Battery-Free Game Boy

De Winkel, Jasper; Kortbeek, Vito; Hester, Josiah; Pawelczak, Przemyslaw

DOI
10.1145/3411839

Publication date
2020

Document Version
Final published version

Published in
Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies

无电池游戏男孩

德温克尔，贾斯帕；科特贝克，维托；赫斯特，乔赛亚；帕维尔查克，普热梅斯瓦夫

DOI
10.1145/3411839
出版日期
2020
文档版本
最终发布版本
发表在
ACM关于互动、移动、可穿戴和无处不在技术的会议录

Battery-Free Game Boy

JASPER DE WINKEL, Delft University of Technology, The Netherlands
VITO KORTBEEK, Delft University of Technology, The Netherlands
JOSIAH HESTER, Northwestern University, USA
PRZEMYSŁAW PAWEŁCZAK, Delft University of Technology, The Netherlands

We present ENGAGE, the first battery-free, personal mobile gaming device powered by energy harvested from the gamer actions and sunlight. Our design implements a power failure resilient Nintendo Game Boy emulator that can run off-the-shelf classic Game Boy games like Tetris or Super Mario Land. This emulator is capable of intermittent operation by tracking memory usage, avoiding the need for always checkpointing all volatile memory, and decouples the game loop from user interface mechanics allowing for restoration after power failure. We build custom hardware that harvests energy from gamer button presses and sunlight, and leverages a mixed volatility memory architecture for efficient intermittent emulation of game binaries. Beyond a fun toy, our design represents the first battery-free system design for continuous user attention despite frequent power failures caused by intermittent energy harvesting. We tackle key challenges in intermittent computing for interaction including seamless displays and dynamic incentive-based gameplay for energy harvesting. This work provides a reference implementation and framework for a future of battery-free mobile gaming in a more sustainable Internet of Things.

CCS Concepts: • **Human-centered computing** → **Handheld game consoles**; • **Hardware** → **Renewable energy**; • **Computer systems organization** → **Embedded systems**;

Additional Key Words and Phrases: Energy Harvesting, Intermittent Computing, Battery-free

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

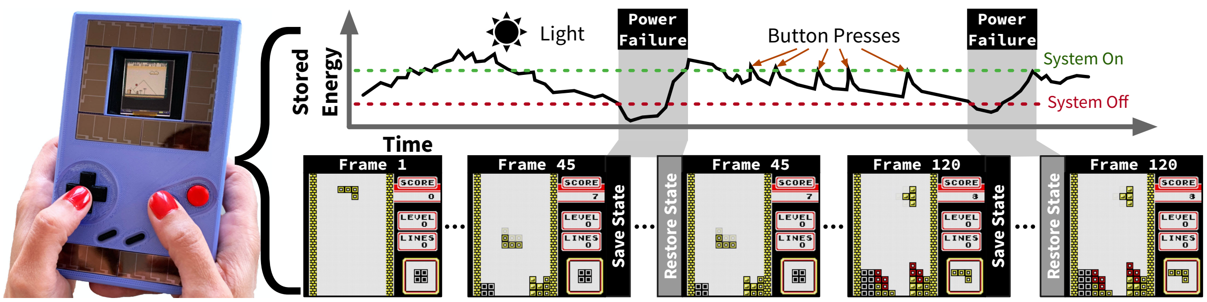


Fig. 1. Energy harvested from button presses and sunlight powers our custom handheld platform, ENGAGE, running a Nintendo Game Boy emulator which can play classic 8 bit games. ENGAGE efficiently preserves game progress despite power failures, demonstrating for the first time battery-free mobile entertainment.

Authors' addresses: Jasper de Winkel, Delft University of Technology, Delft, The Netherlands, j.dewinkel@tudelft.nl; Vito Kortbeek, Delft University of Technology, Delft, The Netherlands, v.kortbeek-1@tudelft.nl; Josiah Hester, Northwestern University, Evanston, IL, USA, josiah@northwestern.edu; Przemysław Pawełczak, Delft University of Technology, Delft, The Netherlands, p.pawelczak@tudelft.nl.

2474-9567/2020/9-ART111 $15.00
https://doi.org/10.1145/3411839

无电池游戏男孩

贾斯珀·德·温克尔,代尔夫特理工大学，荷兰
维托·科特比克,代尔夫特理工大学，荷兰
乔西亚·赫斯特,西北大学，美国
普热梅斯瓦夫·帕维尔恰克,代尔夫特理工大学，荷兰

我们推出了ENGAGE，这是第一款无电池的个人移动游戏设备，利用玩家的动作和阳光收集的能量供电。我们的设计实现了一个电源故障恢复的任天堂游戏男孩模拟器，可以运行现成的经典游戏男孩游戏，如《俄罗斯方块》或《超级马里奥乐园》。该模拟器能够通过跟踪内存使用情况实现间歇性操作，避免了始终检查所有易失性内存的需要，并将游戏循环与用户界面机制解耦，从而允许在电源故障后恢复。我们构建定制硬件，从游戏者的按钮按压和阳光中收集能量，并利用混合波动内存架构高效地间歇性模拟游戏二进制文件。我们的设计不仅仅是一个有趣的玩具，它代表了第一个无电池系统设计，能够在间歇性能量采集导致的频繁电源故障下持续吸引用户注意。我们解决了交互中间歇性计算的关键挑战，包括无缝显示和基于动态激励的能量采集游戏玩法。这项工作提供了一个参考实现和框架，为未来的无电池移动游戏在更可持续的物联网中奠定基础。CCS概念：•以人为本的计算 →手持游戏机; •硬件 →可再生能源; •计算机系统组织 →嵌入式系统;

附加关键词和短语：能量采集，间歇性计算，无电池

ACM参考格式：
Jasper de Winkel, Vito Kortbeek, Josiah Hester, 和 Przemysław Pawełczak. 2020. 无电池游戏男孩。*Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 3, 文章111（2020年9月），34页。https://doi.org/10.1145/3411839

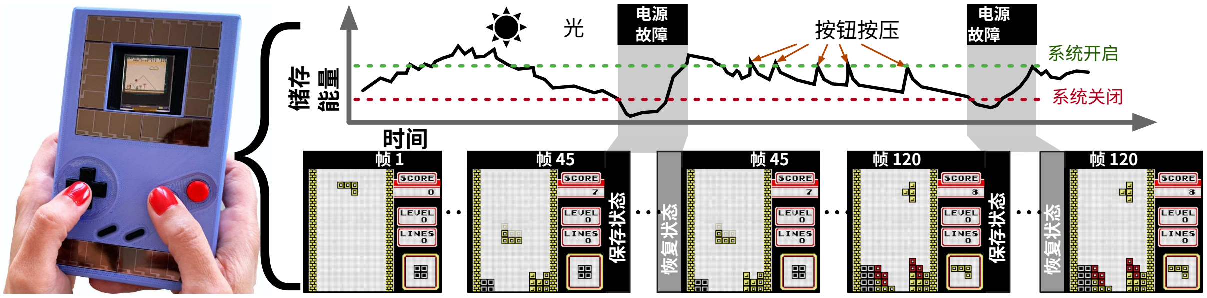


图 1.从按钮按压和阳光中收集的能量为我们的定制手持平台ENGAGE提供动力，该平台运行任天堂游戏男孩模拟器，可以玩经典的8位游戏。ENGAGE有效地保存游戏进度，尽管发生电源故障，首次展示了无电池移动娱乐的可能性。

作者地址：Jasper de Winkel，代尔夫特理工大学，荷兰代尔夫特，j.dewinkel@tudelft.nl；Vito Kortbeek，代尔夫特理工大学，荷兰代尔夫特，v.kortbeek-1@tudelft.nl；Josiah Hester，西北大学，美国伊凡斯顿，IL，josiah@northwestern.edu；Przemysław Pawełczak，代尔夫特理工大学，荷兰代尔夫特，p.pawelczak@tudelft.nl。

2474-9567/2020/9-ART111 $15.00
https://doi.org/10.1145/3411839

1 INTRODUCTION

This paper originates from the question: *can we game-on-the-go without batteries*? Batteries add size, weight, bulk, cost and especially inconvenience because of constant recharging—to any device. Energy can be generated by mashing buttons while gaming, and readily available energy from sunlight is all around us, so why not use this energy for battery-free mobile gaming! Significant challenges in software resiliency and efficiency, hardware operation and energy usage first need to be solved, but would represent a fundamental advancement over non-interactive (and not very fun) battery-free devices that currently exist.

Beyond the convenience and novelty factor of battery-free operation, the ecological impact of battery powered smart devices is troubling; technology advancements that reduce reliance on batteries in consumer devices would blunt the environmental impact of the trillion smart device [30, 49, 122] vision without losing the computational gains. With the emergence of low cost, small, and high-performance microcontrollers (MCUs), along with more efficient micro-energy harvesting devices that can harness the power of sunlight, motion, and heat, a new revolution in computing has come where Internet of Things devices are, in fact, increasingly leaving their batteries behind and relying only on ambient power from sunlight, motion, thermal gradients, and other modalities to power all operation [44, 89, 107].

Prototype battery-free devices have been used to make phone calls [126], deployed for machine learning [70], greenhouse monitoring [42], video streaming [92], eye tracking [73] and even built into a robot [150]. However, none of these techniques or prototypes have enabled *interactive battery-free devices*—like a smartwatch, in-place interactive display or even a **handheld video game console**. This is a critical gap in the research around battery-free devices, as these types of *reactive, interactive, and screen-focused* systems are a significant portion of the current and anticipated smart systems.

In this paper we focus specifically on this ignored part of the battery-free device ecosystem, *mobile gaming*, and use this application to elucidate the essential challenges that must be explored to get us to a future where reactive and user facing applications can also be battery-free. From a market perspective there is deep need to explore this area. The global gaming industry is massive and generates unprecedented revenues, which already exceeded 100 billion USD in 2016 [95]. Handheld console game sales constitute a large portion of the industry [95]. In terms of units sold, as of September 30, 2019 Nintendo sold 41.67 million units of its latest Switch console, since its release in March 3, 2019 [98] and these numbers continue to grow rapidly in the presence of a worldwide COVID-19 lockdown [100]. As a comparison, Nintendo's Game Boy handheld, shipped 118.69 million units since its official release in April of 1989.

To enable these types of devices, mobile gaming platforms must be re-imagined at the system and interactivity level. The main challenge is that energy harvesting is dynamic and unpredictable. This is intuitively apparent when considering a solar panel; a cloud, the time of day, weather conditions, movement and orientation of the panel, even the electrical load all change the amount of harvested energy. Because of this dynamism, these devices run out of energy and lose power frequently, only *intermittently* computing with the device having to wait seconds or minutes to gain enough energy to turn back on. This long recovery process can be energy and resource intensive, causing responsiveness delays. Worse, it can leave the game in an inconsistent state. Naturally, going through this entire re-loading process (from loading screen of a game to starting play) every time is burdensome, so just blindly replacing batteries in a game console with an energy harvester is not enough to ensure smooth game operation.

To address this challenge this paper presents a *framework of solutions based around energy-aware interactive computing* and a *reference implementation of a popular game console*—8 bit Nintendo Game boy [24, 99]—as a demonstration, see Figure 1. To reduce the unpredictability of energy harvesting, we take advantage of mechanical energy generated by "button mashing" of the console, harvesting this energy generated by actually playing a game on a handheld, and using it, along with solar panels, to power all operation. We design the system hardware

1 引言

本文源于一个问题：我们能否在没有电池的情况下进行移动游戏？电池增加了设备的体积、重量、笨重、成本，尤其是由于不断充电而带来的不便。在游戏时通过按键产生能量，以及随处可得的阳光能量，为什么不利用这些能量进行无电池的移动游戏呢！在软件的弹性和效率、硬件的操作和能量使用方面需要首先解决重大挑战，但这将代表相对于目前存在的非互动（且不太有趣的）无电池设备的根本性进步。

除了无电池操作的便利性和新奇性外，电池供电智能设备的生态影响令人担忧；减少消费设备对电池依赖的技术进步将减轻万亿智能设备[30, 49, 122]愿景的环境影响，而不会失去计算收益。随着低成本、小型和高性能微控制器（MCU）的出现，以及能够利用阳光、运动和热量的更高效的微能量采集设备，计算机领域迎来了新的革命，物联网设备实际上越来越多地抛弃了电池，仅依赖于来自阳光、运动、热梯度和其他方式的环境能量来驱动所有操作 [44, 89, 107]。

原型无电池设备已被用于拨打电话 [126]，用于机器学习 [70]，温室监测 [42]，视频流 [92]，眼动追踪 [73]，甚至被集成到机器人中 [150]。然而，这些技术或原型都未能实现互动式无电池设备——如智能手表、就地互动显示器或甚至手持视频游戏机。这是无电池设备研究中的一个关键空白，因为这些类型的反应式、互动式和以屏幕为中心的系统是当前和预期的智能系统的重要组成部分。

在本文中，我们特别关注这一被忽视的无电池设备生态系统中的部分，移动游戏，并利用这一应用阐明必须探索的基本挑战，以便让我们迈向一个反应式和用户面对的应用也可以无电池的未来。从市场的角度来看，深入探索这一领域的需求非常迫切。全球游戏产业庞大，产生了前所未有的收入，2016年已超过1000亿美元[95]。掌机游戏销售占据了该行业的很大一部分[95]。截至2019年9月30日，任天堂自2019年3月3日发布以来，已售出4167万台最新的Switch主机[98]，而在全球新冠病毒疫情封锁的情况下，这一数字仍在迅速增长[100]。作为对比，任天堂的Game Boy掌机自1989年4月正式发布以来，已出货1.1869亿台。

为了使这些类型的设备能够运行，移动游戏平台必须在系统和交互层面进行重新构想。主要挑战在于能量采集是动态且不可预测的。当考虑到太阳能电池板时，这一点直观上是显而易见的；云层、一天中的时间、天气条件、面板的运动和方向，甚至电负载都会改变采集到的能量量。由于这种动态性，这些设备经常耗尽能量并失去电力，只能间歇性地计算，设备必须等待几秒钟或几分钟才能获得足够的能量重新启动。这一漫长的恢复过程可能会消耗大量能量和资源，导致响应延迟。更糟糕的是，这可能会使游戏处于不一致的状态。

自然地，每次都经历整个重新加载过程（从游戏的加载屏幕到开始游戏）是繁琐的，因此仅仅盲目地用能量采集器替换游戏机中的电池并不足以确保游戏的顺利运行。

为了解决这个挑战，本文提出了一种基于能量感知交互计算的解决方案框架，以及一个流行游戏控制台的参考实现——8位任天堂游戏男孩[24,99]——作为演示，见图1。为了减少能量采集的不可预测性，我们利用控制台“按键狂按”产生的机械能，收集在手持设备上实际玩游戏时产生的能量，并与太阳能电池板一起使用，以供电所有操作。我们从零开始设计系统硬件

and software from the ground up to be energy-aware and reactive to changing energy situations to mitigate the issues caused by frequent power failures. Specifically we design a technique to create minimal *save games* that can be quickly created, updated, and saved to non-volatile memory before a power failure, then quickly restored once power returns. Unlike save games seen in traditional gaming systems, these capture the entire state of the system, so the player can recover from the exact point of power loss; for example mid-jump in a platform game; all this despite the device fully losing power.

Contributions. In this paper we present a practical, usable mobile gaming device, *Energy Aware Gaming* (abbreviated as *ENGAGE*). To date this is the first time full system emulation of a real world platform has been done battery-free, and the first intermittently powered interactive gaming platform. Our contributions follow:

(1) We introduce the concept of intermittently powered, battery-free mobile gaming;
(2) We develop an approach to failure resilient, memory-efficient, fast, whole system save games for interactive, display driven devices. A just-in-time differential checkpointing scheme is used based on the concept of tracking changed memory in *patches*;
(3) We develop a hardware platform as a reference implementation with a novel multi-input architecture for harvesting energy from button presses and sunlight. This device also enables any interactive-based system (not necessarily a game);
(4) As a stress test and demonstrative exercise of the promise of battery-free gaming, we use these systems and hardware to develop a full system Nintendo Game Boy emulator which plays unmodified Game Boy games despite power failures.

2 CHALLENGES

Personal, handheld gaming, has brought entertainment and fun to hundreds of millions of people in the past three decades. In the time of the COVID-19 pandemic, when so many are in lockdown, gaming is one of the activities that reduce stress and boredom [116, 118, 127]. The goal of this work is to develop the systems and hardware foundations for battery-free mobile gaming. This is motivated by two reasons: (i) the enhanced availability and usability of a platform that never needs to be recharged or plugged in—making the platform more convenient for the average user, and more accessible for everyone, and (ii) the need for alternative and sustainable forms of entertainment—a nod to the various industry consortia such as *Playing for the Planet* [108] which aim to reduce the gaming industries ecological impact. A battery-free handheld game console reduces ecological costs and disappointment, as it is always ready to be picked up and played *without needing to be recharged*.

Numerous explorations of battery-free smart devices address the calls for sustainable/carbon-neutral electronic device interaction and electronic design and computing [12, 65, 67, 85, 145] while preparing human-interactive electronics for the "post-collapse society" [131]. Other work has developed core systems [44], hardware [23, 28], and programming languages [82, 150] for serious systems focusing on solving the *intermittent computing problem* caused by energy harvesting and battery-free operation, where frequent power failures prevent a program from finishing a task (see Figure 2). In all cases the electronic device is powered by harvested energy from the environment [107] and stored in (super-)capacitors of much smaller energy density and size than batteries. None have yet explored the question of mobile handheld entertainment, going beyond the simple forms of battery-free gaming devices demonstrated commercially in early 1980's [26]. This is because making such a device is challenging due to complex system difficulties stemming from frequent power failures, listed below.

Challenge 1: Unpredictable Energy Harvesting. Environmental conditions change, this is exacerbated by mobile gaming. When players move from place to place, most forms of ambient energy change drastically (for instance, by moving from sun to shade), or increasing distance from a radio frequency power source. Without a more predictable source of power, it is hard to envision being able to play continuously without a battery.

和软件，使其具备能量感知能力，并对变化的能量情况做出反应，以减轻频繁电源故障带来的问题。具体而言，我们设计了一种技术，以创建最小的存档游戏，这些游戏可以在电源故障之前快速创建、更新并保存到非易失性存储器中，然后在电源恢复后迅速恢复。与传统游戏系统中看到的存档游戏不同，这些存档捕获了系统的整个状态，因此玩家可以从断电的确切点恢复；例如，在平台游戏中跳跃的中间；尽管设备完全失去电力，仍然可以做到这一点。

贡献。在本文中，我们提出了一种实用的、可用的移动游戏设备，能源感知游戏（简称 *ENGAGE*）。迄今为止，这是首次实现无电池的真实平台的完整系统仿真，也是第一个间歇供电的互动游戏平台。我们的贡献如下：

(1) 我们引入了间歇供电、无电池移动游戏的概念；
(2) 我们开发了一种针对互动显示驱动设备的故障弹性、内存高效、快速的全系统存档游戏的方法。基于跟踪内存变化的补丁概念，使用了一种及时差异检查点方案；
(3) 我们开发了一个硬件平台作为参考实现，具有新颖的多输入架构，用于从按钮按压和阳光中收集能量。该设备还可以支持任何基于互动的系统（不一定是游戏）；
(4) 作为对无电池游戏承诺的压力测试和示范性练习，我们使用这些系统和硬件开发一个完整的任天堂游戏男孩模拟器，该模拟器能够在电力故障的情况下播放未修改的游戏男孩游戏。

2 个挑战

个人手持游戏在过去三十年中为数亿人带来了娱乐和乐趣。在新冠疫情期间，当许多人处于封锁状态时，游戏是减少压力和无聊的活动之一 [116, 118, 127]。本工作的目标是开发无电池移动游戏的系统和硬件基础。这有两个原因：（i）增强了一个无需充电或插电的平台的可用性和可用性——使该平台对普通用户更方便，对每个人更易于获取，以及（ii）对替代和可持续娱乐形式的需求——这是对各种行业联盟的致敬，例如为地球而玩 [108]，旨在减少游戏行业的生态影响。无电池手持游戏机减少了生态成本和失望，因为它始终可以随时拿起并玩耍而无需充电。

对无电池智能设备的众多探索回应了对可持续/碳中和电子设备交互和电子设计与计算的呼声 [12, 65, 67, 85, 145] 同时为“后崩溃社会”准备人机交互电子设备 [131]。其他工作开发了核心系统 [44]、硬件 [23, 28] 和编程语言 [82, 150]，专注于解决由能量采集和无电池操作引起的间歇性计算问题，在这种情况下，频繁的电力故障阻止程序完成任务（见图2）。在所有情况下，电子设备由从环境中采集的能量供电 [107]，并存储在能量密度和体积远小于电池的（超）电容器中。

尚未有人探讨移动手持娱乐的问题，超越了在1980年代早期商业展示的简单形式的无电池游戏设备 [26]。这是因为制造这样的设备具有挑战性，主要由于频繁的电力故障带来的复杂系统困难，具体如下。

挑战 1：不可预测的能量采集。环境条件变化，这在移动游戏中尤为明显。当玩家从一个地方移动到另一个地方时，大多数形式的环境能量会发生剧烈变化（例如，从阳光下移动到阴影中），或者与无线电频率电源的距离增加。没有更可预测的电源，很难设想能够在没有电池的情况下持续游戏。

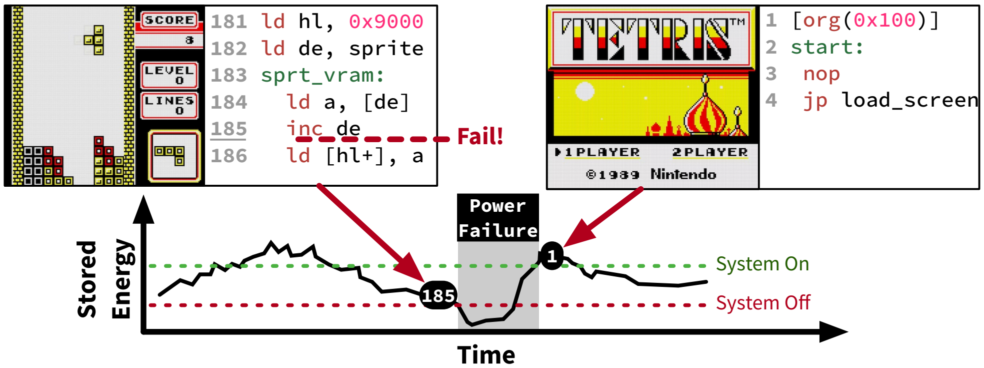


Fig. 2. Dynamic energy harvesting causes voltage fluctuations which cause frequent power failures. Shown is what would typically happen if a battery was removed from a Game Boy and replaced with solar panels. The game would play until energy is lost (i.e. at line 185) and then restart at the loading screen. *Intermittent computing techniques* seek to make it such that after the power failure, line 186 is then executed proceeding from the exact system state as before the failure.

Challenge 2: Keeping Track of System/Game State. Maintaining state of computation—let alone game state—through power failures from intermittent harvested energy is hard [77, 89]. Many software frameworks that support computation progress despite these power failures exist, saving state in non-volatile memory like FRAM and then restoring state after power resumes (see Figure 2), such as TICS [68], TotalRecall [144], and many others. Most systems trade memory efficiency for performance, this approach is opposite of that needed for gaming, where a display buffer and numerous sprites and large game state variables must be saved, requiring high memory efficiency.

Challenge 3: Enormous Variability of Games. These previous issues are compounded by the huge variability of games, both in terms of memory size, number of sprites, actions, difficulty, and even number of button presses per second. Each game is unique, and could pose difficulties when creating a general battery-free solution.

Challenge 4: Gaming's High Computational Load. To date, no full system emulation of any complex system has been attempted on battery-free, intermittently computing devices. Games and gaming platforms require more performant processors even when running natively—when running in emulation, this is compounded. All existing popular runtimes for intermittent computing are based on Texas Instrument's mixed-memory MSP430 MCU [130], which is order of magnitudes slower than the fastest ARM MCU on the market. To meet the high computational load of games, a practical runtime for ARM microconrollers must first be built.

Challenge 5: Capturing User Actions. Playing a video game means a system needs to be highly reactive: button presses post immediate updates to a screen. But *none of the existing battery-free interactive devices demonstrated this level of interactive complexity* with continuously reacting buttons and instantly-refreshing screen. Only simple touch button interfaces that do not need constant pressing for interaction are demonstrated with electronic screens that usually present static content not informed by user actions. Guaranteeing reactivity with unpredictable energy is difficult.

Challenge 6: Realistic Demonstration. The over-arching goal is to play a real, unmodified, video game on a battery-free console that everyone around the world knows (like Tetris)—in other words to be able to execute preexisting game code (or any existing code for that matter), *not to design a custom game* only to demonstrate the

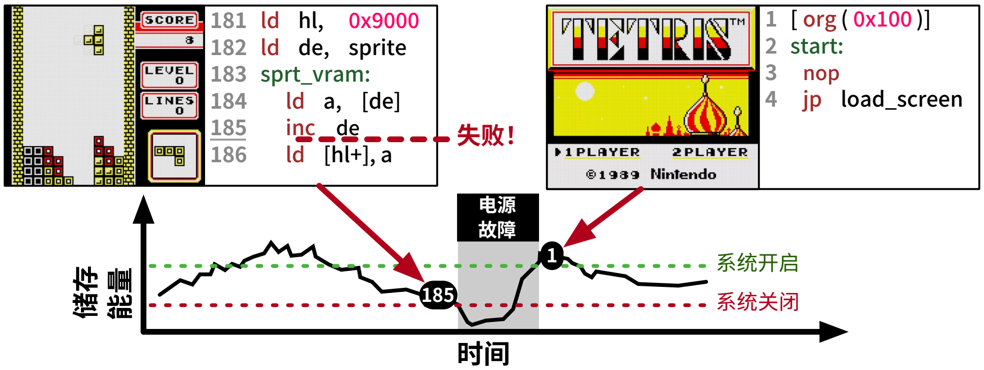


图 2.动态能量收集会导致电压波动，从而导致频繁的电源故障。显示的是如果从游戏男孩中移除电池并用太阳能电池板替换时通常会发生的情况。游戏将持续进行，直到能量耗尽（即在第 185行），然后在加载屏幕处重新启动。间歇计算技术旨在使得在电源故障后，第 186行能够从故障前的确切系统状态继续执行。

挑战 2：跟踪系统/游戏状态。在间歇性收集的能量导致的电源故障中，保持计算状态——更不用说游戏状态——是困难的 [77, 89]。许多支持计算在这些电源故障中持续进行的软件框架存在，它们将状态保存在非易失性存储器中，如FRAM，然后在电源恢复后恢复状态（见图2），例如TICS [68]、TotalRecall [144]以及许多其他框架。
大多数系统在性能与内存效率之间进行权衡，而这种方法与游戏所需的相反，因为必须保存显示缓冲区、众多精灵和大型游戏状态变量，这需要高内存效率。

挑战 3：游戏的巨大变异性。这些先前的问题因游戏的巨大变异性而加剧，包括内存大小、精灵数量、动作、难度，甚至每秒按钮按压次数。每个游戏都是独特的，这可能在创建通用的无电池解决方案时带来困难。

挑战 4：游戏的高计算负载。迄今为止，尚未在无电池、间歇计算设备上尝试对任何复杂系统进行完整系统仿真。游戏和游戏平台即使在本地运行时也需要更高性能的处理器——在仿真中，这种需求更为严重。所有现有的间歇计算流行运行时都基于德州仪器的混合存储器 MSP430 微控制器 [130]，其速度比市场上最快的 ARM 微控制器慢几个数量级。为了满足游戏的高计算负载，必须首先构建一个适用于 ARM 微控制器的实用运行时。

挑战 5：捕捉用户操作。玩视频游戏意味着系统需要高度响应：按钮按下后立即更新屏幕。但是现有的无电池交互设备并未展示出这种级别的交互复杂性具有持续反应的按钮和即时刷新的屏幕。仅展示了简单的触摸按钮接口，这些接口不需要持续按压即可进行交互，通常呈现的电子屏幕内容是静态的，并未受到用户操作的影响。在不可预测的能量下保证反应性是困难的。

挑战 6：现实演示。总体目标是在一个无电池控制台上玩一个全球皆知的真实未修改视频游戏（如俄罗斯方块）——换句话说，能够执行现有的游戏代码（或任何现有代码），而不是设计一个自定义游戏only to demonstrate the

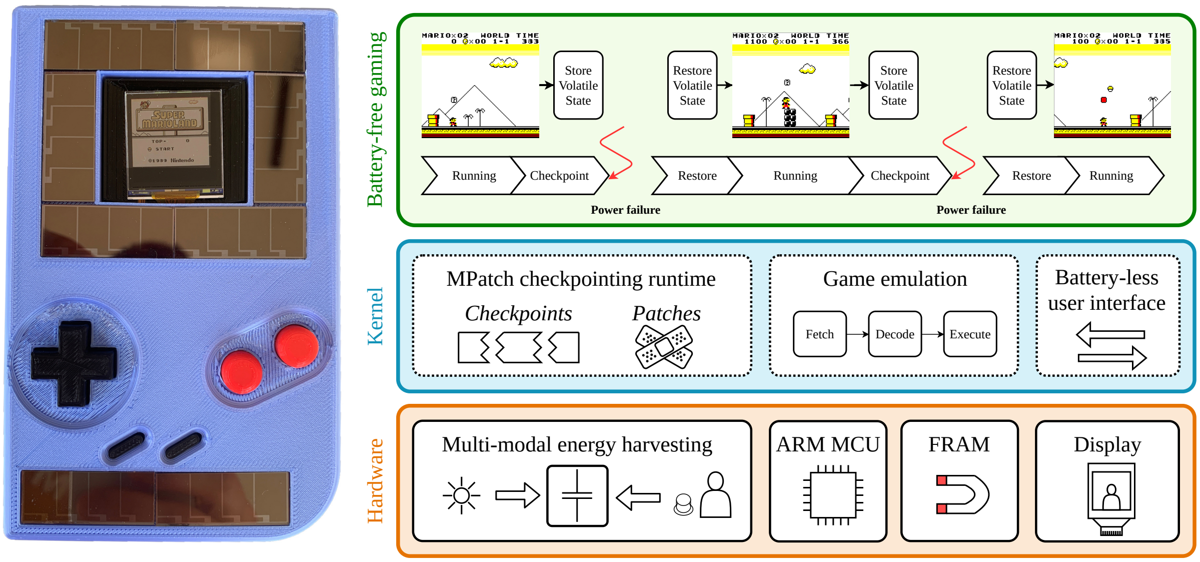


Fig. 3. ENGAGE hardware platform (left) and its internal architecture (right).

potential of battery-free gaming (we refer to extensive survey on this topic in Section 7). This could be possible only when all above challenges are addressed.

To tackle above *Challenge 1–6* we took one of the most popular gaming consoles of all-time [98]—original 8 bit Nintendo Game Boy [24, 99]—and redesigned its hardware-software, powering gameplay from the solar panels and button presses of the user, building the first ARM based intermittent computing hadware and runtime system, and doing the first full system emulation of a real world platform (Nintendo Game Boy) with intermittent computing techniques.

3 BATTERY-FREE HANDHELD GAMING

We designed the *Energy Aware Gaming* (ENGAGE) platform as a proof by demonstration that the discussed challenges could be overcome. The design and architecture of the ENGAGE platforms is shown in Figure 3. The *ENGAGE hardware* is the size and form factor of a Nintendo Game Boy, it is built around (i) user input via mechanical energy harvesting buttons (on the A, B, and D-Pad of the original Game Boy), (ii) a display, (iii) a slot for Game Boy game cartridges to be inserted, and (iv) energy harvesting circuitry from solar cells and the buttons which store energy in a small internal capacitor. The *ENGAGE kernel* consists of (i) a patch-based differential checkpointing system (denoted as MPatch) which handles low level memory movement and automatically saves and restores state of the entire system by efficiently moving necessary data to non-volatile memory (FRAM) and back (SRAM), and (ii) an extensively rewritten full-system Nintendo Game Boy emulator, which can run unmodified Game Boy games. ENGAGE is the first full system emulation, and the first gaming platform built for battery-free, energy harvesting, intermittently powered computing devices.

Usage and Impact. We intend to release the hardware designs, firmware and software as open-source, living repositories on Github [1]. We target a broad audience with our platform. Our battery-free platform (i) must be *open-source*—allowing hobbyists to expand it and improve it, including developing bespoke gaming libraries

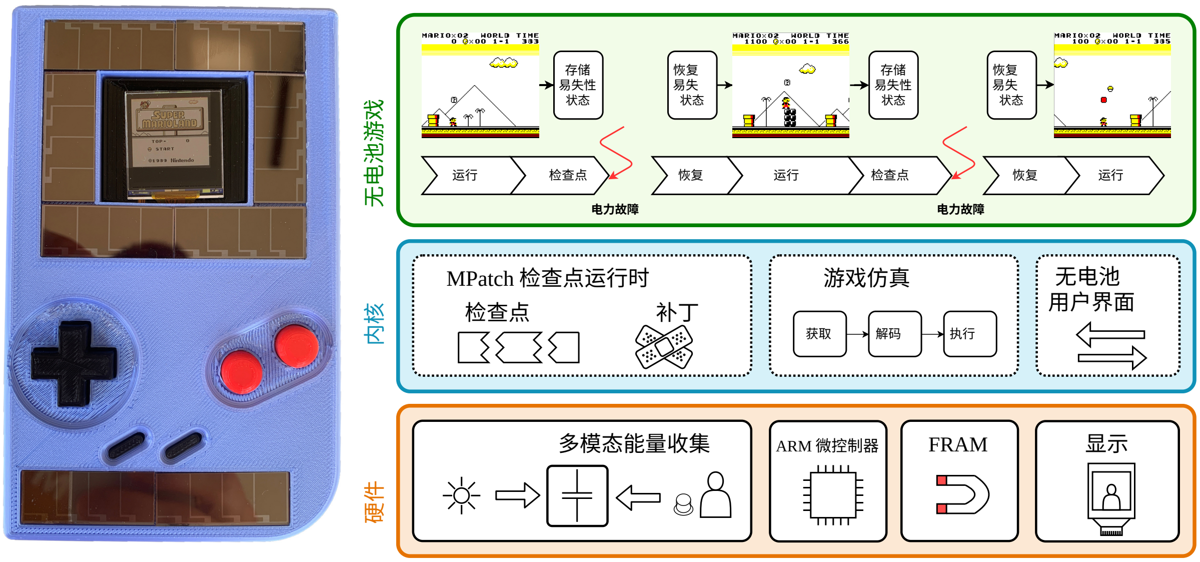


图 3.ENGAGE 硬件平台（左）及其内部架构（右）。

无电池游戏的潜力（我们在第 7 节中提到对此主题的广泛调查）。只有在解决上述所有挑战后，这才有可能。

为了应对上述挑战 *1–6*，我们选择了有史以来最受欢迎的游戏主机之一 [98]——原版 8 位任天堂游戏男孩 [24, 99]——并重新设计了其硬件-软件，利用太阳能电池板和用户的按钮按压为游戏供电，构建了第一个基于 ARM 的间歇计算硬件和运行时系统，并首次对真实世界平台（任天堂游戏男孩）进行了完整的系统仿真，采用间歇计算技术。

3 无电池手持游戏

我们设计了能源感知游戏（ENGAGE）平台，作为一个演示证明所讨论的挑战可以被克服。ENGAGE平台的设计和架构如图3所示。

*ENGAGE*硬件的大小和形状与任天堂游戏男孩相同，构建围绕（i）通过机械能量收集按钮（原始游戏男孩的*A*、B和十字键）进行用户输入，（ii）一个显示屏，（iii）一个插槽用于插入游戏男孩游戏卡带，以及（iv）来自太阳能电池和按钮的能量收集电路，这些电路将能量存储在一个小型内部电容器中。*ENGAGE*内核由（i）一个基于补丁的差分检查点系统（称为MP*atch*）组成，该系统处理低级内存移动，并通过有效地将必要数据移动到非易失性存储器（FRAM）并返回（SRAM）来自动保存和恢复整个系统的状态，以及（ii）一个 extensively 重写的全系统任天堂游戏男孩仿真器，可以运行未修改的游戏男孩游戏。*ENGAGE*是第一个完整系统仿真，也是第一个为无电池、能量收集、间歇性供电计算设备构建的游戏平台。

使用与影响。我们打算将硬件设计、固件和软件作为开源项目，在Github上发布，作为活跃的代码库[1]。我们的平台目标是广泛的受众。我们的无电池平台(i)必须是开源—允许爱好者扩展和改进它，包括开发定制的游戏库。

completely separate from the Game Boy emulator, (ii) have comparable *gameplay and feel* as the original Nintendo Game Boy, and (iii) be able to play any popular retro game.

3.1 Key Ideas

Existing handheld gaming devices rely on large batteries because they need continuous high power to support high compute load, energy cost, and reactivity. We want to enable playing retro 8 bit console games, such as *Tetris* and *Super Mario Land*, on a battery-free console that is similar in user interface and gameplay to the original Nintendo Game Boy. Removing the battery and only using harvested energy causes intermittent operation, which leads to the challenges discussed in Section 2. The ENGAGE platform design navigates these challenges based on four key ideas.

Gather Energy from Gaming Actions and the Environment. ENGAGE harvests energy from button pressing/mashing and solar panels facing towards the player. As opposed to other techniques that rely on nearby dedicated wireless power generation, this approach allows for truly mobile gaming; anywhere a player can find light to see the screen and press/mash buttons. By pairing energy generation with the game actions, and the natural environment where gaming happens, ENGAGE overcomes challenges stemming from unpredictability of energy harvesting, and also ensures that energy is more likely to be available when a user initiates an action (since actions generate energy). This *addresses* **Challenge 1** and **Challenge 5**.

Track and Checkpoint Minimal State at the System Level. We must handle intermittent power failures to maintain state of play. Unfortunately, in games large amounts of memory is moved back and forth to the display, often in the form of sprites, with computation happening in between. Naively checkpointing the entire system state would be impractical, significantly increasing latency of operation. We note that while large memory movements happen, the changes in these memories are often small, meaning we can reduce checkpoints to only the changed memory, save that state just in time before a power failure, and then restore that state and resume game play. This *addresses* **Challenge 2**, **Challenge 3** and **Challenge 4**.

Use Processor Emulation to Play Retro Games. While ENGAGE could be used for custom gaming libraries made specifically for intermittent operation, the more challenging and interesting problem is full system emulation enabling the play of the thousands of existing games, and even home-brewed games. This also allows us to explore and understand the variability of real world games. This *addresses* **Challenge 3** and **Challenge 6**.

Speedup Intermittent Computing. We embrace ultra low powered, high performance ARM Cortex microcontrollers, and external FRAM memory to speed up computation. While a seemingly trivial technology advancement, with this approach we increase compute speed but increase our I/O burden for checkpointing, as the traditional MSP430 FRAM-enabled MCUs have internal FRAM memory accessible at CPU speeds. This is a different tradeoff space than any other intermittent hardware platform [23, 28, 43]. This *addresses* **Challenge 2** and **Challenge 4**.

3.2 ENGAGE Full System Nintendo Game Boy Emulator

A key part of our approach is running a full system emulation on ENGAGE hardware. To be able to run Nintendo Game Boy games an emulator is used to emulate the instruction set of the Game Boy processor, i.e. 8 bit 4.19 MHz custom-built Sharp LR35902 MCU with a processor closely based on the Z80 instruction set [24]. An emulator reads bitcode instructions and executes them in native code, mimicking the emulated CPU as closely as possible to ensure it executes in an identical fashion to the emulated CPU. With the restrictions of battery-free systems additional scenarios are introduced that normally do not exist, such as the loss of power while running a game and then attempting to restore the system to the state it lost power. Additionally emulation efficiency is of critical importance in regards of power consumption.

与游戏男孩仿真器完全分开，(ii)具有与原任天堂游戏男孩相当的游戏体验和感觉，以及(iii)能够玩任何流行的复古游戏。

3.1 关键思想

现有的手持游戏设备依赖于大型电池，因为它们需要持续的高功率以支持高计算负载、能量成本和反应性。我们希望能够在一个无电池的控制台上玩复古8位控制台游戏，如俄罗斯方块和超级马里奥乐园，该控制台在用户界面和游戏玩法上与原任天堂游戏男孩相似。去掉电池并仅使用收集的能量会导致间歇性操作，这导致了在第2节中讨论的挑战。ENGAGE平台的设计基于四个关键理念来应对这些挑战。

从游戏动作和环境中收集能量。ENGAGE通过按键/猛击和面向玩家的太阳能电池板收集能量。与依赖附近专用无线电源生成的其他技术不同，这种方法允许真正的移动游戏；玩家可以在任何有光线可以看到屏幕和按键/猛击的地方进行游戏。通过将能量生成与游戏动作以及游戏发生的自然环境相结合，ENGAGE克服了能量收集的不确定性带来的挑战，并确保在用户发起动作时更有可能获得能量（因为动作会产生能量）。这解决了挑战**1**和挑战**5**。

在系统级别跟踪和检查最小状态。我们必须处理间歇性的电源故障以维持游戏状态。不幸的是，在游戏中，大量内存在显示器之间来回移动，通常以精灵的形式出现，并且在此之间进行计算。天真地检查整个系统状态是不切实际的，这将显著增加操作的延迟。我们注意到，在大规模内存移动发生时，这些内存中的变化通常很小，这意味着我们可以将检查点减少到仅仅是变化的内存，在电力故障之前及时保存该状态，然后恢复该状态并继续游戏。这解决了挑战 **2**，挑战 **3**和挑战 **4**。

使用处理器仿真来玩复古游戏。虽然 ENGAGE 可以用于专门为间歇操作制作的自定义游戏库，但更具挑战性和趣味性的问题是完整系统仿真，使得可以玩成千上万的现有游戏，甚至是自制游戏。这也使我们能够探索和理解现实世界游戏的变异性。这解决了挑战 **3**和挑战 **6**。

加速间歇计算。我们采用超低功耗、高性能的 ARM Cortex 微控制器和外部 FRAM 内存来加速计算。虽然这看似是一个微不足道的技术进步，但通过这种方法，我们提高了计算速度，但增加了检查点的I/O负担，因为传统的MSP430 FRAM启用的微控制器具有可在CPU速度下访问的内部FRAM存储器。这与任何其他间歇性硬件平台的权衡空间是不同的 [23, 28, 43]。这解决了挑战 **2**和挑战 **4**。

3.2 ENGAGE 全系统任天堂游戏男孩仿真器

我们方法的一个关键部分是在ENGAGE硬件上运行全系统仿真。为了能够运行任天堂游戏男孩游戏，使用仿真器来模拟游戏男孩处理器的指令集，即8位4.19 MHz定制的夏普LR35902微控制器，其处理器与Z80指令集密切相关 [24]。仿真器读取位代码指令并以本地代码执行它们，尽可能接近地模拟被仿真的CPU，以确保其以与被仿真的CPU相同的方式执行。由于无电池系统的限制，出现了通常不存在的额外场景，例如在运行游戏时失去电源，然后尝试将系统恢复到失去电源时的状态。此外，仿真效率在功耗方面至关重要。

Table 1. Measurement statistics of all button pressing during a regular Game Boy game for four example games, showing variation between games depending on the type of game. Data was extracted by logging key presses during game play on a Game Boy emulator running on a stationary personal computer. Three similar three-minute playthroughs are averaged in the presented results.

Game name	**Time between button presses (second)**			**Button presses per second**
	Max	*Mean*	*Variance*	
Tetris	3.230	0.508	0.169	1.981
Space Invaders	3.542	0.372	0.129	2.715
Super Mario Land	12.46	0.652	1.091	1.543
Bomberman	7.534	0.765	0.762	1.313

The emulator allocates non-volatile and volatile Game Boy game memory within the memory space of ENGAGE, removing the need to keep cartridges continuously powered. Only upon loading a new game is the cartridge interface used to retrieve the non-volatile game data.

The process of emulating also requires emulation of the Game Boy I/O for the user to be able to interact with the device, most importantly the buttons and the screen. Changing behavior regarding interaction with the I/O might have an influence on the user experience and interaction with the device.

Emulating Button I/O. As energy can be harvested from the press of a button, the frequency of button presses determines the amount of energy generated. This button press frequency is game dependent: in-game-cut-scenes usually require no game-user interaction through the buttons compared to games like *Space Invaders* where buttons are continuously pressed. As a proof, in Table 1 we present statistics about the time between button presses in four popular Nintendo Game Boy games. The maximum time varies greatly depending on the presence of cut-scenes in the game. *Tetris* and *Space Invaders* have few, or short, cut scenes, and thus have a lower maximum time between presses and lower variance. On the other hand, *Super Mario Land* has an animation upon death and at the end of a level, and *Bomberman* has several cut-scenes, hence the higher maximum time between between presses and greater inter-press time variance.

This simple experiment shows that the maximum time between button presses directly pertains to the required energy buffer size, where button mashing games could run on smaller energy buffers increasing the reactivity of the platform by reducing the required charge time.

By changing emulator behavior of handling buttons more energy can be generated. To generate more energy we prevented the holding of buttons. For example, in *Super Mario Land* game it is common to hold the right button to keep moving, but in *Space Invaders* this is less common. This results in a trade-off space between energy generation and changes to the user interaction with Game Boy[1]. Finding the optimum between energy harvesting without changing the user interaction is therefore game specific. In Section 4 we further elaborate on our design choices regarding button emulation.

Emulating Screen Writes. The original Nintendo Game Boy does not employ a frame buffer. Instead, it uses a tile-based approach where tiles are rendered in a CRT-like fashion on the screen to save memory. This means when power to the system is lost, the state of the screen can not be restored since knowledge of the full screen state is not maintained. To combat this we employ a frame buffer and map the Game Boy tile-based rendering

[1]We note that the original Game Boy handles I/O through polling of I/O registers, meaning that every game can have a different handling of the I/O all together, since the Game Boy directly executes machine code from the game.

表1。在四个示例游戏中，常规游戏男孩游戏期间所有按钮按压的测量统计，显示不同类型游戏之间的变化。数据通过在运行于固定个人计算机上的游戏男孩仿真器中记录游戏过程中的按键按压而提取。在所呈现的结果中，三个相似的三分钟游戏过程的平均值。

游戏名称	按钮按压间隔（秒）			每秒按钮按压次数
	最大值	平均值	方差	
俄罗斯方块	3.230	0.508	0.169	1.981
太空入侵者	3.542	0.372	0.129	2.715
超级马里奥乐园	12.46	0.652	1.091	1.543
炸弹人	7.534	0.765	0.762	1.313

仿真器在ENGAGE的内存空间中分配非易失性和易失性游戏男孩游戏内存，消除了持续为卡带供电的需求。仅在加载新游戏时，卡带接口才用于检索非易失性游戏数据。

模拟过程还需要模拟游戏男孩的输入/输出，以便用户能够与设备进行交互，最重要的是按钮和屏幕。关于与输入/输出交互的行为变化可能会影响用户体验和与设备的交互。

模拟按钮输入/输出。由于可以从按钮按压中收集能量，因此按钮按压的频率决定了产生的能量量。这种按钮按压频率依赖于游戏：游戏中的过场动画通常不需要用户通过按钮进行交互，而像太空入侵者这样的游戏则需要持续按压按钮。作为证明，在表1中，我们展示了四款流行任天堂游戏男孩游戏中按钮按压之间的时间统计数据。最大时间因游戏中过场动画的存在而有很大差异。俄罗斯方块和太空入侵者几乎没有或只有短暂的过场动画，因此它们的最大按压时间较低且方差较小。另一方面，超级马里奥乐园在角色死亡和关卡结束时有动画，而炸弹人有多个过场动画，因此按键之间的最大时间更长，按键间隔时间的变化更大。

这个简单的实验表明，按键之间的最大时间直接关系到所需的能量缓冲区大小，其中快速按键游戏可以在较小的能量缓冲区上运行，从而通过减少所需的充电时间来提高平台的反应性。

通过改变仿真器处理按键的行为，可以产生更多的能量。为了产生更多的能量我们防止了按键的持续按压。例如，在超级马里奥乐园游戏中，通常需要按住右键以保持移动，但在太空入侵者中，这种情况较少见。这导致了能量生成与用户与游戏男孩[1]的交互变化之间的权衡空间。因此，在不改变用户交互的情况下找到能量收集的最佳点是特定于游戏的。在第4节中，我们进一步阐述了我们关于按钮仿真的设计选择。

屏幕写入仿真。原始任天堂游戏男孩不使用帧缓冲区。相反，它采用基于图块的方法，在屏幕上以类似CRT的方式渲染图块，以节省内存。这意味着当系统失去电源时，屏幕的状态无法恢复，因为没有维护完整屏幕状态的知识。为了解决这个问题，我们采用了帧缓冲区，并映射游戏男孩的图块渲染。

[1]我们注意到，原始游戏男孩通过轮询I/O寄存器处理I/O，这意味着每个游戏可以有不同的I/O处理方式，因为游戏男孩直接执行来自游戏的机器代码。

into this frame buffer. The buffer is then checkpointed to be able to restore the state of the screen upon power failure.

3.3 Gaming Through Power Failures

ENGAGE is protected from the loss of progress by the custom-designed runtime that guarantees data consistency despite power interrupts. The goal of this runtime is to save (i.e. to checkpoint) the current state of the emulator. This entails the current volatile memory content and the registers of both the host processor and the emulated system. Doing this will allow the system to continue execution from this point as if a power failure never happened.

There are multiple intermittent runtime systems (all of them are summarized in Section 7) which can be broadly divided into two classes: (i) those that use a *special (C program) code instrumentation* to guarantee correctness of computation despite power interrupts and (ii) those that use a *special version of the checkpointing*, of which a subset is designed for systems that use volatile memory—such as SRAM—as their main memory, and use a separate non-volatile memory that contains the checkpointed data. While designing ENGAGE we chose to use a checkpoint based system to allow emulation of arbitrary game code. We did not consider task-based runtimes simply because they are too complex to comprehend by a programmer and more difficult to design than a checkpoint-based system, see related discussion on this topic in [68]. But first and foremost, task-based system cannot execute a binary (machine) code, which ENGAGE is mostly executing.

The main requirement for ENGAGE is responsiveness, hence the checkpointing system needs to be as light-weight as possible. Naturally all of the checkpoint systems have some overhead, so when searching for a good solution we would like to minimize checkpoint size as much as possible—resulting in minimum overhead from data restoration. Checkpointing the entire system state, including game, and emulator, would be impossible. One core idea, proposed first by the DICE runtime [3], is *to checkpoint only parts of device memory that have been changed since the last checkpoint*. To check whether this idea applies to battery-free handheld gaming system we have performed a simple experiment. For four example Nintendo Game Boy games: (i) *Tetris*, (ii) *Space Invaders*, (iii) *Super Mario Land*, and (iii) *Bomberman*, we have measured to which memory regions of an MCU each game was writing to during one minute of game play. The result is presented in Figure 4. Indeed, we see that memory writes are very unevenly distributed for each game, hinting that such approach, which we broadly denote as *differential* checkpointing, is well suited for our ENGAGE needs.

The checkpoint runtimes, including differential ones, can be further divided into two unique classes: (i) *corruptible* and (ii) *incorruptible*.

- **Corruptible Checkpoint:** Such systems copy the current state of the MCU (memory, registers, etc.) to a predetermined location in non-volatile memory. This location is the same every time, as this eases the runtime development and reduces the non-volatile memory requirements. However, it is required that a checkpoint operation must guarantee to complete, otherwise part of the previous checkpoint may be overwritten with the current checkpoint[2]. Often these corruptible runtimes include a check whether a checkpoint was completed successfully, otherwise they start the program execution from the beginning. Such systems require exact prediction of the energy (required to perform a checkpoint) and the energy currently consumed by the complete system (to be able to guarantee that a checkpoint is only performed when its completion can be guaranteed). Such a requirement is *unrealistic* for a computing platform, such as ENGAGE, that includes many peripherals and components all connected to the same energy buffer, as correctly predicting the required energy—even the CPU alone—is difficult;

[2]If the system where to run out of power during the creation of a checkpoint, with the next checkpoint restoration a corrupt state will be restored leading to undefined behaviour—thus to a corrupt system.

到这个帧缓冲区。然后将缓冲区检查点化，以便在电源故障时能够恢复屏幕状态。

3.3 通过电源故障进行游戏

ENGAGE通过定制设计的运行时保护进度不丢失，该运行时保证数据一致性，尽管存在电源中断。该运行时的目标是保存（即检查点）仿真器的当前状态。
这涉及到当前易失性存储器的内容以及主处理器和仿真系统的寄存器。这样做将允许系统从这一点继续执行，就好像电力故障从未发生过。

有多种间歇性运行时系统（所有这些在第7节中总结），可以大致分为两类：（i）那些使用特殊（*C*程序）代码插装以保证计算的正确性，尽管存在电源中断，以及（ii）那些使用特殊版本的检查点技术，其中一个子集是为使用易失性存储器（如SRAM）作为主存储器的系统设计的，并使用一个单独的非易失性存储器来存储检查点数据。在设计ENGAGE时，我们选择使用基于检查点的系统，以便能够仿真任意游戏代码。我们没有考虑基于任务的运行时，主要是因为它们对程序员来说过于复杂，且比基于检查点的系统更难设计，相关讨论见[68]。但首先，基于任务的系统无法执行二进制（机器）代码，而ENGAGE主要执行的正是这些代码。

ENGAGE的主要要求是响应性，因此检查点系统需要尽可能轻量化。自然，所有检查点系统都有一定的开销，因此在寻找良好解决方案时，我们希望尽量减少检查点的大小，从而使数据恢复的开销最小化。检查整个系统状态，包括游戏和仿真器，将是不可能的。一个核心思想，最初由DICE运行时提出[3]，是仅检查自上次检查点以来已更改的设备内存部分。为了检查这个想法是否适用于无电池手持游戏系统，我们进行了一个简单的实验。对于四个示例任天堂游戏男孩游戏：(i)俄罗斯方块，(ii)太空入侵者，(iii)超级马里奥乐园，以及 (iv)炸弹人，我们测量了在一分钟游戏过程中每个游戏写入微控制器的内存区域。结果如图4所示。确实，我们看到每个游戏的内存写入非常不均匀分布，这暗示了这种方法（我们广泛称之为差异检查点技术）非常适合我们的ENGAGE需求。

检查点运行时间，包括差异检查点，可以进一步分为两类独特的类别：(i) 可腐蚀的和 (ii)不可腐蚀的。

- 可腐蚀检查点：这样的系统将微控制器的当前状态（内存、寄存器等）复制到非易失性存储器中的预定位置。此位置每次都是相同的，因为这简化了运行时开发并减少了非易失性存储器的需求。然而，要求检查点操作必须保证完成，否则之前的检查点的一部分可能会被当前检查点[2]覆盖。通常，这些易损的运行时包括检查检查点是否成功完成的步骤，否则它们将从头开始执行程序。

  这样的系统需要准确预测执行检查点所需的能量以及当前整个系统消耗的能量（以便能够保证仅在可以保证完成时执行检查点）。这样的要求对于一个计算平台来说是不现实的，例如ENGAGE，它包括许多外设和组件，所有这些都连接到同一个能量缓冲区，因为正确预测所需的能量——即使是仅CPU——也是困难的；

[2]如果系统在创建检查点期间耗尽电源，则在下一个检查点恢复时将恢复到一个损坏的状态，导致未定义的行为——从而导致系统损坏。

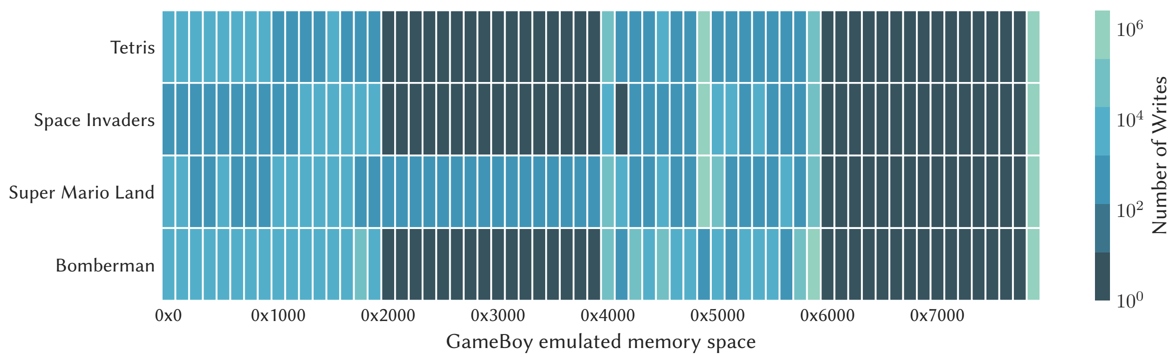


Fig. 4. Memory writes heat map of four popular 8 bit Game Boy games for one minute of play. Writes tend to cluster in a few large regions; tracking and checkpointing these regions would allow for performant intermittent execution. Note the log-scale of the number of writes.

- **Incorruptible Checkpoint:** Such systems take a different approach: at all times they *guarantee* that there is a valid checkpoint which can be restored. This means that a new checkpoint will never overwrite part of the previous checkpoint in non-volatile memory. Such guarantee is often implemented through double-buffering.

As of now, there are *no known incorruptible differential checkpoint systems*, and just one corruptible differential checkpoint system, DICE [3], refer also to Table 2 where existing intermittent runtimes are qualitatively compared from ENGAGE requirements point of view. Therefore, to realize a working ENGAGE we developed a new checkpointing runtime, denoted as MPatch, that performs incorruptible differential checkpoints. The proposed runtime is aided by a new concept of *patch checkpointing*, discussed below.

MPatch—a Patch Checkpointing Intermittent Runtime. Memory is constantly being modified during the execution of a program. However, as Figure 4 clearly illustrates, it is unlikely that during an on-period of any intermittently powered embedded system, including ENGAGE, all memory is modified. Therefore, when creating a checkpoint containing all the known or active memory regions of the system, one will inevitably copy memory locations that have not changed since the last checkpoint.

It is thus desirable to copy as little of the (embedded) system state as possible while keeping the checkpointing incorruptible. The most fundamental method to do this efficiently is to track which memory regions have been changed since the last checkpoint, in other words, to see memory modification *difference* in-between checkpoints. As mentioned earlier, the only checkpoint runtime that has employed this form of differential checkpoint so far was DICE [3], see again Table 2. It is, however, difficult to apply the techniques used by DICE while maintaining an incorruptible system (that uses double buffering). Specifically, assuming that only one of the buffers is active, if part of the checkpoint resides in the previous buffer, and yet another checkpoint occurs, then it is impossible to keep the incorruptibility trait with DICE without still copying all checkpoint data between the two buffers. Therefore, to achieve differential checkpointing that is incorruptible, a new system has to be designed, which resulted in MPatch.

MPatch Just-in-Time Checkpoints. As we have shown in Figure 4 not all of the emulator and display memory is written to at every MCU clock cycle. Hence we only checkpoint the modified memory regions, which we denote as *patches*. Then, we monitor the voltage level of the storage capacitor, as in other existing runtimes,

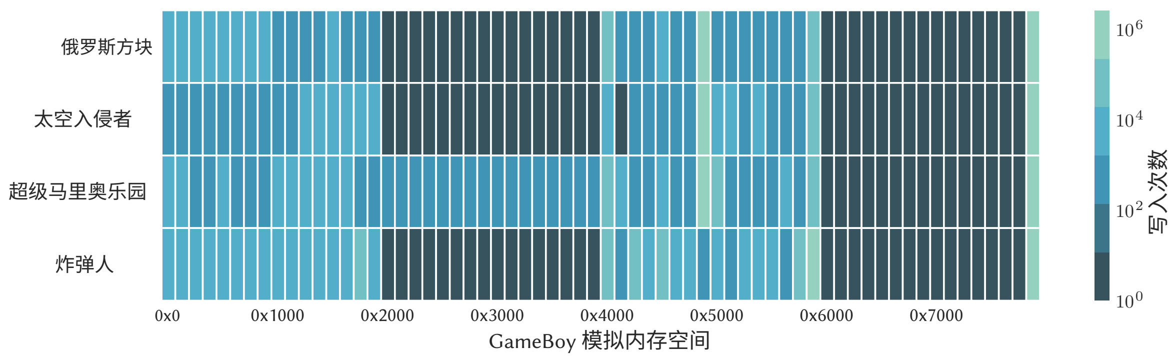


图 4.四款流行的8位游戏男孩游戏在一分钟内的内存写入热图。写入往往集中在几个大区域；跟踪和检查这些区域将允许高效的间歇性执行。注意写入次数的对数尺度。

- 不可腐蚀检查点:这样的系统采取不同的方法：始终保证存在一个可以恢复的有效检查点。这意味着新的检查点永远不会覆盖非易失性存储器中先前检查点的部分内容。这种保证通常通过双缓冲实现。

截至目前，尚无已知的不可腐蚀差分检查点系统，只有一个可腐蚀的差分检查点系统DICE [3]，另请参见表2，其中从ENGAGE要求的角度对现有的间歇性运行时进行了定性比较。因此，为了实现一个可行的ENGAGE，我们开发了一种新的检查点运行时，称为MPatch，它执行不可腐蚀的差分检查点。所提出的运行时得益于一个新的补丁检查点概念，下面将讨论。

MPatch——一个补丁检查点间歇性运行时。在程序执行期间，内存不断被修改。然而，如图4清楚地说明的那样，在任何间歇性供电的嵌入式系统（包括ENGAGE）的开机期间，所有内存都被修改的可能性很小。因此，在创建一个包含系统所有已知或活动内存区域的检查点时，必然会复制自上一个检查点以来未发生变化的内存位置。

因此，在保持检查点不可腐蚀的同时，尽可能少地复制（嵌入式）系统状态是理想的。有效地做到这一点的最基本方法是跟踪自上一个检查点以来已更改的内存区域，换句话说，查看检查点之间的内存修改差异。
如前所述，迄今为止唯一采用这种差异检查点形式的检查点运行时是DICE [3]，再次参见表2。然而，在保持一个不可腐蚀的系统（使用双缓冲）的同时，应用DICE所使用的技术是困难的。具体而言，假设只有一个缓冲区处于活动状态，如果检查点的一部分位于前一个缓冲区中，而又发生另一个检查点，那么在不复制两个缓冲区之间的所有检查点数据的情况下，无法保持DICE的不可腐蚀性特征。

因此，为了实现不可腐蚀的差异检查点技术，必须设计一个新系统，这就是MPatch。

MPatch即时检查点。正如我们在图4中所示，并非所有的仿真器和显示内存在每个微控制器时钟周期内都被写入。因此，我们仅对修改过的内存区域进行检查点记录，我们称之为补丁。然后，我们监测储存电容器的电压水平，正如其他现有运行时所做的那样，

Table 2. Comparison of MPatch with state-of-the-art intermittent checkpointing runtimes.

System	Incorruptible	Differential	Just-in-time	Volatile main memory	ARM support
Mementos [109]	Yes ✓	No ✗	Yes ✓	Yes ✓	Yes ✓
Hibernus++ [8]	No[1]✗	No ✗	Partially —	Yes ✓	No ✗
QuickRecall [57]	No ✗	No ✗	Yes ✓	No ✗	No ✗
Chinchilla [82]	Yes ✓	N/A	No ✗	No ✗	No ✗
Rachet [137]	Yes ✓	No ✗	No ✗	Yes ✓	Yes ✓
HarvOS [11]	No[1]✗	No ✗	Yes ✓	Yes ✓	Yes ✓
TICS [68]	Yes ✓	N/A	No ✗	No ✗	No ✗
TotalRecall [144]	No[1]✗	No ✗	Yes ✓	Yes ✓	No ✗
Elastin [19]	Yes ✓	N/A	No ✗	No ✗	No ✗
WhatsNext [35]	Yes ✓	No ✗	No ✗	Yes ✓	Yes ✓
DICE [3]	No [1]✗	Yes ✓	Yes ✓	Yes ✓	Yes ✓
MPatch	Yes ✓	Yes ✓	Yes ✓	Yes ✓	Yes ✓

[1] These systems require perfect energy prediction to not get corrupted. Any changes in, for example, capacitor size [19], power consumption due to peripheral use, or harvested energy, **will** lead to incorrect predictions and therefore **corruption**.

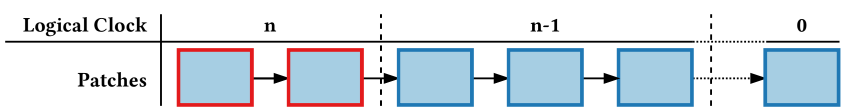


Fig. 5. MPatch stage operation. Patches outlined with red are staged, but not committed. Patches outlined with blue signify committed patches.

e.g. [3, 8, 57, 144] and only checkpoint the state when nearing a power failure—we call this *just-in-time checkpoint*. We purposefully do not perform checkpoints at an interval timer: game players are susceptible to lagging in a game, hence interval-based checkpointing (which introduces frequent fixed-interval delay) is not desirable.

Patch Handling. A patch is a non-volatile copy of a *consecutive* region of volatile memory that has changed since the last successfully created checkpoint. As different memory regions are modified during execution, multiple patches of different memory sections might be required for a complete checkpoint. During the restoration, the most recent patches (in combination with the pre-existing patches) are used to restore the volatile memory to the state it was in during the last checkpoint. By only storing the modified regions the checkpoint time is significantly reduced, as often only a small part of the memory is changed between the two consecutive checkpoints (we will investigate this further in Section 5).

As with traditional checkpoint-based systems that use double-buffering, an atomic variable n, determines which of the two buffers should be used to restore the system in case of a power failure [68, 109]. This variable n is changed—often incremented—to mark the completion of a checkpoint. The requirement on n is that for its increment, $n + 1$, it holds that $(n \bmod 2) \neq (n + 1 \bmod 2)$. MPatch patch management is also built around the atomic variable. However, MPatch extends the function of this variable to act as a *logical clock*, with the additional requirement that $n \neq n + 1$.

We now define three fundamental patch operations (i) *Patch Stage*, (ii) *Patch Commit*, and (iii) *Patch Restore*.

表2。MPatch与最先进的间歇性检查点运行时的比较。

系统	不可腐蚀的	差异	即时	易失性主存	**ARM**支持
纪念品 [109]	是 ✓	否 ✗	是 ✓	是 ✓	是 ✓
Hibernus++ [8]	否[1] ✗	否 ✗	部分 –	是 ✓	否 ✗
快速回忆 [57]	否 ✗	否 ✗	是 ✓	否 ✗	否 ✗
南美洲栗鼠 [82]	是 ✓	不适用	否 ✗	否 ✗	否 ✗
Rachet [137]	是 ✓	否 ✗	否 ✗	是 ✓	是 ✓
HarvOS [11]	否[1] ✗	否 ✗	是 ✓	是 ✓	是 ✓
TICS [68]	是 ✓	不适用	否 ✗	否 ✗	否 ✗
完全回忆 [144]	否[1] ✗	否 ✗	是 ✓	是 ✓	否 ✗
弹性蛋白 [19]	是 ✓	不适用	否 ✗	否 ✗	否 ✗
接下来是什么 [35]	是 ✓	否 ✗	否 ✗	是 ✓	是 ✓
DICE [3]	否 [1] ✗	是 ✓	是 ✓	是 ✓	是 ✓
MPatch	是 ✓	是 ✓	是 ✓	是 ✓	是 ✓

[1]这些系统需要完美的能量预测以避免被破坏。例如，电容器大小的任何变化 [19]、由于外设使用导致的功耗或收集的能量，都将导致不正确的预测，因此数据损坏。

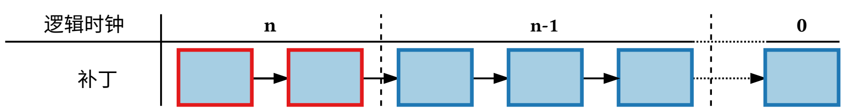


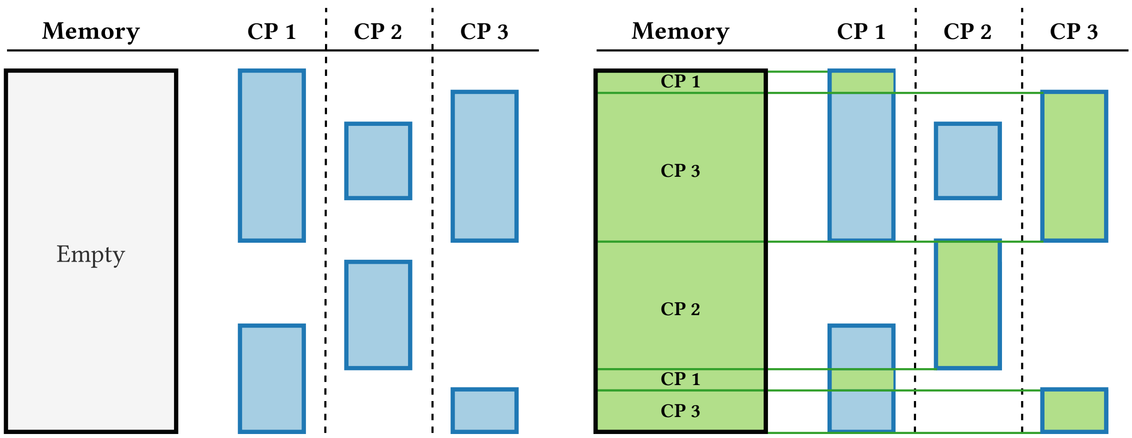
图 5. MPatch 阶段操作。用红色标出的补丁已被暂存，但尚未提交。用蓝色标出的补丁表示已提交的补丁。

例如 [3, 8, 57, 144]，并且仅在接近电力故障时检查状态——我们称之为及时检查点。我们故意不在间隔定时器上执行检查点：游戏玩家容易在游戏中出现延迟，因此基于间隔的检查点技术（引入频繁的固定间隔延迟）并不可取。

补丁处理。补丁是自上次成功创建检查点以来已更改的连续易失性存储器区域的非易失性副本。由于在执行过程中不同的内存区域被修改，可能需要多个不同内存部分的补丁以完成检查点。在恢复过程中，最近的补丁（与现有补丁结合使用）用于将易失性存储器恢复到上一个检查点时的状态。通过仅存储修改过的区域，检查点时间显著减少，因为在两个连续检查点之间，通常只有一小部分内存发生变化（我们将在第5节进一步探讨这一点）。

与使用双缓冲的传统检查点系统一样，一个原子变量 n决定在电力故障时应使用哪个缓冲区来恢复系统 [68, 109]。该变量 n被更改——通常是递增——以标记检查点的完成。对 n的要求是，对于其递增，$n+1$，必须满足 $(n \bmod 2) \neq (n+1 \bmod 2)$。MPatch 补丁管理也围绕原子变量构建。然而，MPatch 扩展了该变量的功能，使其充当逻辑时钟，附加要求是 $n \neq n+1$。

我们现在定义三种基本补丁操作 (i)补丁阶段, (ii)补丁提交, 和 (iii)补丁恢复。



(a) MPatch memory **before** restoration.

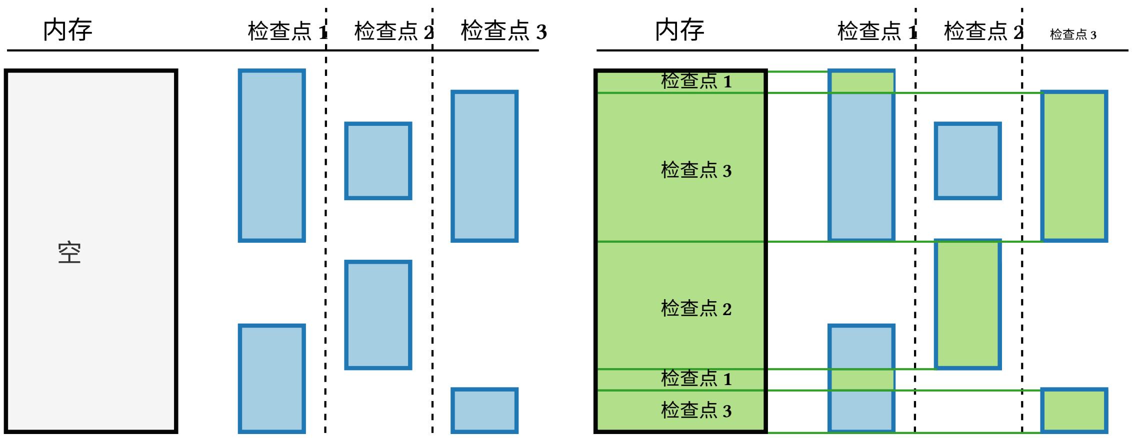
(b) MPatch memory **after** restoration. The restore sequence applies only the parts of patches that are required to reconstruct the memory.

Fig. 6. MPatch patch restore procedure after three successful checkpoints (CP). For the ease of illustration, we assume that the memory is initiated as empty; blue rectangles depict patches that have been successfully committed to non-volatile memory and green rectangles signify the parts of the patches that are applied during restoration.

- *Patch Stage:* When a patch is created, the required amount of non-volatile memory is allocated and the volatile-memory is copied to the patch. Next, the patch is *staged* by signing it with the current logical clock n added to the front of the *patch chain*, i.e. the list of patches, ordered from newest to oldest, that will be applied during restoration. Staged patches are outlined in red color in Figure 5. While a patch is staged it will be discarded if a power failure (and thus a restoration procedure) occurs.
- *Patch Commit:* When the logical clock n is incremented, all previously staged patches will become *committed*. These patches are outlined in blue in Figure 5. Committed patches will be considered during the *patch restore* procedure.
- *Patch Restore:* When ENGAGE inevitably fails due to a lack of energy, it should be restored to the last completed checkpoint. Patches hold copies of consecutive volatile memory regions and are linked together to form the patch chain. This moves the complication of deciding what *part* of the patch to apply, if any, to the restore operation. To reconstruct the state of the most recent checkpoint the (partial) content of multiple patches has to be combined. This reconstruction, due to the implicit ordering in the patch chain, starts from newest to oldest. For each patch, only the parts that were not already applied during the current restore operation are copied to volatile memory, as illustrated in Figure 6. In contrast, for a traditional incorruptible checkpoint runtime, restoring a checkpoint means reading the logical clock n and copying the checkpoint content from the selected buffer to the corresponding volatile memory and registers.

4 ENGAGE IMPLEMENTATION

We proceed with the implementation details of ENGAGE, following from the design description provided in Section 3. All hardware, software and tools, as well as documentation for ENGAGE are publicly available via [1].



(a) MPatch 内存恢复之前。

(b) MPatch 内存恢复之后。恢复序列仅应用重建内存所需的补丁部分。

图 6。MPatch 补丁恢复程序在三个成功的检查点 (CP) 之后。为了便于说明，我们假设内存初始化为空；蓝色矩形表示已成功提交到非易失性存储器的补丁，绿色矩形表示在恢复过程中应用的补丁部分。

- 补丁阶段:当创建补丁时，将分配所需的非易失性存储器，并将易失性存储器复制到补丁中。接下来，补丁通过用当前逻辑时钟 n 签名并添加到补丁链的前面进行分阶段，即按从最新到最旧的顺序排列的补丁列表，这些补丁将在恢复过程中应用。分阶段的补丁在图5中用红色轮廓标出。当补丁处于分阶段状态时，如果发生电力故障（因此发生恢复程序），则将被丢弃。

- 补丁提交:当逻辑时钟 n递增时，所有先前分阶段的补丁将变为已提交。
这些补丁在图5中用蓝色轮廓标出。已提交的补丁将在补丁恢复过程中被考虑。

- 补丁恢复:当ENGAGE因缺乏能量而不可避免地失败时，应恢复到最后完成的检查点。补丁保存连续易失性存储器区域的副本，并链接在一起形成补丁链。这将决定在恢复操作中应用补丁的哪一部分（如果有的话）的复杂性转移到了恢复操作中。为了重建最近检查点的状态，必须结合多个补丁的（部分）内容。由于补丁链中的隐式排序，这一重建过程是从最新到最旧进行的。对于每个补丁，仅将当前恢复操作中尚未应用的部分复制到易失性存储器，如图6所示。相比之下，对于传统的不可腐蚀检查点运行时，恢复检查点意味着读取逻辑时钟n并将检查点内容从选定的缓冲区复制到相应的易失性存储器和寄存器中。

4 实施 ENGAGE

我们将继续实施ENGAGE的详细信息，基于第3节提供的设计描述。所有硬件、软件和工具，以及ENGAGE的文档，均可通过[1]公开获取。

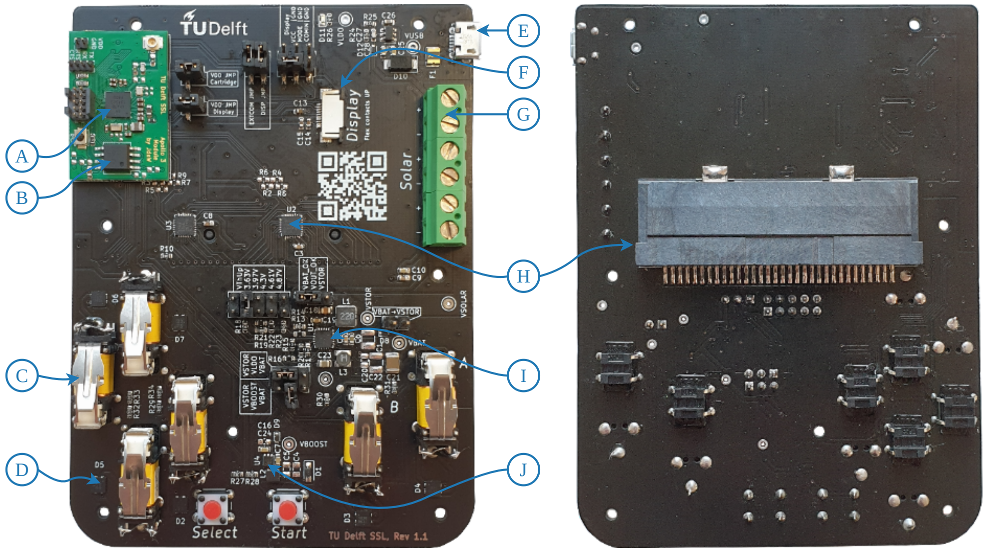


Fig. 7. Energy Aware Gaming (ENGAGE) fabrication details. The main components are: (A) Ambiq Apollo3 Blue ARM Cortex-M4 MCU, (B) Fujitsu MB85RS4MT 512 KB FRAM, (C) ZF AFIG-0007 energy harvesting switch, (D) Semiconductor Components Industries NSR1030QMUTWG low forward voltage diode bridge, (E) micro USB debugging port, (F) display connector, (G) solar panels connector, (H) cartridge interface, (I) Texas Instruments BQ25570 harvester/power management chip, and (J) Texas Instruments TPS61099 boost converter.

4.1 ENGAGE Hardware

We built a handheld, energy harvesting, battery-free hardware platform to enable the development and testing of our approach to battery-free mobile gaming. ENGAGE is built using the following components.

4.1.1 Processing and Memory. Stemming from the requirements (Section 2), for compatibility and popularity reasons, we build our ENGAGE around an ARM MCU architecture. However, none of the ARM architecture MCUs we are aware of contains on-chip fast, byte-addressable non-volatile memory—such as FRAM—serving as a main memory. Only slow and energy-expensive FLASH memory is present. Therefore we equip our battery-free console with external dedicated FRAM. Central to ENGAGE is the *Ambiq Apollo3 Blue ARM Cortex-M4 MCU* operating at a clock frequency of 96 MHz [5], chosen for its good energy efficiency. The Apollo3 runs the Game Boy emulator and MPatch software. External *Fujitsu MB85RS4MT 512 KB FRAM* [34] is connected through SPI to the MCU providing a fast and durable method of non-volatile storage for patch checkpoints, see Figure 7. With the availability of power switches within the MCU, it gates power to the screen and cartridge interface, see Figure 7.

To be able to read game cartridges, a cartridge connector is placed on the back of the ENGAGE platform. The cartridge interfaces with the MCU using *Semtech SX1503 I/O expanders* [120], in this case the extenders also translate the 3 V system voltage to 5 V logic required by the cartridge.

4.1.2 Mixed Source Energy Harvesting. We extract energy from two sources: (i) button presses of a regular Game Boy, using mechanical off-the-shelf button press harvesters, and (ii) a set of solar panels attached to the front of the Game Boy chassis. The selected buttons were *ZF AFIG-0007 energy harvesting switches* [153]. Six of these kinetic harvesting switches are used (see Figure 7) located at the position of the D-pad (four switches) and “A”,

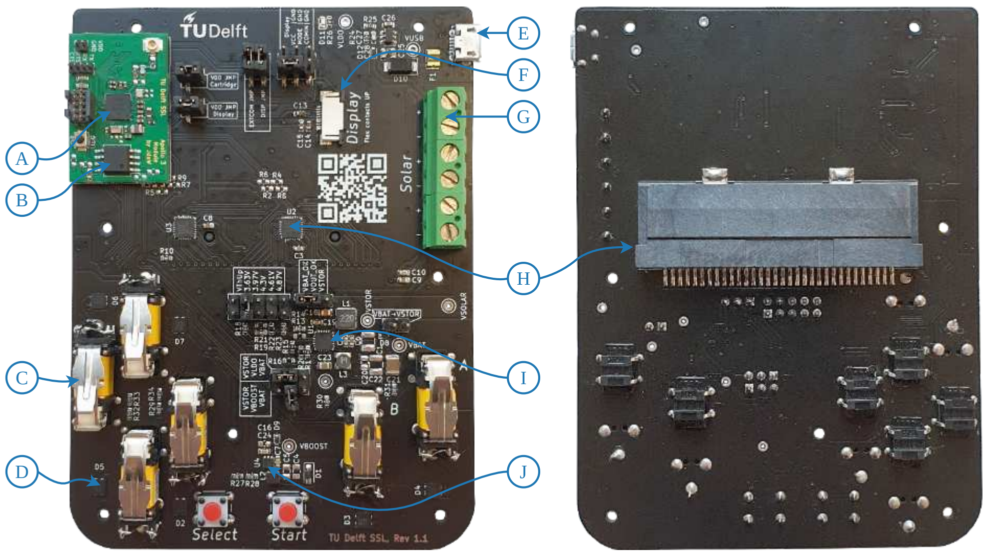


图 7.能源感知游戏 (ENGAGE) 制造细节。主要组件包括：(A) Ambiq Apollo3 Blue ARM Cortex-M4 微控制器，(B) 富士通 MB85RS4MT 512 KB FRAM，(C) ZF AFIG-0007 能量收集开关，(D) 半导体组件工业 NSR1030QMUTWG 低正向电压二极管桥，(E) micro USB 调试端口，(F) 显示连接器，(G) 太阳能电池板连接器，(H) 卡带接口，(I) 德州仪器 BQ25570 收集器/电源管理芯片，以及 (J) 德州仪器 TPS61099 升压转换器。

4.1 ENGAGE 硬件

我们构建了一个手持式、能量收集、无电池的硬件平台，以支持我们对无电池移动游戏的方法的开发和测试。ENGAGE是基于以下组件构建的。

4.1.1 处理与存储。基于需求（第2节），出于兼容性和流行性原因，我们围绕ARM微控制器架构构建我们的ENGAGE。然而，我们所知的ARM架构微控制器中没有任何一个包含片上快速、字节可寻址的非易失性存储器——如FRAM——作为主存储器。仅存在缓慢且耗能的FLASH存储器。因此，我们为无电池控制台配备了外部专用FRAM。ENGAGE的核心是*Ambiq Apollo3 Blue ARM Cortex-M4*微控制器，工作频率为96 *M*Hz [5]，因其良好的能效而被选中。Apollo3运行游戏男孩仿真器和MPatch软件。外部富士通 *MB85RS4MT 512 KB FRAM* [34]通过SPI连接到微控制器，为补丁检查点提供了一种快速且耐用的非易失性存储方法，见图7。

通过在微控制器内提供电源开关，它控制屏幕和卡带接口的电源，见图7。

为了能够读取游戏卡带，卡带连接器被放置在ENGAGE平台的背面。该卡带通过*Semtech SX1503 I/O* 扩展器 [120]与微控制器接口，在这种情况下，扩展器还将3 V系统电压转换为卡带所需的5 V逻辑。

4.1.2 混合源能量收集。我们从两个来源提取能量：(i) 使用现成的机械按钮按压收集器的常规游戏男孩的按钮按压，以及 (ii) 附加在游戏男孩机壳前面的太阳能电池板。所选按钮是*ZF AFIG-0007*能量收集开关 [153]。这六个动能收集开关被使用（见图7），位于十字键的位置（四个开关）和“A”键。

"B" operation buttons (two switches). The energy harvesting switches generate energy by moving a magnet inside a coil. Since both up and downward motion generates energy, the output of the switches is rectified using a *Semiconductor Components Industries NSR1030QMUTWG low forward voltage diode bridge* [119] before being boosted using a *Texas Instruments TPS61099 boost converter* [129] to be stored in a small intermediate energy storage capacitor. When the intermediate energy storage reaches 4 V the system turns on and the buck converter of the power management chip steps down the voltage to 3 V to power the system. Additionally solar energy is harvested from eight small solar panels [103], each measuring 35.0×13.9 mm, affixed on the front of the ENGAGE chassis (see Figure 7).

Harvesting of solar energy is managed by a *Texas Instruments BQ25570 harvester/power management chip* [128], integrating both a buck and a boost converter. The harvester employs a boost converter and maximum power point tracking to harvest the solar energy and stores the harvested energy in the intermediate energy storage capacitor. All energy from energy harvesting is stored in a main 3.3 mF capacitor, chosen to enable even in the worst energy harvesting conditions a few seconds of game play before the system reaches a critical voltage and powers down to harvest more energy.

4.1.3 Ultra-low Power Display. Displaying game content consumes the most energy of any embedded platform. Making energy-efficient display is research on its own, which is beyond the scope of this work. At the same time, we want ENGAGE to be accessible to any hobbyist by being built out of easily available and inexpensive components. Therefore we have relied on super-low power state-of-the-art off-the-shelf commercial display. Note that we have excluded e-ink displays as their refresh rates are too low for a good gaming experience (especially with rapidly changing game states such as *Super Mario Land*).

We have chosen a low power non-back lit reflective *Japan Display LPM013M126A LCD* [56], noting that the Game Boy also does not have a backlight. The LCD measures 26.02×27.82 mm, which means the display is smaller compared to the original Game Boy screen by 47×43 mm. Our chosen display offers a greater resolution of 176×176 compared to 160×144 pixels of the original Nintendo Game Boy and is capable of displaying eight colors compared to the four shades of gray the Game Boy uses. Just like in case of cartridge interface, the MCU gates power to the screen.

4.1.4 Form Factor and Fabrication. For the same gaming feel we encapsulate the electronics of ENGAGE in a 3D-printed chassis reminiscent of the original Game Boy. The only differences are: (i) the removal of sound outlet, as *we do not support sound generation on an intermittent power supply*, (ii) addition of slits to house the solar panels, (iii) slit for the USB port to provide constant power supply to ENGAGE for debugging, and (iv) removal of slits for battery charge cable and an on/off switch—as they are obviously *not needed in a battery-free system.* Since the Apollo3 Blue MCU is only available in BGA packaging, we separated the MCU from the main PCB creating a separate module containing this MCU only—see the green PCB in the top left corner of Figure 7—to reduce the risk of soldering errors during small batch manufacturing. This module is connected through connectors to the main ENGAGE PCB. Complete fabricated PCB, front and back, are shown in Figure 7.

4.2 ENGAGE Emulator Implementation

As many Nintendo Game Boy emulators have already been written we have decided not to build yet another one and relied on the existing emulator implementation that targets a different MCU. Specifically, to run with ENGAGE we extensively modified and rewrote a pre-existing freely-available implementation of original Nintendo Game Boy emulator targeting a STM32F7 MCU [10]. All the modifications to this emulator, enabling to reproduce our work, are part of our open-source repository freely available to download from [1].

ENGAGE Screen Handling. Due to the availability of displaying colours on the chosen display we remapped the default gray-scale colour palette (white, light gray, dark gray and black) of the Game Boy to a colour version

“B”操作按钮（两个开关）。能量收集开关通过在线圈内移动磁铁来产生能量。由于上下运动均可产生能量，因此开关的输出通过半导体元件工业*NSR1030QMUTWG*低正向电压二极管桥 [119]整流，然后使用德州仪器*TPS61099*升压转换器 [129]进行升压，以存储在一个小型中间能量存储电容器中。当中间能量存储达到4 V时，系统开启，电源管理芯片的降压转换器将电压降至3 V以为系统供电。此外，从八个小型太阳能电池板[103]收集太阳能，每个电池板的尺寸为35.0 ×13.9 mm，固定在ENGAGE机壳的前面（见图7）。

太阳能的收集由德州仪器*BQ25570*收集器/电源管理芯片 [128]管理，该芯片集成了降压和升压转换器。该收集器采用升压转换器和最大功率点跟踪技术来收集太阳能，并将收集到的能量存储在中间能量存储电容器中。所有来自能量收集的能量都存储在一个主3.3 mF电容器中，选择该电容器是为了在最糟糕的能量收集条件下仍能实现几秒钟的游戏玩法，直到系统达到临界电压并关闭以收集更多能量。

4.1.3 超低功耗显示 . 显示游戏内容消耗了任何嵌入式平台中最多的能量。
制作节能显示屏本身就是一项研究，超出了本工作的范围。同时，我们希望ENGAGE对任何爱好者都可及，通过使用易于获得且价格低廉的组件来构建。因此，我们依赖于超低功耗的最先进的现成商业显示器。请注意，我们已排除电子墨水显示屏，因为它们的刷新率对于良好的游戏体验来说太低（尤其是在快速变化的游戏状态下，例如超级马里奥乐园）。

我们选择了一款低功耗非背光反射式日本显示器 *LPM013M126A LCD* [56]，并注意到游戏男孩也没有背光。该LCD的尺寸为26.02 ×27.82 mm，这意味着与原始游戏男孩屏幕相比，显示器的尺寸小了47 ×43 mm。我们选择的显示器提供了176 ×176的更高分辨率，而原始任天堂游戏男孩的分辨率为160 ×144像素，并且能够显示八种颜色，而游戏男孩仅使用四种灰度。就像在卡带接口的情况下，微控制器为屏幕提供电源。

4.1.4 形状和制造。为了保持相同的游戏感觉，我们将ENGAGE的电子元件封装在一个3D打印的外壳中，外形类似于原始游戏男孩。唯一的区别是：（i）去除了声音输出，因为我们不支持在间歇性电源下生成声音，（ii）增加了用于容纳太阳能电池板的缝隙，（iii）为USB端口留出缝隙，以便为ENGAGE提供持续的电源以进行调试，以及（iv）去除了电池充电线和开关的缝隙——因为它们在无电池系统中显然不需要。由于Apollo3 Blue微控制器仅以BGA封装形式提供，我们将微控制器与主PCB分离，创建了一个仅包含该微控制器的独立模块——见图7左上角的绿色PCB——以降低小批量生产过程中焊接错误的风险。该模块通过连接器与主ENGAGE PCB连接。完整的PCB，包括正面和背面，如图7所示。

4.2 ENGAGE仿真器实现

由于许多任天堂游戏男孩仿真器已经被编写，我们决定不再构建另一个，而是依赖于针对不同微控制器的现有仿真器实现。具体而言，为了与ENGAGE兼容，我们对一个预先存在的、可自由获取的原始任天堂游戏男孩仿真器实现进行了广泛的修改和重写，该实现针对STM32F7微控制器 [10]。所有对该仿真器的修改，使得能够重现我们的工作，都是我们开放源代码库的一部分，免费提供下载 [1]。

ENGAGE 屏幕处理。由于所选显示器能够显示颜色，我们将游戏男孩的默认灰度调色板（白色、浅灰色、深灰色和黑色）重新映射为彩色版本

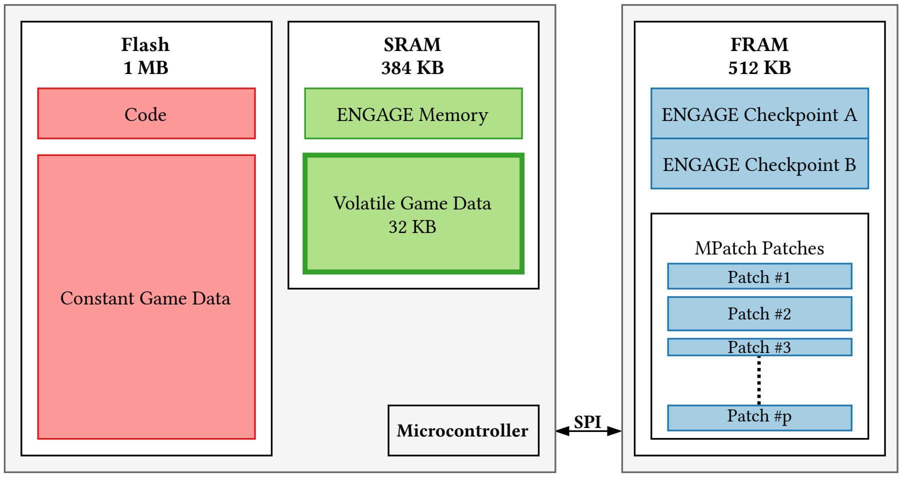


Fig. 8. ENGAGE physical memory structure. Constant game data is executed from Flash with its volatile memory in SRAM, avoiding overhead from accessing the external FRAM. Only checkpoints and patches are stored in external FRAM.

(white, yellow, red and black). This approach is similar to popular emulators that enable colour remapping by default, transforming Nintendo Game Boy games into the modern era where most games are rendered in colour.

ENGAGE Button Handling. In Section 3.2 we outlined the trade-off between energy generation and user satisfaction by altering the emulators handling of the buttons, for example removing the option to hold buttons for an extended duration of time. We limit the duration a button can be held to 300 ms. This is a duration similar to the button presses per second of *Space Invaders* as shown in Table 1. This approach forces the user to press buttons in a frequent pace but does not require excessive button pressing, balancing user satisfaction with energy generation. Due to the flexible nature of the ENGAGE platform future work can focus on user interaction with intermittent gaming devices. We will discuss this further in Section 6.1.

ENGAGE Memory Configuration. The Apollo3 ARM Cortex-M4 features flash and SRAM as on-board memory, where the Flash memory contains all the code (MPatch and Game Boy game emulator code) and non-volatile game data copied from the Game Boy game cartridge. SRAM contains memory of the whole ENGAGE platform and the volatile game memory—both separated from each other. Two buffers, *Checkpoint A* and *Checkpoint B*, for double buffering during checkpointing, as well as all patches created by MPatch reside in external FRAM. The complete memory map of ENGAGE is presented also in Figure 8.

4.3 MPatch Implementation

4.3.1 ENGAGE Core Checkpoints. MPatch is built upon a basic double-buffered checkpoint scheme which we denote as the **core checkpoint** system. The core checkpoint encompasses all the emulation management logic of ENGAGE, except for the emulated game memory, which is checkpointed using patches as described in Section 4.3.2. Specifically, the core checkpoint system checkpoints the .data, .bss and active stack sections of the MCU's volatile memory as well as the registers of the MCU, as can be seen in Algorithm 1. All this is double-buffered in the external non-volatile memory of ENGAGE. Naturally, this means that for every byte of volatile memory in the checkpoint, we need twice as much bytes in non-volatile memory. We remark that not all memory of ENGAGE is checkpointed. Specifically, we do not checkpoint memory buffers required for peripherals (as the peripheral state needs to be re-initialized every ENGAGE reboot). The restoration of a checkpoint will

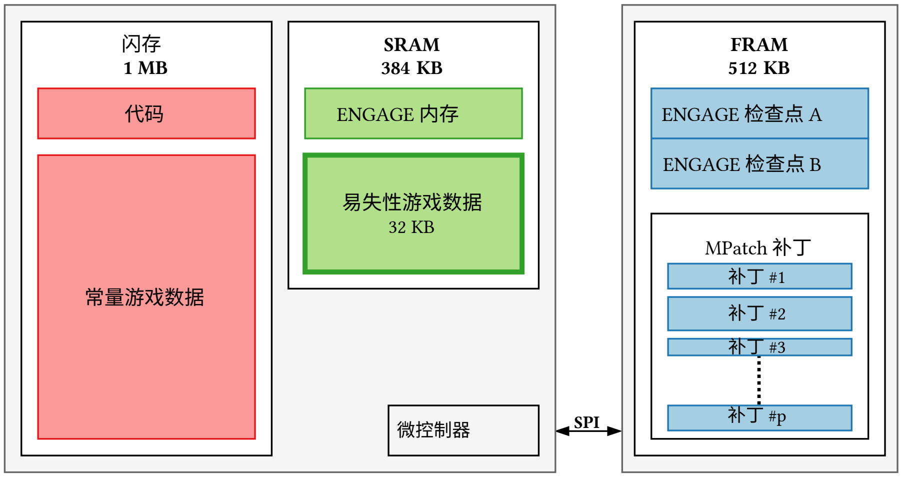


图 8.启用物理内存结构。常量游戏数据从Flash中执行，其易失性存储器位于SRAM中，避免了访问外部FRAM的开销。只有检查点和补丁存储在外部FRAM中。

(白色、黄色、红色和黑色)。这种方法类似于流行的仿真器，它们默认启用颜色重映射，将任天堂游戏男孩游戏转变为现代时代，在现代时代大多数游戏都是以彩色呈现的。

启用按钮处理。在第3.2节中，我们概述了通过改变仿真器对按钮的处理来平衡能量生成和用户满意度的权衡，例如，移除长时间按住按钮的选项。我们将按钮可以按住的时间限制为300毫秒。这一时长类似于太空入侵者每秒的按钮按压次数，如表1所示。这种方法迫使用户以频繁的节奏按压按钮，但不需要过度按压按钮，从而在用户满意度和能量生成之间取得平衡。由于ENGAGE平台的灵活性，未来的工作可以集中在与间歇性游戏设备的用户交互上。我们将在第6.1节进一步讨论这一点。

ENGAGE内存配置。Apollo3 ARM Cortex-M4具有闪存和SRAM作为板载内存，其中闪存包含所有代码（MPatch和游戏男孩游戏仿真器代码）以及从游戏男孩游戏卡带复制的非易失性game数据。SRAM包含整个ENGAGE平台的内存和易失性游戏内存——两者相互独立。两个缓冲区，检查点A和检查点B，用于在检查点技术中进行双缓冲，以及所有由MPatch创建的补丁都存储在外部FRAM中。ENGAGE的完整内存映射也在图8中呈现。

4.3 MPatch实现

*4.3.1 ENGAGE*核心检查点。MPatch建立在一个基本的双缓冲检查点方案之上，我们将其称为核心检查点系统。核心检查点包含了ENGAGE的所有仿真管理逻辑，除了被补丁保存的仿真游戏内存，如第4.3.2节所述。具体而言，核心检查点系统检查点 .data、 .bss和M*C*U的活动 stack部分的易失性存储器，以及MCU的寄存器，如算法1所示。所有这些都在ENGAGE的外部非易失性存储器中进行双缓冲。自然，这意味着对于检查点中的每个字节的易失性存储器，我们需要在非易失性存储器中两倍的字节。我们指出，并非ENGAGE的所有内存都被检查点化。具体而言，我们不检查点为外设所需的内存缓冲区（因为外设状态需要在每次ENGAGE重启时重新初始化）。检查点的恢复将

Algorithm 1 Checkpoint Creation

```
function CheckpointCreate
    CoreCheckpoint              ▹ Checkpoint memory not manged by MPatch
    PatchesCreate               ▹ Create and stage patches; see Section 4.3.2 and Algorithm 3
    RegisterCheckpoint          ▹ Checkpoint the CPU registers
    RestorePoint                ▹ Continuation point after a restore operation
    if isNotRestore then
        CheckpointCommit        ▹ Call function that commits the checkpoint
    end if
end function
```

Algorithm 2 Checkpoint Restoration

```
1: function CheckpointRestore
2:     CoreCheckpointRestore        ▹ Restore memory not manged by MPatch
3:     PatchesRestore               ▹ Restore committed patches; see Algorithm 5
4:     PeripheralRestore            ▹ Restore peripherals
5:     RegisterCheckpointRestore    ▹ Restore the CPU registers
6:     RestorePoint                 ▹ Continue at the restore point; see Algorithm 1
7: end function
```

restore the state of the system to that of the last successful checkpoint. If the system does not experience a *first time boot*, the default memory initialization step (which traditionally runs before any user code) will be skipped. After this, the steps listed in Algorithm 2 are performed to continue executing as if no power failure had occurred. In line 3 of Algorithm 2 the MPatch patch restoration process is started to restore the emulated game memory which will be discussed further in Section 4.3.2.

We designed the core checkpoint system from the ground up, implementing special keywords enabling the exclusion of certain volatile memory parts from a checkpoint. Also, the core checkpoint provides hooks for every stage of the checkpoint for ease of extension, which is required to incorporate patches from MPatch.

4.3.2 Patch Checkpoints Implementation. The emulated memory, i.e. the memory used by the Game Boy games, is a region in SRAM accessed only by emulated read and write instructions form the emulator. Leveraging this fact makes tracking modification to the emulated memory straightforward, and doing so has little impact on the overall performance. ENGAGE tracks these modifications, and when a checkpoint is created, this information is used to create the required patches as can be seen in Algorithm 3. Tracking of these modifications is done using the memory protection unit of the MCU. Upon writing to a region of emulated game memory, the memory protection unit triggers an interrupt allowing the memory region to be marked as modified. After a region is marked as modified the interrupt for the region is disabled. This results in an efficient method of tracking memory writes since the introduced overhead is only present during the first write after a reboot. The memory protection unit features eight regions which each have eight sub-regions, for a total of 64 sub-regions. We equally divided the memory space of the emulated Game Boy memory between these sub-regions resulting in patches containing 32 kB / 64 = 512 B of emulated memory.

Content of a Patch. In addition to the copy of a volatile memory region, a patch contains accompanying metadata required to successfully manage and restore a patch. This metadata is: (i) the *value of the logical clock n* from when the patch was staged, (ii) the *interval of the volatile memory* that is stored within the patch, (iii) the

算法 1检查点创建

```
函数CheckpointCreate
    核心检查点                          ▹ 检查点内存未由MPatch管理
    创建补丁                            ▹ 创建并准备补丁；见第4.3.2节和算法3
    注册检查点                          ▹ 检查CPU寄存器
    恢复点                              ▹ 恢复操作后的继续点
    如果不是恢复那么
        检查点提交                      ▹ 调用提交检查点的函数
  结束如果
结束函数
```

算法2检查点恢复

```
1: 函数检查点恢复
2:     核心检查点恢复                   ▹ 恢复未由MPatch管理的内存
3:     补丁恢复                         ▹ 恢复已提交的补丁；见算法5
4:     外设恢复                         ▹ 恢复外设
5:     注册检查点恢复                   ▹ 恢复CPU寄存器
6:     恢复点                           ▹ 在恢复点继续；见算法1
7: 结束函数
```

将系统状态恢复到最后一个成功检查点的状态。如果系统没有经历首次启动，默认的内存初始化步骤（通常在任何用户代码之前运行）将被跳过。
在此之后，将执行算法2中列出的步骤，以继续执行，仿佛没有发生电力故障。
在算法2的第3行，MPatch补丁恢复过程开始，以恢复仿真游戏内存
将在第4.3.2节中进一步讨论。

我们从零开始设计了核心检查点系统，实现了特殊关键字，使某些易失性存储器部分可以从检查点中排除。此外，核心检查点为检查点的每个阶段提供了钩子，以便于扩展，这对于整合来自MPatch的补丁是必需的。*4.3.2*补丁检查点实现。仿真内存，即游戏男孩游戏使用的内存，是SRAM中的一个

区域，仅通过仿真器的读写指令访问。利用这一事实，使得跟踪模拟内存的修改变得简单，并且这样做对整体性能影响很小。ENGAGE 跟踪这些修改，当创建检查点时，这些信息用于生成所需的补丁，如算法 3 所示。这些修改的跟踪是通过微控制器的内存保护单元完成的。当写入模拟游戏内存的某个区域时，内存保护单元会触发中断，从而将该内存区域标记为已修改。在一个区域被标记为已修改后，该区域的中断会被禁用。这导致了一种高效的内存写入跟踪方法，因为引入的开销仅在重启后的第一次写入时存在。内存保护单元具有八个区域，每个区域有八个子区域，总共有 64 个子区域。我们将模拟游戏男孩内存的内存空间均匀分配到这些子区域中， resulting in patches containing 32 kB/ 64 = 512 Bof emulated memory.

补丁的内容。除了易失性存储器区域的副本外，补丁还包含成功管理和恢复补丁所需的附带元数据。这些元数据包括：(i) 补丁分阶段时的逻辑时钟值 n，(ii) 存储在补丁中的易失性存储器的区间，(iii)

Algorithm 3 Patch Creation

```
function PatchesCreate
    while p ← ModifiedMemory do                     ▷ For each of the modified regions of memory
        PatchStage(address_start, address_end)      ▷ Create and stage the patch; see Algorithm 4
    end while
end function
```

Algorithm 4 Patch Staging

```
function PatchStage(address_start, address_end)
    patch ← AllocatePatch(address_start, address_end)     ▷ Allocate memory for a patch
    PatchCreate(patch)          ▷ Copy the volatile memory region into the non-volatile patch
end function
```

next patch in the patch chain, (iv) the *metadata to build an augmented interval tree* to speed up the restoration procedure, which will be discussed later in this section.

Patch Allocation. Patch sizes are allowed to differ, therefore some form of dynamic memory allocation is required. This brings challenges, as dynamic allocation leads to fragmentation, which is undesirable in an embedded system. Therefore patches are allocated using a *fixed-size block allocator* [64]. These allocated blocks are chained together to create enough room required to store the volatile memory within the non-volatile blocks. Each block contains: (i) a link to the *next block* in the chain, and (ii) a link to the *next free block* in the chain. All blocks are stored and managed in non-volatile memory. This creates challenges when trying to synchronize its non-volatile and volatile state. If these are not kept in sync, blocks will be lost, and the system may become corrupt. Additionally, write-after-read (WAR) violations [27] should be avoided when interacting with the non-volatile state. These two separate links in a block are required to eliminate one of these WAR violations, this violation could also be eliminated by introducing forced checkpoints, as inserting a checkpoint will break a WAR violation [27]. The total memory overhead of a patch in ENGAGE as it is currently implemented is 29 B. By excluding the interval tree required for the metadata, this can be reduced further to 17 B, but this would require an additional dynamic memory allocator to allocate this memory in volatile memory during a restoration (e.g. standard heap). For the final version used in ENGAGE, this was deemed undesirable, and therefore we integrated the interval tree metadata within non-volatile patches.

Patch Restoration. Restoring patches involves first discarding all staged—but not yet committed—patches, and then iterating through the patch chain while applying only the regions of a patch that were not previously applied during the restoration process. To keep track of the regions of volatile memory that were already restored we maintain an augmented *interval tree* during the restoration process. After a patch is applied, its range is added to the interval tree, and when a patch is applied, the interval tree is queried to detect overlaps. If there are no overlaps, the path is applied (i.e. written to the corresponding region in volatile memory). However, if the patch region overlaps with any region in the interval tree, the patch is split-up and all sub-patches are attempted to be applied. The complete algorithm for patch restoration is shown in Algorithm 5, with its accompanying patch apply algorithm shown in Algorithm 6.

Memory Recovery. One of the features of MPatch is its constant time patch creation while being incorruptible. However, patches that are no longer useful, i.e. that will not be applied during restoration, should be deleted. To avoid WAR violations, removing a patch (reclaiming its memory), consists of two operations. Firstly, the patch is freed, and secondly, the patch is deleted. Between these two operations, a checkpoint of only the MPatch

算法 3补丁创建

```
函数PatchesCreate
    当 p ←ModifiedMemory时                          ▷ 对于每个修改过的内存区域
        PatchStage(地址开始,地址结束)                 ▷ 创建并分阶段补丁；见算法 4
    结束当
结束函数
```

算法 4补丁分阶段

```
函数 PatchStage(地址开始,地址结束)
    补丁 ←AllocatePatch(地址开始,地址结束)          ▷ 为补丁分配内存
    PatchCreate(补丁)                                ▷ 将易失性存储区域复制到非易失性补丁中
结束函数
```

下一个补丁在补丁链中，(iv) 元数据用于构建增强的区间树以加速恢复过程，这将在本节后面讨论。

补丁分配。补丁大小可以不同，因此需要某种形式的动态内存分配。这带来了挑战，因为动态分配会导致碎片化，这在嵌入式系统中是不可取的。因此，补丁是使用固定大小块分配器 [64]分配的。这些分配的块被链接在一起，以创建足够的空间来存储非易失性块中的易失性内存。

每个块包含：(i) 指向链中下一个块的链接，以及 (ii) 指向链中下一个空闲块的链接。所有块都存储在非易失性存储器中并进行管理。这在尝试同步其非易失性和易失性状态时会造成挑战。如果这些未保持同步，块将会丢失，系统可能会变得损坏。此外，在与非易失性状态交互时，应避免写后读（WAR）违规 [27]。块中的这两个独立链接是消除这些WAR违规所必需的，这种违规也可以通过引入强制检查点来消除，因为插入检查点将打破WAR违规 [27]。在ENGAGE中，当前实现的补丁的总内存开销为29 B。通过排除元数据所需的区间树，这可以进一步减少到17 B，但这将需要一个额外的动态内存分配器在恢复期间在易失性存储器中分配此内存（例如，标准堆）。对于ENGAGE中使用的最终版本，这被认为是不理想的，因此我们将区间树元数据集成到非易失性补丁中。

补丁恢复。恢复补丁的过程首先涉及丢弃所有已暂存但尚未提交的补丁，然后遍历补丁链，仅应用在恢复过程中未先前应用的补丁区域。为了跟踪已经恢复的易失性存储器区域，我们在恢复过程中维护一个增强的区间树。在应用补丁后，其范围被添加到区间树中，当应用补丁时，查询区间树以检测重叠。如果没有重叠，则应用该路径（即写入易失性存储器中的相应区域）。然而，如果补丁区域与区间树中的任何区域重叠，则补丁被拆分，所有子补丁将尝试应用。补丁恢复的完整算法在算法5中展示，其伴随的补丁应用算法在算法6中展示。

内存恢复。MPatch的一个特点是其在不可腐蚀的情况下实现常量时间的补丁创建。然而，已不再有用的补丁，即在恢复过程中不会被应用的补丁，应当被删除。为了避免WAR冲突，删除一个补丁（回收其内存）包括两个操作。首先，释放补丁，其次，删除补丁。在这两个操作之间，仅对MPatch进行检查点

Algorithm 5 Patch Restoration (note: $low(p)$, $high(p)$ denote the low, high component of range p, respectively)

function PatchesRestore
  DiscardUncommitted ▹ Call function that discards uncommitted patches
  while $p_{\text{apply}} \leftarrow next(PatchChain)$ **do** ▹ Extract next patch from patch chain to restore it
    PatchApply(p_{apply}, $low(p_{\text{apply}})$, $high(p_{\text{apply}})$) ▹ Apply patch; see Algorithm 6
    IntervalInsert($low(p_{\text{apply}})$, $high(p_{\text{apply}})$) ▹ Insert the patch range into the interval tree
  end while
end function

Algorithm 6 Patch Apply (note: $low(p)$, $high(p)$ denote the low, high component of range p, respectively)

function PatchApply(p_{apply}, low, $high$)
  if $p_{\text{overlap}} \leftarrow$ IntervalOverlap(low, $high$) **then** ▹ Check for overlapping region in interval tree
    if $low < low(p_{\text{overlap}})$ **then**
      PatchApply(p_{apply}, low, $low(p_{\text{overlap}}) - 1$) ▹ Recursively apply patch with a new, partial, range
    end if
    if $high > high(p_{\text{overlap}})$ **then**
      PatchApply(p_{apply}, $high(p_{\text{overlap}}) + 1$, $high$) ▹ Recursively apply patch with a new, partial, range
    end if
  else
    Write(p_{apply}, low, $high$) ▹ Write patch content between *low* and *high* to the volatile memory
  end if
end function

management state is made containing patch and block allocation related metadata. During the deletion of a patch special care is taken to avoid WAR violations when modifying non-volatile memory in the patch chain. Memory recovery is not needed during every time a checkpoint is created or restored, is automatically done when there is no more non-volatile memory available to allocate a patch.

5 ENGAGE EVALUATION

We built ENGAGE as a proof by demonstration that battery-free mobile gaming was possible. In this section we demonstrate that the system can play unmodified retro games despite intermittent power failures. We analyze the real-world execution of the platform while playing *Tetris* in different lighting scenarios (i.e. with different energy scarcity) to show the effect of energy availability. We then benchmark the ENGAGE hardware platform for power consumption and, investigate the performance of the MPatch system. We find that in well-lit environments playing games that require at least moderate amounts of clicking, play is only slightly interrupted by power failures (less than one second of failure per every ten seconds of play). Our measurements of MPatch across four different games show that checkpoints are fast (less than 50 ms and restoration time after a power failures is not noticeable (average of 140 ms).

5.1 End-to-End ENGAGE Performance

First, we look at the typical play of ENGAGE executing an example Nintendo Game Boy game *Tetris*, chosen due to its requirement for moderate/high button presses and a small number of cut-scenes. We show how the system operates only on harvested energy. We execute two experiments, each in different lighting conditions: (i) 'daylight' with approximately 40 klx and (ii) 'shade' with approximately 20 klx, where a gamer plays ENGAGE

算法 5 补丁恢复（注意：*low*(*p*)和 *high*(*p*)分别表示范围 *p* 的低、高分量）

```
函数PatchesRestore
    DiscardUncommitted                                   ▷ 调用函数以丢弃未提交的补丁
    当 p应用 ← next(PPPPhPhPPP) do                        ▷ 从补丁链中提取下一个补丁以恢复它
        PatchApply(p应用,低（p应用）,高 （p应用））           ▷ 应用补丁；见算法6
        IntervalInsert(低（p应用）,高 （p应用））            ▷ 将补丁范围插入区间树
    结束循环
结束函数
```

算法6 补丁应用（注意：低(*p*),高(*p*)表示范围 *p* 的低、高分量）

```
函数 PatchApply(p应用, lll, hhi h)
    如果 p\重叠←|ntervalOverlap( lll, hhi h)那么          ▷ 检查区间树中的重叠区域
        如果 lll < low(p overlap)那么
            PatchApply(p应用, lll, low(p overlap) − 1)    ▷递归应用补丁，使用新的部分范围
        结束如果
        如果 h hi h> high(p overlap)那么
            PatchApply(p应用, high(p overlap) + 1, hhi h) ▷递归应用补丁，使用新的部分范围
        结束如果
    否则
        写入( p应用, lll, hhi h)                           ▷在易失性存储器中写入补丁内容，范围在低和\高之间
    结束如果
结束函数
```

管理状态包含与补丁和块分配相关的元数据。在删除补丁的过程中特别注意避免在修改补丁链中的非易失性存储器时发生写后读（WAR）冲突。在创建或恢复检查点时并不需要每次都进行内存恢复，当没有更多的非易失性存储器可用于分配补丁时，会自动进行。

5 参与评估

我们构建ENGAGE作为一个演示证明，无电池移动游戏是可能的。在本节中，我们演示该系统可以在间歇性电源故障的情况下播放未修改的复古游戏。我们分析平台在不同照明场景下（即在不同能源稀缺情况下）播放俄罗斯方块的实际执行，以展示能源可用性的影响。然后我们对ENGAGE硬件平台的功耗进行基准测试，并调查MPatch系统的性能。我们发现，在光线充足的环境中，进行需要至少中等点击量的游戏时，游戏的进行仅会因电源故障而稍有中断（每十秒游戏中不到一秒的故障）。我们对四款不同游戏的MPatch测量显示，检查点的响应速度很快（少于50毫秒），而电源故障后的恢复时间几乎不可察觉（平均140毫秒）。

5.1 端到端ENGAGE性能

首先，我们观察ENGAGE执行一个示例任天堂游戏男孩游戏俄罗斯方块的典型玩法，该游戏因其对中等/高按钮按压的要求和少量过场动画而被选中。我们展示了系统如何仅依靠收集的能量运行。我们进行了两个实验，每个实验在不同的光照条件下进行：(i) ‘日光’约为40 klx和(ii) ‘阴影’约为20 klx，玩家在ENGAGE中进行游戏。

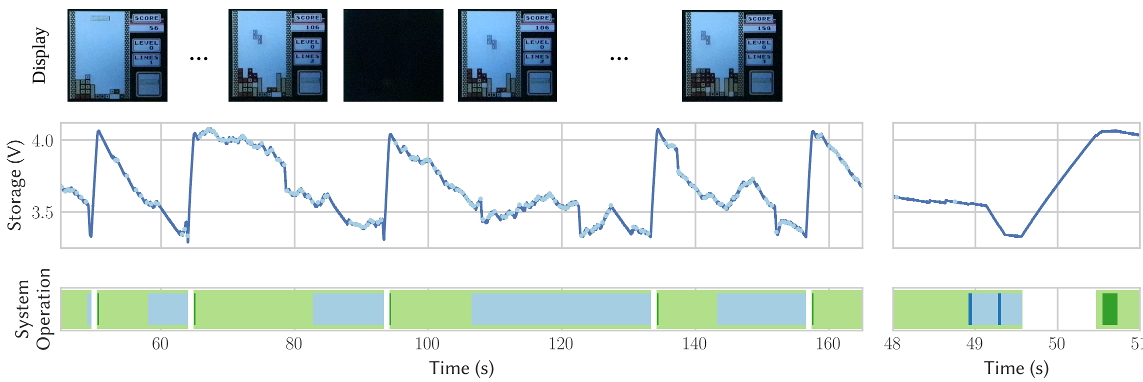


Fig. 9. End-to-end evaluation of ENGAGE operating in 'daylight' (approximately 40 klx during *Tetris* gameplay using harvested energy only. Storage capacitor voltage is shown, overlaid by unique button presses (marked as light blue dots). Additionally the following system events are shown at the bottom of the figure: initialization time (marked in dark green), system on time (marked in light green), low energy state (marked in light blue, denoting moments of ENGAGE periodically checkpointing due to critical system voltage) and checkpoint time (shown in dark blue in the separate zoomed-in window on the right). The actual game frames are shown on top, taken from recording the ENGAGE display during the evaluation scenario. The scenario shows that user interaction prolongs the on time of ENGAGE, by pressing buttons during gameplay—achieving ten seconds or more of on time with small off times. We consider this to be a playable *Tetris* scenario.

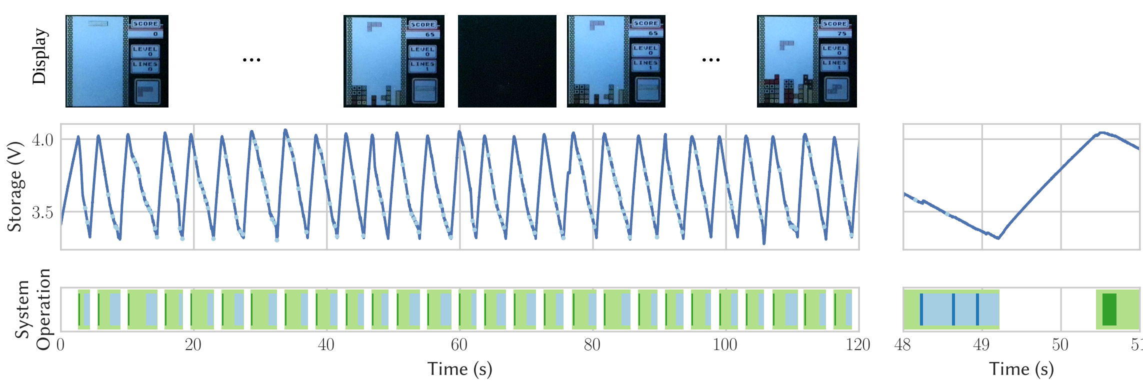


Fig. 10. End-to-end evaluation of ENGAGE operating in 'shade' (approximately 20 klx. Description of figure elements is the same as in Figure 9. With less energy available to ENGAGE as in the scenario in Figure 9, on times are reduced to around 3.5 s, with off times of more than a second. This scenario creates a noticeable impact to the user experience.

fully untethered, operating on harvested energy only. In the experiment the voltage of the main supply capacitor of ENGAGE is recorded together with various debugging signals indicating different system states. The system state and button presses are recorded using a Saleae logic pro 8 logic analyzer [113]. The ENGAGE platform was placed in a light box with two remotely controllable lights generating the two different light exposure conditions. The luminance of both scenarios was verified using a UNI-T UT383 lux meter [133].

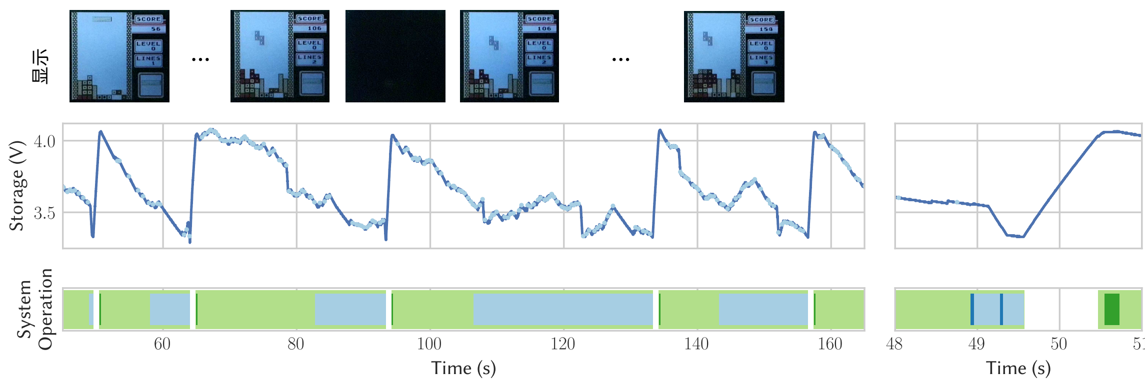


图 9.ENGAGE在'日光'下（大约40 klx）进行的端到端评估，使用收集的能量进行俄罗斯方块游戏。存储电容器电压显示在图中，并叠加了独特的按钮按压（标记为浅蓝色点）。此外，图底部显示了以下系统事件：初始化时间（标记为深绿色）、系统开启时间（标记为浅绿色）、低能量状态（标记为浅蓝色，表示ENGAGE因关键系统电压而定期检查点的时刻）和检查点时间（在右侧单独放大的窗口中以深蓝色显示）。实际游戏帧显示在顶部，取自评估场景中ENGAGE显示屏的录制。该场景表明，用户交互通过在游戏过程中按压按钮延长了ENGAGE的开启时间——实现了十秒或更长的开启时间，且小的关闭时间。我们认为这是一个可玩的俄罗斯方块场景。

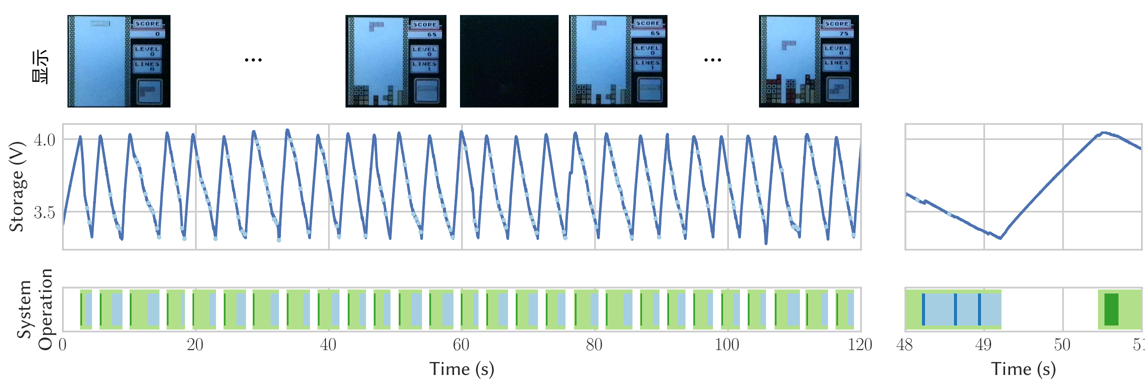


图10.ENGAGE在'阴影'下的端到端评估（约20 klx）。图形元素的描述与图9相同。在图9的场景中，ENGAGE可用的能量减少，开机时间缩短至约3.5秒，关机时间超过一秒。此场景对用户体验产生了显著影响。

完全无缝，仅依靠收集的能量运行。在实验中，记录了ENGAGE主供电电容器的电压，以及指示不同系统状态的各种调试信号。系统状态和按钮按压使用Saleae logic pro 8逻辑分析仪进行记录[113]。ENGAGE平台被放置在一个光箱中，配有两个可远程控制的灯，生成两种不同的光照条件。

使用UNI-T UT383光度计[133]验证了两种场景的亮度。

In the first scenario ('daylight', Figure 9) we show a period of execution with both little and many button presses. Here clearly the contribution of the energy harvesting by the switches is shown, significantly prolonging the on time of the device (marked in green). The figure shows the complete sequence from startup until the ENGAGE reaches a critically low energy level when it starts checkpointing. Due to the variability in the incoming energy pattern, ENGAGE can spend some time in this state, since it always needs to account for the worst-case scenario of no additional incoming energy. This scenario results in on times of ten seconds or more with small off times of less then a second, making it a very playable experience.

In the second scenario ('shade', Figure 10) we halved the amount of light the solar panels are exposed to compared to 'daylight', a more challenging condition for ENGAGE. This reduces on times to around 3.5 s with off times of more than a second. Despite the system still functioning correctly the lack of incoming energy becomes noticeable and even button mashing cannot compensate for the lack of energy. As with any energy harvesting platform, the limits of operation are defined to a major degree by the available energy in the environment. Full-system emulation is challenging and energy-intensive, but the game is still playable and functional; just with longer intermittent outages. We note that the downward peaks of storage voltage in Figure 9 and Figure 10 are caused by the energy harvester: during maximum power point tracking no energy is harvested causing the quick drop in the storage capacitor voltage.

Full-System Restoration Time. We have also measured end-to-end time of ENGAGE restoration: from the moment of applying power to the MCU to the moment of executing game code within the Game Boy emulator. In the case of *Tetris* this is 264 ms. The other games we tested resulted in comparable restore times, the main difference resulting from MPatch operations, as is further described in Section 5.3.

5.2 ENGAGE Power Consumption and Energy Generation

We have measured ENGAGE's power consumption, looking into overall power consumption whilst first measuring the consumption of MCU together with the FRAM and display. The MCU and FRAM combined consume 11.15 mW and the screen consumes 344.31 µW during game execution taking a ten second average. During idle time the screen only consumes 3.90 µW, resulting in a combined system average power consumption of 11.50 mW. As a comparison, the original Nintendo Game Boy consumes 232.08 mW during game execution, varying slightly per game and cartridge architecture. While not necessarily a useful or meaningful number, we conclude that our platform is *more than 20 times* more power-efficient than the original Nintendo Game Boy (representing normal technology advancement, but noting that ENGAGE is an emulator). The measurements were conducted using a Fluke 87V [33] multimeter and the X-NUCLEO-LPM01A [123] programmable power supply source with power consumption measurement capability.

Energy Generation. Then, to give more insight in the energy harvesting on the ENGAGE platform we have measured the amount of energy the solar panels generate using a Fluke 87V [33] multimeter and compare this to the energy generated from the buttons. For the buttons we use the minimal energy generation figures from the specification of the harvester [153, summary] as a worst case scenario[3]. Assuming that a single button press generates a minimum of 0.66 mJ and knowing the amount of button presses per game is specific to the game as per Table 1, we can assume the buttons generate between 0.66 mJ for one press per second and 1.98 mJ for three presses per second. At 40 klx and 20 klx, the solar panels generated an average of 10.14 mW and 8.33 mW, respectively, i.e. less than the required system average power consumption of 11.50 mW. We can conclude that ENGAGE is mostly powered by solar panels and supplemented by the button presses although the button presses can significantly increase the on-time of the platform, as shown in Section 5.1.

[3]Harvesting energy from the button energy harvesters is highly dependent on numerous factors such as the force applied and the manner of pressing the button hence the choice for the minimal figure.

在第一个场景（'日光'，图9）中，我们展示了一个执行周期，其中包括少量和大量的按钮按压。这里清楚地显示了开关的能量收集贡献，显著延长了设备的开启时间（以绿色标记）。该图显示了从启动到ENGAGE达到临界低能量水平的完整序列，此时它开始进行检查点技术。由于输入能量模式的变化，ENGAGE可以在此状态下花费一些时间，因为它始终需要考虑没有额外输入能量的最坏情况。这种情况导致开启时间达到十秒或更长，而关闭时间则少于一秒，使其成为一种非常可玩体验。

在第二个场景（'阴影'，图10）中，我们将太阳能电池板暴露的光线量减半，与'日光'相比，这对ENGAGE来说是一个更具挑战性的条件。这将开启时间减少到大约3.5，关闭时间超过一秒。尽管系统仍然正常运行，但缺乏输入能量变得明显，即使是疯狂按键也无法弥补能量的不足。与任何能量收集平台一样，操作的限制在很大程度上由环境中可用的能量决定。

全系统仿真具有挑战性且能量密集，但游戏仍然可以玩且功能正常；只是会有更长的间歇性停机。我们注意到图9和图10中存储电压的下降峰值是由能量收集器引起的：在最大功率点跟踪期间没有收集到能量，导致存储电容器电压的快速下降。

全系统恢复时间。我们还测量了ENGAGE恢复的端到端时间：从施加电源到微控制器的那一刻起，到执行游戏代码在游戏男孩仿真器中的那一刻。
在俄罗斯方块的情况下，这个时间为264毫秒。我们测试的其他游戏的恢复时间相当，主要差异来自于MPatch操作，进一步在第5.3节中描述。

5.2 ENGAGE功耗和能量生成

我们测量了ENGAGE的功耗，首先测量微控制器、FRAM和显示屏的整体功耗。微控制器和FRAM的组合功耗为11.15毫瓦，而屏幕在游戏执行期间的功耗为344. 31微瓦，取十秒的平均值。在空闲时间，屏幕仅消耗3.90微瓦，导致系统的平均功耗为11.50毫瓦。作为比较，原始的任天堂游戏男孩在游戏执行期间的功耗为232.08毫瓦，具体数值因游戏和卡带架构而略有不同。虽然这并不一定是一个有用或有意义的数字，但我们得出结论，我们的平台的功耗效率比原始任天堂游戏男孩高出*20*倍以上（代表正常的技术进步，但需要注意的是ENGAGE是一个仿真器）。测量使用了Fluke 87V [33]万用表和具有功耗测量能力的X-NUCLEO-LPM01A [123]可编程电源。

能量生成。然后，为了更深入地了解ENGAGE平台上的能量收集，我们使用Fluke 87V [33]万用表测量了太阳能电池板产生的能量，并将其与按钮产生的能量进行比较。对于按钮，我们使用收集器规格中的最小能量生成数据[153，摘要]作为最坏情况[3]。假设单次按钮按压生成至少0.66 mJ，并且知道每个游戏的按钮按压次数是特定于该游戏的，如表1所示，我们可以假设按钮每秒生成0.66 mJ（每秒一次按压）到1.98 mJ（每秒三次按压）之间的能量。在40 klx和20 klx下，太阳能电池板分别产生了平均10.14 mW和8.33 mW的能量，即低于所需的系统平均功耗11.50 mW。我们可以得出结论，ENGAGE主要由太阳能电池板供电，并通过按钮按压进行补充，尽管按钮按压可以显著增加平台的在线时间，如第5.1节所示。

[3]从按钮能量收集器中收集能量高度依赖于多个因素，例如施加的力量和按压按钮的方式，因此选择最小的数字。

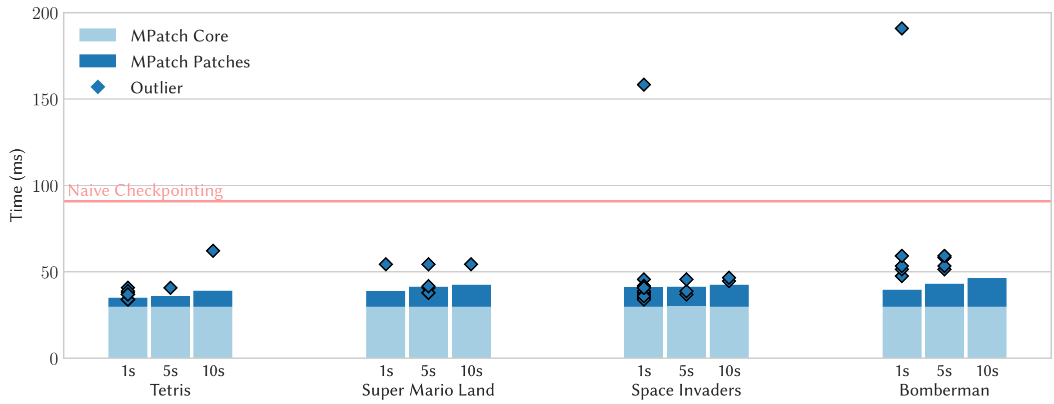


Fig. 11. MPatch checkpoint time comparison of approximately two minutes of game play per game using three different on times (1 s, 5 s, and 10 s) between successive checkpoints. ENGAGE has noticeably better performance than naive system, across all on times and games.

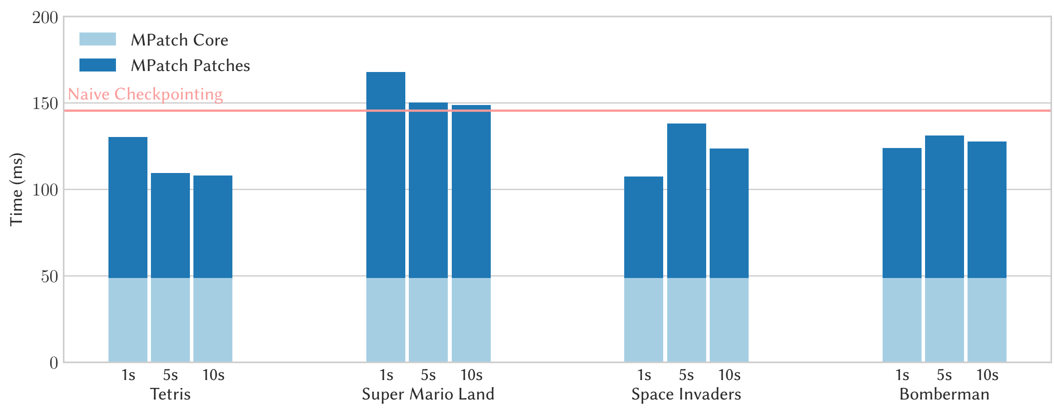


Fig. 12. Restoration time comparison of after approximately two minutes of game play per game using three different on times (1 s, 5 s, and 10 s) between successive checkpoints. ENGAGE has comparable or better performance than naive system, across all on times and games.

5.3 MPatch Performance

To better understand and quantify the effect of patches on the checkpoint and restore time, we evaluate MPatch against a *naive* approach—comparable in operation to Mementos [109]—where all active memory in the system is copied to non-volatile memory during a checkpoint, even if it was not modified since the last checkpoint. We compare these two strategies, MPatch and *naive*, by running multiple different games on ENGAGE. These games include: (i) *Tetris*, (ii) *Super Mario Land*, (iii) *Space Invaders*, and (iv) *Bomberman*. These games represent a wide variety of play styles, developers, and even release dates.

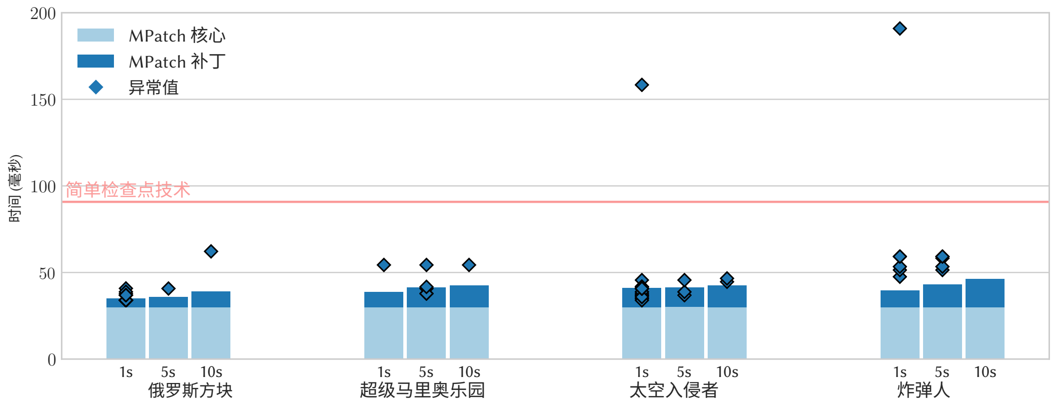


图 11.MPatch 检查点时间比较，约为每款游戏两分钟的游戏时间，使用三种不同的开启时间（1 s，5 s和10 s）在连续检查点之间。ENGAGE在所有开启时间和游戏中表现明显优于简单系统。

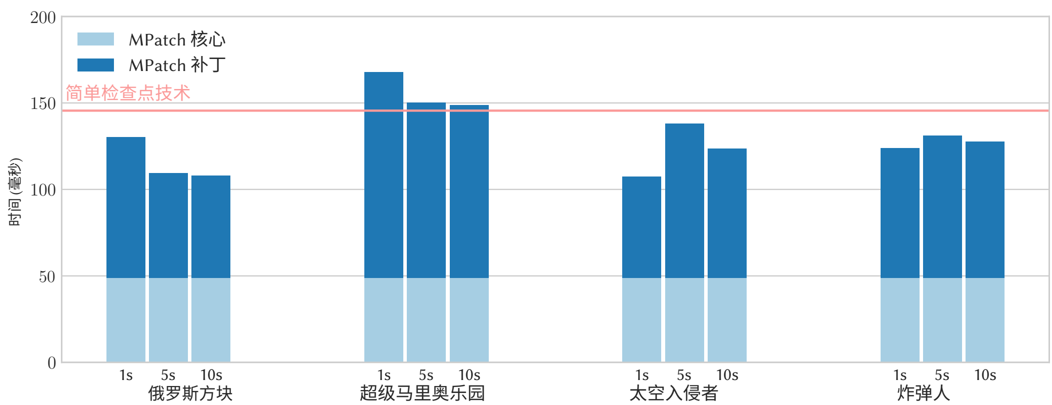


图 12.在使用三种不同的开启时间（1 s，5 s和10 s）之间的连续检查点进行游戏后，恢复时间比较约为两分钟。ENGAGE在所有开启时间和游戏中具有可比或更好的性能，优于简单系统。

5.3 MPatch性能

为了更好地理解和量化补丁对检查点和恢复时间的影响，我们将MPatch与一种简单方法进行评估——其操作与Mementos [109]相当——在检查点期间，系统中所有活动内存都被复制到非易失性存储器，即使自上一个检查点以来未被修改。我们通过在ENGAGE上运行多款不同的游戏来比较这两种策略，MPatch和简单。这些游戏包括：（i）俄罗斯方块，（ii）超级马里奥乐园，（iii）太空入侵者，以及（iv）炸弹人。这些游戏代表了多种多样的游戏风格、开发者，甚至发行日期。

5.3.1 MPatch Checkpoint Time. To measure only the impact of the MPatch patch checkpoints, we disable the just-in-time checkpoints—used in Section 5.1—and run the system on constant power during these measurements. Instead, we perform a checkpoint every c execution cycles of the emulator, we chose three different values for c, which correspond to different *on times*, i.e. 1 s, 5 s, and 10 s. During normal operation checkpoints will only be created when the voltage reaches a critical threshold, as seen in Section 5.1, these fixed on times represent a simplified scenario where the critical voltage threshold is reached after the specified on time. The *on time* affects the number and size of the checkpoints, as it allows for more memory writes between two consecutive checkpoints. The on time does not affect the *naive* checkpoint, as it always checkpoints all memory, with the only variable size being the system stack of ENGAGE. However, because of the way ENGAGE works—as an emulation loop—the system stack size is virtually constant.

During the emulation of each game, with the three different on times, we measured the cost of each component of the checkpoint using the same logic analyzer as used in experiments in Section 5.1. A checkpoint of ENGAGE consists of a core checkpoint of ENGAGE (Section 4.3.1) and additionally patches created by MPatch. The core checkpoint includes the management of both the emulator and the emulated memory, but excludes the emulated memory itself. This emulated memory is the largest memory component of the system, and therefore also the largest component of a naive checkpoint. For this reason we checkpoint this part of the system using MPatch, as the other components of ENGAGE are virtually constant in the amount of memory that is modified and are thus covered by the core checkpoint.

Figure 11 illustrates the naive checkpoint time as the horizontal line, the average checkpoint time of a core checkpoint (light blue bar), the differential component of a checkpoint using MPatch (dark blue bar), and the outliers (blue diamonds). As can be seen, the cost of the core checkpoint is around 30 % of the complete *naive* checkpoint, the rest being the emulated memory. However, when using MPatch to checkpoint the emulated memory, the core checkpoint dominates the total checkpoint time. In total MPatch is on average *more than two times faster* than the *naive* approach. This confirms our hypothesis that only a small amount of emulated memory is modified during execution. This reduction in checkpoint time directly leads to a lower energy requirement for each checkpoint and leaves more time for game emulation. Interestingly this assumption seems to hold even when the on time approaches 10 s, which is substantial for intermittent devices. Some outliers take longer than a *naive* checkpoint, this is due to a periodically performed memory recovery procedure (Section 4.3.2)—which was introduced to keep the creation of patches constant while keeping the system incorruptible.

5.3.2 MPatch Restoration Time. We also evaluate restoration time of patch checkpointing of MPatch, in a similar manner as in the previous section (i.e. the same set of games, comparison against three other reference mechanisms). The results are presented in Figure 12.

Restoring a patch-based checkpoint requires more time than the creation of a patch, as described in Section 4.3.2, due to the need to apply only the parts of the patches that are required, and because all the volatile memory has to be restored. Additionally, the restoration procedure must take into account all the committed patches when trying to restore the volatile memory, as each of these might hold some region that was only checkpointed using that specific patch. Therefore it is not directly influenced by the on-period, but influenced by the time since a *memory recovery.* Nevertheless, as can be seen in the figure, MPatch often *reduces the restoration time* compared to the *naive* restoration. We can also conclude from this that tested games often only modify a portion of their memory (in this case the emulated memory), as can also be seen in Figure 4.

6 DISCUSSION AND FUTURE WORK

Our evaluation of ENGAGE has shown that retro games are playable without batteries, making a next step in self-sustainable gaming made first decades ago by e.g. Bandai Corporation's LCD Solarpower game series [26]. Although the core gameplay mechanisms of the mobile handheld gaming have been successfully implemented,

5.3.1 MPatch 检查点时间。为了仅测量 MPatch 补丁检查点的影响，我们禁用即时检查点——在第 5.1 节中使用——并在这些测量期间让系统在恒定功率下运行。
相反，我们每 $c$execution cycles 执行一次检查点，我们选择了三个不同的 c值，这些值对应于不同的*on times*，即 1 秒、5 秒和 10 秒。在正常操作中，检查点仅在电压达到临界阈值时创建，如第 5.1 节所示，这些固定的 on times 代表了一个简化的场景，其中在指定的 on time 之后达到临界电压阈值。on time 影响检查点的数量和大小，因为它允许在两个连续检查点之间进行更多的内存写入。时间的准确性不会影响简单检查点，因为它始终检查所有内存，唯一的可变大小是ENGAGE的系统栈。然而，由于ENGAGE的工作方式——作为一个仿真循环——系统栈的大小几乎是恒定的。

在每个游戏的仿真过程中，我们测量了使用与第5.1节实验中相同的逻辑分析仪的检查点每个组件的成本，采用三种不同的时间。ENGAGE的检查点由ENGAGE的核心检查点（第4.3.1节）和由MPatch创建的补丁组成。核心检查点包括对仿真器和被仿真内存的管理，但不包括被仿真内存本身。这个被仿真的内存是系统中最大的内存组件，因此也是简单检查点中最大的组件。出于这个原因，我们使用MPatch对系统的这一部分进行检查点，因为ENGAGE的其他组件在修改的内存量上几乎是恒定的，因此被核心检查点覆盖。

图11展示了天真的检查点时间作为水平线，核心检查点的平均检查点时间（浅蓝色条），使用MPatch的检查点的差异组件（深蓝色条），以及离群值（蓝色菱形）。可以看出，核心检查点的成本大约是完整天真的检查点的30%，其余部分是模拟内存。然而，当使用MPatch对模拟内存进行检查点时，核心检查点主导了总检查点时间。总体而言，MPatch的平均速度比天真的方法快两倍以上。这证实了我们的假设，即在执行过程中只有少量的模拟内存被修改。检查点时间的减少直接导致每个检查点的能量需求降低，并为游戏模拟留出更多时间。有趣的是，即使在开机时间接近10秒时，这一假设似乎仍然成立，这对于间歇性设备来说是相当可观的。一些异常值的处理时间比简单的检查点要长，这是由于定期执行的内存恢复程序（第4.3.2节）所致——该程序的引入旨在保持补丁创建的恒定，同时保持系统的不可腐蚀性。

*5.3.2 MPatch*恢复时间。我们还评估了MPatch的补丁检查点恢复时间，方法与前一节相似（即相同的游戏集，与其他三种参考机制进行比较）。结果如图12所示。

恢复基于补丁的检查点所需的时间比创建补丁的时间更长，如第4.3.2节所述，这是因为需要仅应用所需的补丁部分，并且所有易失性存储器都必须恢复。此外，恢复程序在尝试恢复易失性存储器时必须考虑所有已提交的补丁，因为这些补丁中的每一个可能持有仅使用该特定补丁进行检查点的某个区域。因此，它并不直接受到开启周期的影响，而是受到自记忆恢复以来的时间影响。然而，如图所示，MPatch通常减少恢复时间与简单恢复相比。我们还可以从中得出结论，测试的游戏通常只修改其内存的一部分（在这种情况下是模拟内存），这在图4中也可以看到。

6 讨论与未来工作

我们对ENGAGE的评估表明，复古游戏在没有电池的情况下是可玩的，这标志着自我可持续游戏的下一步，早在几十年前就由例如万代公司的LCD太阳能游戏系列所开创[26]。
尽管移动掌上游戏的核心游戏机制已成功实现，

i.e. interaction with a screen-displayed data (for the original Nintendo Game Boy), other forms of interaction that make game experience complete are waiting to be researched and implemented.

6.1 Limitations, Alternatives and Future Work

Of course ENGAGE is just a first step in the direction of battery-free gaming and the proposed platform has still many limitations that need to be addressed. First, our battery-free platform *plays no sound*. We agree that no sound play is the main hurdle of complete game immersion. How to make sound enjoyable despite power supply intermittency is the core research question, but at the same time (in our opinion) an exciting research area. Some approaches to the sound problem we anticipate as worth-considering are (i) to include separate storage for sound buffering and play, following the architecture of [23, 42], (ii) introduce superficial pauses in the original game tone—effectively making the game sounds identical to the original battery-based game but punctured by silence at pre-selected moments—to make sound interrupts less irritating during gameplay, or (iii) to create intermittent system-specific game sounds—sounds that inform the user that the system is about to die or has just become operational again—to enrich battery-free gameplay.

Second, *playing in the dark has not been addressed*, as screen we used has no back light[4]. In the context of battery-free gaming, provision of back light for screen is very difficult. Simply, lack of light reduces amount of energy from harvesting, which in consequence reduces chance to perform *any* task—let alone supporting LEDs that are the most energy-consuming components of any embedded system.

Third, *haptics for battery-free games needs deeper investigation*. The energy harvesting buttons we have used in our prototype [153] are designed for industrial *sporadic* single-press cases (think of a battery-free wireless light switch). In frequent pressing cases these switches are much sturdy than the original Game Boy buttons, which for gamer can be a distracting feature. This necessitates a quest for more natural press buttons with equal (or better) energy harvesting. Furthermore, solar panels have to be placed on console's chassis such that their obstruction by fingers is minimized (as in our prototype). This, however, might downgrade the aesthetics of the device or require to make it bigger than necessary.

Fourth, *networking with battery-free game consoles* is another important point to consider, which was not addressed by us. Original Game Boy had an ability to connect, via cable, to another Game Boy for tandem gaming. This cable connection can be actually used for energy sharing and balancing between two consoles. Wireless networking of battery-free is an ongoing research task, which we did not want to cover with this work (for state of the art battery-free networking overview we refer to Section 7).

Fifth, *we cannot claim that all games will have the same playability* when ported to the intermittently-powered domain. Only when the off-times are negligible for the player we can safely assume that any existing game could be played intermittently/battery-free. Negligible off-times will cause no irritation to the person who is accustomed to always-on style of play. This observation would hold for any game system—not only classical (but old) Nintendo GameBoy we used as a basis for ENGAGE, but also recent systems such as PlayStation Portable or Nintendo Switch. An open research question is to find how long this off time is (less than a second or maybe less than a millisecond)? Our intuition says that this time is game-dependent and the longer the off times are present in a battery-free console, the set of games that can be ported to the battery-free platform gets smaller. Games that do not need frequent button pushes intuitively would be less irritating to play intermittently (e.g. *Chess*) or *Solitaire*), refer also to qualitative comparison of 8-bit Nintendo Games portability in Table 3. However, this creates an interesting paradox of button-based interaction. More button presses during the game result in more energy supplied to the game console, see also Table 1 and Section 3.2 (in extreme case games that are based on button bashing, such as classical *Track & Field* arcade game from Konami Corporation, gamer would be able to continuously generate energy purely from gameplay). At the same time less button presses result in less

[4]To be fair, the original Game Boy had the same deficiency, so do some of the upcoming gaming consoles [104].

即与屏幕上显示的数据进行交互（对于原始任天堂游戏男孩），使游戏体验完整的其他交互形式仍待研究和实现。

6.1 限制、替代方案与未来工作

当然，ENGAGE 只是朝着无电池游戏方向迈出的第一步，所提议的平台仍然存在许多需要解决的限制。首先，我们的无电池平台没有声音播放。我们同意，没有声音播放是完全游戏沉浸的主要障碍。如何在电源供应不稳定的情况下使声音变得愉悦是核心研究问题，但与此同时（在我们看来）也是一个令人兴奋的研究领域。我们认为值得考虑的一些解决声音问题的方法包括 (i) 为声音缓冲和播放单独设置存储，遵循 [23, 42] 的架构，(ii) 在原始游戏音调中引入表面暂停——有效地使游戏声音与原始基于电池的游戏相同，但在预选时刻被沉默打断——以减少游戏过程中声音中断的烦扰，或 (iii) 创建间歇性的系统特定游戏声音——通知用户系统即将关闭或刚刚重新启动的声音——以丰富无电池游戏体验。

其次，在黑暗中玩耍的问题尚未解决，因为我们使用的屏幕没有背光[4]。在无电池游戏的背景下，为屏幕提供背光是非常困难的。简单来说，缺乏光线减少了能量收集的数量，进而降低了执行任何任务的机会——更不用说支持LED，这些是任何嵌入式系统中最耗能的组件。

第三，无电池游戏的触觉反馈需要更深入的研究。我们在原型中使用的能量收集按钮[153]是为工业偶发单次按压情况设计的（可以想象成一个无电池的无线灯开关）。在频繁按压的情况下，这些开关比原始的游戏男孩按钮要坚固得多，这对玩家来说可能是一个分心的特征。这需要寻找更自然的按压按钮，具有相等（或更好的）能量收集能力。此外，太阳能电池板必须放置在控制台的机壳上，以尽量减少手指对其的遮挡（如我们原型中所示）。然而，这可能会降低设备的美观，或者需要使其比必要的更大。

第四，与无电池游戏机的网络连接是另一个重要的考虑点，而我们并未对此进行讨论。原版游戏男孩具有通过电缆连接到另一台游戏男孩进行双人游戏的能力。
这种电缆连接实际上可以用于两个游戏机之间的能量共享和平衡。无电池的无线网络连接是一个正在进行的研究任务，我们不想在这项工作中涵盖（有关无电池网络的最新概述，请参见第7节）。

第五，我们不能声称所有游戏在移植到间歇供电领域时将具有相同的可玩性。只有当关闭时间对玩家来说可以忽略不计时，我们才能安全地假设任何现有游戏都可以间歇性地/无电池地进行游戏。可忽略的关闭时间不会对习惯于始终在线游戏风格的人造成任何干扰。这一观察适用于任何游戏系统——不仅是我们作为ENGAGE基础的经典（但老旧的）任天堂游戏男孩，还包括最近的系统，如*PlayStation Portable*或任天堂Switch。一个开放的研究问题是找出这个关闭时间有多长（少于一秒，或者可能少于一毫秒）？我们的直觉认为，这个时间是与游戏相关的，电池无电的控制台中存在的关闭时间越长，可以移植到无电池平台的游戏集合就越小。

那些不需要频繁按键的游戏直观上在间歇性玩耍时会更少令人烦恼（例如国际象棋或纸牌），另请参见表3中8位任天堂游戏可移植性的定性比较。然而，这创造了一个有趣的按钮交互悖论。游戏中更多的按键按下会导致更多的能量供应给游戏机，另请参见表1和第3.2节（在极端情况下，基于按键狂按的游戏，例如Konami Corporation的经典田径街机游戏，玩家将能够仅通过游戏玩法持续产生能量）。与此同时，按键按下次数减少会导致更少的

[4]公平地说，原版游戏男孩也有同样的缺陷，一些即将推出的游戏机也是如此 [104]。

Table 3. This table describes the difficulty (or irritability) of playing types of Nintendo GameBoy games on intermittent power, assuming the intermittent effect is noticeable to the player and that enough energy is available for some level of play.

Game name	Type	Button presses	Intermittent play	Comments
Baseball	Sports	Very High	Hard	Reaction time is part of the game
Super Mario Land	Platformer	High	Hard	Button press order is crucial
Tetris	Puzzle	High	Medium	Tile rotation is often infrequent
Solitaire	Cards	Low	Easy	No penalty for missing a press
WordZap	Puzzle	Low	Easy	Easy with "no solving time" penalty
Chess	Strategy	Very Low	Easy	Most time spent on thinking

energy being created, causing reduction in continuous duration of play. To verify the above claims *detailed user studies considering large pool of gamers and games* need to be performed, where users play different games with artificially-induced intermittent operation (varying duration of on and off times).

Sixth, *screen retention needs to be introduced* (keeping screen state in-between on times) which our ENGAGE has not implemented yet. This simple extension would significantly reduce perceived negative effect of intermittent operation (think of a *Chess* game where state of the screen does not change much when the player is thinking and often user would not distinguish between off time and regular game operation, see again Table 3).

Finally, the overarching goal is to be able to *play state of the art 21 century handheld game consoles battery-free*, such as Playstation Vita Nintendo Switch—going beyond 8 bit architecture. This however requires years of research and can only be achieved by further advances in intermittently-powered software frameworks and ultra-low power electronics, which hopefully this work made a first step in achieving this goal.

General Software Framework for Battery-free Games. The goal of being able to run any existing or future game battery-free requires the introduction of general software framework for such games, going beyond checkpoining mechanism or a driver design presented in this paper, which is of course tailored towards the 8 bit Game Boy emulator. We envision a game engine, inspired by game engines of existing video games, such as the *Source* game engine [136] used in first-person shooter games such as *Counter-Strike*, that supports battery-free interaction abstracting underlying frameworks for intermittent operation form an actual game design.

Further Reduction of Game Console Carbon Footprint. We have made a first step towards making Game Console fabrication more environmentally friendly, however this is just a first step. Needless to say, original game cartridges of Nintendo Game Boy contain battery, and our console is based on many electronic components that are responsible for large CO_2 emissions in production [38], not to mention chassis that is made of plastic. More radical ideas need to be exploited, such as on the electronic level design with minimum amount of components (e.g. crystal-free design), going beyond policy changes in electronic fabrication enlisted, e.g. in [36].

Behavior Nudges to Generate More Energy. Many types of games have natural gaming mechanics that could be leveraged to increase energy harvesting actions. *Dance Dance Revolution*, *Bop-It*, and others, exploring this gaming induced behavior change for increasing energy is an interesting research direction. For example, a specific rapid button pressing sequence can trigger new game events (new levels, extra game points, etc.). Then, there are great user interfaces for battery-free interaction, for instance a crank[5], that can be researched further.

Native Execution. We chose the hard path: running a game emulator on an intermittent platform. This was to demonstrate the range of capabilities available to intermittent computing, and to leverage the vast amount of

[5]Which is already used in the upcoming post-retro *Playdate* console [104], which sadly is not used for internal battery charging.

表 3.该表描述了在间歇性供电下玩任天堂游戏男孩游戏的难度（或烦躁度），假设间歇性效应对玩家是显著的，并且有足够的能量进行一定程度的游戏。

游戏名称	类型	按钮按压	间歇性游戏	评论
棒球	体育	非常高	困难	反应时间是游戏的一部分
超级马里奥乐园	平台游戏	高	困难	按钮按压顺序至关重要
俄罗斯方块	益智	高	中等	方块旋转通常不频繁
纸牌	纸牌	低	简单	错过按键没有惩罚
WordZap	拼图	低	简单	简单且没有“解题时间”惩罚
国际象棋	策略	非常低	简单	大部分时间用于思考

能量被创造，导致连续游戏时间的减少。为了验证上述说法需要进行详细的用户研究，考虑到大量玩家和游戏，用户在不同游戏中进行人工诱导的间歇性操作（变化的开关时间持续时间）时进行测试。

第六，需要引入屏幕保留（在开机时间之间保持屏幕状态），而我们的ENGAGE尚未实现。这一简单的扩展将显著减少间歇性操作的感知负面影响（想象一下国际象棋游戏，当玩家思考时屏幕状态变化不大，用户通常无法区分关机时间和正常游戏操作，再次参见表3）。最后，整体目标是能够无电池地玩*21*世纪的尖端掌上游戏机，如Playstation Vita和任天堂Switch——超越8位架构。然而，这需要多

年的研究，并且只能通过间歇性供电的软件框架和超低功耗电子设备的进一步进展来实现，希望这项工作在实现这一目标方面迈出了第一步。无电池游戏的一般软件框架。能够无电池运行任何现有或未来游戏的目标需要引入这样的游戏的一般软件框架，超越本文中提出的检查点机制或驱动程序设计，当然，这些都是针对8位游戏男孩仿真器量身定制的。我们设想一个游戏引擎，受到现有视频

游戏的游戏引擎的启发，例如在第一人称射击游戏中使用的*S*ource游戏引擎[136]，如反恐精英，支持无电池交互，抽象出间歇性操作的底层框架，形成实际的游戏设计。

进一步减少游戏主机的碳足迹。我们在使游戏主机制造更加环保方面迈出了第一步，但这仅仅是第一步。不必说，任天堂游戏男孩的原版游戏卡带包含电池，而我们的控制台基于许多电子组件，这些组件在生产中负责大量的二氧化碳排放 [38]，更不用说由塑料制成的机壳。需要开发更激进的想法，例如在电子层面设计最少量的组件（例如，无晶体设计），超越电子制造政策的变化，如 [36] 中所列。

行为引导以产生更多能量。许多类型的游戏具有自然的游戏机制，可以利用这些机制来增加能量收集的行为。舞蹈革命， *Bop-It*等，探索这种gaming诱导的行为变化以增加能量是一个有趣的研究方向。例如，特定的快速按键序列可以触发新的游戏事件（新关卡、额外游戏积分等）。然后，有很好的用户界面用于无电池交互，例如一个曲柄[5]，可以进一步研究。

本地执行。我们选择了艰难的道路：在间歇平台上运行游戏仿真器。这旨在展示间歇计算可用的能力范围，并利用大量的

[5]这已经在即将推出的后复古 *Playdate*控制台中使用[104]，遗憾的是并未用于内部电池充电。

pre-built games that can play unchanged on the platform. However, one could imagine that native gameplay would significantly increase the performance of the platform, by orders of magnitude, since a single emulated instruction has significant overhead over native code for the platform. This could be accomplished by compiling game binaries to native ARM code, or by leveraging a bespoke gaming API from bare-metal C code. The latter is intriguing as an exercises to take advantage of the unique aspects of intermittently-powered and battery-free gaming, where the situation and context, as well as the gameplay, will effect how much energy is harvested. Game mechanics leveraging this system attribute might increase engagement.

6.2 Gaming and the Environment

Electronic games are an important part of the world's economy [95]. First and foremost they are crucial to mental well being of many people around the world. Especially in the time of the COVID-19 pandemic, when millions of people are stranded at home, various forms of electronic gaming are one of the activities that reduce stress and boredom due to lockdown implemented by most of world's governments [116, 118, 127]. At the same time, the electronic gaming industry is and important job creator, and although being a financially non-struggling industry, to say the least [95, 100, 118], is also actively supported by international governments': as an latest example refer to CD Projekt—creator of *The Witcher* video game series—and its list of European Union-funded projects the company participated in [15]. At the same time it is apparent that gaming industry contributes significantly to global warming. In the United States alone gaming is responsible for "24 MT/year of associated carbon-dioxide emissions equivalent to that of 85 million refrigerators" [90]. To tackle that challenge the gaming industry is joining various industry consortia such as *Playing for the Planet* [108] aiming at reducing its ecological impact. Independently, some national governments aim at influencing the gaming industry requesting content providers to throttle-down data rate of streaming services with too high demand [36]—resulting in smaller electricity consumption of data centers.

But all the above actions to address climate impact of the gaming industry do not tackle the effect of battery-based/handheld/mobile gaming (the above-mentioned study of [90] explicitly excludes such devices from the analysis). Beyond any doubt handheld gaming devices, while extremely popular [98], contribute independently to increased worldwide CO_2 emissions. While we are not aware of any detailed studies on the carbon footprint of popular handheld gaming consoles, such as Nintendo Switch[6], its impact is beyond negligible. For example, Nintendo was *the least* environmentally-friendly company of Greenpeace 2010 *Guide to Greener Electronics* ranking [152], while none of the video-game oriented companies are listed among the world's most sustainable corporations in year 2020 [25].

There are numerous components that gaming handheld console/mobile phone is made of that cause substantial environmental impact [38] so removing some of them without compromising the usability would by highly appreciated from the environment point of view. A first potent candidate for such removal is a battery. Production of batteries has great environmental impact by itself and many research projects are devoted to making *batteries-only* more sustainable [31]. But even if most of the goals of more sustainable batteries are met by 2030, they will *still* have to be produced, collected and recycled. And while we conjecture that majority of console game players do not consider reliance on batteries as a problem[7] it is the *environmental responsibility of the electronic designers* to address the battery issue for the users of handheld gaming consoles.

[6]None of the handheld gaming platforms are listed in the Electronic Product Environmental Assessment register [39]; the closest study of environmental impact of Nintendo Switch we are aware of is given in [69]. As a reference, in-depth analysis of carbon footprint of one of the most popular non-handheld gaming console, Sony's Play Station 4, is available in [38]. To quote from this study: "(s)ince the PlayStation 4's release in 2013, approximately 8.9 billion kilograms of carbon dioxide have been generated and subsequently released into the atmosphere".
[7]Actually, in many cases game console players are close to a power socket playing their games tethered, making a problem of battery replacement or recharge even less profound.

可以在该平台上不加修改地播放的预构建游戏。然而，人们可以想象，本地游戏玩法将显著提高平台的性能，数量级地提升，因为单个仿真指令相对于平台的本地代码具有显著的开销。这可以通过将游戏二进制文件编译为本地 ARM 代码，或通过利用来自裸机 C 代码的定制游戏 API 来实现。后者作为一种利用间歇供电和无电池游戏独特方面的练习是非常有趣的，在这种情况下，环境和上下文以及游戏玩法将影响能量的收集。

利用该系统属性的游戏机制可能会增加参与度。

6.2 游戏与环境

电子游戏是全球经济的重要组成部分 [95]。首先，它们对世界上许多人的心理健康至关重要。尤其是在COVID-19大流行期间，当数百万人被困在家中时，各种形式的电子游戏成为减少因大多数国家政府实施的封锁而产生的压力和无聊的活动之一 [116, 118, 127]。与此同时，电子游戏产业是一个重要的就业创造者，尽管它在财务上并不挣扎，至少可以这么说 [95, 100, 118]，但也得到了国际政府的积极支持：最新的例子是CD Projekt——《巫师》视频游戏系列的创作者——及其参与的欧盟资助项目列表 [15]。同时显而易见，游戏产业对全球变暖的贡献显著。仅在美国，游戏就负责"每年2400万吨的二氧化碳排放，相当于8500万台冰箱"[90]。为了应对这一挑战，游戏产业正在加入各种行业联盟，如为地球而玩[108]，旨在减少其生态影响。

一些国家政府则独立地旨在影响游戏产业，要求内容提供商降低数据流服务的速率，以应对过高的需求[36]——从而减少数据中心的电力消耗。

但上述所有应对游戏产业气候影响的行动并未解决基于电池的/手持的/移动游戏的影响（上述[90]的研究明确将此类设备排除在分析之外）。毫无疑问，手持游戏设备虽然极受欢迎[98]，但独立地对全球二氧化碳排放的增加做出了贡献。虽然我们并不清楚关于流行手持游戏机（如任天堂Switch[6]）的碳足迹的详细研究，但其影响绝非微不足道。例如，任天堂在2010年绿色和平组织的《更环保电子产品指南》中被评为最不环保的公司[152]，而在2020年[25]，专注于视频游戏的公司中没有一家被列为全球最可持续的企业。

手持游戏机/手机由许多组件组成，这些组件对环境造成了显著影响[38]，因此在不影响可用性的情况下去除其中一些组件将从环境的角度受到高度赞赏。一个首要的去除候选者是电池。电池的生产本身对环境有很大的影响，许多研究项目致力于使电池更加可持续[31]。但即使到2030年大多数可持续电池的目标得以实现，它们仍然需要被生产、收集和回收。虽然我们推测大多数游戏主机玩家并不认为依赖电池是一个问题[7]但电子设计师有责任为手持游戏主机的用户解决电池问题。

[6]没有任何手持游戏平台被列入电子产品环境评估注册表[39]；我们所知的关于任天堂Switch环境影响的最接近研究见于[69]。作为参考，关于最受欢迎的非手持游戏主机之一，索尼PlayStation 4的碳足迹的深入分析可在[38]中找到。引用该研究的话："自2013年PlayStation 4发布以来，约有89亿千克二氧化碳被产生并释放到大气中。"

[7]实际上，在许多情况下，游戏主机玩家在玩游戏时靠近电源插座，使用有线连接，这使得电池更换或充电的问题显得不那么突出。

7 RELATED WORK

Battery-free Sensors. Long before our idea of a batter-free gaming console, non-gaming embedded platforms were realized in a battery-free manner—making these sensor more environmentally-friendly. The first such battery-free platforms were wireless sensors [106]. First battery-free sensors were based on the idea of computational RFID tags: programmable RFID tags with on-board sensors (such as accelerometers or temperature sensors) communicating with the outside world by radio frequency backscatter to a RFID reader. WISP [114, 135] and Moo [134] are the first realization of such RFID tags. Since the introduction of WISP and Moo many research groups have focused on making battery-free backscatter communication more efficient [146], for instance, by making it free from dedicated energy sources [105], by enabling communication with non-backscatter networks such as IEEE 802.11 [63] or LoRa [125], or by improving backcatter-based networks—either based on standard RFID protocols [80], or based on dedicated backscatter network stack [41]. A separate line of research focused on introducing camera-based image processing to backscatter-based sensors. First, a backscatter-based battery-less cameras, as an extension to WISP platform, has been demonstrated in [93, 94], later followed by a dedicated (non-WISP) backscatter-based system [92, 112]. Additionally, non-radio frequency backscatter systems based on passive visible light communication backscatter, such as PassiveVLC mote [148], have also been demonstrated. It is important to remark that the biggest drawback of backscatter-based systems is the reliance on external energy source (itself powered by batteries or power line) that downscales the benefit of removing battery from a complete system.

Additionally, battery-free sensors that communicate using non-backscatter, i.e. active, communication techniques also become actively researched. These include simple sense and transmit sensor powered by ambient temperature differences [156], UFoP [42] and Capybara [23]—energy-harvesting storage-adaptive sensors, Battery Free Phone [126], SkinnyPower—wearable sensor powered by intra-body power transfer [121], Camaroptera—image-inferring sensor [96], SoZu—battery-free activity detector [155], or Botoks—time-aware wireless sensor [28]. Non-wireless/non-communicating battery-free sensors include CapHarvester—local energy monitor powered by harvesting stray voltage from AC power lines-[40], self-powered step motion counter [60], Saturn—battery-free microphone [6], and active radio battery-less eye tracker [73].

Battery-free Interactive Devices. It is imperative to extend battery-less devices beyond a simple 'sense-and-transmit' functionality (as summarized above) demonstrating simple forms of user interaction. The same RFID technology that lay the foundation for battery-free sensing was also used to demonstrate battery-less interaction. Such systems include RFID-based tags displaying external information [102], elderly monitoring based on embedded-in-clothes RFID tags [58], surface shape detection [59], speech recognition [142], augmented reality with (i) unmodified RFID tags [72] and (ii) modified RFID tags (to enable touch sensing) [47], interactive building block system with augmented RFID tags[8] [48, 76] or finger gesture measurement [62]. It needs to be emphasized that any RFID tags-based interaction is very sensitive to interference and signal mis-matches as demonstrated in [141].

Separately from RFID-based battery-free interactive devices, non-RFID counterparts are also actively researched. Most of these devices focus on remote device control through touch. Examples of such devices are capacitance-based touch sensor (although communicating with FM radio receiver through backscatter) [140], Ohmic-Sticker—force-to-capacitance sensor attachable to laptop touchpad [51], aesthetically pleasing self-powered interactive surfaces based on photovoltaic cells [87] and self-powered gesture recognition based on (i) photovoltaic panels [79][9], (ii) photodiodes [74] and (iii) capacitance sensing [132]. E-ink battery-free wearable displays embedded in clothes, energized by NFC-enabled smartphones were demonstrated in [29].

[8] A similar concept for NFC-based tags has been presented in [14].

[9] System claims to be battery-less, while in evaluation a battery-based version was used.

7 相关工作

无电池传感器。在我们构思无电池游戏控制台之前，非游戏嵌入式平台已经以无电池的方式实现，使这些传感器更加环保。首个此类无电池平台是无线传感器 [106]。首个无电池传感器基于计算射频识别标签的理念：可编程的射频识别标签，配备有板载传感器（如加速度计或温度传感器），通过射频反向散射与外部世界进行通信。WISP [114, 135] 和Moo [134] 是此类射频识别标签的首次实现。自WISP和Moo推出以来，许多研究小组专注于提高无电池反向散射通信的效率 [146]，例如，通过使其不依赖于专用能源 [105]，通过与非反向散射网络（如IEEE 802.11 [63] 或LoRa [125]）进行通信，或通过改进基于反向散射的网络——无论是基于标准射频识别协议 [80]，还是基于专用反向散射网络栈 [41]。一项独立的研究集中于将基于反向散射的传感器引入基于相机的图像处理。首先，作为WISP平台的扩展，已经在[93, 94]中展示了基于反向散射的无电池相机，随后又出现了一个专用的（非WISP）基于反向散射的系统[92, 112]。此外，基于被动可见光通信反向散射的非射频反向散射系统，如PassiveVLC节点[148]，也已被展示。

重要的是要指出，基于反向散射的系统最大的缺点是依赖于外部能源（本身由电池或电源线供电），这降低了从完整系统中去除电池的好处。

此外，使用非反向散射，即主动通信技术进行通信的无电池传感器也成为了积极研究的对象。这些包括由环境温度差异供电的简单感知和传输传感器[156]，UFoP[42]和Capybara[23]——能量收集存储自适应传感器，Battery Free Phone[126]，SkinnyPower——由体内能量传输供电的可穿戴传感器[121]，Camaroptera——图像推断传感器[96]，SoZu——无电池活动检测器[155]，或Botoks——时间感知无线传感器[28]。

非无线/非通信的无电池传感器包括CapHarvester——由从交流电力线收集的杂散电压供电的本地能量监测器-[40]，自供电步态计数器 [60]，无电池麦克风Saturn [6]，以及主动无线电无电池眼动追踪器 [73]。

无电池互动设备。扩展无电池设备的功能超越简单的'感知与传输'（如上所述）以展示简单形式的用户交互是至关重要的。奠定无电池传感基础的同一RFID技术也被用于展示无电池交互。

此类系统包括基于RFID的标签显示外部信息 [102]，基于嵌入衣物中的RFID标签的老年人监测 [58]，表面形状检测 [59]，语音识别 [142]，增强现实（i）未修改的RFID标签 [72] 和（ii）修改过的RFID标签（以启用触摸感应） [47]，带有增强RFID标签的互动积木系统[8] [48, 76] 或手势测量 [62]。需要强调的是，任何基于RFID标签的交互对干扰和信号不匹配非常敏感，如[141]所示。

除了基于RFID的无电池交互设备外，非RFID的对应设备也在积极研究中。这些设备大多数集中在通过触摸进行远程设备控制。此类设备的例子包括基于电容的触摸传感器（尽管通过反向散射与FM无线电接收器通信）[140]，Ohmic-Sticker——可附加到笔记本触摸板的力-电容传感器[51]，基于光伏电池的美观自供电交互表面[87]以及基于（i）光伏面板[79][9]，（ii）光电二极管[74]和（iii）电容传感的自供电手势识别[132]。嵌入衣物中的电子墨水无电池可穿戴显示屏，由NFC启用的智能手机供电，如[29]所示。

[8]在文献[14] 中提出了一个类似的基于NFC标签的概念。
[9]系统声称是无电池的，但在评估中使用了基于电池的版本。

Another approach for battery-free embedded devices is to equip the area where the sensor resides in some form of wireless power transfer system. Many end-to-end wireless power solutions can be found in the literature, including recent systems build on top of capacitive power transfer [154], magnetic resonant coupling [124], quasistatic cavity resonance [115], lasers [54] or distributed RF beamforming [32]. As in the case of backscatter-based sensors, wirelesly-powered sensors require external (complex, bulky and still having not fully resolved safety issues) infrastructure. This limits applicability of this approach to ubiquitous battery-free gaming.

Battery-free Gaming. An ultimate form of interaction is through a gaming system. A first, commercial battery-free/solar-powered gaming platform was Bandai's LCD Solarpower [26], released already in 1982, that enabled manipulation of hard-coded elements on a liquid crystal display. Unfortunately, Bandai's console and modern existing academic-grade battery-free gaming systems are limited to a simple game forms, such as attachable touch pad extenders for better (but still battery-powered) mobile game experience [18, 151] (similar to an earlier referred design [51]), extra controllers for smartphones based on its front/rear cameras [149], or based on RFID technology that requires heavy-lifting of battery-less features by an expensive RFID reader using either (i) computational RFID tags [134, 135] as for instance in [86], or (ii) using commercial off-the-shelf RFID tags as in [71]. Battery-free non-RFID touch pad extender for the introduction of physical manipulation into touch screen-based games was prototyped in [97]. Battery-free gaming aimed at children includes system based on rubbing/touching electrostatic surfaces to power simple electronics [16, 17, 61] and attachable energy harvester mote for learning and understanding concepts of energy generation and consumption [111].

New Electronic Game Forms. Battery-less handheld gaming console, ENGAGE, presented in this paper introduce a novel form of self-powering play, where user (to continue playing a normal electronic handheld game platform) is (sometimes) required to push buttons to continuously power a device. This is a twist on movement-inducing (exer)games [9, 53] such as Pokémon GO [66] where movement is required only to *perform better* in game instead of *perform better and continue* to play. This is a new form of game interaction that use the human body as an immanent component of gaming experience, as advocated in [91]. We note that novel forms of games with dedicated hardware (albeit battery-powered) are introduced, where the energy of the body is used to introduce a novel form of interaction. A recent example of such game is based on swallowable temperature measurement pills [75] or through-body electric field propagation [138, 139].

Gaming as a Behavioral Intervention. Research community is in constant search for new forms of gaming interaction and our battery-less gaming console aims at defining yet another gaming behavior. Such new forms of gaming are for instance, 'idle games' [4] or new game forms with custom-made haptics, such as virtual reality games for blind people [117]). Design challenges in behaviour-inducing games (such as exergames referred earlier) have been discussed recently in [66].

Considering classical gaming behavioral studies we can refer to game design that activate children to play outdoors [101], study on the effect of 'gamification' of cognitive tasks [143] or observation of gaming experience as an indication of cognitive abilities [52]. We are not aware of any non-orthodox gaming behavioral studies.

A separate line of research, although not strictly related to games, touches upon behavioral change of battery-powered smartphones usage. These studies include crowdsensing of battery usage for suggestion of better user behaviour extending battery lifetime [21], optimization of frame rate for mobile (smartphone-based) games saving energy on frame rendering [50], or a proposal for new form of interaction with mobile devices with turned-off screen to conserve energy [147]. Our battery-free game console is the first study that considers a behavioral intervention for *battery-free* device.

Sustainable Design of Interactive Devices. Design of any future interactive devices must consider sustainability and reuse, as advocated already a decade ago in [12, 85]. The same plea, but in the context of pervasive devices, was presented in [55]. Since almost a decade many studies call for sustainable 'upstream' HCI by making

另一种无电池嵌入式设备的方法是为传感器所在区域配备某种形式的无线电力传输系统。文献中可以找到许多端到端无线电力解决方案，包括基于电容电力传输的最新系统[154]、磁共振耦合[124]、准静态腔共振[115]、激光[54]或分布式射频波束形成[32]。与基于反向散射的传感器一样，无线供电的传感器需要外部（复杂、笨重且仍未完全解决安全问题的）基础设施。这限制了这种方法在无处不在的无电池游戏中的适用性。无电池游戏。一种终极的交互形式是通过游戏系统。首个商业化的无电池/太阳能供电游戏平台是万代的LCD太阳能游戏机[26]，该设备于1982年发布，能够在液晶显示屏上操控硬编

码元素。不幸的是，万代的游戏机和现代现有的学术级无电池游戏系统仅限于简单的游戏形式，例如可附加的触控板扩展器，以改善（但仍然是电池供电的）移动游戏体验[18, 151]（类似于之前提到的设计[51]），基于其前/后摄像头的智能手机额外控制器[149]，或基于RFID技术，这需要通过昂贵的RFID读取器来实现无电池功能的重负荷，使用（i）计算型RFID标签[134, 135]，例如在[86]中，或（ii）使用商业现成的RFID标签，如在[71]中。用于将物理操控引入基于触摸屏的游戏的无电池非RFID触控板扩展器在[97]中进行了原型设计。面向儿童的无电池游戏包括基于摩擦/触摸静电表面来为简单电子设备供电的系统 [16, 17, 61]，以及可附加的能量收集器模块，用于学习和理解能量生成和消耗的概念 [111]。

新电子游戏形式。无电池手持游戏机ENGAGE，在本文中介绍了一种自供电游戏的新形式，用户（为了继续玩正常的电子手持游戏平台）有时需要按下按钮以持续为设备供电。这是一种对运动诱导（锻炼）游戏的变革 [9, 53]，例如《口袋妖怪GO》 [66]，在这种游戏中，运动仅仅是为了在游戏中表现更好，而不是为了表现更好并继续玩。这是一种新的游戏互动形式，利用人体作为游戏体验的内在组成部分，如 [91] 中所倡导的。我们注意到，尽管是电池供电的专用硬件游戏形式被引入，身体的能量被用来引入一种新型的互动形式。最近的一个例子是基于可吞咽的温度测量药丸 [75] 或通过体内电场传播 [138, 139] 的游戏。

作为行为干预的游戏。研究界不断寻找新的游戏互动形式，而我们的无电池游戏机旨在定义另一种游戏行为。这种新的游戏形式例如'闲置游戏' [4] 或具有定制触觉的新游戏形式，例如为盲人设计的虚拟现实游戏 [117]。行为诱导游戏（如前面提到的运动游戏）的设计挑战最近在 [66] 中进行了讨论。

考虑到经典的游戏行为研究，我们可以提到激励儿童户外游戏的游戏设计 [101]，关于认知任务'游戏化'效果的研究 [143] 或将游戏体验作为认知能力的指示的观察 [52]。我们并不知晓任何非正统的游戏行为研究。

一条独立的研究方向，尽管与游戏并不严格相关，但涉及到电池供电智能手机使用的行为变化。这些研究包括对电池使用的众包感知，以建议更好的用户行为以延长电池寿命 [21]，优化移动（基于智能手机的）游戏的帧率以节省帧渲染的能量 [50]，或提出一种新的与关闭屏幕的移动设备互动的方式以节省能量 [147]。我们的无电池游戏机是首个考虑无电池设备的行为干预研究。

互动设备的可持续设计。任何未来互动设备的设计必须考虑可持续性和再利用，这一点在十年前的研究中已经被提倡 [12, 85]。同样的呼吁，但在普遍设备的背景下，在 [55] 中提出。近十年来，许多研究呼吁可持续的'上游'人机交互设计。

conscious choices in HCI design process in selecting materials that are sustainable, recyclable and reusable [65] or using post-apocalyptic terms— HCI "designed for use after the industrialized context has begun to decay" [131]. We are unaware of any studies on whether the (handheld) gaming community considers sustainable gaming as important, let alone existing, problem. Loosely related study to our posted problem is the study on the motivations behind leading green households [145].

Intermittent Computing Systems. The goal of intermittent computing frameworks is to guarantee correctness and completion of the computation of battery-less energy harvesting embedded platforms *despite* frequent power interrupts[10]. Such framework is essential for the usability of battery-free gaming platform.

From the publication of a first framework supporting intermittently-powered devices, Mementos [109]—voltage threshold-triggered checkpoining system, more efficient checkpoint systems are being published. These include Hibernus++ [8] and QuickRecall [57] (just like Mementos, both hardware-activated chceckpoints), Chinchilla [82], Rachet [137] and HarvOS [11] (all three compiler-instrumented checkpoints), TICS (time-aware checkpoints) [68], TotalRecall (checkpoints using volatile memory) [144], Elastin (adaptive checkpoints) [19], DICE (differential checkpoints) [2, 3] and WhatsNext (checkpointing augmented with approximate computing) [35]. A second class of systems include runtimes based on specially instrumented code (by form for a *task*) such as Dino [78], Chain [22], Alpaca [81], MayFly [45], InK [150], Coati [110], CoSpec [20] and Coala [84]. A third class of intermittent computation support systems are hardware-assisted systems such as Clank [46] that check for memory inconsistencies. Important to recall are workload-specific computation systems such as on-device inference on intermittently-powered devices with off-line and on-line learning, see [37] and [70], respectively.

A separate stream of work targets peripheral support for intermittently-powered devices, such as Restop (through dedicated middleware) [7], Samoyed (through just-in-time checkpoints) [83] and Karma (supporting parallel or asynchronous peripheral operations) [13], or targeting handling of dedicated peripherals such as e-displays (to improve their update rate) [88].

8 CONCLUSIONS

This paper presented a first working example of a battery-free gaming console, and the first full system emulation on intermittent power: ENGAGE. We demonstrate we can port existing battery-based gaming platforms—such as in our case 8 bit Nintendo Game Boy—to the battery-free domain. With this platform we have shown that deeply interactive devices, like gaming platforms, are possible to create without batteries, and in spite of frequent power failures. We developed a novel hardware and software platform to facilitate this new class of device: (i) a hybrid energy harvesting device tailored towards battery-free gaming and (ii) a new system for persistent computation across power failures based on a novel concept of patch checkpointing of volatile memory state into non-volatile memory regions. ENGAGE represents a bright future of deeply interactive, maintenance and battery-free devices.

ACKNOWLEDGMENTS

We thank our anonymous reviewers for their useful comments. In addition, we would like to thank Izabela Grudzińska for help in preparing the photographs for the paper. This research was supported by Netherlands Organisation for Scientific Research, partly funded by the Dutch Ministry of Economic Affairs, through TTW Perspective program ZERO (P15-06) within Project P1 and P4, and by the National Science Foundation through grants CNS-1850496 and CNS-2032408. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] 2020. ENGAGE Open Source Repository. https://github.com/tudssl/engage. (July 2020). Last accessed: Jul. 22, 2020.

[10]For a good overview of intermittent computing concepts we independently refer to [44, 77, 89].

在HCI设计过程中做出有意识的选择，选择可持续、可回收和可重复使用的材料 [65]，或者使用后启示录术语——HCI"设计用于工业化背景开始衰退后的使用" [131]。
我们不知道是否有研究表明（手持）游戏社区是否认为可持续游戏重要，更不用说现存的问题。与我们提出的问题 loosely 相关的研究是关于推动绿色家庭的动机的研究 [145]。

间歇计算系统。间歇计算框架的目标是确保无电池能量收集嵌入式平台的计算正确性和完成性，尽管频繁的电源中断[10]。这样的框架对无电池游戏平台的可用性至关重要。

自第一个支持间歇供电设备的框架Mementos [109]——电压阈值触发的检查点系统发布以来，越来越多的高效检查点系统被发布。这些包括 Hibernus++ [8] 和 QuickRecall [57]（与 Mementos 类似，都是硬件激活的检查点）、Chinchilla [82]、Rachet [137] 和 HarvOS [11]（这三者都是编译器插桩的检查点）、TICS（时间感知检查点）[68]、TotalRecall（使用易失性存储器的检查点）[144]、Elastin（自适应检查点）[19]、DICE（差分检查点）[2, 3] 和 WhatsNext（增强近似计算的检查点技术）[35]。第二类系统包括基于特殊插桩代码的运行时（通过形式为 a *task*）如 Dino [78]、Chain [22]、Alpaca [81]、MayFly [45]、InK [150]、Coati [110]、CoSpec [20] 和 Coala [84]。第三类间歇性计算支持系统是硬件辅助系统，例如 Clank [46]，用于检查内存不一致性。重要的是要回忆起特定工作负载的计算系统，例如在间歇供电设备上进行的设备内推理，以及离线和在线学习，见 [37] 和 [70]。

一系列独立的工作针对间歇供电设备的外围支持，例如 Restop（通过专用中间件）[7]，Samoyed（通过即时检查点）[83] 和 Karma（支持并行或异步外围操作）[13]，或针对专用外设的处理，例如电子显示器（以提高其更新速率）[88]。

8 结论

本文展示了第一个无电池游戏控制台的工作示例，以及在间歇供电下的第一个完整系统仿真：ENGAGE。我们证明可以将现有的基于电池的游戏平台——例如我们案例中的 8 位任天堂游戏男孩——移植到无电池领域。通过这个平台，我们展示了深度互动设备（如游戏平台）可以在没有电池的情况下创建，并且能够应对频繁的电源故障。我们开发了一种新颖的硬件和软件平台，以促进这一新类型设备的实现：(i) 一种针对无电池游戏的混合能量收集设备，以及 (ii) 一种基于补丁检查点技术的新系统，用于在电源故障期间实现持久计算，将易失性存储器状态保存到非易失性存储器区域。ENGAGE 代表了深度互动、无需维护和无电池设备的光明未来。

致谢

我们感谢匿名评审员的有用意见。此外，我们还要感谢伊莎贝拉·格鲁金斯卡（Izabela Grudzińska）在准备论文照片方面的帮助。本研究得到了荷兰科学研究组织的支持，部分资金来自荷兰经济事务部，通过TTW Perspective项目ZERO（P15-06）中的项目P1和P4，以及国家科学基金会通过资助CNS-1850496和CNS-2032408提供的支持。本文中表达的任何意见、发现、结论或建议均为作者（们）的观点，并不一定反映国家科学基金会的观点。

参考文献

[1] 2020.ENGAGE开源库。https://github.com/tudssl/engage。（2020年7月）。最后访问时间：2020年7月22日。

[10]关于间歇计算概念的良好概述，我们独立参考[44, 77, 89]。

[2] Saad Ahmed, Muhammad Hamad Alizai, Junaid Haroon Siddiqui, Naveed Anwar Bhatti, and Luca Mottola. 2018. Poster Abstract: Towards Smaller Checkpoints for Better Intermittent Computing. In *Proc. IPSN* (Nov. 11–13). ACM/IEEE, Porto, Portugal, 132–133.

[3] Saad Ahmed, Naveed Anwar Bhatti, Muhammad Hamad Alizai, Junaid Haroon Siddiqui, and Luca Mottola. 2019. Effient Intermittent Computing with Differential Checkpointing. In *Proc. LCTES*. ACM, Phoenix, AZ, USA, 70–81.

[4] Sultan A. Alharthi, Olaa Alsaedi, Zachary O. Toups, Joshua Tanenbaum, and Jessica Hammer. 2018. Playing to Wait: A Taxonomy of Idle Games. In *Proc. CHI* (April 21–26). ACM, Montréal, QC, Canada, 210:1–210:13.

[5] Ambiq Micro Inc. 2018. Apollo3 Blue Ultra-Low Power Microcontroller. https://ambiqmicro.com/static/mcu/files/Apollo3_Blue_MCU_Data_Sheet_v0_11_0.pdf. (2018). Last accessed: Apr. 25, 2020.

[6] Nivedita Arora, Steven L. Zhang, Fereshteh Shahmiri, Diego Osorio, Yicheng Wang, Mohit Gupta, Zhengjun Wang, Thad Eugene Starner, Zhonglin Wang, and Gregory D. Abowd. 2018. SATURN: A Thin and Flexible Self-powered Microphone Leveraging Triboelectric Nanogenerator. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 2 (June 2018), 60:1–60:28.

[7] Alberto Rodriguez Arreola, Domenico Balsamo, Geoff V. Merrett, and Alex S. Weddell. 2018. RESTOP: Retaining External Peripheral State in Intermittently-powered Sensor Systems. *Sensors* 18, 1 (2018), 172.

[8] Domenico Balsamo, Alex S. Weddell, Anup Das, Alberto Rodriguez Arreola, Davide Brunelli, Bashir M. Al-Hashimi, Geoff V. Merrett, and Luca Benini. 2016. Hibernus++: a Self-calibrating and Adaptive System for Transiently-powered Embedded Devices. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.* 35, 12 (2016), 1968–1980.

[9] Soumya C. Barathi, Daniel J. Finnegan, Matthew Farrow, Alexander Whaley, Pippa Heath, Jude Buckley, Peter W. Dowrick, Burkhard C. Wünsche, James L. J. Bilzon, Eamonn O'Neill, and Christof Lutteroth. 2018. Interactive Feedforward for Improving Performance and Maintaining Intrinsic Motivation in VR Exergaming. In *Proc. CHI* (April 21–26). ACM, Montréal, QC, Canada, 408:1–408:14.

[10] Uwe Becker. 2017. Gameboy-Emulator per STM32F746 (in German). https://mikrocontroller.bplaced.net/wordpress/?page_id=1290. (Nov. 2017). Last accessed: May 5, 2020.

[11] Naveed Bhatti and Luca Mottola. 2017. HarvOS: Efficient Code Instrumentation for Transiently-powered Embedded Devices. In *Proc. IPSN* (April 18–20). ACM/IEEE, Pittsburgh, PA, USA, 209–219.

[12] Eli Blevis. 2007. Sustainable Interaction Design: Invention & Disposal, Renewal & Reuse. In *Proc. CHI* (April 28 – May 3). ACM, San Jose, CA, USA, 503–512.

[13] Adriano Branco, Luca Mottola, Muhammad Hamad Alizai, and Junaid Haroon Siddiqui. 2019. Intermittent Asynchronous Peripheral Operations. In *Proc. SenSys* (Nov. 10–13). ACM, New York City, NY, USA, 55–67.

[14] Lars Büthe, Michael Hardegger, Patrick Brulisauer, and Gerhard Tröster. 2014. RFID-Die: Battery-free Orientation Sensing Using an Array of Passive Tilt Switches. In *Proc. UbiComp Adjunct*. ACM, Seattle, WA, USA, 215–218.

[15] CD Projekt. 2020. European Union Projects. https://www.cdprojekt.com/en/capital-group/eu-projects. (April 2020). Last accessed: Apr. 28, 2020.

[16] Arunkumar Chandrasekhar, Gaurav Khandelwa, Nagamalleswara Rao Alluri, Venkateswaran Vivekananthan, and Sang-Jae Kim. 2018. Battery-Free Electronic Smart Toys: A Step toward the Commercialization of Sustainable Triboelectric Nanogenerators. *Sustainable Chemistry and Engineering* 6, 5 (April 2018), 6110–6116.

[17] Arunkumar Chandrasekhar, Gaurav Khandelwal, Nagamalleswara Rao Alluri, Venkateswaran Vivekananthan, and Sang-Jae Kim. 2017. Sustainable Biomechanical Energy Scavenger toward Self-Reliant Kids' Interactive Battery-Free Smart Puzzle. *Sustainable Chemistry and Engineering* 5, 8 (June 2017), 7310–7316.

[18] Tzuwen Chang, Neng-Hao Yu, Sung-Sheng Tsai, Mike Y. Chen, and Yi Ping Hung. 2012. Clip-on Gadgets: Expandable Tactile Controls For Multi-touch Devices. In *Proc. MobileHCI* (Sept. 21–24). ACM, San Francisco, CA, USA, 163–166.

[19] Jongouk Choi, Hyunwoo Joe, Yongjoo Kim, and Changhee Jung. 2019a. Achieving Stagnation-Free Intermittent Computation with Boundary-Free Adaptive Execution. In *Proc. RTAS*. IEEE, Montréal, QC, Canada, 331–344.

[20] Jongouk Choi, Qingrui Liu, and Changhee Jung. 2019b. CoSpec: Compiler Directed Speculative Intermittent Computation. In *Proc. MICRO*. ACM, Columbus, OH, USA, 399–412.

[21] Yohan Chon, Gwangmin Lee, Rhan Ha, and Hojung Cha. 2016. Crowdsensing-based Smartphone Use Guide for Battery Life Extension. In *Proc. UbiComp*. ACM, Heidelberg, Germany, 958–969.

[22] Alexei Colin and Brandon Lucia. 2016. Chain: Tasks and Channels for Reliable Intermittent Programs. In *Proc. OOPSLA* (Oct. 30 – Nov. 4). ACM, Amsterdam, The Netherlands, 514–530.

[23] Alexei Colin, Emily Ruppel, and Brandon Lucia. 2018. A Reconfigurable Energy Storage Architecture for Energy-harvesting Devices. In *Proc. ASPLOS* (March 24–28). ACM, Williamsburg, VA, USA, 767–781.

[24] Rodrigo Copetti. 2019. Architecture of Consoles: GameBoy. https://copetti.org/projects/consoles/game-boy. (Feb. 2019). Last accessed: May 3, 2020.

[25] Corporate Knights. 2020. 2020 Global 100 Ranking: Index of the World's Most Sustainable Corporations. https://www.corporateknights.com/reports/2020-global-100. (Winter 2020). Last accessed: Apr. 30, 2020.

[26] Bandai Corporation. 1982. LCD Solarpower Handheld Electronic Games Series. https://en.wikipedia.org/wiki/Bandai_LCD_Solarpower. (1982). Last accessed: Sep. 22, 2020.

[2] Saad Ahmed, Muhammad Hamad Alizai, Junaid Haroon Siddiqui, Naveed Anwar Bhatti, 和 Luca Mottola. 2018. 海报摘要：为了更小的检查点以改善间歇计算。在*Proc. IPSN*（11月11日至13日）。ACM/IEEE，葡萄牙波尔图，132–133。

[3] Saad Ahmed, Naveed Anwar Bhatti, Muhammad Hamad Alizai, Junaid Haroon Siddiqui, 和 Luca Mottola. 2019.高效间歇计算与差分检查点技术. 在*Proc. LCTES*中.ACM, 凤凰城, 亚利桑那州, 美国, 70–81.

[4] Sultan A. Alharthi, Olaa Alsaedi, Zachary O. Toups, Joshua Tanenbaum, 和 Jessica Hammer. 2018.等待游戏：闲置游戏的分类法。在*Proc. CHI*(4月 21–26).ACM, 蒙特利尔, 魁北克省, 加拿大, 210:1–210:13.

[5] Ambiq Micro Inc. 2018.Apollo3 蓝色超低功耗微控制器. https://ambiqmicro.com/static/mcu/files/Apollo3_Blue_MCU_Data_Sheet_v0_11_0.pdf. (2018). 最后访问时间：2020年4月25日.

[6] Nivedita Arora, Steven L. Zhang, Fereshteh Shahmiri, Diego Osorio, Yicheng Wang, Mohit Gupta, Zhengjun Wang, Thad Eugene Starner,Zhonglin Wang, 和 Gregory D. Abowd. 2018.SATURN：一种利用摩擦电纳米发电机的薄型柔性自供电麦克风。*ACM Interact. Mob. Wearable Ubiquitous Technol.*2, 2 (2018年6月), 60:1–60:28.

[7] Alberto Rodriguez Arreola, Domenico Balsamo, Geoff V. Merrett, 和 Alex S. Weddell. 2018年.RESTOP：在间歇供电传感器系统中保留外部外设状态。传感器18, 1 (2018年), 172.

[8] Domenico Balsamo, Alex S. Weddell, Anup Das, Alberto Rodriguez Arreola, Davide Brunelli, Bashir M. Al-Hashimi, Geoff V. Merrett, 和 Luca Benini. 2016年.Hibernus++：一种自校准和自适应的瞬时供电嵌入式设备系统。*IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.* 35, 12 (2016年), 1968–1980.

[9] Soumya C. Barathi, Daniel J. Finnegan, Matthew Farrow, Alexander Whaley, Pippa Heath, Jude Buckley, Peter W. Dowrick, Burkhard C.Wünsche, James L. J. Bilzon, Eamonn O'Neill, 和 Christof Lutteroth. 2018.交互式前馈以提高性能并保持虚拟现实健身游戏中的内在动机。在*Proc. CHI*(4月21–26日)。ACM, 蒙特利尔, QC, 加拿大, 408:1–408:14.

[10] Uwe Becker. 2017.基于STM32F746的Gameboy仿真器（德语）。https://mikrocontroller.bplaced.net/wordpress/?page_id=1290.（2017年11月）。最后访问时间：2020年5月5日。

[11] Naveed Bhatti 和 Luca Mottola. 2017.HarvOS：瞬时供电嵌入式设备的高效代码工具。在 *Proc. IPSN* （4月18–20日）。ACM/IEEE，匹兹堡，宾夕法尼亚州，美国，209–219。

[12] Eli Blevis. 2007年。可持续交互设计：发明与处置，更新与重用。在*Proc. CHI*（4月28日 – 5月3日）。ACM，圣荷西，加利福尼亚州，美国，503–512。

[13] Adriano Branco, Luca Mottola, Muhammad Hamad Alizai 和 Junaid Haroon Siddiqui. 2019年。间歇性异步外设操作。在*Proc. SenSys*（11月10–13日）。ACM，纽约市，纽约州，美国，55–67。

[14] Lars Büthe, Michael Hardegger, Patrick Brulisauer 和 Gerhard Tröster. 2014年。RFID-Die: 使用被动倾斜开关阵列的无电池方向感应。在*Proc. UbiComp Adjunct*中。ACM，西雅图，华盛顿，美国，215–218。

[15] CD Projekt. 2020。欧洲联盟项目。https://www.cdprojekt.com/en/capital-group/eu-projects。（2020年4月）。最后访问时间：4月。28, 2020.

[16] Arunkumar Chandrasekhar, Gaurav Khandelwa, Nagamalleswara Rao Alluri, Venkateswaran Vivekananthan 和 Sang-Jae Kim. 2018。无电池电子智能玩具：朝着可持续摩擦电纳米发电机商业化迈出的一步。可持续化学与工程6, 5（2018年4月），6110–6116。

[17] Arunkumar Chandrasekhar, Gaurav Khandelwal, Nagamalleswara Rao Alluri, Venkateswaran Vivekananthan 和 Sang-Jae Kim. 2017。可持续生物机械能量收集器，面向自给自足儿童互动无电池智能拼图。可持续化学与工程5, 8 (2017年6月), 7310–7316.

[18] Tzuwen Chang, Neng-Hao Yu, Sung-Sheng Tsai, Mike Y. Chen, 和 Yi Ping Hung. 2012.夹式小工具：可扩展的触觉控制用于多点触控设备。见*Proc. MobileHCI*(9月21–24日).ACM, 旧金山, CA, 美国, 163–166.

[19] Jongouk Choi, Hyunwoo Joe, Yongjoo Kim, 和 Changhee Jung. 2019a.实现无停滞间歇计算与无边界自适应执行。见*Proc. RTAS*.IEEE, 蒙特利尔, QC, 加拿大, 331–344.

[20] Jongouk Choi, Qingrui Liu, 和 Changhee Jung. 2019b.CoSpec: 编译器导向的推测性间歇计算. 在 *Proc. MICRO*.ACM, 哥伦布, 俄亥俄州, 美国, 399–412.

[21] Yohan Chon, Gwangmin Lee, Rhan Ha, 和 Hojung Cha. 2016.基于众包感知的智能手机使用指南以延长电池寿命. 在*Proc. UbiComp*.ACM, 海德堡, 德国, 958–969.

[22] Alexei Colin 和 Brandon Lucia. 2016.Chain: 可靠间歇程序的任务和通道. 在*Proc. OOPSLA*(10月30日 – 11月4日).ACM, 阿姆斯特丹, 荷兰, 514–530.

[23] Alexei Colin, Emily Ruppel, 和 Brandon Lucia. 2018.一种可重构的能量存储架构用于能量收集设备。发表于*ASPLOS*（3月24日至28日）。ACM，威廉斯堡，弗吉尼亚州，美国，767–781。

[24] 罗德里戈・科佩提。2019年。控制台架构：游戏男孩。https://copetti.org/projects/consoles/game-boy。（2019年2月）。最后访问时间：2020年5月3日。

[25] 企业骑士。2020年。2020全球100强排名：全球最可持续公司的指数。https://www.corporateknights.com/reports/2020-global-100。（2020年冬季）。最后访问时间：2020年4月30日。

[26] 万代公司。1982年。LCD太阳能便携电子游戏系列。https://en.wikipedia.org/wiki/Bandai_LCD_Solarpower。（1982年）。最后访问时间：2020年9月22日。

[27] Marc de Kruijf and Karthikeyan Sankaralingam. 2013. Idempotent Code Generation: Implementation, Analysis, and Evaluation. In *Proc. CGO*. ACM/IEEE, Shenzhen, China.

[28] Jasper de Winkel, Carlo Delle Donne, Kasım Sinan Yıldırım, Przemysław Pawełczak, and Josiah Hester. 2020. Reliable Timekeeping for Intermittent Computing. In *Proc. ASPLOS* (March 16–20). ACM, Lausanne, Switzerland, 53—-67.

[29] Christine Dierk, Molly Jane Pearce Nicholas, and Eric Paulos. 2018. AlterWear: Battery-Free Wearable Displays for Opportunistic Interactions. In *Proc. CHI* (April 21–26). ACM, Montréal, QC, Canada, 210:1–210:13.

[30] Economist Intelligence Unit. 2020. The IoT Business Index 2020: a Step Change in Adoption. https://learn.arm.com/rs/714-XIJ-402/images/economist-iot-business-index-2020-arm.pdf. (Feb. 2020). Last accessed: May 7, 2020.

[31] Kristina Edström (Executive Publisher). 2020. Horizon 2020 EU Program Battery 2030+: Inventing the Sustainable Batteries of the Future: Research Needs and Future Actions. https://battery2030.eu/digitalAssets/861/c_861350-l_1-k_roadmap-27-march.pdf. (March 2020). Last accessed: Apr. 30, 2020.

[32] Xiaoran Fan, Han Ding, Sugang Li, Michael Sanzari, Yanyong Zhang, Wade Trappe, Zhu Han, and Richard E. Howard. 2018. Energy-Ball: Wireless Power Transfer for Batteryless Internet of Things through Distributed Beamforming. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 2 (June 2018), 65:1–65:22.

[33] Fluke. 2020. Fluke 87V Industrial Multimeter. https://www.fluke.com/en-us/product/electrical-testing/digital-multimeters/fluke-87v. (March 2020). Last accessed: Jun. 22, 2020.

[34] Fujitsu Semiconductor Ltd. 2018. MB85RS4MT 512 KB SPI FRAM. https://www.fujitsu.com/uk/Images/MB85RS4MT.pdf. (2018). Last accessed: Apr. 25, 2020.

[35] Karthik Ganesan, Joshua San Miguel, and Natalie Enright Jerger. 2019. The What's Next Intermittent Computing Architecture. In *Proc. HPCA*. IEEE, Washington, DC, USA, 211–223.

[36] German Federal Minister for the Environment, Nature Conservation, and Nuclear Safety. 2020. Umweltpolitische Digitalagenda (in German). https://www.bmu.de/fileadmin/Daten_BMU/Pools/Broschueren/broschuere_digitalagenda_bf.pdf. (Feb. 2020). Last accessed: Apr. 28, 2020.

[37] Graham Gobieski, Brandon Lucia, and Nathan Beckmann. 2019. Intelligence Beyond the Edge: Inference on Intermittent Embedded Systems. In *Proc. ASPLOS*. ACM, Providence, RI, USA, 199–213.

[38] Lewis Gordon. 2019. The Environmental Impact of a PlayStation 4. https://www.theverge.com/2019/12/5/20985330/ps4-sony-playstation-environmental-impact-carbon-footprint-manufacturing-25-anniversary. (Dec. 2019). Last accessed: Apr. 28, 2020.

[39] Green Electronics Council. 2020. Electronic Product Environmental Assessment Tool. https://epeat.net. (April 2020). Last accessed: Apr. 30, 2020.

[40] Manoj Gulati, Farshid Salemi Parizi, Eric Whitmire, Sidhant Gupta, Shobha Sundar Ram, Amarjeet Singh, and Shwetak N. Patel. 2018. CapHarvester: A Stick-on Capacitive Energy Harvester Using Stray Electric Field from AC Power Lines. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 3 (Sept. 2018), 110:1–110:20.

[41] Mehrdad Hessar, Ali Najafi, and Shyamnath Gollakota. 2019. NetScatter: Enabling Large-Scale Backscatter Networks. In *Proc. NSDI*. USENIX, Boston, MA, USA, 271–283.

[42] Josiah Hester, Lanny Sitanayah, and Jacob Sorber. 2015. Tragedy of the Coulombs: Federating Energy Storage for Tiny, Intermittently-Powered Sensors. In *Proc. SenSys* (Nov. 1–4). ACM, Seoul, South Korea, 5–16.

[43] Josiah Hester and Jacob Sorber. 2017a. Flicker: Rapid Prototyping for the Batteryless Internet-of-Things. In *Proc. SenSys* (Nov. 6–8). ACM, Delft, The Netherlands, 19:1–19:13.

[44] Josiah Hester and Jacob Sorber. 2017b. The Future of Sensing is Batteryless, Intermittent, and Awesome. In *Proc. SenSys* (Nov. 6–8). ACM, Delft, The Netherlands, 21:1–21:6.

[45] Josiah Hester, Kevin Storer, and Jacob Sorber. 2017. Timely Execution on Intermittently Powered Batteryless Sensors. In *Proc. SenSys* (Nov. 6–8). ACM, Delft, The Netherlands, 17:1–17:13.

[46] Matthew Hicks. 2017. Clank: Architectural Support for Intermittent Computation. In *Proc. ISCA* (June 24–28). ACM, Toronto, ON, Canada, 228–240.

[47] Meng-Ju Hsieh, Jr-Ling Guo, Chin-Yuan Lu, Han-Wei Hsieh, Rong-Hao Liang, and Bing-Yu Chen. 2019. RFTouchPads: Batteryless and Wireless Modular Touch Sensor Pads Based on RFID. In *Proc. UIST*. ACM, New Orleans, LA, US, 999–1011.

[48] Meng-Ju Hsieh, Rong-Hao Liang, Da-Yuan Huang, Jheng-You Ke, and Bing-Yu Chen. 2018. RFIBricks: Interactive Building Blocks Based on RFID. In *Proc. CHI* (April 21–26). ACM, Montréal, QC, Canada, 1–10.

[49] Nick Huber. 2020. Internet of Things: Smart Cities Pick up the Pace. https://www.ft.com/content/140ae3f0-1b6f-11ea-81f0-0c253907d3e0. (Jan. 2020). Last accessed: May 7, 2020.

[50] Chanyou Hwang, Saumay Pushp, Changyoung Koh, Jungpil Yoon, Yunxin Liu, Seungpyo Choi, and Junehwa Song. 2017. RAVEN: Perception-aware Optimization of Power Consumption for Mobile Games. In *Proc. MobiCom*. ACM, Snowbird, UT, USA, 422–434.

[51] Kaori Ikematsu, Masaaki Fukumoto, and Itiro Siio. 2019. Ohmic-Sticker: Force-to-Motion Type Input Device for Capacitive Touch Surface. In *Proc. CHI* (May 4–9). ACM, Glasgow, Scotland, UK, LBW0223:1–LBW0223:6.

[27] Marc de Kruijf 和 Karthikeyan Sankaralingam. 2013.幂等代码生成：实现、分析与评估。在 *Proc. CGO*.ACM/IEEE, 中国深圳.
[28] Jasper de Winkel, Carlo Delle Donne, Kasım Sinan Yıldırım, Przemysław Pawełczak 和 Josiah Hester. 2020.间歇计算的可靠时间保持。在*Proc. ASPLOS*(3月16–20).ACM, 瑞士洛桑, 53—-67.
[29] Christine Dierk, Molly Jane Pearce Nicholas 和 Eric Paulos. 2018.AlterWear：用于机会交互的无电池可穿戴显示器。在*Proc. CHI*(4月21–26).ACM, 蒙特利尔, 魁北克省, 加拿大, 210:1–210:13.
[30] 经济学人智库. 2020.物联网商业指数2020：采用的重大变化。https://learn.arm.com/rs/714-XIJ-402/images/economist-iot-business-index-2020-arm.pdf. (2020年2月).最后访问时间：2020年5月7日.
[31] 克里斯蒂娜・埃德斯特伦（执行出版人）. 2020.地平线2020欧盟计划电池2030+：发明未来的可持续电池：研究需求与未来行动。https://battery2030.eu/digitalAssets/861/c_861350-l_1-k_roadmap-27-march.pdf. (2020年3月). 最后访问时间：2020年4月30日.
[32] Xiaoran Fan, Han Ding, Sugang Li, Michael Sanzari, Yanyong Zhang, Wade Trappe, Zhu Han, 和 Richard E. Howard. 2018. 能量球：通过分布式波束成形实现无电池物联网的无线电力传输。*ACM Interact. Mob. Wearable Ubiquitous Technol.*2, 2 (2018年6月), 65:1–65:22.
[33] 福禄克. 2020.福禄克 87V 工业万用表. https://www.fluke.com/en-us/product/electrical-testing/digital-multimeters/fluke-87v. (2020年3月). 最后访问时间: 2020年6月22日.
[34] 富士通半导体有限公司. 2018.MB85RS4MT 512 KB SPI FRAM. https://www.fujitsu.com/uk/Images/MB85RS4MT.pdf. (2018年). 最后访问时间: 2020年4月25日.
[35] Karthik Ganesan, Joshua San Miguel, 和 Natalie Enright Jerger. 2019.下一步间歇计算架构。在 *Proc. HPCA*中。IEEE，华盛顿特区，美国，211–223。
[36] 德国联邦环境、自然保护和核安全部长。2020年。环境政策数字议程（德语）。https://www.bmu.de/fileadmin/Daten_BMU/Pools/Broschueren/broschuere_digitalagenda_bf.pdf。（2020年2月）。最后访问时间：2020年4月28日。
[37] 格雷厄姆・戈比斯基，布兰登・露西亚和内森・贝克曼。2019年。边缘智能：在间歇嵌入式系统上的推理。在*Proc. ASPLOS*中。ACM，普罗维登斯，罗德岛，美国，199–213。
[38] 刘易斯・戈登。 2019年。PlayStation 4的环境影响。https://www.theverge.com/2019/12/5/20985330/ps4-sony-playstation-environmental-impact-carbon-footprint-manufacturing-25-anniversary. (2019年12月). 最后访问时间：2020年4月28日。
[39] 绿色电子委员会. 2020年.电子产品环境评估工具. https://epeat.net. (2020年4月). 最后访问时间：4月. 30, 2020.
[40] Manoj Gulati, Farshid Salemi Parizi, Eric Whitmire, Sidhant Gupta, Shobha Sundar Ram, Amarjeet Singh, 和 Shwetak N. Patel. 2018年. CapHarvester: 一种使用交流电力线的杂散电场的粘贴式电容能量收集器.*ACM* 互动. 移动可穿戴普适技术.2, 3 (2018年9月), 110:1–110:20.
[41] Mehrdad Hessar, Ali Najafi, 和 Shyamnath Gollakota. 2019.NetScatter: 启用大规模反向散射网络. 在*Proc. NSDI*. USENIX, 波士顿, MA, 美国, 271–283.
[42] Josiah Hester, Lanny Sitanayah, 和 Jacob Sorber. 2015.库仑的悲剧: 为微型间歇性供电传感器联合能源存储. 在*Proc. SenSys*(11月 1–4).ACM, 首尔, 韩国, 5–16.
[43] Josiah Hester 和 Jacob Sorber. 2017a.Flicker: 无电池物联网的快速原型制作. 在*Proc. SenSys*(11月 6–8). ACM, 代尔夫特, 荷兰, 19:1–19:13.
[44] Josiah Hester 和 Jacob Sorber. 2017b.未来的传感技术是无电池的、间歇性的，并且令人惊叹。在*Proc. SenSys*(11月6–8日)。ACM，荷兰代尔夫特，21:1–21:6。
[45] Josiah Hester, Kevin Storer 和 Jacob Sorber. 2017.在间歇供电的无电池传感器上及时执行。在*Proc. SenSys* (11月6–8日)。ACM，荷兰代尔夫特，17:1–17:13。
[46] Matthew Hicks. 2017.Clank: 间歇计算的架构支持。在*Proc. ISCA*(6月24–28日)。ACM, 多伦多, 安大略省, 加拿大, 228–240.
[47] Meng-Ju Hsieh, Jr-Ling Guo, Chin-Yuan Lu, Han-Wei Hsieh, Rong-Hao Liang 和 Bing-Yu Chen. 2019.RFTouchPads：基于RFID的无电池无线模块化触摸传感器垫。在*Proc. UIST*中。ACM, 新奥尔良, LA, 美国, 999–1011.
[48] 孟居慧, 梁荣浩, 黄大源, 柯政佑, 和 陈炳宇. 2018.RFIBricks：基于RFID的互动积木。在*Proc. CHI*中（4月21–26日）。ACM, 蒙特利尔, QC, 加拿大, 1–10.
[49] 尼克・胡伯. 2020.物联网：智慧城市加速发展。https://www.ft.com/content/140ae3f0-1b6f-11ea-81f0-0c253907d3e0.（2020年1月）。最后访问时间：2020年5月7日.
[50] 黄昌佑, Pushp Saumay, 高昌勇, 尹正必, 刘云鑫, 崔承炳, 和 宋俊辉. 2017. RAVEN：移动游戏的感知感知优化电力消耗。在*Proc. MobiCom*.ACM，犹他州斯诺布莱，422–434.
[51] Ikematsu Kaori, Fukumoto Masaaki, 和 Siio Itiro. 2019.Ohmic-Sticker：用于电容触摸表面的力-运动类型输入设备。在*Proc. CHI*（5月4–9日）。ACM，苏格兰格拉斯哥，LBW0223:1–LBW0223:6.

[52] Jittrapol Intarasirisawat, Cheesiang Ang, Christos Efstratiou, Luke William Feidhlim Dickens, and Rupert Page. 2019. Exploring the Touch and Motion Features in Game-Based Cognitive Assessments. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 3 (Sept. 2019), 87:1–87:25.

[53] Christos Ioannou, Patrick Archard, Eamonn O'Neill, and Christof Lutteroth. 2019. Virtual Performance Augmentation in an Immersive Jump & Run Exergame. In *Proc. CHI* (May 4–9). ACM, Glasgow, Scotland, UK, 158:1–158:15.

[54] Vikram Iyer, Elyas Bayati, Rajalakshmi Nandakumar, Arka Majumdar, and Shyam Gollakota. 2017. Charging a Smartphone Across a Room Using Lasers. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 4 (Dec. 2017), 143:1–143:21.

[55] Ravi Jain and John Wullert II. 2002. Challenges: Environmental Design for Pervasive Computing Systems. In *Proc. MobiCom* (Sept. 23–28). ACM, Atlanta, GA, USA, 263–270.

[56] Japan Display Inc. 2016. LPM013M126A 1.28" MIP Reflective Color LTPS TFT LCD. https://www.j-display.com/product/pdf/Datasheet/4LPM013M126A_specification_Ver02.pdf. (2016). Last accessed: Apr. 25, 2020.

[57] Hrishikesh Jayakumar, Arnab Raha, Woo Suk Lee, and Vijay Raghunathan. 2015. Quickrecall: A HW/SW Approach for Computing Across Power Cycles in Transiently Powered Computers. *J. Emerg. Technol. Comput. Syst.* 12, 1 (July 2015), 8:1–8:19.

[58] Asangi Jayatilaka, Quoc Hung Dang, Shengjian Jammy Chen, Renuka Visvanathan, Christophe Fumeaux, and Damith C. Ranasinghe. 2019. Designing Batteryless Wearables for Hospitalized Older People. In *Proc. ISWC*. ACM, London, UK, 91–95.

[59] Haojian Jin, Jingxian Wang, Zhijian Yang, Swarun Kumar, and Jason Hong. 2018. WiSh: Towards a Wireless Shape-aware World using Passive RFIDs. In *Proc. MobiSys*. ACM, Munich, Germany, 428–441.

[60] Kumara Kahatapitiya, Chamod Weerasinghe, Jinal Jayawardhana, Hiranya Kuruppu, Kanchana Thilakarathna, and Dileeka Dias. 2018. Low-power Step Counting Paired with Electromagnetic Energy Harvesting for Wearables. In *Proc. ISWC* (Oct. 8–12). ACM, Singapore, 218–219.

[61] Mustafa Emre Karagozler, Ivan Poupyrev, Gary K. Fedder, and Yuri Suzuki. 2013. Paper Generators: Harvesting Energy from Touching, Rubbing and Sliding. In *Proc. UIST*. ACM, St. Andrews, UK, 23–30.

[62] Keiko Katsuragawa, Ju Wang, Ziyang Shan, Ningshan Ouyang, Omid Abari, and Daniel Vogel. 2019. Tip-Tap: Battery-free Discrete 2D Fingertip Input. In *Proc. UIST*. ACM, New Orleans, LA, US, 1045–1057.

[63] Bryce Kellogg, Aaron Parks, Shyamnath Gollakota, Joshua R. Smith, and David Wetherall. 2014. Wi-Fi Backscatter: Internet Connectivity for RF-Powered Devices. In *Proc. SIGCOMM*. ACM, Chicago, IL, USA, 607–618.

[64] Ben Kenwright. 2012. Fast Efficient Fixed-Size Memory Pool: No Loops and No Overhead. In *Proc. Computation Tools*. IARIA, Nice, France.

[65] Azam Khan. 2011. Swimming Upstream in Sustainable Design. *Interactions* 18, 5 (Sept. 2011), 12—-14.

[66] Yoojung Kim, Arpita Bhattacharya, Julie A. Kientz, and Jin Ha Lee. 2020. "It Should Be a Game for Fun, Not Exercise": Tensions in Designing Health-Related Features for Pokémon GO. In *Proc. CHI* (April 25–30). ACM, Honolulu, HI, USA.

[67] Bran Knowles, Lynne Blair, Mike Hazas, and Stuart Walker. 2013. Exploring Sustainability Research in Computing: Where we Are and Where we go Next. In *Proc. UbiComp* (Sept. 8–12). ACM, Zurich, Switzerland, 305–314.

[68] Vito Kortbeek, Kasım Sinan Yıldırım, Abu Bakar, Jacob Sorber, Josiah Hester, and Przemysław Pawełczak. 2020. Time-sensitive Intermittent Computing Meets Legacy Software. In *Proc. ASPLOS* (March 16–20). ACM, Lausanne, Switzerland, 85-–99.

[69] Andre Lam, Ivan Talev, and Jennifer Kim. 2020. Design life-Cycle: University of California, Davis, CA, USA, Department of Design's Undergraduate Students Project Designed by Christina Cogdell. http://www.designlife-cycle.com/nintendo-switch. (2020). Last accessed: Apr. 30, 2020.

[70] Seulki Lee, Bashima Islam, Yubo Luo, and Shahriar Nirjon. 2019. Intermittent Learning: On-Device Machine Learning on Intermittently Powered System. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 4 (Dec. 2019), 141:1–141:30.

[71] Dong Li, Feng Ding, Qian Zhang, Run Zhao, Jinshi Zhang, and Dong Wang. 2017. TagController: A Universal Wireless and Battery-free Remote Controller using Passive RFID Tags. In *Proc. MobiQuitous*. ACM, Melbourne, VIC, Australia, 166–175.

[72] Hanchuan Li, Eric Brockmeyer, Elizabeth J. Carter, Josh Fromm, Scott E. Hudson, Shwetak N. Patel, and Alanson Sample. 2016. PaperID: A Technique for Drawing Functional Battery-Free Wireless Interfaces on Paper. In *Proc. CHI* (May 7–12). ACM, San Jose, CA, USA, 5885–5896.

[73] Tianxing Li and Xia Zhou. 2018. Battery-Free Eye Tracker on Glasses. In *Proc. MobiCom* (October 29 — November 2). ACM, New Delhi, India, 67–82.

[74] Yichen Li, Tianxing Li, Xing-Dong Yang Ruchir A. Patel, and Xia Zhou. 2018. Self-Powered Gesture Recognition with Ambient Light. In *Proc. UIST*. ACM, Berlin, Germany, 595–608.

[75] Zhuying Li, Yan Wang, Wei Wang, Weikang Chen, Ti Hoang, Stefan Greuter, and Florian 'Floyd' Mueller. 2019. HeatCraft: Designing Playful Experiences with Ingestible Sensors via Localized Thermal Stimuli. In *Proc. CHI* (May 4–9). ACM, Glasgow, Scotland, UK, 576:1–576:12.

[76] Rong-Hao Liang, Meng-Ju Hsieh, Jheng-You Ke, Jr-Ling Guo, and Bing-Yu Chen. 2018. RFIMatch: Distributed Batteryless Near-Field Identification Using RFID-Tagged Magnet-Biased Reed Switches. In *Proc. UIST*. ACM, Berlin, Germany, 473–483.

[52] Jittrapol Intarasirisawat, Cheesiang Ang, Christos Efstratiou, Luke William Feidhlim Dickens, 和 Rupert Page. 2019.探索游戏基础的认知评估中的触摸和运动特征。*ACM Interact. Mob. Wearable Ubiquitous Technol.*3, 3 (2019年9月), 87:1–87:25.

[53] Christos Ioannou, Patrick Archard, Eamonn O'Neill, 和 Christof Lutteroth. 2019.沉浸式跳跃与跑步锻炼游戏中的虚拟表现增强. 在*Proc. CHI*(5月 4–9).ACM, 格拉斯哥, 苏格兰, 英国, 158:1–158:15.

[54] Vikram Iyer, Elyas Bayati, Rajalakshmi Nandakumar, Arka Majumdar, 和 Shyam Gollakota. 2017.通过激光在房间内为智能手机充电. *ACM Interact. Mob. Wearable Ubiquitous Technol.*1, 4 (2017年12月), 143:1–143:21.

[55] Ravi Jain 和 John Wullert II. 2002.挑战：普适计算系统的环境设计. 在*Proc. MobiCom* (9月 23–28).ACM，亚特兰大，乔治亚州，美国，263–270。

[56] 日本显示器公司。2016年。LPM013M126A 1.28" MIP反射彩色LTPS TFT LCD。https://www.j-display.com/product/pdf/Datasheet/4LPM013M126A_specification_Ver02.pdf。（2016年）。最后访问时间：2020年4月25日。

[57] Hrishikesh Jayakumar, Arnab Raha, Woo Suk Lee 和 Vijay Raghunathan。2015年。Quickrecall：一种在瞬态供电计算机中跨越电源周期的HW/SW方法。*J. Emerg. Technol. Comput. Syst.*12, 1（2015年7月），8:1–8:19。

[58] Asangi Jayatilaka, Quoc Hung Dang, Shengjian Jammy Chen, Renuka Visvanathan, Christophe Fumeaux 和 Damith C. Ranasinghe。2019年。为住院老年人设计无电池可穿戴设备。在*Proc. ISWC*。ACM，伦敦，英国，91–95。

[59] Haojian Jin, Jingxian Wang, Zhijian Yang, Swarun Kumar, 和 Jason Hong. 2018。WiSh：利用被动RFID迈向无线形状感知世界。在*Proc. MobiSys*。ACM，慕尼黑，德国，428–441。

[60] Kumara Kahatapitiya, Chamod Weerasinghe, Jinal Jayawardhana, Hiranya Kuruppu, Kanchana Thilakarathna, 和 Dileeka Dias. 2018。低功耗步数计与电磁能量收集相结合的可穿戴设备。在*Proc. ISWC*（10月8–12日）。ACM，新加坡，218–219。

[61] Mustafa Emre Karagozler, Ivan Poupyrev, Gary K. Fedder, 和 Yuri Suzuki. 2013。纸发电机：从触摸、摩擦和滑动中收集能量。在*Proc. UIST.* ACM, 圣安德鲁斯, 英国, 23–30.

[62] Katsuragawa, K., Wang, J., Shan, Z., Ouyang, N., Abari, O., 和 Vogel, D. 2019.Tip-Tap：无电池离散二维指尖输入。在*Proc. UIST.*ACM, 新奥尔良, 洛杉矶, 美国, 1045–1057.

[63] Kellogg, B., Parks, A., Gollakota, S., Smith, J. R., 和 Wetherall, D. 2014.Wi-Fi 反向散射：为 RF 驱动设备提供互联网连接。在*Proc. SIGCOMM.*ACM, 芝加哥, 伊利诺伊州, 美国, 607–618.

[64] Kenwright, B. 2012.快速高效的固定大小内存池：无循环且无开销。在计算工具的会议上。IARIA，法国尼斯。

[65] Azam Khan. 2011年。在可持续设计中逆流而上。交互18, 5 (2011年9月)，12—-14。

[66] Yoojung Kim, Arpita Bhattacharya, Julie A. Kientz 和 Jin Ha Lee. 2020年。"这应该是一个有趣的游戏，而不是锻炼"：设计 Pokémon GO 健康相关功能中的紧张关系。在*CHI* 会议上（4月25–30日）。ACM，夏威夷檀香山，美国。

[67] Bran Knowles, Lynne Blair, Mike Hazas 和 Stuart Walker. 2013年。探索计算中的可持续性研究：我们在哪里以及接下来要去哪里。在*UbiComp* 会议上（9月8–12日）。ACM，瑞士苏黎世，305–314。

[68] Vito Kortbeek, Kasım Sinan Yıldırım, Abu Bakar, Jacob Sorber, Josiah Hester 和 Przemysław Pawełczak. 2020。时间敏感间歇计算与传统软件相遇。在*Proc. ASPLOS*（3月16–20日）。ACM，瑞士洛桑，85–-99。

[69] Andre Lam, Ivan Talev 和 Jennifer Kim. 2020。设计生命周期：加利福尼亚大学戴维斯分校，CA，美国，设计系本科生项目，由 Christina Cogdell设计。http://www.designlife-cycle.com/nintendo-switch。（2020）。最后访问时间：2020年4月30日。

[70] Seulki Lee, Bashima Islam, Yubo Luo 和 Shahriar Nirjon. 2019。间歇学习：在间歇供电系统上的设备内机器学习。*ACM Interact. Mob. Wearable Ubiquitous Technol.*3, 4 (2019年12月), 141:1–141:30.

[71] Dong Li, Feng Ding, Qian Zhang, Run Zhao, Jinshi Zhang, 和 Dong Wang. 2017.TagController：一种使用被动RFID标签的通用无线无电池遥控器。在*Proc. MobiQuitous*中。ACM，澳大利亚维多利亚州墨尔本，166–175.

[72] Hanchuan Li, Eric Brockmeyer, Elizabeth J. Carter, Josh Fromm, Scott E. Hudson, Shwetak N. Patel, 和 Alanson Sample. 2016. PaperID: 在纸上绘制功能性无电池无线接口的技术。在*Proc. CHI*（5月7–12日）中。ACM, 圣荷西, 加利福尼亚州, 美国, 5885–5896.

[73] Tianxing Li 和 Xia Zhou. 2018.无电池眼动追踪器在眼镜上. 在*Proc. MobiCom*(10月29日 — 11月2日).ACM, 新德里, 印度, 67–82.

[74] Yichen Li, Tianxing Li, Xing-Dong Yang, Ruchir A. Patel, 和 Xia Zhou. 2018.自供电手势识别与环境光. 在*Proc. UIST*.ACM, 柏林, 德国, 595–608.

[75] Zhuying Li, Yan Wang, Wei Wang, Weikang Chen, Ti Hoang, Stefan Greuter, 和 Florian 'Floyd' Mueller. 2019.HeatCraft: 通过局部热刺激设计有趣的体验与可吞食传感器. 在*Proc. CHI*(5月4–9日).ACM, 格拉斯哥, 苏格兰, 英国,576:1–576:12.

[76] 梁荣浩, 谢孟如, 柯政佑, 郭志玲, 和 陈炳宇. 2018.RFIMatch: 使用RFID标签磁偏置簧开关的分布式无电池近场识别. 在*Proc. UIST.*ACM, 柏林, 德国, 473–483.

[77] Brandon Lucia, Vignesh Balaji, Alexei Colin, Kiwan Maeng, and Emily Ruppel. 2017. Intermittent Computing: Challenges and Opportunities. In *Proc. SNAPL* (May 7–10). Alisomar, CA, USA, 8:1–8:14.

[78] Brandon Lucia and Benjamin Ransford. 2015. A simpler, Safer Programming and Execution Model for Intermittent Systems. In *Proc. PLDI* (Aug. 13–17). ACM, Portland, OR, USA, 575–585.

[79] Dong Ma, Guohao Lan, Mahbub Hassan, Wen Hu, Mushfika Baishakhi Upama, Ashraf Uddin, and Moustafa Youssef. 2019. SolarGest: Ubiquitous and Battery-free Gesture Recognition using Solar Cells. In *Proc. MobiCom* (April 21–25). ACM, Los Cabos, Mexico, 12:1–12:15.

[80] Yunfei Ma, Nicholas Selby, and Fadel Adib. 2017. Drone Relays for Battery-Free Networks. In *Proc. SIGCOMM* (Aug. 21–25). ACM, Los Angeles, CA, USA, 335—-347.

[81] Kiwan Maeng, Alexei Colin, and Brandon Lucia. 2017. Alpaca: Intermittent Execution without Checkpoints. In *Proc. OOPSLA* (Oct. 22–27). ACM, Vancouver, BC, Canada, 96:1–96:30.

[82] Kiwan Maeng, Alexei Colin, and Brandon Lucia. 2018. Adaptive Dynamic Checkpointing for Safe Efficient Intermittent Computing. In *Proc. OSDI* (Oct. 8–10). USENIX, Carlsbad, CA, USA, 129–144.

[83] Kiwan Maeng and Brandon Lucia. 2019. Supporting Peripherals in Intermittent Systems with Just-in-time Checkpoints. In *Proc. PLDI*. ACM, Phoenix, AZ, USA, 1101–1116.

[84] Amjad Yousef Majid, Carlo Delle Donne, Kiwan Maeng, Alexei Colin, Kasım Sinan Yıldırım, Brandon Lucia, and Przemysław Pawełczak. 2020. Dynamic Task-based Intermittent Execution for Energy-harvesting Devices. *ACM Transactions on Sensor Networks* 16, 1 (Feb. 2020), 5:1–5:24.

[85] Jennifer C. Mankoff, Eli Blevis, Alan Borning, Batya Friedman, Susan R. Fussell, Jay Hasbrouck, Allison Woodruff, and Phoebe Sengers. 2007. Environmental Sustainability and Interaction. In *Proc. CHI* (April 28 – May 3). ACM, San Jose, CA, USA, 2121–2124.

[86] Gaia Maselli, Mauro Piva, Giorgia Ramponi, and Deepak Ganesan. 2016. Demo: JoyTag: a battery-less videogame controller exploiting RFID backscattering. In *Proc. MobiCom*. ACM, New York City, NY, USA, 515–516.

[87] Yogesh Kumar Meena, Krishna Seunarine, Deepak Ranjan Sahoo, Simon Robinson, Jennifer Pearson, Chi Zhang, Matt Carnie, Adam Pockett, Andrew Prescott, Suzanne K. Thomas, Harrison Ka Hin Lee, and Matt Jones. 2020. PV-Tiles: Towards Closely-Coupled Photovoltaic and Digital Materials for Useful, Beautiful and Sustainable Interactive Surfaces. In *Proc. CHI* (April 25–30). ACM, Honolulu, HI, USA.

[88] Hashan Roshantha Mendis and Pi-Cheng Hsiu. 2019. Accumulative Display Updating for Intermittent Systems. *ACM Transactions on Embedded Computing Systems* 18, 5s (Oct. 2019), 72:1–72:22.

[89] Geoff V. Merrett and Bashir M. Al-Hashimi. 2017. Energy-Driven Computing: Rethinking the Design of Energy Harvesting Systems. In *Proc. DATE*. IEEE, Lausanne, Switzerland, 960–965.

[90] Evan Mills, Norman Bourassa, Leo Rainer, Jimmy Mai, Arman Shehabi, and Nathaniel Mills. 2019. Toward Greener Gaming: Estimating National Energy Use and Energy Efficiency Potential. *The Computer Games Journal* 8 (Oct. 2019), 157–178.

[91] Florian 'Floyd' Mueller, Richard Byrne, Josh Andres, and Rakesh Patibanda. 2018. Experiencing the Body as Play. In *Proc. CHI* (April 21–26). ACM, Montréal, QC, Canada, 210:1–210:13.

[92] Saman Naderiparizi, Mehrdad Hessar, Vamsi Talla, Shyamnath Gollakota, and Joshua R. Smith. 2018. Towards Battery-Free HD Video Streaming. In *Proc. NSDI* (April 9–11). USENIX, Renton, WA, USA, 233–247.

[93] Saman Naderiparizi, Aaron N. Parks, Zerina Kapetanovic, Benjamin Ransford, and Joshua R. Smith. 2015a. WISPCam: A Battery-Free RFID Camera. In *Proc. IEEE RFID*. IEEE, San Diego, CA, USA, 166–173.

[94] Saman Naderiparizi, Yi Zhao, James Youngquist, Alanson P. Sample, and Joshua R. Smith. 2015b. Self-Localizing Battery-Free Cameras. In *Proc. UbiComp*. ACM, Osaka, Japan, 445–449.

[95] Yuji Nakamura. 2019. Peak Video Game? Top Analyst Sees Industry Slumping in 2019. https://www.bloomberg.com/news/articles/2019-01-23/peak-video-game-top-analyst-sees-industry-slumping-in-2019. (Jan. 2019). Last accessed: Jan. 7, 2020.

[96] Matteo Nardello, Harsh Desai, Davide Brunelli, and Brandon Lucia. 2019. Camaroptera: a Batteryless Long-Range Remote Visual Sensing System. In *Proc. ENSsys*. ACM, New York, NY, USA, 8–14.

[97] Phuc Nguyen, Ufuk Muncuk, Ashwin Ashok, Kaushik Roy Chowdhury, Marco Gruteser, and Tam Vu. 2016. Battery-Free Identification Token for Touch Sensing Devices. In *Proc. SenSys*. ACM, Stanford, CA, USA, 109–122.

[98] Nintendo Co., Ltd. 2019. Dedicated Video Game Sales Units. https://www.nintendo.co.jp/ir/en/finance/hard_soft/index.html. (Sept. 2019). Last accessed: Apr. 30, 2020.

[99] Nintendo Co., Ltd. 2020. Nintendo Game Boy. https://en.wikipedia.org/wiki/Game_Boy. (July 2020). Last accessed: Jul. 23, 2020.

[100] Sam Nussey and Christopher Cushing. 2020. Nintendo Seen Extending Profit Streak as Housebound Consumers Switch On. https://www.reuters.com/article/us-nintendo-results-preview/nintendo-seen-extending-profit-streak-as-housebound-consumers-switch-on-idUSKBN22C0B4. (April 2020). Last accessed: May 15, 2020.

[101] Netta Ofer, Idan David, Hadas Erel, and Oren Zuckerman. 2019. Coding for Outdoor Play: A Coding Platform for Children to Invent and Enhance Outdoor Play Experiences. In *Proc. CHI* (May 4–9). ACM, Glasgow, Scotland, UK, 1–12.

[77] 布兰登・露西亚, 维格内什・巴拉吉, 亚历克谢・科林, 麦基・万, 和 艾米莉・鲁佩尔. 2017.间歇计算: 挑战与机遇. 在*Proc. SNAPL*(5月7–10日).阿利索玛, 加利福尼亚, 美国, 8:1–8:14.
[78] 布兰登・露西亚 和 本杰明・兰斯福德. 2015.一种更简单、更安全的间歇系统编程和执行模型。在 *Proc. PLDI*（8月13–17日）。ACM，波特兰，俄勒冈州，美国，575–585。
[79] Dong Ma, Guohao Lan, Mahbub Hassan, Wen Hu, Mushfika Baishakhi Upama, Ashraf Uddin, 和 Moustafa Youssef. 2019. SolarGest: 使用太阳能电池的无处不在和无电池手势识别。在*Proc. MobiCom*（4月21–25日）。ACM，洛斯卡波斯，墨西哥，12:1–12:15。
[80] Yunfei Ma, Nicholas Selby, 和 Fadel Adib. 2017.用于无电池网络的无人机中继。在*Proc. SIGCOMM*（8月21–25日）。ACM, 洛杉矶, 加利福尼亚州, 美国, 335—-347.
[81] Kiwan Maeng, Alexei Colin, 和 Brandon Lucia. 2017.Alpaca: 无检查点的间歇执行. 在*Proc. OOPSLA* (10月 22–27).ACM, 温哥华, BC, 加拿大, 96:1–96:30.
[82] Kiwan Maeng, Alexei Colin, 和 Brandon Lucia. 2018.安全高效的间歇计算的自适应动态检查点技术. 在 *Proc. OSDI* (10月 8–10).USENIX, 卡尔斯巴德, CA, 美国, 129–144.
[83] Kiwan Maeng 和 Brandon Lucia. 2019.在间歇系统中支持外设的及时检查点. 在*Proc. PLDI*. ACM, 凤凰城, AZ, 美国, 1101–1116.
[84] Amjad Yousef Majid, Carlo Delle Donne, Kiwan Maeng, Alexei Colin, Kasım Sinan Yıldırım, Brandon Lucia, 和 Przemysław Pawełczak. 2020.基于动态任务的间歇性执行用于能量收集设备。*ACM*传感器网络学报16, 1 (2020年2月), 5:1–5:24.
[85] Jennifer C. Mankoff, Eli Blevis, Alan Borning, Batya Friedman, Susan R. Fussell, Jay Hasbrouck, Allison Woodruff, 和 Phoebe Sengers. 2007年.环境可持续性与交互。在*Proc. CHI*(4月28日 – 5月3日).ACM, 圣荷西, 加利福尼亚州, 美国, 2121–2124.
[86] Gaia Maselli, Mauro Piva, Giorgia Ramponi, 和 Deepak Ganesan. 2016年.演示：JoyTag：一种利用RFID反向散射的无电池视频游戏控制器。在*Proc. MobiCom*.ACM, 纽约市, 纽约州, 美国, 515–516.
[87] Yogesh Kumar Meena, Krishna Seunarine, Deepak Ranjan Sahoo, Simon Robinson, Jennifer Pearson, Chi Zhang, Matt Carnie, Adam Pockett, Andrew Prescott, Suzanne K. Thomas, Harrison Ka Hin Lee, 和 Matt Jones. 2020.PV-Tiles: 朝着紧密耦合的光伏和数字材料，创造有用、美观和可持续的互动表面. 在*Proc. CHI*(4月 25–30). ACM, 檀香山, HI, 美国.
[88] Hashan Roshantha Mendis 和 Pi-Cheng Hsiu. 2019.间歇性系统的累积显示更新.*ACM* 嵌入式计算系统交易18, 5s (2019年10月), 72:1–72:22.
[89] Geoff V. Merrett 和 Bashir M. Al-Hashimi. 2017.能量驱动计算：重新思考能量收集系统的设计。在 *Proc. DATE*.IEEE，瑞士洛桑，960–965。
[90] Evan Mills, Norman Bourassa, Leo Rainer, Jimmy Mai, Arman Shehabi, 和 Nathaniel Mills. 2019.朝向更绿色的游戏：估算国家能源使用和能源效率潜力。计算机游戏杂志8 (2019年10月)，157–178。
[91] Florian 'Floyd' Mueller, Richard Byrne, Josh Andres, 和 Rakesh Patibanda. 2018.体验身体作为游戏。在*Proc. CHI* (4月21–26日)。ACM, 蒙特利尔, 魁北克省, 加拿大, 210:1–210:13.
[92] Saman Naderiparizi, Mehrdad Hessar, Vamsi Talla, Shyamnath Gollakota, 和 Joshua R. Smith. 2018.朝向无电池高清视频流。在*Proc. NSDI*(4月9–11日)。USENIX, 雷顿, 华盛顿州, 美国, 233–247.
[93] Saman Naderiparizi, Aaron N. Parks, Zerina Kapetanovic, Benjamin Ransford, 和 Joshua R. Smith. 2015a.WISPCam: 一种无电池RFID相机. 在*Proc. IEEE RFID*中.IEEE, 圣地亚哥, 加利福尼亚州, 美国, 166–173.
[94] Saman Naderiparizi, Yi Zhao, James Youngquist, Alanson P. Sample, 和 Joshua R. Smith. 2015b.自定位无电池相机. 在*Proc. UbiComp*中.ACM, 大阪, 日本, 445–449.
[95] Yuji Nakamura. 2019. 顶尖分析师认为视频游戏行业在衰退?顶尖分析师看到2019年行业下滑.https://www.bloomberg.com/news/articles/2019-01-23/peak-video-game-top-analyst-sees-industry-slumping-in-2019. (2019年1月). 最后访问时间: 2020年1月7日.
[96] Matteo Nardello, Harsh Desai, Davide Brunelli, 和 Brandon Lucia. 2019.Camaroptera: 一种无电池的长距离远程视觉传感系统。在*Proc. ENSsys*.ACM, 纽约, NY, 美国, 8–14.
[97] Phuc Nguyen, Ufuk Muncuk, Ashwin Ashok, Kaushik Roy Chowdhury, Marco Gruteser, 和 Tam Vu. 2016.无电池识别令牌用于触摸传感设备。在*Proc. SenSys*.ACM, 斯坦福, CA, 美国, 109–122.
[98] 任天堂株式会社. 2019.专用视频游戏销售单位. https://www.nintendo.co.jp/ir/en/finance/hard_soft/index.html. (2019年9月). 最后访问时间: 2020年4月30日.
[99] 任天堂株式会社. 2020. 任天堂游戏男孩. https://en.wikipedia.org/wiki/Game_Boy. (2020年7月). 最后访问时间：2020年7月23日.
[100] 山姆纳斯西和克里斯托弗卡辛 . 2020. 任天堂预计延续盈利增长作为居家 消费者 开关 开启. https://www.reuters.com/article/us-nintendo-results-preview/nintendo-seen-extending-profit-streak-as-housebound-consumers-switch-on-idUSKBN22C0B4. (2020年4月). 最后访问时间：2020年5月15日.
[101] 内塔・奥费尔, 伊丹・大卫, 哈达斯・埃雷尔, 和 奥伦・祖克曼. 2019.户外游戏编程：一个让儿童创造和增强户外游戏体验的编程平台. 在*Proc. CHI*(5月4–9日).ACM，格拉斯哥，苏格兰，英国，1–12。

[102] Kazuya Oharada, Buntarou Shizuki, and Shin Takahashi. 2017. AccelTag: A Passive Smart ID Tag With an Acceleration Sensor for Interactive Applications. In *Proc. UIST Adjunct.* ACM, Québec City, Canada, 63–64.

[103] Panasonic Electric Works Europe AG. 2019. AM-1417CA Amorphous Silicon Solar Cell. https://www.panasonic-electric-works.com/cps/rde/xbcr/pew_eu_en/ca_amorton_solar_cells_en.pdf. (2019). Last accessed: Apr. 25, 2020.

[104] Panic Inc. 2020. Playdate Console Home Page. https://play.date. (March 2020). Last accessed: May 2, 2020.

[105] Aaron N. Parks, Angli Liu, Shyamnath Gollakota, and Joshua R. Smith. 2014. Turbocharging Ambient Backscatter Communication. In *Proc. SIGCOMM.* ACM, Chicago, IL, USA, 619–630.

[106] Matthai Philipose, Joshua R. Smith, Bing Jiang, Alexander Mamishev, Sumit Roy, and Kishor Sundara-Rajan. 2005. Battery-Free Wireless Identification and Sensing. *IEEE Pervasive Comput.* 4, 1 (Jan.–Mar. 2005), 37–45.

[107] R. Venkatesha Prasad, Shruti Devasenapathy, Vijay S. Rao, and Javad Vazifehdan. 2014. Reincarnation in the Ambiance: Devices and Networks with Energy Harvesting. *IEEE Commun. Surveys Tuts.* 11, 1 (First Quarter 2014), 195–213.

[108] United Nations Environment Programme. 2020. Playing for the Planet Consortium. https://playing4theplanet.org. (April 2020). Last accessed: Apr. 28, 2020.

[109] Benjamin Ransford, Jacob Sorber, and Kevin Fu. 2011. Mementos: System Support for Long-running Computation on RFID-scale Devices. In *Proc. ASPLOS* (March 5–11). ACM, Newport Beach, CA, USA, 159–170.

[110] Emily Ruppel and Brandon Lucia. 2019. Transactional Concurrency Control for Intermittent, Energy-Harvesting Computing Systems. In *Proc. PLDI.* ACM, Phoenix, AZ, USA, 1085–1100.

[111] Kimiko Ryokai, Peiqi Su, Eungchan Kim, and Bob Rollins. 2014. EnergyBugs: Energy Harvesting Wearables for Children. In *Proc. CHI* (Apr. 26 – May 1). ACM, Toronto, ON, Canada, 1039–1048.

[112] Ali Saffari, Mehrdad Hessar, Saman Naderiparizi, and Joshua R. Smith. 2019. Battery-Free Wireless Video Streaming Camera System. In *Proc. RFID.* IEEE, Phoenix, AZ, USA.

[113] Saleae. 2020. Saleae Logic Pro 8 Analyzer. http://downloads.saleae.com/specs/Logic+Pro+8+Data+Sheet.pdf. (2020). Last accessed: Jun. 22, 2020.

[114] Alanson P. Sample, Daniel J. Yeager, Pauline S. Powledge, Alexander V. Mamishev, and Joshua R. Smith. 2008. Design of an RFID-based battery-free programmable sensing platform. *IEEE Trans. Instrum. Meas.* 57, 11 (Nov. 2008), 2608–2615.

[115] Takuya Sasatani, Chouchang Jack Yang, Matthew J. Chabalko, Yoshihiro Kawahara, and Alanson P. Sample. 2018. Room-Wide Wireless Charging and Load-Modulation Communication via Quasistatic Cavity Resonance. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 4 (Dec. 2018), 188:1–188:23.

[116] Seth Schiesel. 2020. For the Uninitiated and Bored, an Introduction to the World of Gaming. https://www.nytimes.com/2020/04/01/arts/gaming-introduction-basics-quarantine-coronavirus.html. (April 2020). Last accessed: Apr. 28, 2020.

[117] Oliver Schneider, Jotaro Shigeyama, Robert Kovacs, Thijs Roumen, Sebastian Marwecki, Nico Boeckhoff, Daniel Amadeus Gloeckner, Jonas Bounama, and Patrick Baudisch. 2018. DualPanto: A Haptic Device that Enables Blind Users to Continuously Interact with Virtual Worlds. In *Proc. UIST.* ACM, Berlin, Germany, 877–887.

[118] Jason Schreier. 2020. Gaming Sales Are Up, but Production Is Down. https://www.nytimes.com/2020/04/21/technology/personaltech/coronavirus-video-game-production.html. (April 2020). Last accessed: Apr. 29, 2020.

[119] Semiconductor Components Industries LLC. 2017. NSR1030QMUTWG Schottky Full Diode Bridge. https://www.onsemi.com/pub/Collateral/NSR1030QMU-D.PDF. (2017). Last accessed: Apr. 25, 2020.

[120] Semtech. 2012. SX1503 4/8/16 Channel Low Voltage GPIO with NINT and NRESET. https://www.semtech.com/products/smart-sensing/io-expanders/sx1503. (2012). Last accessed: Apr. 25, 2020.

[121] Rishi Shukla, Neev Kiran, Rui Wang, Jeremy Gummeson, and Sunghoon Ivan Lee. 2019. SkinnyPower: Enabling Batteryless Wearable Sensors via Intra-Body Power Transfer. In *Proc. SenSys* (Nov. 10–13). ACM, New York City, NY, USA, 55–67.

[122] Philip Sparks. 2017. *The Route to a Trillion Devices: The Outlook for IoT investment to 2035.* Technical Report. ARM Limited. https://pages.arm.com/rs/312-SAX-488/images/Arm-The-route-to-trillion-devices_2018.pdf.

[123] STMicroelectronics. 2018. X-NUCLEO-LPM01A Nucleo Expansion Board for Power Consumption Measurement. https://www.st.com/en/evaluation-tools/x-nucleo-lpm01a.html. (March 2018). Last accessed: Jun. 22, 2020.

[124] Kazunobu Sumiya, Takuya Sasatani, Yuki Nishizawa, Kenji Tsushio, Yoshiaki Narusue, and Yoshihiro Kawahara. 2019. Alvus: A Reconfigurable 2-D Wireless Charging System. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 2 (June 2019), 68:1–68:29.

[125] Vamsi Talla, Mehrdad Hassar, Bryce Kellogg, Ali Najafi, Joshua R. Smith, and Shyam Gollakota. 2017a. LoRa Backscatter: Enabling the Vision of Ubiquitous Connectivity. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 3 (Sept. 2017), 105:1–105:24.

[126] Vamsi Talla, Bryce Kellogg, Shyamnath Gollakota, and Joshua R Smith. 2017b. Battery-free Cellphone. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 2 (June 2017), 25:1–25:19.

[127] Haydn Taylor. 2020. COVID-19: The State of the Games Industry. https://www.gamesindustry.biz/articles/2020-04-09-covid-19-the-state-of-the-games-industry. (April 2020). Last accessed: Apr. 28, 2020.

[128] Texas Instruments Inc. 2013. BQ25570 Ultra Low power Harvester power Management IC with Boost Charger, and Nanopower Buck Converter. https://www.ti.com/lit/ds/symlink/bq25570.pdf. (2013). Last accessed: Apr. 25, 2020.

[102] 大原和也，静木文太郎，和高桥信。2017年。AccelTag：一种具有加速度传感器的被动智能ID标签，用于互动应用。在*Proc. UIST Adjunct*。ACM，魁北克市，加拿大，63–64。
[103] 松下电器工业欧洲公司。2019年。AM-1417CA 非晶硅太阳能电池。https://www.panasonic-electric-works.com/cps/rde/xbcr/pew_eu_en/ca_amorton_solar_cells_en.pdf。（2019年）。最后访问时间：2020年4月25日。
[104] Panic Inc. 2020年。Playdate 控制台主页。https://play.date。（2020年3月）。最后访问时间：2020年5月2日。
[105] Aaron N. Parks, Angli Liu, Shyamnath Gollakota, 和 Joshua R. Smith. 2014.增强环境反向散射通信. 在*Proc. SIGCOMM*.ACM, 芝加哥, IL, 美国, 619–630.
[106] Matthai Philipose, Joshua R. Smith, Bing Jiang, Alexander Mamishev, Sumit Roy, 和 Kishor Sundara-Rajan. 2005.无电池无线识别与传感.*IEEE Pervasive Comput.*4, 1 (2005年1月–3月), 37–45.
[107] R. Venkatesha Prasad, Shruti Devasenapathy, Vijay S. Rao, 和 Javad Vazifehdan. 2014.环境中的重生：具有能量收集的设备和网络.*IEEE Commun. Surveys Tuts.*11, 1 (2014年第一季度), 195–213.
[108] 联合国环境规划署. 2020.为地球而玩联盟. https://playing4theplanet.org. (2020年4月). 最后访问时间: 2020年4月28日.
[109] 本杰明・兰斯福德, 雅各布・索伯, 和凯文・傅. 2011.纪念品: RFID规模设备上长期计算的系统支持. 在*Proc. ASPLOS*(3月5–11日). ACM, 新港海滩, 加利福尼亚州, 美国, 159–170.
[110] 艾米莉・鲁佩尔 和 布兰登・露西亚. 2019.间歇性能量收集计算系统的事务并发控制. 在*Proc. PLDI*.ACM, 凤凰城, 亚利桑那州, 美国, 1085–1100.
[111] 艾米子・良井, 隋佩琪, 金英灿, 和 鲍勃・罗林斯. 2014.EnergyBugs：儿童能量收集可穿戴设备。在*Proc. CHI*（4月26日至5月1日）。ACM， 加拿大多伦多， 1039–1048。
[112] Ali Saffari, Mehrdad Hessar, Saman Naderiparizi 和 Joshua R. Smith. 2019。无电池无线视频流摄像头系统。在*Proc. RFID*。 IEEE， 美国凤凰城。
[113] Saleae. 2020。Saleae Logic Pro 8 分析仪。 http://downloads.saleae.com/specs/Logic+Pro+8+Data+Sheet.pdf. （2020）。最后访问时间：6月。22, 2020.
[114] Alanson P. Sample, Daniel J. Yeager, Pauline S. Powledge, Alexander V. Mamishev 和 Joshua R. Smith. 2008。基于RFID的无电池可编程传感平台的设计。*IEEE Trans. Instrum. Meas.*57, 11（2008年11月），2608–2615。
[115] 佐佐谷拓也，张超常，马修・J・查巴尔科，川原嘉宏，阿兰森・P・样本。2018年。房间范围内的无线充电和负载调制通信通过准静态腔共振。*ACM*互动。移动可穿戴普及技术。2, 4 (2018年12月)，188:1–188:23。
[116] 塞斯・谢斯尔。2020年。对于初学者和无聊者，游戏世界的介绍。https://www.nytimes.com/2020/04/01/arts/gaming-introduction-basics-quarantine-coronavirus.html. (2020年4月)。最后访问时间：2020年4月28日。
[117] 奥利弗・施奈德，乔塔罗・重山，罗伯特・科瓦奇斯，蒂斯・鲁门，塞巴斯蒂安・马尔维基，尼科・博克霍夫，丹尼尔・阿马德乌斯・格洛克纳，乔纳斯・布纳马，帕特里克・鲍迪施。2018年。DualPanto：一种使盲人用户能够持续与虚拟世界互动的触觉设备。在*Proc. UIST*中。ACM，德国柏林，877–887。
[118] 杰森・施雷尔。2020年。游戏销售上升，但生产下降。https://www.nytimes.com/2020/04/21/technology/personaltech/coronavirus-video-game-production.html。（2020年4月）。最后访问时间：2020年4月29日。
[119] 半导体组件工业有限责任公司。2017年。NSR1030QMUTWG肖特基全桥二极管。https://www.onsemi.com/pub/Collateral/NSR1030QMU-D.PDF。（2017年）。最后访问时间：2020年4月25日。
[120] Semtech。2012年。SX1503 4/8/16通道低电压GPIO，带NINT和NRESET。https://www.semtech.com/products/smart-sensing/io-expanders/sx1503。（2012年）。最后访问时间：2020年4月25日。
[121] Rishi Shukla, Neev Kiran, Rui Wang, Jeremy Gummeson 和 Sunghoon Ivan Lee. 2019.SkinnyPower: 通过体内电力传输实现无电池可穿戴传感器. 在*Proc. SenSys*(11月 10–13).ACM，纽约市，纽约州，美国，55–67。
[122] Philip Sparks. 2017. 通往万亿设备的路线：*2035*年物联网投资展望. 技术报告. ARM有限公司. https://pages.arm.com/rs/312-SAX-488/images/Arm-The-route-to-trillion-devices_2018.pdf.
[123] STMicroelectronics. 2018.X-NUCLEO-LPM01A Nucleo 扩展板用于功耗测量. https://www.st.com/en/evaluation-tools/x-nucleo-lpm01a.html. (2018年3月). 最后访问时间：2020年6月22日.
[124] 角信清, 佐谷拓也, 西泽幸, 津潮健司, 成末义明, 和川弘志. 2019.Alvus: 一种可重构的二维无线充电系统.*ACM* 交互. 移动可穿戴普适技术.3, 2 (2019年6月), 68:1–68:29.
[125] Vamsi Talla, Mehrdad Hassar, Bryce Kellogg, Ali Najafi, Joshua R. Smith, 和 Shyam Gollakota. 2017a.LoRa 反向散射: 实现普适连接的愿景.*ACM* 交互. 移动可穿戴普适技术.1, 3 (2017年9月), 105:1–105:24.
[126] Vamsi Talla, Bryce Kellogg, Shyamnath Gollakota, 和 Joshua R Smith. 2017b. 无电池手机.*ACM* 交互. 移动可穿戴普适技术.1, 2 (2017年6月), 25:1–25:19.
[127] 海登・泰勒. 2020. COVID-19：游戏产业的现状。 https://www.gamesindustry.biz/articles/2020-04-09-covid-19-the-state-of-the-games-industry. (2020年4月). 最后访问时间：2020年4月28日。
[128] 德州仪器公司. 2013.BQ25570 超低功耗收集器电源管理IC，带升压充电器和纳米功率降压转换器。 https://www.ti.com/lit/ds/symlink/bq25570.pdf. (2013年). 最后访问时间：2020年4月25日。

[129] Texas Instruments Inc. 2016. TPS61099 0.7 V_{in} Synchronous Boost Converter with 800 nA Ultra-Low Quiescent Current. http://www.ti.com/lit/ds/symlink/tps61099.pdf. (2016). Last accessed: Apr. 25, 2020.

[130] Texas Instruments Inc. 2017. MSP430FR59xx Mixed-Signal Microcontrollers (Rev. F). http://www.ti.com/lit/ds/symlink/msp430fr5969.pdf. (March 2017). Last accessed: May 1, 2020.

[131] Bill Tomlinson, M. Six Silberman, Don Patterson, Yue Pan, and Eli Blevis. 2012. Collapse Informatics: Augmenting the Sustainability & ICT4D Discourse in HCI. In *Proc. CHI*. ACM, Austin, TX, USA, 655–664.

[132] Hoang Truong, Shuo Zhang, Ufuk Muncuk, Phuc Nguyen, Nam Bui, Anh Nguyen, Qin Lv, Kaushik Chowdhury, Thang Dinh, and Tam Vu. 2018. CapBand: Battery-free Successive Capacitance Sensing Wristband for Hand Gesture Recognition. In *Proc. SenSys* (Nov. 4–7). ACM, Shenzhen, China, 54–67.

[133] UNI-T. 2020. UT383 Mini Light Meter. https://www.uni-trend.com/html/product/Environmental/Environmental_Tester/Mini/UT383.html. (2020). Last accessed: Jun. 22, 2020.

[134] University of Michigan, MI, USA. 2011. UMich Moo GitHub Page. https://github.com/spqr/umichmoo. (Feb. 2011). Last accessed: Apr. 19, 2020.

[135] University of Washington, Seattle, WA, USA. 2010. Wireless Identification and Sensing Platform GitHub Page. https://github.com/wisp. (Nov. 2010). Last accessed: Apr. 19, 2020.

[136] Valve Inc. 2020. Source 3D Game Engine. https://developer.valvesoftware.com/wiki/source. (March 2020). Last accessed: May 2, 2020.

[137] Joel Van Der Woude and Matthew Hicks. 2016. Intermittent Computation Without Hardware Support or Programmer Intervention. In *Proc. OSDI* (Nov. 2–4). ACM, Savannah, GA, USA, 17–32.

[138] Virag Varga, Gergely Vakulya, Alanson Sample, and Thomas R. Gross. 2017a. Enabling Interactive Infrastructure with Body Channel Communication. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 4 (Dec. 2017), 169:1–169:.

[139] Virag Varga, Gergely Vakulya, Alanson Sample, and Thomas R. Gross. 2017b. Playful Interactions with Body Channel Communication: Conquer it!. In *Proc. UIST Adjunct*. ACM, Québec City, Canada, 81–82.

[140] Anandghan Waghmare, Qiuyue Xue, Dingtian Zhang, Yuhui Zhao, Shivan Mittal, Nivedita Arora, Ceara Byrne, Thad Starner, and Gregory D. Abowd. 2020. UbiquiTouch: Self Sustaining Ubiquitous Touch Interfaces. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 1 (March 2020), 27:1–27:22.

[141] Ju Wang, Liqiong Chang, Omid Abari, and Srinivasan Keshav. 2019a. Are RFID Sensing Systems Ready for the Real World?. In *Proc. MobiSys*. ACM, Seoul, Korea, 366–377.

[142] Jingxian Wang, Chengfeng Pan, Haojian Jin, Vaibhav Singh, Yash Jain, Jason Hong, Carmel Majidi, and Swarun Kumar. 2019b. RFID Tattoo: A Wireless Platform for Speech Recognition. *ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 4 (Dec. 2019), 155:1–155:24.

[143] Katelyn Wiley, Sarah Vedress, and Regan Mandryk. 2020. How Points and Theme Affect Performance and Experience in a Gamified Cognitive Task. In *Proc. CHI* (April 25–30). ACM, Honolulu, HI, USA.

[144] Harrison Williams, Xun Jian, and Matthew Hicks. 2020. Forget Failure: Exploiting SRAM Data Remanence for Low-overhead Intermittent Computation. In *Proc. ASPLOS* (March 16–20). ACM, Lausanne, Switzerland, 53—-67.

[145] Allison Woodruff, Jay Hasbrouck, and Sally Augustin. 2008. A Bright Green Perspective on Sustainable Choices. In *Proc. CHI*. ACM, Florence, Italy, 313–322.

[146] Chenren Xu, Lei Yang, and Pengyu Zhang. 2018a. Practical Backscatter Communication Systems for Battery-Free Internet of Things. *IEEE Signal Process. Mag.* 35, 5 (Sept. 2018), 16–27.

[147] Jian Xu, Suwen Zhu, Aruna Balasubramanian, Xiaojun Bi, and Roy Shilkrot. 2018b. Ultra-Low-Power Mode for Screenless Mobile Interaction. In *Proc. UIST*. ACM, Berlin, Germany, 557–568.

[148] Xieyang Xu, Yang Shen, Junrui Yang, Chenren Xu, Guobin Shen, Guojun Chen, and Yunzhe Ni. 2017. PassiveVLC: Enabling Practical Visible Light Backscatter Communication for Battery-free IoT Applications. In *Proc. MobiCom* (Oct. 16–20). ACM, Snowbird, UT, USA, 180–192.

[149] Wataru Yamada, Hiroyuki Manabe, and Daizo Ikeda. 2018. CamTrackPoint: Camera-Based Pointing Stick Using Transmitted Light through Finger. In *Proc. UIST*. ACM, Berlin, Germany, 313–320.

[150] Kasım Sinan Yıldırım, Amjad Yousef Majid, Dimitris Patoukas, Koen Schaper, Przemysław Pawełczak, and Josiah Hester. 2018. InK: Reactive Kernel for Tiny Batteryless Sensors. In *Proc. SenSys* (Nov. 4–7). ACM, Shenzhen, China, 41–53.

[151] Neng-Hao Yu, Sung-Sheng Tsai, I-Chun Hsiao, Dian-Je Tsai, Meng-Han Lee, Mike Y. Chen, and Yi-Ping Hung. 2011. Clip-on Gadgets: Expanding Multi-touch Interaction Area with Unpowered Tactile Controls. In *Proc. UIST* (Oct. 16–19). ACM, Santa Barbara, CA, USA, 367–371.

[152] Tom Zeller Jr. 2010. Green Guide to Electronics Is Disputed, but Influential. https://www.nytimes.com/2010/01/11/business/energy-environment/11green.html. (Jan. 2010). Last accessed: Apr. 30, 2020.

[153] ZF Friedrichshafen AG. 2015. AFIG-0007 Energy Harvesting Generator. https://switches-sensors.zf.com/us/wp-content/uploads/sites/7/2019/11/19_10_16-TS_AFIG-0007.pdf [specification], https://switches-sensors.zf.com/product/energy-harvesting-generators/ [summary]. (2015). Last accessed: Apr. 25, 2020.

[129] 德州仪器公司. 2016.TPS61099 0.7 V同步升压转换器，超低静态电流为800 nA。 http: //www.ti.com/lit/ds/symlink/tps61099.pdf. (2016年). 最后访问时间：2020年4月25日。
[130] 德州仪器公司 2017年。MSP430FR59xx 混合信号微控制器（修订版 F）。http://www.ti.com/lit/ds/symlink/msp430fr5969.pdf。（2017年3月）。最后访问时间：2020年5月1日。
[131] Bill Tomlinson, M. Six Silberman, Don Patterson, Yue Pan, 和 Eli Blevis. 2012年。Collapse Informatics: 增强可持续性与ICT4D在HCI中的话语。在*Proc. CHI*。ACM, 奥斯丁, 德克萨斯州, 美国, 655–664。
[132] Hoang Truong, Shuo Zhang, Ufuk Muncuk, Phuc Nguyen, Nam Bui, Anh Nguyen, Qin Lv, Kaushik Chowdhury, Thang Dinh, 和 Tam Vu. 2018年。CapBand: 无电池的连续电容传感腕带用于手势识别。在*Proc. SenSys*（11月4–7日）。ACM, 深圳, 中国, 54–67.
[133] UNI-T. 2020. UT383迷你光度计. https://www.uni-trend.com/html/product/Environmental/Environmental_Tester/Mini/UT383.html. (2020). 最后访问时间: 2020年6月22日.
[134] 密歇根大学, MI, 美国. 2011. UMich Moo GitHub页面. https://github.com/spqr/umichmoo. (2011年2月). 最后访问时间: 4月. 19, 2020.
[135] 华盛顿大学, 西雅图, WA, 美国. 2010.无线识别与传感平台GitHub页面. https://github.com/wisp. (2010年11月). 最后访问时间: 2020年4月19日.
[136] Valve Inc. 2020. Source 3D游戏引擎. https://developer.valvesoftware.com/wiki/source. (2020年3月).最后访问时间：2020年5月2日。
[137] Joel Van Der Woude 和 Matthew Hicks. 2016.没有硬件支持或程序员干预的间歇性计算。在 *Proc. OSDI* (11月 2–4日)。ACM, 萨凡纳, GA, 美国, 17–32.
[138] Virag Varga, Gergely Vakulya, Alanson Sample 和 Thomas R. Gross. 2017a.通过身体通道通信实现互动基础设施。*ACM Interact. Mob. Wearable Ubiquitous Technol.*1, 4 (2017年12月), 169:1–169:.
[139] Virag Varga, Gergely Vakulya, Alanson Sample 和 Thomas R. Gross. 2017b.与身体通道通信的有趣互动：征服它！在 *Proc. UIST Adjunct*.ACM, 魁北克市, 加拿大, 81–82.
[140] Anandghan Waghmare, Qiuyue Xue, Dingtian Zhang, Yuhui Zhao, Shivan Mittal, Nivedita Arora, Ceara Byrne, Thad Starner, 和 Gregory D. Abowd. 2020.UbiquiTouch: 自我维持的普适触控接口。*ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 1 (2020年3月), 27:1–27:22.
[141] Ju Wang, Liqiong Chang, Omid Abari, 和 Srinivasan Keshav. 2019a.RFID传感系统准备好进入现实世界了吗？在 *Proc. MobiSys*中。ACM, 韩国首尔, 366–377.
[142] Jingxian Wang, Chengfeng Pan, Haojian Jin, Vaibhav Singh, Yash Jain, Jason Hong, Carmel Majidi, 和 Swarun Kumar. 2019b.RFID纹身：一种用于语音识别的无线平台。*ACM Interact. Mob. Wearable Ubiquitous Technol.*3, 4 (2019年12月)，155:1–155:24。
[143] 凯特琳・威利，莎拉・维德雷斯，和瑞根・曼德里克。2020年。点数和主题如何影响游戏化认知任务中的表现和体验。在*Proc. CHI*（4月25日至30日）。ACM，檀香山，夏威夷，美国。
[144] 哈里森・威廉姆斯，薛健，和马修・希克斯。2020年。忘记失败：利用SRAM数据残留进行低开销间歇计算。在*Proc. ASPLOS*（3月16日至20日）。ACM, 瑞士洛桑, 53—-67.
[145] 艾莉森・伍德拉夫，杰伊・哈斯布鲁克，和莎莉・奥古斯丁。2008年。可持续选择的明亮绿色视角。在*Proc. CHI*。ACM, 意大利佛罗伦萨, 313–322.
[146] 许晨任, 杨磊, 张鹏宇. 2018a.用于无电池物联网的实用反向散射通信系统. *IEEE*信号处理杂志35, 5 (2018年9月), 16–27.
[147] 许剑, 朱苏文, 阿鲁纳・巴拉苏布拉曼尼安, 毕晓军, 和 罗伊・希尔克罗特. 2018b.无屏幕移动交互的超低功耗模式. 在*Proc. UIST*中.ACM, 德国柏林, 557–568.
[148] 许谢阳, 沈扬, 杨俊瑞, 许晨任, 沈国斌, 陈国军, 和 倪云哲. 2017.PassiveVLC: 为无电池物联网应用启用实用的可见光反向散射通信. 在*Proc. MobiCom*中 (10月16–20日).ACM, 雪鸟, 犹他州, 美国,180–192.
[149] 山田航, 真部裕之, 和池田大造. 2018.CamTrackPoint: 基于相机的指点杆，利用透过手指的光线. 在*Proc. UIST*中.ACM, 柏林, 德国, 313–320.
[150] 卡斯姆・西南・伊尔迪里姆, 阿姆贾德・尤塞夫・马吉德, 迪米特里斯・帕图卡斯, 科恩・沙佩尔, 普热梅斯瓦夫・帕维尔恰克, 和乔赛亚・赫斯特. 2018. InK: 用于微型无电池传感器的反应内核. 在*Proc. SenSys*中 (11月 4–7).ACM, 深圳, 中国, 41–53.
[151] 余能豪, 蔡松生, 萧奕君, 蔡典杰, 李孟汉, 陈美瑜, 和洪怡平. 2011. 夹式小工具: 扩展无电触控控制的多点触控交互区域。在*Proc. UIST*（10月16日至19日）。ACM，圣巴巴拉，加利福尼亚州，美国，367–371。
[152] 汤姆・泽勒 Jr. 2010。绿色电子指南受到争议，但影响深远。 https://www.nytimes.com/2010/01/11/business/energy-environment/11green.html. （2010年1月）。最后访问时间：2020年4月30日。
[153] ZF 弗里德里希港股份公司。2015。AFIG-0007 能量收集发电机。 https://switches-sensors.zf.com/us/wp-content/uploads/sites/7/2019/11/19_10_16-TS_AFIG-0007.pdf [规格]，https://switches-sensors.zf.com/product/energy-harvesting-generators/ [摘要]。（2015年）。最后访问时间：2020年4月25日。

[154] Chi Zhang, Sidharth Kumar, and Dinesh Bharadia. 2019b. Capterry: Scalable Battery-like Room-level Wireless Power. In *Proc. MobiSys*. ACM, Seoul, Korea, 1–13.

[155] Yang Zhang, Yasha Iravantchi, Haojian Jin, Swarun Kumar, and Chris Harrison. 2019a. Sozu: Self-Powered Radio Tags for Building-Scale Activity Sensing. In *Proc. UIST*. ACM, New Orleans, LA, USA, 973–985.

[156] Chen Zhao, Sam Yisrael, Joshua R. Smith, and Shwetak N. Patel. 2014. Powering Wireless Sensor Nodes with Ambient Temperature Changes. In *Proc. UbiComp* (Sept. 13–17). ACM, Seattle, WA, USA, 383––387.

[154] Chi Zhang, Sidharth Kumar 和 Dinesh Bharadia. 2019b.Capterry: 可扩展的类电池房间级无线电力. 在*Proc. MobiSys*. ACM, 首尔, 韩国, 1–13.
[155] Yang Zhang, Yasha Iravantchi, Haojian Jin, Swarun Kumar 和 Chris Harrison. 2019a.Sozu: 自供电无线标签用于建筑规模活动感知. 在*Proc. UIST*.ACM, 新奥尔良, LA, 美国, 973–985.
[156] Chen Zhao, Sam Yisrael, Joshua R. Smith 和 Shwetak N. Patel. 2014.利用环境温度变化为无线传感器节点供电 . 在*Proc. UbiComp*(9月 13–17).ACM, 西雅图, WA, 美国, 383-–387.