains. We want flexibility to control how the microcontroller sleeps, and how it can wake up, depending on energy availability. We also want the microcontroller to control the peripheral power domain, including allowing peripherals to remain powered while the microcontroller sleeps, so long as there is energy available. If the high energy source fails, the microcontroller should forcibly switch to sleep (some software support is necessary here, to switch to sleep as quickly as possible on receiving this alert from hardware monitoring rails). To describe this, I borrow from ACPI sleeping states, and define sleep states for this intermittent system as described in table 5.1. I further outline the state transition table in table 5.3. In summary, this intermittent system only uses a battery as a last resort to retain critical state, otherwise it uses its low power rail to do this.

The power management subsystem ingests the "good" signals from the energy harvesting subsystems, and monitors the sleep state signal from the microcontroller. This subsystem implements the state machine managing sleep states, and routes power from the energy sources and the battery to the correct power domains. Tables 5.2 and 5.3 describe the states, what they mean for power distribution, and what happens to the output control signals. Switching between S0, S1, and S2 is handled in software by the microcontroller, and S3, S4, and OFF are handled by the control logic, with some software support to force the microcontroller to sleep.

Table 5.2. Describes the control signals and source of power for the different power domains based on the current state. In S0, the microcontroller is active, and it chooses whether to turn the peripheral domain on or off. In S1, the microcontroller sleeps, and the peripherals domain is on. In S2, S3, and S4, the microcontroller sleeps, the peripheral power domain is powered off, the and power to the microcontroller comes from the high power rail, low power rail, or the battery rail for S2, S3, and S4, respectively.

State	Sleep State	MCU Vcc	Peripheral Vcc	Sleep Now
S0	0	H	H | OFF	0
S1	1	H	H	0
S2	1	H	OFF	0
S3	1	L	OFF	1
S4	1	B	OFF	1

Table 5.3. State transition table for sleep state management. X means that that field does not matter. In general, total loss of power leads to the OFF state. Otherwise, the microcontroller can request to be put in S1 or S2 state, and the loss of the high power rail will cause it to transition to S3, and the loss of the low power rail and the high power rail will cause it to transition to S4. The microcontroller itself cannot tell the difference between S2, S3 and S4, and it requires some sort of external interrupt to move from S2, S3, or S4 to S0. Internal interrupts can wake up the microcontroller from S1.

High Power Good	Low Power Good	Battery Good	Interrupt	Request S1	Request S2	Current State	Next State	Sleep Now
0	0	0	X	X	X	X	OFF	X
0	0	1	X	X	X	!OFF	S4	1
0	1	X	X	X	X	!OFF	S3	1
1	X	X	X	X	1	S0	S2	0
1	X	X	X	1	0	S0	S1	0
1	X	X	1 (ext)	X	X	S2	S0	0
1	X	X	1 (int | ext)	X	X	S1	S0	0
1	X	X	X	X	X	OFF	S0	0

Sleep Power Draw is Similar to That of a SNES Cartridge!

I find that, as a rule of thumb, there now exist chips and discrete components that draw a quiescent current between 100 nA and 1 µA per each chip or component. Of note, this also extends to some PMICs, power switches, and even gates and flip-flops! With these efficient chips, coupled with modern microcontrollers with great sleep quiescent draw, such as the Apollo3, and using Brine as a reference point, I estimate that this theoretical intermittent computing system could draw as low as 10 µA in its sleep states. For comparison, the old SRAM chips in SNES cartridges draw around 6 µA in their sleep state. This puts this theoretical system in the same general performance domain, indicating that such a system could sustain itself with a battery at least a decade, and possibly even longer based on the availability of low power energy sources!

5.3 Future Work

5.3.1 We Need to Rally Around an Operating System

As a research community, we have spent a *lot* of time exploring software runtimes for intermittent computing systems [28, 36, 47, 117, 133, 145, 149, 165, 167, 172]. However, the community has not coalesced behind an operating system, like pre-intermittent academic embedded computing had done around TinyOS and Contiki. Too much time and energy is spent re-implementing drivers for common peripherals, clocks, etc. In industry, probably the most widely used RTOS is FreeRTOS or Zephyr, and there is some interest in Tock as an alternative embedded operating system as well. If we move to 32-bit ARM or RISCV platforms, with more resources than the MSP430FR microcontrollers, we can begin to consider these embedded operating systems for use in intermittent systems.

However, there is not much research in how to modify or work with these more typical embedded operating systems on intermittent computing systems. Especially in a multi-seated or multi-processing environment, do we need new OS-level primitives for intermittent computing? Can we adapt existing runtimes to run in the embedded operating system's userspace? Or do we

need kernel drivers and/or modifications to schedulers? Do we need new scheduling algorithms? Do we need to extend the execution model of these embedded operating systems to accommodate hardware that can sleep or turn off at any time? Or can we manage a lot of this complexity in userspace, and let the operating system deal with hardware bring-up and shutdown?

A major advantage of adopting an embedded operating systems would be plain accessibility. Intermittent computing is *hard*, and it is made harder by having to implement everything from scratch every time a new platform comes out.

5.3.2 Dealing with Clock Drift

Artemia's RTC chip is an improvement over CHRT for long timespans. But even so, like any crystal-based clock, it suffers from clock drift. In my experimentation with a version 2 of Artemia's hardware, shown in fig. 5.4, I saw up to a couple of seconds of drift over 24 hours! Specifically for Artemia, the drift is due to a mismatch in the load capacitance of the crystal, causing it to oscillate faster than expected. Due to the extremely low power requirements, we cannot afford the use of a temperature controlled crystal oscillator (TCXO) or an oven controlled crystal oscillator (OCXO), as both require currents in the milliamperes to function. Thus, we need to address clock drift somehow.

There are a few possible approaches to mitigate clock drift. One of these is to calibrate each individual device with an Artemia module at a standard temperature. The Ambiq RTCs allow for some amount of adjustment to compensate for fast or slow crystal oscillators. However, even with this calibration, changes in humidity and temperature will cause the crystal to change its frequency, and possibly even the aging of the load capacitors! Another option is to use some sort of time network synchronization protocol every so often. With Artemia, we can schedule when the system wakes up, and so long as it is "close enough", we can use that to synchronize and perform a NTP or a variant thereof to synchronize clocks against a more trustworthy source.

A third option, depending on energy availability, geographic deployment, and sensitivity to component count and size, would be to leverage existing time synchronization signals, such

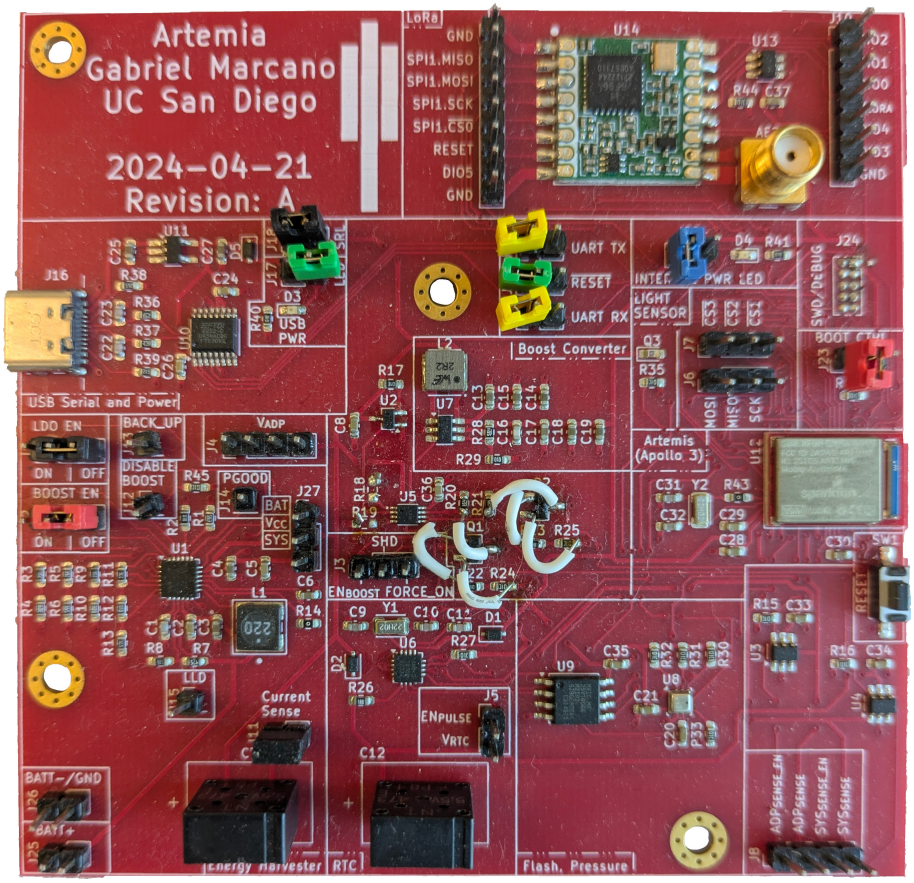


Figure 5.4. The second version of Artemia, on its own custom PCB. This version of Artemia integrates all of the peripherals the original work tests, and adds a LoRa radio on a second SPI bus. The ADP5091 energy harvester is on the lower left, as well as two large supercapacitors, one for the primary energy store, and another for the energy store of the RTC itself. This uses a boost converter to condition the voltage of the main supercapacitor for the rest of the system. There is some isolation circuitry in the center of the PCB, and a USB UART converter near the USB port on the top left (this can be isolated from the system with jumpers). On the microcontroller side, there is a flash chip and an atmospheric sensor, there is a light sensor in the middle, and a LoRa radio at the top. The LoRa radio can be cut off power independently of the rest of the system, as it is an extremely power hungry component, relative to everything else.

as the WWVB radio station time broadcast for the United States. WWVB broadcasts time information at 60 kHz, but it does so very slowly, at one bit per *second*. It takes approximately one minute to decode the entire time and date signal. Available receiver ICs, such as the MAS6180C, uses less than 100 µA while active, and 100 nA when in standby. However, these chips need to remain active to decode bits. A more interesting approach for intermittent computing is to implement a receiver from discrete components, leveraging known signal information, and only enable the receiver intermittently as well. An intermittent computing system can sample the AM signal at specific times in the one second window to detect the symbol, and thus decode timing information more efficiently.

5.3.3 We Need Better Energy Harvesters and a Better Way to Prototype Control Logic

A lot of research and commercial products exist to optimize energy harvesting from solar energy, as it is a relatively abundant energy source. However, as chapter 2 points out, research on optimizing harvesting from other sources lags behind that of solar. That we can already use alternative energy sources, with imperfect energy harvesters, bodes well for the future, if custom or specialized chips and algorithms are developed for these. For instance, sMFCs may benefit not MPPT, but from sustainable maximum power point (SMPP), a point that allows extracting the highest amount of power consistently over time, as sMFCs are very sensitive to an overshoot phenomenon that drastically reduce their power output [5].

It would be really nice to have some kind of PLC (programmable control logic) with about 100 or so LUTs that used extremely low power, for prototyping simple PMIC control. For Brine, I prototyped control logic by combining NAND ICs together. This is frustrating to route on a PCB, but at least electrically feasible due to their extremely low quiescent current. However, I would have to make a new PCB design for any modifications to the logic. One alternative would be to have some sort of breadboard-like insert where the connections can be routed with wires, but this can quickly devolve into a mess if there are a lot of connections that need to be

made. On the commercial front, there are ForgeFPGA and GreenPak, both from Renesas, as the only promising currently-available options that get close to the level of quiescent current that we need for intermittent systems. ForgeFPGA has about 1000 LUTs, at the expense of a static current of approximately 40 µW, while GreenPak has up to around 100 LUTs with a static current of approximately 10 µW. Thus, it would be interesting to see intermittent computing systems with higher power budgets try these to implement more complex power management, possibly as stepping stones to developing dedicated ASICs.

5.4 Conclusion

The intermittent computing field moved away from soft intermittency and batteries, and for good reason at the time. Microcontrollers and off-the-shelf components have now advanced to a point that the cost of hard intermittency and avoiding batteries are no longer insignificant. Modern real-time-clocks and microcontrollers sleep very efficiently, and we provide empirical evidence that at these levels, we can support decade-long intermittent computing with primary cell batteries for state retention. Low power sources, even with inefficient energy harvesting, further enhance intermittent systems, stretching how much longer we can function with or even without a battery. Future improvements on energy harvesting hardware for low-power sources will extend these enhancements further. In sum, my work simplifies state retention for intermittent computing systems, even those without available non-volatile memory.

Bibliography

[1] M. A. Abdullah, A. H. M. Yatim, C. W. Tan, and R. Saidur. 2012. A review of maximum power point tracking algorithms for wind energy systems. *Renewable and Sustainable Energy Reviews* 16, 5 (2012), 3220–3227. https://doi.org/10.1016/j.rser.2012.02.016

[2] Joshua Adkins, Bradford Campbell, Branden Ghena, Neal Jackson, Pat Pannuto, Samuel Rohrer, and Prabal Dutta. 2017. The Signpost Platform for City-Scale Sensing. , Article 47 (2017), 1 pages. https://doi.org/10.1145/3131672.3136990

[3] Saad Ahmed, Bashima Islam, Kasim Sinan Yildirim, Marco Zimmerling, Przemysław Pawełczak, Muhammad Hamad Alizai, Brandon Lucia, Luca Mottola, Jacob Sorber, and Josiah Hester. 2024. The Internet of Batteryless Things. *Commun. ACM* 67, 3 (Feb. 2024), 64–73. https://doi.org/10.1145/3624718

[4] Firoz Alam and Yingai Jin. 2023. The Utilisation of Small Wind Turbines in Built-Up Areas: Prospects and Challenges. *Wind* 3, 4 (2023), 418–438. https://doi.org/10.3390/wind3040024

[5] Muhannad Alaraj, Miloje Radenkovic, and Jae-Do Park. 2017. Intelligent energy harvesting scheme for microbial fuel cells: Maximum power point tracking and voltage overshoot avoidance. *Journal of Power Sources* 342 (2017), 726–732. https://doi.org/10.1016/j.jpowsour.2016.12.104

[6] Saad M. Alghuwainem. 1994. Matching of a DC motor to a photovoltaic generator using a step-up converter with a current-locked loop. *IEEE Transactions on Energy Conversion* 9, 1 (1994), 192–198. https://doi.org/10.1109/60.282492

[7] Mokhtar Aly and Hegazy Rezk. 2020. A Differential Evolution-Based Optimized Fuzzy Logic MPPT Method for Enhancing the Maximum Power Extraction of Proton Exchange Membrane Fuel Cells. *IEEE Access* 8 (2020), 172219–172232. https://doi.org/10.1109/ACCESS.2020.3025222

[8] Ambiq 2024. *Apollo3 and Apollo3 Blue SoC*. Ambiq. https://contentportal.ambiq.com/documents/20123/388385/Apollo3-Blue-SoC-Datasheet.pdf Revision 1.3.0, Acessed September 2025.

[9] V. Arcidiacono, S. Corsi, and L. Lambri. 1982. Maximum power point tracker for photovoltaic power plants. In *16th Photovoltaic Specialists Conference*. 507–512.

[10] Nivedita Arora, Ali Mirzazadeh, Injoo Moon, Charles Ramey, Yuhui Zhao, Daniela C. Rodriguez, Gregory D. Abowd, and Thad Starner. 2021. MARS: Nano-Power Battery-free Wireless Interfaces for Touch, Swipe and Speech Input. In *The 34th Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '21)*. Association for Computing Machinery, New York, NY, USA, 1305–1325. https://doi.org/10.1145/3472749.3474823

[11] Sid Assawaworrarit, Zunaid Omair, and Shanhui Fan. 2022. Nighttime electric power generation at a density of 50 mW/m² via radiative cooling of a photovoltaic cell. *Applied Physics Letters* 120, 14 (April 2022), 143901. https://doi.org/10.1063/5.0085205

[12] Radio Corporation of America Astro-Electronics Division. 1967. *Advanced Voltage Regulator Techniques as Applied to Maximum Power Point Tracking Systems for the Nimbus Meteorological Satellite*. Technical Report. Prepared for National Aeronautics and Space Administration.

[13] Khalid Tourkey Atta, Andreas Johansson, Michel J. Cervantes, and Thomas Gustafsson. 2015. Maximum power point tracking for micro hydro power plants using extremum seeking control. In *2015 IEEE Conference on Control Applications (CCA)*. 1874–1879. https://doi.org/10.1109/CCA.2015.7320883

[14] Kazuya Azegami, Masashi Takiguchi, Junya Yano, Hirohiko Tsutsumi, and Toshitake Masuko. 2018. Maximum Power Point Tracking Control for Small Hydroelectric Generation. In *2018 International Power Electronics Conference (IPEC-Niigata 2018 -ECCE Asia)*. 3723–3728. https://doi.org/10.23919/IPEC.2018.8507403

[15] Ali Omar Baba, Guangyu Liu, and Xiaohui Chen. 2020. Classification and Evaluation Review of Maximum Power Point Tracking Methods. *Sustainable Futures* 2 (2020), 100020. https://doi.org/10.1016/j.sftr.2020.100020

[16] Simeon Babatunde, Arwa Alsubhi, Josiah Hester, and Jacob Sorber. 2024. Greentooth: Robust and Energy Efficient Wireless Networking for Batteryless Devices. *ACM Trans. Sen. Netw.* 20, 3, Article 66 (April 2024), 31 pages. https://doi.org/10.1145/3649221

[17] Abu Bakar, Rishabh Goel, Jasper de Winkel, Jason Huang, Saad Ahmed, Bashima Islam, Przemysław Pawełczak, Kasım Sinan Yıldırım, and Josiah Hester. 2022. Protean: An Energy-Efficient and Heterogeneous Platform for Adaptive and Hardware-Accelerated Battery-Free Computing. In *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems* (Boston, Massachusetts) *(SenSys '22)*. Association for Computing Machinery, New York, NY, USA, 207–221. https://doi.org/10.1145/3560905.3568561

[18] Anderson José Balbino, Bruno da S. Nora, and Telles B. Lazzarin. 2022. An Improved Mechanical Sensorless Maximum Power Point Tracking Method for Permanent-Magnet Synchronous Generator-Based Small Wind Turbines Systems. *IEEE Transactions on Industrial Electronics* 69, 5 (2022), 4765–4775. https://doi.org/10.1109/TIE.2021.3084176

[19] Fulvio Bambusi, Francesco Cerizzi, Yamin Lee, and Luca Mottola. 2022. The Case for Approximate Intermittent Computing. In *2022 21st ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 463–476. https://doi.org/10.1109/IPSN54338.2022.00044

[20] Saurav Bandyopadhyay and Anantha P. Chandrakasan. 2011. Platform architecture for solar, thermal and vibration energy combining with MPPT and single inductor. In *2011 Symposium on VLSI Circuits - Digest of Technical Papers*. 238–239.

[21] Majd Ghazi Batarseh and Muhy Eddin Za'ter. 2018. Hybrid maximum power point tracking techniques: A comparative survey, suggested classification and uninvestigated combinations. *Solar Energy* 169 (2018), 535–555. https://doi.org/10.1016/j.solener.2018.04.045

[22] Juan P. Bello, Claudio Silva, Oded Nov, R. Luke Dubois, Anish Arora, Justin Salamon, Charles Mydlarz, and Harish Doraiswamy. 2019. SONYC: A System for Monitoring, Analyzing, and Mitigating Urban Noise Pollution. *Commun. ACM* 62, 2 (Jan. 2019), 68–77. https://doi.org/10.1145/3224204

[23] Pallavee Bhatnagar and R. K. Nema. 2013. Maximum power point tracking control techniques: State-of-the-art in photovoltaic applications. *Renewable and Sustainable Energy Reviews* 23 (2013), 224–241. https://doi.org/10.1016/j.rser.2013.02.011

[24] Mehmet Bodur and Muammer Ermis. 1994. Maximum power point tracking for low power photovoltaic solar panels. In *Proceedings of MELECON '94. Mediterranean Electrotechnical Conference*. 758–761 vol.2. https://doi.org/10.1109/MELCON.1994.380992

[25] Charles N. Bolton and Paul S. Nekrasov. 1967. *Analysis of the advanced Nimbus power systems*. Technical Report.

[26] Angelo Brambilla, M. Gambarara, A. Garutti, and F. Ronchi. 1999. New approach to photovoltaic arrays maximum power point tracking. In *30th Annual IEEE Power Electronics Specialists Conference. Record. (Cat. No.99CH36321)*, Vol. 2. 632–637 vol.2. https://doi.org/10.1109/PESC.1999.785575

[27] L. L. Bucciarelli, B. L. Grossman, E. F. Lyon, and N. E. Rasmussen. 1980. Energy balance associated with the use of a maximum power tracer in a 100-kW-peak power system. (Jan. 1980). https://www.osti.gov/biblio/5754878

[28] Michael Buettner, Benjamin Greenstein, and David Wetherall. 2011. Dewdrop: An Energy-Aware Runtime for Computational RFID. In *8th USENIX Symposium on Networked Systems Design and Implementation (NSDI 11)*. USENIX Association, Boston, MA. https://www.usenix.org/conference/nsdi11/dewdrop-energy-aware-runtime-computational-rfid

[29] Michael Buettner, Richa Prasad, Alanson Sample, Daniel Yeager, Ben Greenstein, Joshua R. Smith, and David Wetherall. 2008. RFID Sensor Networks with the Intel WISP. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems* (Raleigh, NC, USA) *(SenSys '08)*. Association for Computing Machinery, New York, NY, USA, 393–394. https://doi.org/10.1145/1460412.1460468

[30] Gabriela E. Bunea, Karen E. Wilson, Yevgeny Meydbray, Matthew P. Campbell, and Denis M. De Ceuster. 2006. Low Light Performance of Mono-Crystalline Silicon Solar Cells. In *2006 IEEE 4th World Conference on Photovoltaic Energy Conference*, Vol. 2. 1312–1314. https://doi.org/10.1109/WCPEC.2006.279655

[31] Bradford Campbell and Prabal Dutta. 2014. An energy-harvesting sensor architecture and toolkit for building monitoring and event detection. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings* (Memphis, Tennessee) *(BuildSys '14)*. Association for Computing Machinery, New York, NY, USA, 100–109. https://doi.org/10.1145/2674061.2674083

[32] Mario Cardullo. 2003. Genesis of the Versatile RFID Tag. https://www.rfidjournal.com/editors-views/genesis-of-the-versatile-rfid-tag/77385/. *RFID Journal* (April 2003).

[33] Salvador Carreon-Bautista, Celal Erbay, Arum Han, and Edgar Sánchez-Sinencio. 2015. Power Management System With Integrated Maximum Power Extraction Algorithm for Microbial Fuel Cells. *IEEE Transactions on Energy Conversion* 30, 1 (2015), 262–272. https://doi.org/10.1109/TEC.2014.2352654

[34] Zhiyuan Chen, Man-Kay Law, Pui-In Mak, Xiaoyang Zeng, and Rui P. Martins. 2020. Piezoelectric Energy-Harvesting Interface Using Split-Phase Flipping-Capacitor Rectifier With Capacitor Reuse for Input Power Adaptation. *IEEE Journal of Solid-State Circuits* 55, 8 (2020), 2106–2117. https://doi.org/10.1109/JSSC.2020.2989873

[35] Youngjin Choi, MyongHwan Kim, JiYoung Park, and Youngmo Goo. 2025. Proton Exchange Membrane Fuel Cell Stack Durability Prediction Using Arrhenius-Based Accelerated Degradation Model. *Applied Sciences* 15, 3 (2025). https://doi.org/10.3390/app15031300

[36] 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* (Amsterdam, Netherlands) *(OOPSLA 2016)*. Association for Computing Machinery, New York, NY, USA, 514–530.

https://doi.org/10.1145/2983990.2983995

[37] Alexei Colin and Brandon Lucia. 2016. Chain: tasks and channels for reliable intermittent programs. *SIGPLAN Not.* 51, 10 (Oct. 2016), 514–530. https://doi.org/10.1145/3022671.2983995

[38] 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* (Williamsburg, VA, USA) *(ASPLOS '18)*. Association for Computing Machinery, New York, NY, USA, 767–781. https://doi.org/10.1145/3173162.3173210

[39] E.T.H. Zurich Computer Engineering Group. 2021. measurement setup. https://github.com/ETHZ-TEC/RocketLogger/wiki/measurement-setup Accessed September 2025.

[40] Semtech Corporation. 2024. LoRaWAN®Device Classes. https://www.semtech.com/uploads/technology/LoRa/lorawan-device-classes.pdf Accessed February 2026.

[41] David Cortes-Vega, Fernando Ornelas-Tellez, and Juan Anzurez-Marin. 2021. Nonlinear Optimal Control for PMSG-Based Wind Energy Conversion Systems. *IEEE Latin America Transactions* 19, 7 (2021), 1191–1198. https://doi.org/10.1109/TLA.2021.9461848

[42] E. Costogue and S. Lindena. 1976. Comparison of candidate solar array maximum power utilization approaches. (Feb. 1976).

[43] Leonardo Lucio Custode, Pietro Farina, Eren Yıldız, Renan Beran Kılıç, Kasım Sinan Yıldırım, and Giovanni Iacca. 2024. Fast-Inf: Ultra-Fast Embedded Intelligence on the Batteryless Edge. In *Proceedings of the 22nd ACM Conference on Embedded Networked Sensor Systems*. Association for Computing Machinery. https://doi.org/10.1145/3666025.3699335

[44] Enrico Dallago, Alessandro Liberale, Davide Miotti, and Giuseppe Venchi. 2015. Direct MPPT Algorithm for PV Sources With Only Voltage Measurements. *IEEE Transactions on Power Electronics* 30, 12 (2015), 6742–6750. https://doi.org/10.1109/TPEL.2015.2389194

[45] Tim Daulby, Anand Savanth, Alex S. Weddell, and Geoff V. Merrett. 2020. Comparing NVM Technologies through the Lens of Intermittent Computation. In *Proceedings of the 8th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems* (Virtual Event, Japan) *(ENSsys '20)*. Association for Computing Machinery, New York, NY, USA, 77–78. https://doi.org/10.1145/3417308.3430268

[46] Jasper de Winkel, Carlo Delle Donne, Kasim Sinan Yildirim, Przemysław Pawełczak, and Josiah Hester. 2020. Reliable Timekeeping for Intermittent Computing. In *Proceedings*

of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems (Lausanne, Switzerland) *(ASPLOS '20)*. Association for Computing Machinery, New York, NY, USA, 53–67. https://doi.org/10.1145/3373376.3378464

[47] Jasper de Winkel, Vito Kortbeek, Josiah Hester, and Przemysław Pawełczak. 2020. Battery-Free Game Boy. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 3, Article 111 (Sept. 2020), 34 pages. https://doi.org/10.1145/3411839

[48] Jasper de Winkel, Haozhe Tang, and Przemysław Pawełczak. 2022. Intermittently-Powered Bluetooth That Works. In *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services* (Portland, Oregon) *(MobiSys '22)*. Association for Computing Machinery, New York, NY, USA, 287–301. https://doi.org/10.1145/3498361.3538934

[49] Samuel DeBruin, Bradford Campbell, and Prabal Dutta. 2013. Monjolo: An Energy-Harvesting Energy Meter Architecture. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems* (Roma, Italy) *(SenSys '13)*. Association for Computing Machinery, New York, NY, USA, Article 18, 14 pages. https://doi.org/10.1145/2517351.2517363

[50] Nicolas Decroix, Pierre Gasnier, and Adrien Badel. 2021. An Efficient Maximum Power Point Tracking Architecture for Weakly Coupled Piezoelectric Harvesters based on the source I–V curve. In *2021 IEEE 20th International Conference on Micro and Nanotechnology for Power Generation and Energy Conversion Applications (PowerMEMS)*. 136–139. https://doi.org/10.1109/PowerMEMS54003.2021.9658370

[51] Vishal Deep, Mathew L. Wymore, Daji Qiao, and Henry Duwe. 2022. Toward a Shared Sense of Time for a Network of Batteryless, Intermittently-powered Nodes. In *2022 IEEE International Performance, Computing, and Communications Conference (IPCCC)*. 138–146. https://doi.org/10.1109/IPCCC55026.2022.9894317

[52] Harsh Desai, Matteo Nardello, Davide Brunelli, and Brandon Lucia. 2022. Camaroptera: A Long-range Image Sensor with Local Inference for Remote Sensing Applications. *ACM Trans. Embed. Comput. Syst.* 21, 3, Article 32 (May 2022), 25 pages. https://doi.org/10.1145/3510850

[53] Harald Dillersberger, Bernd Deutschmann, and Douglas Tham. 2020. A Bipolar ±13 mV Self-Starting and 85% Peak Efficiency DC/DC Converter for Thermoelectric Energy Harvesting. *Energies* 13, 20 (Oct. 2020), 5501. https://doi.org/10.3390/en13205501

[54] Maria Doglioni, Eren Yildiz, Matteo Nardello, Khakim Akhunov, Kasim Sinan Yildirim, and Davide Brunelli. 2025. CapDYN: Adaptive Self-Scaling Energy Storage for Powering Batteryless IoT. *ACM Trans. Embed. Comput. Syst.* 24, 5, Article 77 (Sept. 2025), 32 pages.

https://doi.org/10.1145/3737288

[55] Arup Dutta, Caraline Ann Jacob, Priyanki Das, Eduardo Corton, Devard Stom, Lepakshi Barbora, and Pranab Goswami. 2022. A review on power management systems: An electronic tool to enable microbial fuel cells for powering range of electronic appliances. *Journal of Power Sources* 517 (2022), 230688. https://doi.org/10.1016/j.jpowsour.2021.230688

[56] Jakub Dziegielowski, Benjamin Metcalfe, and Mirella Di Lorenzo. 2021. Towards effective energy harvesting from stacks of soil microbial fuel cells. *Journal of Power Sources* 515 (2021), 230591. https://doi.org/10.1016/j.jpowsour.2021.230591

[57] EEMBC. 2010. EEMBC Benchmark Score Viewer: MSP430F5438. https://www.eembc.org/viewer/?benchmark_seq=1141 Accessed September 2025.

[58] S. E. Ben Elghali, M. E. H. Benbouzid, Jean-Frederic Charpentier, Tarek Ahmed-Ali, J. M. Gahery, and A. Denis. 2008. Modeling and MPPT sensorless control of a DFIG-Based marine current turbine. In *2008 18th International Conference on Electrical Machines*. 1–6. https://doi.org/10.1109/ICELMACH.2008.4800165

[59] Jye Yun Fam, Shen Yuong Wong, Hazrul Bin Mohamed Basri, Mohammad Omar Abdullah, Kasumawati Binti Lias, and Saad Mekhilef. 2021. Predictive Maximum Power Point Tracking for Proton Exchange Membrane Fuel Cell System. *IEEE Access* 9 (2021), 157384–157397. https://doi.org/10.1109/ACCESS.2021.3129849

[60] Ahmed Fathy, Hegazy Rezk, and Turki M. Alanazi. 2021. Recent Approach of Forensic-Based Investigation Algorithm for Optimizing Fractional Order PID-Based MPPT With Proton Exchange Membrane Fuel Cell. *IEEE Access* 9 (2021), 18974–18992. https://doi.org/10.1109/ACCESS.2021.3054552

[61] Nicola Femia, Giovanni Petrone, Giovanni Spagnuolo, and Massimo Vitelli. 2005. Optimization of perturb and observe maximum power point tracking method. *IEEE Transactions on Power Electronics* 20, 4 (2005), 963–973. https://doi.org/10.1109/TPEL.2005.850975

[62] D. A. Fox, K. C. Shuey, and D. L. Stechschulte. 1979. Peak power tracking technique for photovoltaic arrays. In *1979 IEEE Power Electronics Specialists Conference*. 219–227. https://doi.org/10.1109/PESC.1979.7081028

[63] Shay Gal-On and Markus Levy. 2012. Exploring coremark a benchmark maximizing simplicity and efficacy. *The Embedded Microprocessor Benchmark Consortium* 6, 23 (2012), 87.

[64] GCC. 2025. LRA is the default. https://gcc.gnu.org/wiki/LRAIsDefault Accessed

November 2025.

[65] GCC. 2025. reload. https://gcc.gnu.org/wiki/reload Accessed November 2025.

[66] Kai Geissdoerfer and Marco Zimmerling. 2021. Bootstrapping Battery-free Wireless Networks: Efficient Neighbor Discovery and Synchronization in the Face of Intermittency. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. USENIX Association, 439–455. https://www.usenix.org/conference/nsdi21/presentation/geissdoerfer

[67] Kai Geissdoerfer and Marco Zimmerling. 2022. Learning to Communicate Effectively Between Battery-free Devices. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. USENIX Association, Renton, WA, 419–435. https://www.usenix.org/conference/nsdi22/presentation/geissdoerfer

[68] Alessandro Giustiniani, Giovanni Petrone, Giovanni Spagnuolo, and Massimo Vitelli. 2010. Low-Frequency Current Oscillations and Maximum Power Point Tracking in Grid-Connected Fuel-Cell-Based Systems. *IEEE Transactions on Industrial Electronics* 57, 6 (2010), 2042–2053. https://doi.org/10.1109/TIE.2009.2034175

[69] Lawrence H. Goldstein and Glenn R. Case. 1978. PVSS—A photovoltaic system simulation program. *Solar Energy* 21, 1 (1978), 37–43. https://doi.org/10.1016/0038-092X(78)90114-7

[70] Shay "Blargg" Green. 2016. snes_spc. https://github.com/elizagamedev/snes_spc Accessed September 2025, mirror of original content.

[71] Hajera Gul, Waseem Raza, Jechan Lee, Mudassar Azam, Mujtaba Ashraf, and Ki-Hyun Kim. 2021. Progress in microbial fuel cell technology for wastewater treatment and energy harvesting. *Chemosphere* 281 (2021), 130828. https://doi.org/10.1016/j.chemosphere.2021.130828

[72] Baoling Guo, Amgad Mohamed, Seddik Bacha, and Mäzen Alamir. 2018. Variable speed micro-hydro power plant: Modelling, losses analysis, and experiment validation. In *2018 IEEE International Conference on Industrial Technology (ICIT)*. 1079–1084. https://doi.org/10.1109/ICIT.2018.8352328

[73] Agrim Gupta, Daegue Park, Shayaun Bashar, Cedric Girerd, Nagarjun Bhat, Siddhi Mundhra, Tania K. Morimoto, and Dinesh Bharadia. 2023. ForceSticker: Wireless, Batteryless, Thin & Flexible Force Sensors. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 7, 1, Article 13 (March 2023), 32 pages. https://doi.org/10.1145/3580793

[74] Josiah 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 (Memphis, Tennessee) *(SenSys '14)*. Association for Computing Machinery, New York, NY, USA, 1–15. https://doi.org/10.1145/2668332.2668336

[75] Josiah Hester, Lanny Sitanayah, and Jacob Sorber. 2015. Tragedy of the Coulombs: Federating Energy Storage for Tiny, Intermittently-Powered Sensors. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems* (Seoul, South Korea) *(SenSys '15)*. Association for Computing Machinery, New York, NY, USA, 5–16. https://doi.org/10.1145/2809695.2809707

[76] Josiah 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* (Delft, Netherlands) *(SenSys '17)*. Association for Computing Machinery, New York, NY, USA, Article 19, 13 pages. https://doi.org/10.1145/3131672.3131674

[77] Josiah Hester and Jacob Sorber. 2017. The Future of Sensing is Batteryless, Intermittent, and Awesome. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems* (Delft, Netherlands) *(SenSys '17)*. Association for Computing Machinery, New York, NY, USA, Article 21, 6 pages. https://doi.org/10.1145/3131672.3131699

[78] Josiah Hester, Kevin Storer, and Jacob Sorber. 2017. Timely Execution on Intermittently Powered Batteryless Sensors. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems* (Delft, Netherlands) *(SenSys '17)*. Association for Computing Machinery, New York, NY, USA, Article 17, 13 pages. https://doi.org/10.1145/3131672.3131673

[79] Josiah Hester, Nicole Tobias, Amir Rahmati, Lanny Sitanayah, Daniel Holcomb, Kevin Fu, Wayne P. Burleson, and Jacob Sorber. 2016. Persistent Clocks for Batteryless Sensing Devices. *ACM Trans. Embed. Comput. Syst.* 15, 4, Article 77 (Aug. 2016), 28 pages. https://doi.org/10.1145/2903140

[80] Andrew P. Hill, Peter Prince, Jake L. Snaddon, C. Patrick Doncaster, and Alex Rogers. 2019. AudioMoth: A low-cost acoustic device for monitoring biodiversity and the environment. *HardwareX* 6 (2019), e00073. https://doi.org/10.1016/j.ohx.2019.e00073

[81] Takashi Hiyama, S. Kouzuma, and Tomofiimi Imakubo. 1995. Identification of optimal operating point of PV modules using neural network for real time maximum power tracking control. *IEEE Transactions on Energy Conversion* 10, 2 (1995), 360–367. https://doi.org/10.1109/60.391904

[82] Takashi Hiyama, S. Kouzuma, Tomofiimi Imakubo, and Thomas H. Ortmeyer. 1995. Evaluation of neural network based real time maximum power tracking controller for PV system. *IEEE Transactions on Energy Conversion* 10, 3 (1995), 543–548. https://doi.org/10.1109/60.464880

[83] D. P. Hohm and Michael E. Ropp. 2000. Comparative study of maximum power point tracking algorithms using an experimental, programmable, maximum power point tracking test bed. In *Conference Record of the Twenty-Eighth IEEE Photovoltaic Specialists Conference - 2000 (Cat. No.00CH37036)*. 1699–1702. https://doi.org/10.1109/PVSC.2000.916230

[84] D. P. Hohm and Michael E. Ropp. 2003. Comparative study of maximum power point tracking algorithms. *Progress in Photovoltaics: Research and Applications* 11, 1 (2003), 47–62. https://doi.org/10.1002/pip.459

[85] K. H. Hussein, I. Muta, T. Hoshino, and M. Osakada. 1995. Maximum photovoltaic power tracking: an algorithm for rapidly changing atmospheric conditions. *IEE Proceedings - Generation, Transmission and Distribution* 142 (Jan. 1995), 59–64(5). Issue 1. https://doi.org/10.1049/ip-gtd:19951577

[86] Neal Jackson. 2022. *A Case for Application Driven Design of Energy Harvesting Sensor Systems*. Ph. D. Dissertation. EECS Department, University of California, Berkeley. http://www2.eecs.berkeley.edu/Pubs/TechRpts/2022/EECS-2022-268.html

[87] Neal Jackson, Joshua Adkins, and Prabal Dutta. 2019. Capacity over Capacitance for Reliable Energy Harvesting Sensors. In *Proceedings of the 18th International Conference on Information Processing in Sensor Networks* (Montreal, Quebec, Canada) *(IPSN '19)*. Association for Computing Machinery, New York, NY, USA, 193–204. https://doi.org/10.1145/3302506.3310400

[88] Dhananjay Jagtap and Pat Pannuto. 2020. Reliable Energy Sources as a Foundation for Reliable Intermittent Systems. In *Proceedings of the 8th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems* (Virtual Event, Japan) *(ENSsys '20)*. Association for Computing Machinery, New York, NY, USA, 22–28. https://doi.org/10.1145/3417308.3430276

[89] Dhananjay Jagtap and Pat Pannuto. 2021. Repurposing Cathodic Protection Systems as Reliable, in-situ, Ambient Batteries for Sensor Networks. In *Proceedings of the 20th International Conference on Information Processing in Sensor Networks (Co-Located with CPS-IoT Week 2021)* (Nashville, TN, USA) *(IPSN '21)*. Association for Computing Machinery, New York, NY, USA, 357–368. https://doi.org/10.1145/3412382.3458277

[90] Yohan Jang, Chaeeun Lee, Sanghyuk Ji, and Sungwoo Bae. 2021. Hybrid Global Maximum Power Point Tracking Algorithm for a Thermoelectric Generation System. In *2021 24th International Conference on Electrical Machines and Systems (ICEMS)*. 267–271. https://doi.org/10.23919/ICEMS52562.2021.9634344

[91] Xiaofan Jiang, Joseph Polastre, and David Culler. 2005. Perpetual environmentally powered sensor networks. In *Proceedings of the 4th International Symposium on Information*

Processing in Sensor Networks (IPSN '05). IEEE Press, Los Angeles, California, 65–es. https://doi.org/10.1109/IPSN.2005.1440974

[92] K. Jyotheeswara Reddy and N. Sudhakar. 2018. High Voltage Gain Interleaved Boost Converter With Neural Network Based MPPT Controller for Fuel Cell Based Electric Vehicle Applications. *IEEE Access* 6 (2018), 3899–3908. https://doi.org/10.1109/ACCESS.2017.2785832

[93] J. M. Kahn, R. H. Katz, and K. S. J. Pister. 1999. Next century challenges: mobile networking for "Smart Dust". In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking* (Seattle, Washington, USA) *(MobiCom '99)*. Association for Computing Machinery, New York, NY, USA, 271–278. https://doi.org/10.1145/313451.313558

[94] Aman Kansal, Dunny Potter, and Mani B. Srivastava. 2004. Performance aware tasking for environmentally powered sensor networks. In *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems* (New York, NY, USA) *(SIGMETRICS '04/Performance '04)*. Association for Computing Machinery, New York, NY, USA, 223–234. https://doi.org/10.1145/1005686.1005714

[95] Syed Muhammad Raza Kazmi, Hiroki Goto, Hai-Jiao Guo, and Osamu Ichinokura. 2011. A Novel Algorithm for Fast and Efficient Speed-Sensorless Maximum Power Point Tracking in Wind Energy Conversion Systems. *IEEE Transactions on Industrial Electronics* 58, 1 (2011), 29–36. https://doi.org/10.1109/TIE.2010.2044732

[96] James Kennedy and Russell Eberhart. 1995. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, Vol. 4. 1942–1948. https://doi.org/10.1109/ICNN.1995.488968

[97] Hyung-Sin Kim, Michael P. Andersen, Kaifei Chen, Sam Kumar, William J. Zhao, Kevin Ma, and David E. Culler. 2018. System Architecture Directions for Post-SoC/32-Bit Networked Sensors. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems (SenSys '18)*. Association for Computing Machinery, Shenzhen, China, 264–277. https://doi.org/10.1145/3274783.3274839

[98] JeongGil Ko, Kevin Klues, Christian Richter, Wanja Hofer, Branislav Kusy, Michael Bruenig, Thomas Schmid, Qiang Wang, Prabal Dutta, and Andreas Terzis. 2012. Low Power or High Performance? A Tradeoff Whose Time Has Come (and Nearly Gone). In *Wireless Sensor Networks*, Gian Pietro Picco and Wendi Heinzelman (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 98–114.

[99] kokke. 2024. tiny-AES-c. https://github.com/kokke/tiny-AES-c Accessed September 2025.

[100] C. S. Kong. 1995. A general maximum power transfer theorem. *IEEE Transactions on Education* 38, 3 (1995), 296–298. https://doi.org/10.1109/13.406510

[101] Na Kong and Dong Sam Ha. 2012. Low-Power Design of a Self-powered Piezoelectric Energy Harvesting System With Maximum Power Point Tracking. *IEEE Transactions on Power Electronics* 27, 5 (2012), 2298–2308. https://doi.org/10.1109/TPEL.2011.2172960

[102] Vito Kortbeek, Kasim Sinan Yildirim, Abu Bakar, Jacob Sorber, Josiah Hester, and Przemysław Pawełczak. 2020. Time-Sensitive Intermittent Computing Meets Legacy Software. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems* (Lausanne, Switzerland) *(ASPLOS '20)*. Association for Computing Machinery, New York, NY, USA, 85–99. https://doi.org/10.1145/3373376.3378476

[103] Eftichios Koutroulis and Kostas Kalaitzakis. 2006. Design of a maximum power tracking system for wind-energy-conversion applications. *IEEE Transactions on Industrial Electronics* 53, 2 (2006), 486–494. https://doi.org/10.1109/TIE.2006.870658

[104] Eftichios Koutroulis, Kostas Kalaitzakis, and Nicholas C. Voulgaris. 2001. Development of a microcontroller-based, photovoltaic maximum power point tracking control system. *IEEE Transactions on Power Electronics* 16, 1 (2001), 46–54. https://doi.org/10.1109/63.903988

[105] C. Kraif. 1984. Maximum power point tracking of a microprocessor - controlled wind turbine.

[106] City Labs. 2023. Betavoltaic Battery Products: Tritium Batteries Designed and Manufactured by City Labs. ttps://citylabs.net/products/ Accessed August 2023.

[107] E. E. Landsman. 1978. Maximum power trackers for photovoltaic arrays. In *13th Photovoltaic Specialists Conference*. 996–1000.

[108] Shuo Li, Xinjian Liu, and Benton H. Calhoun. 2022. A 32nA Fully Autonomous Multi-Input Single-Inductor Multi-Output Energy-Harvesting and Power-Management Platform with 1.2×105 Dynamic Range, Integrated MPPT, and Multi-Modal Cold Start-Up. In *2022 IEEE International Solid- State Circuits Conference (ISSCC)*, Vol. 65. 1–3. https://doi.org/10.1109/ISSCC42614.2022.9731732

[109] Zhijian Liang and Jie Yuan. 2021. An Event-Driven Multi-Input Multi-Output Buck-Boost Converter with Adaptive MPPT for Wide Power Range RF Energy Harvesting. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*. 1–5. https://doi.org/10.1109/ISCAS51556.2021.9401256

[110] B.-R. Lin. 1993. Analysis of fuzzy control method applied to DC-DC converter control.

In *Proceedings Eighth Annual Applied Power Electronics Conference and Exposition,*. 22–28. https://doi.org/10.1109/APEC.1993.290655

[111] Xiaomeng Liu, Hongyan Gao, Lu Sun, and Jun Yao. 2023. Generic Air-Gen Effect in Nanoporous Materials for Sustainable Energy Harvesting from Air Humidity. *Advanced Materials* (2023), 2300748. https://doi.org/10.1002/adma.202300748

[112] Chao Lu, Chi-Ying Tsui, and Wing-Hung Ki. 2011. Vibration Energy Scavenging System With Maximum Power Tracking for Micropower Applications. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 19, 11 (2011), 2109–2119. https://doi.org/10.1109/TVLSI.2010.2069574

[113] Brandon Lucia, Vignesh Balaji, Alexei Colin, Kiwan Maeng, and Emily Ruppel. 2017. Intermittent Computing: Challenges and Opportunities. In *2nd Summit on Advances in Programming Languages (SNAPL 2017) (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 71)*, Benjamin S. Lerner, Rastislav Bodík, and Shriram Krishnamurthi (Eds.). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 8:1–8:14. https://doi.org/10.4230/LIPIcs.SNAPL.2017.8

[114] Zhihong Luo, Lei Zeng, Benjamin Lau, Yong Lian, and Chun-Huat Heng. 2018. A Sub-10 mV Power Converter With Fully Integrated Self-Start, MPPT, and ZCS Control for Thermoelectric Energy Harvesting. *IEEE Transactions on Circuits and Systems I: Regular Papers* 65, 5 (2018), 1744–1757. https://doi.org/10.1109/TCSI.2017.2757505

[115] Charles M. MacKenzie, Richard C. Greenblatt, and Arnold S. Cherdak. 1966. Nimbus Power Systems (1960-1969). *IEEE Trans. Aerospace Electron. Systems* AES-2, 6 (1966), 26–37. https://doi.org/10.1109/TAES.1966.4501985

[116] Georgios Mademlis, Yujing Liu, Peiyuan Chen, and Eric Singhroy. 2020. Design of Maximum Power Point Tracking for Dynamic Power Response of Tidal Undersea Kite Systems. *IEEE Transactions on Industry Applications* 56, 2 (2020), 2048–2060. https://doi.org/10.1109/TIA.2020.2966189

[117] Kiwan Maeng and Brandon Lucia. 2020. Adaptive Low-Overhead Scheduling for Periodic and Reactive Intermittent Execution. In *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation* (London, UK) *(PLDI 2020)*. Association for Computing Machinery, New York, NY, USA, 1005–1021. https://doi.org/10.1145/3385412.3385998

[118] Gabriel Marcano, Colleen Josephson, and Pat Pannuto. 2022. Early Characterization of Soil Microbial Fuel Cells. In *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*. 1362–1366. https://doi.org/10.1109/ISCAS48785.2022.9937297

[119] Gabriel Marcano and Pat Pannuto. 2021. Soil Power? Can Microbial Fuel Cells Power

Non-Trivial Sensors?. In *Proceedings of the 1st ACM Workshop on No Power and Low Power Internet-of-Things* (New Orleans, LA, USA) *(LP-IoT'21)*. Association for Computing Machinery, New York, NY, USA, 8–13. https://doi.org/10.1145/3477085.3478989

[120] Magical Microbes. 2021. MudWatt: Grow a Living Fuel Cell. https://www.magicalmicrobes.com/products/mudwatt-clean-energy-from-mud Accessed June 2021.

[121] Andrea Montecucco and Andrew R. Knox. 2015. Maximum Power Point Tracking Converter Based on the Open-Circuit Voltage Method for Thermoelectric Generators. *IEEE Transactions on Power Electronics* 30, 2 (2015), 828–839. https://doi.org/10.1109/TPEL.2014.2313294

[122] NIST. 2025. Cryptographic Algorithm Validation Program, AES Known Answer Test (KAT) Vectors. https://csrc.nist.gov/CSRC/media/Projects/Cryptographic-Algorithm-Validation-Program/documents/aes/KAT_AES.zip Accessed September 2025.

[123] Thurein Paing, Jason Shin, Regan Zane, and Zoya Popovic. 2008. Resistor Emulation Approach to Low-Power RF Energy Harvesting. *IEEE Transactions on Power Electronics* 23, 3 (2008), 1494–1501. https://doi.org/10.1109/TPEL.2008.921167

[124] John Paulkovich and G. Ernest Rodriguez. 1970. Maximum power transfer by conductance comparison. In *1970 IEEE Power Electronics Specialists Conference*. 114–127. https://doi.org/10.1109/PECS.1970.7066247

[125] Xiaotao Peng, Wei Yao, Cai Yan, Jinyu Wen, and Shijie Cheng. 2020. Two-Stage Variable Proportion Coefficient Based Frequency Support of Grid-Connected DFIG-WTs. *IEEE Transactions on Power Systems* 35, 2 (2020), 962–974. https://doi.org/10.1109/TPWRS.2019.2943520

[126] Manuel Piñuela, Paul D. Mitcheson, and Stepan Lucyszyn. 2013. Ambient RF Energy Harvesting in Urban and Semi-Urban Environments. *IEEE Transactions on Microwave Theory and Techniques* 61, 7 (2013), 2715–2726. https://doi.org/10.1109/TMTT.2013.2262687

[127] J. Polastre, R. Szewczyk, and David Culler. 2005. Telos: enabling ultra-low power wireless research. In *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.* 364–369. https://doi.org/10.1109/IPSN.2005.1440950

[128] Neeraj Priyadarshi, Sanjeevikumar Padmanaban, Jens Bo Holm-Nielsen, Frede Blaabjerg, and Mahajan Sagar Bhaskar. 2020. An Experimental Estimation of Hybrid ANFIS–PSO-Based MPPT for PV Grid Integration Under Fluctuating Sun Irradiance. *IEEE Systems Journal* 14, 1 (2020), 1218–1229. https://doi.org/10.1109/JSYST.2019.2949083

[129] Nick T. Purcell, James D. Stevens, Eric Carlson, Graham Boyer, and Orlando R. Baiocchi. 2019. Harvesting Energy from Tree Trunks. In *2019 International Energy and Sustainability Conference (IESC)*. 1–12. https://doi.org/10.1109/IESC47067.2019.8976597

[130] Amir Rahmati, Mastooreh Salajegheh, Dan Holcomb, Jacob Sorber, Wayne P. Burleson, and Kevin Fu. 2012. TARDIS: Time and Remanence Decay in SRAM to Implement Secure Protocols on Embedded Devices without Clocks. In *21st USENIX Security Symposium (USENIX Security 12)*. USENIX Association, Bellevue, WA, 221–236. https://www.usenix.org/conference/usenixsecurity12/technical-sessions/presentation/rahmati

[131] Yogesh K. Ramadass and Anantha P. Chandrakasan. 2011. A Battery-Less Thermoelectric Energy Harvesting Interface Circuit With 35 mV Startup Voltage. *IEEE Journal of Solid-State Circuits* 46, 1 (2011), 333–341. https://doi.org/10.1109/JSSC.2010.2074090

[132] Kanwar Pal Singh Rana, Vineet Kumar, Nitish Sehgal, and Sunitha George. 2019. A Novel dPdI feedback based control scheme using GWO tuned PID controller for efficient MPPT of PEM fuel cell. *ISA Transactions* 93 (2019), 312–324. https://doi.org/10.1016/j.isatra.2019.02.038

[133] Benjamin Ransford, Jacob Sorber, and Kevin Fu. 2011. Mementos: System support for long-running computation on RFID-scale devices. In *Proceedings of the sixteenth international conference on Architectural support for programming languages and operating systems*. 159–170.

[134] Ali Reza Reisi, Mohammad Hassan Moradi, and Shahriar Jamasb. 2013. Classification and comparison of maximum power point tracking techniques for photovoltaic system: A review. *Renewable and Sustainable Energy Reviews* 19 (2013), 433–443. https://doi.org/10.1016/j.rser.2012.11.052

[135] A. Roberts, B. Thomas, P. Sewell, Z. Khan, S. Balmain, and J. Gillman. 2016. Current tidal power technologies and their suitability for applications in coastal and marine areas. *Journal of Ocean Engineering and Marine Energy* 2, 2 (01 May 2016), 227–245. https://doi.org/10.1007/s40722-016-0044-8

[136] Eduardo Roman, Ricardo Alonso, Pedro Ibanez, S. Elorduizapatarietxe, and D. Goitia. 2006. Intelligent PV Module for Grid-Connected PV Systems. *IEEE Transactions on Industrial Electronics* 53, 4 (2006), 1066–1073. https://doi.org/10.1109/TIE.2006.878327

[137] Shad Roundy, Brian P. Otis, Yuen-Hui Chee, Jan M. Rabaey, and Paul Wright. 2003. A 1.9GHz RF Transmit Beacon using Environmentally Scavenged Energy. *International Symposium on Low Power Electronics and Design* (Aug 2003).

[138] Emily Ruppel, Milijana Surbatovich, Harsh Desai, Kiwan Maeng, and Brandon Lucia. 2022. An Architectural Charge Management Interface for Energy-Harvesting Systems. In

2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO). 318–335. https://doi.org/10.1109/MICRO56248.2022.00034

[139] Gaurav Saini, Laxmeesha Somappa, and Maryam Shojaei Baghini. 2021. A 500-nW-to-1-mW Input Power Inductive Boost Converter With MPPT for RF Energy Harvesting System. *IEEE Journal of Emerging and Selected Topics in Power Electronics* 9, 5 (2021), 5261–5271. https://doi.org/10.1109/JESTPE.2020.2979005

[140] Alanson P. Sample, Daniel J. Yeager, Pauline S. Powledge, Alexander V. Mamishev, and Joshua R. Smith. 2008. Design of an RFID-Based Battery-Free Programmable Sensing Platform. *IEEE Transactions on Instrumentation and Measurement* 57, 11 (2008), 2608–2615. https://doi.org/10.1109/TIM.2008.925019

[141] J. J. Schoeman and J. D. van Wyk. 1982. A simplified maximal power controller for terrestrial photovoltaic panel arrays. In *1982 IEEE Power Electronics Specialists conference*. 361–367. https://doi.org/10.1109/PESC.1982.7072429

[142] T. Senjyu and K. Uezato. 1994. Maximum power point tracker using fuzzy control for photovoltaic arrays. In *Proceedings of 1994 IEEE International Conference on Industrial Technology - ICIT '94*. 143–147. https://doi.org/10.1109/ICIT.1994.467196

[143] Sharp [n. d.]. *LH5116/H*. Sharp. https://media.digikey.com/pdf/data%20sheets/sharp%20pdfs/lh511610.pdf Acessed March 2026.

[144] Lukas Sigrist, Andres Gomez, Roman Lim, Stefan Lippuner, Matthias Leubin, and Lothar Thiele. 2017. Measurement and validation of energy harvesting IoT devices. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*. 1159–1164. https://doi.org/10.23919/DATE.2017.7927164

[145] Joshua R. Smith, Alanson P. Sample, Pauline S. Powledge, Sumit Roy, and Alexander Mamishev. 2006. A wirelessly-powered platform for sensing and computation. In *Proceedings of the 8th International Conference on Ubiquitous Computing* (Orange County, CA) *(UbiComp'06)*. Springer-Verlag, Berlin, Heidelberg, 495–506. https://doi.org/10.1007/11853565_29

[146] G. Soundarya, R. Sitharthan, C. K. Sundarabalan, C. Balasundar, D. Karthikaikannan, and Jayant Sharma. 2021. Design and Modeling of Hybrid DC/AC Microgrid With Manifold Renewable Energy Sources. *IEEE Canadian Journal of Electrical and Computer Engineering* 44, 2 (2021), 130–135. https://doi.org/10.1109/ICJECE.2020.2989222

[147] Cleonilson P. Souza, Fabrício B. S. Carvalho, Filype A. N. Silva, Hening A. Andrade, Nathália de V. Silva, Orlando Baiocchi, and Ivan Müller. 2016. On Harvesting Energy from Tree Trunks for Environmental Monitoring. *International Journal of Distributed Sensor Networks* 12, 6 (2016), 9383765. https://doi.org/10.1155/2016/9383765

[148] M. G. Spencer and T. Alam. 2019. High power direct energy conversion by nuclear batteries. *Applied Physics Reviews* 6, 3 (Sept. 2019), 031305. https://doi.org/10.1063/1.5123163

[149] Milijana Surbatovich, Limin Jia, and Brandon Lucia. 2021. Automatically Enforcing Fresh and Consistent Inputs in Intermittent Systems. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation* (Virtual, Canada) *(PLDI 2021)*. Association for Computing Machinery, New York, NY, USA, 851–866. https://doi.org/10.1145/3453483.3454081

[150] Niraja Swaminathan, N. Lakshminarasamma, and Yue Cao. 2022. A Fixed Zone Perturb and Observe MPPT Technique for a Standalone Distributed PV System. *IEEE Journal of Emerging and Selected Topics in Power Electronics* 10, 1 (2022), 361–374. https://doi.org/10.1109/JESTPE.2021.3065916

[151] J. Swenson. 1996. The specification, design and development of a kinetic energy tidal power generator.

[152] Min Keng Tan, Chun Chong Loo, Kit Guan Lim, Pungut Ibrahim, Hui Hwang Goh, and Kenneth Tze Kin Teo. 2021. Optimized Energy Extraction in Tidal Current Technology using Evolutionary Algorithm. In *2021 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*. 1–6. https://doi.org/10.1109/IICAIET51634.2021.9573950

[153] Cheng-Yu Tang, Hsueh-Ju Wu, Chun-Yen Liao, and Hsiu-Hsien Wu. 2022. An Optimal Frequency-Modulated Hybrid MPPT Algorithm for the LLC Resonant Converter in PV Power Applications. *IEEE Transactions on Power Electronics* 37, 1 (2022), 944–954. https://doi.org/10.1109/TPEL.2021.3094676

[154] Jingcheng Tao, Wei Mao, Zhihong Luo, Lei Zeng, and Chun-Huat Heng. 2022. A Fully Integrated Power Converter for Thermoelectric Energy Harvesting With 81% Peak Efficiency and 6.4-mV Minimum Input Voltage. *IEEE Transactions on Power Electronics* 37, 5 (2022), 4968–4972. https://doi.org/10.1109/TPEL.2021.3134367

[155] Ying-Khai Teh and Philip K. T. Mok. 2014. Design of Transformer-Based Boost Converter for High Internal Resistance Energy Harvesting Sources With 21 mV Self-Startup Voltage and 74% Power Efficiency. *IEEE Journal of Solid-State Circuits* 49, 11 (2014), 2694–2704. https://doi.org/10.1109/JSSC.2014.2354645

[156] W. J. A. Teulings, J. C. Marpinard, A. Capel, and D. O'Sullivan. 1993. A new maximum power point tracking system. In *Proceedings of IEEE Power Electronics Specialist Conference - PESC '93*. 833–838. https://doi.org/10.1109/PESC.1993.472018

[157] Texas Instruments 2021. *MSP430FR599x, MSP430FR596x Mixed-Signal Microcon-*

trollers. Texas Instruments. https://www.ti.com/lit/gpn/msp430fr5994 Acessed September 2025.

[158] Ridvan Umaz. 2020. A Power Management System for Microbial Fuel Cells With 53.02% Peak End-to-End Efficiency. *IEEE Transactions on Circuits and Systems II: Express Briefs* 67, 11 (2020), 2592–2596. https://doi.org/10.1109/TCSII.2019.2951810

[159] Bob Violino. 2005. The History of RFID Technology. https://www.rfidjournal.com/expert-views/the-history-of-rfid-technology/76202/. *RFID Journal* (Jan. 2005).

[160] O. Wasynezuk. 1983. Dynamic Behavior of a Class of Photovoltaic Power Systems. *IEEE Transactions on Power Apparatus and Systems* PAS-102, 9 (1983), 3031–3037. https://doi.org/10.1109/TPAS.1983.318109

[161] Mark Weiser. 1991. The Computer for the 21st Century. *Scientific American* 265, 3 (Sept. 1991), 94–104. https://doi.org/10.1038/scientificamerican0991-94

[162] Harrison Williams, Saim Ahmad, and Matthew Hicks. 2024. A Difference World: High-performance, NVM-invariant, Software-only Intermittent Computation. In *2024 USENIX Annual Technical Conference (USENIX ATC 24)*. USENIX Association, Santa Clara, CA, 1223–1238. https://www.usenix.org/conference/atc24/presentation/williams

[163] Harrison Williams, Xun Jian, and Matthew Hicks. 2020. Forget Failure: Exploiting SRAM Data Remanence for Low-overhead Intermittent Computation. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems* (Lausanne, Switzerland) *(ASPLOS '20)*. Association for Computing Machinery, New York, NY, USA, 69–84. https://doi.org/10.1145/3373376.3378478

[164] Chung-Yuen Won, Duk-Heon Kim, Sei-Chan Kim, Won-Sam Kim, and Hack-Sung Kim. 1994. A new maximum power point tracker of photovoltaic arrays using fuzzy controller. In *Proceedings of 1994 Power Electronics Specialist Conference - PESC'94*, Vol. 1. 396–403 vol.1. https://doi.org/10.1109/PESC.1994.349703

[165] Yilun Wu, Byounguk Min, Mohannad Ismail, Wenjie Xiong, Changhee Jung, and Dongyoon Lee. 2024. IntOS: Persistent Embedded Operating System and Language Support for Multi-threaded Intermittent Computing. In *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)*. USENIX Association, Santa Clara, CA, 425–443. https://www.usenix.org/conference/osdi24/presentation/wu-yilun

[166] Fan Yang, Ashok Samraj Thangarajan, Sam Michiels, Wouter Joosen, and Danny Hughes. 2021. Morphy: Software Defined Charge Storage for the IoT. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems* (Coimbra, Portugal) *(SenSys '21)*. Association for Computing Machinery, New York, NY, USA, 248–260.

https://doi.org/10.1145/3485730.3485947

[167] Fan Yang, Ashok Samraj Thangarajan, Gowri Sankar Ramachandran, Wouter Joosen, and Danny Hughes. 2021. AsTAR: Sustainable Energy Harvesting for the Internet of Things through Adaptive Task Scheduling. *ACM Trans. Sen. Netw.* 18, 1, Article 4 (Oct. 2021), 34 pages. https://doi.org/10.1145/3467894

[168] Lohit Yerva, Bradford Campbell, Apoorva Bansal, Thomas Schmid, and Prabal Dutta. 2012. Grafting Energy-harvesting Leaves Onto the Sensornet Tree. In *Proceedings of the 11th International Conference on Information Processing in Sensor Networks* (Beijing, China) *(IPSN'12)*. ACM, New York, NY, USA, 197–208. https://doi.org/10.1145/2185677.2185733

[169] Lohit Yerva, Bradford Campbell, Apoorva Bansal, Thomas Schmid, and Prabal Dutta. 2012. Grafting energy-harvesting leaves onto the sensornet tree. In *2012 ACM/IEEE 11th International Conference on Information Processing in Sensor Networks (IPSN)*. 197–207. https://doi.org/10.1109/IPSN.2012.6920957

[170] Eren Yildiz, Davide Cavedon, and Kasim Sinan Yildirim. 2024. On Tracking Time with Better Resolution and Range in Batteryless Systems. In *Proceedings of the 12th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems* (Hangzhou, China) *(ENSsys '24)*. Association for Computing Machinery, New York, NY, USA, 42–47. https://doi.org/10.1145/3698384.3699617

[171] Chuang Yu and K. T. Chau. 2009. Thermoelectric automotive waste heat energy recovery using maximum power point tracking. *Energy Conversion and Management* 50, 6 (2009), 1506–1512. https://doi.org/10.1016/j.enconman.2009.02.015

[172] Kasım Sinan Yıldırım, Amjad Yousef Majid, Dimitris Patoukas, Koen Schaper, Przemyslaw Pawelczak, and Josiah Hester. 2018. InK: Reactive Kernel for Tiny Batteryless Sensors. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems* (Shenzhen, China) *(SenSys '18)*. Association for Computing Machinery, New York, NY, USA, 41–53. https://doi.org/10.1145/3274783.3274837

[173] Zizhen Zeng, Johan J. Estrada-López, Mohamed A. Abouzied, and Edgar Sánchez-Sinencio. 2020. A Reconfigurable Rectifier With Optimal Loading Point Determination for RF Energy Harvesting From -22 dBm to -2 dBm. *IEEE Transactions on Circuits and Systems II: Express Briefs* 67, 1 (2020), 87–91. https://doi.org/10.1109/TCSII.2019.2899338

[174] Hong Zhang, Jeremy Gummeson, Benjamin Ransford, and Kevin Fu. 2011. *Moo: A batteryless computational RFID and sensing platform*. Technical Report UM-CS-2011-020.

[175] Xiaoshun Zhang, Bo Yang, Tao Yu, and Lin Jiang. 2020. Dynamic Surrogate Model Based Optimization for MPPT of Centralized Thermoelectric Generation Systems Under Heterogeneous Temperature Difference. *IEEE Transactions on Energy Conversion* 35, 2 (2020), 966–976. https://doi.org/10.1109/TEC.2020.2967511

[176] Zhibin Zhou, Franck Scuiller, Jean Frédéric Charpentier, Mohamed El Hachemi Benbouzid, and Tianhao Tang. 2013. Power Smoothing Control in a Grid-Connected Marine Current Turbine System for Compensating Swell Effect. *IEEE Transactions on Sustainable Energy* 4, 3 (2013), 816–826. https://doi.org/10.1109/TSTE.2013.2251918

[177] Zhibin Zhou, Franck Scuiller, Jean Frédéric Charpentier, Mohamed El Hachemi Benbouzid, and Tianhao Tang. 2015. Power Control of a Nonpitchable PMSG-Based Marine Current Turbine at Overrated Current Speed With Flux-Weakening Strategy. *IEEE Journal of Oceanic Engineering* 40, 3 (2015), 536–545. https://doi.org/10.1109/JOE.2014.2356936

[178] Charles Zucker. 1955. Condenser Problem. *American Journal of Physics* 23, 7 (Oct. 1955), 469–469. https://doi.org/10.1119/1.1934050