UNIVERSITY OF
Southampton

University of Southampton Research Repository

Copyright © and Moral Rights for this thesis and, where applicable, any accompanying data are retained by the author and/or other copyright owners. A copy can be downloaded for personal non-commercial research or study, without prior permission or charge. This thesis and the accompanying data cannot be reproduced or quoted extensively from without first obtaining permission in writing from the copyright holder/s. The content of the thesis and accompanying research data (where applicable) must not be changed in any way or sold commercially in any format or medium without the formal permission of the copyright holder/s.

When referring to this thesis and any accompanying data, full bibliographic details must be given, e.g.

Thesis: Jie Zhan (2022) "Energy Budgeting for Intermittently-Powered Systems", University of Southampton, School of Electronics and Computer Science, PhD Thesis, 1-102.

UNIVERSITY OF SOUTHAMPTON

Faculty of Engineering and Physical Science
School of Electronics and Computer Science

Energy Budgeting for Intermittently-Powered Systems

by

Jie Zhan

A thesis for the degree of
Doctor of Philosophy

June 2022

University of Southampton

Abstract

Faculty of Engineering and Physical Science
School of Electronics and Computer Science

Doctor of Philosophy

Energy Budgeting for Intermittently-Powered Systems

by Jie Zhan

Energy harvesting has become a promising power solution for the Internet of Things, liberating numerous wireless sensors from batteries and the power grid. Environmentally harvested power, however, is intrinsic variable and intermittent, which conflicts with conventional electronics. Conventionally, energy-harvesting devices buffer energy in large energy storage devices, such as rechargeable batteries or supercapacitors, to smooth out supply variability. Unfortunately, these increase costs and device dimensions, raise pollution concerns, and have constrained lifespans. To work with only a small energy buffering capacitor, Intermittent-Powered Systems (IPSs) have been studied for intermittent supply. In IPSs, application forward progress, i.e. execution beneficial to the active application, is maintained by saving volatile computing state into non-volatile memory before power interruptions, and restored afterwards.

While IPS research has focussed on efficiently sustaining computing state at the load side, system-wise energy budgeting in IPSs has not yet been widely studied. This thesis investigates the energy budget of IPSs in order to improve forward progress. We studies the issues on sizing energy storage and setting voltage thresholds, both of which determine an energy budget. The main contributions are: (i) exploration and analysis of the energy storage sizing effect on IPS performance, where a reactive IPS model is proposed and validated to quantify and illustrate the relationship between energy storage capacitance and forward progress, showing up to 65% forward progress improvement by sizing energy storage; (ii) an energy storage sizing approach that recommends an appropriate energy storage size after analysing real-world energy availability data and trading off multiple design factors, achieving 93% of the maximum forward progress while saving 83% capacitor volume and 91% interruption periods; (iii) a runtime energy profiling and adaptation method for efficiently and reliably performing atomic tasks in cases of runtime-variable energy consumption, with results showing it can save manual profiling effort (within 5 mV error), alleviates non-termination with even 68% capacitance reduction, and improve forward progress by up to 98% with runtime-variable workloads.

Contents

List of Figures ix

List of Tables xi

Declaration of Authorship xiii

Acknowledgements xv

Acronyms xvii

1 Introduction **1**
- 1.1 Intermittently-Powered Energy-Harvesting Systems 2
- 1.2 Applications of IPSs 4
- 1.3 Research Justification 7
- 1.4 Research Questions 8
- 1.5 Research Contributions 9
- 1.6 Publications 10
- 1.7 Thesis Structure 11

2 Energy-Harvesting Intermittent Systems **13**
- 2.1 Energy Harvesting Techniques 13
  - 2.1.1 Light Energy Harvesting 14
  - 2.1.2 Radio Frequency Energy Harvesting 17
  - 2.1.3 Flow Energy Harvesting 17
- 2.2 Energy Storage for Energy-Harvesting Systems 18
  - 2.2.1 Rechargeable Batteries 19
  - 2.2.2 Capacitors 20
  - 2.2.3 Discussion 21
- 2.3 Energy-Neutral Computing 22
- 2.4 Intermittently-Powered System 24
  - 2.4.1 Static IPS 25
    - 2.4.1.1 Static Checkpointing 25
    - 2.4.1.2 Harvest-Store-Use 26
    - 2.4.1.3 Task-based 28
  - 2.4.2 Reactive IPS 28
    - 2.4.2.1 Reactive Checkpointing 28
    - 2.4.2.2 Non-Volatile Processors 30
  - 2.4.3 Power-Neutral Computing 31

2.4.3.1 Principles of Operation 32
2.4.3.2 Recent Approaches 33
2.4.4 Energy Allocation 35
2.4.5 Tools for Simulation and Emulation 35
2.5 Summary 36

3 Effect of Energy Storage Sizing on IPS Performance 37
3.1 Reactive IPS Modelling 38
3.1.1 Operating Modes of Reactive IPS 39
3.1.2 Formulating Forward Progress 40
3.1.3 Dynamic process 42
3.2 Experimental Methodology 42
3.2.1 Model Configuration 43
3.2.1.1 Energy Storage 44
3.2.1.2 Intermittent Computing Load 44
3.3 Exploration of Energy Storage Sizing 45
3.3.1 Impact of Supply Current 45
3.3.2 Impact of Volatile State Size 47
3.4 Experimental Validation 49
3.4.1 Experiment Setup 49
3.4.2 Model Validation 49
3.4.3 Validation of Sizing Effects 51
3.5 Summary 51

4 Energy Storage Sizing Approach for Deploying IPSs 53
4.1 Energy Storage Sizing Approach 54
4.1.1 Input 54
4.1.2 System Model 55
4.1.3 Trade-off 56
4.2 Sizing under Real-World Energy Conditions 56
4.2.1 Simulation Configuration 56
4.2.2 Exploration with Real-World Energy Source Conditions 57
4.2.2.1 Sizing the Energy Harvester 58
4.2.2.2 Sizing the Energy Storage 58
4.2.2.3 Interruption Period 59
4.2.3 Trading Forward Progress, Dimensions, and Interruption Period 61
4.2.3.1 Metric of Dimensions 61
4.2.3.2 Metric of Interruption Periods 62
4.2.3.3 Cost Function 62
4.2.3.4 Results 63
4.3 Summary 63

5 Runtime Energy Profiling and Adaptation for IPSs 67
5.1 Motivation 69
5.1.1 Variability in Intermittent Systems 69
5.1.1.1 Variable Data Sizes 70
5.1.1.2 Variability in Peripheral Configurations 70

5.1.1.3 Device Variability 71
5.1.1.4 Capacitor Ageing and Tolerance 71
5.1.2 Performance Improvement with Adaptive Thresholds 72
5.1.2.1 Power Analysis 72
5.1.2.2 Runtime Control Models 73
5.1.2.3 Simulation Setup 73
5.1.2.4 Results 74
5.2 OPTIC Runtime Energy Profiling 76
5.2.1 A disconnecting-supply approach 77
5.2.2 Principles 77
5.2.3 Minimising and Compensating Theoretical Profiling Error 79
5.2.4 Effect of Volatile Supply Current 80
5.3 OPTIC Runtime Energy Adaptation 80
5.3.1 Adaptation Routine 81
5.3.2 Linear Adaptation 83
5.4 Implementation 84
5.4.1 External Voltage Monitor 84
5.4.2 Software 87
5.5 Experimental Evaluation 87
5.5.1 Experimental Setup and Benchmarks 87
5.5.2 Profiling Accuracy 88
5.5.3 Reliability with Dynamic Energy Consumption 89
5.5.3.1 Changing once 89
5.5.3.2 Changing infrequently 90
5.5.3.3 Changing frequently 92
5.6 Summary 93

6 Conclusions and Future Work 95
6.1 Answers to Research Questions 95
6.2 Future Work 97
6.3 Conclusion 99

References 101

List of Figures

1.1 A conceptual architecture of an IPS. 4
1.2 Application Suitability of Energy-Harvesting IPSs. 5

2.1 Typical I-V curves of a PV panel at a constant temperature and different irradiance. 15
2.2 Daily global horizontal solar energy available in Los Angeles 2012-2014 [65]. 16
2.3 One-day dynamics of global horizontal solar irradiance in Los Angeles 29 April 2016 [65]. 16
2.4 Dynamics of a micro wind harvester. 18
2.5 Harvest-Store-Use execution (taken from [86] © 2017 ACM). 27
2.6 Voltage trace with hibernation and restoration points in Hibernus (taken from [37] © 2015 IEEE). 29
2.7 Memory architectures of traditional processors and NVPs (taken from [89] © 2015 ACM). 31
2.8 Architecture of an example power neutral system based on TI MSP430FR platform (taken from [70] © 2016 IEEE). 33
2.9 Board power consumption of ODROID XU4 vs operating frequency and core configurations, running CPU intensive application Raytrace (taken from [97] © 2017 IEEE). 34

3.1 Operating modes of reactive IPSs, and achieved forward progress against supply current. 39
3.2 Operating cycles in the *Intermittent* mode. 41
3.3 Methodology of Model Exploration and Experimental Validation. . . . 43
3.4 Forward progress against energy storage capacitance at different levels of constant supply current. Error bars around C_{α_max} denote the impact of typical ±20% capacitance tolerance. The 30–150 µF range is omitted as forward progress in that range increases monotonically. 46
3.5 Maximum forward progress improvement by sizing energy storage given a spectrum of supply current (normalised by the minimum capacitance case), with the corresponding capacitance for maximum and sub-maximum (95% of maximum) forward progress improvement. 47
3.6 Impact of RAM usage (linear to restore/save overheads) on sizing energy storage with 0.4 mA current supply. Improvement and reduction are normalised by the minimum capacitance case. 48
3.7 Model validation with experimental and modelled forward progress. . 50
3.8 The relationship between energy storage capacitance and IPS forward progress, for various supply currents. 50

4.1 Structure of the proposed system model and sizing approach. 55
4.2 System model of a PV-based IPS. 56
4.3 Improvement of average forward progress by sizing energy storage given different PV panel areas under real-world energy source conditions. The model is able to find the PV panel area required for achieving the target mean forward progress. 59
4.4 Time percentiles of forward progress by sizing energy storage with target $\alpha_{\text{exe}} = 0.1$ and the corresponding PV panel area listed in Figure 4.3. The percentiles start from the 60th as the system is off for around 55 % of time due to insufficient energy source. 60
4.5 Distribution of interruption periods. 61
4.6 Tantalum capacitor volume against capacitance for the six series of capacitors analysed. 62
4.7 The sizing approach trades off forward progress, capacitor volume, and interruption periods. The results are plotted against a range of PV panel area, given Denver 2018 energy source dataset. 64

5.1 ΔV_{task} varying linearly with the data size in AES 128-bit encryption. . . . 70
5.2 An I-V curve of a glass-type amorphous PV panel (Sanyo AM-1417CA, 35 mm × 13.9 mm) under a white LED lighting condition. 72
5.3 Numbers of completed and failed operations of DEBS Low, Samoyed, DEBS High, and OPTIC Oracle given random data sizes and configurations and a PV supply in a 10 s simulation. 75
5.4 An instance of supply voltage traces in simulation. 75
5.5 Number of completed operations of Samoyed, DEBS High, and the OPTIC Oracle with capacitance reduction. 76
5.6 Experimental schematic. 77
5.7 An illustrative supply voltage trace for explaining OPTIC's runtime energy profiling method. 78
5.8 Flowchart of OPTIC's runtime energy adaptation routine. The blue blocks represent a configurable control logic to decide when to perform energy profiling. The dashed purple blocks are only run when the profiling is enabled, and represent OPTIC's energy profiling with the last block updating V_{th} with the new ΔV_{task}. 82
5.9 A picture of OPTIC's experimental setup based on an MSP430FR5994 development board. The MCU is intermittently powered by a digital power supply (constant low current source), and can also be powered by a PV panel. The workload in this picture transmits wireless packets to an always-on receiver. 85
5.10 OPTIC system schematic. 86
5.11 Error Distribution of OPTIC's Runtime Energy Profiling given a PV supply, compared to the "Naive" disconnecting-supply method. 88
5.12 A voltage trace of OPTIC adapting to a new operation on a new device. 90
5.13 Effect of Capacitor Degradation on OPTIC and DEBS. 91
5.14 Relative Completion Rates of Samoyed, DEBS, and OPTIC with variable data sizes and a PV supply. 92

6.1 A conceptual architecture of an EMU-centred IPS. 98

List of Tables

2.1 Classification of energy sources and energy harvesters in IoT. 14
2.2 Comparison between commercial NiMH and Li-ion rechargeable batteries. 20

3.1 Model parameters of reactive IPS. 38
3.2 Profiled MCU parameters. 45
3.3 Linear scaling range of volatile state size and restore/save time overheads. 48

4.1 PV cell properties under a 1000 W/cm^2, AM-1.5 light source. 57

5.1 ΔV_{task} Varying with Configurations in AES 4KB Encryption. 71
5.2 ΔV_{task} Varying among Devices in AES 128-bit 4KB Encryption. 71
5.3 Definitions of Mathematical Symbols. 78

Declaration of Authorship

I declare that this thesis and the work presented in it is my own and has been generated by me as the result of my own original research.

I confirm that:

1. This work was done wholly or mainly while in candidature for a research degree at this University;
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
3. Where I have consulted the published work of others, this is always clearly attributed;
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
5. I have acknowledged all main sources of help;
6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
7. Parts of this work have been published as:
   Jie Zhan, Geoff V. Merrett, and Alex S. Weddell. Exploring the effect of energy storage sizing on intermittent computing system performance. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 1–10, 2021
   Jie Zhan, Alex S. Weddell, and Geoff V. Merrett. Adaptive energy budgeting for atomic operations in intermittently-powered systems. In *Proceedings of the 8th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*, ENSsys '20, pages 82–83, New York, NY, USA, 2020. Association for Computing Machinery

Signed:... Date:..................

Acknowledgements

I must take this opportunity to thank all the support received during my PhD journey, without which this thesis could not have been possible.

First of all, I would like to express my greatest appreciation to my PhD supervisors, Professor Geoff V. Merrett and Dr. Alex S. Weddell. They are the role models of my life. Not only did they guide me how to conduct research, but more importantly, they demonstrated a way of thinking and behaving. Their supervision is the invaluable fortune in my life.

Next, I would like to thank the University of Southampton and Xiamen University for offering a PhD opportunity and waiving tuition fees. I also sincerely appreciate the China Scholarship Council for the 4-year scholarship that covered my living costs during the PhD study. I feel very fortunate that I received the opportunity and financial support, and never take it for granted.

It is also a huge pleasure to meet the people in the Cyber Physical Systems research group, where I have learnt and enjoyed very much from our conversations and technical discussions. It is impossible to name all of them, but I will make my poorest attempt to thank: Domenico, Oktay, Sivert, Theo, Alberto, Edward, Samuel Wong, Samuel Isuwa, Charan, Amin, Lei, Tasios, Tim, Sara, Jia, Chenyang, and Runshan. I also cherish those lovely friends in Southampton — we shared the beautiful moments and encouragement.

Finally, I want to express my deepest love to my parents, GUO Xuan and ZHAN Shushao, and my girlfriend, HUANG Yuen. Though we were physically apart for most of the time in the past four years, you have always been connecting with me regularly and supporting me throughout the hardest periods.

Life has to go on, and I am on the way. Best wishes to all the people and take care!

Acronyms

AC	Alternating Current
ADC	Analog-to-Digital Converter
BLE	Bluetooth Low-Energy
CPU	Central Processing Unit
DC	Direct Current
DEBS	Dynamic Energy Burst Scaling
DFS	Dynamic Frequency Scaling
DVFS	Dynamic Voltage and Frequency Scaling
EMU	Energy Management Unit
EN	Energy-Neutral
EWMA	Exponentially Weighted Moving Average
FRAM	Ferroelectric Random Access Memory
IoT	Internet of Things
IPS	Intermittently-Powered System
LDO	Low-Dropout Regulator
Li-ion	Lithium Ion
Li-Po	Lithium ion Polymer
LPM	Low-Power Mode
LQT	Linear-Quadratic Tracking
MCU	Microcontroller Unit
MP-SoC	Multi-Processor System-on-Chip
MPP	Maximum Power Point
MPPT	Maximum Power Point Tracking
NiCd	Nickel Cadmium
NiMH	Nickel Metal Hydride
NVM	Non-Volatile Memory
NVP	Non-Volatile Processor
PG	Power Gate
PID	Proportional-Integral-Derivative
PN	Power-Neutral
PV	Photovoltaic
RAM	Random Access Memory

RF	Radio Frequency
RFID	RF Identification
SLA	Sealed Lead Acid
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory
TRX	Transceiver
WCMA	Weather-Conditioned Moving Average
WISP	Wireless Identification and Sensing Platform

Chapter 1

Introduction

The promising expansion of the Internet of Things (IoT) has drawn research interests on new design paradigms for deploying tens of billions of electronic devices over a wide geographical range and probably in hard-to-reach places [3, 4]. Such a scenario generates considerations on how to enable the devices in networks to operate independently and effectively and how to construct a long-life, maintenance-free, environmentally friendly, and low-cost IoT.

One of the most significant concerns in deploying IoT devices is how to power numerous low-power devices (tens of billions expected [3, 5, 6]). Traditional wired electricity limits flexibility of deployment and involve expensive wiring costs [7]. Traditional primary batteries (i.e. non-rechargeable batteries) are not suitable for such a large number of devices. A widespread use of primary batteries can cause tedious work of battery replacement due to the limited battery lifetime, and also pose pollution concerns as these batteries are typically made of non-disposable heavy-metal materials [8]. Therefore, it is necessary to find an alternative powering solution.

A potential power alternative is energy harvesters. Energy harvesters scavenge energy from environmental sources (e.g. solar irradiation, wind flow, radio frequency (RF) signals, and kinetic energy) [9]. Devices powered by energy harvesters can avoid using power wires and surpass the lifetime limit of primary batteries, enabling a scalable IoT. However, the power generated by energy harvesters in real-world deployment is variable, uncontrollable, and in many cases insufficient for continuous workload operation [10]. Hence, directly using energy harvesters as the power supply without energy buffering may cause a device to keep booting up and shutting down, making little application progress.

Initially, large energy storage, in forms of rechargeable batteries (also known as secondary batteries) or supercapacitors, is allocated with energy harvesters to buffer the temporal variations of energy input and provide reliable power supply. Motivated by such a scenario, energy-neutral (EN) operation was proposed to balance energy input and energy consumption so as to prevent a system from power failures [11]. EN operation intends to sustain systems over a long period of time (e.g. a few days [12] or a year [13]) by adapting system runtime schedules (e.g. duty cycles [12–14] or task schedules [15, 16]) according to the available energy amount.

Rechargeable batteries and supercapacitors are two main choices of energy storage in EN operation. Rechargeable batteries are historically used as energy storage in energy harvesting embedded systems because of their high energy density [17] and stable discharging profile [11]. However, due to electrolyte deterioration, the limited charge-discharge cycles of rechargeable batteries constrain the operating lifetime, causing heavy battery replacement work as well as environmental issues as primary batteries do [18]. To alleviate the problems of rechargeable batteries, supercapacitors are then explored in research. Although the energy density of supercapacitors is several orders of magnitude lower than the energy density of batteries [19], supercapacitors outperform rechargeable batteries in terms of lifetime. Supercapacitors have an estimated operational lifetime of 10 years before its capacitance reduces to 80%, whereas rechargeable batteries usually need to be replaced within 3-5 years [20]). However, to achieve a considerable energy capacity, supercapacitors should be designed to tens of farads or one hundred farads [21, 22]. Supercapacitors in such a scale occupy large volume in contrast to small IoT devices, e.g. a $34\times16\times64$ mm^3 supercapacitor [22].

1.1 Intermittently-Powered Energy-Harvesting Systems

To circumvent the lifetime, pollution, and volume problems in rechargeable batteries and supercapacitors, a research trend in energy-harvesting sensor nodes moved towards eliminating the demand for large energy storage and adopting only a minimum one, where the energy storage is only enough for ensuring the most energy-expensive atomic operation[1], typically in the form of a µF-level capacitor. Despite the benefits over batteries and supercapacitors, small capacitors can only buffer a considerably limited amount of energy. Thus, the harvested power is almost directly given to the load and the system only works when the harvested power is available. This violates the demand for stable power supply in conventional computing systems.

Without any modifications, a conventional system can only work when input power is higher than system power consumption (which is rare for an energy-harvesting supply), and cannot boot up when input power is lower than system power consumption. However, ensuring and improving local processing ability of sensor nodes is crucial for a few reasons. First, to reduce network traffic volume and energy consumption, sensor nodes should be able to process sensing data on-site and transmit only the useful information, typically when the number of sensor nodes increases in orders of magnitude [6]. Second, advanced communications techniques, such as scheduling, routing, coding, and decoding, require local computing ability to ensure timeliness and efficiency in networking [23]. Third, IoT devices are also expected to be able to trigger actions in reaction to the physical world by either receiving commands from other nodes and servers or making a decision based on locally acquired data [24]. Hence, it becomes a major concern that how to guarantee forward execution and functionality of such systems with only minimum energy storage.

---

[1] In this context, an operation is atomic if it should be completed in one consecutive period without power interrupts; otherwise, if interrupted, it should be re-executed from the beginning. Example atomic operations in IoT devices can be peripheral operations and nonvolatile memory read/write operations.

With an energy-harvesting supply and small energy buffering capacitance, a system is powered up *intermittently* once a small amount of energy is accumulated in the capacitor. The system has to utilise these intermittent power-on cycles to make application progress. To this end, many approaches for energy-harvesting intermittently-powered systems (IPSs) have been proposed in the past few years [25–27]. The majority of these approaches have been addressing how to sustain computational progress throughout intermittent power-on cycles by correctly and efficiently saving and restoring volatile computing state. The volatile computing state includes CPU registers, static RAM (SRAM) data, and perhaps peripheral configurations and data, i.e. the volatile part that cannot sustain after a power failure. The volatile state is saved into and restored from non-volatile memory (NVM), where most published approaches use ferroelectric RAM (FRAM).

According to the style of saving and restoring state, approaches in IPSs can be categorised as *proactive* and *reactive* [27]. Proactive IPSs save and restore state at design-time or compile-time defined points by inserting *checkpoints* [28–31] or defining *tasks* [32–35] in a program. A certain amount of progress is achieved and saved into NVM once the program passes a checkpoint or a task boundary; otherwise, if interrupted by a power failure, the program rolls back to the last checkpoint or task boundary. Checkpointing and task-based approaches are mainly different in implementation and usage, where a checkpoint is typically a function call while task-based IPSs require dedicated compilers and programming models. In contrast to the proactive IPSs, reactive IPSs monitor V_{cc} at runtime and save state upon an imminent power failure when supply voltage falls below a low threshold ($V_{\mathrm{cc}}<V_{\mathrm{L}}$) [36–39]. After saving state, reactive IPSs sleep and wait for energy storage to be refilled until a high threshold is reached ($V_{\mathrm{cc}}>V_{\mathrm{H}}$), where it wakes up and restore the state. Detail of these approaches will be further illustrated in Chapter 2.

A conceptual architecture of a typical energy-harvesting IPS is shown in Figure 1.1. The power frontend is an energy harvester, which transduces an ambient energy source into electric power. Then, a power regulator converts the harvested power to a suitable voltage that charges up the energy storage. The type of the power regulator depends

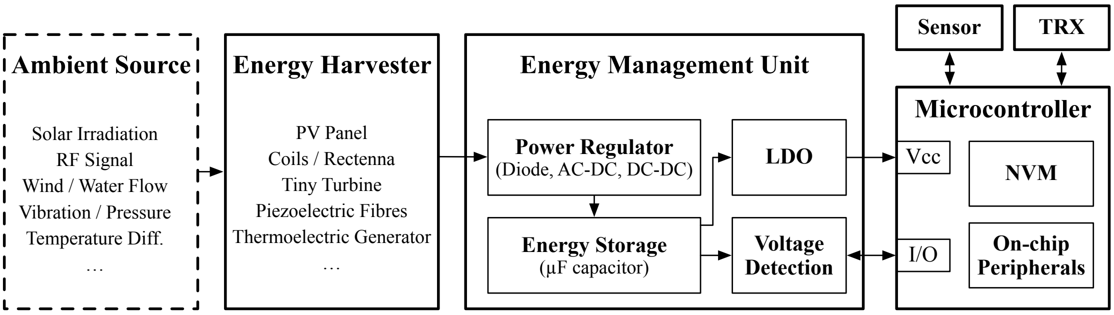

Ambient Source
Solar Irradiation
RF Signal
Wind / Water Flow
Vibration / Pressure
Temperature Diff.
…
Energy Harvester
PV Panel
Coils / Rectenna
Tiny Turbine
Piezoelectric Fibres
Thermoelectric Generator
…
Energy Management Unit
Power Regulator
(Diode, AC-DC, DC-DC)
LDO
Energy Storage
(μF capacitor)
Voltage Detection
Sensor
TRX
Microcontroller
Vcc
NVM
I/O
On-chip Peripherals

FIGURE 1.1: A conceptual architecture of an IPS.

on the pattern of the power input, which can be an AC-DC converter for AC input, e.g. a rectenna, or a DC-DC converter or a diode for DC input, e.g. a photovoltaic (PV) panel. The energy storage is in the form of a μF-level capacitor, which buffers a small amount of energy for the load to operate intermittently in short active cycles. The power given to the load is usually conditioned through a low-dropout regulator (LDO) to lower down supply voltage, and hence current consumption. IPSs are usually equipped with a voltage detection circuit so as to wake up or power up the load when the voltage of the energy storage reaches a threshold. In IPSs, the load is typically a microcontroller (MCU) with NVM to sustain computing state, and with many on-chip or external peripherals, e.g. sensors and wireless transceivers (TRX).

1.2 Applications of IPSs

An inherent limitation of IPSs is that the system can only execute when the supply power is being harvested from ambient environment, as opposed to an EN system where it can still execute with buffered energy if ambient power is not available. This limitation thereby indicates that *application operation periods and power availability should be compatible in time*. While there are various needs of operation periods for various application scenarios, the power availability is constrained and determined by the availability of the target energy source in the deployed environment. Hence, the applications of IPSs should suit, or be adapted to suit, the power availability. Under this consideration, there are two typical categories of application scenarios as seen in recent publications.

To summarise the application suitability of IPSs, a diagram is shown in Figure 1.2. An example unsuitable application can be a periodic sensing task without periodically available power or the period of the energy source does not match the sensing period (left bottom circle in Figure 1.2).

- **Category I: Applications with flexible time requirements.**

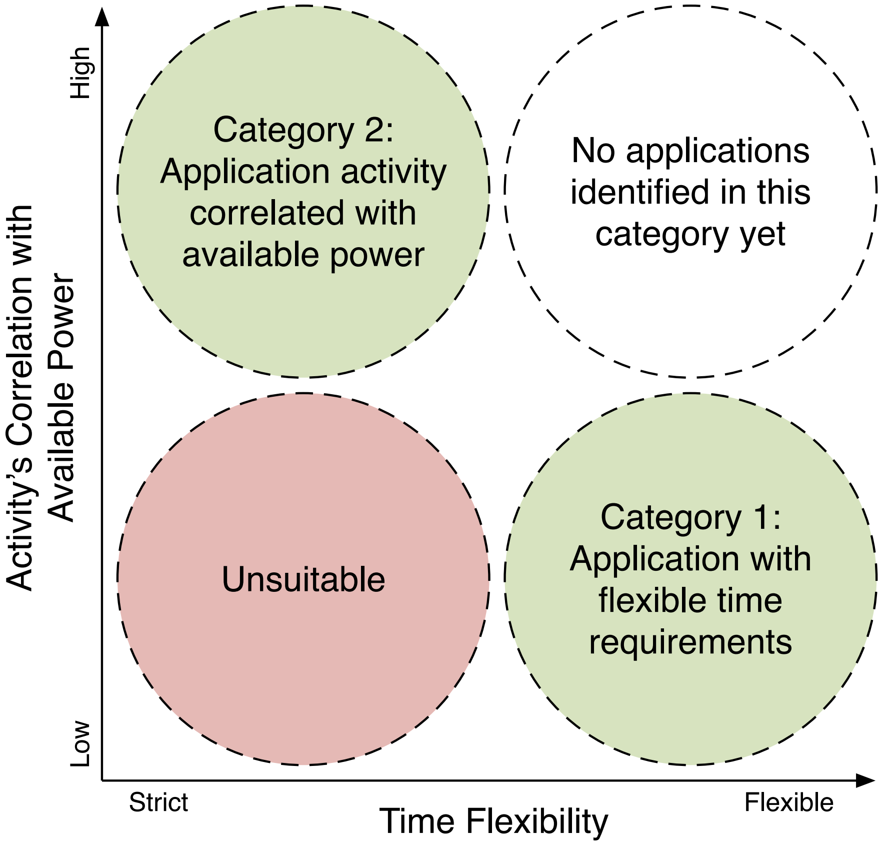

Category 2:
Application activity
correlated with
available power
No applications
identified in this
category yet
Unsuitable
Category 1:
Application with
flexible time
requirements
Activity's Correlation with
Available Power
High
Low
Strict
Time Flexibility
Flexible

FIGURE 1.2: Application Suitability of Energy-Harvesting IPSs.

Applications with flexible requirements on operating periods tolerate the intermittency of energy sources. In such applications, devices are allowed to wait for power-on periods to execute.

Application 1: Kitchen event detection [40]

This application intends to capture kitchen events, such as dishwasher working, fan on, and refrigerator cooling, to record equipment usage. As such events usually last for tens of seconds to a few hours, the device does not need to operate immediately after the event occurs or disappears. The device iterates the following tasks in turn during power-on periods: sampling acoustic information from a microphone, classifying kitchen events with a pre-trained model, and transmitting the results in Bluetooth Low-Energy (BLE) packets to an always-on server. The device harvests ambient RF energy, and the packets are transmitted every several seconds as reported. This application is to complete program iterations as frequently as possible so as to improve the accuracy of event records.

Application 2: Temperature monitor for air conditioning [41]

This application intends to monitor indoor temperature for air conditioning. As the room temperature does not usually change over a few minutes, the temperature monitor does not need to wake up frequently or periodically. During power-on periods, the device samples temperature by an external analog sensor. If the temperature is detected to be out of a pre-defined range, the device sends a BLE packet to alert the server. The device is also powered by ambient RF signals. Similar to Application 1, the device is expected to maximise sampling frequency in order to capture out-of-range temperature as soon as possible.

- **Category II: Application activity in correlation with available power.**

  In such applications, the required operation periods correlate with power-on periods. This correlation is typically linked by an event that comes with harvestable power. When the event occurs, the device is activated by harvesting the power of the event at the same time to start operating. Therefore, the application operation periods and the power availability are inherently simultaneous in such applications.

  Application 3: Bicycle trip counter [42]

  The bicycle trip counter intends to read cycling speeds and calculate total travelled distances. The wheel rotation brings energy for the device to sense the cycling speed; the device does not need to operate without cycle movement. The trip counter is contained in a single-sided $2.0 \times 1.4\,\mathrm{cm}^2$ PCB and installed on the frame of a bicycle, with a magnet on the wheel that brings electromagnetic energy to the system. Each wheel rotation activates the trip counter to calculate the current speed and log the travelled distance. After collecting enough energy over a few wheel rotations, the trip counter transmits the logged information. This application is also expected to report results as frequently as possible.

  Application 4: Monjolo Power meter [43]

  The power meter measures the power flow of a main load wire. The AC power in the wire can be harvested by a coil to activate the power meter. A design is shown in Monjolo, where the power meter transmits a plain packet to a server once it collects a preset amount of energy. The server then calculates the elapsed time between the recent two packets to estimate the main load power.

As implied by the above applications, a common application specification of IPSs is to obtain as much application progress as possible under the same energy conditions because the energy cannot be saved for later use. Depending on applications, the metric of application progress could be the program iterating rate, sensing accuracy or frequency, or the transmitting rate of results. To generally describe the application progress, in IPSs, **forward progress** denotes the effective application progress, excluding lost progress due to power failures and the progress on saving and restoring state [44]. A generic metric of forward progress can be the time spent on effective application progress. As illustrated, an IPS should maximise forward progress using the limited energy.

1.3 Research Justification

As illustrated with the previous background and application examples, a major target for many IPS approaches is to maximise forward progress given restricted energy condition. Various approaches have been proposed for the load to efficiently sustain computing state across power failures, so as to leave more energy for forward progress [34, 45–49]. However, energy efficiency can not only be explored from the load side, but can also be explored from a system perspective, where the energy budgeting in IPSs has not yet been widely studied. The energy budget of an IPS is the energy allocated for one power-on cycle. The energy budget is mainly represented in two aspects — the system energy storage size C_{stor} and the voltage threshold to wake up the load V_{th}, i.e.:

$$E_{\text{budget}} = \frac{1}{2}C_{\text{stor}}(V_{\text{th}}^2 - V_{\text{min}}^2) \tag{1.1}$$

where V_{min} represents the minimum operating voltage below which the load's hardware cannot function correctly or the IPS has to save state. In practice, as existing electronic systems typically use an LDO to keep low supply voltage for the load so as to lower the load current consumption, an IPS typically consumes relatively constant current rather than constant power when the voltage of C_{stor} changes. Hence, the energy budget in an IPS is usually expressed in charge rather than energy:

$$Q_{\text{budget}} = C_{\text{stor}}(V_{\text{th}} - V_{\text{min}}) \tag{1.2}$$

Following the two aspects of the energy budget, this thesis will focus on how to improve the energy budgeting for IPSs in order to increase forward progress, where it can be further discussed on three issues.

1. With the goal of minimising device dimensions and interruption periods, most IPS approaches adopt a minimum amount of energy storage [30, 38, 50–52]. This is typically just sufficient for the most energy-expensive atomic operation, e.g. saving and restoring a complete state [37]. However, a system with minimum energy storage may frequently go through a cycle of: wake up, restore state, execute program, save state, and halt. Provisioning more energy storage can prolong the power-on cycles, reduce the overheads, and hence increase forward progress, but can also increase system leakage and decrease forward progress. The sizing effect of energy storage on forward progress has not been studied. Therefore, a focus of

this thesis will be studying the relationship between energy storage capacitance and forward progress in IPSs.

2. Extending the above, to determine a size of energy storage of IPSs in deployment, developers may also wish to consider, along with forward progress, other design factors, e.g. devices' physical volume and interruption periods. An approach for exploring the effect of energy storage size on multiple design factors has not been proposed yet. Also, there is not a method of determining an energy storage size that balances different design factors. Hence, another focus of this thesis will be exploring an energy storage sizing approach for IPSs that balances multiple design factors in deployment.

3. Apart from the energy storage size, the voltage threshold that wakes up an IPS also determines the energy budget of one power-on cycle. Existing approaches use one or a few fixed voltage thresholds, which are calibrated at design time. Some approaches (e.g. [53]) minimise the threshold for each task, but the fixed threshold can be violated at runtime due to variability in energy consumption, leaving the system in **non-termination**, i.e. unable to finish a task due to insufficient energy and repetitive re-execution. The variable energy consumption can come from many reasons, which include, but not limited to, variability in data amounts, peripheral configurations, devices, and capacitance degradation. In contrast, some approaches (e.g. [40]) set a universal high threshold, such that the energy budget should be sufficient for all tasks. However, waiting for a high voltage threshold can be energy-inefficient because, typically, current input reduces with higher voltage and a high operating voltage also increases system quiescent current consumption. Hence, the final focus of this thesis will be the scheme of voltage threshold settings that avoids non-termination under runtime variable energy consumption while maintaining system energy efficiency.

1.4 Research Questions

Motivated by the previous discussion, the following three research questions are derived:

1. **What is the effect of sizing the energy storage capacity on IPS performance?**

   Specifically, the energy storage capacity in IPSs is presented as the capacitance between V_{cc} and ground. The forward progress rate directly determines application performance, e.g. program iteration rate or task completion time, and hence is regarded as the performance metric in this study. The goal is to explore whether sizing the energy storage capacity can change the forward progress rate in IPSs, and if so, to study and quantify the relationship between them.

2. **How may the energy storage of IPSs be sized to trade off multiple design factors, such as forward progress, device dimensions, interruption periods?**

   While the last question explores the energy storage sizing effect on computational performance, this question encompasses more design factors in IPSs that a capacitor size can affect. Increasing energy storage capacity may benefit forward progress, but may also have significant drawbacks. A larger capacitor typically has larger physical dimensions, which are a key design factor that IPSs should minimise in some application scenarios, e.g. wearable and implantable sensors. Also, a larger capacitor leads to longer charge-discharge cycles, and thus prolongs interruption periods and undermines system reactivity to external events. The goal is to study the trade-off and to propose an approach that recommends an energy storage size for practical deployment.

3. **How can an IPS run safely and efficiently when executing tasks with runtime-variable energy consumption?**

   Energy consumption of tasks can change at runtime with regards to many factors, where we include, but are not limited to, the variability in data amounts to process, peripheral configurations, devices, and capacitor degradation. A design concept is to allocate just enough energy for each task. This design concept can further break into two aspects – safety and efficiency. The safety aspect means that the IPS should intend to avoid non-termination by allocating enough energy for tasks. The efficiency aspect means that, while meeting the safety aim, the IPS should minimise the energy budget, such that the system can set the lowest possible energy threshold, maintaining energy efficiency and forward progress. The goal is to devise an approach that can enable IPSs to run with variable energy consumption of tasks, following the above design concept.

1.5 Research Contributions

The contributions that address the research questions in this thesis are:

1. Exploration and analysis of the energy storage sizing effect on IPS performance, where a reactive IPS model is proposed and validated to quantify and illustrate the relationship between energy storage capacitance and forward progress. The exploration shows adding a relatively small amount of energy storage can significantly improve forward progress by up to 65%. The proposed model demonstrates its potential for design exploration of IPSs. (Addressing Research Question 1, reported in Chapter 3)

2. An energy storage sizing approach for deploying IPSs, which accepts real-world energy availability data and trades off multiple design factors. A cost function

can be incorporated, allowing various properties of the system to be traded off. A demonstration shows it achieves 93% of the maximum forward progress while saving 83% capacitor volume and 91% interruption periods. A simulation tool is available to download[2], enabling researchers to experiment with energy storage sizes to optimise IPS designs. (Addressing Research Question 2, reported in Chapter 4)

3. A runtime energy profiling and adaptation method, named as OPTIC, for efficiently performing atomic tasks in cases of runtime-variable energy consumption. OPTIC enables runtime energy profiling of tasks, so alleviates manual profiling efforts in development. Owing to the ability of runtime energy profiling and setting a barely sufficient energy budget, OPTIC is able to: (i) adapt its threshold for a new task on a new device, (ii) adapt to a higher threshold in cases of increased energy consumption or device ageing, (iii) lower operating voltage and improve energy efficiency and forward progress. OPTIC, along with its software design tools and experimental comparisons, is open-source[3], hence facilitating future development and research. (Addressing Research Question 3, reported in Chapter 5)

1.6 Publications

The research presented in this thesis were published in the following papers:

- J. Zhan, G. V. Merrett, and A. S. Weddell. ”Exploring the Effect of Energy Storage Sizing on Intermittent Computing System Performance.” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2021. [1]
- J. Zhan, A. S. Weddell, and G. V. Merrett. ”Adaptive Energy Budgeting for Atomic Operations in Intermittently-Powered Systems.” In *Proceedings of the 8th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems (ENSsys ’20)*, pp.82-83, 2020. [2]

In additional, the following paper is currently in preparation for a journal submission:

- J. Zhan, A. S. Weddell, and G. V. Merrett. ”Runtime Energy Profiling and Adaptation for Energy-Harvesting Intermittently-Powered Systems.” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (in preparation).

---

[2] `https://git.soton.ac.uk/energy-driven/energy-storage-sizing`

[3] `https://github.com/UoS-EEC/optic`

1.7 Thesis Structure

The remainder of this thesis is organised as follows. Chapter 2 provides background on energy harvesting, energy storage and energy-neutral computing, as well as reviews recent IPS techniques following a taxonomy based on their fundamental mechanisms and focusses. Chapter 3 analyses the sizing effect of energy storage on IPS performance. Chapter 4 explores a wider energy storage sizing effect on IPSs considering multiple design factors and real-world energy conditions to determine an energy storage size when deploying IPSs. Apart from sizing energy storage, Chapter 5 focusses on runtime energy profiling and adaptation through adaptive voltage thresholding, so as to maintain system performance despite runtime variability. Chapter 6 concludes this thesis and discusses potential directions of future research.

Chapter 2

Energy-Harvesting Intermittent Systems

The development of energy harvesters and low power electronics provides the potential for wireless IoT sensors to operate with power autonomy, with their designs adapted to variable harvested power rather than reliable power supply [24]. This chapter first provides a background of various ambient energy sources and their corresponding energy harvesters in Section 2.1, as well as the types of energy storage used in energy-harvesting systems in Section 2.2. Energy-neutral operation, a traditional design of energy-harvesting systems, is introduced in Section 2.3. The recent research towards IPSs is then reviewed in Section 2.4 with a classification of proposed IPS methodologies. Finally, Section 2.5 summarises this chapter.

2.1 Energy Harvesting Techniques

Although all kinds of energy harvesters can extract energy from ambient sources, they have various output characteristics in terms of the amount and dynamics of the harvested power depending on the ambient energy conditions and energy harvesting techniques. To select an energy harvester for powering sensor nodes, one important concern is whether the supply power level matches the consumption of the load [54]. For one certain type of energy harvesters, the harvested power can be scaled by the power density of an environmental energy source and the size of an energy harvester. The power density of environmental energy sources is determined by the deploying environment, which cannot be controlled by the harvesting devices, but the size of energy harvesters can be determined at design-time with considerations on systems requirements, such as energy utilization, form factors, performance, etc.

Energy Source	**Energy Harvester**
Light (solar, artificial)	Photovoltaic cells
Radio waves	Radio frequency harvester (rectenna)
Flow (air, liquid)	Wind turbine, hydrogenerator
Mechanical (vibrations, pressure, stress-strain)	Electromagnetic, electrostatic, piezoelectric harvester
Heat	Theomal electric generator

TABLE 2.1: Classification of energy sources and energy harvesters in IoT.

In order to appropriately size and designate energy harvesters for sensor nodes, the power features of different energy harvesters are widely considered by researchers and engineers [55]. A classification of common energy harvesting sources and corresponding energy harvesters used in IoT is presented in Table 2.1. The voltage and current features of different energy harvesters largely differ from each other, due to the intrinsic differences in temporal distributions of the available amount of different energy sources and the physical principles of power conversion. The following part of this section introduces three kinds of energy sources and energy harvesting techniques listed in Table 2.1.

2.1.1 Light Energy Harvesting

Due to the abundant energy amount of light, whether from outdoor sunlight or indoor artificial light, light becomes a feasible energy source to power sensor nodes and is historically treated as a substitute for battery supplies [56, 57]. Light energy can be converted to DC power by PV cells, which consist of silicon semiconducting materials. When PV cells absorb light, electrons are excited by the photovoltaic effect, producing an electric potential by the separation of electrons and holes.

Given a fixed intensity of light, the output current from PV cells manifests an inverse relationship with the output voltage, as there is a semiconducting bypass within the PV cells. Although the power conversion efficiency may vary among different PV cell techniques (such as monocrystalline, polycrystalline, thin film), the curve shapes of current-voltage relationships are similar. Obviously, higher irradiance leads to higher current output when the output voltage is fixed, because there is more intensive light sources provided. More importantly, the output feature of PV cells can be summarised like ”an inverse semiconductor” —— when the terminal voltage is low, the output

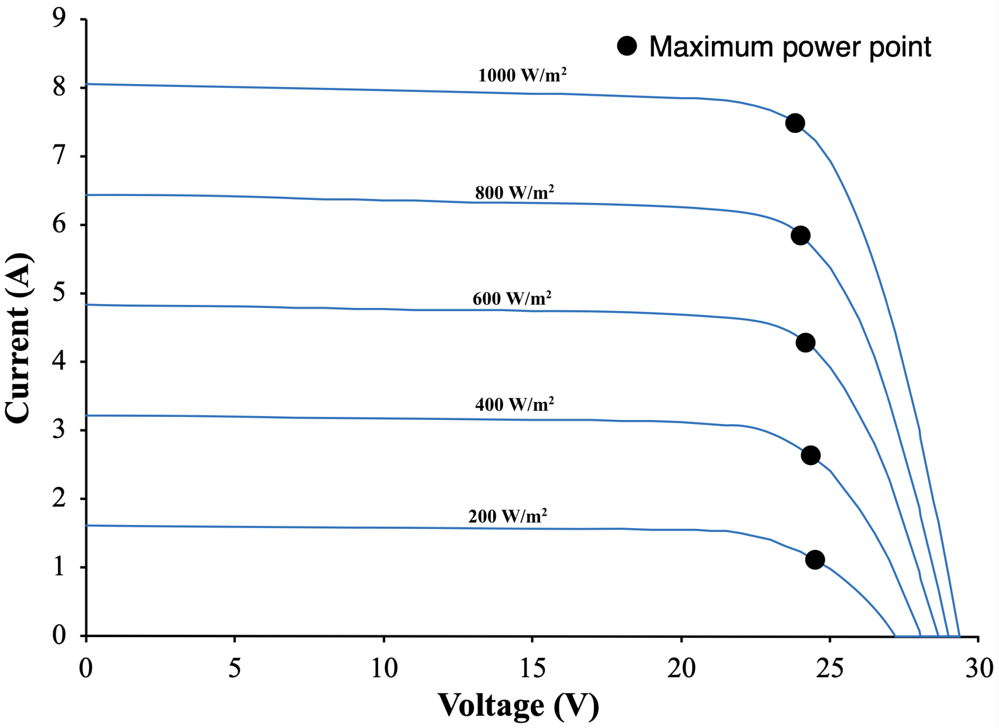


FIGURE 2.1: Typical I-V curves of a PV panel at a constant temperature and different irradiance (adapted from [58][1]).

current is almost constant and close to short-circuit current; when the terminal voltage gets close to open-circuit voltage, the output current significantly decreases and finally terminates at the open-circuit voltage.

Typical current-voltage curves of PV cells are shown in Figure 2.1, with an example of a monocrystalline cell given five values of illumination intensity from 200 W/m^2 to 1000 W/m^2. When the voltage is under 15V, the PV cell is similar to a current generator (so when the voltage increases, the power increases almost linearly). When the output voltage increases above 15V, the output current drops significantly and reaches zero at around 22V. According to this phenomenon, there is a voltage point where the cell produces the maximum power, which is named Maximum Power Point (MPP) as indicated by black dots in Figure 2.1.

In order to extract as much power as possible out of PV panels, most energy harvesting systems with PV modules adopts maximum power point tracking (MPPT) techniques [59–61]. MPPT is achieved by dynamically controlling the output voltage of PV cells around the MPP.

Outdoor sunlight and indoor artificial light are two main sources for light energy harvesting. The illumination intensity of direct sunlight on the earth's surface is typically

---

[1] Reprinted from Renewable and Sustainable Energy Reviews, Volume 32, Ciulla *et al.*, A Comparison of Different One-Diode Models for the Representation of I–V Characteristic of a PV Cell, Pages 684-696, Copyright 2014, with permission from Elsevier.

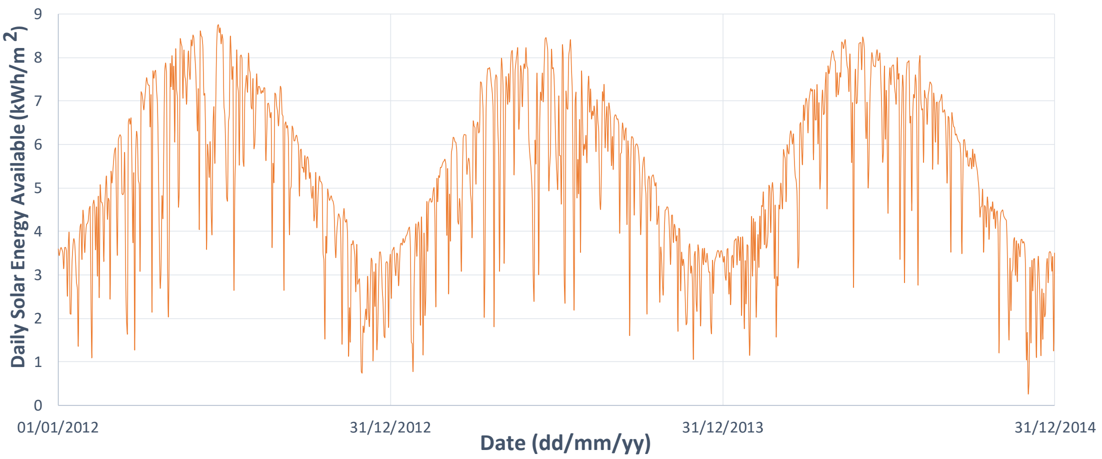


FIGURE 2.2: Daily global horizontal solar energy available in Los Angeles 2012-2014 [65].

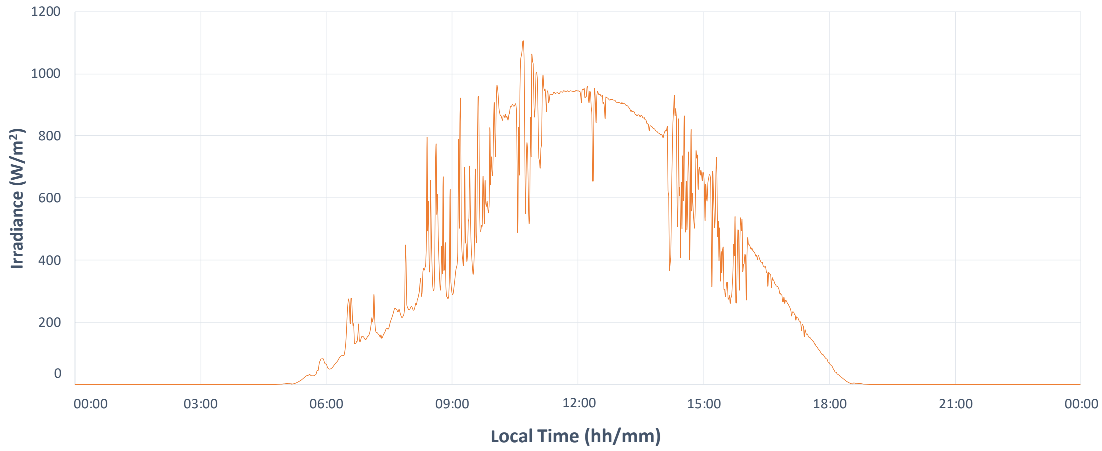


FIGURE 2.3: One-day dynamics of global horizontal solar irradiance in Los Angeles 29 April 2016 [65].

1000 W/m^2 [62], while the typical indoor illumination intensity is 10 W/m^2 [54]. Due to this large difference in the power density of these two circumstances, PV modules are more prospective in outdoor applications for harvesting solar energy. Conversion efficiency of PV cells is typically 15% to 25% in outdoor conditions [63].

Solar energy is an uncontrollable but partially predictable source [13, 64]. Solar irradiance demonstrates daily and annual periodicity due to the regularity of celestial movements, as well as irregular variations due to cloud movements, air mass, etc. A 3-year trace of diurnal global horizontal solar energy available measured in Los Angeles from 2012 to 2014 is presented in Figure 2.2, and an example of daily dynamics of global horizontal solar irradiance of the same location is presented in Figure 2.3. As shown in both figures, the predictability is reflected from the roughly annual and daily periodicity, and the uncontrollability and randomness relates to the irregular variations, which include both daily variations in an annual scale and variations over a few seconds and minutes on a daily scale.

In order to make full use of solar energy, substantial efforts have been made to develop and improve energy harvesting sensor nodes with solar panels [56, 57]. Generally, solar energy harvesting approaches adopt large energy storage, e.g. a rechargeable battery, to smooth out the daily and annual variations. Examples of solar-powered sensor nodes are presented in [12, 56, 66], and a comprehensive review on solar-powered sensor nodes is published in [11].

2.1.2 Radio Frequency Energy Harvesting

Radio Frequency (RF) energy exists in time-varying electromagnetic fields, which widely spread in our environment now due to the propagation of wireless networks, such as Wi-Fi and cellular phone signals [67]. When radio waves pass through an antenna, due to electromagnetic induction, AC voltage is generated. This AC voltage can be rectified and regulated to DC power for sensor nodes. The received RF power is reciprocal to the square of distance from the source to the destination. The maximum conversion efficiency from RF waves to DC power is typically 50-75% given a transmission distance of 100 metres [54].

Due to the widespread deployment of telecommunication networks, RF energy harvesting becomes available in a wide range of locations, both outdoors and indoors. Compared to light energy harvesting, RF harvesting shows its strength in indoor applications as there is often low or no light intensity inside buildings.

A basic and common example of RF harvesting is RF Identification (RFID). In a passive RFID application, an RFID reader transmits RF signals to an RFID tag for asking its tag information. The tag absorbs the signals and energy through its antenna, and then responds the reader with its information. Up to now, Wireless Identification and Sensing Platform (WISP) [68, 69] is presented to show the possibility of the integration of RF energy harvesting in IoT applications.

2.1.3 Flow Energy Harvesting

Flow-based energy harvesting utilises turbines and rotors to collect the kinetic energy in air flow or liquid flow. Air flow is converted by wind turbines and liquid flow is converted by hydrogenerators. Wind turbines and hydrogenerators are normally in different mechanical structures (shapes), but the fundamental principles of them are the same.

Wind turbines are manufactured in a wide spectrum in terms of dimensions, from a large-scale wind farm (arrays of large turbines) to a portable micro wind turbine. Micro wind turbines are suitable for battery charging and powering autonomous electronic devices.

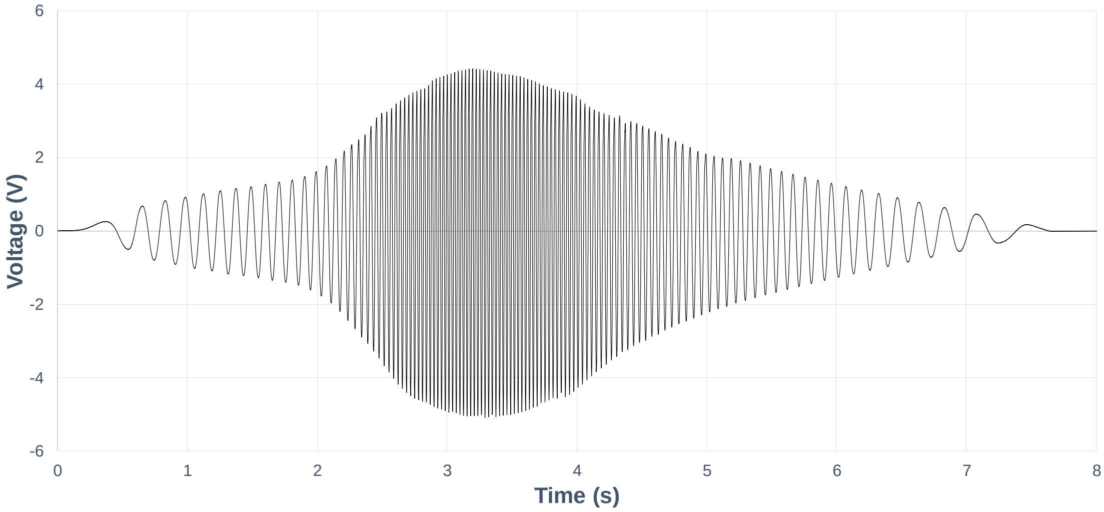


FIGURE 2.4: Dynamics of a micro wind harvester (reproduced from [70] © 2016 IEEE).

A raw voltage output trace of a micro wind turbine given a blast of wind is presented in Figure 2.4. Given a constant blow, a wind turbine should produce a sinusoidal voltage signal. Its voltage output vibrates from the positive domain to the negative domain with time, so a rectifier is normally required in order to utilise this AC power for DC load.

Similar to solar energy, wind energy is uncontrollable but partially predictable. Sharma *et al.* [71] introduce a system that achieves available wind energy predictions based on downloaded weather forecast information within recent 3 days. Also, Cammarano *et al.* [72] present a wind and solar energy predicting method which dynamically adjusts its time horizon of prediction in order to achieve higher accuracy than its prior methods.

Hydrogenerators harness the energy in moving liquids, such as water or a mix of different liquids. Traditionally, hydrogenerators are used for generating large-scale electricity from rivers and streams. However, since the possible underwater applications in IoT, hydrogenerators can be a suitable alternative for powering sensor nodes. For example, Morais *et al.* [73] incorporate a small-sized hydrogenerator as a part of energy harvesting supply for sensor nodes.

2.2 Energy Storage for Energy-Harvesting Systems

Energy harvesting supply is variable and intermittent over time, causing disparity between power supply and power consumption. In order to deliver stable power output from a varying source, a critical component in an energy harvesting power unit is energy storage, which buffers the harvested energy and powers the load when needed. Besides its ability to buffer energy and its effect on overall efficiency, energy storage has a dominant effect on the size, cost, and lifetime of sensor nodes [17]. Therefore, how to design energy storage is a critical concern in deploying energy harvesting sensor nodes.

Technologies of energy storage used in sensor nodes are generally divided into two categories: rechargeable batteries and capacitors, which are different from each other in terms of energy density, power density, lifetime, discharging features, leakage, etc. In general, batteries have higher energy density (containing more energy with the same volume/weight), lower leakage, and a more stable discharging curve (a stable voltage output while discharging), while capacitors have higher power density (higher limits for charging/discharging current), and longer lifetime in terms of charge-discharge cycles [17, 56]. The choice of these two forms of energy storage depends on application requirements. These two technologies and their implementations will be briefly reviewed in the following subsections.

2.2.1 Rechargeable Batteries

Batteries are more energy-dense than capacitors and manifest a stable voltage output when discharging. Rechargeable batteries have been widely adopted in mobile devices. Rechargeable batteries are generally made in the following techniques: Sealed Lead Acid (SLA), Nickel Cadmium (NiCd), Nickel Metal Hydride (NiMH), Lithium Ion (Li-ion), and Lithium ion Polymer (Li-Po). Due to the similar techniques and features of Li-ion and Li-Po batteries, Li-ion will be used to represent Li-ion and Li-Po batteries in this subsection. SLA and NiCd batteries are less likely to be implemented in energy harvesting sensor nodes [17, 56]. SLA batteries suffer from low energy density and are normally bulky and heavy, which is unfavorable for sensor nodes. NiCd batteries involve memory effect, i.e. decrease of energy capacity after repeated partially discharging and recharging, which is a common situation in energy harvesting implementations.

Compared to SLA and NiCd batteries, NiMH and Li-ion batteries show a strength in energy density in both weight and volume, and hence, are more suitable for energy harvesting applications [17, 56, 74, 75]. A comparison of two commercial NiMH and Li-ion batteries is listed in Table 2.2 with a variety of perspectives and features. Li-ion batteries are typical lighter than NiMH batteries, with weight energy density 2-3x and volume energy density 1-2x to NiMH batteries. Also, Li-ion batteries significantly outperform NiNH batteries in terms of charging efficiency and self-discharge rate. However, Li-ion batteries are normally more expensive than NiMH batteries, and require more complicated pulse charging circuits [56]. NiMH batteries also provide a relatively constant voltage supply during discharging [12].

	NiMH (Panasonic BK150AA)	Li-ion (EEMB LIR14500)
Nominal voltage	1.2 V	3.7 V
Charge capacity	1500 mAh	750 mAh
Energy capacity	1.80 Wh	2.775 Wh
Weight	26 g	20 g
Dimensions	⌀14.5mm × 50.5mm	⌀14.1mm × 48.5mm
Weight energy density	69 Wh/Kg	139 Wh/Kg
Volume energy density	216 Wh/L	366 Wh/L
Operating temperature	-20°C to 65°C	-20°C to 60°C
Charging cycles (until 80% capacity)	>500	>300
Reference price	£2.91	£3.25
Charging efficiency [75]	66%	99.9%
Self-discharge [75]	30% per month	10% per month
Charging Method [75]	Trickle	Pulse

TABLE 2.2: Comparison between commercial NiMH and Li-ion rechargeable batteries.

NiMH and Li-ion batteries have been widely implemented in energy harvesting sensor nodes. Heliomote [56] uses two NiMH batteries in series to match the charging voltage (2.2-2.8V) with the MPP of the solar panel. HydroSolar [74] also adopts two NiMH batteries to avoid the Li-ion charging hardware. Jiang *et al.* [21] design a hybrid storage system including a lithium based rechargeable battery as the secondary buffer, due to its high efficiency and charge density.

Despite the high energy density and stable discharging voltage, batteries still show a typical drawback at short lifetime (less than 5 years [20]), which involves manual replacement of batteries or devices after the battery lifetime expires. Also, batteries raise environmental concerns due to the heavy metals and toxic chemicals within. If not properly charged, Li-ion batteries can cause safety issues, i.e. explosion and fire, which are problematic when deployed in distant and wild places. In addition, rechargeable batteries are susceptible to temperature. Most batteries only exhibit their rated characteristics around 20°C, and lose their efficiency and capacity when operating at extreme temperatures (around their rated limits) [75].

2.2.2 Capacitors

Due to the lifetime limits and pollution issues of batteries, capacitors, typically supercapacitors, are considered as an alternative to replace rechargeable batteries as energy storage. Supercapacitor (also known as ultracapacitors or electrostatic double-layer capacitors) are capacitors with higher energy density than electrolytic capacitors. Unlike conventional capacitors, where charges are stored and separated by solid dielectric, supercapacitors maintain charges based on double-layer or pseudo-capacitive charging phenomena [76]. Supercapacitors are still much less energy-dense than batteries, but act as a transition from capacitors to batteries.

Compared to rechargeable batteries, supercapacitors exhibit strengths in a large number of charge/discharge cycles, long lifetime (20 years), high charge/discharge efficiency (98%). The self-discharge rate of supercapacitors is higher than batteries, with 5.9-11% of maximum capacity per day [77, 78], but this leakage is insignificant compared to the small capacity and the total energy gained per day. The main constraint of supercapacitors is still the low energy density, which results in large storage dimensions if the aim were to achieve a comparable capacity with batteries. In order to maintain the same form factors of sensor nodes, designers have to adapt system architecture to a small storage (compared to batteries).

Prometheus [21] introduces supercapacitors into energy storage for sensor nodes whereby two 22F supercapacitors are used in combine with a Li-Po battery. AmbiMax [79] also proposes a hybrid storage design similar to Prometheus, but with two more 10F supercapacitors for wind energy harvesters. To achieve longer lifetime than battery-based sensor nodes, Everlast [22] demonstrates the feasibility of replacing batteries with supercapacitors in energy harvesting sensor nodes, designing a power system that adopts a 100F supercapacitor as the only energy reservoir.

However, farad-level supercapacitors occupy a significant part of device volume. The advent of energy-driven computing [80] introduces the application and design scenario where execution happens only if there is energy available. Within this scenario, energy storage using millifarad-level supercapacitors are investigated in energy harvesting sensing applications [53, 69]. Furthermore, intermittent computing, which will be illustrated in the next section, enables computation given intermittent power, making progress with electrolytic capacitors or even without dedicated storage (only microfarad-level parasitic capacitance).

2.2.3 Discussion

To summarise, due to the requirements on lifetime, environmental-friendliness, and form factors in energy harvesting sensor nodes, the energy storage designs have transformed from batteries to supercapacitors, and eventually eliminated the need for dedicated storage.

Batteries have been the preferable choice for buffering harvested energy and powering sensor nodes because they make sensor nodes easy to program and operate reliably until the battery lifetime expires. However, the environmental issues and short lifetime of batteries limit the deployment of ubiquitous sensors. Supercapacitors avoid the problems of batteries and have been used to replace batteries, but the low energy density of supercapacitors also makes sensor nodes bulky and heavy in order to achieve sufficient capacity for uninterrupted operations. Recent development of intermittent computing enables forward execution over power outages and encourages storage-less designs in energy harvesting sensor nodes.

Although the minimum need for storage capacity to operate sensor nodes decreases with the evolution of computing techniques, decreased storage does limit the flexibility of energy usage. A storage-less system has to consume the incoming power immediately, otherwise the energy is wasted. This fact consequently restricts the application scenarios of storage-less systems to energy-driven applications, where execution needs to run only when there is available energy sources to harvest. However, energy-driven applications do not cover all the demands in IoT, so simply reducing the storage need is not always desirable. A wider spectrum of storage designs should be explored to suit and optimise for different application scenarios.

2.3 Energy-Neutral Computing

Energy-neutral (EN) computing aims to operate sensor nodes with at least a certain performance level over a period of time. Energy-neutrality can be described as the following equation:

$$E_{min} \leq E_{t_0} + \int_{t_0}^{t_0+\Delta t} [P_h(t) - P_c(t)]dt \leq E_{max} \tag{2.1}$$

where $P_h(t)$ and $P_c(t)$ are the harvested and consumed power at time t, t_0 is the time when EN computing is meant to start, Δt is the length of period during which EN conditions are achieved, E_{t_0} is the initial available energy in energy storage at time t_0, E_{min} is the minimum amount of stored energy below which the system cannot sustain (typically due to insufficient supply voltage), and E_{max} is the maximum capacity of energy storage beyond which the harvested energy is wasted. $P_c(t)$ includes the power consumption of the whole system, such as the MCU, peripherals, power conversion circuit, and the power leakage of energy storage. $P_h(t)$ is the harvested power after conversion.

EN devices are typically powered by solar cells [81], and Δt is typically 24 hours or one year to suit the period of the solar energy source. In order to achieve energy neutrality over such a long term, sufficient amount of energy storage, typically in the form of rechargeable batteries, is required to smooth out the large temporal variations of harvested power. The capacity of the energy storage is determined by how long the system tries to maintain a stable performance as larger energy storage tolerates more energy differences. In general, the length of Δt and the difference between P_h and P_c determine how much storage is required, and on the other hand, the capacity of energy storage limits how long Δt can be.

In order to ensure that the system works uninterruptedly by managing the stored energy (the middle term in Equation 2.1) between E_{min} and E_{max}, EN computing dynamically adapts system performance and power consumption over the period Δt. Typical adapting techniques include adjusting workload duty cycles and participation in network activity [80].

Kansal *et al.* [12] illustrate a preliminary power management algorithm by which the incoming energy is estimated by an Exponentially Weighted Moving Average (EWMA) of the past recorded slots of harvested energy, and the system tries to exploit the harvested energy by scaling its duty cycles. Vigorito *et al.* [82] introduce a Linear-Quadratic Tracking (LQT) approach to scale duty cycles based on the current battery level, and as evaluated in its datasets, mean duty cycle is improved by 6-32% and duty-cycle variation is reduced by 6-69% compared to [12], which means the system works with a more a stable performance. In [14], a Proportional-Integral-Derivative (PID) controller is used for monitoring and stabilizing the voltage of a supercapacitor-based energy storage, and hence, the storage level of this system. While these approaches achieve satisfactory energy neutrality for the magnitude of hours, they all show a latency when responding to the harvested power, and high variance of duty cycles when adapting to a new power trace. Additionally, approaches in [82] and [12] rely on an accurate estimating algorithm to detect the remaining battery energy, which is vulnerable to deployed time and temperature.

In [83], a prediction algorithm for solar energy named Weather-Conditioned Moving Average (WCMA) is presented, in which both the current and the past weather data are taken into account to achieve higher accuracy than EWMA methods. It is reported by the authors that the average prediction error is improved from 28.6% in EWMA to 9.8% in WCMA over a test duration of 45 days, but it is unclear in the article that how to harness this prediction to improve the system performance. Similarly, weather forecast is adopted in [71], by which the authors build a model to approximate the available solar and wind energy. Although these two methods based on weather data provide high prediction accuracy, the network overheads for receiving these data are not presented, and how to fully utilise this daily prediction is still a problem.

Different from the aforementioned daily EN operations, a long-term annual power management based on duty-cycling is presented in [13] to achieve annual energy neutrality. The authors use an adjustment factor, which is dynamically calculated from the historical windows, to modify the design-time energy prediction model to a more realistic model, and determine its performance level accordingly. However, this algorithm is only tested in simulation instead of practical experiments. Moreover, for such a long-term EN operation, a large battery is required, but the battery deterioration is ignored in their analysis.

In [15], a task scheduling algorithm for optimising the performance of an energy harvesting system (typically based on PV harvesters) is exhibited. Given a predicted power trace, storage bounds, energy consumption of tasks and quality of tasks, the proposed scheduling algorithm is proved to be able to find the optimal scheduling in a pseudo-polynomial time which leads to the maximum sensing quality. While this algorithm provides an ideal solution for power management, it requires that the energy source should be predictable with high accuracy, and the energy cost and quality of each task should be defined at design time. The first requirement almost constrains this algorithm within the cooperation of solar energy. The second requirement is hard to achieve since a) in practice the energy consumption of tasks may change due to temperature and dynamic data amount [84] and b) the energy cost of a system includes many elements other than the energy consumption of computing tasks.

EN computing efficiently utilises energy and maintains system performance, ensure reliability and periodic task execution despite variable harvesting power input. However, in almost all energy neutral approaches reviewed above, a large energy storage, i.e. a rechargeable battery, is in need in order to buffer temporal energy variations. The usage of batteries poses sustainability challenges due to the limited lifetime and pollution issues. Recent research develops IPSs to minimise the need for energy storage. Next, Section 2.4 reviews the existing IPS methodologies with respect to their focuses.

2.4 Intermittently-Powered System

Energy harvesting provides an autonomous power supply for wireless sensor nodes as an alternative of battery power. However, with small storage, energy harvesting systems inevitably suffer from frequent power outages, which affect forward execution of programs. Intermittently-Powered Systems (IPSs, also known as transient computing systems or intermittent computing systems) aim to maintain forward execution and computation correctness through power failures [28]. Intermittent execution spans its execution and intermittently computes over power outages, while conventional execution restarts after power interruptions. A typical characteristic of an IPS is that it starts executing whenever there is power available and suspends during power outages; after power recovery, it can continue its prior task correctly instead of restarting from the beginning of a program.

Due to different design considerations, IPS methodologies varies in a wide spectrum [85]. These methodologies include saving snapshots of system volatile state into NVM at checkpoints, breaking down execution into small atomic tasks, and hardware circuits for suspend and restore operations. According to the main features in execution styles and focuses, the existing research work on IPSs can be classified into five groups: static IPSs, reactive IPSs, power-neutral computing, energy allocation methods, and tools for simulation and emulation. The following part of this section explains each of the above groups of methodologies with the associated research progress.

2.4.1 Static IPS

2.4.1.1 Static Checkpointing

An static checkpointing IPS require manual insertion of checkpoints into code at compile time. When a checkpoint is called, the system checks the current available energy amount. If this amount is less than a predefined threshold, which indicates the available energy may not be enough to sustain execution, a snapshot saving function is called at this checkpoint. To save a snapshot of the system computing state, the system copies the current stacks and heaps, local and global variables, general registers, the stack pointer, and the program counter, into the NVM. A static checkpointing system continuously operates until it encounters power outages, where the supply voltage is less than the minimum operating voltage of the systems. After the supply voltage recovers, the system restores its state from the last checkpoint, and hence, continues its execution from that checkpoint.

Mementos [28] first presents a static checkpointing solution in which checkpoints are placed at design time. Mementos includes three strategies for placing checkpoints, which are placing at every loop, placing at every function call, and an auxiliary timer delay to determine the minimum cycle between two adjacent checkpoints. Additionally, programmers can also insert or delete checkpoints manually as a custom option. Two NVM blocks are used and snapshots are saved to the two blocks alternately, so there is always at least one available and complete snapshot even if the energy is depleted during saving a new snapshot. One significant shortcoming of Mementos is the instrumenting strategy: with the different sizes of loops and functions, the granularity of checkpoints can be either too small, which introduces high run-time overheads, or too large, which leads to non-termination where the execution can never get to the next checkpoint. Another concern in Mementos is how to set the voltage threshold so that snapshots can be saved successfully while avoiding too many redundant snapshots.

HarvOS [29] is proposed to improve the strategies of inserting checkpoints in Mementos. HarvOS analyses the control-flow graph of a program and splits it into sub-graphs with a checkpoint inserted for each sub-graph. To reduce the number of checkpoints compared to Mementos, the size of sub-graphs is set close to the worst-case number of useful cycles the MCU can execute until the next checkpoint. To reduce the size of snapshots, the RAM usage in each sub-graph is analysed and the checkpoint is placed at the point with the least RAM usage. HarvOS claims to reduce 68% checkpoints on average compared to Mementos.

Chinchilla [30] proposes a static checkpointing tool which automatically overprovisions checkpoints at compile time and adaptively eliminates unnecessary checkpoints at run time. Compared to Mementos and HarvOS, Chinchilla relieves the programming efforts on manually inserting checkpoints while still achieves an efficient number of checkpoints at run time.

An advantage of the checkpointing method is the size of a specific snapshot can be estimated from the program execution flow to find a smaller snapshot [29]. However, there are still two significant challenges remaining unsolved in static checkpointing methods: idempotency violation and non-termination.

An execution is idempotent if it can be repeated while maintaining the same result [32]. Non-idempotent actions include I/O operations and NVM writes, which are fairly common in IPS applications. Repeating non-idempotent actions can leads to undesired results, so these non-idempotent actions should be executed only once. Checkpointing systems repeat executing the code between two adjacent checkpoints, and hence, cause non-idempotency. Current compile-time checkpointing methods as listed above are not able to ensure idempotency.

Non-termination in static checkpointing systems exhibits when the energy consumption between two checkpoints is more than the buffered and harvested energy. Non-termination typically happens when the instrumentation strategy of checkpoints ignores the size of the energy buffer, as in Mementos. HarvOS and Chinchilla manage to mitigate non-termination, but they cannot eliminate this problem as they cannot dynamically insert checkpoints at run time according to varying environmental sources.

2.4.1.2 Harvest-Store-Use

Harvest-store-use IPSs perform a complete task in one consecutive period when the harvested energy in energy storage is enough. A complete task typically includes sensing, processing, and transmitting actions. In order to sustain a successful task execution, the required capacity of energy storage is larger than the minimum required storage in other IPS methodologies. Harvest-store-use systems need to calibrate the energy consumption of the task at design time and set an energy threshold to trigger execution based on that energy consumption. When the threshold is reached, which means there is enough energy for a task, the system performs one task and sleeps until the next threshold trigger. As shown in Figure 2.5, the behaviours of the amount of stored energy can be seen as alternating in turn between two states: the collecting state and the executing state.

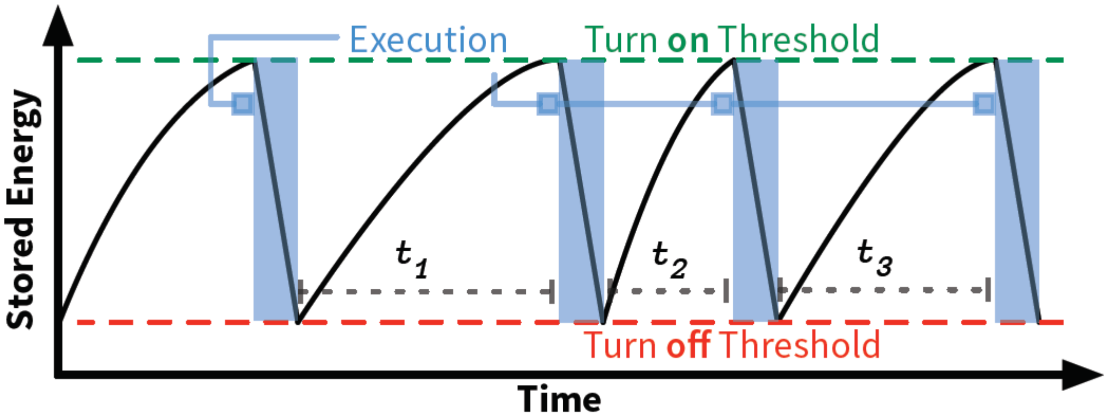


FIGURE 2.5: Harvest-Store-Use execution (taken from [86] © 2017 ACM).

Monjolo [43] is an early design following the harvest-store-use pattern. Monjolo presents a home power meter, whereby a current transformer is installed around the main power cable and provides the energy for this metering system. When the energy stored in a 500μF capacitor reaches a predefined amount, the system transmits a data packet. Another wireless receiver keeps collecting these packets and approximates the power of the main cable based on the receiving frequency of packets. Such a system contains little sensing and processing work on the transmitting node, and instead, it treats the intensity of energy sources as the sensing data, and processes this translation of data on the receiving node which is powered stably.

WISPCam [69] is a wireless camera that obtains energy from an RF harvester. The harvested energy is stored in a 6mF supercapacitor and the data (photos) are saved in NVM. Once the energy is sufficient for taking one photo, the system starts execution and depletes the energy for taking a picture and data transmission.

Similarly, Dynamic Energy Burst Scaling (DEBS) [53] also wakes up and executes tasks when there is enough energy in the 80μF capacitor. The major difference between DEBS and the above two approaches is DEBS can adjust the energy thresholds dynamically for a set of different tasks and generates energy bursts according to which task is in need.

Harvest-store-use paradigms are suitable for occasions where the harvested power is too weak to support the power consumption of any normal execution (other IPS methodologies may quickly deplete energy storage and make little progress). Also, harvest-store-use methods circumvent the idempotency issues by complete tasks in one burst. However, this pattern is task-based, which means its operation is limited to one or several fixed energy-defined tasks and also relies on high-quality design-time profiling of tasks.

2.4.1.3 Task-based

Task-based IPSs decompose a program into a series of atomic tasks, which only deliver non-volatile results after all operations in a task are completed [32]. Task-based IPSs are achieved by programming and execution models, which aim to ensure NVM consistency and idempotency. In such models, accesses to NVM and I/O operations are carefully managed to prevent idempotent violations. To ensure idempotency, the program control flow is divided by task boundaries, and the communication between tasks is enabled by reading or writing NVM data on those boundaries. To avoid non-termination, the maximum size of one task is limited by the capacity of energy storage. Therefore, a task-based IPS can be seen as a rigorously-organized and fine-grained checkpointing method, which eliminates the the non-termination and idempotency problems in static checkpointing IPSs. Task-based systems feature with fast suspend and restore operations because only the runtime and the current task should be versioned and restored through power outages [85].

DINO [32] proposes the first task-based programming and execution model, illustrating the task-based idea and providing a basic groundwork. DINO implements the programming and execution model on the LLVM compiler for C code, with program libraries and compiler passes. Chain [33] improves DINO data flows with ”Channels”, which is dedicated to manage non-volatile data, guaranteeing the correctness on applications with both idempotent and non-idempotent code. Alpaca [34] introduces data privatization which reduces memory usage compared to Chain.

A main drawback of DINO, Chain, and Alpaca is they require great programming efforts for programmers to understand the implemented libraries and redesign a program according to the task-based concept. A recent work, CleanCut [87], proposes an auxiliary tool to check and automatically decompose the non-terminating tasks (the energy consumption of which exceeds the capacity of system energy storage).

Also, like static checkpointing, task-based IPSs inevitably involve re-execution. Alpaca, the state-of-the-art task-based approach, reports a run time overhead of 1.3-3.6x compared to plain C code given constant power supply.

2.4.2 Reactive IPS

2.4.2.1 Reactive Checkpointing

Instead of instrumenting checkpoints at compile time, reactive checkpointing IPSs do not set predefined checkpoints but save snapshots at run time when the supply voltage is detected to be lower than a threshold that indicates an imminent power failure. Therefore, the snapshot saving operations is only invoked when there is an indication

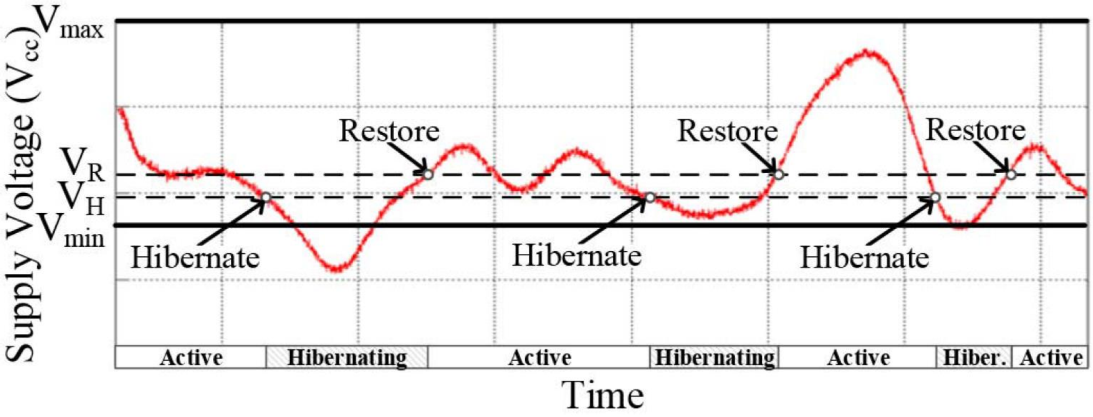


FIGURE 2.6: Voltage trace with hibernation and restoration points in Hibernus (taken from [37] © 2015 IEEE).

of an imminent power outage, i.e. a low supply voltage. Also, after saving a snapshot, a reactive IPS suspends its execution and enter a low-power mode, rather than continues execution until a power outage as checkpointing systems do. When the voltage supply recovers above a restore threshold, the system either restores the last snapshot if the system reboots, or just continues execution if the system comes back from the low-power mode.

Hibernus [37] saves only one snapshot before a power interruption and then enter the sleep mode. Two fixed voltage thresholds, V_H and V_R, are predefined for hibernation (save a snapshot and sleep) and restoring a snapshot. An on-chip voltage comparator and an on-chip voltage reference generator are used for monitoring the supply voltage and triggering hibernation when the supply voltage drops to V_H or restoration when the supply voltage recovers to V_R. A voltage trace is shown in Figure 2.6 to explain Hibernus behaviours, and this trace is representative for a reactive IPS behaviours. To adapt thresholds to variable energy sources, Hibernus++ [38] implements dynamic self-calibration for suspend and restore thresholds by executing a hibernation test. By using adaptive thresholds instead of fixed thresholds as in Hibernus, Hibernus++ makes itself compatible with a variety of energy sources. Compared to Hibernus, Hibernus++ improves application execution time by reducing the overheads of suspend and restore operations.

Quickrecall [36] is a similar approach to Hibernus except replacing RAM with NVM, so that all the run-time volatile data become non-volatile and only registers are necessary to be saved in a snapshot. An external voltage comparator detects a triggering voltage V_{trig} to back up only peripherals and registers. Compared to the voltage thresholds in Mementos and Hibernus, V_{trig} in Quickrecall is lower since the reduced energy and time overheads for saving and restoring a snapshot. However, using NVM as RAM may lead to the higher cost of NVM accesses. A comparison between Hibernus and Quickrecall is presented in [88], showing that Quickrecall performs worse when the frequency of power interrupts is low as the NVM consumes more than volatile RAM, and performs better when the frequency of power interrupts is high as the overheads of saving snapshots are much lower.

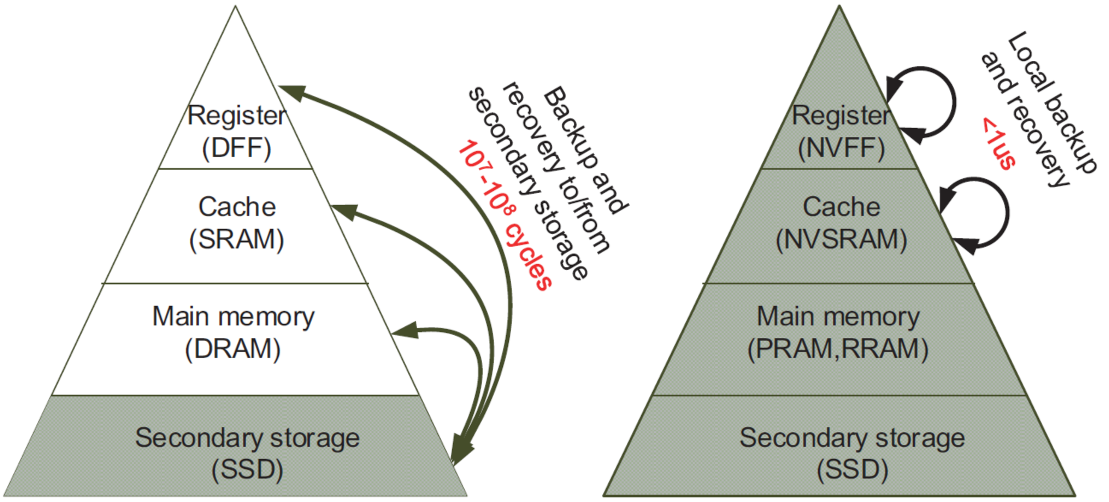
Reactive IPS methods only save snapshots when power failure is imminent, and hence, reduce the number of snapshots compared to checkpointing methods. Also, reactive IPSs avoid code re-execution by suspending execution after saving a snapshot, and hence, ensure idempotency.

The RAM usage varies at run time, so the size of snapshots in reactive checkpointing also varies throughout code execution. To circumvent this issue, Hibernus saves the entire RAM in each snapshot while Quickrecall does not use RAM at all. Comparing Hibernus to checkpointing methods, the overheads of saving snapshots in Hibernus is larger as checkpointing methods can avoid saving large snapshots by analysing the program. Such high saving overheads becomes significant when the frequency of power outages increases. Increasing the size of energy storage in reactive checkpointing should be helpful to mitigate frequent snapshot taking because the increased energy storage can filter the variations of supply voltage and avoid frequent voltage drops.

2.4.2.2 Non-Volatile Processors

Non-Volatile Processors (NVPs) incorporate automatic backup and restore hardware within the chips. A comparison of memory architecture between traditional processors and NVPs is shown in Figure 2.7. The traditional volatile elements are replaced with non-volatile elements to achieve efficient backup and restore operations with a faster speed and lower energy consumption than the conventional memory architecture. To be specific, the registers and cache are equipped with built-in additional non-volatile backup and restore circuits, so that when the supply power is going to disappear, the computing state can be saved locally just beside the elements, rather than being copied out into an external NVM. It is reported that the backup and restore speed of NVPs can be 2-4× magnitudes faster than the state-of-the-art NVM based commercial processors [89].

Wang *et al.* [90] present a preliminary NVP with 3μs backup time and 7μs restore time, which enables the processor to operate safely under a 20 kHz square wave of power. As a comparison, the existing MCUs in TI MSP430 family can only achieve 212μs and 310μs for saving and restoring states respectively. Su *et al.* [91] extend the backup and restore time overheads to a system level, presenting a NVP with 46μs system-level wake-up time and 14μs system-level sleep time. Liu *et al.* [92] integrate a NVP into a system-on-chip with independent backup and restore circuits for peripherals.


Register
(DFF)
Cache
(SRAM)
Main memory
(DRAM)
Secondary storage
(SSD)
Backup and
recovery to/from
secondary storage
10⁷-10⁸ cycles
Register
(NVFF)
Cache
(NVSRAM)
Main memory
(PRAM,RRAM)
Secondary storage
(SSD)
Local backup
and recovery
<1us

(a) Conventional memory hierarchy and data backup scheme

(b) New memory hierarchy and data backup scheme

FIGURE 2.7: Memory architectures of traditional processors and NVPs (taken from [89] © 2015 ACM).

NVP-based research follows with the development of NVP hardware. Ma *et al.* [93] examine the performance and energy consumption of several types of NVPs with different ambient sources, providing a guideline for NVPs selection. The concept of ”Incidental Computing” based on NVPs is proposed in [94] to improve the forward progress under unstable power supply, and also provides an evaluation of performance on NVPs. Essentially, it pays more attention to processing forward data in need than the buffered historical data from recovery, but an incidental recomputing on the historical data is performed when there is abundant energy. It is reported that this approach outperforms the existing save-and-use computing scheme by 2.2-5× in the simulation with respect to an image processing speed, and also the forward progress is improved by 4.28× on average over a basic NVP.

NVPs perform well in terms of the response to power intermittency, but the research on how to deliver better forward progress with NVPs is limited. Dynamic Voltage and Frequency Scaling (DVFS) can be a potentially applicable solution [95]. In a traditional NVP, the small buffering capacitor tends to be either charged to be full or depleted rapidly and frequently [96]. This behaviour accounts for a large part of backup and recovery overheads, so power management based on NVP is in need.

2.4.3 Power-Neutral Computing

While the above IPS methods aim to ensure forward execution despite frequent power outages, energy harvesters may also generate more power than systems can consume when ambient sources are sufficient. Such excessive energy is wasted if not stored for later usage or consumed immediately.

2.4.3.1 Principles of Operation

Power-neutral (PN) computing aims to manage power without additional storage or with only a very limited amount of storage which can only sustain its system for milliseconds. In principle, power-neutral computing is a special case of energy neutral computing when Δt in Equation 2.1 is equal (or close) to zero. Technically, PN computing scales the instantaneous system power consumption to match the instantaneous harvested power with theoretically zero storage (in other words, energy neutrality is met instantaneously). PN operations can be translated into the following expressions:

$$P_h(t) = P_c(t) \tag{2.2}$$

$$where \quad t \in \{t | V_{cc}(t) \geq V_{min}\} \tag{2.3}$$

where V_{cc} is the input voltage of the computing load, and V_{min} is the minimum voltage required for the system to operate. Equation 2.2 describes the methodology of power neutrality (dynamic and instantaneous power adaptation). Equation 2.3 limits the requirement for power neutrality that the system should be powered and active to make reactions of performance scaling. This requirement may change according to different system designs, but for contemporary computing and sensing loads, this is determined by the supply voltage.

Given a very limited amount of storage and a range of scalable performance and power consumption, PN computing scales down performance if P_h is lower than P_c, such that V_{cc} remains stable, which extends execution time and avoids suspend and restore operations. On the other hand, PN computing scales up performance if P_h is higher than P_c, such that the excessive harvested energy is immediately consumed on useful work rather than wasted.

In practice, however, there does not exist a system that can adjust its power consumption instantaneously to the harvested power without any overheads. Any performance scaling costs a small amount of time and energy overheads, which a system cannot afford without any energy storage. Therefore, a minimum storage is still required, normally in the form of decoupling or parasitic capacitance, to provide a small but sufficient amount of energy for scaling performance and adapting power consumption.

In order to achieve power neutrality, a system has to adapt its performance and hence power consumption. Performance scaling can be achieved by hardware controlling,

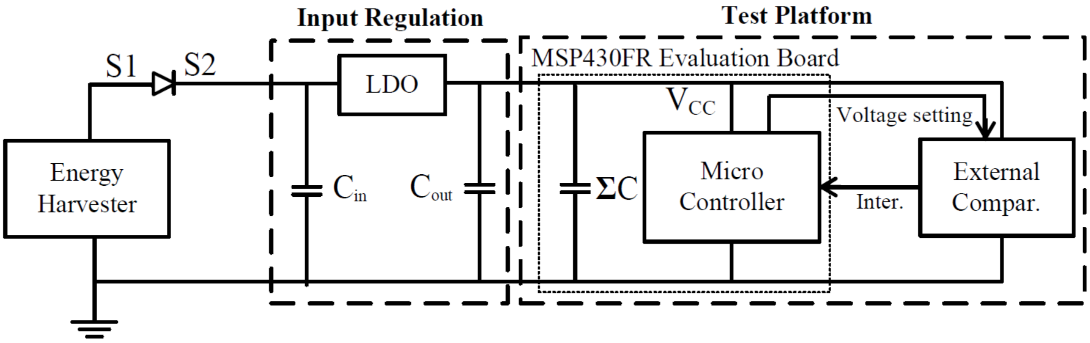


FIGURE 2.8: Architecture of an example power neutral system based on TI MSP430FR platform (taken from [70] © 2016 IEEE).

such as Dynamic Frequency Scaling (DFS) [70], DVFS [97], or switching on/off load elements [97, 98] (also known as Dynamic Power Management, DPM [99] or hot-plugging). Apart from these achieved methods, duty-cycle scaling and task scheduling are also choices for changing performance and consumption, though they have not been implemented in current research yet.

2.4.3.2 Recent Approaches

The concept of PN computing is proposed in [70] and implemented on a Texas Instrument MSP430FR5739 MCU without an external energy buffer. As shown in Figure 2.8, the executing load is directly connected to a regulated energy harvesting source. The control scheme in [70] utilises DFS with a voltage feedback. Specifically, two voltage thresholds, V_{dec} and V_{inc}, are set for detecting voltage variance caused by power inequality and then scaling performance accordingly. In order to respond fast to power difference, the capacitance is reduced to 19μF, which is only the parasitic and on-board decoupling capacitance. When $P_h(t) > P_c(t)$ and the operating voltage V_{cc} increases rapidly due to the small capacitance and reaches V_{inc}, the MCU increases its operating frequency resulting in faster computing speed and higher power consumption, and also increases the thresholds between which the new voltage value is contained; and vice versa, a reverse procedure is executed for $P_h(t) < P_c(t)$. In a word, this control scheme is trying to make the operating voltage stable around a desired value so that $P_h(t)$ equals $P_c(t)$ approximately.

A similar control scheme is adopted in [97] where the platform is an MP-SoC adopting DVFS and DPM, which leads to higher performance, higher power consumption, and more operating points than the MCU in [70]. A 47mF supercapacitor is used for safely overcoming performance switching where the power consumption of the board is normally above 2W. As an illustration for how to scale performance by DVFS and DPM,

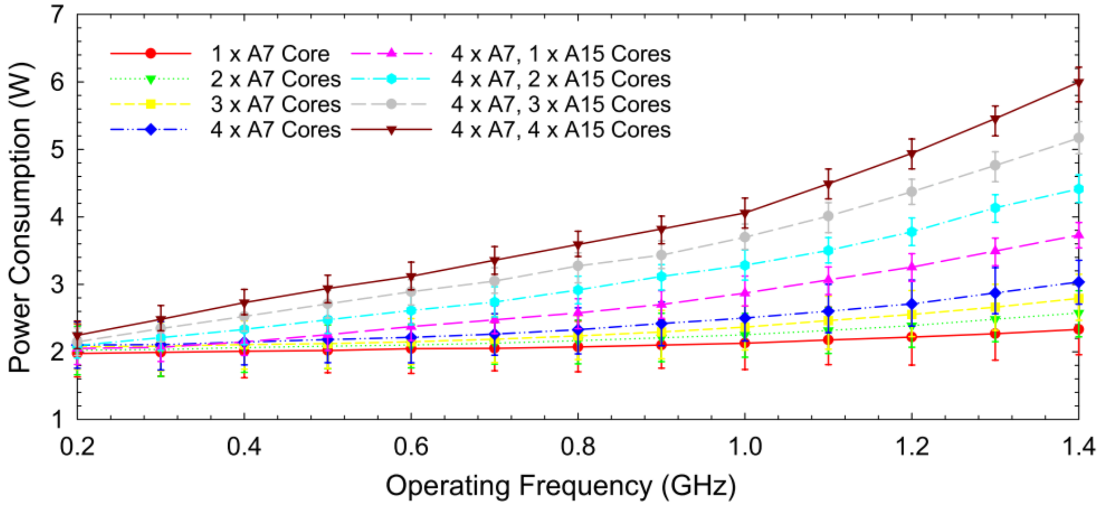


FIGURE 2.9: Board power consumption of ODROID XU4 vs operating frequency and core configurations, running CPU intensive application Raytrace (taken from [97] © 2017 IEEE).

Figure 2.9 presents an example application profile of ’power consumption vs performance’ when DVFS and DPM applied on a heterogeneous multi-processor system-on-chip (MP-SoC) platform. The SoC used in this platform is the Samsung Exynos5422 big.LITTLE SoC with four ’big’ high-performance A15 cores and four ’LITTLE’ low-power A7 cores. In this case, the performance refers to the speed of executing this application for one time and is proportional to the operating frequency under a certain core configuration. As shown in the figure, each performance level (a pair of frequency and core status, also named as an operating point) requires a certain power consumption. At run-time, the system dynamically switch its performance among these operating points so as to timely match $P_c(t)$ with $P_h(t)$.

There are three advantages in this kind of PN control scheme. First, the voltage is stabilized so it can offset ephemeral power drops which cause insufficient voltage supply and power failures, and therefore the lifetime increases (e.g. reported by 4-88% in [70]). Second, as power neutral computing eliminates many elements that required in EN systems, such as large energy storage, power converters and MPPT units, the size and cost of devices is reduced and the number of power consumption components also decreases. Third, if powered by a solar panel and the operating voltage range encompasses the MPP of the given solar panel, the system embraces an intrinsic MPPT characteristic as it can stabilize the voltage around a target value.

Similarly, Wang *et al.* [98] propose a storage-less and converter-less approach which can be classified as a power neutral system. In this design, a 47μF bulk capacitor is equipped with a 3.29mW non-volatile MCU and up to 16.5mW peripherals. This capacitor is also small enough compared to the 19μF capacitance operating with an up to 3 mW MCU in [70]. An external MPPT controlling element dynamically adjusts the power duty-cycle for the non-volatile load in order to match the harvested current and the consumed current, and hence power neutrality is met.

One disadvantage of power neutral computing is that it has to passively scale its power consumption as well as its performance, causing large variations in performance. However, this might not be good in terms of the overall forward progress. In the next chapter, a preliminary analyse is explained about how the forward progress is improved when the capacitor size is increased, while not violating the merits of PN computing.

2.4.4 Energy Allocation

Several existing designs have been able to handle atomic peripheral operations in IPSs, where energy profiling of workloads is an inherent part of their methodologies.

DEBS [53] experimentally profiles the energy consumption of each task at design time, and designates a threshold to each task individually. After completing an operation, DEBS enters a low-power mode (LPM) and waits for energy to be replenished to the next threshold.

Samoyed [40] utilises a custom design-time *energy profiler* to identify an energy storage size that suffices to run an adequate number (hundreds, as suggested) of peripheral operations in one active cycle. At runtime, Samoyed starts execution when energy is refilled to a certain threshold, and keeps executing until energy is depleted. Samoyed differs from proactive intermittent computing approaches, e.g. Alpaca [34], mainly on handling computational workloads where it reactively checkpoints when the buffered energy is below a threshold, and supports user-customised subdivision of peripheral operations when the operation cannot complete in one active cycle.

RESTOP [100] provides programmer-configurable rules that track the instructions issued to peripherals through serial interfaces in a history table. On power recovery, RESTOP re-issues instructions saved in the history table and then resumes the interrupted operation. At design time, RESTOP needs to profile the worst-case energy consumption for restoring peripheral state to identify the minimum (most-efficient) restore threshold.

2.4.5 Tools for Simulation and Emulation

To explore forward progress of IPSs, simulation tools need to represent transient operation (timescales of µs-ms) as well as long-term overall performance (from days up to years). A number of models have been proposed for exploring system designs and parameters in IPSs.

Su *et al.* [101] modelled a dual-channel solar-powered non-volatile sensor node, and Jackson *et al.* [102] provided a model to explore battery usage in IPSs. Both were configured for long-term simulations and large energy storage (from mF-scale supercapacitors to batteries), thus cannot respond to frequent power interruptions and accurately estimate forward progress when using minimized energy storage (e.g. 4.7 µF [52]).

In contrast, a set of fine-grained models have been proposed to accurately simulate the frequent micro-operations in IPSs. NVPsim [103] is a gem5-based simulator for NVPs. Fused [104] is a closed-loop simulator which allows interaction between power consumption, power supply, and forward progress. EH model [105] can compare a range of IPS approaches in a single active period with the same energy budget, quantifying forward progress by the energy spent on effective execution. These fine-grained models are inefficient for processing long-term energy data, especially when iterative tests are needed for various system configurations.

Besides models and simulators, hardware emulators of energy harvesters [106, 107] can provide repeatable power profiles recorded from energy harvesters for experimental comparisons. Though they provide practical results, hardware emulations are limited by hardware options and are generally impractical for performing long-term trials.

2.5 Summary

This chapter introduces a background of energy harvesting techniques, summarises the evolution of energy storage used in energy harvesting computing, and reviews the existing methodologies of battery-less energy harvesting computing.

EN computing emphasizes the continuous activity of devices over a long-term duration (e.g. several days, one year) by buffering harvested energy in large energy storage and adapting energy consumption ”reluctantly”. However, large energy buffers, usually in the form of batteries or large supercapacitors, are demanded for EN operations, whereas such large energy storage limits device lifespans, increases the cost, mass, dimensions of devices, and bring pollution and maintenance issues. This contradicts the design requirements of ubiquitous sensor deployments.

To circumvent the limitations in EN computing, intermittent computing is recently developed. Intermittent computing continues computation after the supply fails rather than restarts from the beginning of programs. Hence, intermittent computing devices can achieve forward execution despite frequent power failures with only minimum storage (e.g. a decoupling capacitor) to secure successful saving and restoring operations of computing states between volatile and non-volatile components. Based on intermittent computing, PN computing introduces run-time performance adaptation to match power consumption with harvested power, such that the number of saving and restoring operations can be reduced and application execution speed is increased.

However, with minimised storage, an intermittent computing device has to frequently wake up, execute for a short time, and halt when the harvested power is less than the load power consumption, consuming much energy in managing system states. As for PN computing, volatile power from environment results in significant performance variations, which then cause performance loss. The remaining part of this thesis reports a study on how to mitigate these two problems and improve system execution speed by adding a small amount of energy storage without significantly affect device dimensions.

Chapter 3

Effect of Energy Storage Sizing on IPS Performance

With the goal of minimising device dimensions and interruption periods, most IPS approaches adopt only a minimum amount of energy storage [30, 38, 50–52, 52, 70], e.g. a decoupling capacitor. This is typically just sufficient for the most energy-expensive atomic operation. However, our assertion is that this can be *inherently inefficient in terms of time and energy*. We show that a system with minimum energy storage frequently goes through a cycle of: wake up, restore state, execute program, save state, and halt. We propose that provisioning *slightly more* energy storage can prolong the operating cycles, reduce the frequency of interruptions, and hence improve forward progress.

However, the relationship between IPS energy storage capacitance and forward progress has not previously been defined. Due to the computational speedup of reactive IPSs over proactive IPSs as discussed in Section 2, we focus on reactive IPSs in this chapter. We develop an experimentally-validated model of reactive IPSs to estimate forward progress. Taking advantage of the model, we explore the effect of energy storage capacitance on forward progress with respect to supply current and volatile state size. The main contributions in this chapter can be summarised as:

- A reactive IPS model which accurately estimates forward progress; experimental validation based on a TI MSP430FR6989 MCU shows a 0.5% mean absolute percentage error across a range of current inputs and energy storage capacitance.
- An exploration based on the model, where we analyse the energy storage sizing effect on forward progress with respect to supply current and volatile state size, showing up to 65% forward progress improvement.

As discussed in Chapter 1, *forward progress* denotes the computation beneficial to the progress of the active application, excluding lost progress due to power failures and

Input Parameters	
I_{harv}	Energy harvester current supply
C	Energy storage capacitance
Configuration Parameters	
I_{exe}	Execution current draw
I_{lpm}	Low-power mode current draw
I_{r}	Restore current draw
I_{s}	Save current draw
I_{leak}	Leakage current draw
V_{r}	Restore voltage threshold
V_{s}	Save voltage threshold
T_{r}	Restore time overhead
T_{s}	Save time overhead
Output Parameter	
α_{exe}	Normalised forward progress

TABLE 3.1: Model parameters of reactive IPS.

state-saving and -restoring operations [44]. The amount of forward progress directly determines application performance, e.g. program iteration rate or task completion time. To allow fair comparison, we define normalised forward progress as *the ratio of the effective execution time to the total elapsed time*, without being restricted to a specific workload. In this chapter, the normalised forward progress, denoted as α_{exe}, is used as the key performance metric of IPSs in the modelling and experiments.

The rest of this chapter is organised as follows. The reactive IPS model is proposed in Section 3.1. The exploration on the energy storage sizing effect is presented in Section 3.3. Section 3.4 validates the proposed model and the energy storage sizing effect. Finally, Section 3.5 summarises this chapter.

3.1 Reactive IPS Modelling

To facilitate the understanding and exploration of reactive IPSs, we present a model which outputs the normalised forward progress α_{exe}. Parameters of this model are listed in Table 3.1. The model assumes that all configuration parameters remain constant. We assume that all input and configuration parameters remain constant in this model derivation, but later provide a dynamic process for cases where parameters change dynamically.

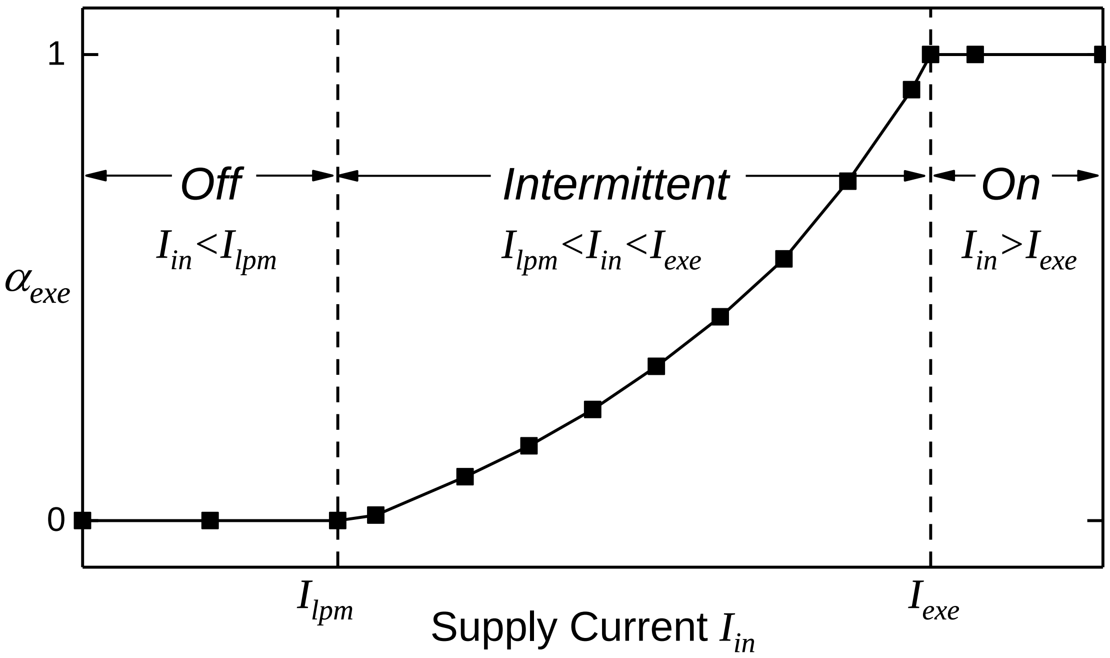


FIGURE 3.1: Operating modes of reactive IPSs, and achieved forward progress against supply current.

For brevity, I_{in} denotes the usable input current as expressed in Equation 3.1. The effect of capacitor leakage current, I_{leak}, is discussed at the end of Section 3.1.2.

$$I_{\text{in}} = I_{\text{harv}} - I_{\text{leak}} \tag{3.1}$$

3.1.1 Operating Modes of Reactive IPS

The behaviour of reactive IPSs can be classified into three operating modes depending on the supply current, as shown in Figure 3.1. These are differentiated by the relationship between input current I_{in} and the system's current draw in LPM or active modes, i.e. I_{lpm} and I_{exe}. We define the three modes as:

- *Off* mode: When $I_{\text{in}} < I_{\text{lpm}}$, the system stays inactive. The supply voltage V_{cc} cannot rise above the restore threshold V_{r} to wake the system and start execution. The LPM current I_{lpm} includes the consumption of voltage monitoring circuits and system idle current.
- *On* mode: When $I_{\text{in}} > I_{\text{exe}}$, the system executes constantly as the supply voltage V_{cc} never drops below V_{s}. V_{cc} grows until I_{in} and I_{exe} are in equilibrium, which may result from I_{in} decreasing due to poor impedance matching, or I_{exe} increasing due to either greater current draw at higher voltage or dissipation through overvoltage protection circuits.

- *Intermittent* mode: When $I_{\mathrm{lpm}} < I_{\mathrm{in}} < I_{\mathrm{exe}}$, the system executes intermittently after $V_{\mathrm{cc}} > V_{\mathrm{r}}$ and before $V_{\mathrm{cc}} < V_{\mathrm{s}}$. V_{cc} can rise above V_{r} and the system starts execution. However, the stored energy is then consumed by the load as $I_{\mathrm{in}} < I_{\mathrm{exe}}$, causing V_{cc} to eventually drop below the save threshold V_{s}, where the system saves its state and enters LPM. The system stays in LPM until V_{cc} rises to V_{r} again and then resumes execution. In general, a higher I_{in} leads to more forward progress in this mode, but the exact relationship between I_{in} and forward progress requires further analysis.

3.1.2 Formulating Forward Progress

Next, we derive formulations to calculate α_{exe} from I_{in} and energy storage capacitance C. We then explore the effect of capacitor leakage on maximum forward progress.

In the *On* and *Off* modes, the normalised forward progress is trivial to find (simply 1 and 0 respectively). In the *Intermittent* mode, as shown in Figure 3.2, the system goes through four intervals in turn, i.e. charging, restoring, executing, and saving, with current consumption of I_{lpm}, I_{r}, I_{exe}, and I_{s} in each interval respectively. The normalised forward progress, i.e. effective execution time ratio, is indicated as $T_{\mathrm{exe}}/T_{\mathrm{cycle}}$, where T_{exe} is the time spent on effective execution in one operating cycle and T_{cycle} is the period of operating cycles. Hence, the forward progress given all supply levels is expressed as:

$$\alpha_{\mathrm{exe}} = \begin{cases} 0 & , \; \mathit{Off}\,(I_{\mathrm{in}} < I_{\mathrm{lpm}}) \\ \dfrac{T_{\mathrm{exe}}}{T_{\mathrm{cycle}}} & , \; \mathit{Intermittent}\,(I_{\mathrm{lpm}} < I_{\mathrm{in}} < I_{\mathrm{exe}}) \\ 1 & , \; \mathit{On}\,(I_{\mathrm{in}} > I_{\mathrm{exe}}) \end{cases} \tag{3.2}$$

In the following analysis, we focus on deriving $T_{\mathrm{exe}}/T_{\mathrm{cycle}}$ in the *Intermittent* mode. Let V_{pr} (post-restore) and V_{ps} (post-save) denote the voltage after restoring and saving operations. Referring to Table 3.1 and Figure 3.2, V_{pr} and V_{ps} can be calculated as:

$$V_{\mathrm{pr}} = V_{\mathrm{r}} + \frac{T_{\mathrm{r}}(I_{\mathrm{in}} - I_{\mathrm{r}})}{C} \tag{3.3}$$

$$V_{\mathrm{ps}} = V_{\mathrm{s}} + \frac{T_{\mathrm{s}}(I_{\mathrm{in}} - I_{\mathrm{s}})}{C} \tag{3.4}$$

With Equation 3.3, the time spent on effective execution T_{exe} in one operating cycle can be expressed as:

$$\begin{aligned} T_{\mathrm{exe}} &= \frac{C(V_{\mathrm{pr}} - V_{\mathrm{s}})}{I_{\mathrm{exe}} - I_{\mathrm{in}}} \\ &= \frac{C(V_{\mathrm{r}} - V_{\mathrm{s}}) + T_{\mathrm{r}}(I_{\mathrm{in}} - I_{\mathrm{r}})}{I_{\mathrm{exe}} - I_{\mathrm{in}}} \end{aligned} \tag{3.5}$$

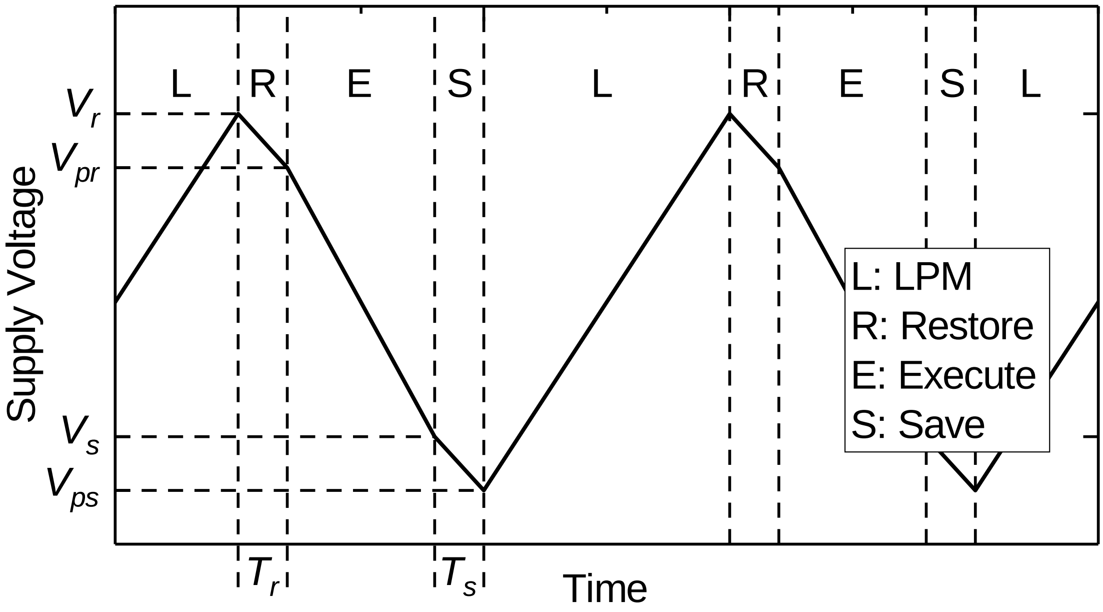


FIGURE 3.2: Operating cycles in the *Intermittent* mode.

Analogously, with Equation 3.4, the charging interval can be described as:

$$\begin{aligned} T_{charge} &= \frac{C(V_\mathrm{r} - V_\mathrm{ps})}{I_\mathrm{in} - I_\mathrm{lpm}} \\ &= \frac{C(V_\mathrm{r} - V_\mathrm{s}) - T_\mathrm{s}(I_\mathrm{in} - I_\mathrm{s})}{I_\mathrm{in} - I_\mathrm{lpm}} \end{aligned} \tag{3.6}$$

With Equation 3.5 and Equation 3.6, the period of an operating cycle is:

$$\begin{aligned} T_\mathrm{cycle} &= T_{charge} + T_\mathrm{r} + T_\mathrm{exe} + T_\mathrm{s} \\ &= \frac{C(V_\mathrm{r} - V_\mathrm{s}) + T_\mathrm{s}(I_\mathrm{s} - I_\mathrm{lpm})}{I_\mathrm{in} - I_\mathrm{lpm}} + \frac{C(V_\mathrm{r} - V_\mathrm{s}) + T_\mathrm{r}(I_\mathrm{exe} - I_\mathrm{r})}{I_\mathrm{exe} - I_\mathrm{in}} \end{aligned} \tag{3.7}$$

Finally, combining Equation 3.3 to Equation 3.7, we obtain normalised forward progress α_exe in the *Intermittent* mode ($I_\mathrm{lpm} < I_\mathrm{in} < I_\mathrm{exe}$) as:

$$\begin{aligned} \alpha_\mathrm{exe} &= \frac{T_\mathrm{exe}}{T_\mathrm{cycle}} \\ &= [\frac{C(V_\mathrm{r} - V_\mathrm{s}) + T_\mathrm{r}(I_\mathrm{in} - I_\mathrm{r})}{I_\mathrm{exe} - I_\mathrm{in}}]/ \\ &[\frac{C(V_\mathrm{r} - V_\mathrm{s}) + T_\mathrm{s}(I_\mathrm{s} - I_\mathrm{lpm})}{I_\mathrm{in} - I_\mathrm{lpm}} + \frac{C(V_\mathrm{r} - V_\mathrm{s}) + T_\mathrm{r}(I_\mathrm{exe} - I_\mathrm{r})}{I_\mathrm{exe} - I_\mathrm{in}}] \end{aligned} \tag{3.8}$$

In the numerator T_exe, $C(V_\mathrm{r} - V_\mathrm{s})$ represents the amount of charge in the capacitor available for restoring and executing. $T_\mathrm{r}(I_\mathrm{in} - I_\mathrm{r})$ represents the charge used by a restore operation. $I_\mathrm{exe} - I_\mathrm{in}$ is the rate of charge consumption from the energy storage during execution.

Also, the ratio of the time overhead on state-saving and -restoring operations to the total elapsed time $T_{\mathrm{rs\%}}$ can be deduced as:

$$\begin{aligned} T_{\mathrm{rs\%}} &= \frac{T_{\mathrm{r}} + T_{\mathrm{s}}}{T_{\mathrm{cycle}}} \\ &= (T_{\mathrm{r}} + T_{\mathrm{s}})/[\frac{C(V_{\mathrm{r}} - V_{\mathrm{s}}) + T_{\mathrm{s}}(I_{\mathrm{s}} - I_{\mathrm{lpm}})}{I_{\mathrm{in}} - I_{\mathrm{lpm}}} + \frac{C(V_{\mathrm{r}} - V_{\mathrm{s}}) + T_{\mathrm{r}}(I_{\mathrm{exe}} - I_{\mathrm{r}})}{I_{\mathrm{exe}} - I_{\mathrm{in}}}] \end{aligned} \tag{3.9}$$

Equation 3.9 can describe the impact of state-saving and -restoring overhead with respect to C and I_{in}.

To explore the effect of energy storage on forward progress, we need to analyse $d\alpha_{\mathrm{exe}}/dC$. Here, if we assume that I_{leak} remains constant, α_{exe} keeps increasing and approaches $(I_{\mathrm{in}} - I_{\mathrm{lpm}})/(I_{\mathrm{exe}} - I_{\mathrm{lpm}})$ when energy storage capacitance C increases. Defining $(I_{\mathrm{in}} - I_{\mathrm{lpm}})/(I_{\mathrm{exe}} - I_{\mathrm{lpm}})$ as $\alpha_{\mathrm{exe\_ideal}}$, $\alpha_{\mathrm{exe}} = \alpha_{\mathrm{exe\_ideal}}$ is an ideal case, where state-saving and -restoring overheads are absent.

In an electrolytic capacitor, however, I_{leak} typically increases with C with the following relationship [108]:

$$I_{\mathrm{leak}} = kCV_{\mathrm{cc}} \tag{3.10}$$

where k is a constant normally in a range of 0.01–0.03 A/(F V). Combining Equation 3.10 with Equation 3.1, dI_{in}/dC is $-kV_{\mathrm{cc}}$, meaning I_{in} decreases linearly as C increases. Thus, when C increases, α_{exe} keeps approaching $\alpha_{\mathrm{exe\_ideal}}$ while $\alpha_{\mathrm{exe\_ideal}}$ decreases. Hence, we believe that there is a capacitance value that leads to the maximum α_{exe} considering I_{leak} increases with C.

3.1.3 Dynamic process

The above model assumes all parameters are constant, which is useful for fast exploration in cases where this can be considered to approximately hold true (this is used for the analysis of principal sizing effects presented in Section 3.3). For dynamically-varying parameters (e.g. a dynamic harvesting profile), we also implement a dynamic process, where the supply voltage is calculated with system's current flows and energy storage capacitance across small time steps, hence updating system state accurately. This is used for the exploration of real-world energy conditions in Section 4.2.

3.2 Experimental Methodology

The presented model (Equation 3.8) was utilised to explore the sizing effect of energy storage. An existing IPS was implemented and profiled to configure the model. With the configured model, the relationship between forward progress and other system

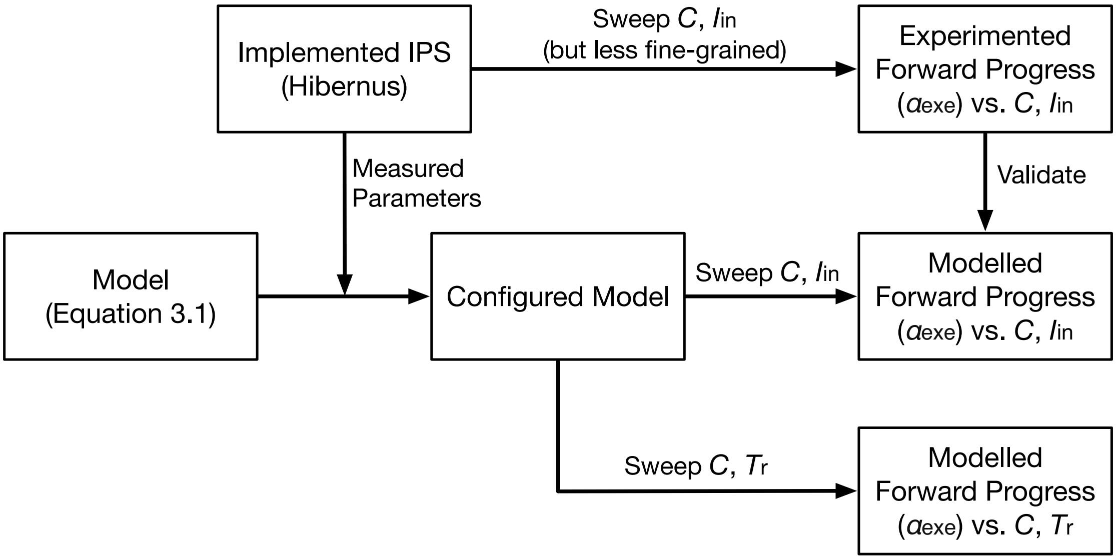

Implemented IPS
(Hibernus)
Sweep C, I_{in}
(but less fine-grained)
Experimented
Forward Progress
(α_{exe}) vs. C, I_{in}
Measured
Parameters
Validate
Model
(Equation 3.1)
Configured Model
Sweep C, I_{in}
Modelled
Forward Progress
(α_{exe}) vs. C, I_{in}
Sweep C, T_r
Modelled
Forward Progress
(α_{exe}) vs. C, T_r

FIGURE 3.3: Methodology of Model Exploration and Experimental Validation.

parameters was explored. The modelled results were then validated with the implemented IPS. For reproducing and understanding the method and results, the source code of the model and the implemented IPS is available online[1].

The modelling and experimental methodology is shown in Figure 3.3. The initial model (Equation 3.5) estimates theoretical forward progress as presented in Section 3.1. The energy storage in the model was configured based on an empirical capacitor leakage model (with detail presented in Section 3.2.1.1). An IPS based on Hibernus [37] was implemented and its load parameters were profiled so as to configure the load model (with detail presented in Section 3.2.1.2). With the configured model, a range of the energy storage size (C, supply current (I_{in}), and volatile state size (equivalent to restore time T_{r} in Hibernus-like IPSs) were swept with a fine granularity and the corresponding forward progress was generated, so as to investigate their relationship against forward progress (T_{exe}). Finally, the actual forward progress was measured on the implemented IPS with a smaller set of C and I_{in} for validation of modelled results.

3.2.1 Model Configuration

The model was configured with an empirical capacitor model and experimentally measured load characteristics so as to approximate a real IPS platform.

---

[1] `https://git.soton.ac.uk/energy-driven/energy-storage-sizing`

3.2.1.1 Energy Storage

The energy storage is represented as an ideal capacitor with empirically defined leakage current. Its terminal voltage is directly applied to the load, so is modelled as:

$$C\frac{dV_{\text{cc}}}{dt} = I_{\text{harv}} - I_{\text{load}} - I_{\text{leak}} \tag{3.11}$$

where I_{load} is the current consumption of the load. In this exploration, we refer to the empirical I_{leak} of AVX TAJ low-profile series tantalum capacitors [109], which depends on capacitance C, rated voltage V_{rated}, and terminal voltage V_{cc} [108]:

$$I_{\text{leak}} = 0.01\lambda C V_{\text{rated}} \quad (A) \tag{3.12}$$

where λ denotes the ratio of the actual current leakage at V_{cc} to the current leakage at V_{rated}, and λ is approximated as:

$$\lambda = 0.05 \times 20^{\frac{V_{\text{cc}}}{V_{\text{rated}}}} \tag{3.13}$$

We assume a typical load of $< 4.0\,\text{V}$ so, to minimise leakage, we select a device with $V_{\text{rated}} = 10\,\text{V}$ so as to operate between 25-40% of its rated voltage [108].

3.2.1.2 Intermittent Computing Load

We configured the load with experimentally measured current draws and time overheads. We only consider computational loads in this study, as handling of peripherals in intermittent systems is still an ongoing research topic [100, 110].

We implemented and parameterised a reactive IPS [37] on a TI MSP430FR6989-based development board. The load parameters were profiled with the MCU running a Dijkstra path finding algorithm with 1696 B RAM usage at 8 MHz. The supply voltage monitoring circuits use the MCU's internal comparator and an external 3 MΩ voltage divider. The restore and save voltage thresholds are set as V_{r} = 2.4 V and V_{s} = 2.1 V respectively. The MCU shutdown voltage V_{off} is 1.8 V.

The measured current draws and time overheads are listed in Table 3.2. The current draw was profiled with experimental measurements at a range of supply voltages. The variation of I_{lpm} between V_{off} (1.8 V) and V_{r} (2.4 V) is 2%, and for I_{exe} between V_{s} (2.1 V) and 3.3 V is 1.5%. I_{exe} also has a run-time variation of 2.8% due to a variable memory access rate. As these variations of I_{exe} and I_{lpm} in their effecting voltage range are minor, we therefore omit the variations and use the mean of I_{exe} and I_{lpm} in the model. I_{r} and I_{s} are measured at V_{r} and V_{s} respectively.

Parameter	Value
I_{exe}	887 µA
I_{lpm}	26 µA
I_{r}	971 µA
I_{s}	811 µA
T_{r}	1.903 ms
T_{s}	1.880 ms

TABLE 3.2: Profiled MCU parameters.

Given the voltage thresholds and the current consumption, the minimum energy storage capacitance is 6.2 µF. This guarantees that a save and restore operation can complete even if the incoming supply current drops instantaneously to zero. Moreover, the model parameters in Table 3.2 are given as an example, and can be changed for different load characteristics. For example, T_{r} and T_{s} can be tuned for different volatile state sizes.

3.3 Exploration of Energy Storage Sizing

In this section, we present an exploration of the relationship between α_{exe} and C with respect to supply current I_{harv} and volatile state size.

3.3.1 Impact of Supply Current

Increasing energy storage capacitance beyond the minimum one can improve forward progress by reducing the frequency of power interruptions, but this improvement may be offset by increased leakage. Figure 3.4 shows the relationship between forward progress and energy storage capacitance for a range of constant supply currents. In this section, we denote the capacitance that leads to the maximum forward progress α_{exe} as C_{α_max}. C_{α_max} for each current value is also shown in Figure 3.4.

The minimum capacitance (dashed line in Figure 3.4) is calculated to deliver correct operation even if the supply current instantaneously drops to zero. If it does not drop to zero, this means that correct operation could have continued even with a smaller capacitance given that the current supply keeps providing energy during execution, though designing a system in this way would be inadvisable owing to unpredictability of the supply. This property is illustrated in Figure 3.4, in the area on the left of the dashed line. It may be observed that, for each of the current values, there is a sudden drop-off towards zero forward progress. This illustrates the hazard of setting the capacitance too small: the stored energy is too low to allow a restore and save to be undertaken. The reason for this step change rather than a continuous change is that the

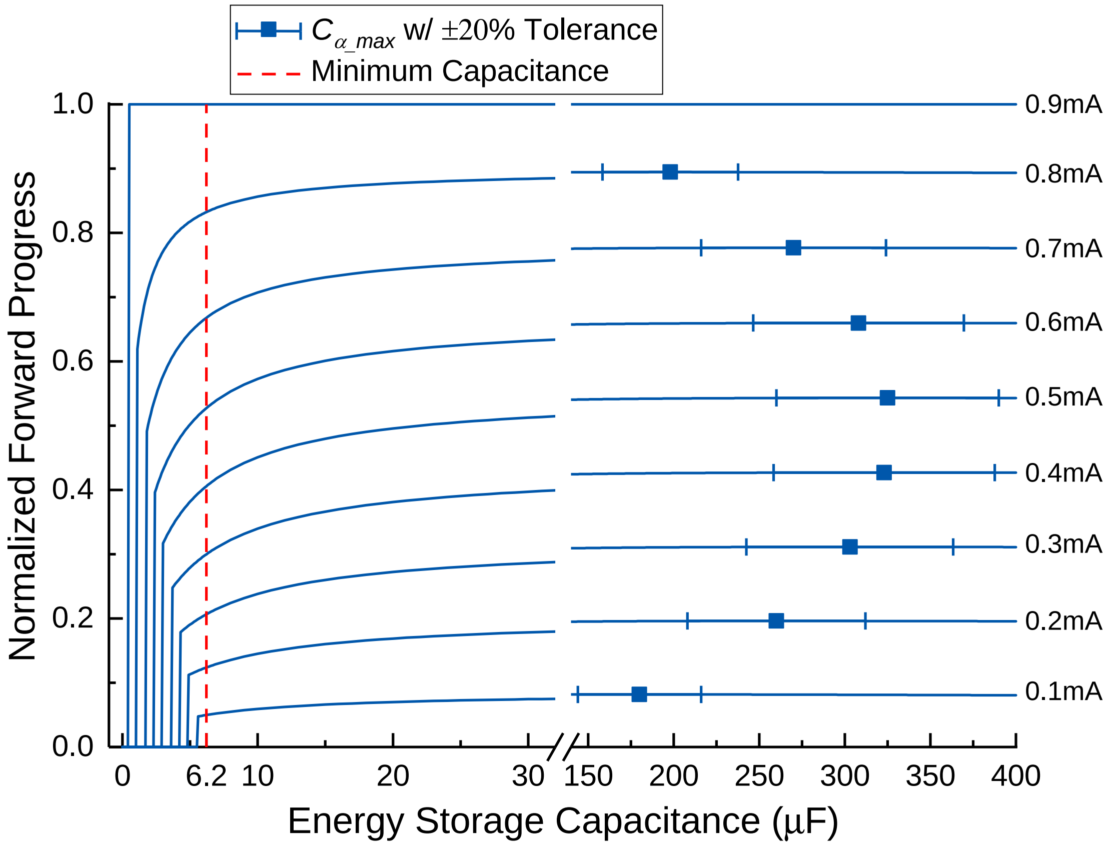


FIGURE 3.4: Forward progress against energy storage capacitance at different levels of constant supply current. Error bars around C_{α_max} denote the impact of typical ±20% capacitance tolerance. The 30–150 µF range is omitted as forward progress in that range increases monotonically.

implemented IPS only performs one restore operation in the first execution cycle after a reboot and enters a low-power mode with volatile state retained after a save operation. Hence the energy used for restoring state is then used for effective execution in the following operating cycles as long as the supply voltage recovers to the restore threshold without a power interruption.

Typically, commercially-available capacitors have a ±20% tolerance. The effect of this variation on maximum forward progress is shown to be negligible ($< 0.23\%$) in Figure 3.4. However, it must be pointed out that the effect would be much more pronounced if operating at the minimum capacitance as the variation of forward progress is larger with smaller capacitance values. Thus, it is recommended that a tolerance is considered when designing IPSs with minimum capacitance.

Figure 3.5 shows that an improvement in forward progress of up to 65% can be achieved when using C_{α_max} instead of the minimum. However, it may not be desirable to set the capacitance solely for maximising forward progress, because there are often trade-offs with other factors including increased interruption periods and dimensions (later explored in Section 4.2.3). In real-world energy source conditions, the supply current

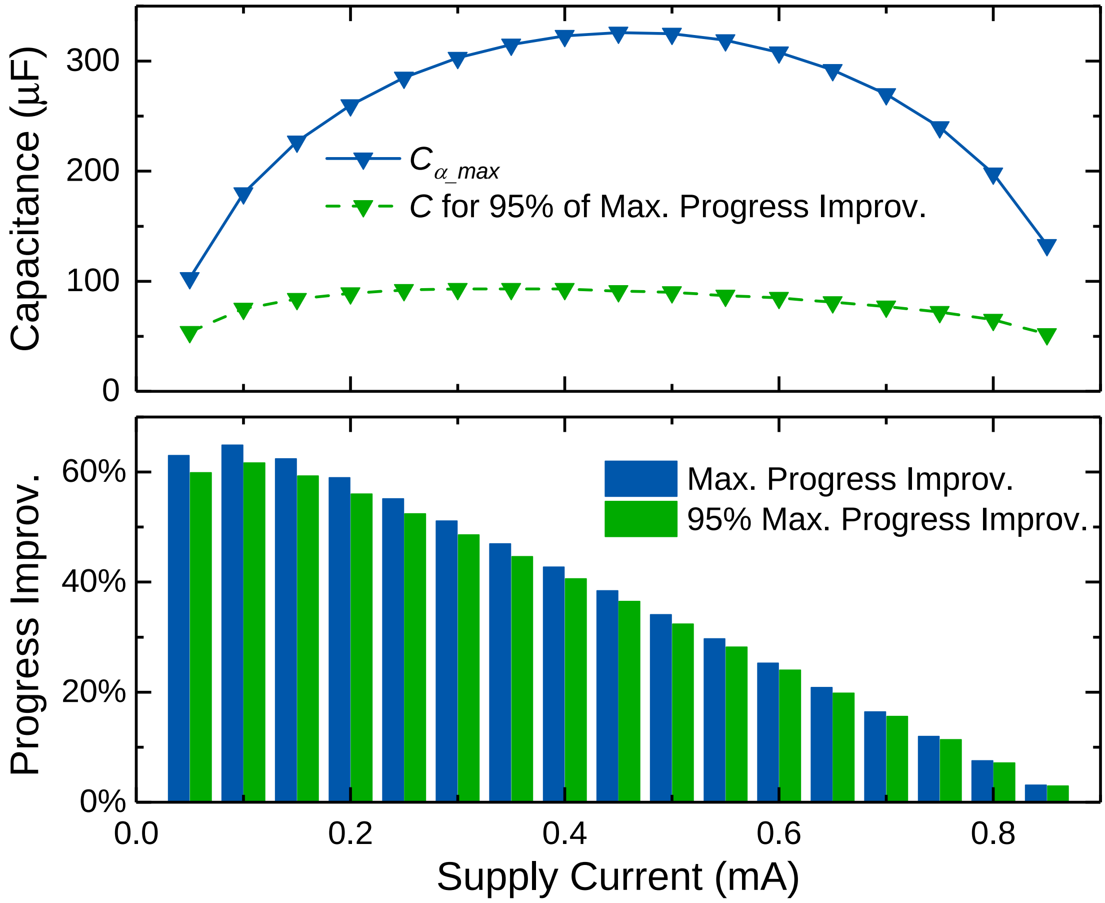


FIGURE 3.5: Maximum forward progress improvement by sizing energy storage given a spectrum of supply current (normalised by the minimum capacitance case), with the corresponding capacitance for maximum and sub-maximum (95% of maximum) forward progress improvement.

varies across this spectrum, and hence leads to an overall progress improvement based on its supply distribution. This improvement exists only when the device operates in the *Intermittent* mode, since the device keeps either inactive in the *Off* mode or active in the *On* mode without the need for restoring and saving state. Correspondingly, C_{α_max} is also plotted against supply current. While a large improvement can be delivered with C_{α_max}, as shown in Figure 3.5, 95% of this gain can still be obtained with significantly smaller capacitances (mean 31% of C_{α_max}). For example, reducing from 325 µF to 90 µF gives 95% of the maximum improvement with a 0.5 mA supply.

3.3.2 Impact of Volatile State Size

The size of volatile state differs across applications with different amounts of RAM usage, and hence incurs varying time and energy overheads for restore and save operations. We measured time overheads of restore and save operations in the minimum case (64B register data and a 160B stack) and the maximum case (64B register data and a full 2048B RAM) respectively as shown in Table 3.3. As these time overheads are

State Size (Registers + SRAM)	**Restore Time**	**Save Time**
64B + 160B (lower bound)	232 µs	208 µs
64B + 2048B (upper bound)	2.298 ms	2.274 ms

TABLE 3.3: Linear scaling range of volatile state size and restore/save time overheads.

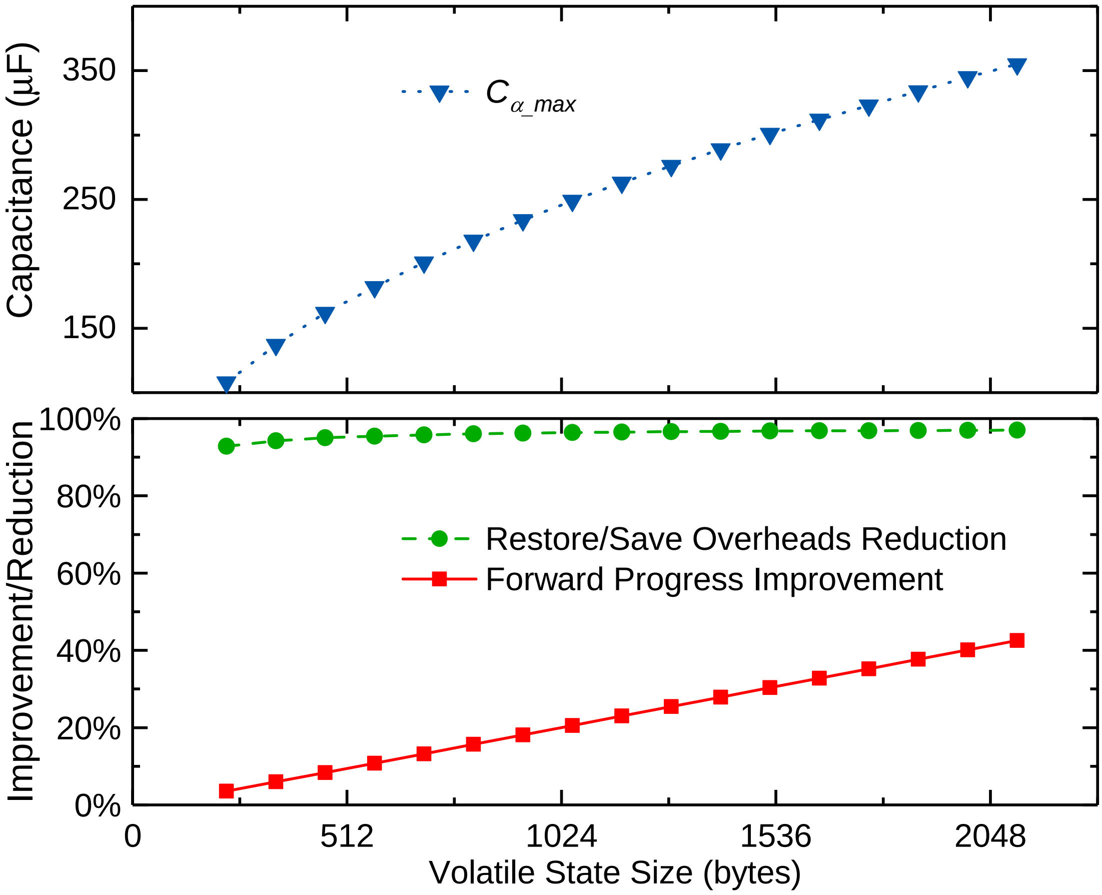


FIGURE 3.6: Impact of RAM usage (linear to restore/save overheads) on sizing energy storage with 0.4 mA current supply. Improvement and reduction are normalised by the minimum capacitance case.

expected to be linear to the state size [45], the model can be tuned for various volatile state sizes by linearly scaling the profiled values.

An example of this is plotted in Figure 3.6. The forward progress improvement by sizing energy storage increases with the volatile state size, and C_{α_max} grows accordingly. The improvement becomes insignificant when the volatile state size is small because the restore and save overheads are already negligible. For example, when the workload uses the least volatile state (the leftmost point), the maximum progress improvement is only 3.6% although the restore and save overheads are reduced by 93%.

Where the size of the volatile state may vary at run time, a different capacitor size within the range 108–355 µF may have been recommended (Figure 3.6). However, as can be seen from Figure 3.4, there is a minimal difference in forward progress across this range. In the worst case, a 2.7% reduction results from setting C_{α_max} for the minimum state size, while running with the largest state size.

3.4 Experimental Validation

This section compares experimental forward progress with the modelled one to validate the proposed reactive IPS model and the presented energy storage sizing effects.

3.4.1 Experiment Setup

We validated our model on the IPS system that we parameterised for exploration (Section 3.3), so its IPS method, voltage thresholds, current draws, and workload are as mentioned. The on-board decoupling capacitance was measured as 10.0 µF, and hence was the minimum capacitance that could be tested. Further capacitance was added to provide extra energy storage up to a maximum of 43 µF, as forward progress with this capacitance can approximate $\alpha_{\mathrm{exe\_ideal}}$ (an upper bound), which is linear to supply current I_{in} when $I_{\mathrm{lpm}} < I_{\mathrm{in}} < I_{\mathrm{exe}}$ (mentioned in Section 3.1.2).

In practical IPSs, the effective execution time ratio (the metric of forward progress in this chapter) that contributes to the effective output is not easy to differentiate from the execution time that is wasted. Therefore, in the experiment, the task completion rate, i.e. the number of tasks completed per second, was measured because it directly reflects forward progress and closely relates to the effective execution time ratio. A task completion was indicated by a positive output pulse from the MCU at the end of a task. The task completion rate was then obtained by giving an observation window in the oscilloscope that covers a number of output pulses and measuring the frequency of these pulses. The task complete rate in the *Intermittent* mode was divided by the one in the *On* mode to obtain the experimental normalised forward progress.

3.4.2 Model Validation

To validate the accuracy of our model, we powered the device with a range of supply currents (0–0.9 mA) to operate the device in *Intermittent* mode, and repeated the tests with three energy storage capacities: a) 10.0 µF decoupling capacitance; b) 21.5 µF (11.5 µF added); c) 43.0 µF (33.0 µF added). We compared the actual forward progress against predictions generated from our model. As shown in Figure 3.7, the model-generated forward progress matches closely with the experimental results with only 0.5% mean absolute percentage error across all the results. Hence, the proposed model is able to accurately estimate forward progress for design exploration.

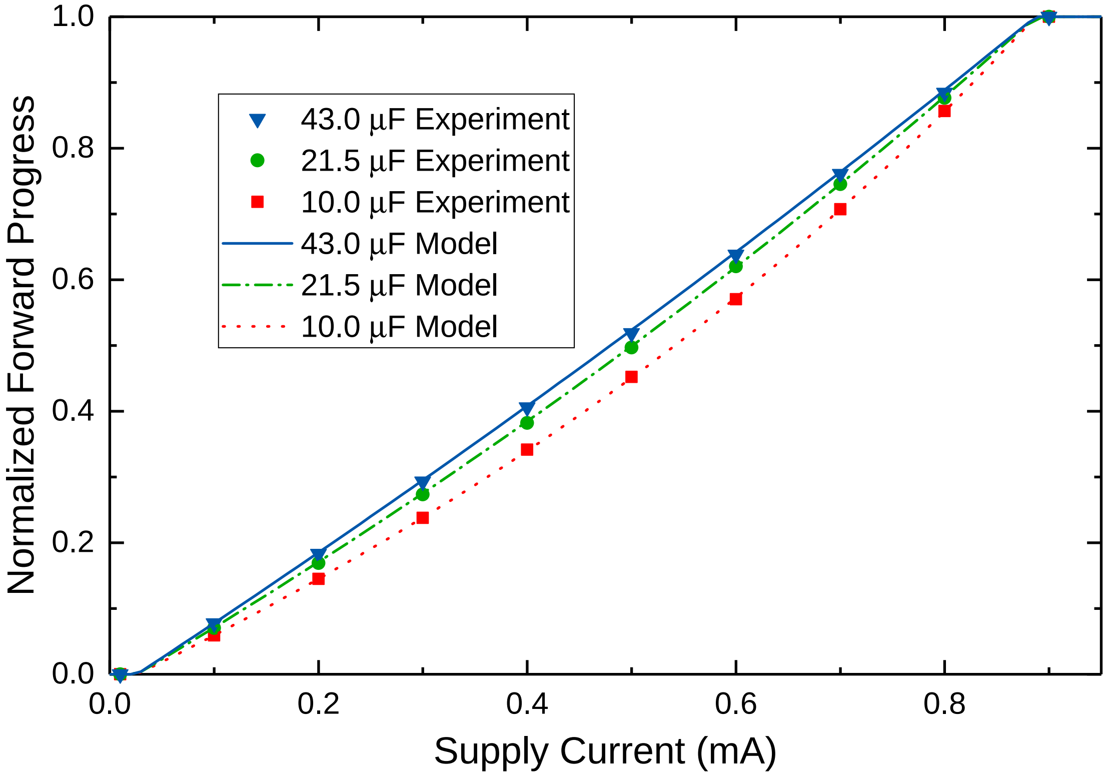


FIGURE 3.7: Model validation with experimental and modelled forward progress.

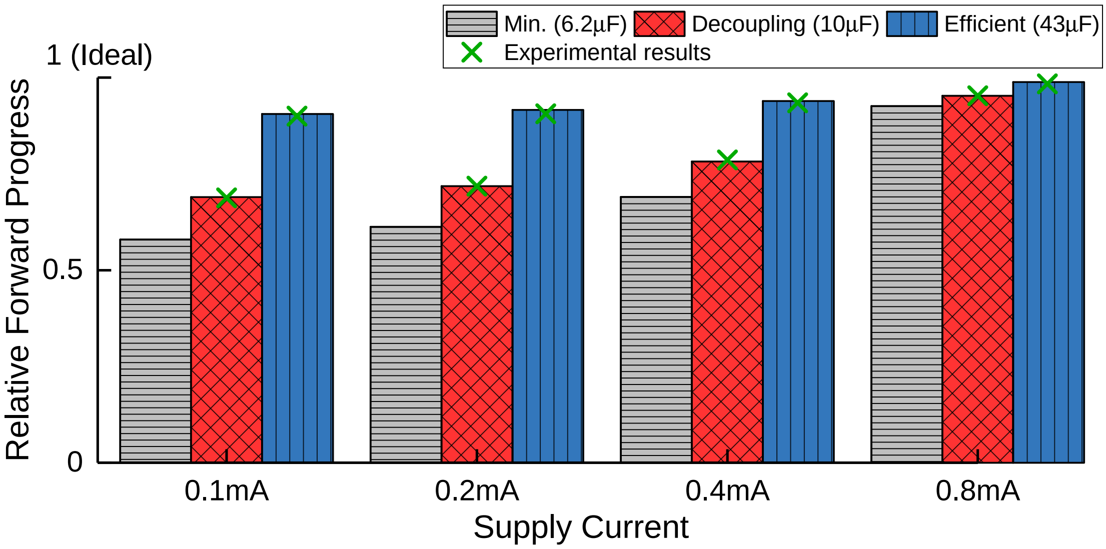


FIGURE 3.8: The relationship between energy storage capacitance and IPS forward progress, for various supply currents.

3.4.3 Validation of Sizing Effects

As shown with modelled and experimental results in Figure 3.8, the efficiently-sized energy storage capacitance (43 µF) improves forward progress by up to 55% and 30% compared against using the minimum and decoupling capacitance respectively. We notice that this improvement becomes significant when the supply current attenuates because the save and restore overheads consume a larger proportion of the available energy. Also, the efficiently-sized capacitor achieves at least 90% of the ideal forward progress across the tested supply currents. The ideal case only switches between LPM and execution, without restoring and saving overheads (explained in Section 3.1.2). These results illustrate the importance of this technique, in particular for conditions where the supply current is low.

3.5 Summary

While conventional IPSs have used minimal levels of capacitance, this chapter explored the energy storage sizing effect on IPS's forward progress. A model of reactive IPSs was proposed to facilitate the understanding and design exploration of IPSs, explaining a mathematical relationship between forward progress and system parameters, e.g. energy storage capacitance and supply current. We then configured the proposed model with an experimentally-profiled IPS platform, and utilised it to explore the energy storage sizing effect. The exploration results showed that sizing energy storage can improve forward progress by up to 65%. We also found that this improvement becomes significant when the supply current attenuates, which implies the importance of sizing energy storage in energy harvesting conditions where the supply current is low. The model was experimentally validated across a range of supply current and energy storage capacitance, showing a mean absolute percentage error of 0.5%. We conclude that adding a relatively small amount of energy storage can significantly improve forward progress.

The proposed model has demonstrated its potential for design exploration of IPSs. In the next chapter, we will incorporate the model in a simulation framework that recommends an appropriate energy storage size in real-world energy conditions.

Chapter 4

Energy Storage Sizing Approach for Deploying IPSs

Having analysed the sizing effect of energy storage on IPS performance, this chapter focusses on the energy storage sizing effect when deploying IPSs under real-world energy conditions, and providing an approach that recommends an energy storage size.

As presented in Chapter 3, providing more energy storage than the minimum can improve forward progress. However, it was also revealed in Section 3.3.1 that to obtain the maximum forward progress improvement with $C_{\alpha\_\max}$ can take 3.2× capacitance compared to the one that achieves 95% of the maximum improvement. A larger capacitor typically occupies more physical space, which may contradict with some IPS applications that require miniaturised size, e.g. implantable medical devices [111]. Also, a larger capacitor takes longer to recharge, hence prolonging the period of power interruption and compromising reactivity. A systematic method is in need to decide the energy storage size for deploying IPSs, considering a trade-off of multiple design factors.

However, as reviewed in Section 2.4.5, current tools for IPSs are not practical for fast estimation of forward progress in a long-term deployment, and lack a method of sizing energy storage to improve forward progress while moderating the physical size and interruption periods. Coarse-grained models cannot react to frequent power interruptions and accurately estimate forward progress, while fine-grained models become inefficient when processing long-term data and iterating for multiple system configurations. Hardware emulators, though reliable for repeatable experiments, are limited by hardware options and also impractical for long-term and iterative tests.

To address the above problem, this chapter presents an approach for sizing energy storage in IPSs, quantifying and trading off forward progress, capacitor volume, and interruption periods. With the model in Section 3 integrated, the proposed sizing approach is able to fast explore the relationship between the energy storage size and forward progress with long-term real-world energy conditions. An example cost function is also provided to trade off multiple design factors so as to recommend an energy storage size. The main contributions in this chapter are as follows:

- A model-based sizing approach that recommends appropriate energy storage capacitance in IPSs (Section 4.1).
- An evaluation of the impact of sizing in real-world conditions using real energy availability data (Section 4.2). This includes a cost function-based method for trading off parameters. In an example, this reduced capacitor volume and interruption periods by 83% and 91% respectively, while sacrificing 7% of forward progress, compared to solely maximising forward progress.

The associated simulation tool, coded in C, is available open-source[1].

The remainder of this chapter is organised as follows. Section 4.1 proposes an approach for sizing energy storage in IPS deployment Section 4.2 configures and demonstrates the proposed sizing approach with real-world energy source data, with forward progress, capacitor volume, and interruption periods being estimated and traded off. Finally, Section 4.3 summarises this chapter.

4.1 Energy Storage Sizing Approach

As mentioned, previous IPS designs typically adopt a minimised capacitor size so as to minimise device dimensions and interruption periods, but this can also reduce forward progress. An appropriate capacitor size instead may balance the three factors.

We propose a sizing approach which recommends appropriate energy storage capacitance for an IPS, trading off forward progress against capacitor volume and interruption periods. We present a system model which accepts real long-term data on environmental energy conditions. The three inputs can be swept for design exploration, but we focus on energy storage in this chapter. The model outputs forward progress, capacitor volume, and interruption periods (defined in Section 4.2.2). These are subsequently traded off in a cost function to obtain the appropriate energy storage capacitance. This process is summarised Figure 4.1 with details explained as follows.

4.1.1 Input

A time trace of representative environmental energy conditions in the intended deployment location is provided as an input, along with the energy harvester size. For design

---

[1] `https://git.soton.ac.uk/energy-driven/energy-storage-sizing`

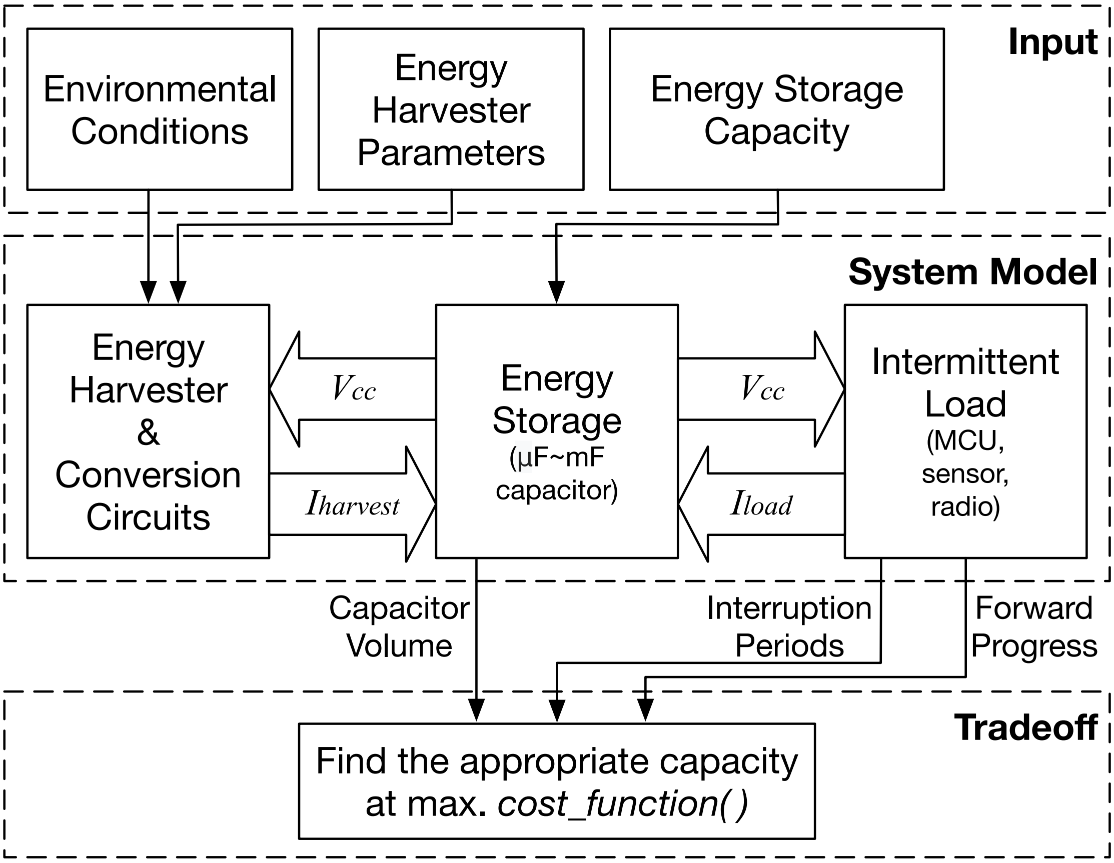

Input
Environmental Conditions
Energy Harvester Parameters
Energy Storage Capacity
System Model
Energy Harvester & Conversion Circuits
V_{cc}
$I_{harvest}$
Energy Storage (µF~mF capacitor)
V_{cc}
I_{load}
Intermittent Load (MCU, sensor, radio)
Capacitor Volume
Interruption Periods
Forward Progress
Tradeoff
Find the appropriate capacity at max. cost_function( )

FIGURE 4.1: Structure of the proposed system model and sizing approach.

exploration, assuming the energy source is equally distributed in the deployed space, these can optionally be changed to explore variations and scales of harvested power. A pre-defined set of energy storage capacitance values are swept through.

4.1.2 System Model

This contains three modules, i.e. *Energy Harvester and Conversion Circuits*, *Energy Storage*, and *Intermittent Load*. The three modules communicate by their voltage and current flows. The current production I_{harv} and consumption I_{load} are buffered in the energy storage, which then calculates V_{cc} for the load and the harvester output. Due to the variety in each module, they should be individually specified to represent the target platform according to the techniques implemented.

- *Energy Harvester and Conversion Circuits:* The energy harvester module transduces environmental energy into electricity. The harvested power is typically conditioned to provide a suitable voltage for charging the energy storage and supplying the load efficiently. In IPSs, conversion circuits may simply be a diode to inhibit backflow of current. The energy harvester and conversion circuits can be modelled together as a module because they are usually coupled or integrated.

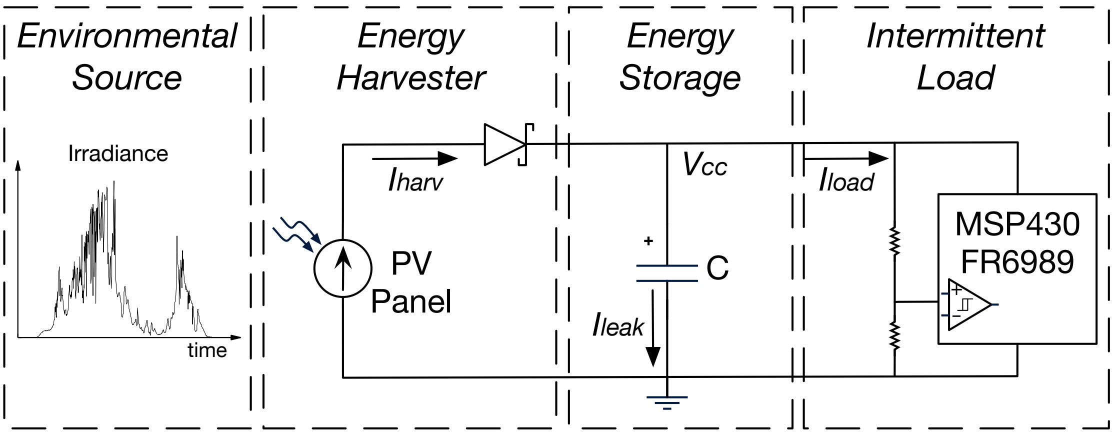


FIGURE 4.2: System model of a PV-based IPS.

- *Energy Storage:* Energy storage in IPSs is usually in the form of a µF- to mF-scale capacitor. It must be sufficient to complete the most energy-expensive atomic operation, and may be formed only of the decoupling capacitor(s). This also includes an empirical model relating capacitance to capacitor volume (discussed in Section 4.2.3).
- *Intermittent Load:* It includes all the power consumers in an IPS, such as a microcontroller, sensors, and a radio. This module outputs forward progress and interruption periods using the model presented in Section 3.1.

4.1.3 Trade-off

The appropriate capacitance is then found through a cost function (an example of which is presented in Section 4.2.3). This may trade-off forward progress against capacitor volume and interruption periods.

4.2 Sizing under Real-World Energy Conditions

In this section, we model an IPS with a PV energy harvester to explore the energy storage sizing effect in real-world energy conditions, and demonstrate use of the proposed sizing approach.

4.2.1 Simulation Configuration

We integrate the validated reactive IPS model into a system model with a PV energy-harvesting supply as shown in Figure 4.2. The energy storage model and the intermittent load model are as presented in Section 3.3.

Parameter	Value
Open-Circuit Voltage	0.89 V/cell
Short-Circuit Current	$14.8\,\mathrm{mA/cm^2}$
Maximum Power Voltage	0.65 V/cell
Maximum Power Current	$12.1\,\mathrm{mA/cm^2}$

TABLE 4.1: PV cell properties under a $1000\,\mathrm{W/cm^2}$, AM-1.5 light source.

We use a converter-less supply circuit where only a Schottky diode is connected to the energy harvester output in order to prevent current backflow. The energy source conditions are imported from NREL outdoor solar irradiance data [112] and EnHANTs indoor irradiance data [113]. Four sets of light conditions are used to encompass different energy environments. To convert irradiance into harvested power, we adopt a PV cell model [114] which uses the parameters available in common datasheets, so it can easily be reconfigured to suit various devices. The output current I_o of the PV cell model can then be described as:

$$I_\mathrm{o} = \frac{G}{G_\mathrm{ref}} I_\mathrm{sc} \left(1 - \left(1 - \frac{I_\mathrm{mpp}}{I_\mathrm{sc}}\right)^{\frac{V_\mathrm{O} - V_\mathrm{oc}}{V_\mathrm{mpp} - V_\mathrm{oc}}}\right) \tag{4.1}$$

where V_o is the output voltage of the PV cell, G is the ambient irradiance, G_ref is the reference irradiance (normally $1000\,\mathrm{W/cm^2}$), and I_sc, V_oc, I_mpp, V_mpp are respectively short-circuit current, open-circuit voltage, and the current and voltage at the MPP given the reference irradiance. V_o and G are dynamic at run time, while other parameters in this model are constant. We refer to Panasonic Amorton glass type solar cells [115] for PV cell properties as shown in Table 4.1. We set four cells in series (with V_oc = 3.56V) to match the operating voltage of the MCU (maximum 3.6V), and model energy harvester sizing by scaling the cell area.

Our simulation tool can perform two simulation processes: (a) sort and process the time distribution of environmental conditions, and (b) simulate system state chronologically with a fine-grained time step. Process (a) reduces simulation time significantly, e.g. from hours to seconds, but ignores the restore operation after a brownout reset, hence overestimating forward progress, and it overestimates more with smaller capacitance and lower supply current. In the following results, Figure 4.3 comes from Process (a) for fast exploration, and Figure 4.5 and Figure 4.7 come from Process (b) for accurate records of interruption periods.

4.2.2 Exploration with Real-World Energy Source Conditions

In real-world deployments, ambient energy source conditions are dependent on time and location. The energy harvester and storage need to be sized to achieve the desired forward progress across the range of expected conditions.

4.2.2.1 Sizing the Energy Harvester

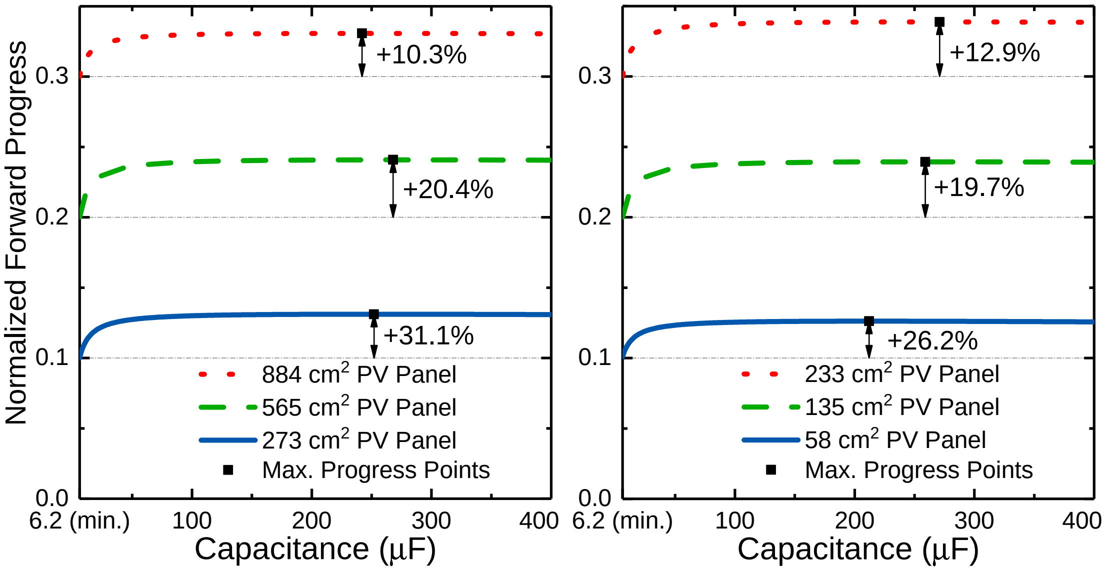
For the purposes of this exploration, three levels of baseline mean forward progress (α_{exe}) are set as 0.1, 0.2, and 0.3. We use the system model to find the PV panel area that achieves the expected forward progress under the different energy source conditions with minimum energy storage. We scale the PV panel area to find that which achieves each baseline α_{exe}. As shown in Figure 4.3, the energy harvester sizes that achieve the desired α_{exe} may span orders of magnitude given different energy source conditions from mm^2 for outdoor sources ((c) and (d)) to cm^2 for indoor sources ((a) and (b)).

4.2.2.2 Sizing the Energy Storage

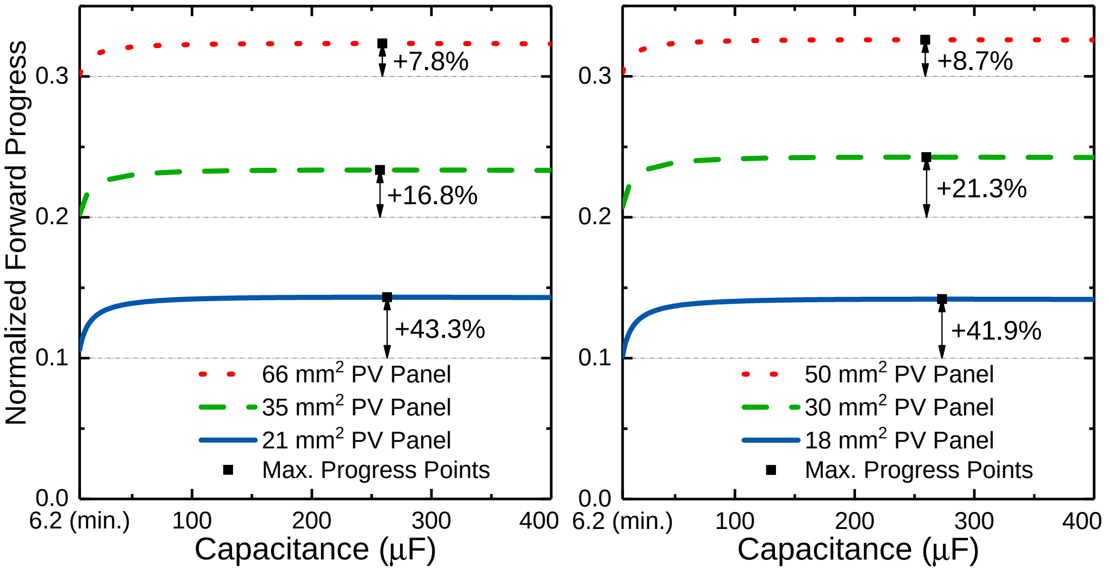
Having obtained the energy harvester sizes for the baseline forward progress, we then use the modelling approach to size energy storage. We analyse the sizing effect of energy storage on forward progress given real-world energy conditions. Figure 4.3 shows a 7.8-43.3% improvement in forward progress by sizing energy storage under the given real-world energy conditions and baseline energy harvester sizes. It can also be inferred that optimising energy storage can either improve forward progress for a given energy harvester size, or reduce the energy harvester size that achieves the target forward progress. Given higher-power energy sources (e.g. Denver 2018 and Hawaii 2018 outdoor solar), increasing the harvester size efficiently improves forward progress with minor dimensional overheads, e.g. tens of mm^2; however, given lower-power sources (e.g. EnHANTs Setup A and Setup D indoor light), optimising energy storage capacitance can save tens of cm^2 of PV panel area to achieve the same forward progress.

Also, the progress improvement by sizing energy storage varies accordingly with energy source conditions. As mentioned in Chapter 3, this improvement stems from the reduction of restore and save overheads when the supply current is low and the device work in the *Intermittent* mode. Thus, the results of EnHANTs Setup A and Setup D show a higher progress improvement from sizing energy storage than those of Denver 2018 and Hawaii 2018.

The mean forward progress given target $\alpha_{\mathrm{exe}} = 0.1$ is plotted in Figure 4.4, with the 60th and 90th time percentiles of forward progress. In all the above datasets, the energy source is absent and the system is off for around 55 % of time, so we plot the percentiles from the 60th. The mean progress during the energy-available periods is averaged over the energy-absent periods, so the actual mean forward progress during the energy-available periods is nearly double the annual mean.



(A) EnHANTs Setup A (B) EnHANTs Setup D

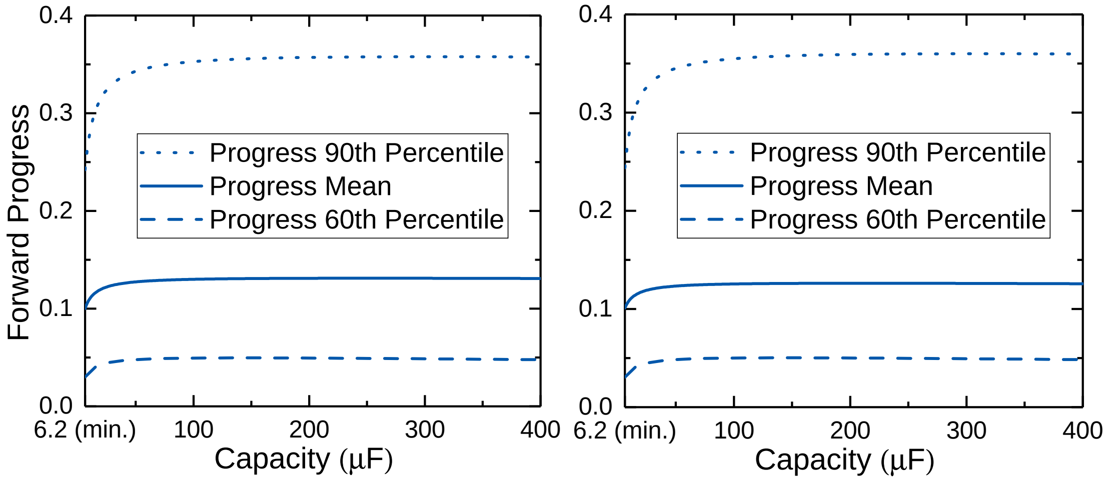


(C) NREL Denver 2018 (D) NREL Hawaii 2018

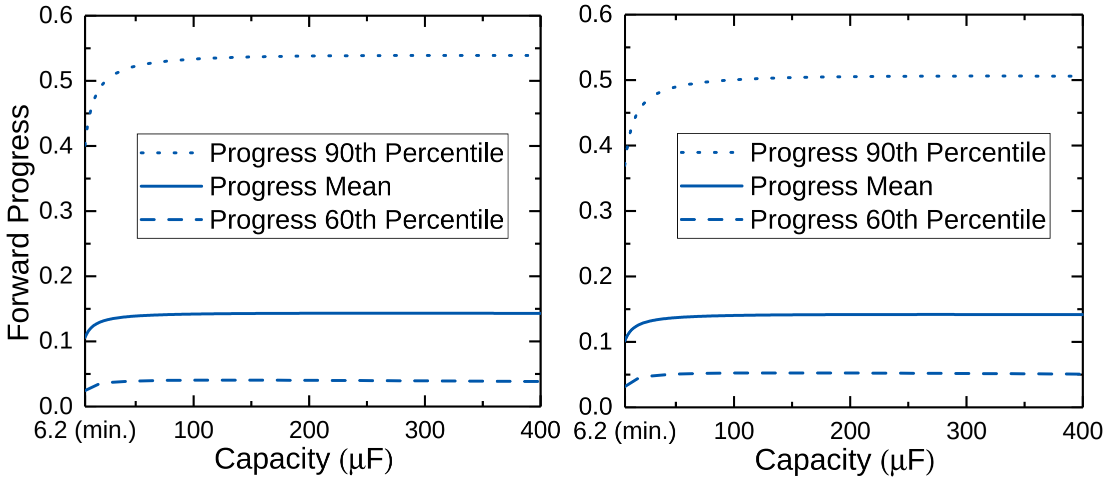
FIGURE 4.3: Improvement of average forward progress by sizing energy storage given different PV panel areas under real-world energy source conditions. The model is able to find the PV panel area required for achieving the target mean forward progress.

4.2.2.3 Interruption Period

Besides forward progress, we also explore how the capacitance can change the interruption periods. When interrupted by insufficient power supply, an IPS enters an interruption period where it saves its volatile state, waits for supply voltage to recover, and restores the state to resume execution, without making any forward progress. Applications that require frequent sensing may be negatively affected by long interruption periods. We measure an interruption period as *the period between two successive execution periods*, e.g. a consecutive 'SLR' period in Figure 3.2 forms an interruption period.



(A) EnHANTs Setup A (B) EnHANTs Setup D



(C) NREL Denver 2018 (D) NREL Hawaii 2018

FIGURE 4.4: Time percentiles of forward progress by sizing energy storage with target $\alpha_{\mathrm{exe}} = 0.1$ and the corresponding PV panel area listed in Figure 4.3. The percentiles start from the 60th as the system is off for around 55 % of time due to insufficient energy source.

We record all the interruption periods during a one-year simulation with 10–50 µF capacitors, the Denver 2018 dataset, and an 80 mm^2 PV panel. Figure 4.5 presents the distribution of all the interruption periods. With increased energy storage, the interruption period is prolonged. For example, the 90th percentile of interruption periods increases from 32.2 ms at 10 µF to 123.4 ms at 50 µF at an approximate rate of 23 ms per 10 µF. Facilitated by the simulator, developers are enabled to estimate whether the distribution of interruption periods meet their application requirement.

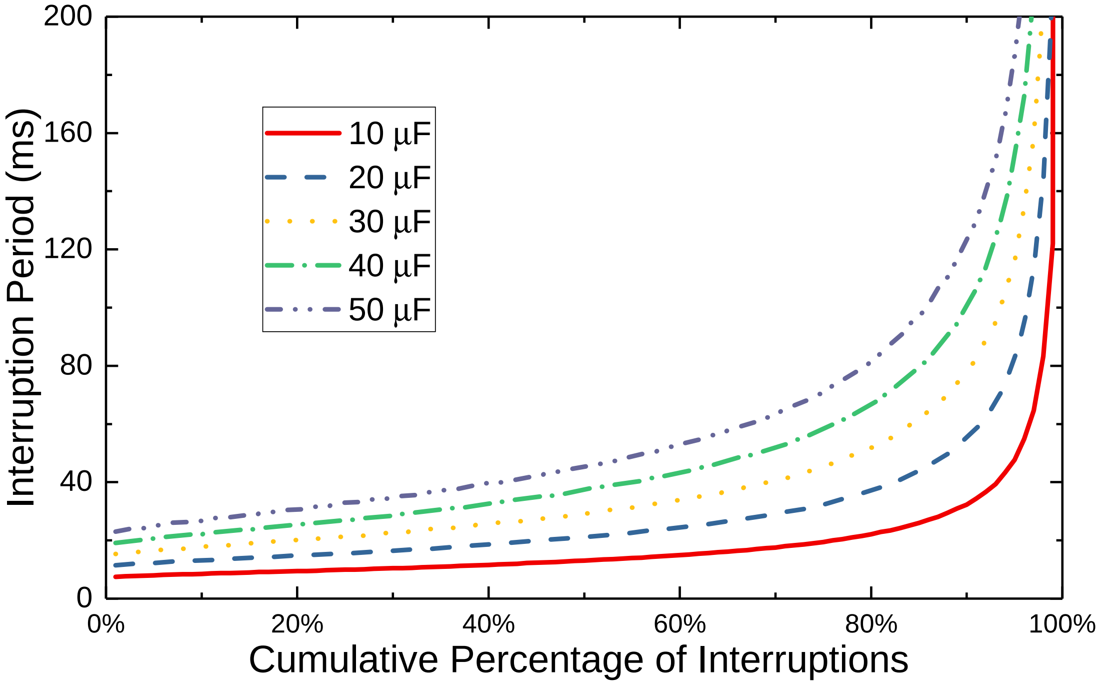


FIGURE 4.5: Distribution of interruption periods.

4.2.3 Trading Forward Progress, Dimensions, and Interruption Period

Although increasing energy storage capacitance improves forward progress, larger capacitance increases both dimensions and interruption periods. We evaluate the overheads of increased capacitor dimensions and interruption periods, and then trade them off against forward progress using a cost function to suggest an optimal capacitance value.

4.2.3.1 Metric of Dimensions

The overhead of capacitor dimensions is evaluated by characteristics of off-the-shelf tantalum capacitors. We narrow down the range of sample capacitors within a set of characteristics: low-profile, 10V rated voltage, and surface-mount package, and select six series of capacitors[2]. The volume and capacitance of these devices are plotted in Figure 4.6. We use the regression of these data to approximate a capacitance-volume relationship.

---

[2] The series of capacitor considered were: AVX TAJ, AVX TACmicrochip, AVX F92, Vishay 572D, Vishay 591D, and Vishay 592D.

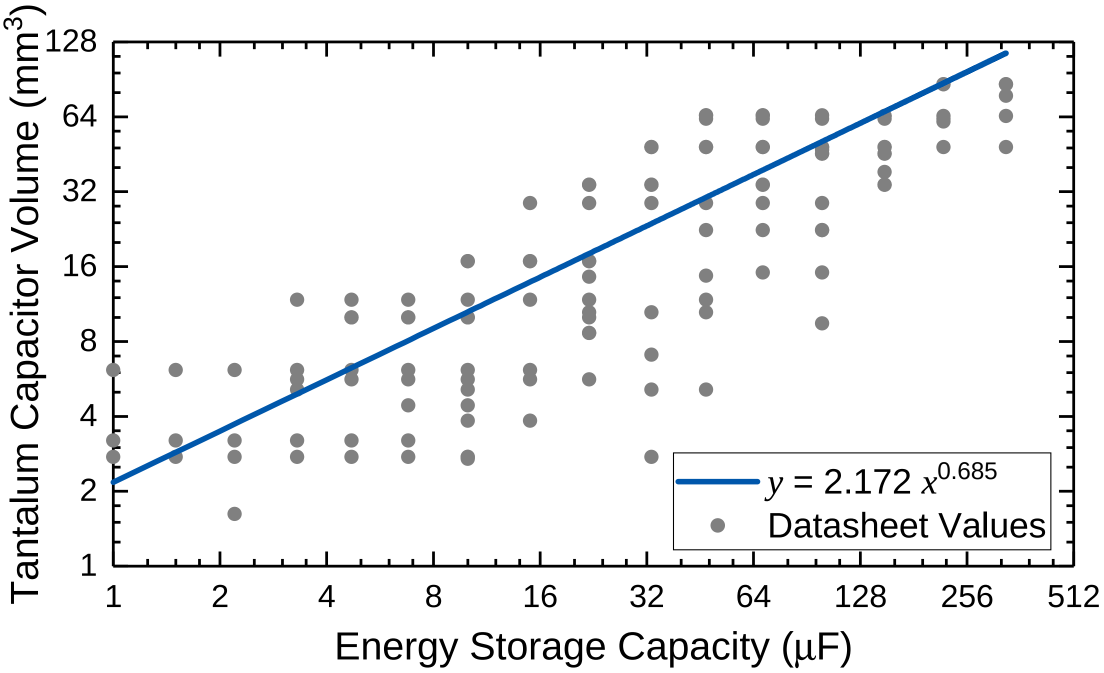


FIGURE 4.6: Tantalum capacitor volume against capacitance for the six series of capacitors analysed.

4.2.3.2 Metric of Interruption Periods

Applications may have various requirements on interruption periods. To demonstrate the usage of our sizing approach, we consider a designer requests the 90th percentile of all interruption periods as an example metric of interruption periods, denoted as T_{int}. This metric indicates 90% of interruption periods are shorter than T_{int}. This metric can be adapted for particular application requirements.

4.2.3.3 Cost Function

From the previous observations (Figure 3.5) we can see that achieving the optimal progress improvement costs much more capacitance (mean 3.2×) than to achieve 95% improvement. A trade-off is necessary to improve forward progress while restricting the overheads of increased capacitor volume and interruption periods. This involves a problem of multi-criteria decision making [116], which is outside the scope of this work. Nevertheless, we provide a cost function in (4.2) as an example to illustrate how these three factors could be traded-off, but designers are expected to customise a cost function with parameters of importance to specific application requirements. Note that the function (4.2) is to be maximised to find the recommended capacitance.

$$f = \frac{\alpha_{\text{exe}}}{k_1} - \left(\frac{v_{\text{cap}}}{k_2}\right)^2 - \left(\frac{T_{\text{int}}}{k_3}\right)^2 \tag{4.2}$$

α_{exe} denotes normalised forward progress, v_{cap} denotes capacitor volume, and T_{int} denotes application interruption periods as mentioned in Section 4.2.3.2. α_{exe}, v_{cap}, and T_{int} can be generated from the simulation tool given C as an input. k_1, k_2, and k_3 are coefficients for normalising each metric, and they are empirically determined according to applications. In this example, the undesirable parameters are expressed as quadratic and negative terms to give an increasing cost to higher values. While only three parameters are considered here, others (such as the energy harvester size) could be included for a system-wise sizing scenario. As an example to demonstrate its usage, we arbitrarily configure the function by setting $k_1 = 0.2$, $k_2 = 200\,\mathrm{mm}^3$, and $k_3 = 500\,\mathrm{ms}$.

4.2.3.4 Results

The effect of the trade-off is plotted in Figure 4.7 using the Denver 2018 energy source dataset. Compared to the capacitor size that solely maximises forward progress, on average, an appropriately-sized capacitor achieves 93% of the maximum forward progress, while saving 83% of capacitor volume and 91% of interruption periods. This also demonstrates the efficacy of the cost function and the chosen coefficients. Compared to the minimum storage case, the appropriately-sized capacitor improves forward progress by 12-124% with energy storage increased from 6.2 µF to 30 µF.

As shown in Figure 4.6, the closest available capacitance that satisfies the 6.2 µF minimum capacitance is 6.8 µF, whereas the closest available capacitance to the appropriate 30 µF is 33 µF. The minimum volumes of 6.8 µF and 33 µF capacitors are both $2.75\,\mathrm{mm}^3$, which means using the appropriate capacitance, instead of the minimum one, may not incur dimensional overhead. The regressed volume of the above two capacitance values are $8.1\,\mathrm{mm}^3$ and $23.8\,\mathrm{mm}^3$ respectively. However, the selection of capacitors can be dependent on factors other than physical volume, such as reliability, operation temperature, and more specific application needs. These factors can also be added into the cost function if necessary.

4.3 Summary

This chapter has presented an approach for sizing energy storage when deploying IPSs, trading off forward progress against capacitor volume and interruption periods. The work includes a simulation tool which is available to download, enabling researchers to experiment with energy storage sizes to optimise IPS designs. The approach was configured and demonstrated with an experimentally-profiled IPS and real-world data of PV sources, showing up to a 43% annual forward progress gain by sizing energy storage. A cost function can be incorporated, allowing various properties of the system to be traded off. The results showed that the suggested energy storage capacitance

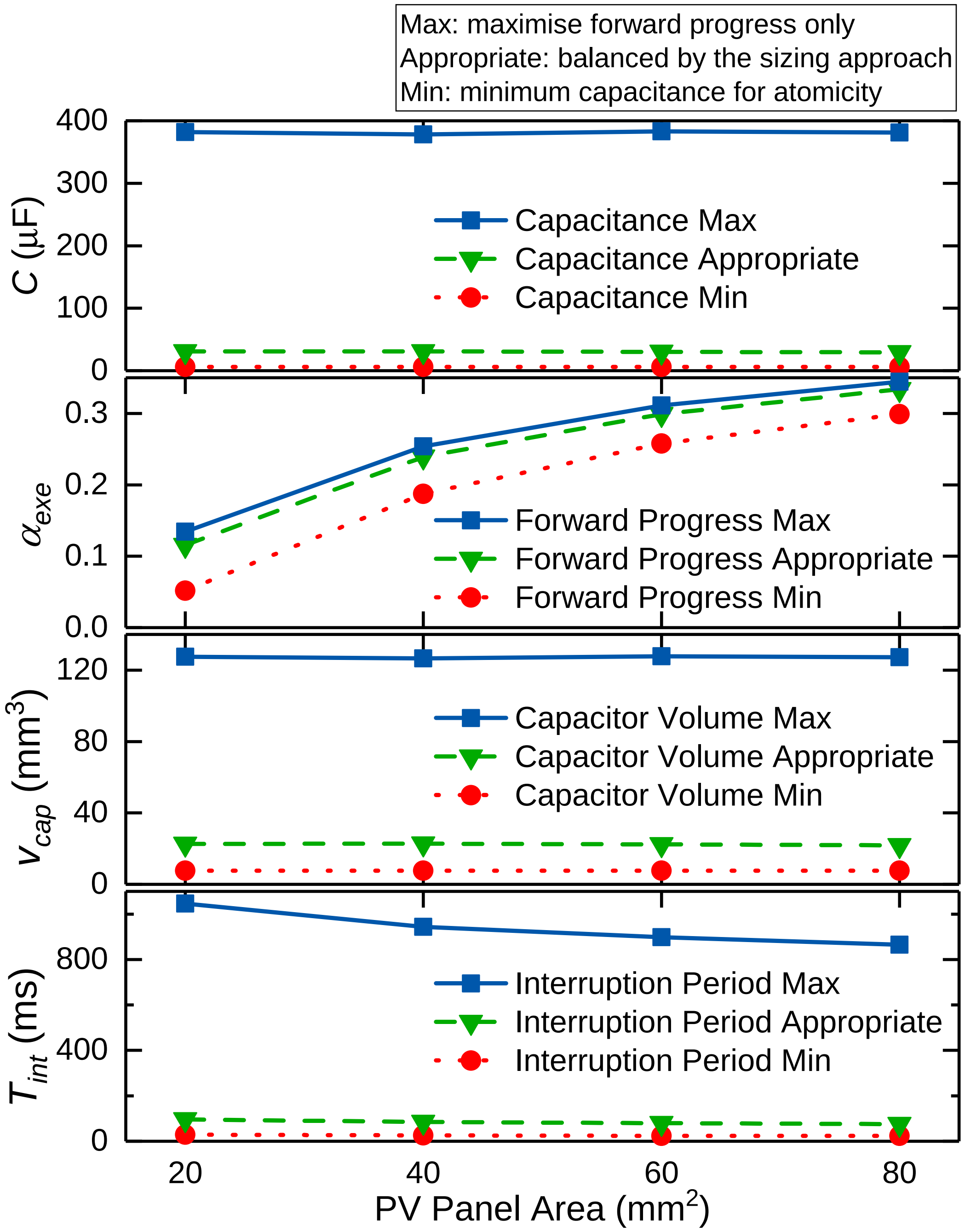


FIGURE 4.7: The sizing approach trades off forward progress, capacitor volume, and interruption periods. The results are plotted against a range of PV panel area, given Denver 2018 energy source dataset.

achieves 93% of the maximum forward progress while saving 83% capacitor volume and 91% interruption periods. Our conclusion is that energy storage should be carefully designed, rather than minimised or indiscriminately picked, to efficiently operate IPSs.

Chapter 5

Runtime Energy Profiling and Adaptation for IPSs

Apart from computational workloads, embedded sensing systems need to utilise peripherals, such as sensors, computational accelerators, and radios, which typically require *atomicity* [117]. In the context of IPSs, an atomic operation should not be checkpointed during execution; if interrupted by power failures, it should restart rather than checkpoint and resume. A peripheral operation is considered atomic because it is usually problematic to checkpoint and restore the operation later, even if the intermediate peripheral state is also checkpointed. For example, checkpointing during a sensor reading and resuming it later can cause incorrect results or an infinite wait as the initialisation is lost, and violate timeliness as the sensor does not render the latest and consecutive results [40]. As presented in Section 2.4, prior works on intermittent peripheral operations either customise a design-time calibrated energy budget for each peripheral operation individually [53], or allocate a universal and large energy budget that ensure the most energy-hungry operation can finish in one active cycle [40].

However, we argue that manually profiling each peripheral operation and customising energy thresholds is impractical due to variability in IPSs, where we have considered the variability in the data amount to process, peripheral configurations, devices, and energy buffering capacitance (detailed in Section 5.1.1). A fixed threshold can be violated if any of the above cases happen, and lead to non-termination[1]. In practical deployment, considering the complexity and labour effort, it is unrealistic to profile every atomic operation for every device under every runtime scenario at design time and customise the energy budgets accordingly.

---

[1] Non-termination happens when the pre-defined energy budget is less than how much the operation consumes and the supply is not strong enough to fill the energy gap. It is one of the main causes for failures in intermittent systems.

On the other hand, using only one high voltage threshold, though probably avoiding non-termination, can affect system energy efficiency. IPSs typically minimise operating voltage in order to lower quiescent power consumption from power conversion loss and system leakage [53]. Also, a high operating voltage can decrease the output current of energy harvesters, making it harder to charge up the buffering capacitor [118]. Hence, setting a high wake-up voltage threshold results in a longer charging time than a linear scale of the voltage threshold, which therefore slows down the system execution or even leave the system in an infinite wait under poor energy conditions.

As reviewed in Section 2.4.4, existing IPS methods profile the energy consumption of atomic peripheral operations at design time to determine a voltage threshold or a capacitor size that avoids non-termination. However, this does not actually guarantee the completion of every atomic operation because energy consumption can change with any runtime conditions different to the profiling setup (demonstrated in Section 5.1.1). Hence, to tolerate dynamic variations, previous IPS designs should usually leave an inefficiently large margin when allocating energy budgets. If this large margin is not given, they can cause either non-termination or high overheads of tracking and restoring state, where DEBS fails, Samoyed undo-logs the NVM data, and RESTOP re-issues peripheral instructions.

To address the above issues, we propose OPTIC[2], which profiles energy consumption of operations at runtime and dynamically adapts energy thresholds based on newly profiled energy consumption and user-defined parameters. A naive approach of runtime energy profiling can be disconnecting the power supply during profiling and taking two readings of supply voltage before and after a task [2], but this can waste the energy harvested during the operation. In contrast, OPTIC's energy profiling operates with the energy harvesting supply connected. The profiling strategy is to measure the input current in the charging cycle so as to estimate the energy budget ΔV_{task}, *the drop of supply voltage caused by an operation without any incoming energy meanwhile*, which directly determines the minimum voltage threshold that safely guarantees the completion of an atomic operation. The runtime profiled energy budget can thus, compared to a design-time profiled one, closely match with the latest energy consumption. Based on the profiling results, OPTIC dynamically adapts the threshold for each atomic operation, with an option of scaling thresholds by user-defined parameters, e.g. a variable data size. Therefore, OPTIC enables IPSs to allocate a barely sufficient energy budget despite runtime energy variations, and hence mitigates non-termination while achieves high energy efficiency, eventually improving the workload throughput.

The main contributions of this chapter can be summarised as follows:

---

[2] OPTIC: Online Energy Profiling and Threshold Adaptation for Intermittent Computers.

1. An exploration of the runtime variations of energy consumption in IPSs that compromise existing approaches in comparison with an adaptive thresholding scheme.
2. A method of runtime energy profiling of tasks for IPSs without disconnecting supply, showing a high accuracy within 5 mV.
3. An adaptive thresholding scheme that, utilising the runtime energy profiling method, dynamically allocates barely sufficient energy budgets for tasks, with an optional scaling based on user-defined parameters. The proposed scheme enables a system to survive with 68% less energy buffering capacitance than the initially allocated amount and presents up to a 98% speedup with variable data sizes, compared to SoA approaches.
4. Implementation of the proposed runtime energy profiling and threshold adaptation method, with an efficient supply voltage monitor.

The rest of this chapter is organised as follows. Runtime energy variations of workloads are explored in Section 5.1, with simulated performance of an adaptive thresholding scheme compared against SoA approaches given the variations. OPTIC's runtime energy profiling method and runtime energy adaption routine are proposed in Section 5.2 and Section 5.3 respectively. An implementation of OPTIC is presented in Section 5.4 Experimental evaluation is shown in Section 5.5. Finally, Section 5.6 summarises the main findings in this chapter.

5.1 Motivation

In this section, we study the variability in IPSs that can violate a predefined fixed threshold. We then investigate how existing approaches fail or become inefficient under this variability, and explore the potential of an adaptive thresholding scheme.

5.1.1 Variability in Intermittent Systems

Design-time profiling of workloads' energy consumption in the prior work can be potentially violated by the variability of IPSs. To study and demonstrate the variability, we chose the built-in AES accelerator on the TI MSP430FR5994 MCU as an example peripheral workload. The example AES function encrypts data in the cypher block chain mode, and can process up to 4KB data with a 128-, 192-, or 256-bit key length. As mentioned, we measured ΔV_{task} as the energy budget of these tasks. We used Device 1 in Table 5.2, which has 11.5 µF energy buffering capacitance, for the tests for variable data sizes and peripheral configurations, whereas in the device variability test we tested

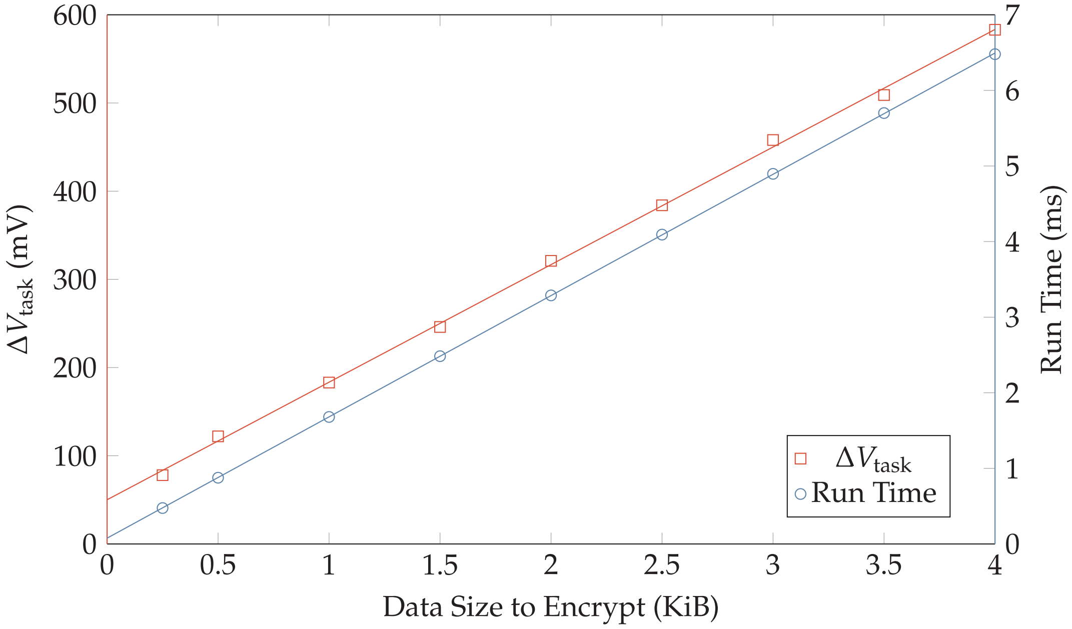


FIGURE 5.1: ΔV_{task} varying linearly with the data size in AES 128-bit encryption.

3 devices. We explored four factors that can possibly change ΔV_{task}, which are variable data amounts, variable peripheral configurations, devices variability, and capacitor degradation and tolerance. Besides the above four, energy consumption can also change with other factors, such as temperature, clock frequency, and silicon ageing, but we have found them either insignificant or hard to validate on our experimental platform.

5.1.1.1 Variable Data Sizes

A peripheral function can accept a runtime variable amount of data, such as a variable data size to encrypt or different lengths of packets for a radio to transmit. An example of this is plotted in Figure 5.1, where the size of the square dots represent a 5 mV precision error of the scope and the lines represent linear regression. We observed that ΔV_{task} has a linear relationship with the data size, with an offset energy consumption that accounts for the initialisation. In this case, the linearly scaled ΔV_{task} comes from linearly scaled run time.

5.1.1.2 Variability in Peripheral Configurations

A peripheral can run with variable configurations at runtime, and demonstrate variable performance and energy consumption. For example, as shown in Table 5.1, an AES accelerator can encrypt data with 128-, 192-, or 256-bit keys. A longer key provides higher security, but also takes more time and energy to complete. The dynamic range of

Configuration	$\Delta V_{\textbf{task}}$	**Run Time**
128-bit key	583 mV	6.479 ms
192-bit key	690 mV	7.638 ms
256-bit key	736 mV	8.606 ms

TABLE 5.1: ΔV_{task} Varying with Configurations in AES 4KB Encryption.

Device No.	$\Delta V_{\textbf{task}}$	**Run Time**
1	583 mV	6.479 ms
2	555 mV	6.444 ms
3	535 mV	6.462 ms

TABLE 5.2: ΔV_{task} Varying among Devices in AES 128-bit 4KB Encryption.

configuration variability in this case can be a 26% increase in ΔV_{task} and a 33% increase in run time.

5.1.1.3 Device Variability

Devices have their variation in power consumption, even with the same part number. A threshold profiled on one device can be inadequate on another. We did a test on the same three development boards, where they run 128-bit AES encryption on 4KB data. As listed in Table 5.2, the effect of device variability on ΔV_{task} is up to 9% among the three devices, though with almost the same run time (0.5% variation). It should also be noticed that device variability can present across platforms that run the same or similar code.

5.1.1.4 Capacitor Ageing and Tolerance

As the component for buffering energy in IPSs, capacitors typically present a $\pm$10-20% tolerance on rated capacitance as reported in many commercial capacitors [109, 119–121]. Capacitors also age over time [122]. It is shown that capacitance can decrease by 7.2% in 3000 hours (125 days) under a 25 °C ambient temperature in experiments [123], and by 50% within 10 years under 40 °C as manufacturers stated [124]. A degraded capacitor does not change the load consumption, but can increase ΔV_{task}, and hence makes the pre-defined voltage threshold unsafe or inefficient.

The above four examples present that the variability in IPSs can potentially make a predefined ΔV_{task} insufficient. It is unrealistic to profile the ΔV_{task} in each scenario at design time in practice considering the complexity of the variations, and still cannot encompass unexpected situations, necessitating a runtime energy profiling approach.

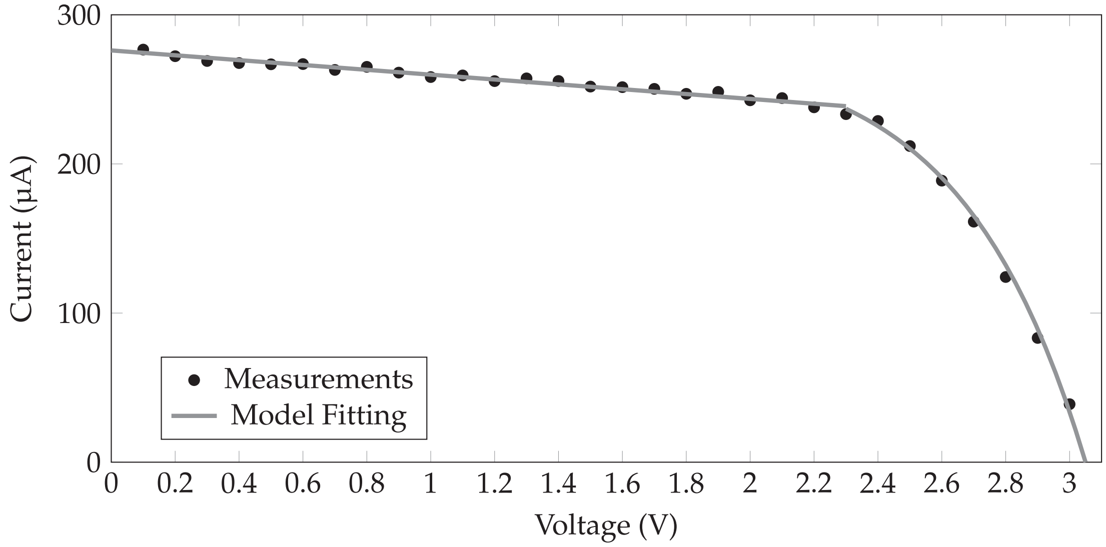


FIGURE 5.2: An I-V curve of a glass-type amorphous PV panel (Sanyo AM-1417CA, 35 mm × 13.9 mm) under a white LED lighting condition.

5.1.2 Performance Improvement with Adaptive Thresholds

Having presented the variability in IPSs, we explore in modelling and simulation the potential of adaptive thresholds on coping with such variability, as opposed to existing fixed-threshold approaches, which may fail or run inefficiently under such variability.

5.1.2.1 Power Analysis

As suggested in prior work [53, 118], operating at a lower voltage can improve system energy efficiency due to a higher charging efficiency and a lower power consumption.

To validate this, we analysed the charging characteristic of a glass-type amorphous PV panel in an white LED lighting environment. We used the PV panel to charge a capacitor with 103 µF capacitance as measured from 0 V to 3.05 V, at which point the capacitor cannot be charged further. The voltage-time charging trace was then differentiated to gain an I-V curve that represents the PV panel in the model (Figure 5.2). To model this curve, we performed a linear regression for the data in 0–2.3 V, and adapted a published PV panel model [114] to represent the curve in 2.3–3.05 V. The model function of this I-V curve is then expressed as:

$$I_{\text{in}} = \begin{cases} (-16.25V_{\text{in}} + 276.10) \times 10^{-6} & , \quad 0\,\text{V} < V_{\text{in}} <= 2.3\,\text{V} \\ I_{\text{sc}}(1 - (1 - \frac{I_{\text{mpp}}}{I_{\text{sc}}})^{\frac{V_{\text{in}} - V_{\text{oc}}}{V_{\text{mpp}} - V_{\text{oc}}}}) & , \quad 2.3\,\text{V} < V_{\text{in}} <= 3.05\,\text{V} \end{cases} \quad \text{(A)} \qquad (5.1)$$

where we set $I_{\text{sc}} = 276\,\mu\text{A}$, $V_{\text{oc}} = 3.05\,\text{V}$, $I_{\text{mpp}} = 237\,\mu\text{A}$, and $V_{\text{mpp}} = 2.3\,\text{V}$.

In the MSP430FR5994 platform, we did not observe a significant change in current consumption with supply voltage (only up to 2%). This is predominantly due to an on-chip LDO that lowers down the external supply voltage to a constant internal supply voltage, and hence maintains a relatively stable current draw as the external supply voltage changes. Hence, we omitted the voltage effect on current consumption in this simulation.

We used the energy and time overheads of AES encryption presented in Section 5.1.1 to simulate the workload characteristics. In simulation, the current draw remains constant during one operation, but changes with dynamic data sizes and configurations due to the variable charge consumption and run time.

5.1.2.2 Runtime Control Models

We modelled an ideal adaptive threshold scheme, named as OPTIC Oracle, against two State-of-the-Art fixed-threshold schemes, i.e. DEBS [53] and Samoyed [40]. We focussed on modelling the control logic and threshold settings, and omitted the state management overhead as it can be dependent on the actual implementation.

In OPTIC Oracle, the system knows exactly how much energy is needed for the next operation and sets the lowest threshold that suffices the energy budget.

DEBS sets a minimum threshold for a fixed operation. We explored two cases of DEBS, labelled as DEBS Low and DEBS High. We firstly modelled DEBS Low, which does not foresee any possible changes in data sizes and configurations. DEBS Low's threshold was profiled with 1KB data and a 128-bit key length without considering any variability. We then modelled DEBS High in a case where it foresees the possible dynamic increase in workload consumption due to variable data sizes and configurations and sets its threshold based on the most energy-hungry setup, while it does not consider further capacitor ageing.

Samoyed differs from DEBS and OPTIC Oracle in its control, where, when completing an operation, it keeps executing until it dies rather than sleeps and waits for the next threshold. Samoyed suggests allocating an abundant energy budget, so its threshold is also set to the highest possible operating voltage.

5.1.2.3 Simulation Setup

The above models were implemented as a numeric simulation program in Python. In simulation, the system has 10 µF system capacitance without charge at the start. The shutdown threshold is 1.8 V, against which DEBS and OPTIC Oracle set their threshold, with a 10 mV small margin. The system consumes 10 µA when it is inactive. To evaluate the performance of the three schemes, we conducted two tests that simulate a variable workload and capacitance reduction respectively. The variable workload test runs a random data amount from 16B to 4080B (1 to 255 blocks of data, 16B per block), and also a random 128-, 192-, or 256-bit key length, both uniformly distributed. The energy harvesting characteristics presented in Figure 5.2 are used as the supply for this test. All the schemes take the same random series of data sizes and configurations. The capacitance reduction test runs with 0-60% reduced capacitance, in line with the maximum possible reduction shown in Section 5.1.1.4. The system in this test is supplied with a 50 µA constant current and runs only the most energy-hungry operation, in order to examine whether the system can avoid non-termination even under the worst case. We ran 10 rounds of simulations for each setup, and each round simulates for 10 s.

5.1.2.4 Results

Figure 5.3 shows the mean, maximum, and minimum numbers of completed and failed operations in the variable workload test. DEBS Low cannot terminate once it encounters an operation that consumes more than what it is profiled for and the supply is too weak to provide the energy gap. DEBS Low can only occasionally get progress on lightweight operations before non-termination. Samoyed also suffers performance loss from waiting for a high energy threshold (2.9 V in this case), and failing an operation at the end of an active cycle. DEBS High is relatively efficient because it does not usually fail due to a sufficient energy budget and a sleep-after-completion control. OPTIC Oracle runs the most efficiently among these four. It runs at reduced operating voltage that improves system energy efficiency, and also guarantees the completion of every task by setting a minimised but safe threshold. As an example voltage trace shown in Figure 5.4, OPTIC Oracle runs with 2.11 V mean voltage, while the ones for Samoyed and DEBS High are 2.36 V and 2.40 V respectively. Due to the above reasons, OPTIC Oracle completes more operations over Samoyed by 50% and DEBS High by 15% on average.

Figure 5.5 shows the results of the capacitance reduction test, where we have omitted DEBS Low as it has already failed in non-termination with original capacitance (so will still fail with reduced capacitance). With the capacitance decreased, DEBS High also falls into non-termination like DEBS Low. Samoyed, where its threshold is set to 3.6 V in this case, can still progress until a 60% reduction of capacitance because its abundant energy budget can support at least one or a few operations in one active cycle. OPTIC Oracle still maintains the highest forward progress among these control scheme.

The above exploration presents that using a fixed low threshold can leave the system in non-termination (e.g. DEBS Low and DEBS High) but allocating an abundant energy budget compromises system efficiency (DEBS High and Samoyed). An adaptive threshold can potentially overcome both problems.

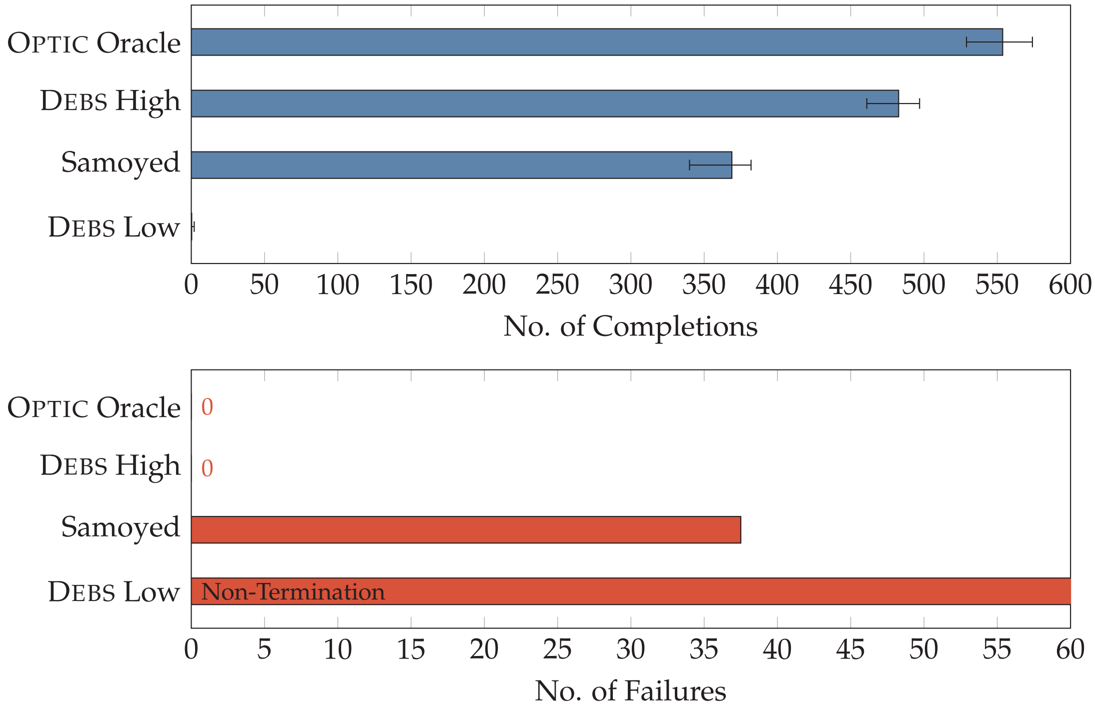


FIGURE 5.3: Numbers of completed and failed operations of DEBS Low, Samoyed, DEBS High, and OPTIC Oracle given random data sizes and configurations and a PV supply in a 10 s simulation.

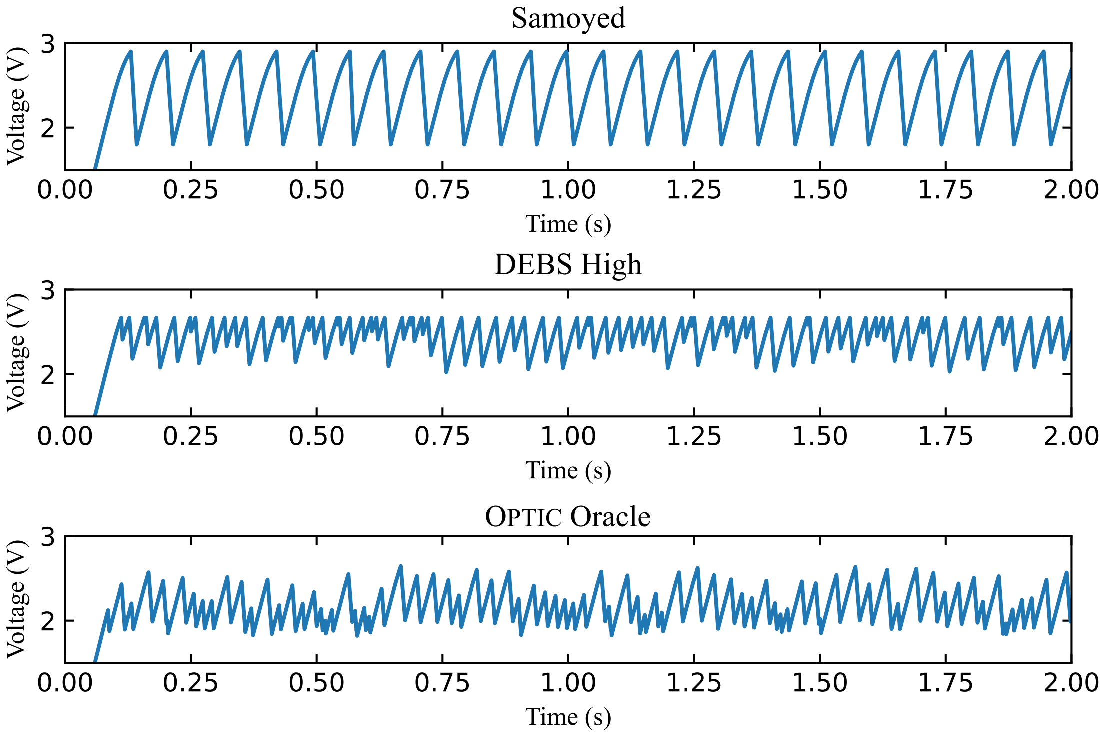


FIGURE 5.4: An instance of supply voltage traces in simulation.

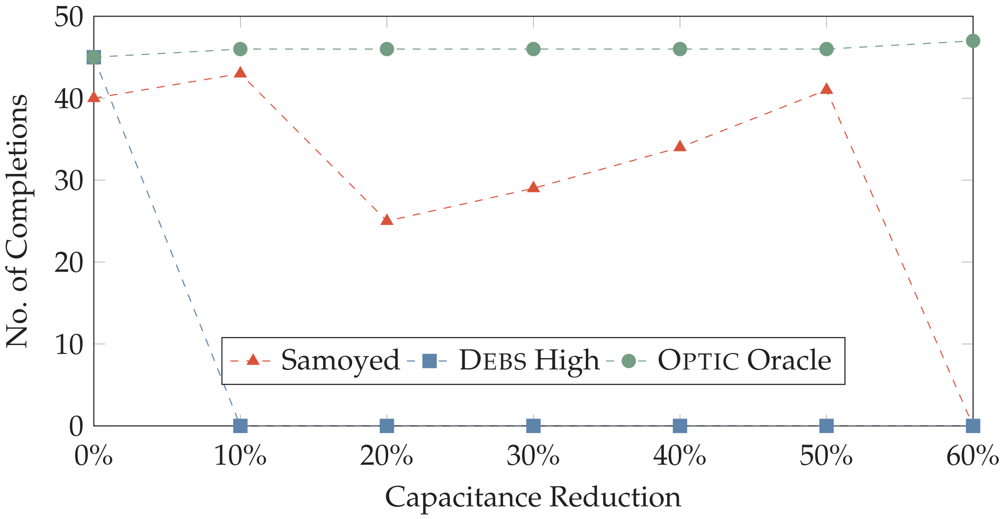


FIGURE 5.5: Number of completed operations of Samoyed, DEBS High, and the OPTIC Oracle with capacitance reduction.

Motivated by the previous examples of variable energy consumption and the benefit of an adaptive threshold, we propose OPTIC, a new methodology for profiling energy consumption of tasks at runtime and adapting energy budgets to the variable energy consumption of tasks.

5.2 OPTIC Runtime Energy Profiling

OPTIC's runtime energy profiling method efficiently profiles the maximum drop of supply voltage that a task can cause, i.e. the aforementioned ΔV_{task}. Unlike the previous disconnecting-supply method, OPTIC's performs energy profiling with the supply connected, so as to reserve energy input during profiling. When the supply is connected, ΔV_{task} cannot be measured simply by two voltage measurements at the beginning and the end of a task because the supply keeps charging the system during execution. Instead, OPTIC analyses the supply current in the charge cycle, and uses it to derive ΔV_{task} in the discharge cycle.

OPTIC's runtime energy profiling method assumes an IPS is able to measure the supply voltage and to record time when asleep and active. Usually, these two functions can be achieved with efficient on-chip ADCs and timers on common off-the-shelf MCUs, e.g. MSP430FR series (an example platform used in the IPS literature).

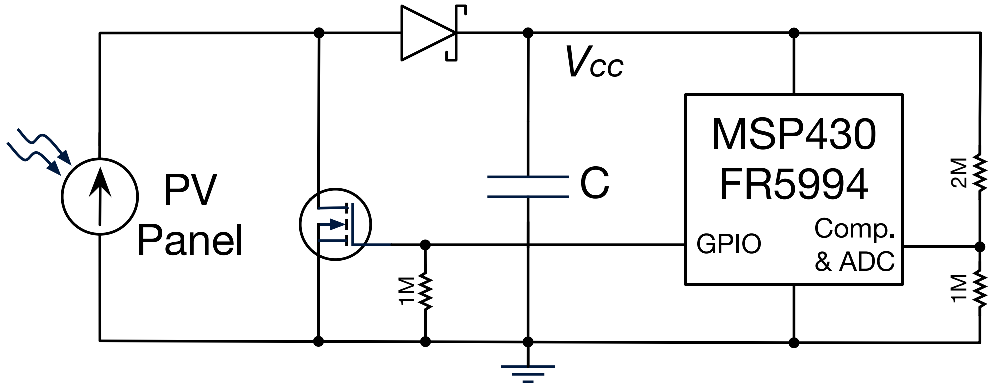


FIGURE 5.6: Experimental schematic.

5.2.1 A disconnecting-supply approach

Before introducing OPTIC's energy profiling method, a naive method for runtime energy profiling is disconnecting the supply and measure the supply voltage at the beginning (V_1) and the end (V_2) of a task, hence ΔV_{task} can be calculated as:

$$\Delta V_{\text{task}} = V_1 - V_2 \tag{5.2}$$

However, this can waste the energy input during the task execution. The total wasted charge is:

$$Q_{\text{waste}} = I_{\text{in}} T_{\text{task}} \tag{5.3}$$

which increases linearly with the current input and the time length of a task.

An example circuit to achieve this method is shown in Figure 5.6. It utilises an N-channel FET to short-circuit the energy harvesting supply during calibration, with a pull-down resistor to keep the gate low when MCU is not powered. The supply is decoupled by a diode to prevent current backflow. The MSP430FR5994 MCU uses its internal comparator and ADC to monitor and measure a divided supply voltage $\frac{1}{3}V_{cc}$.

5.2.2 Principles

To obtain ΔV_{task}, OPTIC's runtime energy profiling method compensates the voltage difference before and after an operation by an estimated voltage gain brought by the supply during the operation. The estimated voltage gain is calculated by measuring the charging ability in the last charge cycle and scaling it with the duration of the discharge cycle. Thus, OPTIC takes three voltage readings and two timer readings to perform one energy profiling.

To focus on the profiling rationale in the following illustration, we temporarily assume the mean supply current in a charge cycle remains the same in the next discharge cycle,

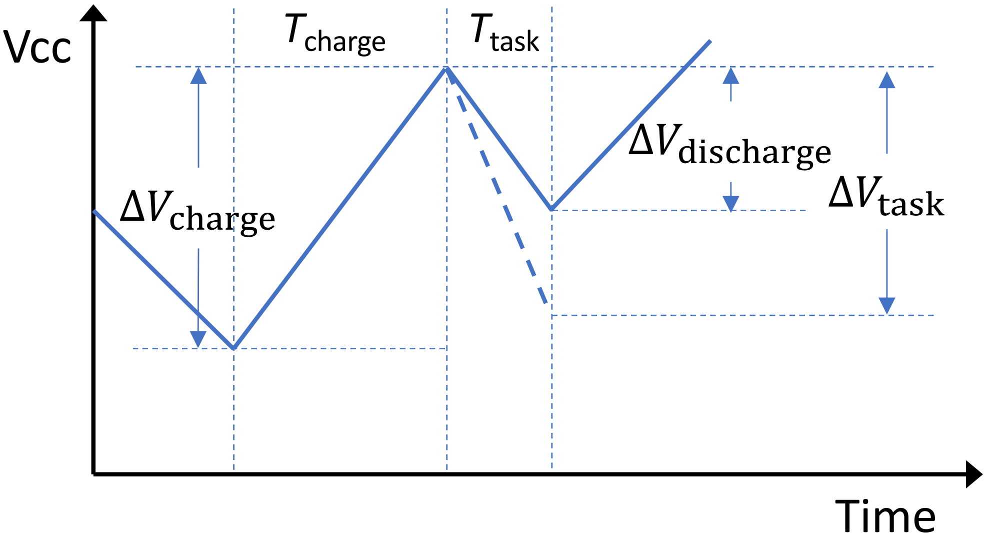


FIGURE 5.7: An illustrative supply voltage trace for explaining OPTIC's runtime energy profiling method.

Symbol	Definition
ΔV_{task}	Maximum voltage decrease of a task
ΔV_{charge}	Voltage increase of a charge cycle
$\Delta V_{\text{discharge}}$	Voltage decrease of a discharge cycle
C	System energy storage capacitance
I_{in}	Input current from energy harvester
I_{sleep}	System sleep current draw
I_{task}	System execution current draw during a task
T_{charge}	Time length of a charge cycle
T_{task}	Time length of a task execution cycle

TABLE 5.3: Definitions of Mathematical Symbols.

both denoted as I_{in}. We will discuss the effect of volatile supply current shortly. We also omit the overhead of ADC voltage reading here.

We show an illustrative trace of supply voltage across a charge-discharge cycle in Figure 5.7 with the symbols listed in Table 5.3.

The system charge increase in the charging cycle can be described as

$$\Delta V_{\text{charge}} C = (I_{\text{in}} - I_{\text{sleep}}) T_{\text{charge}} \tag{5.4}$$

where I_{sleep} is the current draw of the whole system when it sleeps and waits for energy refilling, including the current consumption for voltage monitoring, time recording, and other quiescent or leakage current.

The system charge reduction in the discharging cycle can be described as

$$\Delta V_{\text{discharge}} C = (I_{\text{task}} - I_{\text{in}}) T_{\text{task}} \tag{5.5}$$

where I_{task} consists of the current draw of the main workload, voltage monitoring, time recording, and other quiescent or leakage current.

The actual charge consumption of a task comes from the consumer part in Equation 5.5, which is

$$\Delta V_{\text{task}} C = I_{\text{task}} T_{\text{task}} \tag{5.6}$$

Hence, combining and rearranging Equation 5.4, Equation 5.5, and Equation 5.6, we can get the expression of ΔV_{task} as

$$\Delta V_{\text{task}} = \Delta V_{\text{discharge}} + \Delta V_{\text{charge}} \frac{T_{\text{task}}}{T_{\text{charge}}} + \frac{I_{\text{sleep}} T_{\text{task}}}{C} \tag{5.7}$$

ΔV_{task} is the actual voltage drop that a task can cause, and directly determines the voltage threshold a system should set for the task to safely complete. In Equation 5.7, ΔV_{charge}, $\Delta V_{\text{discharge}}$, T_{task}, and T_{charge} are all perceivable values. ΔV_{charge} and $\Delta V_{\text{discharge}}$ can be measured by three voltage readings at the transition points of charge and discharge cycles. T_{task} and T_{charge} can be measured with an on-chip timer.

$\frac{I_{\text{sleep}} T_{\text{task}}}{C}$ is a theoretical profiling error of this approach. If it is negligible or compensable, ΔV_{task} can be derived at runtime with all perceivable values.

5.2.3 Minimising and Compensating Theoretical Profiling Error

As the profiling method ignores the last term $\frac{I_{\text{sleep}} T_{\text{task}}}{C}$ in Equation 5.7, the profiled value can be theoretically smaller than the actual one. However, the empirical values of I_{sleep}, T_{task}, and C in IPSs indicate that this error is relatively small. The system sleep current I_{sleep} is a key property that is to be minimised in IPSs, and can be down to even sub-µA with modern low power techniques. The system's energy buffering capacitance C is typically in the µF level in IPSs. The execution time of a task T_{task} is typically a few or tens of ms as the energy buffering capacitor cannot afford a long, energy-hungry task. Hence, $\frac{I_{\text{sleep}} T_{\text{task}}}{C}$ should be typically under 10 mV. This is insignificant compared to the voltage drop of a task (potentially hundreds of mV), and can be easily or intrinsically compensated by margins in implementation. For example, a voltage comparator may not have such resolution and precision, and thus may over-provision a small energy budget that compensate this error. Manually adding a small software offset to the profiling results can also overcome this error. Therefore, this theoretical profiling error is insignificant in implementation and can be easily compensated.

5.2.4 Effect of Volatile Supply Current

The profiling method uses the average current input in the charge cycle as the current input in the next discharge cycle to derive the actual charge consumption. A charge-discharge cycle can be typically from tens to hundreds of ms, considering the capacitor size and the supply power in IPSs. From our practical observation on some types of energy harvesters (e.g. PV cells), OPTIC's profiling method performs stably (results presented in Section 5.5) as the supply current pattern complies with the assumption on supply current. However, we still anticipate there can be more volatile energy sources and discuss the consequent effect.

We denote the the mean current in charge and discharge cycles as $I_{\text{in\_charge}}$ and $I_{\text{in\_discharge}}$ respectively. When $I_{\text{in\_charge}} > I_{\text{in\_discharge}}$, the system over-profiles ΔV_{task} by $(I_{\text{in\_charge}} - I_{\text{in\_discharge}})T_{\text{task}}/C$ higher. When $I_{\text{in\_charge}} < I_{\text{in\_discharge}}$, the system under-profiles ΔV_{task} by $(I_{\text{in\_discharge}} - I_{\text{in\_charge}})T_{\text{task}}/C$ lower. While the over-profiled energy budget should be safe, the under-profiled energy budget could be inadequate, making the following task failed. An unfortunate case is when the system first under-profiles a task with a rapidly increasing supply current, and then executes the task again using the newly profiled budget while no further energy is harvested during the execution. This can lead to a task failure, where the system needs the existing approaches in IPSs to maintain atomic progress, e.g. disabling checkpoints during atomic sections. The over- or under-profiled results can be corrected when $I_{\text{in\_charge}}$ and $I_{\text{in\_discharge}}$ match again.

As discussed, it is indicated that OPTIC's profiling method is suitable for energy sources that are not liable to change significantly across a charge-discharge cycle. If the energy source is too volatile to obtain reliable profiling results, a disconnecting-supply profiling method could be adopted as a workaround.

5.3 OPTIC Runtime Energy Adaptation

OPTIC runtime energy adaptation utilises the presented runtime energy profiling to dynamically adapts the voltage threshold to the latest energy consumption of a task. The adaptation method assumes the system is able to monitor the supply voltage and signal the MCU to wake up or sleep when a high or low threshold is hit, and the threshold is configurable by the MCU at runtime. In practice, this voltage monitoring ability is widely adopted by IPSs in the forms of a voltage comparator [38, 39], an energy management unit [40, 53], or a periodic ADC polling [45].

The fundamental goal of the runtime energy adaptation is to allocate a barely sufficient energy budget for each task. Utilising the presented runtime energy profiling method, OPTIC is able to obtain the latest ΔV_{task} of a task and update its threshold accordingly.

The voltage threshold V_{th} of a barely sufficient energy budget is defined as:

$$V_{\text{th}} = \Delta V_{\text{task}} + V_{\text{end}} \tag{5.8}$$

where V_{end} is the target end voltage below which the whole or part of system's hardware cannot function correctly. V_{end} can be higher than the MCU shutdown voltage, e.g. a peripheral that has a higher operating voltage. Besides allocating the lowest V_{th}, an ideal adaptation scheme is also expected to have low overheads, run energy profiling only when necessary, and react to energy variations immediately.

Based on the above aims, we design OPTIC's runtime adaptation scheme. It consists of a basic adaptation routine for a fixed workload and an optional linear adaptation method for workloads that has a linearly-scaled ΔV_{task} with dynamic parameters. OPTIC's runtime energy adaptation is decoupled with the energy profiling method. The adaptation scheme allows the energy profiling method to be integrated in the routine but only requires an interface which allows it to trigger an instance of profiling and return a profiling result.

5.3.1 Adaptation Routine

A flowchart of OPTIC's adaptation routine is shown in Figure 5.8. The routine operates at the entry and the exit of a task. Checkpoints are disabled during the atomic task, so the program rolls back to a point before the task entry if a power interruption happens between the entry and the exit. The system executes the task body when V_{cc} is above V_{th}. If V_{cc} is below V_{th}, the system sleeps and waits until V_{th} is reached.

A non-volatile flag, ”failed”, is assigned for each task in order to monitor whether V_{end} is met with the current V_{th}. The ”failed” flag is set when a power interruption happens in the task body or when V_{cc} falls below V_{end} after the task body finishes. At the entry of a task, OPTIC checks whether ”failed” is set, i.e. whether V_{th} fails to meet V_{end} last time, and increment V_{th} if it is set. The increment amount can be dependent on volatility of energy consumption and the resolution of the adopted voltage monitor. In practice (Section 5.4), we found that incrementing one unit step of the voltage monitor, which corresponds to about 30 mV in our implementation, suffices both stability and reactivity.

Following the failure check, the routine has a control of when to trigger energy profiling (blue blocks in Figure 5.8), such that energy profiling is not performed every time that the task is run so as to save the energy and time overheads on unnecessary profiling. The control logic can be configured as per the requirements of users or applications. We have exemplified this with a delay counter, where the energy profiling is enabled every a number of completions. Alternatively, persistent timekeepers [125–127] can also be used to trigger energy profiling once a period of real time passes.

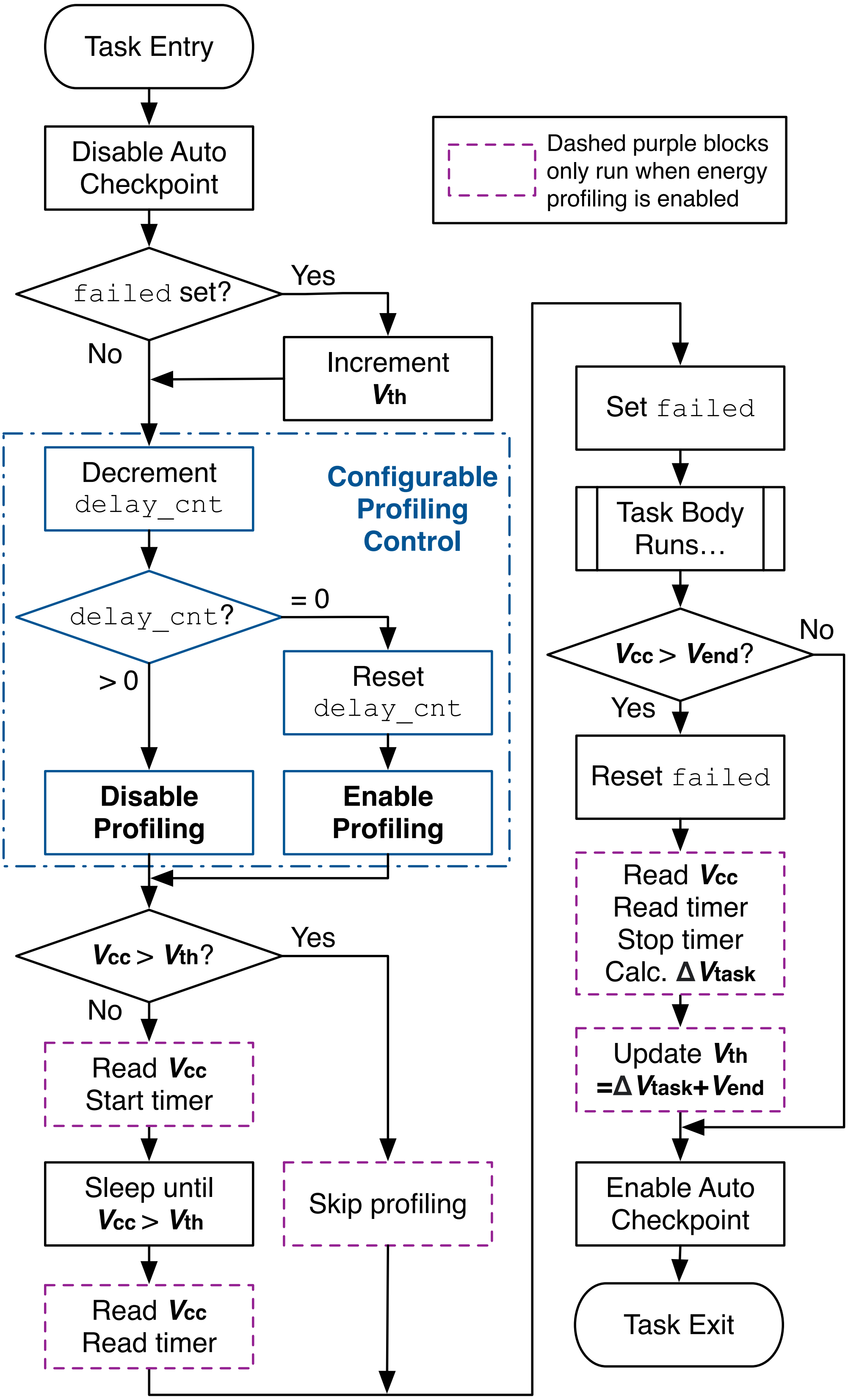

Task Entry
Disable Auto Checkpoint
failed set?
Yes
No
Increment Vth
Dashed purple blocks only run when energy profiling is enabled
Decrement delay_cnt
Configurable Profiling Control
delay_cnt?
= 0
> 0
Reset delay_cnt
Disable Profiling
Enable Profiling
Vcc > Vth?
Yes
No
Read Vcc
Start timer
Sleep until Vcc > Vth
Read Vcc
Read timer
Skip profiling
Set failed
Task Body Runs…
Vcc > Vend?
No
Yes
Reset failed
Read Vcc
Read timer
Stop timer
Calc. ΔVtask
Update Vth =ΔVtask+Vend
Enable Auto Checkpoint
Task Exit

FIGURE 5.8: Flowchart of OPTIC's runtime energy adaptation routine. The blue blocks represent a configurable control logic to decide when to perform energy profiling. The dashed purple blocks are only run when the profiling is enabled, and represent OPTIC's energy profiling with the last block updating V_{th} with the new ΔV_{task}.

If energy profiling is enabled, the dashed purple blocks in Figure 5.8 are performed in the routine. The particular operations involved can be dependent on the profiling method, while we have illustrated this in the flowchart with OPTIC's profiling method. As explained, OPTIC' energy profiling operates at three points when the charge cycle starts, when the charge cycle ends and the discharge cycle starts, and when the discharge cycle ends. V_{th} is then updated after a new ΔV_{task} is profiled following Equation 5.8. If a charge cycle is not needed, i.e. the energy stored is already sufficient for the task, the profiling is skipped and performed next time as this contradicts the design of OPTIC's profiling method.

5.3.2 Linear Adaptation

The above threshold adaptation is design for workloads that have a fixed amount of computational work and a determined configuration, the ΔV_{task} variation of which can change slowly with non-computational factors, e.g. capacitor ageing or temperature variations. For workloads that have runtime changeable parameters that scale energy consumption significantly, e.g. data sizes and peripheral configurations, the above adaptation can cost a number of failures before adapting to the new threshold. While dynamic configurations can be solved with multiple thresholds that switched by the configuration, data sizes can be fine-grained and can introduce a high memory overhead considering the number of thresholds needed. Hence, we propose a linear adaptation method as an option for workloads that have linearly-scaled energy consumption with its parameter.

Thus, a linearly-scaled ΔV_{task} can be represented as:

$$\Delta V_{\text{task}} = \theta_1 x + \theta_0 \tag{5.9}$$

where x is the parameter that is supposed to scale ΔV_{task}. θ_1 and θ_0 are the slope and y-intercept of the linear relationship between ΔV_{task} and x. Hence, V_{th} for the task should be set as:

$$V_{\text{th}} = \theta_1 x + \theta_0 + V_{\text{end}} \tag{5.10}$$

A straightforward solution to obtain θ_1 and θ_0 can be taking a series of profiling results and calculating the regression function. Though viable, this can introduce relatively high overheads on sampling and calculation.

To lower the overheads, we adopt an efficient method where energy profiling is performed to update θ_1 and θ_0 when x reaches its minimum or maximum values. θ_1 and θ_0 can then be calculated with less computation than linear regression. When the task is run with a x value other than the minimum or maximum, the energy profiling is disabled and θ_0 is incremented when necessary, e.g. a x value that causes a higher ΔV_{task} than what the equation predicts.

The routine of the linear adaptation method is then similar to the one shown in Figure 5.8, with modifications on the profiling control and the increment and update of V_{th}, where it controls whether to profile based on x, increments θ_0, and updates θ_1 and θ_0 rather than V_{th}.

5.4 Implementation

OPTIC was implemented based on an MSP430FR5994 development board. Its runtime is implemented as a C library and used with function calls. OPTIC is available open-source[3], along with the simulation program of design exploration, comparisons of runtime (Samoyed and DEBS), and benchmarks. A picture of OPTIC in experiments is shown in Figure 5.9.

The system schematic is shown in Figure 5.10. To reduce power consumption, the system utilises the MCU's on-chip ADC, timer, and voltage reference to perform energy profiling, and an external voltage monitor to monitor and control the threshold. The energy harvester is decoupled from the rest of the system by a diode to prevent the backflow of current when the harvester's power output drops off. The harvested energy is then buffered in a 10 µF capacitor (C_{ext}). Together with 1.5 µF on-board decoupling capacitance, the system has an energy buffering capacity of 11.5 µF in total.

The energy profiling is achieved through the on-chip modules. The ADC reads voltage from a built-in 1/2 V_{cc} channel, and thus an external voltage divider is not needed. A 2 V voltage reference is used by the ADC to convert the voltage reading, hence providing a 0–4 V reading range. A timer, driven by a 10 kHz low-power clock, records the charge and discharge cycle for OPTIC's energy profiling.

5.4.1 External Voltage Monitor

As shown in Figure 5.10, we built an external voltage monitor to control the threshold that signals the MCU to wake up or sleep. The voltage monitor consists of a 100 kΩ 129-step digital potentiometer (MCP4131-104) controlled through SPI, a voltage comparator (LT6703HVIS5-3) with 400 mV internal reference. Two resistors, 560 kΩ and 68 kΩ, were connected with the digital potentiometer to provide a detection range of 1.73–4.28 V, which covers the operating voltage range of the system. A 1 MΩ pull-up resistor was added at the comparator's open-collector output. Hence, the detected

---

[3] `https://github.com/UoS-EEC/optic`

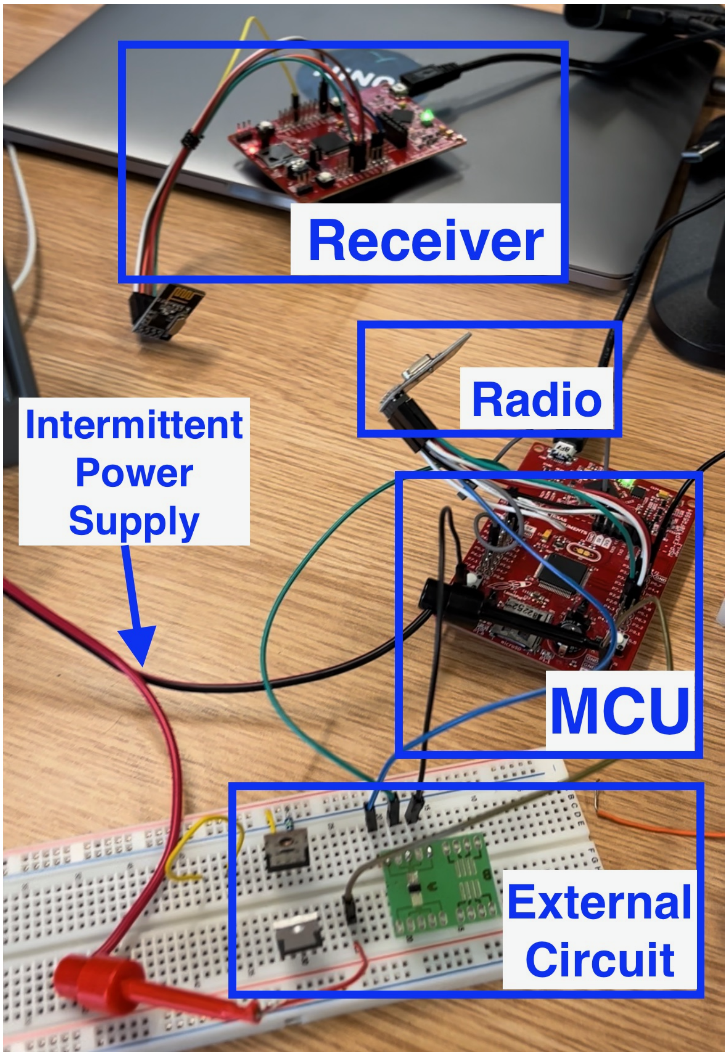


FIGURE 5.9: A picture of OPTIC's experimental setup based on an MSP430FR5994 development board. The MCU is intermittently powered by a digital power supply (constant low current source), and can also be powered by a PV panel. The workload in this picture transmits wireless packets to an always-on receiver.

voltage threshold of V_{cc} is:

$$V_{\text{th}} = \frac{560 + 100 + 68}{\frac{N_{\text{wiper}}}{128} * 100 + 68} \times 0.4 \quad (\text{V}) \tag{5.11}$$

where N_{wiper} is the wiper step of the potentiometer, ranged in 0-128 inclusively. Also, the profiling result of ΔV_{task} is stored as $N_{\text{profiling}}$ in a digital ADC-scale format:

$$\Delta V_{\text{task}} = \frac{N_{\text{profiling}}}{N_{\text{adcmax}}} \times V_{\text{adcmax}} \tag{5.12}$$

where ΔV_{task} is as defined in Equation 5.8. N_{adcmax} and V_{adcmax} are the maximum digital ADC reading and its corresponding voltage, which are 4095 and 4 V respectively in our implementation. Combining Equation 5.8, Equation 5.11, and Equation 5.12, we can

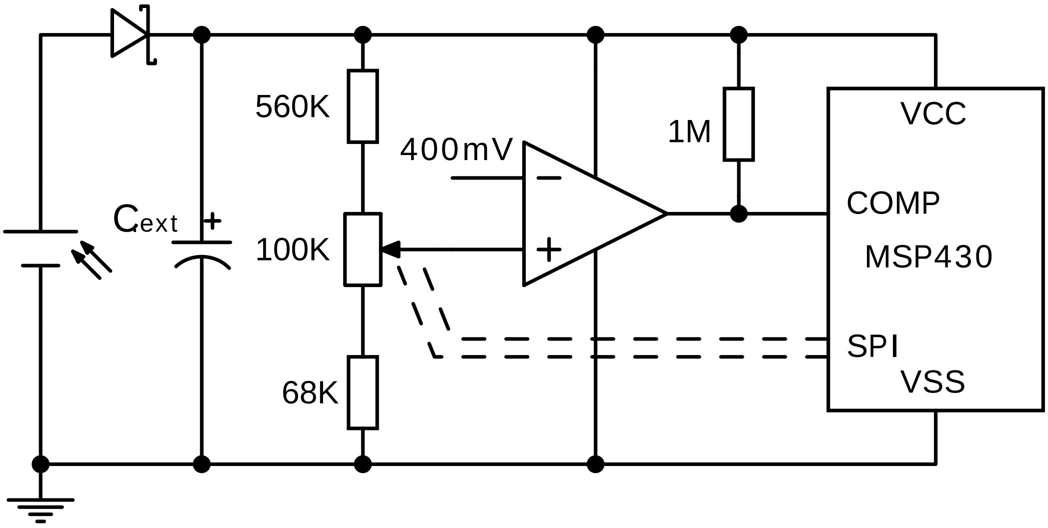


FIGURE 5.10: OPTIC system schematic.

obtain the relationship between $N_{\text{profiling}}$ and N_{wiper} as:

$$\frac{N_{\text{profiling}}}{N_{\text{adcmax}}} \times V_{\text{adcmax}} + V_{\text{end}} = \frac{560 + 100 + 68}{\frac{N_{\text{wiper}}}{128} * 100 + 68} \times 0.4 \tag{5.13}$$

where V_{end} is the target end voltage, which we set at 2 V because the energy profiling uses the 2 V voltage reference for ADC reading such that the energy profiling can correctly work above V_{end}.

In order to speed up the threshold setting from this non-linear relationship (Equation 5.13), We generated a look-up table to efficiently convert a profiling result $N_{\text{profiling}}$ into the corresponding voltage threshold setting N_{wiper}. To avoid unnecessarily fine-grained steps, we equally divide $N_{\text{profiling}}$ by a step of N_{step}. We recommend setting N_{step} as a power of 2 for an efficient threshold conversion, and we set N_{step} as 32, which translates to a voltage step of about 31 mV. We traversed N_{wiper} to find the closest V_{th} for each step of $N_{\text{profiling}}$, and the look-up table is then formed by the array of N_{wiper}. We also shifted the look-up table by one step higher so that the look-up table can inherently round up the threshold. Therefore, the corresponding threshold setting N_{wiper} of a profiling result $N_{\text{profiling}}$ can be found in the look-up table with a computation-efficient index of $\frac{N_{\text{profiling}}}{N_{\text{step}}}$ as shown in Equation 5.14.

$$N_{\text{wiper}} = lookup\_table(\frac{N_{\text{profiling}}}{N_{\text{step}}}) \tag{5.14}$$

5.4.2 Software

OPTIC's software is implemented as a library that accounts for the bootstrap configuration, function interfaces, memory mapping, and state retention. The bootstrap performs necessary system initialisation, such as configuring clocks, GPIOs, essential peripherals, and loading RAM data. OPTIC's software interface is implemented as two function calls at the entry and exit of an atomic task. Each atomic task should be assigned with a function ID such that its state is independent from other atomic tasks. The state of an atomic task consists of a minimum of 2 bytes non-volatile data that accounts for a failure check and an adaptive threshold, with optional data for a user-defined control logic (e.g. a delay counter) or linear adaptation. The state retention mechanism is implemented as a style of reactive intermittent computing as in [36, 37]. The usage of OPTIC's software is straightforward by assigning an ID to an atomic task in the library's header and calling the functions with the ID at the entry and exit of the atomic task.

5.5 Experimental Evaluation

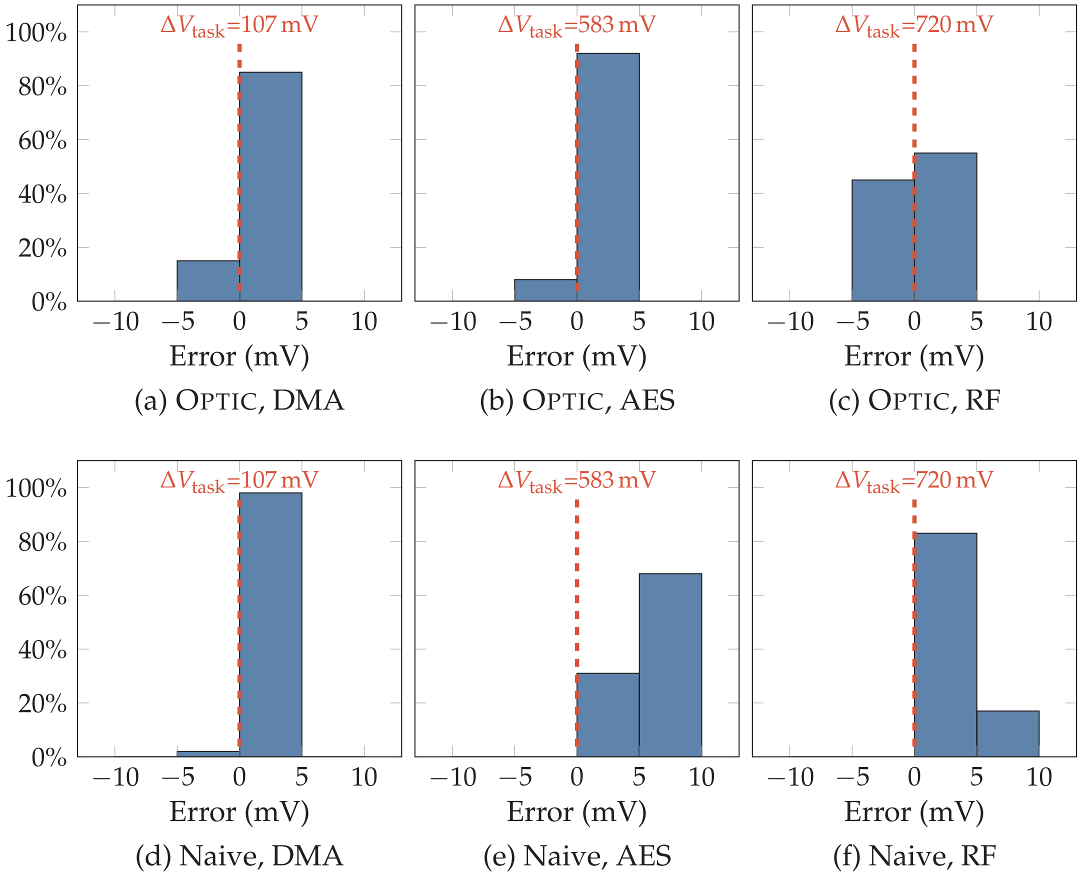
We experimentally evaluated OPTIC, showing its ability to run with an adaptive minimum threshold that mitigate non-termination and improve energy efficiency. OPTIC's runtime energy profiling presents a low and relatively consistent error across different task scales. We show that, despite with reduced capacitance, OPTIC is able to adapt V_{th} to meet a target end voltage V_{end} until the highest threshold is reached, while the fixed-threshold comparison DEBS fails. We also show that OPTIC improves performance over DEBS and Samoyed with a PV panel supply owning to its reduced operating voltage.

5.5.1 Experimental Setup and Benchmarks

A PV panel (Sanyo AM-1417CA) provided the sole power supply for the system. It is covered in a black box with a white LED light as the only energy source, producing a consistent supply characteristic (as shown in Figure 5.2) during the experiments. For the experiment on capacitance reduction only, we instead use a constant low-current supply so as to examine whether the system is able to survive with little energy income during task execution.

Three common peripheral tasks in IoT sensors were used as the benchmarks for evaluation.

- **DMA**: Data transfer using an on-chip DMA module, frequently used in data logging.



(a) OPTIC, DMA (b) OPTIC, AES (c) OPTIC, RF

(d) Naive, DMA (e) Naive, AES (f) Naive, RF

FIGURE 5.11: Error Distribution of OPTIC's Runtime Energy Profiling given a PV supply, compared to the "Naive" disconnecting-supply method.

- **AES**: AES encryption using an on-chip AES accelerator processing up to 4KB data at a time for secure communication.
- **RF**: Wireless communication through an external nRF24L01 radio module, transmitting a payload up to 96B at a time, configured as a 2Mbps air data rate and a 0 dBm output power. The radio module is connected through an LDO with a 2 V output voltage to lower the quiescent current consumption, with a 10 µF at the LDO's low side.

5.5.2 Profiling Accuracy

We first measured the profiling accuracy of OPTIC's runtime energy profiling ability. A hundred profiling results were obtained for each workload. Manual profiling was also conducted by disconnecting the power supply during task execution and reading ΔV_{task} from an oscilloscope, and used as a reference that we evaluate the profiling results against. The results are divided in a 5 mV step because the resolution of our scope is 5 mV, below which the manual profiled reference is not even accurate.

As shown in Figure 5.11, the profiling errors are within 5 mV and relatively consistent across the three workload with different levels of energy consumption. The error becomes insignificant with energy-hungry tasks, e.g. 0.7 % with RF. Compared to the step of voltage thresholds in our implementation (around 30 mV), this 5 mV error is acceptable as it can convert to a relatively stable threshold assuming a fixed energy consumption.

Additionally, the average profiling results are shown to be a slightly higher than the reference, which seems to contradict the theoretical error that is supposed to make the profiling undershoot. This is due to a positive error in the MCU's internal $1/2\ V_{cc}$ divider from an observation that the Naive approach produced a higher reading. This also evidences that the theoretical error is insignificant and easily compensated by other factors.

5.5.3 Reliability with Dynamic Energy Consumption

We evaluated whether OPTIC can adapt to variability in IPSs and keep making forward progress. We classified the variability in three categories according to the frequency of these changes:

- **Changing once**, such as new workloads, devices, and components.
- **Changing infrequently**, such as capacitor ageing, device ageing, temperature changes, and configurations that last for a long term.
- **Changing frequently**, such as variable data sizes and peripheral configurations that can frequently change.

To reduce similar results, we show one example in each category to illustrate OPTIC's adaptation to variability.

5.5.3.1 Changing once

Figure 5.12 shows an example of how OPTIC adapts its threshold to a new workload and a new device. The system runs an AES-128 encryption on 1KB data repetitively. The energy adaptation algorithm does not have any knowledge on the energy consumption of the platform or the workload.

The system first waits for the initial profiling threshold, which was set at 2.7 V in this case. Then it performs energy profiling as this was the first time it executes this task. The next threshold for the task was then adapted to a lower one. In the following execution, the system was able to maintain the same threshold that guarantees the completion of the task. The end voltage after completing a task matches closely with the target

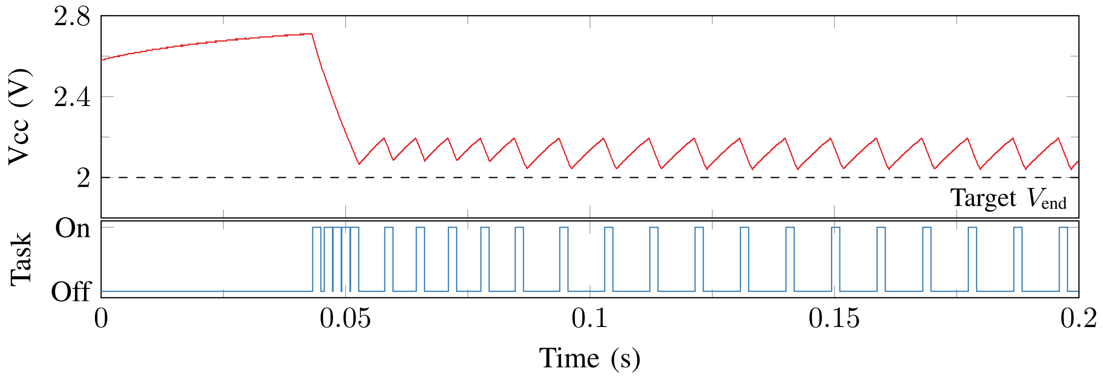


FIGURE 5.12: A voltage trace of OPTIC adapting to a new operation on a new device.

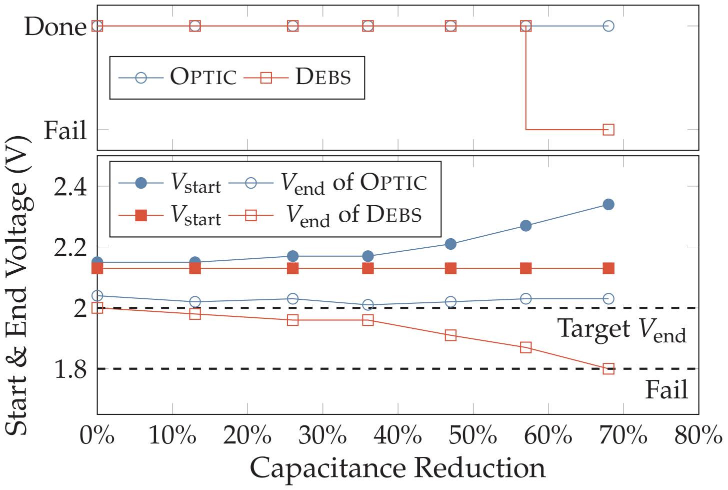
V_{end}, with a small margin that comes from both the round-up threshold and the energy harvested during the task execution. Hence, the above example shows OPTIC's ability to adapt to a new workload or device, obviating the need for manual energy profiling for various scenarios, e.g. updating workloads or deploying new devices.

5.5.3.2 Changing infrequently

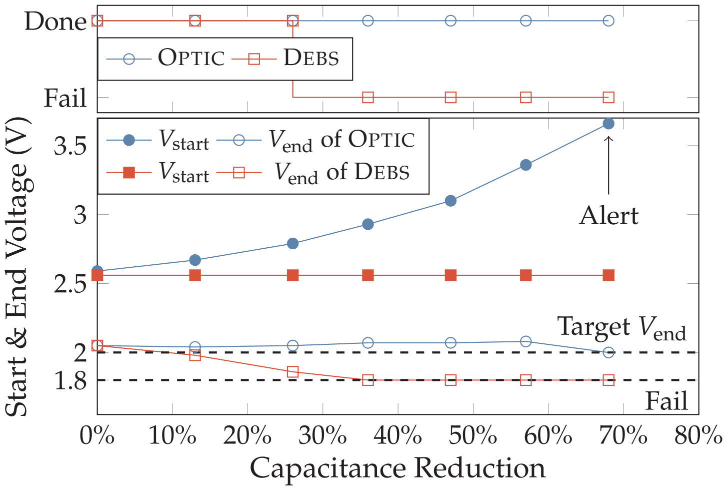
We then evaluated OPTIC's adaptation on infrequently or slowly changing ΔV_{task}. We took capacitor ageing as an example for this category of changes. The capacitor ageing was emulated with a capacitor bank consisting of 1 µF capacitors. The capacitor bank replaced the 10 µF C_{ext} in Figure 5.10, and hence the system capacitance could then be tuned in the range of 1.5–11.7 µF with 1.2–1.5 µF per step as measured.

In this experiment, the initial system capacitance was 11.7 µF, and was reduced step by step to test the system's ability against capacitor ageing. We compared OPTIC against DEBS in terms of whether it may fail. As the target end voltage for OPTIC in this implementation is 2 V, we configured the thresholds of DEBS against 2 V as well for a fair comparison, allowing additional energy before the shutdown threshold (1.8 V) is reached. We omitted the results of Samoyed as it assumes an abundant energy budget and the simulation results in Figure 5.5 indicate it is resilient to reduction of capacitance though with performance loss.

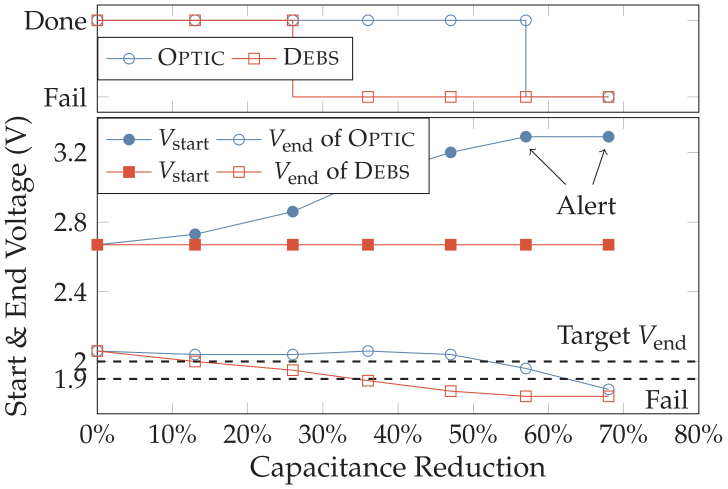
Figure 5.13 shows whether OPTIC and DEBS can safely complete the tasks with their threshold settings, along with their start and end voltages. In terms of meeting the target V_{end}, OPTIC is able to increase its threshold to prevent its end voltage dropping below the target V_{end}, while DEBS fails to do so with reduced capacitance as its threshold is fixed. The increase of OPTIC's threshold has a limit, where we set with the maximum operating voltage (3.6 V) for DMA and AES, and 3.3 V for RF beyond which the system's quiescent current draw becomes larger than the supply. OPTIC's threshold is increased with reduced capacitance until the upper limit is met, where it signals an



(A) DMA



(B) AES



(C) RF

FIGURE 5.13: Effect of Capacitor Degradation on OPTIC and DEBS.

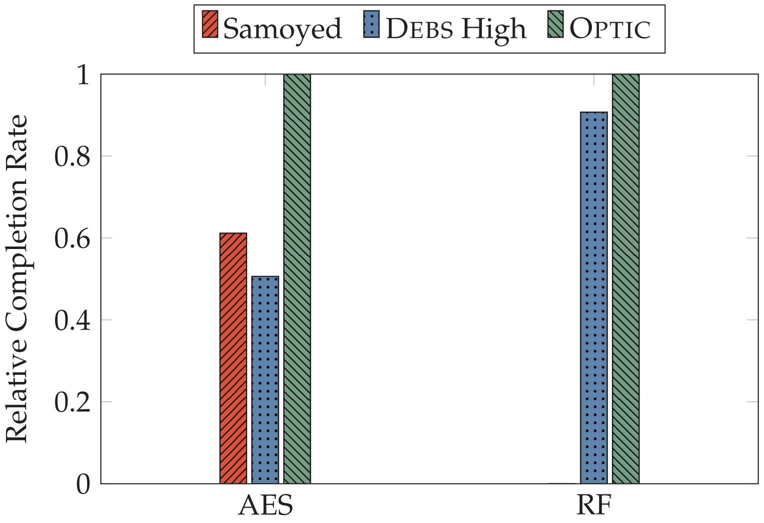


FIGURE 5.14: Relative Completion Rates of Samoyed, DEBS, and OPTIC with variable data sizes and a PV supply.

alert. In a practical scenario, the alert could be sent to maintainers and indicate further actions needed. Owing to the threshold adaptation, OPTIC can still survive with much lower capacitance, only failing the RF task with the lowest capacitance (67.5% reduction).

5.5.3.3 Changing frequently

A task may have runtime variable data sizes and configurations, which frequently change ΔV_{task}. While DEBS and Samoyed can set a high threshold that suffices the most energy-hungry task, OPTIC adapts its threshold to frequently changing ΔV_{task} so as to lower operating voltage and increase forward progress.

We demonstrate OPTIC's linear threshold adaptation on workloads that have variable data sizes, where AES encrypts 512B to 4KB data with a 256-bit key length, and RF transmits 16B to 96B data to an always-on receiver. An array of randomised numbers was generated to switch the data sizes. The average completion rate in a 30 s window was recorded as a performance metric. As shown in Figure 5.14, OPTIC managed to make 64% and 98% more progress compared to Samoyed and DEBS respectively on the AES workload. On the RF workload, the improvement compared to DEBS is 10% while Samoyed failed because the radio cannot reset to the correct state after a power failure and draws large current. The improvement of OPTIC comes from a lower threshold from which the system can harvest more energy and save the time on waiting for unnecessary energy.

5.6 Summary

Though previous IPS designs adopt fixed energy thresholds profiled at design time for atomic tasks, this chapter has shown that this can cause non-termination or reduce system energy efficiency. We found that the variability in IPSs can significantly change energy consumption. We presented four examples of such variability, i.e. variable data sizes, variable peripheral configurations, device variability, and capacitor ageing, all of which can, at runtime, violate a predefined energy threshold.

To address this issue, we proposed OPTIC, a runtime energy profiling and adaptation method. We proposed two methods of runtime energy profiling. A disconnecting-supply method measures the supply voltage difference before and after executing a task while short-circuiting the supply. While this is straightforward, it wastes the energy input during the task execution. A connecting-supply method for runtime energy profiling is also proposed, where it estimates the energy input during the task execution by measuring the current input before the task and compensating the supply voltage difference caused by a task. Experimental results has shown the proposed profiling method has a low error of less than 5 mV. This enables IPSs to profile energy consumption of tasks at runtime and alleviates manual profiling efforts in development. We also proposed a runtime energy adaptation routine that adapts the voltage threshold for a task utilising the proposed runtime energy profiling, with an option of linearly scaling the threshold by user-defined parameters.

We implemented OPTIC on a TI MSP430FR5994 MCU, with an external supply voltage monitor that can be efficiently configured and wake up the MCU when a threshold is hit. The experimental results showed that OPTIC can reliably adapt its threshold for a new task on a new device. OPTIC can also adapt to an increased ΔV_{task} caused by increased energy consumption or device ageing, e.g. up to 67.5% capacitance reduction that emulates an ageing capacitor, while the SoA fails, making IPSs possible to operate beyond capacitor lifetime. Finally, with variable data sizes or configurations, OPTIC is able to efficiently set barely sufficient energy thresholds that lowers operating voltage and improves energy efficiency, thus improving up to 98% progress over the SoA.

Chapter 6

Conclusions and Future Work

Enabled by energy harvesting and low-power computing techniques, IPSs are expected to be a promising system paradigm for numerous IoT sensors in the near future, with forecasts of hundreds of billions being installed [128]. IPSs adapt to the intrinsically variable and uncontrollable power input of energy harvesters, and thus, circumvent large volume, environmental impact, and limited lifespan of large energy buffers, i.e. batteries and supercapacitors, that contradict with the requirements of future IoT sensors.

While existing IPS technology has mainly focussed on the computing part, i.e. efficiently and correctly retaining the system state across power interruptions, this thesis has explored energy budgeting in IPSs. The energy budget in an IPS is represented as the energy allocated for an active cycle when the load wakes up and executes program. As the energy budget is determined by the energy storage size and the voltage threshold that wakes up the load, this thesis has conducted the research following the two aspects, where Chapter 3 and Chapter 4 explored the energy storage sizing effect and Chapter 5 proposed runtime profiling and adaptation of voltage thresholds.

6.1 Answers to Research Questions

1. **What is the effect of sizing the energy storage capacity on IPS performance?**

   To minimise device dimensions and interruption periods, most IPSs have adopted only a minimum amount of energy storage. However, as found in Chapter 3, this can be energy inefficient as the system has to frequently save and restore the state. Hence, a reactive IPS model was proposed to explore the sizing effect of energy storage on forward progress. Derived from the pattern of operating cycles in IPSs, the proposed model can fast and accurately estimates forward progress given supply current and energy storage capacitance, facilitating exploration and understanding of IPSs. The proposed model was configured with experimentally profiled parameters on a reactive IPS platform. The model was experimentally validated with a 0.5% mean error across multiple conditions of supply current and energy storage capacitance. The energy storage sizing effect was then explored with respect to supply current and volatile state size, showing a forward progress improvement of up to 65% compared to using minimised energy storage. The forward progress improvement from sizing energy storage becomes significant when supply current is low and when volatile state size is large. Additionally, the energy storage capacitance that achieves the maximum forward progress improvement (i.e. $C_{\alpha\_\max}$ as denoted in Chapter 3) can be 3.2× as large as the one that gains a 95% improvement. With considerations on volume and charging time of a large capacitor, this indicates that an energy storage sizing approach is in need to comprehend multiple design factors in IPSs instead of maximising forward progress only.

2. **How may the energy storage of IPSs be sized to trade off multiple design factors, such as forward progress, device dimensions, interruption periods?**

   As indicated in Chapter 3, an energy storage sizing approach for recommending an energy storage size considering multiple design factors when deploying IPSs was proposed in Chapter 4. Following a modelling and simulation process, the sizing approach is able to output forward progress, capacitor volume, and interruption periods given long-term energy source data, energy harvester configurations, and energy storage capacitance. Through iterations with different capacitance values, the sizing approach trades off various properties of the system with a cost function and recommends an appropriate energy storage size. The sizing approach was configured and demonstrated with parameters profiled on an IPS and real-world data of indoor and outdoor PV sources. The results showed up to a 43% annual forward progress gain by sizing energy storage. Corresponding to Chapter 3, this improvement is more significant with weaker power input, e.g. a smaller PV panel size. With an example cost function, the results showed that the suggested energy storage capacitance achieves 93% of the maximum forward progress while saving 83% capacitor volume and 91% interruption periods, compared to the one that solely maximises forward progress. Combining the findings in Chapter 3 and Chapter 4, the conclusion is that energy storage should be carefully designed, rather than minimised or indiscriminately picked, to efficiently operate IPSs.

3. **How can an IPS run safely and efficiently when executing tasks with runtime-variable energy consumption?**

While the energy storage size studied, Chapter 5 has focussed on the voltage thresholds of an energy budget. With the runtime variability of energy consumption, the prior SoA approaches can cause non-termination or reduce system energy efficiency. The variable energy consumption has been exemplified in four cases, which are variable data sizes, variable peripheral configurations, device variability, and capacitor ageing. Motivated by this variability, OPTIC, a runtime energy profiling and adaptation method, was presented in Chapter 5. OPTIC's runtime energy profiling measures ΔV_{task}, the drop of supply voltage caused by a task without any incoming energy meanwhile, with supply connected to save the input energy during profiling. To obtain ΔV_{task}, it measures the input current before the task and compensates the supply voltage difference of executing a task by the input current. Utilising the runtime energy profiling method, OPTIC's runtime energy adaptation adapts the voltage threshold for a task, efficiently allocating a barely sufficient threshold according to its runtime energy consumption. OPTIC's runtime energy adaptation also provides an option of linearly scaling the threshold by user-defined parameters, allowing a fast switching of thresholds without excessive profiling. OPTIC was implemented on a TI MSP430FR5994 MCU with an external supply voltage monitor. The experimental results has shown multiple findings as follows. OPTIC's runtime energy profiling has a low error within 5 mV, enabling IPSs to perform energy profiling at runtime and alleviating manual profiling efforts. OPTIC can adapt its threshold for a new task or on a new device. OPTIC is also able to cope with an increased ΔV_{task} from increased energy consumption or capacitor ageing, where it survived with up to 68% capacitance reduction while the SoA failed, allowing IPSs to operate beyond capacitor lifetime. Finally, OPTIC efficiently adjusts to a barely sufficient threshold with variable data sizes, which lowers operating voltage and improves energy efficiency, thus improving up to 98% progress over the SoA approaches.

6.2 Future Work

While this thesis has presented extensive research work contributed to the energy budgeting in IPSs, some interesting research topics can be explored in the future so as to achieve a more energy-efficient IPS. One interesting research research could be: *can IPSs transform from an MCU-centred system to an EMU-centred system, i.e. powering the load modules in a controllable and programmable order?*

Currently, an off-the-shelf energy management unit (EMU) for energy harvesting applications, e.g. TI BQ25504 [129], has the major features of boost charging, voltage detection, and power gating. Due to its features, an EMU can improve the energy efficiency of an IPS typically with low power input, where energy can be collected from a very weak supply (e.g. 130 mV input voltage) at a low quiescent current draw (e.g.

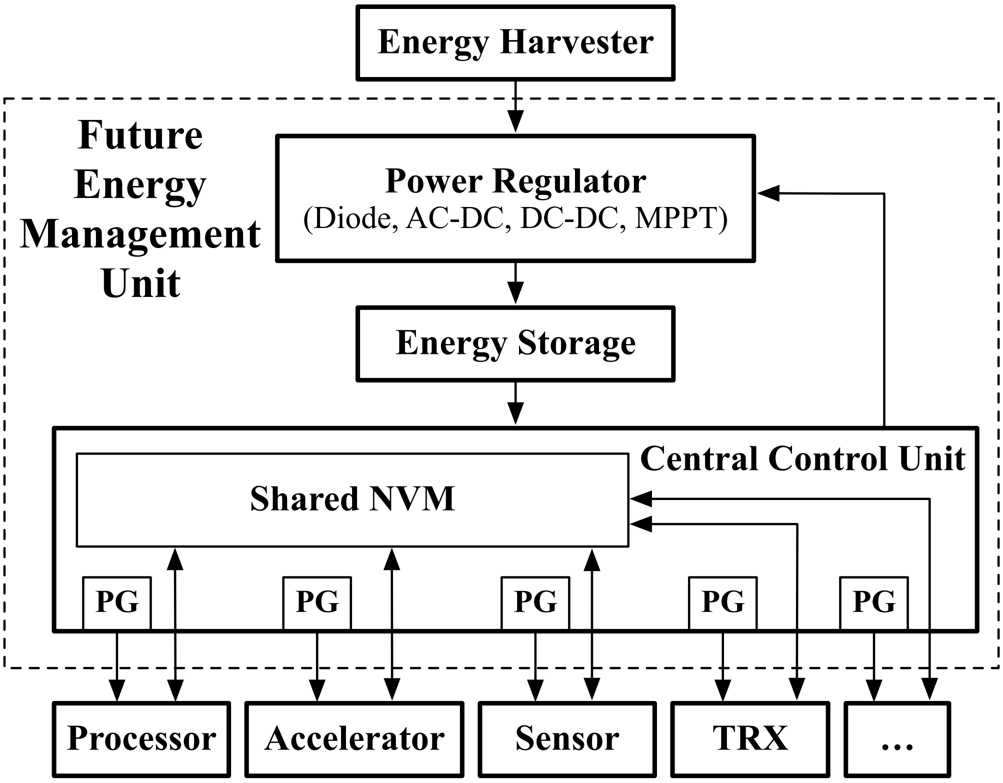


FIGURE 6.1: A conceptual architecture of an EMU-centred IPS.

330 nA). While energy management makes a significant impact on IPS performance as illustrated in this thesis, published IPS approaches are implemented on an MCU, where the MCU may interface with an EMU. To cope with the intermittent power supply, the MCU manages computing state and stored energy while executing application software, which deviates from what a processor is initially designed for.

It could be worthwhile to explore a dedicated EMU for IPSs that manages energy and state with power-gating on load modules. A major benefit of this could be a lower quiescent current consumption. External peripherals in existing MCU-centred IPSs are powered together with the MCU as a unified load. This increases the quiescent or sleep current when the system waits for energy to refill, especially with more peripherals connected or peripherals with large quiescent current consumption. With an EMU-centred IPS, each load module is power-gated and only powered when it is due to work. Thus, the system can work with a lower input current, ensuring the operation of IPSs in a poor energy condition. Other advantages could include (i) a lower control overhead as the IPS software could be offloaded to hardware, (ii) each module can independently work without a processor being active immediately before and after the operation of a module, and (iii) an integral IPS hardware that saves the area overhead of external components.

A conceptual architecture of such an EMU-centred IPS is depicted in Figure 6.1. The central control unit is responsible for the configuration of input power regulation (e.g.

MPPT), control of which module to power, and the data flow among modules through a shared NVM. The load modules, such as accelerators, sensors, and wireless transceivers (TRX), are connected to the EMU through a power gate (PG). The central control unit opens a corresponding PG when a load module is supposed to work. The load modules communicate through the shared NVM, where the volatile state and shared data are saved. The concept could be possibly achieved by incorporating IPS functions (e.g. state consistency) into existing EMUs. This may reconstruct the hardware architecture of existing IPSs.

Some potential subdivided research questions related to this topic could be:

1. Is an EMU-centred IPS theoretically more energy-efficient than existing MCU-centred IPSs considering the energy trade-off, e.g. initialisation overheads of modules vs. energy savings of power-gating?
2. How could an EMU-centred IPS be implemented?
3. Is it possible to integrate the functions of IPSs into a dedicated chip?
4. How can such a system be programmed so that it can have a configurable control logic?
5. Is a new programming model necessary for an EMU-centred IPS?
6. How can the system state be retained across multiple modules in a shared NVM?
7. Could an EMU-centred IPS condition the supply voltage for each module such that modules with different operating voltage levels can cooperate?

6.3 Conclusion

This thesis has presented extensive research work towards the energy budgeting for efficient operation of IPSs. A reactive IPS model was proposed to reveal the sizing effect of energy storage on forward progress, emphasizing that adding a small amount of energy storage beyond the minimum can improve IPS performance. A simulation tool was provided to estimate forward progress of IPSs given real environmental energy conditions, allowing an appropriate energy storage to be picked considering multiple design factors. To safely and efficiently execute atomic tasks, a method of runtime energy profiling and adaptation was presented, enabling IPSs to adapt the wake-up voltage threshold to variable energy consumption and future device degradation. All software related to the research work in this thesis was open-source available, facilitating reproduction of results and future research.

To summarise, this thesis has contributed methods of energy budgeting to improve IPS performance alongside the existing research work on efficient state-retention techniques. Researchers and developers are enabled to observe how the performance of their IPSs may change with the energy storage size and the supply current without comprehensive and repetitive experimental measurements. Given environmental energy data, a recommended energy storage size for deploying an IPS can be obtained through a simulation process with a customizable cost function that trades off different design factors, such as total forward progress, physical dimensions, and interruption periods. With OPTIC, the IPS is able to automatically adjust its voltage threshold to runtime variable operating conditions, dispelling the need for manual energy profiling of atomic tasks and improving performance by a just-sufficient dynamic threshold.

References

[1] Jie Zhan, Geoff V. Merrett, and Alex S. Weddell. Exploring the effect of energy storage sizing on intermittent computing system performance. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 1–10, 2021.

[2] Jie Zhan, Alex S. Weddell, and Geoff V. Merrett. Adaptive energy budgeting for atomic operations in intermittently-powered systems. In *Proceedings of the 8th International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*, ENSsys '20, pages 82–83, New York, NY, USA, 2020. Association for Computing Machinery.

[3] Oliver Hahm, Emmanuel Baccelli, Hauke Petersen, and Nicolas Tsiftes. Operating systems for low-end devices in the internet of things: a survey. *IEEE Internet of Things Journal*, 3(5):720–734, 2016.

[4] Luca Mainetti, Luigi Patrono, and Antonio Vilei. Evolution of wireless sensor networks towards the internet of things: A survey. In *Software, Telecommunications and Computer Networks (SoftCOM), 2011 19th International Conference on*, pages 1–6. IEEE, 2011.

[5] Tosiron Adegbija, Anita Rogacs, Chandrakant Patel, and Ann Gordon-Ross. Microprocessor optimizations for the internet of things: A survey. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(1):7–20, 2017.

[6] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.

[7] Jan M. Rabaey, M. Josie Ammer, Julio L. Da Silva, Danny Patel, and Shad Roundy. Picoradio supports ad hoc ultra-low power wireless networking. *Computer*, 33(7):42–48, 2000.

[8] Mohd Shahnawaz Khan, Mehjbeen Javed, Md Tabish Rehman, Maryam Urooj, and Md Irshad Ahmad. Heavy metal pollution and risk assessment by the battery of toxicity tests. *Scientific Reports*, 10(1):1–10, 2020.

[9] Paul D. Mitcheson, Eric M. Yeatman, G. Kondala Rao, Andrew S. Holmes, and Tim C. Green. Energy harvesting from human and machine motion for wireless electronic devices. *Proceedings of the IEEE*, 96(9):1457–1486, 2008.

[10] Sravanthi Chalasani and J. M. Conrad. A survey of energy harvesting sources for embedded systems. In *IEEE SoutheastCon*, pages 442–447, April 2008.

[11] Sujesha Sudevalayam and Purushottam Kulkarni. Energy harvesting sensor nodes: Survey and implications. *IEEE Communications Surveys & Tutorials*, 13(3):443–461, 2011.

[12] Aman Kansal, Jason Hsu, Sadaf Zahedi, and Mani B. Srivastava. Power management in energy harvesting sensor networks. *ACM Transactions on Embedded Computing Systems (TECS)*, 6(4):32, 2007.

[13] Bernhard Buchli, Felix Sutton, Jan Beutel, and Lothar Thiele. Dynamic power management for long-term energy neutral operation of solar energy harvesting systems. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, pages 31–45. ACM, 2014.

[14] Trong Nhan Le, Olivier Sentieys, Olivier Berder, Alain Pegatoquet, and Cecile Belleudy. Power manager with pid controller in energy harvesting wireless sensor networks. In *Green Computing and Communications (GreenCom), 2012 IEEE International Conference on*, pages 668–670. IEEE, 2012.

[15] Antonio Caruso, Stefano Chessa, Soledad Escolar, Xavier del Toro, and Juan Carlos López. A dynamic programming algorithm for high-level task scheduling in energy harvesting iot. *IEEE Internet of Things Journal*, 2018.

[16] Peter Wägemann, Tobias Distler, Heiko Janker, Phillip Raffeck, Volkmar Sieh, and Wolfgang SchröDer-Preikschat. Operating energy-neutral real-time systems. *ACM Transactions on Embedded Computing Systems (TECS)*, 17(1):11, 2018.

[17] Fayaz Akhtar and Mubashir Husain Rehmani. Energy replenishment using renewable and traditional energy resources for sustainable wireless sensor networks: A review. *Renewable and Sustainable Energy Reviews*, 45:769–784, 2015.

[18] Daler Rakhmatov, Sarma Vrudhula, and Deborah A. Wallach. Battery lifetime prediction for energy-aware computing. In *Proceedings of the 2002 international symposium on Low power electronics and design*, pages 154–159. ACM, 2002.

[19] Geoff V. Merrett and Alex S. Weddell. Supercapacitor leakage in energy-harvesting sensor nodes: Fact or fiction? In *Networked Sensing Systems (INSS), 2012 Ninth International Conference on*, pages 1–5. IEEE, 2012.

[20] Farhan Simjee and Pai H. Chou. Efficient charging of supercapacitors for extended lifetime of wireless sensor nodes. *IEEE Transactions on power electronics*, 23(3):1526–1536, 2008.

[21] Xiaofan Jiang, Joseph Polastre, and David Culler. Perpetual environmentally powered sensor networks. In *Proceedings of the 4th international symposium on Information processing in sensor networks*, page 65. IEEE Press, 2005.

[22] Farhan Simjee and Pai H. Chou. Everlast: long-life, supercapacitor-operated wireless sensor node. In *Proceedings of the 2006 international symposium on Low power electronics and design*, pages 197–202. ACM, 2006.

[23] Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. Wireless sensor networks: a survey. *Computer networks*, 38(4):393–422, 2002.

[24] Daniele Miorandi, Sabrina Sicari, Francesco De Pellegrini, and Imrich Chlamtac. Internet of things: Vision, applications and research challenges. *Ad hoc networks*, 10(7):1497–1516, 2012.

[25] Brandon Lucia, Vignesh Balaji, Alexei Colin, Kiwan Maeng, and Emily Ruppel. Intermittent Computing: Challenges and Opportunities. In Benjamin S. Lerner, Rastislav Bodík, and Shriram Krishnamurthi, editors, *2nd Summit on Advances in Programming Languages (SNAPL 2017)*, volume 71 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 8:1–8:14, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[26] Rishad Shafik, Alex Yakovlev, and Shidhartha Das. Real-power computing. *IEEE Transactions on Computers*, 67(10):1445–1461, 2018.

[27] Sivert T. Sliper, Oktay Cetinkaya, Alex S. Weddell, Bashir Al-Hashimi, and Geoff V. Merrett. Energy-driven computing. *Philosophical Transactions of the Royal Society A*, 378(2164), 2020.

[28] Benjamin Ransford, Jacob Sorber, and Kevin Fu. 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*, ASPLOS XVI, pages 159–170, New York, NY, USA, 2011. ACM.

[29] Naveed Anwar Bhatti and Luca Mottola. Harvos: Efficient code instrumentation for transiently-powered embedded sensing. In *Proceedings of the 16th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 209–219. ACM, 2017.

[30] Kiwan Maeng and Brandon Lucia. Adaptive dynamic checkpointing for safe efficient intermittent computing. In *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*, pages 129–144, 2018.

[31] Priyanka Singla, Shubhankar Suman Singh, and Smruti R. Sarangi. Flexicheck: An adaptive checkpointing architecture for energy harvesting devices. In *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 546–551. IEEE, 2019.

[32] Brandon Lucia and Benjamin Ransford. A simpler, safer programming and execution model for intermittent systems. In *Proceedings of the 36th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI ’15, pages 575–585, New York, NY, USA, 2015. ACM.

[33] Alexei Colin and Brandon Lucia. Chain: tasks and channels for reliable intermittent programs. *ACM SIGPLAN Notices*, 51(10):514–530, 2016.

[34] Kiwan Maeng, Alexei Colin, and Brandon Lucia. Alpaca: Intermittent execution without checkpoints. *Proc. ACM Program. Lang.*, 1(OOPSLA), October 2017.

[35] Amjad Yousef Majid, Carlo Delle Donne, Kiwan Maeng, Alexei Colin, Kasim Sinan Yildirim, Brandon Lucia, and Przemysław Pawełczak. Dynamic task-based intermittent execution for energy-harvesting devices. *ACM Trans. Sen. Netw.*, 16(1), February 2020.

[36] Hrishikesh Jayakumar, Arnab Raha, and Vijay Raghunathan. Quickrecall: A low overhead hw/sw approach for enabling computations across power cycles in transiently powered computers. In *2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems*, pages 330–335. IEEE, 2014.

[37] Domenico Balsamo, Alex S. Weddell, Geoff V. Merrett, Bashir M. Al-Hashimi, Davide Brunelli, and Luca Benini. Hibernus: Sustaining computation during intermittent supply for energy-harvesting systems. *IEEE Embedded Systems Letters*, 7(1):15–18, March 2015.

[38] Domenico Balsamo, Alex S. Weddell, Anup Das, Alberto Rodriguez Arreola, Davide Brunelli, Bashir M. Al-Hashimi, Geoff V. Merrett, and Luca Benini. Hibernus++: A self-calibrating and adaptive system for transiently-powered embedded devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(12):1968–1980, 2016.

[39] Chih-Kai Kang, Chun-Han Lin, Pi-Cheng Hsiu, and Ming-Syan Chen. Homerun: Hw/sw co-design for program atomicity on self-powered intermittent systems. In *Proceedings of the International Symposium on Low Power Electronics and Design*, ISLPED ’18, New York, NY, USA, 2018. Association for Computing Machinery.

[40] Kiwan Maeng and Brandon Lucia. Supporting peripherals in intermittent systems with just-in-time checkpoints. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 1101–1116. ACM, 2019.

[41] Alexei Colin, Emily Ruppel, and Brandon Lucia. 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, pages 767–781. ACM, 2018.

[42] Samuel Wong Chang Bing, Domenico Balsamo, and Geoff V. Merrett. An energy-driven wireless bicycle trip counter with zero energy storage. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*, pages 404–405. ACM, 2018.

[43] Samuel DeBruin, Bradford Campbell, and Prabal Dutta. Monjolo: An energy-harvesting energy meter architecture. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, page 18. ACM, 2013.

[44] Kaisheng Ma, Xueqing Li, Karthik Swaminathan, Yang Zheng, Shuangchen Li, Yongpan Liu, Yuan Xie, John Jack Sampson, and Vijaykrishnan Narayanan. Nonvolatile processor architectures: Efficient, reliable progress with unstable power. *IEEE Micro*, 36(3):72–83, 2016.

[45] Sivert T. Sliper, Domenico Balsamo, Nikos Nikoleris, William Wang, Alex S. Weddell, and Geoff V. Merrett. Efficient state retention through paged memory management for reactive transient computing. In *Proc. 56th DAC*, pages 26:1–26:6, June 2019.

[46] Saad Ahmed, Naveed Anwar Bhatti, Muhammad Hamad Alizai, Junaid Haroon Siddiqui, and Luca Mottola. Efficient intermittent computing with differential checkpointing. In *Proceedings of the 20th ACM SIGPLAN/SIGBED International Conference on Languages, Compilers, and Tools for Embedded Systems*, pages 70–81, 2019.

[47] Tim Daulby, Anand Savanth, Geoff V. Merrett, and Alex S. Weddell. Improving the forward progress of transient systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 40(3):444–452, 2020.

[48] Songran Liu, Wei Zhang, Mingsong Lv, Qiulin Chen, and Nan Guan. Latics: A low-overhead adaptive task-based intermittent computing system. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 39(11):3711–3723, 2020.

[49] Wei Zhang, Sonran Liu, Mingsong Lv, Qiulin Chen, and Nan Guan. Intermittent computing with efficient state backup by asynchronous dma. In *2021 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 543–548, 2021.

[50] Hrishikesh Jayakumar, Arnab Raha, Woo Suk Lee, and Vijay Raghunathan. Quickrecall: A HW/SW approach for computing across power cycles in transiently powered computers. *J. Emerg. Technol. Comput. Syst.*, 12(1), August 2015.

[51] Josiah Hester, Lanny Sitanayah, and Jacob Sorber. Tragedy of the coulombs: Federating energy storage for tiny, intermittently-powered sensors. In *Proc. 13th SenSys*, page 5–16, Seoul, South Korea, 2015.

[52] Domenico Balsamo, Benjamin J. Fletcher, Alex S. Weddell, Giorgos Karatziolas, Bashir M. Al-Hashimi, and Geoff V. Merrett. Momentum: Power-neutral performance scaling with intrinsic mppt for energy harvesting computing systems. *ACM Trans. Embed. Comput. Syst.*, 17(6), January 2019.

[53] Andres Gomez, Lukas Sigrist, Michele Magno, Luca Benini, and Lothar Thiele. Dynamic energy burst scaling for transiently powered systems. In *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 349–354. IEEE, 2016.

[54] Faisal Karim Shaikh and Sherali Zeadally. Energy harvesting in wireless sensor networks: A comprehensive review. *Renewable and Sustainable Energy Reviews*, 55:1041–1054, 2016.

[55] Scott D. Moss, Owen R. Payne, Genevieve A. Hart, and Chandarin Ung. Scaling and power density metrics of electromagnetic vibration energy harvesting devices. *Smart Materials and Structures*, 24(2):023001, 2015.

[56] Vijay Raghunathan, Aman Kansal, Jason Hsu, Jonathan Friedman, and Mani Srivastava. Design considerations for solar energy harvesting wireless embedded systems. In *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pages 457–462. IEEE, 2005.

[57] Winston K. G. Seah, Zhi Ang Eu, and Hwee-Pink Tan. Wireless sensor networks powered by ambient energy harvesting (WSN-HEAP)-survey and challenges. In *Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology, 2009. Wireless VITAE 2009. 1st International Conference on*, pages 1–5. IEEE, 2009.

[58] Giuseppina Ciulla, Valerio Lo Brano, Vincenzo Di Dio, and Giovanni Cipriani. A comparison of different one-diode models for the representation of i–v characteristic of a pv cell. *Renewable and Sustainable Energy Reviews*, 32:684–696, 2014.

[59] Oscar López-Lapeña, Maria Teresa Penella, and Manel Gasulla. A new mppt method for low-power solar energy harvesting. *IEEE Transactions on Industrial Electronics*, 57(9):3129–3138, 2010.

[60] Francisco Paz and Martin Ordonez. High-performance solar mppt using switching ripple identification based on a lock-in amplifier. *IEEE Transactions on Industrial Electronics*, 63(6):3595–3604, 2016.

[61] Deepak Verma, Savita Nema, A. M. Shandilya, and Soubhagya K. Dash. Maximum power point tracking (mppt) techniques: Recapitulation in solar photovoltaic systems. *Renewable and Sustainable Energy Reviews*, 54:1018–1034, 2016.

[62] Shad Roundy, Dan Steingart, Luc Frechette, Paul Wright, and Jan Rabaey. Power sources for wireless sensor networks. In *European workshop on wireless sensor networks*, pages 1–17. Springer, 2004.

[63] Cian O. Mathuna, Terence O'Donnell, Rafael V. Martinez-Catala, James Rohan, and Brendan O'Flynn. Energy scavenging for long-term deployable wireless sensor networks. *Talanta*, 75(3):613–623, 2008.

[64] Detlev Heinemann, Elke Lorenz, and Marco Girodo. Forecasting of solar radiation. *Solar energy resource management for electricity generation from local level to global scale. Nova Science Publishers, New York*, 2006.

[65] A. Andreas and S. Wilcox. Solar resource & meteorological assessment project (solrmap): Rotating shadowband radiometer (rsr); los angeles, california (data). *NREL Report No. DA-5500-56502*, 2012.

[66] Peter Corke, Philip Valencia, Pavan Sikka, Tim Wark, and Les Overs. Long-duration solar-powered wireless sensor networks. In *Proceedings of the 4th workshop on Embedded networked sensors*, pages 33–37. ACM, 2007.

[67] Aaron N. Parks, Alanson P. Sample, Yi Zhao, and Joshua R. Smith. A wireless sensing platform utilizing ambient rf energy. In *Power Amplifiers for Wireless and Radio Applications (PAWR), 2013 IEEE Topical Conference on*, pages 160–162. IEEE, 2013.

[68] Alanson P. Sample, Daniel J. Yeager, Pauline S. Powledge, Alexander V. Mamishev, and Joshua R. Smith. Design of an rfid-based battery-free programmable sensing platform. *IEEE transactions on instrumentation and measurement*, 57(11):2608–2615, 2008.

[69] Saman Naderiparizi, Aaron N. Parks, Zerina Kapetanovic, Benjamin Ransford, and Joshua R. Smith. Wispcam: A battery-free rfid camera. In *RFID (RFID), 2015 IEEE International Conference on*, pages 166–173. IEEE, 2015.

[70] Domenico Balsamo, Anup Das, Alex S. Weddell, Davide Brunelli, Bashir M. Al-Hashimi, Geoff V. Merrett, and Luca Benini. Graceful performance modulation for power-neutral transient computing systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(5):738–749, 2016.

[71] Navin Sharma, Jeremy Gummeson, David Irwin, and Prashant Shenoy. Cloudy computing: Leveraging weather forecasts in energy harvesting sensor systems. In *Sensor Mesh and Ad Hoc Communications and Networks (SECON), 2010 7th Annual IEEE Communications Society Conference on*, pages 1–9. IEEE, 2010.

[72] Alessandro Cammarano, Chiara Petrioli, and Dora Spenza. Pro-energy: A novel energy prediction model for solar and wind energy-harvesting wireless sensor networks. In *Mobile Adhoc and Sensor Systems (MASS), 2012 IEEE 9th International Conference on*, pages 75–83. IEEE, 2012.

[73] Raul Morais, Samuel G. Matos, Miguel A. Fernandes, António L. G. Valente, Salviano F. S. P. Soares, P. J. S. G. Ferreira, and M. J. C. S. Reis. Sun, wind and water flow as energy supply for small stationary data acquisition platforms. *Computers and electronics in agriculture*, 64(2):120–132, 2008.

[74] Jay Taneja, Jaein Jeong, and David Culler. Design, modeling, and capacity planning for micro-solar power sensor networks. In *Proceedings of the 7th international conference on Information processing in sensor networks*, pages 407–418. IEEE Computer Society, 2008.

[75] Michal Prauzek, Jaromir Konecny, Monika Borova, Karolina Janosova, Jakub Hlavica, and Petr Musilek. Energy harvesting sources, storage devices and system topologies for environmental wireless sensor networks: A review. *Sensors*, 18(8):2446, 2018.

[76] Paulo R. Bueno. Nanoscale origins of super-capacitance phenomena. *Journal of Power Sources*, 414:420–434, 2019.

[77] Jiří Libich, Josef Máca, Jiří Vondrák, Ondřej Čech, and Marie Sedlaříková. Supercapacitors: Properties and applications. *Journal of Energy Storage*, 17:224–227, 2018.

[78] Christian Renner, Jürgen Jessen, and Volker Turau. Lifetime prediction for supercapacitor-powered wireless sensor nodes. *Proceedings of FGSN*, pages 1–6, 2009.

[79] Chulsung Park and Pai H. Chou. Ambimax: Autonomous energy harvesting platform for multi-supply wireless sensor nodes. In *2006 3rd annual IEEE communications society on sensor and ad hoc communications and networks*, volume 1, pages 168–177. IEEE, 2006.

[80] Geoff V. Merrett and Bashir M. Al-Hashimi. Energy-driven computing: Rethinking the design of energy harvesting systems. In *Proceedings of the Conference on Design, Automation & Test in Europe*, pages 960–965. European Design and Automation Association, 2017.

[81] Soledad Escolar, Stefano Chessa, and Jesús Carretero. Energy-neutral networked wireless sensors. *Simulation Modelling Practice and Theory*, 43:1–15, 2014.

[82] Christopher M. Vigorito, Deepak Ganesan, and Andrew G. Barto. Adaptive control of duty cycling in energy-harvesting wireless sensor networks. In *Sensor,*

Mesh and Ad Hoc Communications and Networks, 2007. SECON'07. 4th Annual IEEE Communications Society Conference on, pages 21–30. IEEE, 2007.

[83] Joaquin Recas Piorno, Carlo Bergonzini, David Atienza, and Tajana Simunic Rosing. Prediction and management in energy harvested wireless sensor nodes. In *Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology, 2009. Wireless VITAE 2009. 1st International Conference on*, pages 6–10. IEEE, 2009.

[84] Matthew J. Walker, Stephan Diestelhorst, Andreas Hansson, Domenico Balsamo, Geoff V. Merrett, and Bashir M. Al-Hashimi. Thermally-aware composite runtime cpu power models. In *Power and Timing Modeling, Optimization and Simulation (PATMOS), 2016 26th International Workshop on*, pages 17–24. IEEE, 2016.

[85] Sivert T. Sliper, Domenico Balsamo, Alex S. Weddell, and Geoff V. Merrett. Enabling intermittent computing on high-performance out-of-order processors. In *Proceedings of the 6th International Workshop on Energy Harvesting & Energy-Neutral Sensing Systems*, pages 19–25. ACM, 2018.

[86] Josiah David Hester and Jacob Sorber. The future of sensing is batteryless, intermittent, and awesome. In *15th ACM Conference on Embedded Networked Sensor Systems (SenSys' 17)*, 2017.

[87] Alexei Colin and Brandon Lucia. Termination checking and task decomposition for task-based intermittent programs. In *Proceedings of the 27th International Conference on Compiler Construction*, pages 116–127. ACM, 2018.

[88] Alberto Rodriguez Arreola, Domenico Balsamo, Anup K. Das, Alex S. Weddell, Davide Brunelli, Bashir M. Al-Hashimi, and Geoff V. Merrett. Approaches to transient computing for energy harvesting systems: A quantitative evaluation. In *Proceedings of the 3rd International Workshop on Energy Harvesting & Energy Neutral Sensing Systems*, pages 3–8. ACM, 2015.

[89] Yongpan Liu, Zewei Li, Hehe Li, Yiqun Wang, Xueqing Li, Kaisheng Ma, Shuangchen Li, Meng-Fan Chang, Sampson John, Yuan Xie, et al. Ambient energy harvesting nonvolatile processors: from circuit to system. In *Proceedings of the 52nd Annual Design Automation Conference*, page 150. ACM, 2015.

[90] Yiqun Wang, Yongpan Liu, Shuangchen Li, Daming Zhang, Bo Zhao, Mei-Fang Chiang, Yanxin Yan, Baiko Sai, and Huazhong Yang. A 3us wake-up time nonvolatile processor based on ferroelectric flip-flops. In *ESSCIRC (ESSCIRC), 2012 Proceedings of the*, pages 149–152. IEEE, 2012.

[91] Fang Su, Yongpan Liu, Yiqun Wang, and Huazhong Yang. A ferroelectric nonvolatile processor with 46μs system-level wake-up time and 14μs sleep time for

energy harvesting applications. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 64(3):596–607, 2017.

[92] Yongpan Liu, Fang Su, Yixiong Yang, Zhibo Wang, Yiqun Wang, Zewei Li, Xueqing Li, Ryuji Yoshimura, Takashi Naiki, Takashi Tsuwa, et al. A 130-nm ferroelectric nonvolatile system-on-chip with direct peripheral restore architecture for transient computing system. *IEEE Journal of Solid-State Circuits*, 2019.

[93] Kaisheng Ma, Yang Zheng, Shuangchen Li, Karthik Swaminathan, Xueqing Li, Yongpan Liu, Jack Sampson, Yuan Xie, and Vijaykrishnan Narayanan. Architecture exploration for ambient energy harvesting nonvolatile processors. In *High Performance Computer Architecture (HPCA), 2015 IEEE 21st International Symposium on*, pages 526–537. IEEE, 2015.

[94] Kaisheng Ma, Xueqing Li, Jinyang Li, Yongpan Liu, Yuan Xie, Jack Sampson, Mahmut Taylan Kandemir, and Vijaykrishnan Narayanan. Incidental computing on iot nonvolatile processors. In *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 204–218. ACM, 2017.

[95] Kaisheng Ma, Xueqing Li, Karthik Swaminathan, Yang Zheng, Shuangchen Li, Yongpan Liu, Yuan Xie, John Jack Sampson, and Vijaykrishnan Narayanan. Nonvolatile processor architectures: Efficient, reliable progress with unstable power. *IEEE Micro*, 36(3):72–83, 2016.

[96] Fang Su, Kaisheng Ma, Xueqing Li, Tongda Wu, Yongpan Liu, and Vijaykrishnan Narayanan. Nonvolatile processors: Why is it trending? In *2017 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 966–971. IEEE, 2017.

[97] Benjamin J. Fletcher, Domenico Balsamo, and Geoff V. Merrett. Power neutral performance scaling for energy harvesting mp-socs. In *Proceedings of the Conference on Design, Automation & Test in Europe*, pages 1520–1525. European Design and Automation Association, 2017.

[98] Yiqun Wang, Yongpan Liu, Cong Wang, Zewei Li, Xiao Sheng, Hyung Gyu Lee, Naehyuck Chang, and Huazhong Yang. Storage-less and converter-less photovoltaic energy harvesting with maximum power point tracking for internet of things. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(2):173–186, 2016.

[99] Yung-Hsiang Lu, Luca Benini, and Giovanni De Micheli. Low-power task scheduling for multiple devices. In *Hardware/Software Codesign, 2000. CODES 2000. Proceedings of the Eighth International Workshop on*, pages 39–43. IEEE, 2000.

[100] Alberto Rodriguez Arreola, Domenico Balsamo, Geoff V. Merrett, and Alex S. Weddell. Restop: Retaining external peripheral state in intermittently-powered sensor systems. *Sensors*, 18(1), 2018.

[101] Fang Su, Yongpan Liu, Xiao Sheng, Hyung Gyu Lee, Naehyuck Chang, and Huazhong Yang. A task failure rate aware dual-channel solar power system for nonvolatile sensor nodes. *ACM Trans. Embed. Comput. Syst.*, 18(4), July 2019.

[102] Neal Jackson, Joshua Adkins, and Prabal Dutta. Capacity over capacitance for reliable energy harvesting sensors. In *Proceedings of the 18th International Conference on Information Processing in Sensor Networks*, IPSN '19, pages 193–204, New York, NY, USA, 2019. ACM.

[103] Yizi Gu, Yongpan Liu, Yiqun Wang, Hehe Li, and Huazhong Yang. NVPsim: A simulator for architecture explorations of nonvolatile processors. In *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 147–152. IEEE, January 2016.

[104] Sivert T. Sliper, William Wang, Nikos Nikoleris, Alex S. Weddell, and Geoff V. Merrett. Fused: closed-loop performance and energy simulation of embedded systems. In *2020 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 263–272. IEEE, 2020.

[105] Joshua San Miguel, Karthik Ganesan, Mario Badr, Chunqiu Xia, Rose Li, Hsuan Hsiao, and Natalie Enright Jerger. The EH model: Early design space exploration of intermittent processor architectures. In *2018 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 600–612, Fukuoka, Japan, Oct 2018.

[106] Josiah Hester, Timothy Scott, and Jacob Sorber. Ekho: Realistic and repeatable experimentation for tiny energy-harvesting sensors. In *Proc. 12th SenSys*, page 1–15, Memphis, Tennessee, 2014.

[107] Kai Geissdoerfer, Mikołaj Chwalisz, and Marco Zimmerling. Shepherd: A portable testbed for the batteryless iot. In *Proc. 17th SenSys*, page 83–95, New York, NY, USA, 2019.

[108] Radovan Faltus. Low leakage current aspect of designing with tantalum and niobium oxide capacitors. Technical report, AVX, 2012.

[109] AVX. TAJ low profile series tantalum capacitors. `http://datasheets.avx.com/TAJ_LOW_PROFILE.pdf`. Last accessed: 25 December, 2021.

[110] Kiwan Maeng and Brandon Lucia. Supporting peripherals in intermittent systems with just-in-time checkpoints. In *Proc. 40th PLDI*, pages 1101–1116, Phoenix, AZ, USA, 2019.

[111] Achraf Ben Amar, Ammar B. Kouki, and Hung Cao. Power approaches for implantable medical devices. *Sensors*, 15(11):28889–28914, 2015.

[112] A. Andreas and T. Stoffel. Nrel solar radiation research laboratory (srrl): Baseline measurement system (bms); golden, colorado (data). *NREL Report NO.DA-5500-56488*, 1981.

[113] Maria Gorlatova, Aya Wallwater, and Gil Zussman. Networking low-power energy harvesting devices: Measurements and algorithms. *IEEE Trans. Mobile Comput.*, 12(9):1853–1865, September 2013.

[114] Silvano Vergura. A complete and simplified datasheet-based model of pv cells in variable environmental conditions for circuit simulation. *Energies*, 9(5), 2016.

[115] Panasonic. Amorphous silicon solar cells. `https://mediap.industry.panasonic.eu/assets/download-files/import/ca_amorton_solar_cells_en.pdf`. Last accessed: 25 December, 2021.

[116] Evangelos Triantaphyllou. *Multi-criteria decision making methods: A comparative study*. Springer, 2000.

[117] Gautier Berthou, Pierre-Évariste Dagand, Delphine Demange, Rémi Oudin, and Tanguy Risset. Intermittent computing with peripherals, formally verified. In *The 21st ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems*, LCTES '20, page 85–96, New York, NY, USA, 2020. Association for Computing Machinery.

[118] Chen Pan, Mimi Xie, and Jingtong Hu. Maximize energy utilization for ultra-low energy harvesting powered embedded systems. In *2017 IEEE 23rd International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 1–6, 2017.

[119] AVX. AVX tantalum leaded capacitors. `http://www.farnell.com/datasheets/11903.pdf`. Last accessed: 25 December, 2021.

[120] Vishay. Solid tantalum surface mount chip capacitors. `https://www.vishay.com/docs/40002/293d.pdf`. Last accessed: 25 December, 2021.

[121] Nichicon. UPW aluminum electrolytic capacitors. `http://www.nichicon.co.jp/english/products/pdfs/e-upw.pdf`. Last accessed: 25 December, 2021.

[122] Alexander Teverovsky. Insulation resistance and leakage currents in low-voltage ceramic capacitors with cracks. *IEEE Transactions on Components, Packaging and Manufacturing Technology*, 4(7):1169–1176, 2014.

[123] Chetan Kulkarni, Gautam Biswas, Xenofon Koutsoukos, Jose Celaya, and Kai Goebel. Experimental studies of ageing in electrolytic capacitors. Technical report, 2010.

[124] Vishay. *Aluminum Electrolytic Capacitors Radial Low Leakage Current*, 2016.

[125] Jasper de Winkel, Carlo Delle Donne, Kasim Sinan Yildirim, Przemysław Pawełczak, and Josiah Hester. Reliable timekeeping for intermittent computing. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS ’20, pages 53–67, New York, NY, USA, 2020. Association for Computing Machinery.

[126] Vishal Deep, Vishak Narayanan, Mathew Wymore, Daji Qiao, and Henry Duwe. Harc: A heterogeneous array of redundant persistent clocks for batteryless, intermittently-powered systems. In *2020 IEEE Real-Time Systems Symposium (RTSS)*, pages 270–282, 2020.

[127] Josiah Hester, Nicole Tobias, Amir Rahmati, Lanny Sitanayah, Daniel Holcomb, Kevin Fu, Wayne P. Burleson, and Jacob Sorber. Persistent clocks for batteryless sensing devices. *ACM Trans. Embed. Comput. Syst.*, 15(4), aug 2016.

[128] Philip Sparks. The route to a trillion devices. Technical report, ARM Ltd., 2017.

[129] Texas Instruments. Bq25504 ultra low power boost converter with battery management for energy harvester. `https://www.ti.com/product/BQ25504`. Last accessed: 25 December, 2021.