LEVERAGING BACKSCATTER FOR INTERNET OF THINGS

A Dissertation Presented

by

PENGYU ZHANG

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2016

College of Information and Computer Sciences

LEVERAGING BACKSCATTER FOR INTERNET OF THINGS

A Dissertation Presented

by

PENGYU ZHANG

Approved as to style and content by:

Deepak Ganesan, Chair

Prashant Shenoy, Member

Arun Venkataramani, Member

Dennis L. Goeckel, Member

James Allan, Chair
College of Information and Computer Sciences

ACKNOWLEDGMENTS

I would not have been able to complete this thesis without the guidance of my advisor and the support of my friends and family. Professor Deepak Ganesan taught me key skills needed for any researcher: how to identify and formulate a research problem, tackle a research problem from a unique angle, think critically about a potential solution, develop a big research vision, etc. I especially thank Deepak for his hands-on advising when I started my Ph.D. study and had little knowledge of what is good research. I also appreciate Deepak for reminding me to take one step back from details and looking at the problem again.

I also enjoyed working with Professor Prashant Shenoy, Arun Venkataramani, and Ben Marlin in such a diverse and collaborative department. I would like to thank Professor Kevin Fu for being my synthesis advisor. I was fortunate to have discussions with Professor Dennis Goeckel who gave me lots of feedback from the perspective of wireless communication theory.

I have collaborated with researchers outside of UMass while on an internship at Microsoft Research Redmond. I would like to thank Bodhi Priyantha, Jie Liu, and Matthai Philipose for working closely with me on the mobile vision project. They taught me how to think about a project from the perspective of industry research.

I would like to thank Jeremy Gummeson and Negin Salajegheh for sharing hands-on experiences about building systems and doing system research. I am grateful for being able to have five years' discussion with Bo Jiang about how to use probability theory to analyze system performance. I also thank Pan Hu for teaching me hardware design principles.

My life in Amherst would not be so much fun if I did not have incredible friends from sensors research group, CS department, and other departments at UMass. I would like to thank Tingxin Yan, Abhinav Parate, Addison Mayberry, Moaj Musthag, Annamalai Natarajan, Yamin Tun, and other folks who made my life in Amherst wonderful.

I finally want to thank my parents who raised me up and have been supporting me for years. I felt lucky to spend my graduate school with my wife, Shaofang Wang, together. I am grateful for having our beautiful daughter Elly Zhang born in the middle of my Ph.D. study. I am looking forward to continuing our journey together.

ABSTRACT

LEVERAGING BACKSCATTER FOR INTERNET OF THINGS

MAY 2016

PENGYU ZHANG

B.Sc., TSINGHUA UNIVERSITY, BEIJING, CHINA

M.Sc., TSINGHUA UNIVERSITY, BEIJING, CHINA

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Deepak Ganesan

The past few years have seen a dramatic growth in the Internet of Things (IoT), with millions of wirelessly connected sensors becoming first-class citizens of the Internet. The number of IoT devices is expected to surpass 6.75 billion by 2017, more than the world's population as well as the combined market of smartphones, tablets, and PCs. However, the growth of Internet of Things faces two pressing challenges: battery energy density and wireless radio power consumption. Battery energy density looms as a fundamental limiting factor due to slow improvements over the past several decades (3× over 22 years). Wireless radio power consumption is another key challenge because high-speed wireless communication is often far more expensive energy-wise than computation, storage or sensing, and IoT devices are generating an increasing amount of data.

These challenges raise a fundamental question — how should we power and communicate with IoT devices. More specifically, instead of using batteries, can we

leverage other energy sources to reduce, if not eliminate, the dependence on batteries? Similarly, instead of optimizing existing wireless radios, can we fundamentally change how radios transmit wireless signals to achieve lower power consumption?

A promising technique to address these questions is backscatter — a primitive that enables RF energy harvesting and ultra-low-power wireless communication. Backscatter has the potential to reduce dependence on batteries because it can obtain energy by rectifying the wireless signals transmitted by a backscatter reader. Backscatter can also work by reflecting existing wireless signals (WiFi, BLE) when these are available nearby. Because signal reflection only consumes μWs of power, backscatter can enable ultra-low-power wireless communication.

However, the use of backscatter for communicating with IoT devices presents several challenges. First, decreasing RF power across distance limits the operational range of micro-powered backscatter devices. This raises the question of how to maintain a communication link with a backscatter device despite tiny amount of harvested power. Second, even though the backscatter RF front-end is extremely power-efficient, the computational and sensing overhead on backscatter sensors limit its ability to operate with a few μWs of power. Such overhead is a negligible factor of overall power consumption for platforms where radio power consumption is high (e.g. WiFi or Bluetooth based devices). However, it becomes the bottleneck for backscatter based platforms. Third, backscatter readers are not currently deployed in existing indoor environments to provide a continuous carrier for carrying backscattered information. As a result, backscatter deployment is not yet widespread.

My thesis addresses these challenges by making the following contributions. First, we design a network enables continuous operation despite decreasing harvested power across distance by employing an OS abstraction — task fragmentation. We show that such a network stack enables packet transfer even when the whole system is powered by a 3cm$\times$3cm solar panel under natural indoor light condition. Second, we design

a hardware architecture that minimizes the computational overhead of backscatter to enable over 1Mbps backscatter transmission while consuming less than 100μWs of power, a two order of magnitude improvement over the state-of-the-art. Finally, we design a system that can leverage both ambient WiFi and BLE signals for backscatter. Our empirical evaluation shows that we can backscatter 350bps data on top of a WiFi stream and 3bps data on top of a BLE stream when the backscatter device is 2m away from the commercial WiFi and BLE receivers.

TABLE OF CONTENTS

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

The past few years have seen a dramatic growth in the Internet of Things, with millions of wirelessly connected sensors becoming first-class citizens of the Internet. Figure 1.1 shows that the number of IoT devices is expected to surpass 6.75 billion by 2017, more than the world's population as well as the combined market of smartphones, tablets, and PCs. However, the growth of Internet of Things faces two pressing challenges: battery energy density and wireless radio power consumption.

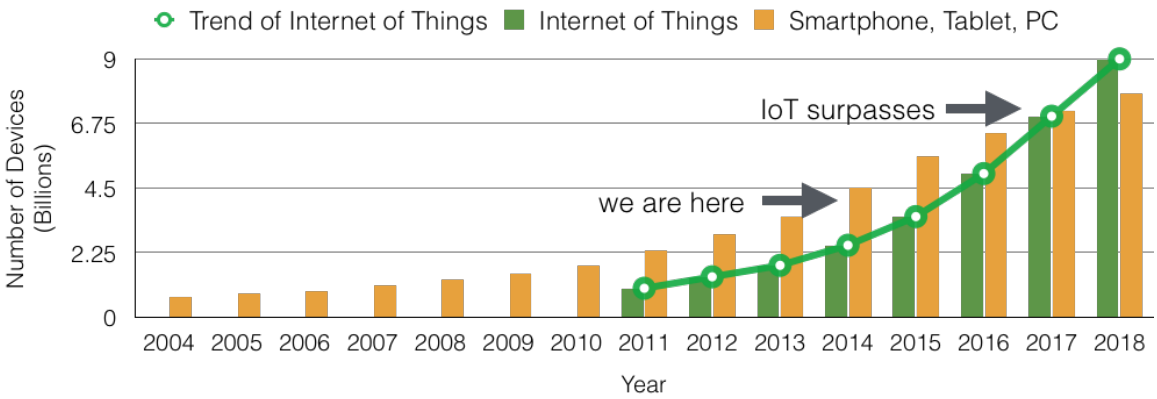


Figure 1.1. The Internet of Things alone will surpass the smartphone, tablet, and PC market combined by 2017 [1].

Battery energy density looms as a fundamental limiting factor in the Internet of Things particularly since improvements have been slow over the past several decades. Figure 1.2 shows that battery energy density has improved by only 3× over the past 22 years. Research forecasts, such as Forbes research, also identifies this limitation

and says "*Currently, connected devices, such as Pebble and Galaxy Gear, run on batteries, which have limited shelf life. Given current energy availability, powering these devices will be impossible. Prolonged battery life that sources energy from unconventional power sources is a must for future development for the Internet of Things*" [6].

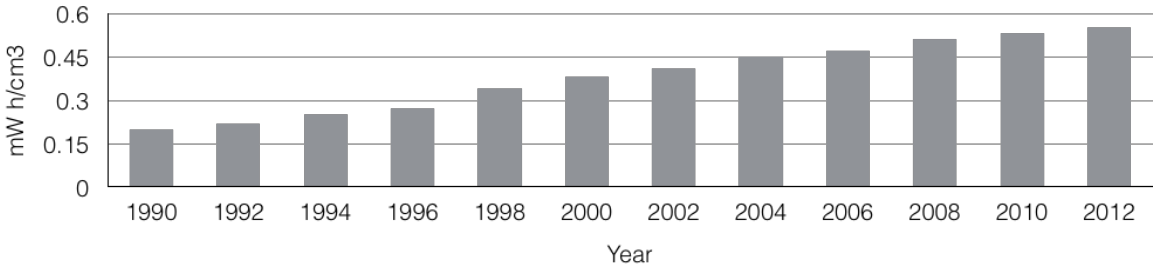


Figure 1.2. 3× battery energy density improvement from 1990 to 2012.

Wireless radio power consumption is another key challenge as high-speed wireless communication is often far more expensive energy-wise than computation, storage or sensing. Figure 1.3 shows the power consumption of a variety of wireless radios and other components of an IoT device. Even Bluetooth Low Energy (BLE), which has the lowest power consumption among these radios, consumes 15mW during active transmission, orders of magnitude higher than the power needed for computation (MSP430 MCU), storage (SRAM), or in many cases sensing (e.g. accelerometer). Therefore, there is a dire need to design novel wireless communication techniques to achieve higher data rates while simultaneously minimizing energy consumption.

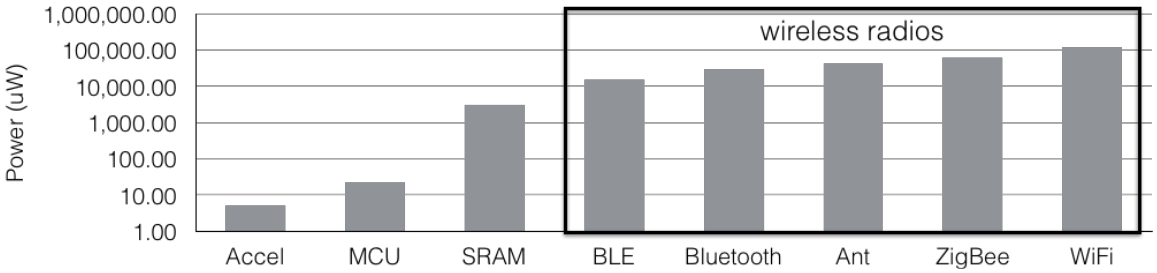


Figure 1.3. Power consumption of several low-power wireless radios.

These challenges raise a fundamental question — how should we power and communicate with IoT devices. More specifically, instead of using batteries, can we leverage other energy sources to reduce, if not eliminate, the dependence on batteries? Similarly, instead of optimizing existing wireless radios, can we fundamentally change how radios transmit wireless signals to achieve lower power consumption?

A possible answer to these questions is backscatter — a primitive that enables RF energy harvesting and ultra-low-power wireless communication. Backscatter has the potential to reduce the dependence on batteries because it can obtain energy by rectifying the wireless signals transmitted by a backscatter reader. In addition, instead of directly transmitting high-power wireless signals, backscatter modulates information by reflecting existing wireless signals. Because signal reflection only consumes μWs of power, backscatter has the potential to enable ultra-low-power and high-speed wireless connection for IoT devices.

However, the use of backscatter for the Internet of Things presents several challenges. First, backscatter RF power decreases sharply across the distance, which limits the operational range of micro-powered backscatter devices. This trend is a result of path loss and can be modeled with the Friis model shown in equation 1.1 (in logarithmic form). In this model, P_T is the transmit power of the reader, λ is the carrier wave length, G_T is the transmit antenna gain, G_R is the receive dipole antenna gain of the backscatter device, d is the distance between the reader and the backscatter device, and L_P is the polarization loss. The amount of RF power, P_R, available for harvesting decreases as the distance d increases. As a result, a micro-powered backscatter device does not have enough energy for operation at a longer distance even though the SNRs of both backscatter reader-to-tag and tag-to-reader links are still sufficient for data communication.

$$P_R = P_T - 20\log(\frac{4\pi d}{\lambda}) + G_T + G_R - L_P P_R \propto \frac{1}{d^2} \tag{1.1}$$

Second, computational overheads on backscatter sensors limit their ability to operate at μWs of harvested power. These overheads include acquiring data from sensors, migrating sensor data to the radio, and executing network protocols. These overheads are negligible on platforms where wireless communication is expensive (e.g. WiFi based sensors). However, because of the ultra-low power consumption of backscatter radios, they become the bottleneck on backscatter-based systems and increase power consumption while limiting throughput. Therefore, there is a dire need to systematically investigate the sources of these computational overheads and understand how to eliminate them.

Third, backscatter readers are not yet integrated into commonly used mobile and wearable devices nor deployed widely in urban settings. An alternative would be to leverage ambient signals (e.g. WiFi and BLE) that already exist. However, leveraging ambient signals for backscatter is hard primary because an ambient signal itself causes strong interference to a backscatter receiver. As a result, decoding weak reflected signal becomes harder. Another challenge comes from the bursty nature of an ambient signal where an ambient signal is not always available for carrying backscattered information. As a result, we cannot directly use a ambient signal receiver (e.g. WiFi or BLE receiver) to decode backscattered information.

1.2 Thesis Contribution

My thesis tackles these challenges and seeks to enable the practical adoption of backscatter for the Internet of Things.

1.2.1 Contribution Summary

Overall, the key systems and contributions of this thesis are:

- QuarkNet – A network stack that tackles the challenge of operating under decreasing harvested power across distance by employing task fragmentation. Our

network stack enables packet transfer even when the whole system is powered by a 3cm×3cm solar panel under natural indoor light.

- Ekho – A hardware architecture that minimizes the computational overheads of a backscatter system to enable ≥1Mbps backscatter transmission while only consuming ≤100μW of power, two orders of magnitude improvement over the state-of-the-art.

- A system that can leverage ambient WiFi and Bluetooth signals in an environment for carrying backscattered information. Our system can achieve 350bps and 3bps data rate when the backscatter device is 2m away from the commercial WiFi and Bluetooth receivers respectively.

1.2.2 Thesis Overview

Figure 1.4 shows an overview of the thesis. It includes three components to tackle the three key challenges of backscatter systems.

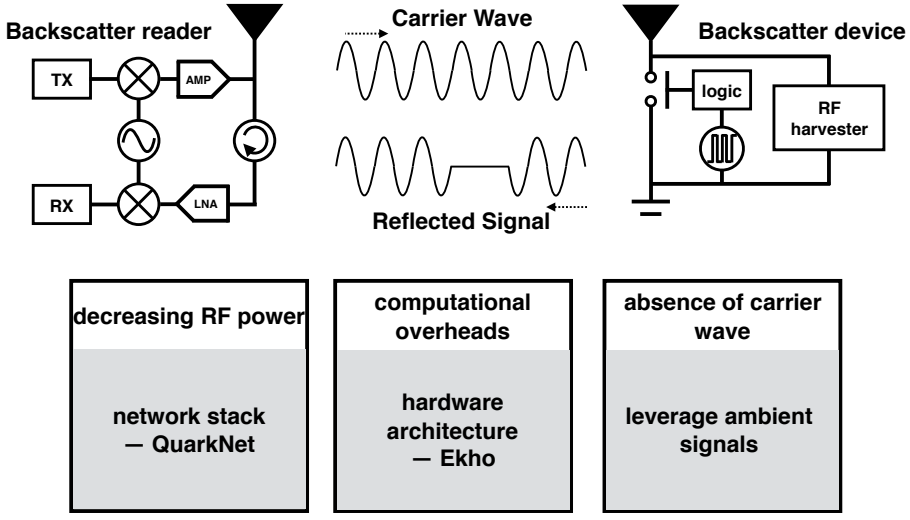


Figure 1.4. An overview of thesis.

1.2.3 Network Stack for Micro-powered Sensors — QuarkNet

In chapter 3, we present the design of a network stack to tackle the challenge of operating under decreasing harvested power across distance. Its design is based on the observation that communication often fails not because energy cannot be harvested but because packet transfer involves hundreds of instructions and cannot fit into the available energy budget. To address this problem, we develop a simple but powerful abstraction — by fragmenting any networking task into its smallest atomic units, we can enable the system to scale down to resource impoverished regimes. Our network stack enables packet transfer even when the whole system is powered by a 3cm×3cm solar panel under natural indoor light condition.

1.2.4 High Speed Ultra Low-power Backscatter — Ekho

In chapter 4, we design a hardware architecture to minimize the computational overheads on backscatter sensors for achieving high speed ultra low-power backscatter. A fundamental assumption that has driven the design of sensor networks for decades is that communication is the most power-hungry component of an individual sensor system. We argue that this assumption does not hold when it comes to passive radios such as backscatter, where communication is much cheaper energy-wise compared to computation. Therefore, we overturn the design principle governing wireless sensor design from one that focuses on minimizing communication to one focuses on optimizing the computational elements between the sensor and RF interface. We design a hardware sensing architecture that minimizes computational blocks between the sensors and the backscatter RF interface. We implement our architecture on an FPGA and show that we are able to achieve $\geq$1Mbps backscatter transmission while only consuming $\leq$100μW of power, a two order of magnitude improvement over the state of the art.

1.2.5 Leveraging Ambient Wireless Signals for Backscatter

In chapter 5, we introduce a system that can leverage ambient WiFi and Bluetooth signals in the environment for carrying backscattered information. A key feature of this system is that it can decode backscattered information simply with commercial WiFi and Bluetooth radios. By eliminating the need for deploying backscatter readers, our system provides a promising solution for deploying backscatter on existing IoT devices even though existing radios on IoT devices are not designed for backscatter. Decoding backscattered information on top of WiFi and Bluetooth signals is hard because the signal strength of WiFi and Bluetooth is usually several orders of magnitude higher than the reflected signal strength. To deal with such strong interference, we leverage FSK modulation to move the backscattered signals to adjacent clean channels where the interference from the primary WiFi or Bluetooth channel becomes smaller. Our empirical evaluation shows that we can achieve 350bps and 3bps data transmission when the backscatter device is 2m away from the commercial WiFi and BLE receivers.

CHAPTER 2

BACKGROUND

This thesis discusses how and why a backscatter device experiences limited operational range, has a significant amount of computational overhead, and cannot obtain a continuous carrier wave in an ambient environment. All these challenges prevent us from deploying backscatter devices in the nearby environment. To understand these challenges, we present background material on backscatter systems to set the context for our contributions. More detailed related work sections are also provided in the remaining chapters.

Our background introduction starts from looking at the overall system architecture, which gives us an overview of backscatter systems. We then turn to study the hardware architecture of a backscatter device, which sets the context about why existing backscatter devices have a significant amount of computational overhead. We also investigate the channel model of backscatter communication, which helps us understand why SNR is not the limiting factor of operational range. In the last section, we discuss protocols and reader configurations used by a commercial backscatter system — RFID, which inspires us to think why commercial backscatter readers are not yet widely deployed in the nearby environment. Let us now start from looking at an overview of backscatter systems.

2.1 Backscatter system overview

Figure 2.1 shows the architecture of a backscatter system. A backscatter reader, which has a form factor as large as an RFID reader or as small as a wearable device,

sends out a continuous carrier wave. The carrier wave travels through a certain distance and reaches a backscatter device. A portion of the carrier power is converted by the backscatter device into DC current and is stored locally as harvested energy. Another portion is reflected back to the backscatter reader by toggling an RF transistor. When the backscatter device toggles the RF transistor, the amount of reflected signal changes. Such changes can be detected by the reader and is interpreted as the information transmitted by the backscatter device.

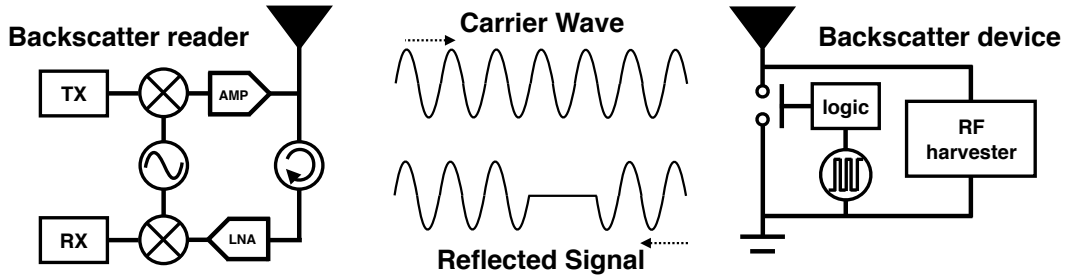


Figure 2.1. Backscatter system architecture.

2.2 Hardware overview

2.2.1 Backscatter radio analog RF front end

Backscatter radios are designed to enable ultra low power wireless communication. As shown in Figure 2.2, a reader provides a carrier wave, which can be modulated with information. To transmit data, a sensor toggles the state of a transistor to detune its antenna and reflect the carrier wave back to the reader with its information bits. Because the sensor does not actively generate an RF carrier signal unlike active radio systems, the power consumption of the backscatter radio is very low. In addition, the on-off transition overhead of backscatter radios is very low because backscatter radios do not have to warm up the RF analog circuits for data transmission, unlike active radio systems. As a result, there is little overhead incurred while transmitting via backscatter, even when transmitting at a high rate. For example, one key component

of the backscatter analog RF front end of the WISP [13] is a MOSFET transistor (BF1212WR). Its power consumption follows the equation of $\frac{1}{2}CV^2F$ where C is the capacitance of the transistor, V is the digital drain-source voltage, and F is the frequency of operating the transistor. When this transistor is toggled at a slow rate of 10Hz, it consumes 55pW of power, and even when toggled at a high rate of 1MHz, it only consumes 5.5μW of power. Thus, backscatter radios consume of the order of μWs of power, even for high rate data transfer.

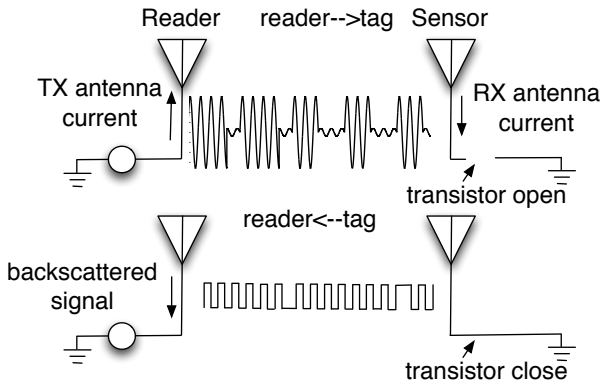


Figure 2.2. Backscatter communication analog RF front end.

2.2.2 RF energy harvesting

In addition to ultra low-power wireless communication, backscatter also enables wireless energy delivery. As shown in Figure 2.2, the backscatter reader provides a carrier wave, which can be rectified by the sensor to produce DC voltage. This voltage is boosted to an appropriate level by a charge pump at the sensor and accumulated in a small storage capacitor until the voltage reaches an appropriate threshold before any computation (or sensing) can begin. Once the voltage is sufficient to power the device, it can begin to receive and transmit data, both of which are done by modulating the same carrier wave.

Table 2.1. Parameters used for modeling bi-static and mono-static backscatter.

	Explanation
P_t	TX antenna transmit power
G_t	TX antenna gain
G_r	RX antenna gain
G_n	Sensor's backscatter antenna gain
λ	Wave length of carrier wave
D	Distance between transceiver and sensor
D_t	Distance between transmitter and sensor
D_r	Distance between receiver and sensor

2.3 Backscatter channel model

2.3.1 Backscatter link budget

Figure 2.3 shows the wireless channel link budget of bi-static and mono-static backscatter systems across distance. These two types of backscatter have different types of wireless channel link budget because they use different mechanisms for transmission and reception. For mono-static backscatter, TX and RX antennas are deployed in the same location or are hosted on the same object. In contrast, for bi-static backscatter, the deployment of TX and RX antennas are geophysically separated. The mathematical models of mono-static and bi-static backscatter are shown in equation 2.1 and equation 2.2 respectively. Table 2.1 summarizes the parameters used in the two models. For both models, the wireless link budget of backscatter decreases significantly even when the device is slightly further from the carrier wave transmitter because the strength of the backscattered signal decreases at the fourth power of distance.

$$P = \frac{P_t G_t G_r G_n}{(\frac{4\pi}{\lambda})^4 D^4} \tag{2.1}$$

$$P = \frac{P_t G_t G_r G_n}{(\frac{4\pi}{\lambda})^4 D_t^2 D_r^2} \tag{2.2}$$

To illustrate why we prefer bi-static backscatter, we consider the following scenario where a backscatter device moves away from a mono-static reader and a bi-static

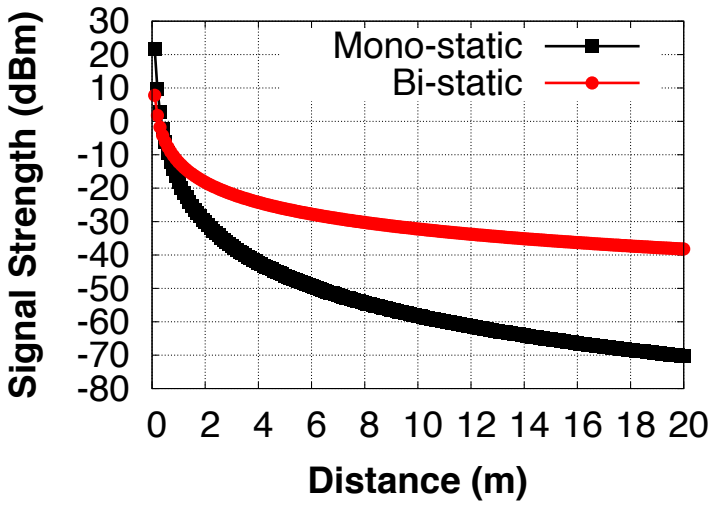


Figure 2.3. Link budget of bi-static and mono-static backscatter.

reader where the device is always 0.1m away from the RX antenna. We find two interesting observations from Figure 2.3. First, bi-static backscatter has higher link budget compared to mono-static backscatter when the sensor is more than 0.5m away from the carrier wave transmitter. This benefit comes from the geophysical separation between TX and RX antennas where one of them can be deployed close to the backscatter device. Second, distance has a larger impact on the signal strength of mono-static backscatter compared to a bi-static backscatter system. For example, at 2m, the signal strength of a mono-static backscatter system is 12 dBm lower than a bi-static backscatter system. To make matters worse, this gap becomes larger at a longer distance. At 10m, this gap is 26 dBm, much larger than the 2m case. Both observations suggest that we should use bi-static backscatter if possible.

2.3.2 Asymmetric forward and backward links

The forward (reader-to-sensor) and backward (sensor-to-reader) links in backscatter communication differ in several ways. First, the path loss is very different for the two links. The signal to noise ratio (SNR) for typical backscatter communication decays with the square of distance for the forward link and to the fourth power of dis-

tance for the backscatter link. Second, the encoding schemes for the links are different. In the EPC Gen 2 network stack, reader to sensor communications use pulse-interval encoding (PIE), which allows easy decoding, whereas sensor to reader communication uses more complex encodings (FM0, Miller2, Miller4, Miller8). Third, the antenna sensitivity at the sensor and reader are vastly different. A typical backscatter reader (e.g. Impinj [11]) uses a mono-static antenna for sending and receiving data, which has a sensitivity of -80 dBm. In contrast, an RFID-scale sensor (e.g. the Intel WISP [13]) uses a simple dipole antenna for data transfer, which is significantly less sensitive than the reader antenna. These factors contribute to different link qualities in the two directions. The forward link uses weaker encoding and is received by a less sensitive antenna, but has lower path loss. The backward link uses robust encoding and is received by a highly sensitive antenna, but has much higher path loss.

2.4 Commercial backscatter protocol — EPC Gen 2

2.4.1 EPC Gen 2 protocol

EPC Gen 2 is a communication protocol designed for RFIDs, one type of backscatter devices that have been widely used in industry for supply chain monitoring, localizing objects, etc. It is a TDMA based MAC control protocol and targets at reading RFID tags as fast as possible by minimizing the probability of collision. Figure 2.4 shows four types of typical messages exchanged between a reader and a backscatter devices — Query, RN16, ACK, and EPC ID. The communication process is initiated by the Query message sent by a reader, which contains information about the number of time slots assigned to RFID tags within the inventory range of the backscatter reader. An RFID tag randomly chooses a time slot and sends an RN16 message to the reader for verifying whether the chosen time slot is occupied or not. If not, the reader sends an ACK message to the device, which responds its EPC ID and finishes the

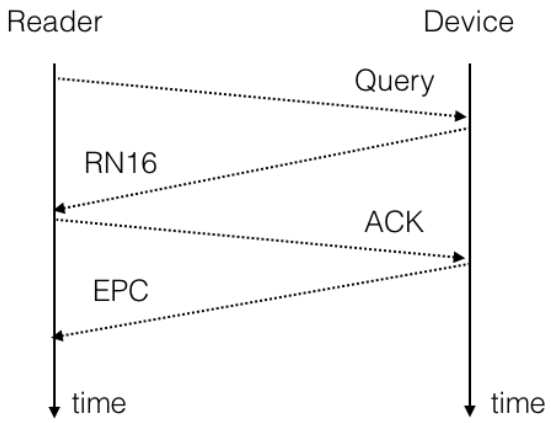


Figure 2.4. Messages exchanged between reader and device in EPC Gen 2 protocol.

communication process. If yes, instead of ACK, the reader sends an NAK message to the device to terminate its communication immediately.

2.4.2 Frequency hopping of backscatter readers

Backscatter readers use frequency hopping to avoid interference across readers when reading sensors in the same area. A typical UHF reader hops between 50 channels in the 902MHz $\sim$ 928MHz ISM band. FCC regulations specify that a reader can have a maximum channel dwell time of 0.4 seconds in any ten second period to reduce interference in a channel [19]. Commercial reader implementations address this by spending an equal amount of time in each of the 50 channels (0.2 secs) and hopping in sequence from the first to last channel. This is inefficient, and more intelligent choice of channels while remaining within FCC specifications can improve throughput.

CHAPTER 3

PUSHING THE OPERATING LIMITS OF MICRO-POWERED SENSORS

Existing micro-powered sensors make a slew of design choices that limit the ability to scale down to severe energy harvesting environment. In this chapter, we address this issue with QuarkNet, a network stack that is designed to enable continuous communication even if there is only enough harvested energy to transmit a few bits at a time.

3.1 Background and Motivation

The idea of networks of perpetual self-powered sensing, communication and actuation devices that can fly in swarms, swim through the bloodstream, and navigate through pipes and debris has propelled the imagination of science fiction writers for decades, but reality is finally catching up. While practical instantiations of self-powered devices have largely been limited to RFID tags, a new generation of micro-powered devices promises to go beyond simple identification towards computation, sensing, and actuation. Among the key technology trends enabling this vision are advances in micro-harvesters that scavenge energy from light, electro-magnetic waves, vibrations, temperature, and other sources [17]. Such micro-harvesters enable platforms to cut their reliance on stored energy in batteries, thereby enabling true miniaturization and perpetual operation [49, 51].

While micro-powered devices present an exciting opportunity, they present tremendous challenges due to the amount of energy they harvest and the sizes of their energy

reservoirs. The amount of harvested power using a micro-energy harvester is of the order of *nano*Watts to μWatts, which is three to six orders of magnitude lower than the average power draw of a Mote. At first glance, this seems to suggest that if we wait long enough, the device can trickle charge to accumulate sufficient energy to operate similar to a battery-powered device. But there are three problems. First, long delays before performing useful work are often unacceptable, particularly for continuous sensing and communication. Second, the voltage from the incoming energy source is often low, therefore accumulating energy into an energy reservoir requires boosting voltage which is wasteful compared to incoming energy (imagine pumping water up a hill to store for future use). Third, micro-powered platforms often have small energy reservoirs to reduce form-factor. For example, the Intel WISP [13] and Michigan Micro Mote (M^3) [37] have energy reservoirs that are 4 – 6 orders of magnitude smaller than a coin cell respectively.

The dual limitations of low harvesting rates and tiny energy reservoirs have profound implications on the design of a network stack for micro-powered devices. Every communication task needs to be small enough to fit within the available energy in the reservoir. Enabling communication despite such minuscule energy budgets is akin to working on a micro-sculpture — optimizations at the granularity of individual instructions, bits, on-off transitions, and analog-to-digital conversions are needed. To compound matters, small short-term variations in harvesting conditions that typically would be smoothed out by a larger energy reservoir begin to impact system operation, and can cause an order of magnitude variation in available energy for a task.

These challenges are not addressed by existing protocols such as EPC Gen 2. RFID tags operate solely on continuous harvested power without buffering energy, therefore EPC Gen 2 assumes a regime where the tag either has enough power to operate continuously, or not at all. In contrast, micro-powered devices can buffer

energy, thereby enabling operation in regimes where there is insufficient power to operate continuously, but enough power to operate intermittently.

Recent systems such as MementOS [43] and Dewdrop [20] tackle this problem in different ways. Both these systems use backscatter similar to RFIDs, but the challenge is fitting the communication stack within the energy budget. MementOS introduces checkpoints within computation tasks such that it can recover from outages and continue execution. Dewdrop continually adapts task execution to harvesting conditions such that the efficiency of execution is optimized. To evaluate the ability of these systems to scale down, we consider two harvesting conditions — strong light (2000 lux) and natural indoor light (200 lux), both of which should, in principle, provide enough energy to operate a micro-powered sensor. But while both Mementos and Dewdrop operate under strong light, they are inoperable under natural light.

The inability of current systems to scale-down illustrates the central challenge in designing a network stack for micro-powered devices. A wireless network stack involves a variety of tasks that are simply too large to fit into the extreme energy constraints of this regime. Even the core primitive of a network stack — packet transfer — can involve hundreds of instructions and bits. In this work we ask the following question — what are the general principles that we, as systems designers, should use to enable these micro-powered platforms to communicate continuously despite trickles of energy, tiny energy reservoirs, and dynamic harvesting conditions?

We present QuarkNet, a network stack that embodies a simple but powerful abstraction — by fragmenting a backscatter network stack into its smallest atomic units, we can enable the system to scale down to resource-impoverished regimes. The fundamental building block of QuarkNet is the ability to dynamically fragment a larger packet transfer into μframes that can be as small as a single bit under severe energy constraints, and as large as the whole packet when sufficient energy is available. On top of this abstraction, we design a variety of innovative techniques to handle

dynamic frames that can be abruptly terminated in low energy settings, maximize throughput by tracking harvesting dynamics in a low-overhead manner, interleave μframes across nodes to maximize throughput despite different harvesting rates, and minimize overhead across the entire stack.

Our results on a USRP reader and Moo nodes show that:

- The maximum communication distance achieved by QuarkNet is 21 feet, 3.5× longer than Dewdrop and 4.2× longer than EPC ID transfer. QuarkNet achieves close to the maximum achievable range, beyond which decoding even a single bit fails.

- The minimum illuminance required for QuarkNet to operate is 150 lux, which is 13× lower than the 2000 lux requirement of 12 byte EPC ID transfer. This suggests that μframe can operate when a device is powered by natural indoor illuminance, dramatically increasing utility of micro-powered devices for practical deployments.

- The throughput of QuarkNet for node to reader transfer is 18 kbps, 10.5× higher than EPC Gen 2, 5.8× higher than Dewdrop, and 3.3× higher than Flit. For reader to node transfer, we obtain throughput of 1.5 kbps, 2× higher than a battery-assisted device which uses the EPC Gen 2 write command.

- When ten nodes transmit simultaneously to a reader, we achieve a throughput of 16.5 kbps as a result of variability-aware scheduling and interleaving of μframes, which is 5.4× higher than the throughput when devices are inventoried individually. Flit and EPC Gen 2 obtain zero throughput in this case.

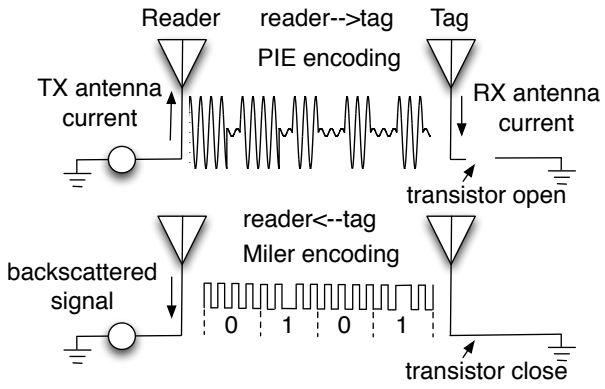


Figure 3.1. Backscatter signaling at PHY.

3.2 Case for μframes

A backscatter radio is designed to both provide power to a passive device as well as to enable communication. As shown in Figure 3.1, the reader provides a carrier wave, which can be reflected by a passive device back to the reader with its own information bits. This makes backscatter a considerably more energy-efficient communication mechanism compared to active radios, and ideally suited to the constraints of micro-powered devices. The Intel WISP [13] and UMass Moo [53] are examples of backscatter-enabled sensor platforms.

Despite the energy benefits of backscatter radios, existing network stacks achieve only short communication range and low throughput. We make an empiric argument these limitations are, in part, due to the design of the network stack. To do this, we compare the range and throughput of existing network stacks versus achievable performance. Our experiment uses a UMass Moo [53] and a USRP reader [21]. Since combining multiple micro-power sources can enable higher performance, broader operating conditions, and enable wider range of applications, we augment the Moo with a small solar panel [26, 17, 25]. We vary the distance from the reader by small steps, and at each step, we vary RF power from 17dBm to 26dBm, while not changing the light levels (normal indoor light).

Table 3.1. EPC Gen 2 vs Achievable Performance.

	Range(ft)	Throughput(kbps)	SNR(dB)
Gen 2	3.6±0.8	3.6±0.3	9.6±1
Optimal	18.6±3.3	21.7±3.7	6.9±0.9

To measure the achievable range, we look at the raw backscattered signal at the reader, and find the distance at which the reader is unable to decode even a single bit. This would be the edge of the communication range for our hardware platform.

Measuring the maximum achievable throughput is harder since it is influenced by several system parameters including voltage at the energy reservoir when communication starts, the length of each transmission unit, and control overheads associated with the protocol. We brute-force search across all possible voltages and packet lengths to find the setting that results in the maximum number of transistor flips at the node. We then convert the transistor flips to a maximum number of bits transmitted using the default Miller-4 encoding scheme, and assume zero control overhead for each packet, which gives us an estimate of the maximum throughput.

Table 3.1 shows the range and throughput while executing the EPC Gen 2 stack (used in Mementos [43], Dewdrop [20], and Blink [57]) versus achievable limits. We see that the achievable range is 18.6 feet, which is over 5× longer than the communication range of EPC Gen 2. Surprisingly, we find that EPC Gen 2 ceases to operate even when its SNR is 9.6dB, 1.4× higher than the optimal case. Similarly, we see that the achievable throughput is 21.7 kbps, whereas EPC Gen 2 achieves barely 1.7 kbps, an order of magnitude difference.

We now investigate the fundamental factors underlying this performance gap, and outline the core challenges that need to be addressed to bridge the gap.

Challenge 1: Variable energy per transmission: A key challenge in designing a backscatter network stack is handling variability in the amount of energy accumulated in the energy reservoir. To understand the reasons, let us look at how micro-powered devices work. As shown in Figure 3.2, micro-powered devices operate in a sequence

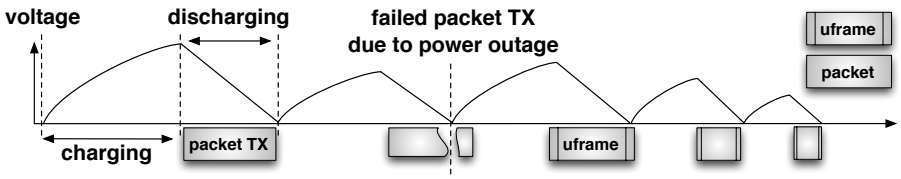
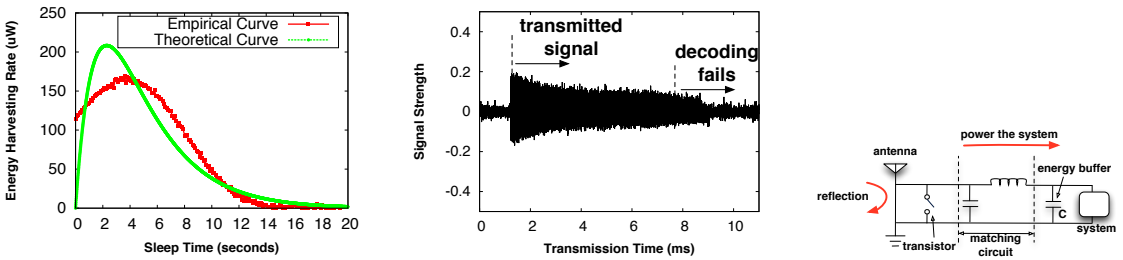


Figure 3.2. Energy harvesting systems.

of charge-discharge cycles since there is too little energy to continually operate the device. The device sleeps for a short period during which it harvests energy and charges a small energy reservoir, and then wakes up and transmits a packet during which the reservoir discharges.

There are several reasons why it is difficult to anticipate how much energy will be available in each discharge cycle. First, if harvesting conditions are too low, it is often too expensive to push more energy into a reservoir due to the inefficiencies of stepping up the voltage. As a result, the maximum amount of energy that can be accumulated depends on current harvesting conditions. Second, RF energy harvested by a node depends on how much energy is output by the reader. When a reader is doing nothing, the RF output power is roughly constant. However when a reader is communicating, this RF carrier wave is being modulated which changes the amount of harvested energy. In a multi-node network, the reader is communicating with different nodes, therefore harvesting rates continually vary at each node. Third, even if the node were to wait until it has a certain amount of energy prior to communication, this requires measurement of energy levels using analog-to-digital conversions (ADC). Each ADC operation consumes 327 uJ on the Moo platform [53], which is equal to the energy budget for transferring 27 bits of data. Such overhead is far too substantial on a micro-powered platform.

While choosing a smaller transmission unit might seem like a straightforward solution to this problem, this over-simplifies the design challenge. As the distance between the node and reader increases to the limit of the achievable range in Table 3.1,



(a) Energy harvesting rate. (b) Decreasing SNR over time. (c) RF matching circuit.

Figure 3.3. Factors that impact communication throughput.

the number of bits that can be successfully transmitted reduces. Thus, we need to use frames that may be as small as one or a few bits in size when the energy levels are low, which requires a network stack that can scale down to unprecedented levels. But such scale down often comes at the expense of throughput, which suffers due to the overheads associated with each transmission, including preambles, headers, and hardware transition overheads. To simultaneously optimize throughput, it is important to transmit as large a transmission as is possible given available energy. Thus, the problem faced by a node is that it needs to scale down its transmission unit to the bare minimum under poor harvesting conditions, while scaling up to improve throughput when the conditions allow.

Challenge 2: Variable harvesting rate:

The energy harvesting rate has significant impact on the communication throughput, since higher harvesting rate means that more energy can be used for data transfer. While energy harvesting rate might seem like a characteristic of the harvesting source, system parameters have a surprisingly high impact. Figure 3.3(a) shows the empirically measured harvesting rate as we vary the amount of time for which the node replenishes energy between two transmissions. The results are counter-intuitive — while one might expect more energy to be harvested over time, the harvesting rate drops to zero for longer sleep durations.

This observation can be explained analytically by looking at how capacitors buffer energy. The charging process of a capacitor follows its charging equation $V = V_{max}(1 - e^{-t_s/\tau})$, where t_s is the sleep time, τ is the RC circuit time constant, and V_{max} is the maximum voltage to which the capacitor can be charged under the current harvesting conditions. Its energy harvesting rate follows the equation: $H = C \times V_{max}^2 \times \tau^{-1}(1 - e^{-t_s/\tau})e^{-t_s/\tau}$. When the harvesting conditions are constant (i.e. V_{max} and τ are fixed), H is a concave function of t_s, which is shown both analytically and empirically in Figure 3.3(a). When harvesting conditions change, both V_{max} and τ change, therefore the maximum operating point changes as well. Thus, to optimize throughput, it is important to adapt to current harvesting conditions, and continually track the maximum harvesting point.

One factor that should not be overlooked is keeping the overhead of adaptation low. Most methods to track the charging rate of batteries and capacitors use analog-to-digital conversions to obtain the voltage at the energy reservoir. This overhead is minuscule for most platforms, but a significant part of the harvested energy in our case. Thus, it is important to minimize such overheads while adapting to harvesting conditions.

Challenge 3: Time-decaying SNR: A peculiar aspect of backscatter communication is that the signal to noise ratio (SNR) of the received signal at the reader degrades steadily as the size of the transmission unit increases. Figure 3.3(b) shows that the signal strength of a node response decreases gradually from 0.18 at 1.5ms to 0.05 at 8ms during the transmission process. While decoding the initial part of the transmission is straightforward due to high SNR, it becomes much more challenging after about 8ms since the SNR is too low for reliable decoding, resulting in packet losses.

In order to understand why this happens, let us look at how a backscatter radio works. A backscatter radio provides power to a passive device and enables communi-

cation. The reader provides a carrier wave, which can be reflected by a passive device back to the reader with its own information bits. The modulation is achieved by toggling the state of the transistor of a backscatter device shown in Figure 3.3(c). Since the same RF power source is shared by different system components, some fraction of the incoming power is used to operate the micro-powered device while the rest is reflected back to the reader for communication. The exact fraction depends on the state of the energy reservoir C and the state of the matching circuit, which is designed to charge the energy reservoir C when the voltage is low. Therefore, when the transmission begins, C is fully charged, the antenna resistance is mismatched with the resistance of other hardware components of the system. As a result, most of the incoming power will be reflected back to the reader, which receives a strong signal that can be easily decoded. As the transfer progresses, C slowly discharges, and the antenna resistance matches the resistance of the system load. Therefore, most of the incoming power is harvested to operate the system, and less RF power is reflected. This leads to decreased backscatter signal strength at the reader, and consequently, packet losses. Thus, to ensure that packets are received successfully, the tag needs to adapt the size of each packet such that the SNR at the tail of the packet is higher than the minimum decoding requirement.

Challenge 4: Energy-induced reader to node losses: While time-decaying SNR only presents a problem when a node communicates with a reader, reader to node communication presents other challenges. The central issue is that that the energy level on the receiving node might dip below the low watermark at any point during the reception, at which point the node has to shut off its RF circuit and go to sleep to recharge. The reader, however, does not know that the node has gone to sleep, and only realizes this fact after a timeout.

While such losses can be attributed to small energy harvesting variations at longer ranges, we observed to our surprise that such losses occur even when a tag is placed

relatively close to the reader — 40% losses at 2 ft. The reason for this behavior is that data transfer from the reader to tag comes at the expense of RF power being transmitted to the tag. Since the reader is actively transmitting to the tag, the carrier wave from the reader to tag is intermittent, causing substantial variations in RF energy harvesting and consequently variations in energy levels at the tag.

The energy dynamics at the tag makes it difficult to use reader-side estimation to identify the best transmission unit to communicate with a tag. In addition, explicitly providing information to the reader about the current energy level has considerable overhead while not being robust to dynamics. Thus, the challenge we face is that the reader needs to have a way of knowing the instantaneous energy state at the tag, and detecting its shut-off point without using cumbersome protocol-level mechanisms to enable this information exchange.

3.3 Fragmenting packets into μframes

At the heart of QuarkNet is a simple hypothesis — by breaking down packet transmission into its smallest atomic units, which we refer to as μframes, we can enable the system to scale down to severely limited harvesting regimes. We address the challenges in enabling such extreme fragmentation both for node-to-reader and reader-to-node communication.

3.3.1 Fragmentation at bit boundaries

The first question we ask is: what are the practical considerations that determine how we can dynamically fragment a logical transmission unit (packet) into μframes? Ideally, we would want to insert fragment boundaries at arbitrary positions within a packet so that we can make μframes as small or large as needed, however, this makes decoding extremely error-prone.

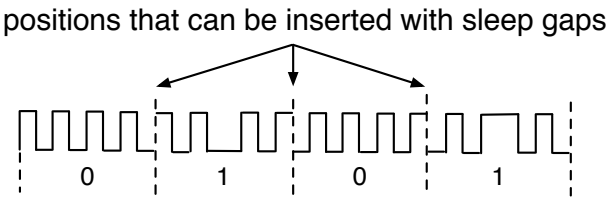


Figure 3.4. Sleep gaps can be inserted into backscatter pulses at various position (lines with dots).

To understand where to place fragmentation boundaries, we need to give some more detail about how backscatter modulation works. Figure 3.4 shows a sequence of backscatter pulses that compose bits in a packet. Backscatter modulation uses On-Off-Keying (OOK), therefore each bit is composed of a sequence of on and off pulses. As can be seen, the template for a '0' pulse and '1' pulse differ only slightly in the phase information of the pulses within the bit.

The key observation is that placing boundaries at certain points in a packet can be done without disrupting the phase information required for decoding, whereas other boundaries would disrupt decoding. For example, suppose that a fragment boundary is inserted between two adjacent bits, the phase information of each bit is maintained, thereby not impacting the ability to match the template to the bit. On the other hand, suppose that a fragment boundary is inserted within a single bit, the phase information within the bit is disrupted, thereby causing a mismatch at the decoder between received bit pulses and its template.

This leads us to a general principle for fragmenting a packet into μframes — μframe boundaries can be inserted between bits but not within a bit. The ability to fragment at any bit boundary gives us the requisite combination of fine-grained fragmentation as well as low decoding error.

3.3.2 Tuning inter-μframe gap

We now have a method for fine-grained fragmentation of larger packets, but how do we use this to dynamically fragment packets? How do we decide the length of each μframe and the sleep gap between μframes where the node replenishes energy?

We first answer this question for node-to-reader communication. In this case, we need to address two of the challenges discussed in §3.2: a) how to optimize throughput by operating at the optimal harvesting rate, and b) how to ensure the tail of each μframe transmitted from a node has sufficiently high SNR to be decoded at the reader.

Gradient descent algorithm: As can be seen in Figure 3.3(a), the harvesting rate curve is a concave function of the gap between μframes (under constant harvesting conditions). A fast and effective method for converging to the optimum of a concave function is to use gradient descent [7]. The gradient descent algorithm works as follows: first, we start with an initial guess about the optimal sleep gap. Second, we compute the gradient at this point, and look for the direction of the positive gradient. Third, we take a step along the direction of the positive gradient with step size proportional to the gradient. We repeat this process until convergence (i.e. step is smaller than a threshold). The algorithm takes large steps when the gradient is steep (i.e. point is far from optimal), and small as the gradient reduces (i.e. point is near optimal).

What if the harvesting conditions change and the curve itself shifts to create a new optimal harvesting point? Our gradient descent-based sleep gap adaptation algorithm operates continually — once it converges to the optimal, it periodically probes the gradient at the current optimal, and moves along the positive gradient if the optimal harvesting rate changes. In this manner, the algorithm seamlessly adapts to such dynamics.

Handling time-varying SNR: We need to add another constraint to to the gradient descent algorithm — the SNR at the tail of the frame should be higher than

the decoding threshold at the reader, otherwise the frame cannot be decoded. This constraint is easy to add since it simply translates to a bound on the maximum length of the inter-μframe gap. Since the length of the gap directly impacts the length of the μframe, capping the inter-μframe gap ensures that the length of each μframe is lower than the decoding threshold. The only change to the gradient descent algorithm is that a step cannot exceed the maximum inter-μframe as determined by the SNR constraint.

Duty-cycling the radio: One important aspect of the inter-μframe gap is that we shut off the node's RF circuit for this length of time. In a multi-node environment, the reader is constantly talking to other nodes, so leaving the RF circuit on results in substantial reception overhead since backscatter is a broadcast-based protocol, and wakes up every node that has its radio circuit turned on. To avoid these costs, we turn off the RF circuit during the recharge cycle. Once the node has slept for the intended duration, it switches on its RF circuit. One side-effect of our decision to turn off the RF circuit during gaps is that the reader now has to be more careful to avoid transmitting to a node or scheduling a node for transmission while it is inactive. We return to this question in §3.4.2.

3.3.3 Remote interrupts

We now turn to μframe adaptation for communication from a reader to a node. As described in §3.2, the key challenge is that the reader cannot detect when a node's energy level drops below a low watermark, and it should stop transmitting. Similarly, once a node has gone to sleep, a reader does not know when it will wake up for the next μframe. Given these constraints, how can we enable reader-to-node communication?

Estimating μframe length: Our idea is to use a remote interruption mechanism, where a node issues an in-band interrupt during reader transmission, and informs the reader that it has reached a low-energy state. This remote interrupt is generated by

toggling its transistor while receiving the current frame. In other words, the remote interrupt is a signal that is overlaid on the same time-slot and frequency signal as the message from the reader to node.

How can the reader decode an in-band interrupt from the node? The key insight is that the reader modulates the carrier by toggling the carrier wave whereas the node communicates back to the reader by changing the amplitude of the backscattered signal. In other words, both can occur simultaneously! Thus, when the reader is sending an ON pulse, the amplitude of the backscattered signal that it receives depends on whether the state of the transistor at the node is ON or OFF — the amplitude is higher when the node's transistor is ON and lower when it is OFF. When the carrier is OFF at the reader, then the state of the node's transistor does not matter since there is no backscattered signal. The reader can detect the remote interrupt by looking for a large signal variance in the carrier wave when the reader has the carrier wave turned on.

Figure 3.5 shows an example signal where toggling the transistor causes a large variance on the carrier wave, which is monitored by a reader and can be identified by tracking the signal variance within a reader pulse. However, the signal variance is detected only when the carrier wave is on. As shown in the figure, a reader cannot observe the large signal variance when the carrier wave is off. Fortunately, the carrier wave is on for 50% of the time when the reader transmits 0s and 75% for 1s. Thus, as long as a remote interrupt is longer than 50% of the length of a '0' bit from a reader, it can reliably detect the interrupt and pause its transfer.

Finally, an auxiliary benefit of the remote interrupt is that it acts as an inexpensive μframe ACK from the node, which obviates the need for more explicit protocol-level mechanisms and reduces our overhead.

One limitation of our current design is that it is not robust to noise spikes in the frequency band. Such spikes can occur because of multiple readers transmitting to

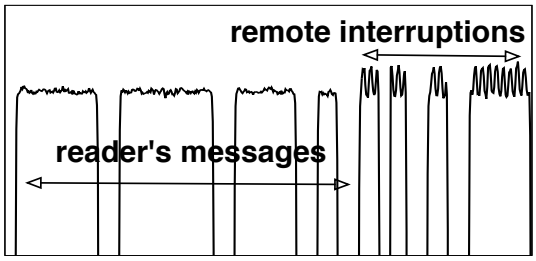


Figure 3.5. In-band remote interruptions from nodes.

nodes since backscatter is a broadcast medium and reader-to-node communication has to be serialized. Robustness against external interference could be improved by making the remote interrupt longer and encoding the signal, but we do not do this in our current implementation.

Estimating inter-μframe gap: We now have a way for the node to interrupt a reader when it needs to replenish energy, but how long should the reader wait before initiating the next μframe transfer? Clearly, this duration should be at least as long as the inter-μframe gap that the node is using, otherwise the reader might be trying to communicate to a node that has its RF circuit turned off. We address this by using a simple probing-based approach at the reader — for each μframe gap that the reader selects, it knows whether the frame was received or not by checking the presence of a remote interrupt. If no remote interrupt is received, the reader knows the node does not receive the frame properly. The reader continually adjusts the gap to minimize missed frames at the node.

3.4 QuarkNet for multi-node networks

So far, we have focused on communication between a single node and reader. We now turn to the case where there are several nodes in the vicinity of a reader. The key difference between a single node and multi-node setting is that in the former, the reader stays idle during times when the node is asleep to replenish energy, whereas

in the latter, these inter-μframe intervals present an opportunity to schedule another node's μframe transfer, thereby ensuring that throughput is maximized.

3.4.1 Design Options

Before launching into the details of our design, lets step back and look at the design options. Co-ordination mechanisms for backscatter networks are more restrictive than typical active radio-based networks for two reasons: a) nodes cannot overhear each other's transfer, hence carrier sense-based approaches are infeasible, and b) the stringent resource constraints of nodes render approaches that require complex coding and synchronization infeasible. As a result, existing proposals have focused on two classes of techniques — EPC Gen 2 and variants which use a sequence of random-access slots, and rateless transfer where nodes transfer concurrently, and the reader simultaneously and successively decodes all transmissions.

While the deficiencies of EPC Gen 2 for severely energy constrained regimes have been detailed earlier in this chapter, other alternatives and enhancements are surprisingly poor in dealing with this regime as well. In particular, consider two prominent recent techniques — Flit [27] and Buzz [48]. Our earlier work, Flit, re-purposes EPC Gen 2 slots for bulk transfer, thereby amortizing overhead, but it assumes that nodes are able to sustain a long stream of transfer, which we realized was not the case in severe harvesting conditions. Buzz uses rateless codes, but in-order to get these codes to work, it has to use synchronous single-bit slots across nodes. Each single-bit slot incurs substantial overhead due to slot indicators, and turning on and off the radio, which dramatically impacts performance. Given that existing approaches are not well-suited to our nodes, the question is what protocol to use for co-ordinating nodes.

3.4.2 Variability-aware node scheduling

Our scheduler is designed to interleave μframes from different nodes, thereby fully utilizing the inter-μframe gaps. The reader divides time into variable-sized μslots,

during which it explicitly schedules a single node to transmit its μframe. The length of each μslot depends on the size of the μframe — a node-to-reader μframe terminates when the node reaches its low watermark energy level and the reader ACK is received, and a reader-to-node μframe terminates when the node issues a remote interrupt. In both cases, there is a maximum bound on the μframe size to deal with nodes that have plentiful energy.

While the μslot mechanism appears relatively straightforward, the main challenge is handling the fact that nodes turn off their RF circuit when they are asleep. As a result, if a node is scheduled too early by the reader, then it may not be awake to utilize the slot, but if it is scheduled too late, then it is not operating at its maximal harvesting rate.

To handle this, we use a token-based scheduler to deal with the stochastic nature of harvesting conditions, while optimizing throughput. For each node, the scheduler maintains a running estimate of the gap between μslots assigned to a specific node, and whether the μslot resulted in a successful transfer. It uses the estimate to select the inter-μframe gap that ensures a high likelihood of obtaining a node response.

The reader's estimate of the inter-μframe gap is used as input to a token bucket scheduler, which assigns tokens to nodes at a rate inversely proportional to its inter-μframe gap. Once a node has accumulated sufficient tokens, it is likely to have woken up after sleep, therefore the reader places the node into a ready queue since it is ready to be scheduled. The ready nodes can be scheduled based on a suitable metric — for example, the highest throughput node may be selected from the queue to maximize throughput, or the node that has received least slots may be selected for fairness.

3.5 Implementation

In this section, we describe key implementation details not covered in earlier sections. We use the USRP reader and UMass Moo for our instantiation of QuarkNet. The source code of QuarkNet is available at [10].

3.5.1 Platforms

USRP Reader: QuarkNet is built based on the USRP software radio reader developed by Buettner [21] with a ANT-NA-2CO antenna [12]. We modify the signal processing pipeline to enable variable sized μframe decoding, harvesting-aware tag scheduling, and detection of in-band remote interrupts. The RFX900 USRP RF daughterboard on our platform is only able to transmit 200mW of power, which is 5$\times$ smaller than the 1W of power issued by a commercial reader. Therefore, we attach a 3cm$\times$3cm solar panel to each Moo to increase the amount of harvested energy. The use of hybrid power (RF + ambient) is known to increase range from a reader, which enhances the regimes where backscatter can be used [26].

Backscatter node: The UMass Moo is a passive computational RFID that operates in the 902MHz $\sim$ 928MHz band. Perhaps the most challenging aspect of our implementation is debugging under extreme low energy conditions. Traditional methods for debugging embedded systems, such as using JTAG, supply power to the node and change its behavior. Instead, we instrument the Moo to toggle GPIO pins at key points during its execution, and a logic analyzer to record the toggle events. In many cases, however, it is difficult to insert sufficient instrumentation to have visibility while still working with tiny energy harvesting levels. Thus, intuition and experience is particularly important in designing systems for these regimes.

3.5.2 Trimming Overheads

One important aspect of our system is careful measurement and tuning of all overheads, which impacts our ability to scale-down to severe harvesting conditions.

Radio transition overhead:: An important source of overhead is transition times for turning on or off the radio. Fortunately, since hardware timers are responsible for generating the pulses on the backscatter radio, sleep gaps can be inserted by clearing the hardware timers and turning the micro-controller into its low power mode. These operations are inexpensive energy-wise, and consume roughly the same amount of energy as a data frame of size 3 bits. Note that this observation does not hold for more complex active radios — for example, a WiFi radio takes 79.1ms to be on, and 238.1ms to be turned off [33], which is five orders of magnitude higher than the corresponding numbers for a backscatter radio.

Pilot tone:: Each backscatter frame can potentially include a pilot tone in addition to the payload. A pilot tone is used when a tag changes its baud rate [44]. We focus on a minimalist protocol that uses a fixed baud rate, therefore we remove the pilot tone. The total overhead per μframe is 6 bits of preamble, in contrast to the 22 bits overhead of EPC Gen 2 (and variants such as Flit [27]).

Probing energy state:: As mentioned earlier, analog-to-digital conversions are expensive, and should be avoided while tracking the maximum energy harvesting rate. Our key insight is that rather than measure the voltage on the node, we can leverage the existing low watermark threshold detector that is already present on such nodes. Such a detector is common on harvesting-based sensor platforms for two reasons: a) the platform needs to know when to save state and go to sleep to avoid an outage, and b) the platform needs to know when to wake up after sleep to continue operation. Thus, QuarkNet gets an interrupt both when the voltage crosses above the threshold, as well as when it drops below the threshold, and uses this information as a one-bit proxy for the actual voltage. The voltage threshold is chosen to be 2V which is slightly higher than 1.8V, the minimum voltage required for operating a micro controller. This information is input to a sleep time tracker, which determines

how long to wait after crossing the threshold in the upward direction before initiating transfer. Our approach is 100× less expensive energy-wise than an ADC conversion.

3.5.3 Protocols and Algorithms

While we do not describe the complete protocol in the interest of space, more details as well as pseudocode for our algorithms can be found in our technical report [55].

3.6 Evaluation

The evaluation consists of three parts: 1) demonstrating the range and throughput benefit of μframe transmission, 2) benchmarking the performance of our reader-to-node communication, and 3) evaluating the benefit of interleaving μframes from multiple nodes.

3.6.1 Benefit of μframes

In this section, we validate our claim that the ability to breakdown packets into μframes that can be as small as a single bit can allow us to operate under lower energy conditions and achieve higher operating range. To focus on the effect of the choice of frame size, we strip off overheads (slot indicators, handshakes, etc) for all protocols that we compare.

Minimum operating conditions: We look at two harvesters — RF and solar — and ask what is the minimum power requirements for different approaches. We find that the minimum illuminance required for a 1 bit μframe is 150 lux, which is 13× lower than the 2000 lux budget of 12 byte packet transmission (the same packet size used by EPC Gen 2, Dewdrop, Flit, etc). We choose 12 byte packet size for EPC Gen 2-based protocols because the 12 byte EPC identifier needs to be transmitted in a singulation phase prior to executing Read or Write commands. Thus, this packet is the bottleneck for operation. To translate from lux to the typical energy available

from indoor energy sources, we measure the natural indoor illuminance in 30 positions in an office room. We find that 92% the measured illuminance value is between 150 lux and 1000 lux. This suggests that μframes can operate in most of natural indoor illuminance conditions while a canonical 12 byte transfer scheme can almost never operate under natural indoor light.

The minimum RF power required for a 1 bit μframe is 13dBm, which is 20$\times$ smaller than the 26dBm budget of a 12 byte packet transmission that is the minimum needed for EPC Gen 2 and its variants to operate. Both experiments illustrate the benefits of using tiny μframes.

Increased operational range: Our second claim is that we can improve operational range by using μframes. Figure 3.6 shows the maximum range that is achieved by QuarkNet with 1 bit μframes, EPC Gen 2 with fixed 12 byte packets, Dewdrop with fixed 12 byte packets, Buzz with two slot choices, and a battery-assisted node which represents the best-case scenario. We adjust the RF power of the USRP RFID reader from 17dBm to 25.7dBm, which represents the range of RF power that can be generated by the USRP RFX900 daughterboard.

The results show that the communication range of QuarkNet is longer than other schemes across all RF power levels. At the lowest power level (17.5dBm), μframes do not improve range since the node is not able to decode the reader signal beyond 5ft. But as the RF level increases, the operational range increases dramatically, and is about 4$\times$ longer than EPC Gen 2 at the highest power. In fact, the performance of 1 bit μframe transfer while using harvested energy almost matches the performance of a battery-assisted node, which shows that we are able to reach the ceiling of operational range despite operating on micro-power.

Figure 3.6 also shows that Buzz [48] performs poorly compared to other schemes. This can be attributed to the fact that each one-bit slot in Buzz has substantial overhead — the reader sends a pulse, followed by one bit from the node, random

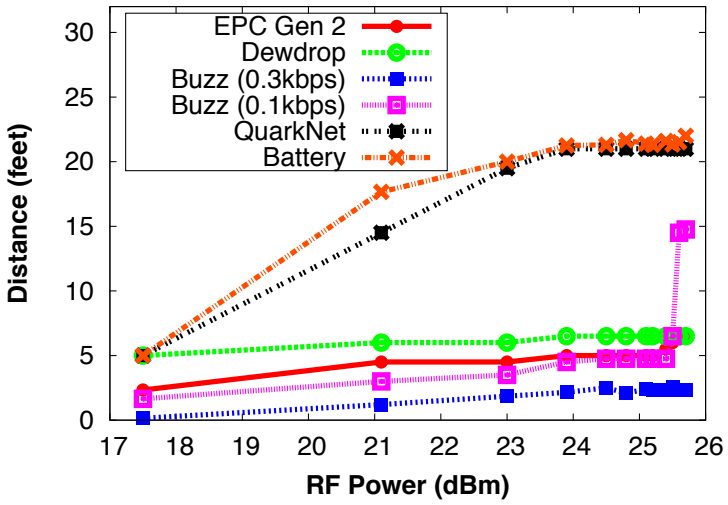


Figure 3.6. The maximum range achieved by EPC Gen 2, Dewdrop, Buzz, QuarkNet, and a battery assisted node. QuarkNet operates at ranges close to the battery assisted node.

number generation for deciding whether to transfer in the next slot, and a recharge period. Thus, while Buzz has high range in some settings, the overhead is too high to scale gracefully.

3.6.2 Benefits of μframe adaptation

We now turn to the benefits of adapting the inter-μframe gap to maximize throughput.

Convergence of gradient descent: How well does the gradient-descent algorithm learn the optimal harvesting rate? Figure 3.7 shows the results for a node placed in three RF+light harvesting combinations that include short and medium range, and low and medium light. In all cases, we see convergence to close to the optimal point — the best inter-μframe gap ranges from 1ms for 350lux at 1 foot, 4ms when the node is moved to 6ft, to 12ms when the light conditions dip further. In all cases, our tracking algorithm converges in very few steps (≤ 4).

Throughput benefits: We now know that QuarkNet picks close to the optimal harvesting rate, but what are the benefits in terms of throughput? To understand this,

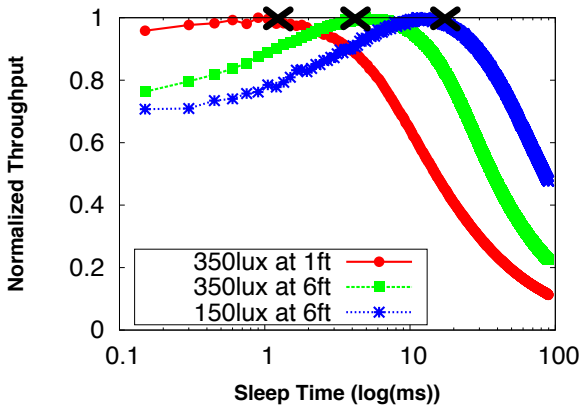
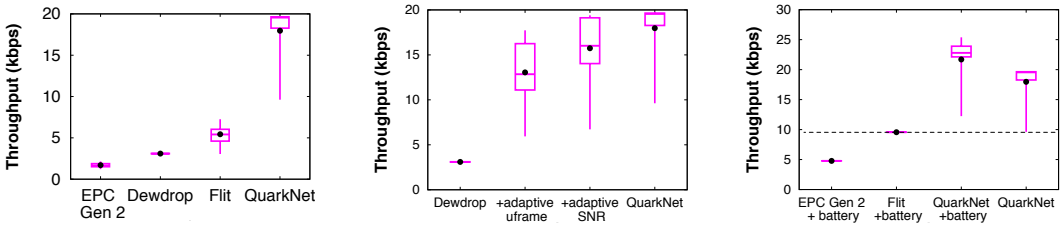


Figure 3.7. Throughput achieved for different sleep times (inter-μframe gaps). The sleep time chosen by QuarkNet is within 98% of the optimal.

we place a node 3 feet from a reader, vary RF power from 17dBm to 26dBm in small steps of 0.3dBm, and inventory the node 2000 times for each scheme. Figure 3.8(a) shows the throughput achieved by EPC Gen 2, Dewdrop, Flit, and QuarkNet. We find that the throughput achieved by QuarkNet is higher than EPC Gen 2, Dewdrop and Flit across all RF power levels. The average communication throughput of QuarkNet is 18kbps, 10.5× higher than EPC Gen 2, 5.8× higher than Dewdrop, and 3.3× higher than Flit. While the figure does not show Buzz's throughput, note that Figure 3.6 already showed that this number is low since the per-slot overhead dominates. The lowest slot size we achieved in our implementation of Buzz is 3ms, which means about 0.3kbps throughput.

The previous experiments were done by varying the RF power level. To be sure that these results translate to the case where nodes are placed at different locations in front of a reader, we measure the throughput achieved by EPC Gen 2, Dewdrop, Flit, and QuarkNet at 30 different randomly chosen locations between 2 to 13 ft in front of a reader. Figure 3.9 shows that the throughput achieved by QuarkNet is higher than the other three schemes across all locations. The average throughput of QuarkNet is 7.8× higher than EPC Gen 2, 6.4× higher than Dewdrop, and 4.4× higher than Flit.



(a) QuarkNet vs micro-powered nodes.

(b) QuarkNet vs Dewdrop.

(c) QuarkNet vs battery assisted nodes.

Figure 3.8. For micro-powered devices, QuarkNet improves throughput by at least 3.3× over all other schemes, and even performs better than battery assisted nodes. The benefit comes from reducing overhead, and adapting μframe sizes to energy and SNR.

In particular, QuarkNet continues to operate in many locations where other schemes cease to operate.

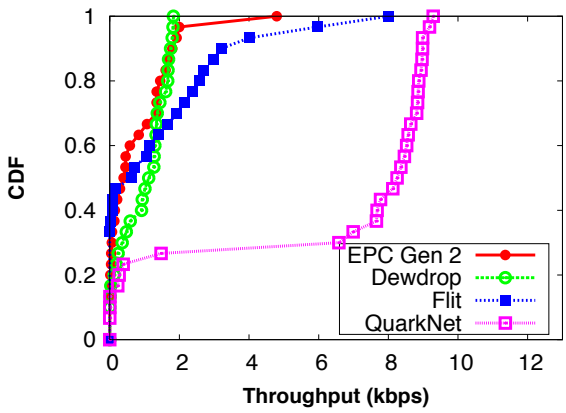


Figure 3.9. Throughput achieved by EPC Gen 2, Dewdrop, Flit, and QuarkNet across 30 locations. QuarkNet has at least 4.4× higher throughput than other schemes.

Breaking down the benefits: QuarkNet has a variety of optimizations including reduced overheads, variable-sized μframes, and SNR adaptation. To understand the contributions of these techniques to throughput, we start with the default implementation of Dewdrop, and add one optimization at a time: a) Dewdrop + adaptive frame, which includes variable-length μframes, and b) Dewdrop + SNR adaptation which includes the SNR adaptation. Figure 3.8(b) shows the throughput achieved by the three variants of Dewdrop vs QuarkNet. Clearly, each of the optimizations

plays a major role in the throughput improvements observed by QuarkNet. The average communication throughput of μframe is 18kbps, 5.79× higher than Dewdrop, 1.37× higher than Dewdrop with adaptive μframes, and 1.14× higher than the case when SNR adaptation is included. In the final step, we replace Dewdrop's adaptation algorithm with our version that eliminates ADC conversions to get QuarkNet.

QuarkNet vs battery-assisted alternatives: Another interesting question is how QuarkNet performs when compared to battery-assisted versions of the other protocols (excluding Dewdrop + battery, which is identical to EPC Gen 2 + battery). Some protocols, such as Flit [27], improve in performance when there is more energy since there is more opportunity for bulk transfer. Would these outperform QuarkNet in battery-assisted scenarios? Figure 3.8(c) shows that throughput achieved by QuarkNet is consistently better. The average throughput of QuarkNet is 18kbps, 3.75× higher than EPC Gen 2 + battery, and 1.87× higher than Flit + battery. This result shows the benefit of reducing per-frame overheads in QuarkNet.

3.6.3 Reader-to-node communication

We now turn to an evaluation of reader-to-node communication. We begin by looking at the effectiveness of remote interrupts. We find that remote interrupts are extremely reliable — the reader detects remote interrupts with 100% accuracy across all distances where the node can communicate with the reader, and detection rate directly drops to 0% at roughly 19 – 20 feet where the node cannot detect the signal sent by the reader. While the accuracy will degrade under external interference, we plan to extend remote interruption to include encoded bits to improve robustness.

Next, we look at the throughput of reader-to-node communication when a node is placed at different distances from the reader. Figure 3.10 shows that the throughput achieved by QuarkNet is always higher than fixed 100 bit transfer across all distances. (We chose 100 bits instead of 12 bytes because of the slower baud rate of the reader-

to-node link, as a result of which 12 byte transfer ceases to operate even when the node is deployed 1 feet from the reader.) The throughput of QuarkNet is higher than even a battery-assisted EPC Gen 2 node. This shows that the benefit of variable sized μframes is substantial even for reader-to-node communication.

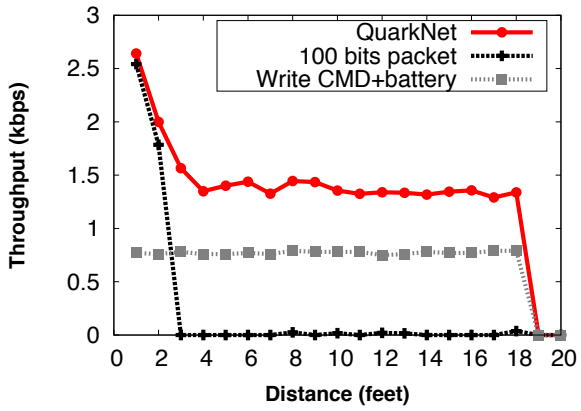


Figure 3.10. Throughput of reader-to-node communication. QuarkNet has 2× higher throughput than battery-assisted EPC Gen 2 Writes.

One trend in the graph that requires a bit more explanation is the fact that throughput decreases rapidly when the node is close to the reader (less than 4 feet), and plateaus until about 18 ft after which it quickly drops to zero. This is because RF-harvesting only works until 4ft (because of the limitations of the USRP reader), and beyond this distance, indoor light harvesting plays the dominant role.

3.6.4 Evaluating the QuarkNet MAC layer

We now turn to the evaluation of our MAC layer that includes all components of the protocol including various co-ordination overheads, frame interleaving, and scheduling. Figure 3.11 shows the communication throughput when we deploy 10 nodes in front of the reader and adjust the RF power from 17dBm to 26dBm. We use a throughput-maximizing scheduling policy in this experiment. For each RF power level, we plot the averaged throughput across the ten nodes and the confidence interval when they are scheduled in an interleaved manner and when they are inventoried

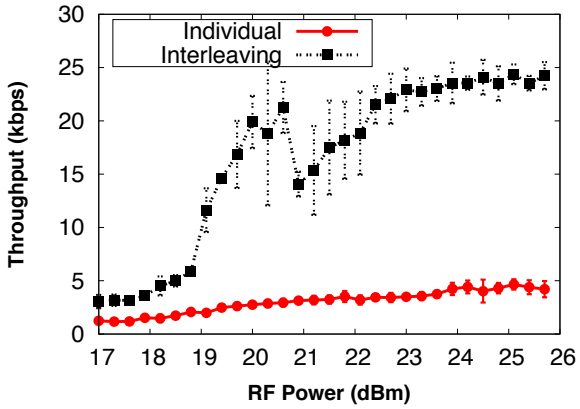


Figure 3.11. Throughput of 10 nodes is 5.4× higher when interleaved than when individually inventoried.

individually. The throughput achieved by other MAC layer designs — EPC Gen 2 and Flit — are close to zero, so we do not plot them.

We find that even at the lowest RF power level, almost all nodes get to transmit data to the reader, and the average throughput steadily increases with higher RF power. In addition, the throughput achieved by interleaving the 10 nodes is 5.4× higher than the throughput when those 10 nodes are inventoried individually. These results show that our algorithm scales well across a wide dynamic range of harvesting conditions, and uses gaps between μframes efficiently.

3.6.5 Microbenchmarks

Table 3.2 shows the overhead incurred by different components of QuarkNet. The biggest system overhead is the switch from inactive mode to transmission mode (47.5 us), to configure several registers associated with transmission, such as the hardware timer register and data register. The overhead of the entire μframe size and inter-μframe gap adaptation algorithm (47.2us), is comparable to the total system overhead, and 10× smaller than the cost of an ADC conversion. Overall, the results show that our performance tuning measures have substantial benefits — the sum total of these overheads is smaller than the cost of transmitting 7 bits.

Table 3.2. Overhead of μframe transmission.

System overhead (us)		μframe overhead (us)	
TX to inactive	9.9	interrupt config	10.58
inactive to TX	47.5	handle interrupt	9.3
RX to TX	4.08	μframe adaptation	24.3
sleep to wakeup	9.83	voltage detection	3

3.7 Discussion

Interoperability with other PHY mechanisms: While our work does not explicitly address co-existence of QuarkNet with other physical layer and upper layer mechanisms, many of these can be easily layered above the methods described in this chapter. For example, rate adaptation is widely used to adapt to wireless channel conditions, thereby maximizing communication throughput. This method operates at the bit-level, where each bit is composed of several symbols. Such an approach can be layered above QuarkNet, with gaps introduced between bits. Similarly, error correction codes or other encoding mechanisms that reduce bit error rate can be implemented above QuarkNet.

QuarkNets role with evolving technology: As micro-harvesters continue to improve in efficiency, one question is whether QuarkNet will continue to remain relevant. We argue that QuarkNet's relevance will increase for two reasons. First, the maximum harvesting rates are fundamentally limited by the physics of the harvesting source and form-factor. For example, RF energy harvesting is limited by the antenna size and the amount power issued by antennas, solar energy harvesting is limited by the panel size and the intensity of illuminance, and thermal energy harvesting is limited by the surface area and the temperature differential. Even if micro-harvesters become extremely efficient (say upwards of 80%), there is still a small amount of energy available, and systems optimizations similar to QuarkNet are critical to using the energy in an efficient manner. Second, trends in nano-electronics and low-power embedded systems are resulting in sensing and computing platforms that consume only tens of micro-watts of power [2]. These trends will make it possible to design

many more micro-power based applications such as implantables and on-body sensors, enhancing the relevance of QuarkNet.

Fragmenting other tasks: While our focus in this chapter is on fragmenting the network stack, the abstraction of task fragmentation presented by QuarkNet can be potentially used for breaking down other components of a task such as sensing and computation into smaller atomic units. In our position paper [56], we presented preliminary results that demonstrated the ability to fragment an image sensing task such that the entire sensor can operate with a 3cm×3cm solar panel under natural indoor illuminance. However, many questions remain to fully enable such fragmentation, requiring a combination of architectural modifications to the sensing and computing blocks to facilitate fine-grained fragmentation, systems techniques similar to QuarkNet that can take advantage of the fragmentation capability, as well as data processing techniques to enable useful applications over a layer that dynamically fragments sensing tasks.

3.8 Related Work

We have already discussed Dewdrop, Flit, Buzz, and EPC Gen 2, so we focus on other approaches.

Computational RFIDs (CRFIDs): There has been increasing emphasis on CRFIDs in recent years given its potential for battery-less perpetual sensing. Ambient Backscatter [38] uses the backscatter of FM signals for short-range communication between tags. This is a severely energy limited platform, and could leverage QuarkNet when harvested energy is low. BLINK [57] is a bit-rate and channel adaptation protocol to maximize communication throughput, which can also leverage QuarkNet for performance. [45] introduces a power-optimized waveform which is a new type of multiple-tone carrier and modulation scheme that is designed to improve the read range and power efficiency of charge pump-based passive RFIDs. [46] presents a sys-

tem architecture for backscatter communication which reaches 100m communication distance at the cost of slow bit rate (10 bits per second). Such techniques are complementary to QuarkNet — each bit transmitted at slow bit rate can be fragmented into several segments where the information within each bit is still preserved. Also of note is MementOS [43], which uses non-volatile flash storage for checkpoints within a task such that the it can continue execution after an outage. Flash checkpointing is useful for outage tolerance but is more than the cost of transmitting an entire EPC Gen 2 packet, hence it has limited utility in our case.

EPC Gen 2 optimizations: Much of the work on backscatter communication is specific to EPC Gen 2 tags, for example, better tag density estimation [47], better search protocols to reduce inventorying time [36], better tag collision avoidance [39], more accurate tag identification [52], better recovery from tag collisions [15], and more efficient bit-rate adaptation [57]. None of these tackle the problem of maximizing range and throughput from RFID-scale sensors, which have the ability to offload sensing data back to a reader.

EPC Gen 2 supports tag user memory operations in addition to simple EPC queries including the Read and Write command, however they are second-class citizens in the protocol since the main goal is to inventory tags. As a result, both are inefficient primitives for data transfer from tag to reader or vice-versa. In our experiments, we found that the Read and Write commands simply do not work at all under low energy conditions.

3.9 Conclusion

In this chapter, we present a powerful network stack, QuarkNet, that can enable systems to seamlessly scale down to severe harvesting conditions as well as substantial harvesting dynamics. At the core, our approach deconstructs every packet into μframes, handles dynamics with variable-sized μframes, and maximizes throughput

via low-cost adaptation algorithms and interleaving of μframes. Results show that QuarkNet provides substantial benefits in pushing the limits of micro-powered devices, and allow them to perform useful work under more extreme environments than previously imagined possible. Our network stack tolerates such conditions, thus makes it valuable to a wide range of emerging micro-powered embedded systems and applications.

CHAPTER 4

HIGH SPEED ULTRA LOW-POWER BACKSCATTER

Existing sensing architectures incur substantial overhead for a variety of computational blocks between the sensor and RF front end — while these overheads were negligible on platforms where communication was expensive, they become the bottleneck on backscatter-based systems and increase power consumption while limiting throughput. In this chapter, we propose a radically new design that is minimalist, yet efficient, and designed to operate end-to-end at tens of μWs while enabling high-data rate backscatter at rates upwards of many hundreds of Kbps.

4.1 Background and Motivation

A fundamental assumption that has driven the design of sensor networks for decades is that communication is the most power-hungry component of an individual sensor system. The power consumption gap between communication and other modules has driven a plethora of design choices in sensor networks, primarily by encouraging designers to reduce data at the source, thereby minimizing the amount of data that needs to be communicated.

We argue that this assumption does not hold when it comes to passive radios such as backscatter. Backscatter requires extraordinarily simple circuitry since the carrier wave is generated by a reader, and a sensor only needs to modulate the signal to transmit information, thereby eschewing power-hungry components of a typical active radio. The simplicity and inherent efficiency of backscatter means that the

Table 4.1. Power consumption of accelerometer, audio, ecg, and image sensors.

	Accel [3]	Audio [4]	ECG [5]	Camera [28]
Power	6μW	15.3μW	60μW	0.7μW

energy gap between communication and other components of a system has narrowed dramatically.

These observations have profound implications on the design of next-generation wireless sensing systems that operate using backscatter. The primary implication is that the bottleneck in terms of power consumption has shifted away from communication to computation and sensing. But sensing is often not the bottleneck as well — the past decade has seen dramatic reductions in the power consumption of sensors such as microphones, cameras, ECG, accelerometers, and others, many of which consume only μWs of power while sampling at high rates (Table 4.1). Thus, both backscatter communication and a variety of low-power sensors can operate at μWs of power, and the key question becomes one of optimizing the rest of the system to match these numbers. This requires that we re-think every component between the sensor and RF interface — data acquisition, data processing, buffering, packetizing, MAC, and many others now become the bottleneck for achieving ultra-low power operation.

In this chapter, we overturn the design principle governing wireless sensor design from one that is focused on minimizing communication to one focused on optimizing the computational elements between the sensor and RF interface. But optimizing computation is easier said than done, and requires an understanding of every module of the sensing platform, in-depth analysis of how to eliminate overhead from these modules, and design of a modified architecture to support an optimized design.

But our efforts to optimize computation raises an unexpected problem. If we do nothing to reduce data at the source, we need the bandwidth to be able to transfer raw data from the sensor to infrastructure. While backscatter communication is efficient in terms of power, throughputs achieved by practical backscatter-based systems have

been abysmal. Despite several efforts at improving throughputs of backscatter [27, 57, 20, 48, 26], the best case throughput is still only around 20 kbps even when only a single node is present, and drops dramatically to barely hundreds of bits/second when there are multiple devices sharing the network. These numbers are not encouraging — for example, a microphone sampled at 8-44 KHz requires transmit rates upwards of 704 kbps, a far cry from the throughput that backscatter platforms are able to support today.

This leads us to the central question that we address in this chapter: *how can we design a backscatter-based wireless sensor system that achieves whole-system power consumption of μWs, while simultaneously increasing data rates to support raw data transfer from sensors at several hundreds of kilobits/second.* Our goal is aggressive — as a point of comparison, an existing backscatter-based sensor, the UMass Moo (or the UW WISP) consumes about 2mW of power while transmitting at a few kilobits/second when there are multiple devices present. Thus, we seek to drop the system-wide power consumption by more than two orders of magnitude while simultaneously enabling two orders of magnitude increase in the data rates.

Our contributions are two-fold. First, we present a novel backscatter-based sensor platform, Ekho, that achieves our design goal to optimize power by eliminating computational overhead from the sensor to RF pipeline. We start with a deep dive at what computational modules are present between the sensor and RF interface on a typical low-power sensor platform, and measure their power consumption, before launching into a minimalist design that is optimized for power. Our second contribution is a network stack, EkhoNet, that is designed to be minimalist and enable bandwidth scale up to support data rates of hundreds of Kbps while supporting tens of nodes. While each Ekho node is minimalist, our MAC layer leverages resources at the reader to enable utility-energy and channel-aware optimization of bit rates and slot sizes across nodes.

Our results on a USRP reader and Ekho nodes show that:

- For operating an accelerometer at 400Hz, Ekho consumes 35μW of power, 7.6× lower than the 266μW of the Moo and 3.3× lower than the 118μW of WISP5.0. For operating an audio sensor at 44kHz, Ekho consumes 37μW of power, 76× lower than the Moo and 13.5× lower than the WISP5.0.

- We show that EkhoNet can scale to a network of several high bandwidth sensors. When a network of ten Ekho nodes equipped with microphones transmit simultaneously to a reader, we achieve a throughput of 780 kbps as a result of interleaving the data streams at the MAC layer. We also use an energy-utility-channel aware scheduler, and show that over 50% of the audio sensors achieve a median MOS score larger than 2, significantly higher than a baseline scheme that assigns sampling rates evenly across all nodes.

4.2 Case for Ekho

In this section, we make the case that backscatter communication is extremely cheap and overturns the widely held premise that communication is more expensive than computation. We focus on the tradeoff between computation and communication since many commonly used sensors are already extremely efficient in terms of power. We begin with a discussion of why backscatter is efficient.

4.2.1 Backscatter radio RF front end

Backscatter radios are designed to enable ultra low power wireless communication. As shown in Figure 4.1, a reader provides a carrier wave, which can be modulated with information to enable ultra low power wireless communication. While the carrier wave can also be rectified by a sensor for energy harvesting, our focus in this chapter is on backscatter as a low-power radio, whether energy is obtained via harvesting or

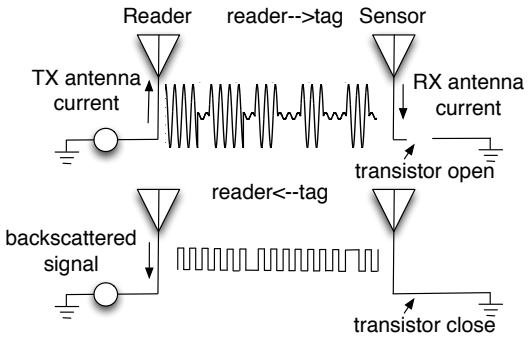
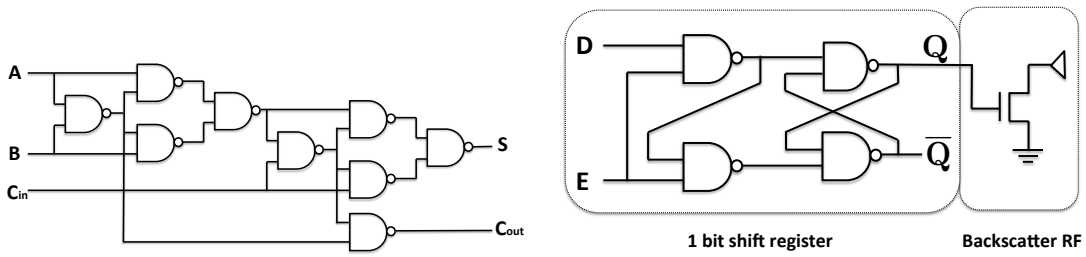


Figure 4.1. Backscatter communication basics.

a battery, hence we focus on the communication rather than harvesting aspects of backscatter.

To transmit data, a sensor toggles the state of a transistor to detune its antenna and reflect the carrier wave back to the reader with its own information bits. Because the sensor does not actively generate RF signal as active radio systems, the power consumption of the backscatter radio is very low. In addition, the on-off transition overhead of backscatter radios is very short because backscatter radios do not have to warm up the RF analog circuits for data transmission unlike active radio systems. As a result, there is little overhead incurred while transmitting via backscatter, even when transmitting at a high rate. For example, one key component of the backscatter analog RF front end of the WISP [13] is a MOSFET transistor (BF1212WR). Its power consumption follows the equation of CV^2F where C is the capacitance of the transistor, V is the digital drain-source voltage, and F is the frequency of operating the transistor. When this transistor is toggled at a slow rate of 10Hz, it consumes 55pW of power, and even when toggled at a high rate of 1MHz, it only consumes 5.5μW of power. Thus, backscatter radios consume of the order of μWs of power, even for high rate data transfer.



(a) 1 bit adder circuits for computation.

(b) 1 bit shift register controls a backscatter transistor.

Figure 4.2. 1 bit adder and 1 bit shift register circuit.

4.2.2 Why compute if its cheaper to transmit?

The power consumption of backscatter radio has surprising implications on sensor system design, and challenges long-held views about communication vs computation tradeoffs in these systems.

Computation vs Communication: A common assumption in designing sensor systems has been that computation is significantly cheaper than communication, often by many orders of magnitude. This view has shaped a plethora of efforts for in-network processing, signal compression, sub-sampling, and other such approaches to reduce data at the source prior to communication. Indeed, this tradeoff has been reinforced by performance/power trends over the past decade — power consumption of embedded processors have dropped dramatically, while power reduction in active radios has been relatively slower.

However, backscatter communication challenges this long-held view. Backscatter is inherently extraordinarily efficient since the carrier wave is generated by the reader, and the tag only backscatters the signal without any additional amplification. Thus, each bit of backscatter is extremely simple, and only requires a handful of gates (Figure 4.2). This implies that for computation to be cheaper energy-wise, the computational operations on each bit would have to use fewer gates than that required to communicate the bit. This is often a tall order due to the simplicity of backscatter.

Consider, for example, a simple aggregation operation that sums ten sensor readings before transmitting the aggregate value over the radio. On traditional sensor platforms, such data reduction would have direct and significant power benefits since communication dominates power, and our aggregation scheme cuts this cost by a factor of ten. The same operation on a backscatter-based platform has dubious benefits. Figure 4.2 shows that the number of NAND gates required for summing two bits is roughly nine (thirty six transistors), but only four NAND gates (sixteen transistors) and an additional transistor for backscattering the signal are needed to transmit the same data via the shift-register controlled backscatter RF! As power consumption is proportional to number of transistors, a nine gate adder consumes 2.1× more power than the shift-register controlled backscatter RF.

It is necessary to add a few caveats to our simplified comparison of computation and communication. The clock rates of communication subsystems are limited by signal to noise ratio considerations, whereas the clock rates of processors can be higher, and thereby reduce power. In addition, low-power processors use many tricks to reduce power consumption including optimized signal processing circuits, different power domains, extremely tight duty-cycling, and so on. Despite these optimizations, the cards are stacked against computation. Backscatter is so incredibly simple in terms of circuitry that even matching the efficiency of backscatter becomes a challenging architectural design problem.

Thus, the crux of our argument is the following: *backscatter drives down the optimal cross-over point between computation vs communication, such that communication of raw data may be preferable to computation in a wider spectrum of real-world scenarios.*

Implications on architecture design: This observation has an immediate implication on the architecture of a backscatter-based sensor platform. Traditional sensing platforms add a lot of computational modules between the sensor and the radio for

sensor data acquisition, processing, filtering, buffering, etc. The contribution of these components to overall power consumption of an active radio-based sensor system is minimal and can largely be ignored. However, on backscatter-based platforms, these components become the bottleneck.

This raises an intriguing question — with the power consumption of backscatter being so low, would it in fact be more efficient to eliminate all of these modules en-bloc, and just connect the sensor directly to the radio? In other words, would it be better to just stream every bit of data that is sensed directly through the radio?

We take a measurement-driven approach towards answering these questions. First, we look at the computational blocks between sensing and the RF interface on existing backscatter-based sensing platforms to understand how much power they consume, as well as why they suffer in terms of throughput. Second, we build on our empirical study and design a radically new backscatter-based sensor platform that addresses these limitations.

4.3 Investigating existing wireless sensing architectures

In this section, we investigate why current backscatter-based platforms are unable to achieve end-to-end power consumption of μWs for high-rate sensing and transfer. We also investigate why they are unable to achieve high-data rate communication, particularly while operating at low power. To empirically understand these factors, we look at the UMass Moo/UW WISP class platforms that are equipped with sensors, a low-power MCU (MSP 430 family) and a backscatter radio.

4.3.1 Poor energy efficiency

We start with a break down of the power consumed by three key computational modules on a UMass Moo (Figure 4.3): 1) the sensor data acquisition subsystem which handles the protocols for operating sensors, 2) the data handling subsystem on

a micro-controller where sensor data is stored, processed (if needed), formatted into packet, and sent to the network stack, and 3) the network stack implemented in a combination of hardware and software.

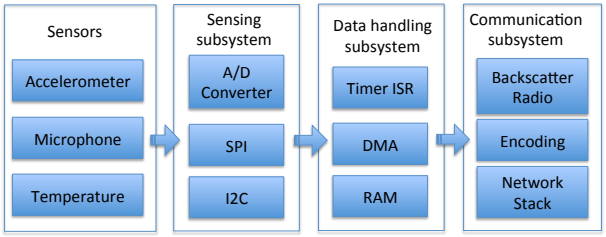


Figure 4.3. Computational blocks on existing backscatter-based sensors.

4.3.1.1 Sensor data acquisition

Sensor data acquisition is a relatively simple operation — some sensors have an on-board ADC, hence data acquisition is via a protocol such as SPI or I2C, whereas other sensors just provide an analog signal which is digitized using the micro-controller's ADC. Despite its simplicity, even these operations are not as cheap as one might expect. For example, sampling an accelerometer via the SPI bus would require periodic wakeup of the MCU to fill the SPI buffer, sending the read command and read address to the sensor, as well as providing the clock for the SPI bus. The overall result is that the MCU is active for about 40% of the time when acquiring data from an accelerometer sampling at 400 Hz. This acquisition operation, in itself, consumes 84μW of power, 14$\times$ higher than the accelerometer (6μW). The cost of acquiring audio data is equally high — when sampling an audio sensor (ADMP803) at 44KHz, acquisition consumes 492μW of power, 14.5$\times$ higher than the audio sensor (34μW).

4.3.1.2 Data handling subsystem

The data handling subsystem is the block that processes the acquired sensor data, formats and packetizes it, and sends it to the network stack. To minimize this overhead, sensor systems typically operate in a duty-cycled mode where the MCU is

turned on for a minimal amount of time needed to handle the data, before switching back into sleep mode to conserve energy.

However, this optimization is no longer effective when this subsystem handles high-rate sensors. Figure 4.4 shows the power consumption for executing the timer interrupt service routine to handle each acquired audio sample. At high rate, the MCU is rarely able to switch completely back into the ultra-low power sleep mode due to frequent interrupts. Thus, the overall power consumption of the data handling module is roughly the ballpark of active mode power consumption of the MCU (a few mW), which is several orders of magnitude higher than the power consumed by the sensor.

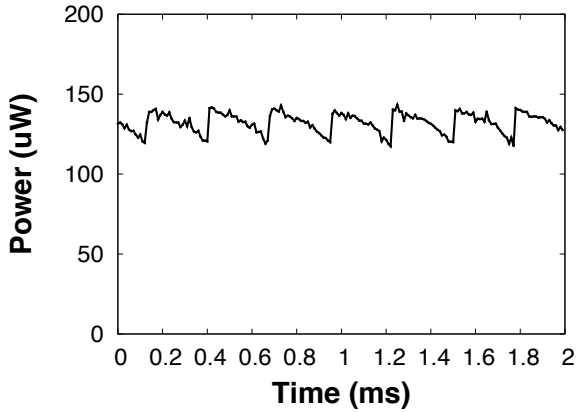


Figure 4.4. Power consumed for handling timer interrupts at 4kHz. The MCU is unable to switch to sleep mode due to frequent interrupts.

One method to reduce power of the data handling subsystem is to use Direct Memory Access (DMA), which allows transfer of data from the sensor to memory without waking up the MCU. This raises the possibility that waking up the MCU can, perhaps, be avoided altogether if the data is transferred directly from the sensor to the network queue without any processing.

Surprisingly, DMA does not reduce power consumption. Figure 4.5 shows empirically measured power consumption for DMA transfer on an MSP 430, which moves

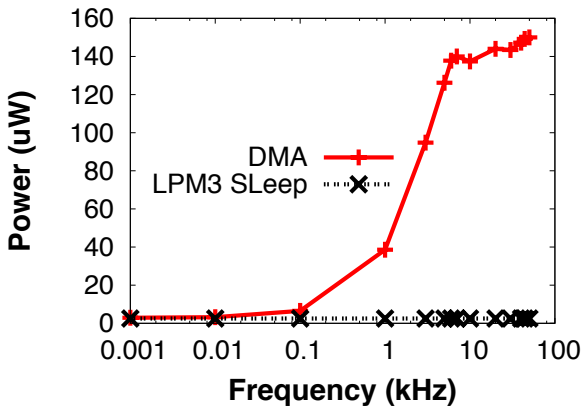


Figure 4.5. The power consumption of DMA transfer at different frequencies.

the sensor data from a sensor to a local memory at different frequencies. We observe that while DMA is efficient at low rates (e..g below 100Hz), it has high power consumption at high transfer rates — for example, DMA transfer consumes 149.2 μW of power at 44 kHz, 60$\times$ higher than the 2.5 μW of LPM3 sleep mode of the MCU. This is surprising since one would expect that the MCU is in sleep mode while DMA operates.

The culprit for high power consumption of DMA turns out to be its tail energy consumption. Figure 4.6 shows the power consumption of repeated DMA transfer at 100 Hz. This experiment is done with an MSP 430 set to LPM3 sleep mode and a timer that periodically triggers DMA transfer. When a DMA transfer is initiated, its power consumption increases to 40μW within 10us, and starts decreasing once the DMA transfer is done. However, the power consumption decays at a relatively slow rate compared to the sharp increase, resulting in a long tail of roughly 3.5ms. When the DMA transfer frequency is high, such as 5kHz shown in Figure 4.5, the long tail leads to high power consumption. While we are not certain about the cause of this behavior, one hypothesis is that the system waits for more data before it times out and switches to a lower power mode. This behavior is common in many power savings

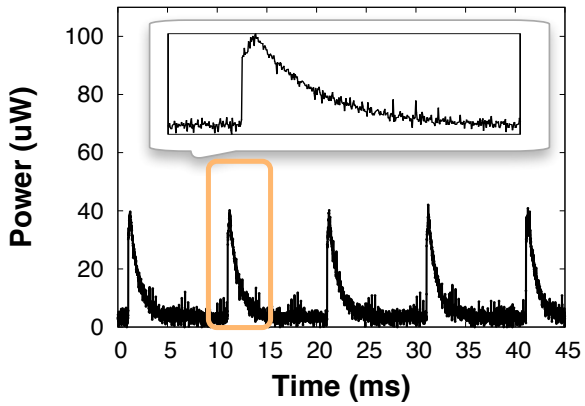


Figure 4.6. Power consumption of DMA transfer at 100Hz. DMA is slow to return to sleep mode.

circuits, for example, in smartphone radios [16, 33], and is typically done to amortize the cost of waking up and shutting down a hardware subsystem.

4.3.1.3 Communication subsystem

The final computational component of a sensor platform is the communication stack, which includes the PHY, MAC and upper layers. While the RF interface of backscatter is extremely low-power, the other layers add more overhead. For example, on the UW WISP or UMass Moo platforms, the backscatter radio is controlled by a hardware timer which needs to be configured and handled in software. In addition, the EPC Gen 2 network stack on these devices is implemented in software, and results in substantial overhead since the MCU needs to handle protocol messages. In fact, the MCU needs to be on for 67% of the time for processing network stack messages at the software layer while only 7% of the time is used for data transmission. As a consequence, the software on UMass Moo platform consumes 2mW of power, which is three orders of magnitude higher than the power consumption of a low-power sensor.

As with the data handling subsystem, the software overhead of the network stack can be reduced by using hardware peripherals to control the radio. One commonly available hardware peripheral on MCUs is the Universal Asynchronous Receiver and

Transmitter (UART). This is particularly useful for a backscatter radio since UART generates an ASK signal, which can be directly transmitted via backscatter (which uses OOK). At the first glance, the UART peripheral has the potential to dramatically reduce the cost of running the network stack because it can operate when the MCU stays in deep sleep mode. However, its buffer needs to be filled with sensor data, which in turn needs to be done with either DMA or software, both of which are expensive energy-wise. As a result, even the UART-driven backscatter radio consumes roughly 2mW.

4.3.2 Poor transmission efficiency

The second key drawback of existing backscatter-based sensors is the abysmal throughputs that they achieve. For example, even though there have been many efforts to improve backscatter throughput, the ceiling is still less than 20kbps for a single node [27, 57, 20, 48], and drops to hundreds of bits/second in a network with multiple devices. Clearly, this is far below what is needed for streaming raw sensor data from high-rate sensors.

One factor that limits the throughput is the poor efficiency in clock utilization. For example, the UMass Moo and WISP take 48 clock ticks to send a single bit of data, which causes a 48× reduction of the maximum possible throughput that is achievable with the system clock. We find three reasons for this inefficiency. First, both transmission and reception logic is implemented in software which, naturally, is inefficient in the use of the clock. Although the transmission and reception code on the Moo and WISP platforms are optimized in assembly instructions, one bit transmission and reception still has substantial overhead. Second, EPC Gen 2 PHY-layer encoding further reduces the clock utilization efficiency. To minimize the DC components during data transmission, each bit is encoded into a sequence of pulses using Miller encoding. For example, the Miller-4 encoding used by Moo and WISP

platforms uses eight pulses to encode one bit of data, resulting in further drop of throughput by a factor of eight. Third, the EPC Gen 2 MAC layer is extremely inefficient for high bandwidth data transfer. While this is a point that has been made many times before [27, 57, 20, 48], an efficient alternative that achieves high throughput using backscatter is lacking.

4.3.3 Summary

Thus, the limitations of the computational blocks on existing backscatter-based sensor platforms lead us to the following observation. The primary culprit in terms of power is the MCU's active mode power consumption, and the fact that many operations (sensor acquisition, data handling, communication) require execution of instructions on the MCU. Surprisingly, optimizing the system by leveraging hardware peripherals such as DMA and UART do not solve the problem, particularly at high data rates due to tail power consumption, and coupling between different components of the sensing to communication pipeline. In terms of throughput, the primary issues stem from inefficient utilization of the clock due to a combination of software overheads, encoding overheads, and an inefficient MAC layer standard. In conjunction, these limitations call for a *clean-slate re-design of a backscatter-based sensor platform* from the ground up for extremely low power consumption and high data rates.

4.4 The Ekho platform

Our solution is Ekho, a backscatter-based sensor platform that is optimized for ultra low power operation and high-speed streaming from sensors. We outline the platform architecture followed by the MAC layer.

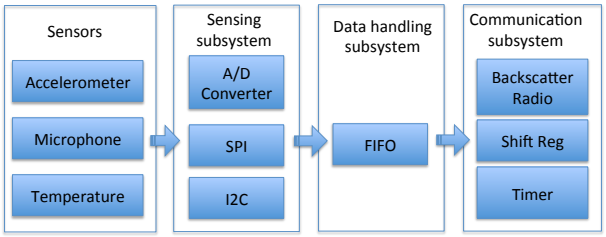


Figure 4.7. The key components of Ekho.

4.4.1 Eliminating computational blocks

At the platform level, the design of Ekho is minimalist. We simply remove as many computational blocks between the sensor and RF analog front end as possible in favor of communicating raw data. Figure 4.7 shows the key components in Ekho.

Ekho reduces the overhead of data acquisition from the sensor by implementing the SPI and ADC sampling logic on a small CPLD (FPGA). Implementing these blocks in hardware means that we can make them as fast as needed without incurring the software overhead of waking up a micro-controller.

Ekho substantially reduces the overhead of handling sensor data by a minimalist approach that uses a FIFO buffer between the sensors with RF analog front end. The FIFO buffer is the minimum element that is needed between sensing subsystem and communication subsystem to deal with short delays in transmitting the data over the backscatter link, for example, due to intermittent scheduling of a device. In this manner, Ekho eliminates software and tail energy overhead that was observed on existing backscatter-based platforms.

The final computational component of the pipeline is the communication subsystem. Unlike EPC Gen 2 that is designed for a broad range of RFID tags, Ekho is designed solely for streaming sensor data from nodes to a reader. A protocol designed solely for streaming data from sensors can be quite simple. The reader informs each node of a timer value that specifies the period with which to transfer data in its FIFO buffer, and a rate that determines how fast to transfer the data. The only hardware

component required for this protocol to work is a timer and shift register. Once the timer fires, a shift register converts the input sensor data to an ASK signal that is used to modulate backscatter radios.

In the current instantiation of Ekho, we do not perform any encoding of data. While the need for encoding to deal with harsh wireless conditions and interference is well-known, it also makes the hardware more complex, and consequently more power hungry. For example, the default configuration on the UMass Moo/UW WISP platforms is Miller-4 encoding incurs overhead of several hundreds of gates. Thus, while encoding may be useful in some cases, we do not employ it in Ekho.

4.4.2 The EkhoNet MAC layer

We now turn to the second part of our performance puzzle — achieving high throughputs that are upwards of many hundreds of kilobits/second across different nodes in the network. A high speed MAC is important for supporting an architecture where raw data transfer is the norm rather than the exception.

MAC layer designs are very well understood, particularly in cases such as ours where a central controller performs TDMA-like scheduling of sensor nodes. However, the key point in our design is two-fold: a) even though the sensor node is designed to be extremely simple, the decision making logic can be placed at the reader, thereby enabling surprisingly complex scheduling mechanisms across a network of extremely simple sensor nodes, and b) our MAC is holistic in that it takes into account utility of data, channel-awareness, energy consumption, as well as other hardware considerations, in-order to maximize throughput.

4.4.2.1 MAC Design Considerations

At the heart of EkhoNet is the logic that is used to determine when each node should transfer, and what rate they should transfer. Before we answer this question, we need to understand several characteristics of Ekho including: a) how do MAC-layer parameters impact the energy-efficiency of the platform? b) what are the signal-to-noise ratios at which data transmitted by Ekho can be successfully decoded? c) what criteria should we use to decide what sampling rate to use when sufficient bandwidth is not available? and d) what are the implications of platform considerations such as clock drift and buffer size? We now empirical examine these considerations in greater detail, and discuss the implications on selection of MAC layer parameters.

Bits/Joule: The first question we ask is how energy-efficiency of data transfer depends on the bit rate. Figure 4.8 shows the efficiency of a shift register controlled backscatter radio across different bit rates. At low rates, there is a steep increase in efficiency as bit rate increases due to the fact that constant power consumption by the system is amortized over more bits being transferred. However, improvements in efficiency diminish once the bit rate increases beyond 1Mbps since the relationship between power and frequency of the shift register is roughly linear, hence there are not much improvements possible. The power curve suggests that, from energy perspective, we should choose the fastest bit rate possible for data transmission.

Signal to Noise Ratio: While faster bit rates are preferable due to higher energy efficiency of transfer, SNR degrades as bit rate increases. Figure 4.9 shows the SNR when we deploy a transmitter 1 meter from the reader and change its transmission bit rate. As bit rate increases, the SNR decreases steadily as one would expect. When the SNR is lower than 10dB, decoding becomes difficult on our software-defined radio based reader platform, which gives us an upper bound on the fastest bit rate that can be supported by the system without losing bits.

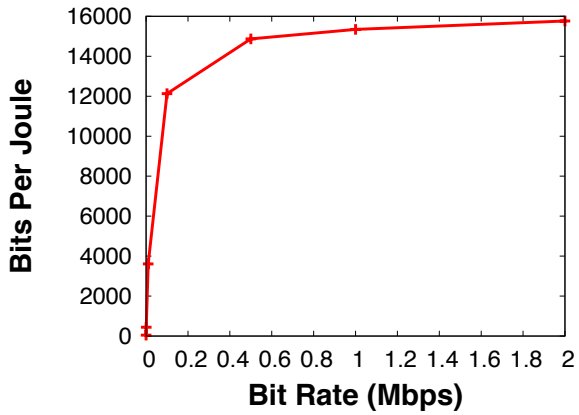


Figure 4.8. Efficiency of backscatter radio (in bits/joule).

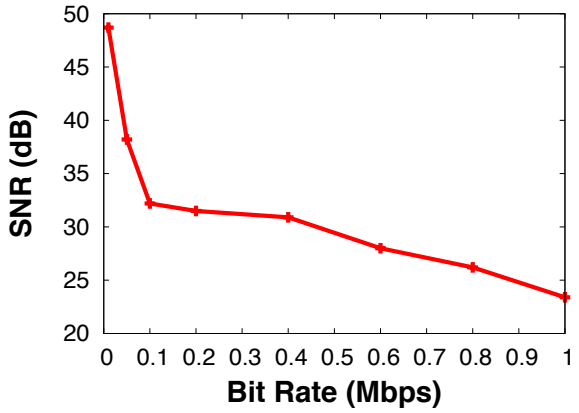


Figure 4.9. SNR at different bit rates when device is placed 1m from a reader.

Utility of data: Since EkhoNet is designed for high-rate sensors, one question that needs to be addressed is how to decide on appropriate sampling rates when the overall data rates at full sampling rates exceed capacity. On our existing system, we are limited to 1Mbps aggregate transfer rate across all nodes since the SDR-based reader is only able to support 8M samples per second due to the limitations of the realtime signal processing logic. This means that we can easily reach the SDR limit when we operate a network of sensors. For example, a network of five audio sensors sampling at 44 KHz, and transmitting raw data generates an aggregate bandwidth of 3.5Mbps, well above what can be supported by EkhoNet.

Our solution is to take into consideration the utility of data generated by the different sensor nodes. Figure 4.10 shows an example of one utility function, Mean Opinion Score (MOS), which is a commonly used metric for characterizing the quality of transmitted audio [9]. The MOS score can be used to guide decisions regarding which node is allocated bandwidth.

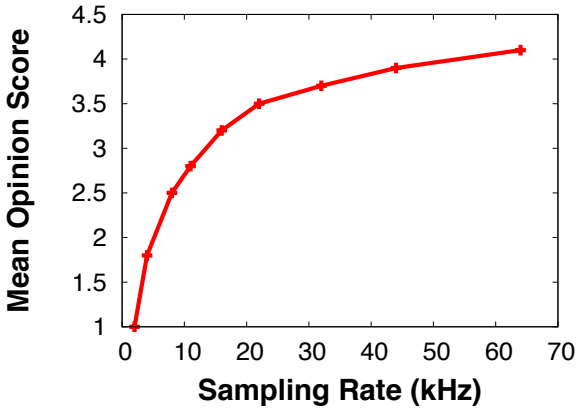


Figure 4.10. Mean Opinion Score (MOS) at different sampling rates for a microphone.

Clock drift: Another consideration in determining slot sizes is clock drift. For example, in our implementation of Ekho, we use a crystal oscillator driven system clock that can drift at upwards of 50 ppm. If two nodes transferred at 1 Mbps, then they would drift by 1200 clock cycles each minute. The reader can handle clock drift in two ways. First, when assigning slots, it can allocate guard bands in each slot to allow for some drift. However, guard bands should be kept to a minimum to reduce bandwidth wastage. Second, the reader has the luxury of observing how the gap between slots varies as nodes transfer, and can detect when collision occurs by looking at the constellation plot of the signal [48]. Thus, when the reader suspects that slots have bled into each other, it can send a reset pulse that informs all nodes to reset their timers. Note that this is possible for backscatter because reader messages are broadcast and received by all nodes. A reset pulse is simply implemented by

shutting off the carrier for a short, pre-defined duration, which is detected by each node. While reset pulses can be short, it should be used infrequently since there can be robustness issues if a node does not receive the pulse. This can result in further collisions resulting in more reset pulses until the network synchronizes.

Buffer size: One additional constraint introduced by the Ekho hardware platform is that the FIFO buffer size on the device is limited, hence if the slot sizes are too long, samples will be lost since the buffer will overflow.

4.4.2.2 Channel-Utility-Energy aware Rate Selection

Given the above constraints, the overall problem that the reader faces can be described as follows: select the optimal bit rate and slot size such that aggregate utility of received data is maximized and aggregate energy consumption minimized, subject to constraints on the buffer sizes, SNR, and guard bands. We formalize this problem below.

We assume that the following parameters are given:

- The minimum SNR, 10 dB in our system, at which the reader can decode bits with low bit-error rate.
- The maximum achievable bit-rate r_i that is higher than the minimum SNR.
- The maximum sampling rate of each node $s_{max}(i)$.
- The size of each sample in bits, b, bits/sample.
- The fraction of each slot that should be a guard band δ.

Given these values, we need to choose the sampling rates for each sensor s_i, and the fraction of time allocated to each node t_i by taking into account the following objective:

- $\sum_{i=1}^{n} U(\mathbf{s})$ which is a measure of the aggregate utility obtained from all sensor data received from the nodes.

The constraints are the following:

- $\sum_i t_i \leq 1$, i.e the fraction of time allotted to nodes sum up to at most one (less than one if the network is operating below its limit).
- $s_i \leq s_{max}(i)$, which restricts the sampling rate for a sensor to be below the maximum.
- $(1-\delta)t_i r_i = b\, s_i$, which ensures that the production of data from the sensor, and transmission of data from the radio are matched i.e. the node can transmit what is being sensed. The term $(1-\delta)$ is present since there's a guard band for each slot.

The overall optimization is shown below (in vector form for compactness). Here, $\mathbf{s}$ and $\mathbf{t}$ are the vectors of sampling rates and the fraction of time allocated to each node, which need to be determined, and $\mathbf{r}$ is the vector of bit rates chosen for each node based on SNR. The symbol $\preceq$ stands for element-wise inequality (i.e. one for each node).

$$\begin{aligned} \underset{\mathbf{s},\mathbf{t}}{\text{maximize}} \quad & \mathbf{1^T} U(\mathbf{s}) \\ \text{subject to} \quad & \mathbf{t^T 1} \leq 1 \\ & \mathbf{s} \preceq s_{max}\mathbf{1} \\ & (1-\delta)\text{diag}(\mathbf{t})\mathbf{r} = b\,\mathbf{s} \end{aligned}$$

Typically, the utility function is concave, for example in the case of MOS score (Figure 4.10). Hence, the objective is to maximize the sum of concave utility functions, and the constraints are linear, hence the optimization can be solved by standard convex optimization methods. Note that the optimization returns the fraction of time

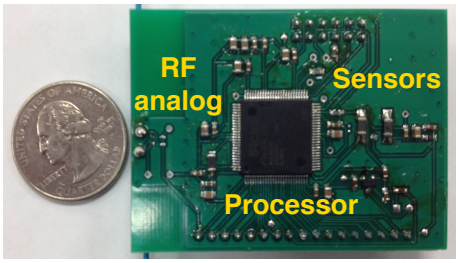


Figure 4.11. Ekho is implemented as a low-profile printed circuit board with small form factor.

for each node — this can be converted to an actual slot size by scaling by an appropriate period such that each node is capable of buffering the data in its local FIFO buffer.

4.5 Implementation

Figure 4.11 shows the prototype of Ekho, which implements all the design elements described in section 4.4. The current prototype measures 1.8 by 2.4 inches, but we believe future revisions can shrink this even further. We now briefly describe the key sub-components used in the prototype.

4.5.1 Hardware

The first key hardware element is an ultra low power FPGA (Igloo Nano AGLN250) that manages the various sub-components of the Ekho platform. Most key components of the Ekho architecture, including the sensing, data handling, and communication subsystems, are implemented within the FPGA. The particular FPGA was chosen because it has low static current consumption and has a 32k bits (2KB) RAM, which also determines the maximum size of our FIFO buffer.

The next key design element is the backscatter circuit that can operate at high speed. As the device toggles the state of a transistor that connects to the antenna, an OOK signal that carries modulated information is generated. However, on existing

backscatter platforms, the static current of the transistor is provided by the harvested RF energy, which might vary across time. The varying RF power affects the amount current that is provided to the transistor and leads to unstable edges of the generated OOK signal. Therefore, decoding becomes challenging when the data rate is high. Our backscatter circuit directly provide a small bias current to the transistor and retains a sharp edge for the generated OOK signal.

A critical element of our hardware design is the clock system which drives the FPGA logic. The core of our clock system is a 1MHz ultra low power crystal oscillator that directly feeds into the FPGA. The 1MHz clock is divided to drive different components of the architecture because sensing, data handling, and communication subsystems operate at different speeds. Our clock system is different from the Moo and WISP platforms, where a digital generated clock (DCO) is used. Although the DCO can also be divided for driving different components, it couples the operational modes of the system and its clock speed, as a result of which the high speed clock is only available when the system operates as a whole in a high power mode.

4.5.2 Software defined backscatter reader

We used the USRP N210 mother board and the SBX RF daughterboard to build our software defined backscatter reader for receiving high speed backscatter signals from Ekho. We construct a signal processing pipeline that is able to track the amplitude of the carrier wave that is used as the reference for decoding the OOK signal generated by Ekho. Our decoding is different from Moo and WISP platforms where Miller-4 encoding is used on top of the OOK signal and a decoding template can be used for correlating the received signal and output a bit when the template matches the received signal. In Ekho, the data is sent directly via OOK and encoding is not used. Therefore, we need to track the amplitude of carrier wave to determine whether the received signal is a high or low pulse.

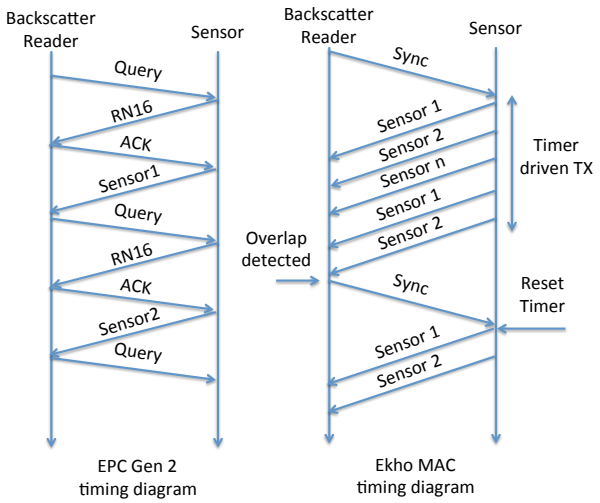


Figure 4.12. Timeline of Ekho MAC.

4.5.3 MAC layer protocol

Figure 4.12 shows the timing diagram of the Ekho MAC layer. The first stage is to inventory the nodes in the network, and obtain information about their SNR and other sensor-related information. This phase executes very similar to an EPC Gen 2 singulation phase, where nodes can select a slot to transfer in, and send a short sequence of bits with the appropriate information. After the singulation phase, the reader executes the optimization algorithm described in §4.4 and determines the time period and bit rate for each sensor, which is then relayed to the sensor. The reader initiates the singulation phase under several circumstances: a) when significant changes are observed in SNR, which might signify changes in position or orientation, and b) when collisions are detected, which might signify that a new node is attempting to join the network.

Once the reader informs each sensor of its bit rate and period, it initializes slots by sending a synchronization signal during which it shuts down the carrier for a short 10 μs window. This pulse informs all nodes simultaneously that they should start their timers, thereby initiating the TDMA schedule. The length of the sync message needs

to be chosen small enough to amortize overhead, but large enough to be detectable at the sensor, hence our choice of 10μs.

When the reader detects that data transmitted during adjacent slots are overlapping into each other (due to clock drift), it re-issues a synchronization pulse to restart the timers on all nodes. Overlap between sensors can be detected by looking at the constellation map of the received signal — if two clusters are present, it indicates that a collision-free signal is received and if more clusters are present, it indicates that a collided signal is observed [48]. If multiple synchronization pulses fail to eliminate collisions, the reader switches back into inventory mode.

4.6 Evaluation

We now evaluate the overall performance of EkhoNet including 1) demonstrating the power benefit of the Ekho architecture, 2) benchmarking the performance of the EkhoNet MAC, and 3) evaluating EkhoNet's ability to support high-rate streams from many sensors while operating at extremely low power consumption.

4.6.1 Experimental setup

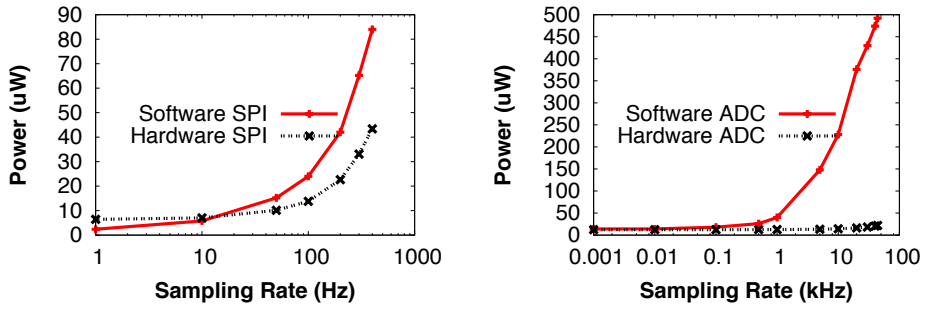
We deploy 10 Ekho nodes 1 feet to 9 feet from a backscatter reader. Our experiments do not cover distances larger than 9 feet because of the poor signal quality beyond 9 feet. This is a result of the 100mW maximum power issued by the SBX RF daughterboard, which is 10× smaller than commercial RFID readers.

To understand the power benefits of Ekho, we compare against the UMass Moo (equivalent of Intel WISP 4.0) and the WISP5.0 platforms. Since the WISP5.0 platform is not currently available, we evaluate its power consumption with a prototype that uses the same MCU (MSP430FR5969). Since the MCU is the main power hog in the system, this provides a good proxy for measuring power consumption.

4.6.2 Ekho power benchmarks

We begin our evaluation by validating the claim that the power optimizations on Ekho can substantially reduce the overheads incurred by existing platforms. We follow the organization in §4.4, and show benchmarks for each module — sensor data acquisition, sensor data handling, and network stack.

Figure 4.13 measure the power of the sensing subsystem when Ekho interacts with two types of sensors — an accelerometer with on-board ADC that connects to the MCU via a SPI interface, and an audio sensor where the MCU's ADC is used to sample the sensor. We compare Ekho versus a WISP/Mote-class sensor device (i.e. a device where the sensor connects to an MCU that acquires data). In both cases, we can see that Ekho reduces power consumption substantially — for sampling the accelerometer, Ekho reduces power by 1.5× at 400Hz by eliminating the overhead of software-controlled SPI, and for sampling the audio sensor, Ekho reduces power by 22× by trimming the overhead of running the software-controlled ADC at 44kHz.



(a) Accelerometer. (b) Microphone.

Figure 4.13. Power reduction for sensing subsystem: a) sampling an accelerometer, b) sampling a microphone.

Figure 4.14 measures the power consumption of the data handling subsystem of Ekho, which is composed by a 2kB FIFO buffer for connecting sensors to the RF analog front end. The 2kB FIFO buffer only consumes 26.5μW of power when data is written into the FIFO at 500kHz, 14.4× lower than the 384μW consumed by DMA

driven data migration and 92× lower than the 1.5mW consumed by timer driven data migration.

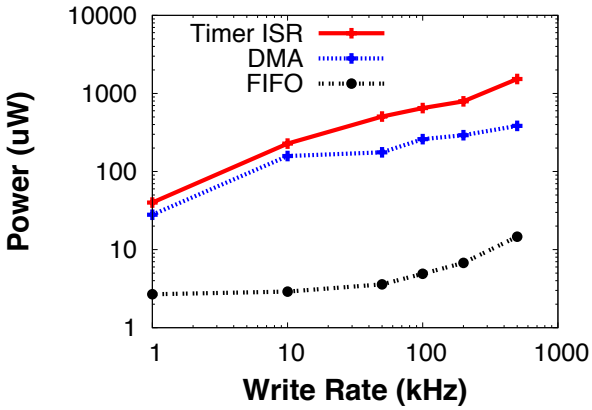


Figure 4.14. Power reduction for data transfer to network queue.

Figure 4.15 shows the power consumption of the communication subsystem which is composed of a shift register and backscatter radio. At 1Mbps, Ekho's communication subsystem consumes only 77μW of power, 13.4× lower than a UART controlled backscatter radio implemented on the WISP and 44× lower than a software controlled backscatter radio implemented on the WISP. For software and UART controlled backscatter radios, we do not measure power at bit rates higher than 6Mbps because the maximum clock on rate on WISP platform is 24MHz, which limits the maximum achievable bit rate.

4.6.3 Whole-system power consumption

Having looked at power benchmarks for individual components of Ekho, we turn to a whole-system power measurement from sensing to transmission. We look at the overall power consumed by Ekho when operating the same two sensors as earlier — accelerometer and microphone.

We start with a measurement of Ekho with an accelerometer. The sensor has a built-in ADC and talks via SPI to the sensor platform. Figure 4.16 shows that at 1Hz,

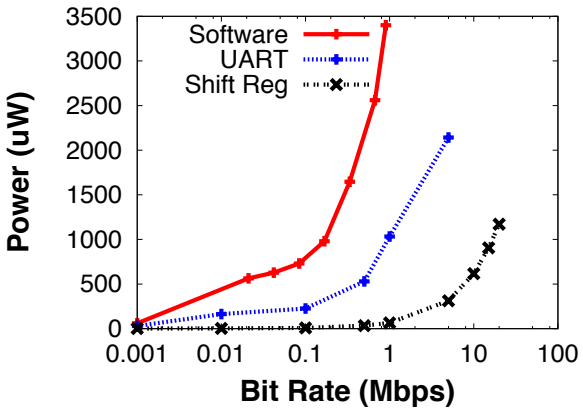


Figure 4.15. The power consumption of operating a backscatter radio.

the power consumption of Ekho is higher than Moo and WISP5.0 platforms. This is because the static current consumption of the FPGA at the core of Ekho is 8.9μA, much higher than the 0.1μA static current draw of Moo and WISP5.0. However, when the frequency of operating the accelerometer increases, the power consumed by Moo and WISP5.0 platforms increases significantly while the Ekho system still consumes only tens of μW. At 400Hz, the Ekho system consumes 35μW of power, 7.6× lower than the 266μW of Moo and 3.3× lower than the 118μW of WISP5.0.

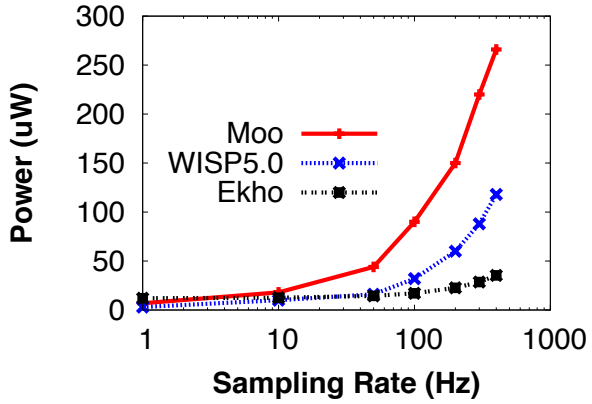


Figure 4.16. Whole-system power consumption for operating an accelerometer sensor.

We now turn to power measurements when Ekho is connected to a microphone. An external ADC is used to sample the audio sensor, and send a digital signal to the core platform (Moo, WISP5.0, or Ekho). Figure 4.17 shows the power consumption of the three platforms. At 44kHz, the Ekho system only consumes 37μW of power, 76× lower than the Moo and 13.5× lower than the WISP5.0.

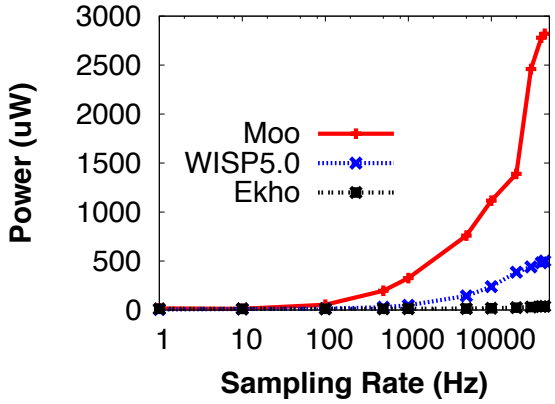


Figure 4.17. Whole-system power consumption for operating an audio sensor.

In conclusion, Ekho is particularly efficient when using higher rate sensors that sample at frequencies of hundreds of Hz. A crucial observation is that even when the sensing rate increases by two orders of magnitude from the accelerometer at 400Hz to the microphone at 44kHz, the overall power consumption remains almost the same. This shows that Ekho scales up very well as sampling rate increases. In addition, Ekho is able to operate with sensors that use SPI or provide an analog signal while retaining high efficiency.

4.6.4 Evaluating EkhoNet's throughput

Having discussed the power benefits of Ekho, we now turn to look at the performance of Ekho's transfer rate.

We start with the throughput achieved by a single node. Since the Moo and WISP platforms currently support only a 256Kbps baud rate, we fix Ekho's clock to operate

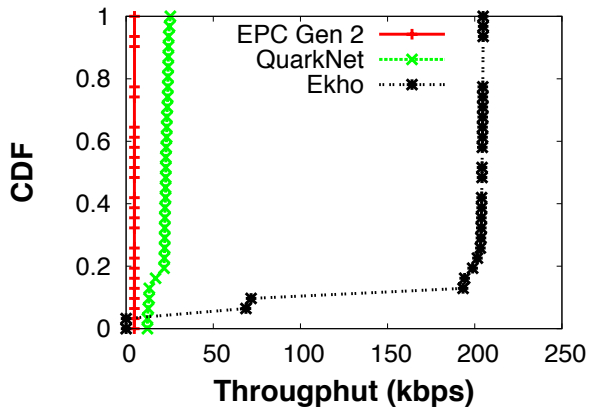


Figure 4.18. Comparing throughputs of EPC Gen 2, QuarkNet on Moo vs Ekho across 30 locations.

at the same rate. We then compare Ekho's throughput against the Moo executing EPC Gen 2 [53], and QuarkNet [54]. Figure 4.18 shows the cumulative throughput across 30 locations. The 30 locations are chosen randomly between 1 feet to 9 feet from a backscatter reader.

There are two key observations. First, we see that the throughput achieved by Ekho is 45× higher than Gen 2 and 8× higher than QuarkNet on the Moo. EPC Gen 2 suffers greatly due to protocol overhead, and therefore achieves abysmal overall throughput. Although QuarkNet is a highly optimized system that is designed for micro powered sensors, its throughput is limited by the fact that the PHY layer (encoding, etc) is implemented in software on Moo, which reduces throughput. Second, we see that there are a few locations where our design decision to eschew encoding hurts us. At those locations, the received signal can still be decoded by EPC Gen 2 and QuarkNet because of the SNR benefit of Miller-4 encoding. However, it can be seen that this is a small fraction of the overall range of the reader. (Note that if encoding is essential, it is possible to add this module to Ekho at the cost of some additional power consumption and reduced throughput.)

We now turn to the throughput achieved by a network of nodes, and evaluate the benefits of our energy and utility function aware bit rate selection algorithm. We deploy 10 Ekho nodes with microphones at three locations (3 feet, 6 feet, and 9 feet from a backscatter reader). The maximum sampling rate of each audio sensor is 44kHz and each sample data is 16 bits. As a result, an audio sensor can generate up to 706k bits data per second. In contrast, the overall network transmission capacity of EkhoNet is 1Mbps in our current instantiation since each device is equipped with a 1Mbps clock. Thus, 10 audio sensors in front a backscatter reader can saturate the 1Mbps network easily, which means that adapting the bit rate as well as the sampling rate of each sensor is necessary.

When channel is saturated, the selection of bit rate is intuitive because maximum bit rate which meets the lowest SNR decoding threshold (10dB) should be used. The selection of sampling rate follows the energy-utility joint optimization we formulated in §4.4.

Figure 4.19 shows the MOS score obtained by 10 audio sensors at 3 locations. Our optimization framework attempts to allocate bandwidth such that sensors with higher SNR can get the bandwidth they need for achieving higher MOS scores. As a baseline, we compare against a scheme that allocates bandwidth equally across all sensors. The median and mean MOS scores achieved by EkhoNet is higher than the baseline scheme — 50% of the nodes have MOS scores higher than two, which is acceptable audio quality, whereas the uniform allocation scheme has MOS scores of about 1.7, which means poor audio quality. A breakdown across nodes shows that our algorithm assigns higher sampling rate to sensor 1 to 5 because they have higher SNR. While other application-specific utility functions are possible, these results demonstrate that despite the simplicity of Ekho platforms, the EkhoNet MAC can be more complex and optimize network-wide throughput, energy and utility.

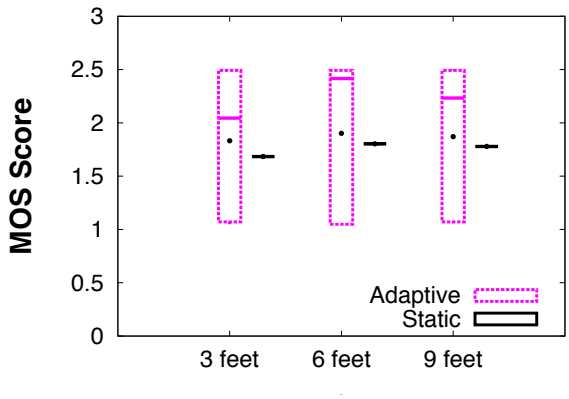


Figure 4.19. Boxplot of the MOS scores for 10 Ekho nodes with microphones at 3 locations (3 ft, 6 ft, 9 ft).

4.7 Related Work

Backscatter communication: There has been much recent emphasis on backscatter communication. Some efforts have explored bandwidth limitations of backscatter communication in terms of throughput including Flit [27], Buzz [48], and Blink [57]. While there are interesting ideas underlying each of these, the overall throughput achieved by EkhoNet is orders of magnitude higher than the above systems as a result of a clean-slate design. Other efforts have focused on using harvested power in an efficient manner including QuarkNet [54][56] and Dewdrop [20] — these approaches are complementary to EkhoNet and can be used in conjunction with the ideas in this chapter.

In addition to the above, there have been many interesting ideas on using backscatter for real-world applications. Ambient Backscatter [38] uses the backscatter of FM signals for short-range communication between tags to enable credit-card transactions. AllSee [35] explores the backscattered signal for gesture recognition. These ideas can potentially benefit from an Ekho-like platform that is designed to reduce power consumption while increasing bandwidth.

Much literature has explored the design of MAC layer protocols for RFIDs, and several of these approaches specifically address data collection from RFID-scale sensors [21, 27, 48, 54]. Viewed in isolation, our MAC layer protocol is simplistic since its merely a stripped down version of TDMA, hence it relates to most of the above protocols. However, our work should be viewed not just as a MAC layer, but a system-wide re-design to strip computational overhead from backscatter-based sensors, and thereby achieve higher efficiency.

Optimized sensing platforms: There have been many highly optimized sensor hardware designs proposed over the past decade. At a high level, these can be separated into two classes — optimized hardware platforms designed for specific applications, and optimized hardware platforms that are intended as a building block for research and applications. One example in the former class is the NeuralWISP [30], a wireless neural interface that operates on harvested RF energy. Some examples in the latter class are the Michigan M^3 [37], an impressive mm^3 sensor that operates at low power, and the Epic Mote [23], which is a modular mote-class platform for enabling low-power wireless sensor network applications.

EkhoNet differs from these efforts in that it is designed for raw data transfer from high-rate sensors at extremely low power levels. Thus, it is a general-purpose platform for sensors similar to the second class of devices, but focused on backscatter and high-rate sensors. As a result, the underlying design principles and optimizations are completely different from those that drive the other class of platforms.

4.8 Discussion

While Ekho provides substantial performance benefits over the state-of-art in backscatter-based sensor platforms, there are several questions that we have not completely addressed in our evaluation. We discuss these in this section.

FPGA v.s. MCU: One of the design choices in Ekho is the use of an FPGA rather than MCU — this choice greatly reduces the computational and data migration overheads between the sensor and radio, but in the process, it sacrifices ease of programmability. While FPGA programming has become easier in recent years due to improved IDEs and GUI interfaces [8], it requires familiarity with logic design at the circuits level. MCUs, on the other hand, are much more natural to program using commonly used high-level languages such as C, which is one of the reasons for its wide use on sensor platforms.

We believe that the greater difficulty in programming FPGAs is not as much of an issue for Ekho as for other platforms. Wireless sensors are designed to be intelligent, autonomous nodes that can adapt to dynamics in energy levels, channel conditions, routing changes, and others. In contrast, Ekho is designed to be a "dumb" peripheral for a powerful reader that simply forwards the raw sensor data over a backscatter link. Much of the decision-making logic that is traditionally implemented on the sensor side are performed at the reader. Thus, Ekho can be viewed as just another sensor, with an interface that allows the reader to set sampling rates and bit rates (as shown in §4.4).

Power benefits: The results presented in this chapter compare Ekho against existing backscatter-based sensing platforms such as the WISP, but one question is whether we would have significant power benefits if we compared against an FPGA implementation of the WISP. Our evaluation did not address this question since re-implementing the entire sensing, computation, and communication pipeline of the WISP on an FPGA is a substantial effort, but we provide a qualitative comparison.

Existing research work [41] on RFIDs suggests that an EPC Gen 2 tag implemented on FPGA usually consumes 5K to 10K logic gates. Clearly, an EPC Gen 2 tag does not perform any operation related to sensing. Therefore, sensor sampling, data migration, buffering, and other tasks would incur additional overhead. For exam-

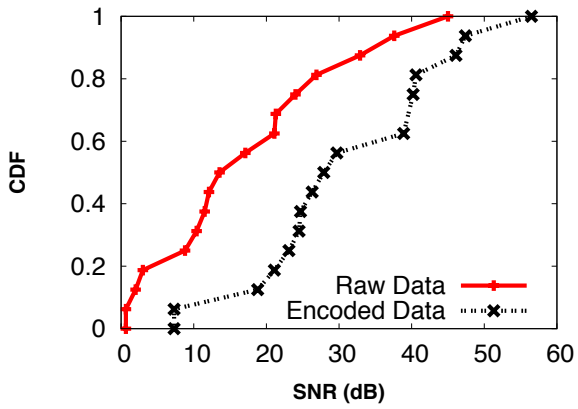


Figure 4.20. SNR of transmitting encoded data and raw data across 20 locations.

ple, Touhafi and Glesner et al [22, 29] investigate an FPGA (Spartan3-2000) based sensing platform which consumes 1200K gates, several orders of magnitude higher than an EPC Gen 2 tag. Our Ekho implementation consumes only 6K gates, which is comparable to an EPC Gen 2 tag and significantly less than what we would expect with an FPGA version of the WISP. Since the power consumption of an FPGA depends on the number of gates used, Ekho should still be significantly more efficient.

Encoding: Another design decision that needs more discussion is that Ekho eschews encoding in an effort to be minimalist. Unsurprisingly, this can be problematic in scenarios where the wireless channel is noisy. Figure 4.20 shows a simple experiment where we place a tag at 20 locations between 1–9 ft in front of a reader, and look at the SNR with EPC Gen 2's Miller-4 encoding, and without encoding. The decoding threshold for our backscatter reader is 10dBm, so any signal lower than this threshold cannot be decoded correctly. As expected, there is about a 10dB difference between encoded and uncoded signal. The SNR is higher than 10dB in 80% of the locations for uncoded data, and higher than 10dB in about 90% of the locations after encoding. This comes at a high cost, however, since the node consumes 8× more power for achieving the same bit rate.

Thus, our point is simply that encoding is yet another computation block on a backscatter-based sensor platform. While the power consumption of techniques like encoding are insignificant in most radios, the pros and cons deserve to be examined more carefully for ultra-low power platforms such as Ekho.

Applications: Finally, this chapter does not focus on applications of Ekho, but we view our work as an enabler for a variety of applications. While the idea of backscatter-based sensing is not new [50], many existing efforts are about networking simple, low rate sensors (e.g. temperature, pressure, etc). But the need for backscatter in such scenarios is debatable — active-radio based wireless sensors operate for years on coin cells at low sensing and communication rates. But rich sensors such as microphones and cameras operate primarily in a tethered manner since data rates are far too high for continuous communication. Our work seeks to bridge the gap, and enable camera networks or microphone networks to stream data continuously in an untethered manner. The benefits of streaming raw sensor data to internet-connected infrastructure is immense since one can use vast amount of computational resources to jointly process the data streams and enable smart applications. A simple example would be continuous speaker recognition and transcription of meeting notes by deploying a tethered reader and dozens of untethered Ekho nodes at different locations in a conference room.

4.9 Conclusion

In this chapter, we present a powerful backscatter wireless sensing architecture, Ekho, that can sample sensors at tens of kHz and transmit data wirelessly at several hundreds of kbps, while only consuming tens of μWatts of power. The key observation in Ekho is that backscatter wireless communication is energy-wise much cheaper than computation. Therefore, by eliminating the overheads of sensing subsystem, data handling subsystem, and communication subsystems, we enable the whole sensing-

communication pipeline to operate at extremely low power. Over the Ekho platform, we design a MAC layer that allocates bit-rates across nodes while taking into account energy-efficiency, utility of data, and a variety of platform-level considerations. We believe that EkhoNet can enable new explorations in backscatter-based sensing systems, and enable new applications that use ultra-low power high-rate sensors.

CHAPTER 5

LEVERAGING AMBIENT WIRELESS SIGNALS FOR BACKSCATTER

Deploying backscatter on mobile and wearable devices is hard because of the lack of an incident carrier for carrying backscattered information. Existing mobile and wearable devices do generate a carrier. However, this carrier is not directly transmitted. Instead, it is used to modulate a baseband signal before transmission. To make matters worse, backscatter readers are not already widely deployed. So we cannot just leverage existing infrastructure for backscatter. In this chapter, we address this challenge by leveraging multiple ambient wireless signals, such as WiFi and BLE, for carrying backscattered information. We explore key factors that enable the deployment of backscatter on commercial WiFi and BLE radios.

5.1 Background and Motivation

One significant bottleneck of deploying backscatter systems on mobile and wearable devices is the lack of an existing backscatter reader infrastructure. Unfortunately, backscatter readers are neither integrated with existing mobile and wearable devices nor deployed in the typical building environment. As a result, a backscatter-based sensor cannot simply leverage existing infrastructure for communication.

In order to tackle this challenge, recent research, including Ambient Backscatter [38], WiFi Backscatter [34], and BackFi [18], investigate a design that exploits existing wireless signals, such as TV and WiFi, for carrying backscattered information. Ambient Backscatter enables communication between multiple battery-less devices by

reflecting and modulating a TV signal. Both WiFi Backscatter and BackFi exploit similar ideas by reflecting and modulating WiFi signals. However, these systems either operate at a close distance and low data rate or require hardware modifications on commercial radios for performance improvement. Both limitations inspire us to ask a question — how should we leverage existing wireless signals for carrying backscattered information?

We argue that Ambient Backscatter, WiFi Backscatter, and BackFi only explore limited positions of the practical design space of leveraging existing wireless signals for backscatter. For example, all three systems leverage a single signal source for reflection. However, we usually can observe multiple and diverse wireless signals in our environment. For example, we can simultaneously observe WiFi, Bluetooth, and Zigbee signals on the 2.4GHz ISM band. Instead of a single signal, many of these signals can be potential carriers for backscatter. One benefit of leveraging multiple incident signals is improving the robustness of backscatter because when one incident signal is weak or even absent, other carriers could be present and strong enough for signal reflection.

More importantly, a backscatter device is able to reflect multiple wireless signals at the same time because of its simplicity on analog RF front end where a filter does not exist to limit the band where it can operate. As a result, a backscatter device is able to simultaneously reflect multiple incident signals as long as these signals can resonate with the backscatter antenna. For example, a backscatter device that has a 2.4GHz antenna can simultaneously reflect both WiFi and BLE signals while modulating the same information.

In this chapter, we thoroughly explore the design space of leveraging multiple existing wireless signals for carrying backscattered information. Our investigation looks at the tradeoff between power consumption, backscatter throughput, and op-

erational distance when multiple existing wireless signals are simultaneously used for backscatter.

In order to leverage multiple ambient signals for backscatter, we need to address four challenges. First, we have to deal with the strong interference introduced by the incident wireless signal, which is usually at least 30dB stronger than the backscattered signal. To make matters worse, we want to reduce interference on commercial radios without any hardware modification. Second, combining backscattered information carried by different types of wireless signals is hard because analog signals samples are not provided by commercial radios. As a result, joint decoding cannot be done on the physical layer where the maximum amount of backscattered information is preserved. Third, backscatter data rate is limited by the rate of packets sent by commercial radio because previous research, such as WiFi Backscatter, leverage packet level information for decoding backscattered bits. Fourth, we have to maintain the power consumption of a backscatter device low.

In order to address these challenges, we design and implement a system that can leverage multiple existing wireless signals for backscatter. Our system includes four core modules: interference isolator, joint decoder, data rate accelerator, and low power tag. Our interference isolator targets at reducing self-interference without modifying commercial radio hardware. The key technique leveraged by our isolator is FSK modulation that can allocate the spectrum of the incident signal and backscattered signal into two separated bands. Then, our joint decoder exploits multiple commercial receivers where each targets at decoding one incident signal. Our joint decoder takes the bits decoded by each receiver and chooses the one with the highest SNR as the final decoded bit. To improve the data rate, we design a data rate accelerator that uses a commercial bare bone radio to detect backscattered signal within a packet of the incident wireless signal. Such intra-packet detection can break the data rate limitation imposed by the packet rate of the incident wireless signal because of its

ability to detect physical layer signals. The last component of our system is a low power tag where we examine and identify the end-to-end power consumption model of a backscatter device and design an oscillator switching system to achieve proportional power consumption as the backscatter frequency increases.

We implement our system on an FPGA and demonstrate that our system can leverage two types of existing wireless signals (WiFi and BLE) on 2.4GHz for backscatter. Our empirical evaluation shows that our system can achieve 350bps throughput and 2m operational distance when reflecting a WiFi signal. Similar results are observed when the backscatter device reflects a BLE signal where 3bps throughput and 2m operational distance are achieved.

5.2 Motivation

Some previous research, such as Ambient backscatter and WiFi backscatter, leverage existing wireless signal (TV and WiFi) for carrying backscattered information. In both Ambient backscatter and WiFi backscatter, a backscatter device tunes and detunes its antenna by toggling an RF transistor. Such tuning can change the strength of the reflected TV and WiFi signals and as a result, change the strength of the received signal, which is a combination of the primary TV and WiFi signal and the reflected version. Once a receiver observes this change, it can simply use a calibrated threshold to decode the backscattered bits. However, such design cannot support long range and robust backscatter on these commercial radios because of three reasons. First, both Ambient backscatter and WiFi backscatter experience strong self-interference. As a result, their operational range is small and achieved data rate is low. Figure 5.1 shows an empirically measured SNR and SINR of WiFi backscatter across distance. Similar to the deployment of [34], we put a 0dBm transmitter 3m away from a backscatter device. Then, we move a receiver away from the backscatter device and measure the TX signal strength as well as the backscatter signal strength. Even when the receiver

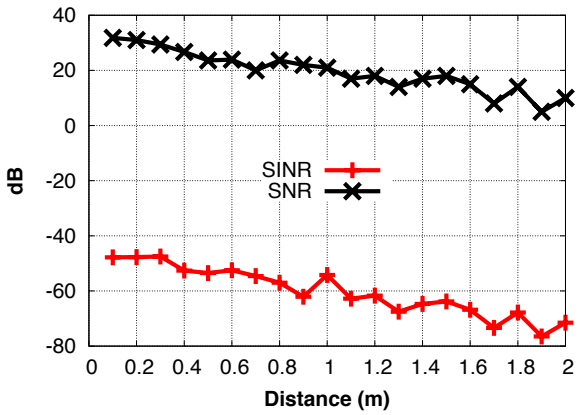


Figure 5.1. SNR and SINR of backscatter across distance.

is 0.1m from the backscatter device, the measured SINR is -47dB, which suggests that the transmitted signal strength is 47dB higher than the backscattered signal strength. When the receiver moves further, the SINR decreases. The SINR at 2m decreases to -71dB, which makes backscatter decoding extremely challenging. Because of the severe interference from the WiFi transmitter, WiFi backscatter only achieves slow backscatter rate and can only operate at a close distance.

We ask what is the backscatter SNR if we move the backscattered signal into another band such that it does not overlap with the spectrum occupied by the incident WiFi signals? We measure the SNR of backscattered signal in the same setting where the measured SNR is much higher than the SINR. For example, the SNR at 0.1m is 31.8dB, much higher than the -47dB SINR at the same distance. Therefore, the performance of backscatter will become much better if we can move the backscattered signal into another band such that it does not share the same spectrum with the incident signal.

Second, the decoding of both Ambient backscatter and WiFi backscatter is not robust on mobile devices because mobility itself also changes the propagation characteristics of the incident signal. As a result, false positives of decoding increase. Figure 5.2 shows the CDF of the received signal strength of a WiFi when a trans-

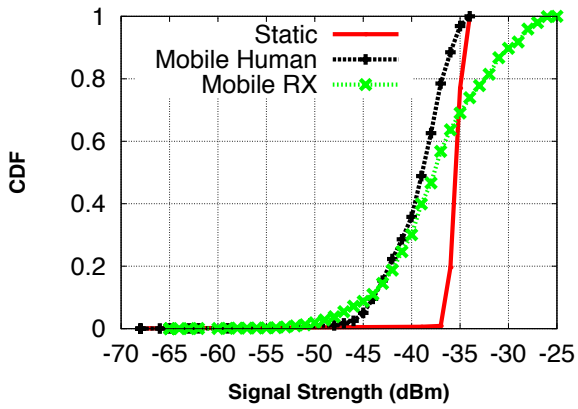


Figure 5.2. Received signal strength of a WiFi.

mitter is 1m away from a receiver. When the receiver is static and nobody moves around, we can observe a stable WiFi signal with a median strength of -35dBm. However, when a person moves around or the receiver itself moves, the received signal strength changes significantly. When a person moves around, the received signal strength varies from -45dBm to -35dBm (90% confidence interval). When the WiFi receiver itself moves, the signal strength varies from -47dBm to -28dBm (90% confidence interval). Such significant variation will introduce significant decoding errors for Ambient backscatter and WiFi backscatter systems if the pre-calibrated threshold is not adapted accordingly.

Third, leveraging a single wireless signal source for reflection cannot provide robust backscatter. The signal strength of the single source can be low such that decoding backscatter is not feasible. To make matters worse, traffic of a single radio is usually bursty, which means that the incident signal is not always available for backscatter and the gap between incident signals is not consistent. Figure 5.3 shows the CDF of the interval between packets on three WiFi channels. We choose the first, sixth, and eleventh channels because the three WiFi channels are the most frequently used channels in US. We observe 1∼81ms gap between packets where backscatter cannot

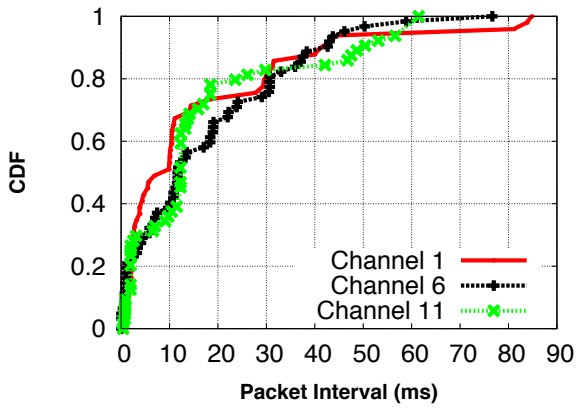


Figure 5.3. Interval between WiFi packets.

operate on. In addition, the gap between packets varies significantly, which makes it hard for a backscatter device to know when it should transfer.

5.3 Design

Our design answers the following question — how should we leverage multiple commercial radios on existing mobile devices for carrying and decoding backscattered information? We start with an overview of our system. Figure 5.4 shows our system architecture. We first determine which ambient signal, either WiFi, Bluetooth, or another ambient signal, is available for reflection. This decision is made by turning on all commercial receivers where each reports the presence of a specific type of incident signal. Then, the reader sends a command to the backscatter sensor to tune its FSK frequencies such that the backscattered signal is shifted into adjacent channels and does not overlap with the incident signal. Once shifted, we start decoding by turning on the decoder on each module. For example, if the incident signal is a mix of WiFi and Bluetooth, both WiFi and Bluetooth decoder start operation. The joint decoder takes the intermediate decoding results output by each module and selects the one with the highest SNR. We describe these components in more details in the following sections.

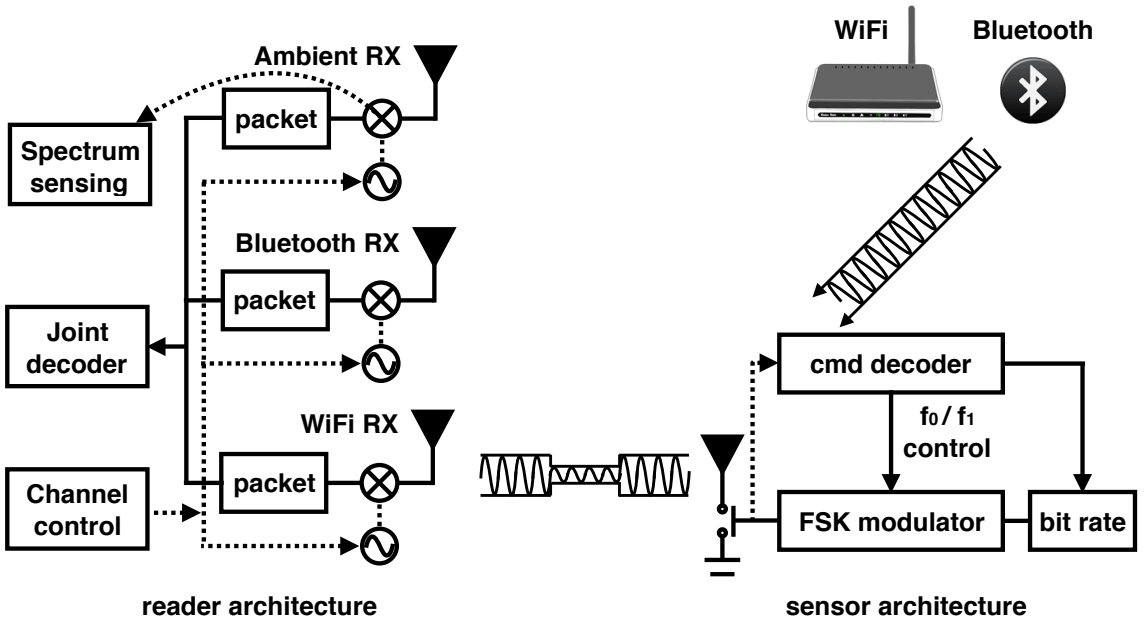


Figure 5.4. System architecture.

5.3.1 Isolator — reduce self-interference

As we discussed in section 5.2, one challenge of leveraging existing wireless signals for backscatter is how to deal with the self-interference introduced by the wireless signals themselves. We propose to separate the incident signal and backscattered signal spectrum allocation for reducing self-interference. Such isolated spectrum allocation benefits the self-interference reduction because of two reasons. First, the analog to digital converter (ADC) itself can reject part of the interference from adjacent channels. For example, the WL1837MOD WiFi chip can achieve up to 42dB rejection for interference coming from adjacent channels. As long as the backscattered signals do not share the same spectrum as the incident signal, the process of analog to digital conversion itself benefits the self-interference reduction. Second, once the incident signal and backscattered signal do not share the same spectrum, digital signal processing, such as filtering, can be used to reduce further self-interference even though we do not have an accurate estimation of the structure of the incident signal. But all

of these discussions assume that we can separately allocate spectrum for the incident and backscattered signals. How do we do this?

Our isolator exploits Frequency Shift Key (FSK) modulation where an FSK modulated backscattered signal can be far from the incident signal. We do not choose Amplitude Shift Key (ASK) and Phase Shift Key (PSK) modulations used by many backscatter systems because ASK and PSK modulated backscatter baseband needs to center around the incident signal. To understand this, we look at the mathematical formulation of an FSK modulated backscattered signal shown in equation 5.1. S_0 and S_1 are square waves with different frequencies that are used for toggling the RF transistor, and S is the data transmitted. When a device transmits data one ($S = 1$), the device toggles the RF transistor at frequency f_1. A square wave S_1 is modulated on the incident signal S_i. In contrast, a square wave S_0 with frequency f_0 is modulated on the incident signal when the device transmits data zero ($S = 0$). By tuning the frequency f_0 and f_1, we can move the baseband spectrum away from the incident signal. For example, when we use $f_0 = 1$MHz and $f_1 = 1.5$MHz, the baseband spectrum is 1MHz away from the center of the incident signal. As a result, we obtain the flexibility to allocate each spectrum separately.

$$\begin{aligned} S_b &= S_0\overline{S} + S_1 S \\ S_i S_b &= S_i(S_0\overline{S} + S_1 S) \end{aligned} \tag{5.1}$$

Once shifted, a receiver is tuned to the targeted frequencies for detecting transmitted bits. For example, when a shifted incident signal is detected at channel $f_i \pm f_0$, data zero ($S = 0$) is transmitted. Similarly, when a signal is detected at channel $f_i \pm f_1$, data one ($S = 1$) is transmitted. One advantage of such frequency shift is that signal detection can be done by existing wireless radios. For example, when the incident signal is a WiFi signal on the first channel (2412MHz), we can exploit 20MHz FSK to move the backscattered signal to 2432MHz and configure the WiFi

receiver on the fifth channel (2432MHz) for detecting the presence of the reflected signal. As a result, existing commercial radios can be used for decoding backscattered information.

5.3.2 Joint decoder — combine reflection from multiple sources

Having discussed how should we reflect an existing wireless signal, let us now turn to look at why we want to leverage multiple sources of ambient signals for reflection. There are two reasons. First, as shown in section 5.2, the incident signal from a single ambient source is not always available for reflection. Therefore, a backscatter device cannot transmit whenever it wants to. Second, the interval between transmissions varies a lot even for a single signal source. A backscatter device cannot easily decide when the incident signal is available for reflection. Therefore, we can improve the robustness of backscatter by leveraging multiple sources of ambient signals such that the gaps between incident signals become smaller. But can a backscatter device reflect multiple sources of ambient signals at the same time?

Our key observation is that the simplicity of the analog RF front end of a backscatter device enables it to reflect multiple sources of signals at the same time. The core components of the analog RF front end only include an RF transistor and an antenna. No filter exists to limit the band where a backscatter radio can operate. Therefore, a backscatter device can reflect multiple incident signals at the same time as long as these signals can resonant with the backscatter antenna. Since both WiFi and Bluetooth share the same 2.4-2.483GHz spectrum, a backscatter device can reflect both at the same time if the incident signal is a mix of both.

Upon receiving the reflection of multiple incident signals, we need to combine information carried by each incident signal for optimum decoding. One optimum decoder is maximal ratio combining that takes the physical layer analog samples of each reflected incident signal for determining an optimum decoding. Since a single

analog sample is not robust enough for determining a transmitted bit, multiple diverse analog samples are combined based on a calculated weight for optimum decoding. [18] leverages this technique and achieves robust backscatter. However, maximal ratio combining cannot be implemented across several commercial wireless radios because of the lack of physical layer hooks. Therefore, we have to combine information on the bits level instead of on the physical layer.

Our decoder is built on top of each commercial receiver. We first decode the backscattered bits on each incident signal. Such decoding is done by turning on each receiver. For example, when the incident signal is a mixer of WiFi and Bluetooth, we turn on the receivers of both WiFi and Bluetooth, and obtain decoded bits and associated received signal strength indicator (RSSI) from each receiver. Then, we simply compare the RSSI given by each receiver and choose the bit with the highest RSSI.

5.3.3 Accelerator — improve backscatter data rate

When we use commercial existing radios for decoding backscattered information, we meet a severe limitation — low backscatter data rate. We observe that the backscatter data rate cannot be higher than the packet rate transmitted by commercial radios, such as WiFi and Bluetooth. Such limitation comes from the fact that commercial radios output the decoding results on the packet level rather than the bit level. In other words, commercial radio can inform a user that it receives a packet. However, it usually does not expose the interface that tells you that a single bit is received or not. Such bit level information needs to be obtained on the physical layer that is not exposed by most commercial radios. As a result, the maximum backscatter data rate is limited by the maximum packet rate on commercial radios. We ask is it possible to decode backscatter information when part of a packet carries backscatter data zero while another part carries data one?

We propose the use of commercial bare bone radios, such as CC2500, for decoding when backscatter data rate is higher than the packet rate of incident signals. These bare bone radios do not need any protocol-level information of incident signals. They just check whether a specific modulation is used on a specific band. Since we move the backscattered signal into a clean band that does not have overlap with the incident signal, instead of detecting the reflected packets, such bare bone radio can be used for detecting whether a backscattered signal exists or not. By augmenting existing commercial radios with such bare bone module, we can significantly improve the backscatter data rate.

5.3.4 Low power tag design

When moving the backscattered signal to adjacent channels, we need to look at the power consumed for generating the backscattered signal. Intuitively, more power will be consumed when we operate the FSK signal with higher intermediate frequencies, which means that f_0 and f_1 have larger values. We look at three subsystems on backscatter devices — RF transistor, transmission logic, and clock generator, which consume most of the power of the whole system. Let us now investigate the power consumed by each subsystem individually.

The RF transistor is a MOSFET transistor with a capacitance around 1pF. Its power consumption can be calculated using the equation $\frac{1}{2}CV^2F$ where C is the capacitance of the transistor, V is the digital drain-source voltage, and F is the frequency of operating the transistor. Even when toggled at a high rate of 20MHz, the RF transistor only consumes 10μW. Thus, the power consumption of the RF transistor itself is low and has a linear relationship with F.

The second subsystem, transmission logic, is a hardware module that toggles the backscatter RF transistor based on data transmitted. When we use digital logic, such as FPGA, to implement the transmission logic, the dynamic power consumption

Table 5.1. Power consumed by oscillators operating at different frequencies and different accuracies.

Oscillators	Frequency	Accuracpy	Power
ASH7K	32 kHz	±10ppm	1.48μW
LTC6990	1 MHz	±50ppm	326μW
LTC6900	10 MHz	±40ppm	2.04mW

increases linearly with the rate of transmission [58]. The exact power budget depends on the transmission logic used for controlling the RF transistor.

The last subsystem is the clock generator that provides clocks for timing the whole system. Oscillators are typical sources for generating clocks. Table 5.1 shows the power consumed by several oscillators operating at different frequencies and accuracies. We choose oscillators that have the smallest power consumption that we can find for a particular frequency. When the frequency of oscillators enters the mega hertz regime, the power consumption increases significantly. To make matters worse, the increase in power consumption does not have a linear relationship as the increase of oscillating frequencies. For example, LTC6990 consumes 220× more power compared to ASH7K while the oscillating frequency only increases by 30×. Unfortunately, our backscatter device needs to generate FSK with different frequencies. So a key question that we have to address is how to design a clock generator subsystem that can have proportional power consumption as the frequency it provides?

Our design leverages multiple oscillators that oscillate at different frequencies, and switches among them to achieve proportional power consumption. When we need to generate low-frequency FSK signals, we turn on the low-frequency oscillator by providing current to this oscillator and turn off the rest by stop feeding current to them. A similar strategy is used when we need high-frequency FSK signals. We stop feeding current to the unused oscillators because of we do not want to waste power on unused oscillators. We do not choose the design of leveraging a single high-speed oscillator as the main clock, such as 20MHz, and dividing the main clock because

of two reasons. First, the high-speed main clock itself is always on even when we need low-frequency FSK signals, and naturally consumes lots of power. Second, the clock division circuit also consumes lots of power because it is always driven by a high-speed clock.

When we switch among multiple oscillators, we have to address the problem that we only have limited number of frequencies for operation. For example, given the three oscillators shown in Table 5.1, we are only able to generate FSK signals with frequencies 32kHz, 1MHz, and 10MHz. However, we cannot directly generate FSK signals at other frequencies, such as 3MHz. We address this problem by leveraging harmonics of the backscatter baseband.

5.3.5 Leverage baseband harmonics

In order to use the band that is not covered by the FSK signals generated by the limited number of oscillators, we leverage harmonics, which carry the same information as the baseband signal. Let us look at the mathematical formulation of FSK modulated backscatter as shown in equation 5.1. Since S_0 and S_1 are square waves, we can observe harmonics as shown in equation 5.2 where a square wave contains a series of frequency components. We notice that the same baseband information is carried by different orders of harmonics. For example, we can observe the frequency component of f_1 as well as $(2n-1)f_1, n > 1$ when the device transmits data one. Therefore, instead of checking the presence of f_0 and f_1 for decoding the FSK signal, we can check harmonics $(2n-1)f_0$ and $(2n-1)f_1$.

$$
\begin{aligned}
S_0 &= \frac{4}{\pi}\sum_{n=1}^{\infty}\frac{\sin(2\pi(2n-1)f_0)}{2n-1} \\
S_1 &= \frac{4}{\pi}\sum_{n=1}^{\infty}\frac{\sin(2\pi(2n-1)f_1)}{2n-1}
\end{aligned}
\tag{5.2}
$$

Power benefit of leveraging harmonics: In addition to reaching the uncovered band, another benefit of leveraging harmonics for embedding information comes from

the reduced power consumption for generating the baseband signal. When a baseband signal is allocated to a spectrum that is f Hz far from the carrier, we can allocate the nth harmonics to this band by setting $(2n-1)f_0 = f$. Therefore, a device only needs to toggle the RF transistor at a frequency of $f_0 = \frac{f}{2n-1}$, which significantly reduces the power needed for generating the baseband square wave S_0. A similar conclusion can be drawn for generating square wave S_1. However, such power benefit comes at the cost of reduced SNR.

Tradeoff between SNR and orders of harmonics: We now turn to look at the tradeoff between SNR versus orders of harmonics. As shown in equation 5.2, the signal strength decreases as the order of harmonics increases. For example, the amplitude of the nth order harmonic is only $\frac{1}{2n-1}$ fraction of the baseband signal. The signal strength degradation of harmonics can be formulated using equation 5.3 where the first order of harmonics (n=2) is 9.5dB lower than the baseband and the second order of harmonics (n=3) is 14dB lower than the baseband. Figure 5.5 shows the empirically measured backscatter baseband and harmonics signal strength. We observe a -72dBm baseband, -81.3dBm first order harmonic and -85.3dBm second order harmonic. Both first and second orders of harmonics match our theoretical formulation and suggest the decreasing signal strength when the order of harmonics increases. Therefore, we have to choose an order of harmonics that can guarantee successful decoding while consuming the smallest amount of power.

$$P_n = P_0 - 20\log(2n-1) \tag{5.3}$$

5.4 Implementation

In this section, we describe key implementation details not covered in earlier sections.

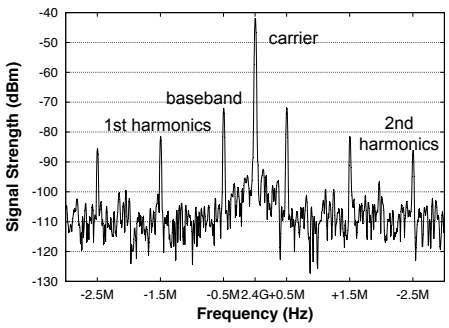


Figure 5.5. Signal strength of baseband and harmonics.

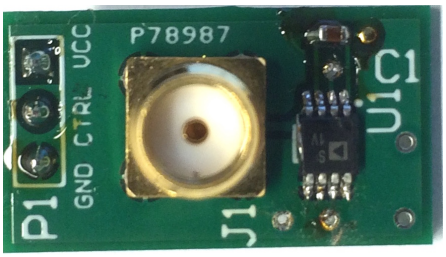


Figure 5.6. Backscatter radio analog front end.

Backscatter radio analog front end: Figure 5.6 shows a prototype of our backscatter radio analog front end. We use an ADG901 transistor to tune and detune the antenna. The antenna is connected to the transistor via an SMA connector. Therefore, we can directly connect to antennas of different frequencies. For example, we connect to a VERT2450 2.4GHz antenna for reflecting 2.4GHz existing wireless signals in our implementation. Such design is more flexible compared to the PCB dipole

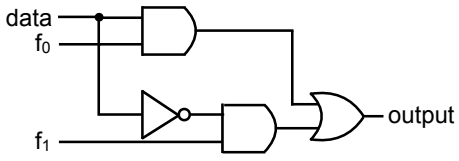


Figure 5.7. FSK transmitter logic gates.

antenna used in WISP. However, we confess that its backscattered signal strength is slightly worse than WISP because of the lack of well tuned matching circuits.

Hardware controller: Our hardware controller is a Igloo Nano AGLN250 FPGA, which only consumes $\sim$10μW of power in sleep mode. Since we do not run complicated logics on the controller, it is clocked by a 32kHz slow oscillator. We implement the oscillator switching logics on the FPGA which determines the intermediate frequencies f_0 and f_1 of the FSK signal. Once f_0 and f_1 are chosen, we use the combinational logic shown in Figure 5.7 to take f_0 and f_1 as input, and generate modulated FSK baseband. The output of the combinational logic is used to toggle the RF transistor.

FSK decoder: Our FSK decoder is implemented on commercial WiFi (CC3200) and BLE (CC2650) chips. When we observe an incident WiFi signal on the ith channel and a BLE signal on the jth channel, we configure CC3200 and CC2650 to detect packets on the $i + n$th and $j + n$th channels. Packets detected by each radio are reported to the joint decoder for deciding the actual bit transmitted by the backscatter device. Both CC3200 and CC2650 have similar sensitivity (-95dBm) of detecting backscattered packets.

5.5 Evaluation

Let us now turn to look the performance of our system when ambient WiFi and BLE signals are leveraged for backscatter. Figure 5.8 shows the throughput achieved by our system when a backscatter device reflects a BLE signal generated by a CC2650 chip. We place the backscatter device 20cm away from the BLE transmitter and move the BLE receiver away across distance. The BLE receiver detects the reflected packets on neighboring channels for determining the bits transmitted by a backscatter device. When we use FSK modulation to move the backscattered BLE signal to an adjacent channel that is 2MHz away, we are able to achieve 2.2m operational distance with 3bps throughput. When the shifted frequency is 20MHz, we are able to achieve

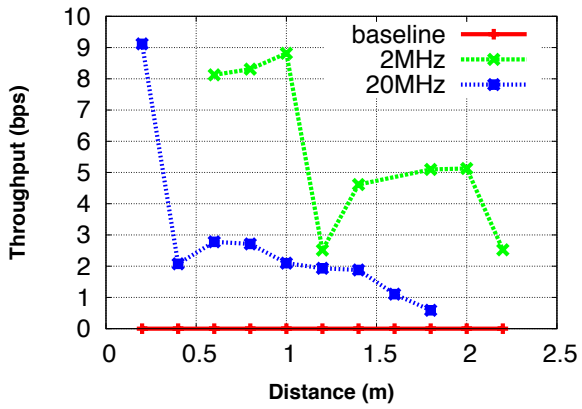


Figure 5.8. Backscatter throughput achieved when reflecting a BLE signal.

1.8m operational distance with a throughput of 0.7bps. Both schemes have better performance compared to our baseline where only 0bps throughput is achieved across distance. Our baseline does not change the frequency of the backscattered signal. Instead, it looks at how a backscattered signal changes the strength of the received signal. Unfortunately, the backscattered signal in our experiment is too weak to change the RSSI detected by the BLE receiver. As a result, 0bps data rate is achieved across distance.

We also evaluate the performance of our system when reflecting a WiFi signal. We use a CC3200 chip to transmit the incident WiFi signal on the 9th channel. Then we configure the FSK intermediate frequency as 20MHz to move the reflected WiFi signal to the 13th channel. Figure 5.9 shows the performance of backscattering WiFi signal when we move the WiFi receiver away from the backscatter device. We are able to achieve 350bps data rate when the receiver is 2m away. Our performance is much better than the baseline where only 0bps data rate is achieved. The baseline relies on RSSI for detecting the bits transmitted by a backscatter as well. However, even when the backscatter device toggles its RF transistor at 10Hz, we cannot observe any change on RSSI. As a result, 0bps data rate is observed.

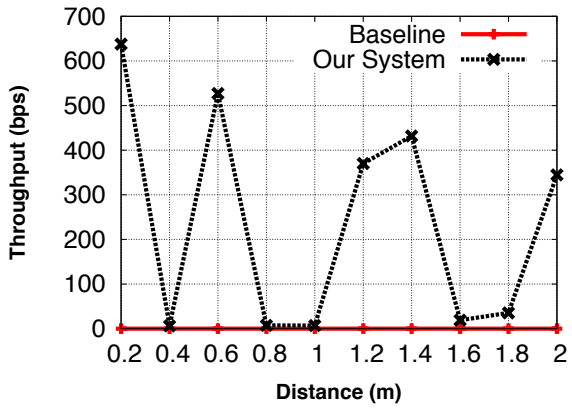


Figure 5.9. Backscatter throughput achieved when reflecting a WiFi signal.

5.6 Related Work

Our work is inspired by recent progress on enabling backscatter on top of existing signal. WiFi backscatter [34] is a system that allows carrying backscattered information on top of existing WiFi streams. Its key idea is that backscattered signal can change the channel state information (CSI) and received signal strength indicator (RSSI) observed by a WiFi receiver. By leveraging CSI and RSSI, WiFi backscatter enables the decoding of backscattered data on a WiFi receiver. Such design leverages the existing WiFi systems for backscatter. However, the SINR experienced by backscatter is low because the backscattered signal shares the same spectrum as the WiFi signal.

BackFi [18] leverages similar ideas and reduces the WiFi interference by canceling the WiFi signal transmitted by a WiFi device itself. As a result, the WiFi device only observes the backscattered signal. This mechanism significantly improves the SINR of backscatter and enables backscatter to operate at longer distance and with higher data rate. However, BackFi requires both WiFi protocol and hardware modification on a WiFi device. Both are not preferred in practical deployment.

In addition to leveraging WiFi signals, [38] looks at the opportunity of exploiting TV signal for backscatter between backscatter devices. [24] proposes a system that

is able to modulate a BLE baseband signal on top of an incident carrier signal. As a result, a BLE receiver observes a backscattered signal that has the same physical layer signaling and upper layer data format as a BLE signal. Commercial BLE receivers can be used for decoding such pseudo BLE signal. Our work differs from these work in terms of exploiting the diversity of incident signals for improving the robustness of backscatter.

Our work also differs from other recent work that target at enabling the practical deployment of backscatter. [40] proposes hardware innovation as well as coding algorithms for achieving long range backscatter. [31] [32] propose algorithms that enables concurrent asynchronous transmission from multiple backscatter devices. [54] proposes a network stack that enables the operation of backscatter devices when harvested energy is extremely small. As far as we know, we are the first work that discusses the opportunity of leveraging multiple incident signals for carrying backscattered information.

5.7 Conclusion

In summary, we look at how a backscatter device should leverage multiple ambient wireless signals for carrying backscattered information. In order to answer this question, we address several challenges including strong self-interference on commercial receivers, joint decoding without physical layer information, slow data rate, and low power backscatter device design. We design and implement a system that is able to exploits ambient WiFi and BLE signal for backscatter. Our empirical evaluation shows that we are able to achieve 350bps and 3bps data rate when reflecting a WiFi and BLE signal respectively at a distance of 2m.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Thesis Summary

This thesis explores fundamental factors that limit the range, power reduction, and deployability of backscatter systems. I have employed a set of techniques, including hardware design, wireless communication, and operating systems, to significantly improve the performance of backscatter systems.

We propose a hardware sensing architecture that minimizes computational blocks between the sensors and the backscatter RF interface. Its design is inspired by studying a variety of computational blocks between sensors and backscatter RF interface. We find that these overheads were negligible on platforms where communication was expensive. However, because of the ultra-low power consumption of backscatter radios, they become the bottleneck on backscatter-based systems and increase power consumption while limiting throughput. Therefore, we overturn the design principle governing wireless sensor design from one that is focused on minimizing communication to one focused on optimizing the computational elements between the sensor and RF interface. FPGA instantiation demonstrates $\geq$1Mbps backscatter transmission while only consuming $\leq 100\mu$W of power, two orders of magnitude improvement over the state of the art.

We propose a network stack that fragments any network task into its smallest atomic units to enable the system to scale down to resource impoverished regimes. Its design is inspired by the observation that communication tasks executed by micro-powered sensors are simply too large to fit into the extreme energy constraints of

this regime. For example, the core primitive of a network stack — packet transfer — can involve hundreds of instructions and bits. A packet transfer might not be successfully executed simply because it is too large compared to the extreme energy constraints of resource impoverished regimes. Therefore, we employ a simple but powerful abstraction — by fragmenting any network task into its smallest atomic units, we can enable the system to operate in resource impoverished regimes. The instantiation enables packet transfer when the whole system is powered by a 3cm×3cm solar panel under natural indoor light condition.

In the last part, we look at how to deploy backscatter systems on mobile and wearable devices. Our key idea is leveraging multiple existing wireless signals for carrying backscattered information. To achieve the goal, we have to deal with several challenges, such as strong self-interference, joint decoding on bits level, slow data rate, and low power consumption. Our design follows two rules. First, we leverage existing wireless signals for backscatter since we do not need to deploy additional backscatter readers. Second, we use commercial radio receiver for decoding backscattered information because these radio already exist on mobile and wearables. Our empirical evaluation with an FPGA controlled backscatter radio, TI CC3200 WiFi chip, and TI CC2560 BLE chip shows that our system is able to achieve 350bps throughput and 2m operational distance when reflecting a WiFi signal. Similar results can be observed when reflecting a bluetooth signal where we achieve 3bps throughput and 2m operational distance.

6.2 Future Work

I would like to continue to explore research problems related to wearable and mobile systems. Below I describe three research directions I will pursue in the near future.

Applications enabled by backscatter: Backscatter has the potential to enable low-power video gaming on mobile devices. Video game applications usually involve complex graphics processing, including object segmentation and 3D rendering, and thereby require significant amount of computational resources and energy. For example, running the "Need for Speed" game on an iPhone consumes 90% of its CPU. To make matters worse, the iPhone cannot last for more than 2 hours when a user continuously plays the game. With the aid of ultra low-power backscatter radios, many graphics processing tasks can be offloaded to a base station which is connected to the cloud. Once graphics processing is done at the cloud, the computed results will be pushed back to mobile devices via backscatter. Because a mobile device only acts as a display for cloud-computed results, the proposed scheme will significantly reduce the energy and computational resources needed for running video gaming. There are several research challenges I will address to enable this application, including omni directional backscatter, partitioning tasks between mobile devices and cloud, and providing software and hardware abstractions for ease of application development.

Passive sensing via backscatter: Backscatter also has the potential to enable passive gesture identification and human activity recognition. To identify gestures and recognize activities, several recent research work [42] [14] exploit the key observation that any motion changes the propagation characteristics of backscattered signal. For example, a "wave" gesture will generate a doppler frequency shift on the received signal of a WiFi AP. [42] detects the doppler shift via physical layer signal processing, and uses it as the indicator of the presence of a gesture. However, detecting gestures presented by fingers via backscatter is challenging. Because of the small movement of fingers, doppler shift introduced is tiny and hard to detect. Instead, I am going to explore frequency domain features to identify fine-grained finger gestures. Specially, I am planning to use Frequency Modulated Continuous Wave (FMCW) radar. The intuition is that tiny finger movement will introduce large frequency offset on the

backscatter signal received by a FMCW radar. Therefore, fine-grained finger gesture identification is possible. Realizing this capability requires research efforts from multiple perspectives, including customizing FMCW systems on a commercial WiFi AP, distinguishing finger gestures versus multi path interferences, etc.

Mobile health: Backscatter has the potential to enable ubiquitous and non-obtrusive health monitoring, which is hard to achieve with existing wearable and implantable devices. Let us look at a specific mobile health application — hearing aids, which grants hearing to people who otherwise would be unable to do so. One type of hearing aids is cochlear implants. However, existing cochlear implants require a disk-shaped transmitter about an inch in diameter, with a wire snaking down to a joint microphone and power source around the patient's ear. We think that backscatter can help significantly shrink down the form factor of cochlear implants. It provides RF energy for operating the whole cochlear system. In addition, the measured electrical signal can be offloaded to a base station via ultra low-power backscatter. By eliminating the battery, the whole cochlear implant has a smaller form factor and can be encapsulated into the middle ear. In addition to cochlear implants, many other mobile health applications will benefit from backscatter as well. For example, an RF-powered glucose monitoring system can be embedded in a contact lens.

BIBLIOGRAPHY

[1] 33 Billion Internet Devices By 2020: Four Connected Devices For Every Person In World. www.strategyanalytics.com.

[2] Advanced Self-Powered Systems of Integrated Sensors and Technologies. http://assist.ncsu.edu.

[3] Analog devices adxl362 accelerometer sensor. http://www.analog.com/static/imported-files/data_sheets/ADXL362.pdf.

[4] Analog devices mems microphone admp803. http://www.analog.com/static/imported-files/data_sheets/ADMP803.pdf.

[5] The cbvs ecg-on-chip. http://www.clearbridgevitalsigns.com/chip.html.

[6] Five Challenges For The Internet of Things Ecosystem. http://www.forbes.com/sites/rakeshsharma/2013/11/12/five-challenges-for-the-the-internet-of-things-ecosystem/.

[7] Gradient descent. http://en.wikipedia.org/wiki/Gradient_descent.

[8] Libero ide. http://www.microsemi.com/products/fpga-soc/design-resources/design-software/libero-ide.

[9] Mean opinion score. http://en.wikipedia.org/wiki/Mean_opinion_score.

[10] Source code of QuarkNet. https://github.com/pengyuzhang/QuarkNet.

[11] The Impinj UHF Gen 2 Speedway RFID Reader. http://www.impinj.com.

[12] ThingMagic Bistatic Antenna. http://buyrfid.com/.

[13] WISP: Wireless Identification and Sensing Platform. http://seattle.intel-research.net/wisp/.

[14] Adib, Fadel, and Katabi, Dina. *See through walls with WiFi!*, vol. 43. ACM, 2013.

[15] Angerer, C., Langwieser, R., and Rupp, M. Rfid reader receivers for physical layer collision recovery. *IEEE Transactions on Communications* (2010).

[16] Balasubramanian, Niranjan, Balasubramanian, Aruna, and Venkataramani, Arun. Energy consumption in mobile phones: a measurement study and implications for network applications. In *ACM SIGCOMM IMC 2009*.

[17] Bandyopadhyay, S., and Chandrakasan, A.P. Platform architecture for solar, thermal and vibration energy combining with mppt and single inductor. In *VLSI Circuits (VLSIC), 2011 Symposium on* (2011), IEEE, pp. 238–239.

[18] Bharadia, Dinesh, Joshi, Kiran Raj, Kotaru, Manikanta, and Katti, Sachin. Backfi: High throughput wifi backscatter. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication* (2015), ACM, pp. 283–296.

[19] Buettner, M., and Wetherall, D. An empirical study of uhf rfid performance. In *ACM MobiCom* (2008).

[20] Buettner, Michael, Greenstein, Ben, and Wetherall, David. Dewdrop: an energy-aware runtime for computational rfid. In *USENIX NSDI 2011*.

[21] Buettner, Michael, and Wetherall, David. A software radio-based uhf rfid reader for phy/mac experimentation. In *RFID (RFID), 2011 IEEE International Conference on* (2011), IEEE, pp. 134–141.

[22] De la Piedra, Antonio, Braeken, An, and Touhafi, Abdellah. Sensor systems based on fpgas and their applications: a survey. *Sensors 12*, 9 (2012), 12235–12264.

[23] Dutta, Prabal, and Culler, David. Epic: An open mote platform for application-driven design. In *IEEE IPSN 2008*.

[24] Ensworth, Joshua F, and Reynolds, Matthew S. Every smart phone is a backscatter reader: Modulated backscatter compatibility with bluetooth 4.0 low energy (ble) devices. In *RFID (RFID), 2015 IEEE International Conference on* (2015), IEEE, pp. 78–85.

[25] Gorlatova, Maria, Kinget, Peter, Kymissis, Ioannis, Rubenstein, Dan, Wang, Xiaodong, and Zussman, Gil. Challenge: ultra-low-power energy-harvesting active networked tags (enhants). In *Proceedings of the 15th annual international conference on Mobile computing and networking* (2009), ACM, pp. 253–260.

[26] Gummeson, Jeremy, Clark, Shane S, Fu, Kevin, and Ganesan, Deepak. On the limits of effective hybrid micro-energy harvesting on mobile crfid sensors. In *ACM MobiSys 2010*.

[27] Gummeson, Jeremy, Zhang, Pengyu, and Ganesan, Deepak. Flit: a bulk transmission protocol for rfid-scale sensors. In *ACM MobiSys 2012*.

[28] Hanson, Scott, Foo, ZhiYoong, Blaauw, David, and Sylvester, Dennis. A 0.5 v sub-microwatt cmos image sensor with pulse-width modulation read-out. In *JSSCC 2010*.

[29] Hinkelmann, Heiko, Reinhardt, Andreas, Varyani, Sameer, and Glesner, Manfred. A reconfigurable prototyping platform for smart sensor networks. In *Programmable Logic, 2008 4th Southern Conference on* (2008), IEEE, pp. 125–130.

[30] Holleman, Jeremy, Yeager, Dan, Prasad, Richa, Smith, Joshua R, and Otis, Brian. Neuralwisp: An energy-harvesting wireless neural interface with 1-m range. In *BioCAS'08* (2008), IEEE, pp. 37–40.

[31] Hu, Pan, Zhang, Pengyu, and Ganesan, Deepak. Leveraging interleaved signal edges for concurrent backscatter. In *Proceedings of the 1st ACM workshop on Hot topics in wireless* (2014), ACM, pp. 13–18.

[32] Hu, Pan, Zhang, Pengyu, and Ganesan, Deepak. Laissez-faire: Fully asymmetric backscatter communication. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication* (2015), ACM, pp. 255–267.

[33] Huang, Junxian, Qian, Feng, Gerber, Alexandre, Mao, Z Morley, Sen, Subhabrata, and Spatscheck, Oliver. A close examination of performance and power characteristics of 4g lte networks. In *ACM MobiSys 2012*.

[34] Kellogg, Bryce, Parks, Aaron, Gollakota, Shyamnath, Smith, Joshua R, and Wetherall, David. Wi-fi backscatter: internet connectivity for rf-powered devices. In *Proceedings of the 2014 ACM conference on SIGCOMM* (2014), ACM, pp. 607–618.

[35] Kellogg, Bryce, Talla, Vamsi, and Gollakota, Shyamnath. Bringing gesture recognition to all devices. In *USENIX NSDI 2014*.

[36] Law, C., Lee, K., and Siu, K.Y. Efficient memoryless protocol for tag identification. In *Proceedings of the 4th international workshop on Discrete algorithms and methods for mobile computing and communications* (2000), ACM, pp. 75–84.

[37] Lee, Y., Kim, G., Bang, S., Kim, Y., Lee, I., Dutta, P., Sylvester, D., and Blaauw, D. A modular $1mm^3$ die-stacked sensing platform with optical communication and multi-modal energy harvesting. In *ISSCC 2012*.

[38] Liu, Vincent, Parks, Aaron, Talla, Vamsi, Gollakota, Shyamnath, Wetherall, David, and Smith, Joshua R. Ambient backscatter: wireless communication out of thin air. In *ACM SIGCOMM 2013*.

[39] Namboodiri, V., and Gao, L. Energy-aware tag anticollision protocols for rfid systems. *Mobile Computing, IEEE Transactions on 9*, 1 (2010), 44–59.

[40] Parks, Aaron N, Liu, Angli, Gollakota, Shyamnath, and Smith, Joshua R. Turbocharging ambient backscatter communication. In *Proceedings of the 2014 ACM conference on SIGCOMM* (2014), ACM, pp. 619–630.

[41] Peris-Lopez, Pedro, Hernandez-Castro, Julio Cesar, Estevez-Tapiador, Juan M, and Ribagorda, Arturo. Lameda prng for epc class-1 generation-2 rfid specification. *Computer Standards & Interfaces 31*, 1 (2009), 88–97.

[42] Pu, Qifan, Gupta, Sidhant, Gollakota, Shyamnath, and Patel, Shwetak. Whole-home gesture recognition using wireless signals. In *Proceedings of the 19th annual international conference on Mobile computing & networking* (2013), ACM, pp. 27–38.

[43] Ransford, Benjamin, Sorber, Jacob, and Fu, Kevin. Mementos: System support for long-running computation on rfid-scale devices. *ACM SIGPLAN Notices 47*, 4 (2012), 159–170.

[44] Sundstrom, Kurt E, Cooper, Scott A, Sarajedini, Amir, Esterberg, Aanand, Humes, Todd E, and Diorio, Christopher J. Rfid reader systems detecting pilot tone, Jan. 8 2013. US Patent 8,350,665.

[45] Trotter, Matthew S, Griffin, Joshua D, and Durgin, Gregory D. Power-optimized waveforms for improving the range and reliability of rfid systems. In *RFID, 2009 IEEE International Conference on* (2009), IEEE, pp. 80–87.

[46] Vannucci, Giovanni, Bletsas, Aggelos, and Leigh, Darren. A software-defined radio system for backscatter sensor networks. *Wireless Communications, IEEE Transactions on 7*, 6 (2008), 2170–2179.

[47] Vogt, H. Efficient object identification with passive rfid tags. *Pervasive Computing* (2002), 98–113.

[48] Wang, Jue, Hassanieh, Haitham, Katabi, Dina, and Indyk, Piotr. Efficient and reliable low-power backscatter networks. In *ACM SIGCOMM 2012*.

[49] Yakovlev, A., Kim, S., and Poon, A. Implantable biomedical devices: wireless powering and communication. *Communications Magazine, IEEE 50*, 4 (2012), 152–159.

[50] Yeager, Daniel J, Powledge, Pauline S, Prasad, Richa, Wetherall, David, and Smith, Joshua R. Wirelessly-charged uhf tags for sensor data collection. In *RFID, 2008 IEEE International Conference on* (2008), IEEE, pp. 320–327.

[51] Yeager, D.J., Holleman, J., Prasad, R., Smith, J.R., and Otis, B.P. Neuralwisp: A wirelessly powered neural interface with 1-m range. *Biomedical Circuits and Systems, IEEE Transactions on 3*, 6 (2009), 379–387.

[52] Zanetti, D., et al. Physical-layer identification of uhf rfid tags. In *ACM MobiCom* (2010).

[53] Zhang, H., Gummeson, J., Ransford, B., and Fu, K. Moo: A batteryless computational rfid and sensing platform. Tech. rep., Tech. Rep. UM-CS-2011-020, 2011.

[54] Zhang, Pengyu, and Ganesan, Deepak. Enabling bit-by-bit backscatter communication in severe energy harvesting environments. In *NSDI 2014*.

[55] Zhang, Pengyu, and Ganesan, Deepak. Enabling bit-by-bit backscatter communication in severe energy harvesting environments. Tech. rep., School of Computer Science, UMass Amherst. URL: `http://cs.umass.edu/~pyzhang/papers/QuarkNet-tech-report.pdf`, 2014.

[56] Zhang, Pengyu, Ganesan, Deepak, and Lu, Boyan. Quarkos: Pushing the operating limits of micro-powered sensors. In *Proceedings of the 14th USENIX conference on Hot Topics in Operating Systems* (2013), USENIX Association, pp. 7–7.

[57] Zhang, Pengyu, Gummeson, Jeremy, and Ganesan, Deepak. Blink: a high throughput link layer for backscatter communication. In *ACM MobiSys 2012*.

[58] Zhang, Pengyu, Hu, Pan, Pasikanti, Vijay, and Ganesan, Deepak. Ekhonet: high speed ultra low-power backscatter for next generation sensors. In *Proceedings of the 20th annual international conference on Mobile computing and networking* (2014), ACM, pp. 557–568.