Title
A Low-Power Mobile Sensing Architecture

Author
Dutta, Prabal

Publication Date
2009

Peer reviewed|Thesis/dissertation

A Low-Power Mobile Sensing Architecture

by

Prabal Kumar Dutta

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

in

Computer Science

in the

GRADUATE DIVISION
of the
UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in charge:
Professor David E. Culler, Chair
Professor Scott Shenker
Professor Ion Stoica
Professor Paul K. Wright

Fall 2009

The dissertation of Prabal Kumar Dutta is approved:

_____

Chair Date

_____

Date

_____

Date

_____

Date

University of California, Berkeley

Fall 2009

A Low-Power Mobile Sensing Architecture

Abstract

A Low-Power Mobile Sensing Architecture

by

Prabal Kumar Dutta
Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor David E. Culler, Chair

The system and network architecture for stationary sensornets is largely solved today with many commercial solutions now available and standardization efforts underway at the IEEE, IETF, ISA, and within many industry groups. However, the existing techniques for reliable, low-power communications in stationary sensornets fail on both counts when confronted with mobility. In this dissertation, we argue that awareness of real or potential mobility enables a solution that handles the mobile case well, and supports stationary networks as a special case. This dissertation addresses micropower mobiscopes, a nascent class of mobile sensornets – small, embedded, and battery-powered systems – that experience unpredictable but structured mobility and are severely energy-constrained. We show how awareness of mobility can simplify their communication challenges, enable low-power operation, and enhance the reliability of data delivery.

We introduce the MOV metric, a measure of mobility, and present techniques to gather it on a near nano-power budget. We also present iCount, a regulator-integrated energy meter design that allows nodes to introspect their own energy usage, and adapt their behavior to the actual energy availability and consumption. Integrating the pieces, we present three concrete hardware platforms that support our mobile sensing architecture. We develop a novel asynchronous neighbor discovery algorithm called Disco that allows nodes to operate their radios at very low duty cycles and yet still discover neighbors without any external synchronization information. Recognizing the necessity of beaconing in mobile networks, and the need for mobile-stationary node interactions, we design a link layer synchronization primitive, Backcast, and a receiver-initiated link layer, HotMac, that are suitable for mobile sensing, but also work for stationary networks across a range of conventional data collection workloads and a broad range of duty cycles.

We evaluate our thesis with three mobile sensing applications that embody our proposed architecture. The three applications – AutoWitness, SleepTrack, and CommonSense – are representative of asset tracking, health and fitness, and participatory urban sensing, and they each stress different aspects of the architecture, including motion detection, neighbor discovery, communications, interaction patterns, energy management, and data transport. These design points illustrate that our architecture is general enough to enable a range of applications but specific enough to support them well.

Professor David E. Culler
Dissertation Committee Chair

To Wendy,
whose boundless patience
is exceeded only by her love and generosity.

Contents

List of Figures

List of Tables

Acknowledgments

This dissertation would not have been possible without the guidance, support, and collaborations of a large cast of outstanding characters whose contributions I can only begin to acknowledge here. Please accept my sincerest apologies if I failed to remember your contribution. The failing is entirely mine.

David Culler was exactly the kind of advisor I needed. He always gave me enough rope to hang myself with, which I did on occasion, but he recognized that that was the only way to keep me motivated and productive. David's ability to identify the essence of a problem is remarkable and his ability to distill that essence into a memorable phrase is still more amazing. These often cryptic, but always insightful, "Cullerisms" are a hallmark of David's tutelage and I consider myself lucky to have understood a few of them. I'll of course pass these pearls of wisdom on to my students.

Scott Shenker became my co-advisor during my second year in graduate school. Although we met infrequently, he was always available via e-mail. Always. Perhaps meeting more often would have been futile for me since I could hardly absorb the decade's worth of truths revealed, ideas explored, and prose transformed that every interaction with Scott led to. Scott's advice on research direction, career options, and paper topics were always appreciated, if not always followed.

Ion Stoica advised me on much of the work described in this dissertation even though he was not officially a co-advisor. Ion's incredible attention to detail, focus on distilling things to their core, and total commitment to his students is rare. I am amazed that Ion found the time to reply to every e-mail I remember ever sending him, and even more amazed that he always found the time to meet and discuss a research problem, read a paper draft, or just provide some career advice. What most amazes me, though, is that he did all of these things while on sabbatical.

Paul Wright's research has always fascinated me, and I have learned a great deal from my interactions with his students. So, when Paul agreed to serve on my committee, in addition to his myriad other commitments across the College of Engineering, I was thrilleed. His timely feedback have been most appreciated.

Kris Pister always took the time to discuss my ideas, no matter how trivial they seemed at first. He also treated my like peer, from our very first interactions, even though it was clear to me that I had not earned it. I am especially thankful to Kris for reading my manuscripts, providing critical feedback, engaging in extended dialogue, and sharing a balanced research/industry perspective. His push to understand the fundamentals with quantitative analysis have left a mark on my work.

Shankar Sastry took a chance on me when it was not at all clear that I deserved one, so I am forever in his debt for his prescience in helping me find a home at Berkeley. I hope I didn't disappoint him.

I arrived at Berkeley just as the first generation of David Culler's sensornet students were beginning to graduate, so I had a chance to work with two groups of extremely talented students. My days in 467 were filled with natural light, wonderful discussions, sharp debates, and cold beer with fellow students including Alec Woo, Joe Polastre, Rob Szewczyk, Cory Sharp, Phil Levis, and Kamin Whitehouse. My days in 410 saw the natural light of 467 replaced with the artificial glow of flourescent and incandescent sources, and the faces of 467 slowly replaced with a new of set of colleagues including Jonathan Hui, Gilman Tolle, Sukun Kim, Jaein Jeong, Xiaofan Jiang, Jay Taneja, Arsalan Tavakoli, Stephen Dawson-Haggerty, Jorge Ortiz, and Albert Goto. This world-class group has taught me a lot about how to do research and have fun, and I've enjoyed their company as both colleagues and friends.

Jonathan Hui has been a long-time compatriot in crime outside of school and collaborator across a range of projects at Berkeley. I have always enjoyed our wide-ranging discussions, especially at Monkey Head Thursdays that Joe Polastre instigated and dragged us to. I've also been a beneficiary of Jonathan's tremendous attention to detail and belief in good workmanship.

Rodrigo Fonseca has been a wonderful collaborator. Although the hours we've spent working together have been relatively small, the bandwidth of our collaborations has been immense. I have also traveled the world, or at least Japan, Atlanta, and San Diego, with him and I can attest to the fact that he still doesn't seem to see a distinction between a double bed and two beds. Perhaps it's a concept that is lost in translation.

Phil Levis and Joe Polastre were graduate student mentors when I arrived at Berkeley, although I suspect neither of them knew it at the time. Phil and Joe are a case study in constrast, and I benefitted tremendously from discussing all manner of topics from research to startups to life with them. Joe shared much of what he had learned, often painstakenly, and Phil shared his office at Stanford while we worked on projects together.

Stephen Dawson-Haggerty has become, over the past year, the software ying to my (and increasingly, everybody else's) hardware yang. It seems like every time we talk, I learn something new about the internals of Linux, the operation of IPv6, or the bugs in TinyOS. And each encounter with Steve includes a new device driver, a routing protocol enhancement, or a more functional TCP. Convincing Steve to spend "two weeks" on HotMac has turned out to be one of the most fun and fruitful collaborations I have experienced in recent years.

My dissertation work has also benefitted from collaborations with many others including Paul Aoki, Mark Feldmeier, Animikh Ghosh, Santanu Guha, RJ Honicky, Bhagavathy Krishna, Santosh Kumar, Brano Kusy, Akos Ledeczi, Minh Van Ly, Alan Mainwaring, Miklos Maroti, Somnath Mitra, Chris Myers, Razvan Musaloiu-E. Joseph Paradiso, Kurt Plarre, Janos Sallai, Thomas Schmid, Prasun Sinha, Kannan Srinivasan, Andreas Terzis, Allison Woodruff, and Zizhan Zheng.

Special thanks to Gary Myers, Jonathan Hui, and Phil Buonadonna for their contributions to the Epic platform. Special thanks to Rodrigo Fonseca, Igor Ganichev, and Timothy Wark for their contributions to the Disco algorithm. Special thanks to Ali Rahami for helpful discussions and detailed derivations on the mathematical limits of Backcast, Steven Lanszisera for discussions about modulation schemes and radio receiver architectures, Rabin Patra for help with wireless channel emulation, and Intel Labs Berkeley for use of their laboratory facilities

Finally, my family's, and especially my wife Wendy's, love, support, and understanding these past few years have made my time at Berkeley possible and enjoyable, and having the whole family visit has led to some truly memorable times. I will always look back fondly on these years and on our family trips along the coast.

Portions of this disseration have appeared in various published forms at ACM and IEEE conferences including Chapter 1 ([43]), Chapter 2 ([42, 44, 40]), Chapter 3 ([48, 45]), Chapter 4 ([42]), Chapter 5 ([47]), and Chapter 7 ([41, 102]).

This dissertation is based upon work supported by the National Science Foundation under grants #0435454 ("NeTS-NR") and #0454432 ("CNS-CRI") and under a National Science Foundation Graduate Research Fellowship. This work was supported in part by a Microsoft Research Graduate Fellowship as well as generous gifts from Aginova, Hewlett-Packard Company, Intel Research, Keck Foundation, Microsoft Corporation, and Sharp Electronics.

Chapter 1

Introduction

A decade ago, the confluence of inexpensive MEMS sensors, low-power radios, reprogrammable microcontrollers, and affordable flash memory signaled the beginnings of a new computing class – tiny, low-power, wireless, networked sensors that could be deeply embedded and densely deployed to monitor macroscopic physical phenomena at previously intractable fidelity and scale. Wireless sensor networks, or simply "sensornets," enabled many new applications from low-rate climate monitoring [138] to high-rate seismic monitoring [145], from static sensors embedded in civil structures [87] to mobile sensors attached to wildlife [93], from sense-and-send applications that transmit every reading [132] to sense-and-filter applications that have no hope of transmitting all their data [49]. They also raised many new research challenges. Kahn et al. outlined the networking challenges [84], Estrin et al. outlined the coordination issues [59], and Hill et al. outlined system architecture concerns [72] for the emerging regime.

Today, the system and network architecture for *stationary* sensornets is largely solved, with many stable commercial solutions now available and standardization efforts underway at the IEEE, IETF, ISA, and within many industry groups. Of course, open problems still remain, like routing over low-power and lossy links, but with many off-the-shelf commercial solutions now available, the technology is finally maturing. As a result, research has shifted to exploring new domains like *mobile* sensor networks, since they enable novel applications in the home and office, for health and safety, and in transportation and asset management.

Researchers recently outlined a vision for monitoring human spaces using *mobiscopes* [6]. A mobiscope, according to their work, "is a federation of distributed mobile sensors into a taskable sensing system that achieves high-density sampling coverage over a wide area through mobility." Mobiscopes extend the traditional sensor network model and introduce new challenges in data management, data integrity, privacy, and network system design. Because of these new challenges, the authors contend that researchers need new architectures and methodologies for designing future mobiscopes. However, the two distinct classes of mobiscopes the paper discusses – *vehicular* and *handheld* – are energy-rich so their networking requirements are easily met with conventional approaches including cellular, Bluetooth, or WiFi technologies.

This thesis addresses *micropower mobiscopes*, a nascent class of mobile sensor networks that operate at low power. While vehicular mobiscopes can draw power from a vehicle's electrical system and handheld mobiscopes are recharged daily, micropower mobiscopes are are long-lived, embedded in the environment, and cannot be recharged easily. They may be integrated into clothing and accessories, or attached to everyday objects in the home, office, factory, farmhouse, or vehicle. To be truly unobtrusive, however, their power sources must be even smaller than the AA batteries used in many prototypical sensor nodes today, and these smaller batteries must either last for a long time or be automatically recharged *in situ*.

1.1 Mobility Changes Everything

Unlike traditional sensornets, the things that micropower mobiscopes monitor – people, animals, packages, and vehicles – exhibit often unpredictable mobility. Therefore, micropower mobiscopes are constrained like traditional sensornets, but are truly defined by their unpredictable link volatility and variable energy supply and usage. Mobility – and its effect on link dynamics and the energy budget – invalidates many assumptions implicit in today's low-power, static sensornets and high-power, handheld and vehicular mobiscopes. Mobility, for example, breaks link estimators and synchronous MAC protocols, and makes asynchronous neighbor discovery and routing fundamentally more challenging, so the techniques and protocols now in standardization for low-power, *static* networks are not well-suited to the needs of low-power, *dynamic* networks, like micropower mobiscopes. A sampling of these changes include:

- **Platform.** Micropower mobiscopes place new demands on the hardware. Motion sensing and processing is the *raison d'être* for many mobile applications, so platforms must provide low-power motion sensing in various forms. Mobile sensors may collect high-rate data or experience long periods of disconnected operation, so they must be able to efficiently store and retrieve potentially vast amounts of locally-logged information. During disconnected operation, the network cannot always provide a stable, global timebase on demand, so nodes must keep accurate time locally. The power budget of a mobile node is also constrained by what it can carry or what it can harvest from the environment.

- **Link.** Low-power link layers are greatly affected by mobility. Continuous, asynchronous neighbor discovery becomes essential for rarely awake or co-located nodes to discover each other quickly, reliably, and securely. Link estimation raises another challenge because history is a poor guide in predicting future performance. Data-driven beaconing and link estimation are poorly suited to a regime in which links are rare and volatile. Mobility also affects retransmission policies. Waiting briefly before retransmitting an unacknowledged frame may be appropriate in low-power static networks, but ineffective in mobile ones. More generally, issues of link estimation become ones of link detection, prediction, and termination, where agility is preferred to stability.

- **Network.** The choice of routing peers is already influenced by many cost metrics today including hopcount, expected number of transmissions (ETX) [34], and expected transmission time (ETT) [37], but such metrics do not incorporate mobility. Peers with little or no mobility may offer stable paths while peers with significant mobility are not likely to do the same. Routing in static networks is very different from routing in mixed networks, which is different from routing in mobile, ad hoc network.

- **Transport.** Mobility and intermittent connectivity raise many well-known transport layer challenges. The same issues arise in micropower mobiscope applications, but the challenges are exacerbated in this regime. Limited connectivity may require the transport layer to support storage. Connectivity may be fleeting and not simply intermittent, which requires prioritizing communications traffic. End-to-end path delays may be high, suggesting that TCP's end-to-end flow control may be unsuitable, especially given limited buffer space, and that bundle-oriented, hop-by-hop transfers are more suitable.

- **Application.** Mobile sensing diverges from the traditional sense-and-send data collection paradigm in many ways. At a minimum, the paradigm becomes one of sense-and-store and store-and-forward. Applications must also filter and prioritize data since connectivity is intermittent. New interaction patterns like talking, docking, and flocking also require new application-network interfaces. Finally, application-specific responses may be necessary to regulate communications in response to mobility.

1.2 Problem Statement

We claim that *five key primitives solve the structured mobility problem in low-power sensornets:* (i) awareness of mobility as a node-level metric and control signal; (ii) efficient and predictable asynchronous neighbor discovery and rendezvous; (iii) an efficient mechanism for probing an unknown number of unidentified neighbors; (iv) a receiver-initiated link layer that offers the conventional services required of the link; and (v) a delay-tolerant transport that offers a message queue abstraction to applications.

1.3 A Low-Power Mobile Sensing Architecture

We demonstrate our claim through the design and implementation of an system architecture for low-power mobile sensornets that embodies the five key primitives. The overall architecture is shown in Figure 1.1. Elements of the architecture span all aspects of the system stack, from the hardware to application layer, but our focus is on the lowest levels – the hardware and link layers – since they are the most affected and least explored in the low-power mobile sensing and mobile computing literature while many efforts have explored the network and transport layer problems.

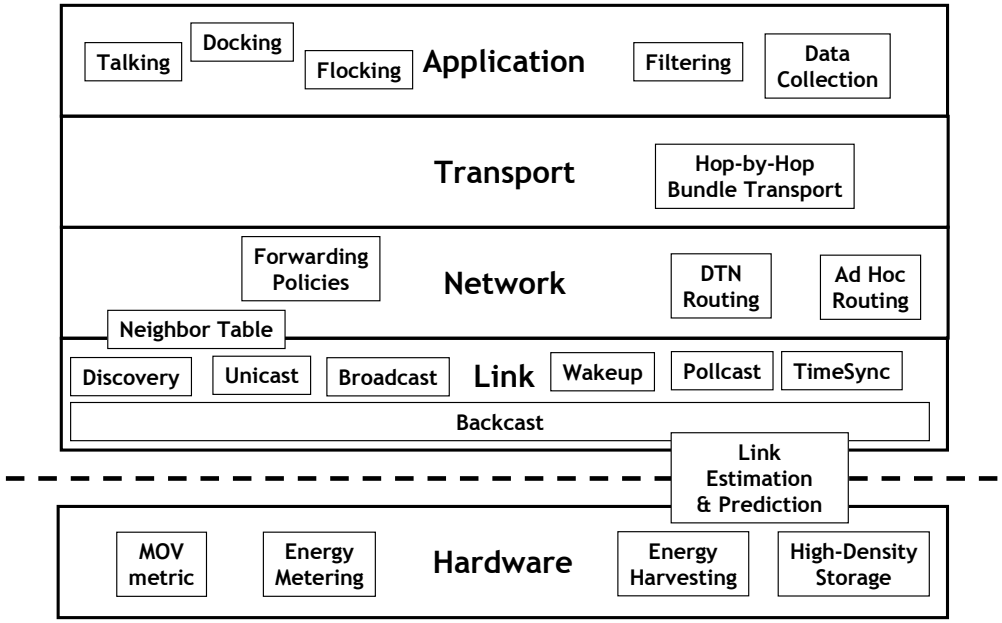


Figure 1.1: Elements of the low-power mobile sensing architecture spans all layers of the system stack.

The hardware platform must: (i) provide support for the MOV metric, an indicator of physical motion, on a near-nanopower budget; (ii) allow applications to introspect their own energy consumption through lightweight energy metering; (iii) support power management and energy harvesting; and (iv) offer high-density storage. Chapter 3 presents our hardware platform architecture for mobile sensing, several concrete design points, and the 3P's design methodology for hardware prototyping, piloting, and production.

In a low-power mobile network, asynchronous neighbor discovery is critical to finding new links and terminating old ones. We present a low-power, flexible, and reliable discovery algorithm in Chapter 4.

At the link layer, a new synchronization primitive is required that better meets the needs of mobile sensing applications. The primitive should be beacon-oriented (since in a mobile network, the topology may change often) and it should have low false positives and negatives, to avoid wasting precious energy. Chapter 5 presents a synchronization primitive that supports these needs.

Mobile sensors may also participate in static networks and have to interact static nodes, so the link layer for mobile sensors must be compatible with the link layer used by static nodes. Chapter 6 presents the design of receiver-initiated link layer that is well-suited to the needs of mobile sensors but also offers unicast, broadcast, wakeup, discovery, and pollcast services suitable for static nodes as well. Other services, like timesync and link estimation, are important but have been covered in our prior work [89, 115, 127, 128].

Chapter 7 presents three low-power mobile sensing applications. These applications – AutoWitness, SleepTrack, and CommonSense – are point studies of a more general set of asset tracking, health/fitness, and participatory urban sensing applications, and they each stress different architectural elements.

1.4 Contributions

This dissertation surveys the landscape of mobile sensornet applications, discusses the many challenges that mobility raises, presents an architecture for micropower mobiscopes, and focuses on a handful of key concerns across several layers of the system stack. We explore the prevailing technology trends and observe that motion sensing technology is approaching near-nanopower budgets, that high-density storage (in the gigabytes) is now less costly than microcontrollers and radios, that energy harvesting technologies are maturing to the point of commercial availability, and that thin-film batteries are available as surface mount components. These enabling technologies portend a bright future for micropower mobiscopes.

Within this new context, we elaborate on our earlier claim: although *mobility changes everything*, the mere *knowledge of mobility can help address the ensuing challenges*; in other words, we can turn the mobility *bug* into a *feature*. Accordingly, we suggest that real-time knowledge of motion should be made available by the platform and fully exploited. We propose a new metric that indicates node-level movement, called MOV, and we explore how it can address many mobility-induced challenges in micropower mobiscopes.

In this dissertation, we present our efforts to build a low-power, mobility-aware platform and network architecture for wireless sensornets. This dissertation makes three major contributions:

1. We develop a modular hardware platform architecture, built on a decade of experience, that incorporates all essential components. These include the mote core (including processor, radio, and storage), peripheral modules (including motion sensing, storage, and energy harvesting), carrier boards (that provide the platform "glue," include additional functionality, and satisfy application-specific concerns), and a modular, building block methodology to support overall platform composition and synthesis. Following this approach, we present several concrete design points suitable to mobile sensing, including the Irene mote, Common Sense Badge, and Open Mesh border router.

2. We develop a mobility-aware network architecture that defines the key services, their interfaces, and their interactions. They include, at the platform layer, the MOV metric, a node-level metric that captures movement in various forms, including shock, vibration, acceleration, and displacement. At the link layer, we develop asynchronous neighbor discovery, link quality prediction, a synchronization primitive, and a receiver-initiated medium access control protocol. At the network and transport

layers, we adapt existing IPv6-based solutions to incorporate mobility into routing decisions and develop a delay-tolerant network stack for data delivery over an intermittently-connected network.

3. We present the implementation and evaluation of all the key elements of the architecture. These include *MOV*, a family of motion detection triggers; *Irene*, a mote that embodies many of the key platform ideas supporting mobility, including hardware support for various forms of motion detection, timekeeping, storage, optional energy metering, and a form factor suitable for mobile sensing applications; *Disco*, an asynchronous neighbor discovery protocol that allows nodes to discover neighbors quickly and efficiently; *Backcast*, a link layer synchronization primitive that allows a node to efficiently poll multiple neighbors in parallel; *HotMac*, a receiver-initiated, mobility-aware MAC protocol based on Backcast that supports unicast, broadcast, discovery, wakeup, pollcast; and *TinyDTN*, a mobility-aware, disruption-tolerant network stack that uses transport-layer storage and opportunistic forwarding. We also evaluate the system using complete applications that incorporate the essential elements of the architecture. This implementation shows that micropower mobiscopes are indeed feasible, that knowledge of mobility improve the power and performance of an application, and that many of the techniques are useful for static sensornets and may be generalized to other systems.

1.5 Roadmap

We begin our journey in Chapter 2 by exploring a range of mobile sensing applications and workloads, identifying three basic application patterns: talking, docking, and flocking. We then present the fundamental design challenges for low-power mobile sensors: link volatility, bandwidth limitations, energy variability, and platform constraints. We then review the state-of-the-art in low-power and mobile sensornets as well as mobile and disruption-tolerant networking. We find that although great progress has been made in each of these areas individually, the intersection of mobility and low-power raises many new challenges. This leads us to conclude that mobility changes everything in low-power wireless sensornets, requiring a new architectural approach.

In Chapter 3, we present a platform architecture for low-power, energy-efficient, and mobility-aware sensor nodes. Our design includes (i) low-power motion sensing and processing; (ii) non-volatile storage, (iii) power subsystems for energy harvesting, storage, tracking, and delivery, and (iv) the ability to keep accurate time. These platform features allow hardware to provide MOV efficiently, offer storage for sensor data and route-through traffic, match the waxing and waning of energy supply with adaptive consumption, and correlate timestamped sensor data from distinct nodes. We also present Irene, a concrete design point embodying these ideas, and an overall architecture and building block approach to platform design.

In Chapter 4, we present Disco, a power-proportional, asynchronous neighbor discovery and rendezvous protocol that allows two or more nodes to operate their radios at low duty cycles (e.g. 1%) and yet still discover and communicate with one another during infrequent, opportunistic encounters without requiring any prior synchronization information. We show how Disco supports the talking, docking and flocking interaction patterns, how it performs with dissimilar duty cycles, how discovery latency is proportional to duty cycle, and why it is robust to clock skew. We also explore why an asynchronous discovery service is essential for low-power mobile sensors.

In Chapter 5, we introduce Backcast, a link layer synchronization primitive that allows a node to determine whether any neighbors have pending traffic. Backcasts are useful as a form of acknowledged anycast when a node does not know its neighbors *a priori*, or has many communicating neighbors, and they

strike a balance between false alarms (staying awake in the absence of pending traffic) and false negatives (falling asleep when there is pending traffic). Backcast serves as our fundamental link primitive, enabling semi-passive vigilance at the link layer. We show that backcast is feasible using commodity hardware, general because it enables multiple network services (which we implement at the MAC layer), efficient because it runs in constant time independent of the number of responding neighbors, and scalable because it works with many interfering acknowledgments.

In Chapter 6, we present HotMac, a receiver-initiated link layer that works well across a broad spectrum of workloads, supporting both mobile-to-mobile and mobile-to-static communications, and offering many attractive features including ultra-low duty cycle support, high channel efficiency, and mobility-aware, power-proportional operation. HotMac supports a host of link layer services including unicast, broadcast, network wakeup, and pollcast. HotMac also supports mobility-aware operation, including beaconing and listening on motion triggers, as well as support for link prediction.

In Chapter 7, we present several mobile sensing applications that embody the ideas developed in this dissertation, including mobility-aware networking and delay-tolerant transport. These applications stress different aspects of the architecture, including interaction patterns, motion detection, energy harvesting, networking, storage, and lifetime. We show that our architecture is general enough to be used across applications as distinct as tracking of periodic limb movements and their causal effects, detecting and tracking asset theft with ultra low-power tags, and sampling air-quality using distributed handheld monitors.

We conclude our journey in Chapter 8 by revisiting the challenges that mobility raises, our solutions to these challenges, and the broader applications of this work. We also identify developments within the IEEE that signal growing interest in the approaches we advocate. Finally, we observe that the mechanisms developed in this thesis are useful for a range of more traditional systems – that what's good for mobile sensors is also good for static sensors, mobile phones, laptop computers, and datacenter networks.

Chapter 2

Background

Our view of the wireless sensor network landscape divides applications according to two basic factors, node mobility and energy constraint, as Figure 2.1 shows.

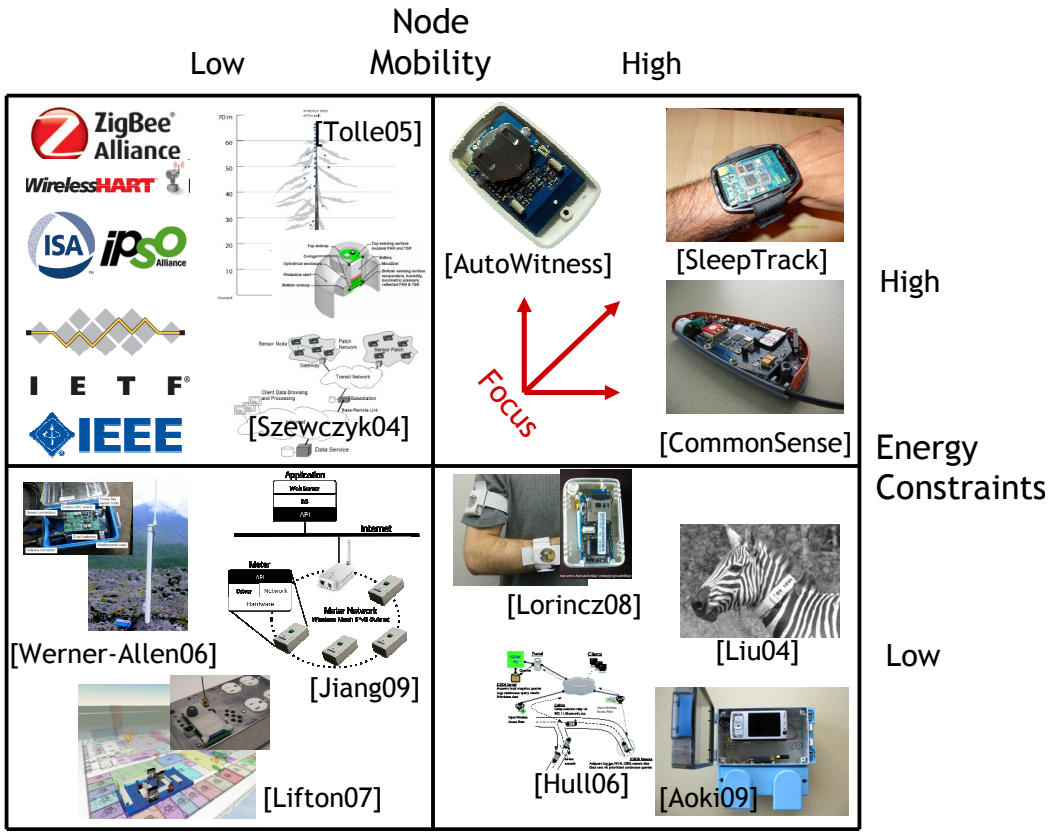


Figure 2.1: The wireless sensornet landscape and the focus of this dissertation.

Low Node Mobility and Low Energy Constraints. These applications are typically mains-powered, solar-powered, short-lived, or high data rate. Although many sensornets have explored this regime, more broadly, this area includes wireless industrial automation, wireless telemetry, stationary RFID readers (but

not tags), and even wireless mesh networks.

Low Node Mobility and High Energy Constraints. These applications represent traditional sensornets that are physically embedded, battery-operated, and stationary. They are often used for environmental data collection or habitat monitoring. They address many challenges that arise in low-power wireless networking, but today, these are largely solved for well-behaved stationary networks, with standardization efforts underway at the IEEE, IETF, ISA, and many industry groups.

High Node Mobility and Low Energy Constraints. These applications include sensing using vehicular or handheld mobiscopes, for example, for vehicle-based traffic monitoring or participatory urban sensing. Although these applications raise many research challenges, energy is not one of their major constraints since they can be powered from the vehicle or recharged daily. With few energy constraints, they can use conventional communications technologies like cellular, Bluetooth, or WiFi.

High Node Mobility and High Energy Constraints. These applications include low-power asset tracking, home health care, and smart attire that experience structured mobility in that a few common interaction patterns have emerged. They are characterized by their often unpredictable mobility which inverts traditional assumptions in low-power design. Designing a suitable architecture for these applications is the focus of this dissertation.

2.1 Traditional Mobiscopes

Mobile sensor networks, or *mobiscopes*, have much to offer for monitoring human spaces. A mobiscopes is "a federation of distributed mobile sensors into a taskable sensing system that achieves high-density sampling coverage over a wide area through mobility," and they extend the traditional sensor network model and introduce new challenges in data management, data integrity, privacy, and network system design. Two classes of mobiscopes – *vehicular* and *handheld* – have garnered considerable interest recently [6].

2.1.1 Vehicular

Vehicular mobiscopes are used for traffic [86], automotive [57], and environmental [9] monitoring applications. They can integrate with vehicle power systems or simply be location-aware phones carried in cars. Regardless, they leverage parasitic mobility to spatially sample traffic, road conditions, weather, or the environment. In some applications, they sample points on a manifold (e.g. weather). In other applications, they monitor the operation of the vehicle itself across space and time (e.g. engine RPM vs location). In still other applications, they estimate flows using probes (e.g. traffic). Figures 2.2(a)-2.2(b) show street sweeper-based vehicular mobiscope for air quality monitoring and Figures 2.2(c) shows a visualization of this information.

2.1.2 Handheld

Handheld mobiscopes principally support mobile participatory sensing approaches to collecting data about people, their interactions, and the environment. Their use converges with daily patterns of human activity, including daily recharging. Mobile participatory sensing uses consumer electronics (e.g., mobile Internet devices and mobile phones) to capture, process, and disseminate sensor data, complementing alternative architectures by "filling in the gaps" where people go but sensor infrastructure has not yet been

(a) Vehicular [9] (b) Electronics [9] (c) Visualization [41] (d) Handheld [41] (e) Electronics [41]

Figure 2.2: Vehicular and handheld mobiscopes. (a) Vehicular mobiscopes can draw power from a vehicle's electrical system, like this air quality monitor, which gets power from the street sweeper on which it is mounted. (b) Many vehicular mobiscopes are built around a cellular phone-based backhaul, as in this case, or using open WiFi-based backhauls. (c) Mobiscopes often report data to a central server which allows web-based visualization of data from many sensors to be overlaid and integrated into mapping software. (d) Handheld mobiscopes are carried by people and recharged daily, like this air quality monitor. (e) Handheld mobiscopes may have a cellular radio for wide area communications as well as a low-power radio like Bluetooth for tethering with cell phones and an 802.15.4 radio for connecting with other sensors (and acting like a mobile router or data mule). Photo credits: (d) Christopher Myers; (e) Mazzarello Media and Arts.

installed. While some types of sensors are already commonly present in consumer devices (e.g. geolocation, motion, and sound), other kinds of sensors (e.g., air quality) are not commonly included but can provide additional data of individual and social interest, as Figures 2.2(c)-2.2(e) show.

2.2 Micropower Mobiscopes

While vehicular mobiscopes can draw power from a vehicle's electrical system and handheld mobiscopes are recharged daily, micropower mobiscopes cannot be recharged easily. Micropower mobiscopes, lacking the cellular radio that continuous power affords in vehicular mobiscopes, and daily recharges enable in handheld mobiscopes, exhibit more nuanced interaction patterns during mobile-to-mobile or mobile-to-static interactions. We call these three patterns *talking*, *docking*, and *flocking*, and suggest that they represent *the* canonical interaction patterns that must be supported.

2.2.1 Typical Application Classes

Talking. In the *talking* pattern, two mobile nodes encounter each other, communicate, and part ways. Such encounters can provide insight into many real-world social network effects [23]. Tracking social interactions can help epidemiologists study the spread of disease in schools or sociologists better understand children's social development patterns. In the workplace, many interactions occur face-to-face and outside the purview of computing: water cooler conversations pass along important gossip, many executives and managers "manage by walking around," and individuals spread information epidemically. Other examples of the talking pattern include detecting zebra-to-zebra encounters [93], tracking networking researcher interactions [90], logging hiker sightings after their trail-side encounters with other hikers [75], tracking doctor-patient interactions to understand nosocomial infections [70], and mapping the social network of laboratory rats in the wild [137]. In all of these examples, the nodes are peers and they may independently set their duty cycles to reflect individual energy supplies, mobility patterns, or data transfer needs.

Docking. In the *docking* pattern, a mobile node discovers a static node situated at a rendezvous point. For example, a company might want to ensure that a guard is doing his rounds by checking the encounter logs in all corners of an office. Other applications from the literature include uploading cargo transit history at readers [98], reporting cattle movements during feeding times [144], tracking hikers via their encounters with trail-side waypoints [75], reporting ski trail conditions using skiers as data mules [53], tracking researchers' whereabouts using Active Badges [142], and ensuring health care workers sanitize their hands immediately before entering a patient's room [70]. In all of these scenarios, the nodes are asymmetric in mobility and function. In some situations, the static node receives, stores, and forwards data received from the mobile node. The static nodes may be mains-powered, as in the case of sensor access points [52], or energy-constrained, as in the case of a DTN throwbox [13]. In other situations, the roles are reversed and the mobile node becomes the data mule, ferrying data between a static node and a wider network [151].

Flocking. In the *flocking* pattern, a group of nodes move together as a unit. For example, an elementary school teacher might want to know if all the children are on the bus before it departs from the zoo and, if a child is missing, when that child was last seen and by whom. A child in the group might want to know if she is about to be separated from her classmates. A railroad might want to determine the manifest of a train en route or the order of its constituent rail cars, especially as it switches cars in and out at sidings and rail yards. Other examples from the literature include a group of bicycle-mounted wireless sensors moving as group [52], a particularly forgetful person who wants to ensure that his personal items remain together and be notified if he leaves one behind [18], one node in a constellation of sensors monitoring a Parkinson's patient wants to inform the other sensors if it discovers an access point where data can be uploaded [109], or an airborne wireless sensor network of micro-air vehicles may move as a flock to perform atmospheric sensing or monitor storm dynamics [7]. In all of these cases, the nodes operate as a group, need to maintain membership information, and may communicate with one another, potentially over multiple wireless hops.

2.2.2 Application Requirements

Traditional *sensornets* are distinguished from traditional *networks* because sensornets are embedded in physical space, composed of large numbers of nodes, and sensitive to a low total cost of ownership [76]. Traditional *mobiscopes* (both vehicular and handheld ones) relax these requirements somewhat since sensors may be only loosely "embedded" and still have access to vehicle power or daily recharge. Or they may trade mobility for density, by spatially sampling with a fewer number of mobile sensors rather than a greater number of static ones, which relaxes unit costs.

The application classes surveyed in Section 2.2.1 show that *micropower mobiscopes* are more similar to traditional sensornets than they are to traditional mobiscopes. However, three fundamental requirements distinguish micropower mobiscopes from traditional sensornets: (i) small form factor, (ii) unpredictable mobility, and (iii) storage-centric operation. These differences have many architectural implications.

Small Form Factor

Micropower mobiscopes will be worn by people and animals, integrated into clothing and accessories, embedded in shipped goods, and attached to everyday objects. Such tight coupling of the objects being sensed and devices doing the sensing requires that the hardware be small and unobtrusive, eventually converging with everyday objects like watches, shoes, clothing labels, dog tags, waybills, nameplates, and stickers. To be unobtrusive, the sensors will need to be wireless and exhibit a mostly two-dimensional footprint. Although integrated electronics can easily scale to such form factors, power sources currently cannot.

As a result, small size demands very low power operation, making wide area communications using cellular radios impractical and other high-power radios unpreferable. Rather, nodes will require low-power, short-range radios to meet non-trivial lifetime constraints. Many devices will be battery-powered, but some may harvest ambient solar, vibration, thermal, or radio frequency energy. However, with a small form factor, the energy available often will be limited, underscoring the need for low-power operation.

Unpredictable Mobility

Micropower mobiscopes and the things they monitor are conjoined, so the sensors will exhibit the same (unpredictable) mobility patterns as their hosts. This has several implications. First, the intersection of unpredictable mobility and short-range radios implies significant link volatility and topological turmoil, meaning beaconing strategies and asynchronous neighbor discovery will be needed, requiring us to revisit how low-power operation is achieved at the link layer. Second, the energy that nodes harvest from the environment will be highly variable because both the environment and the system's exposure to it will change in unexpected ways. This variability in both energy usage and income requires greater energy-awareness and workload adaptivity. Third, nodes will often experience disconnected operation, requiring them to adopt disruption-tolerant networking techniques.

Storage-Centric Operation

In many applications, sensing data rates will exceed the capacity of the network to deliver the data, requiring node-level filtering or long-term storage. This signals a fundamental shift in application semantics from *sense-and-send* to *sense-and-store*, along with attendant techniques needed for querying the stored data. In addition, disruption-tolerant networking requires energy-efficient, non-volatile storage to cache in-flight data, which the hardware platform must provide affordably.

2.3 Technical Problem Statement

Mobility breaks assumptions underlying current approaches to reliable, low-power operation but five primitives solve the structured mobility problem in low-power sensornets. First, awareness of mobility as a node-level metric and control signal, as provide by the MOV family of metrics, allows a node to intelligently regulate its energy usage. Second, efficient and predictable asynchronous neighbor discovery and rendezvous, as provided by Disco, allows a node to quickly discover and efficiently monitor its neighborhood topology with a predictable power draw, despite frequent (or infrequent) link churn and varying neighbor density. Third, an efficient mechanism for probing an unknown number of unidentified neighbors, as provided by Backcast, allows a node to quickly determine if any neighbors are within range or have pending traffic, even when their identities are not known and their presence is not predictable due to mobility. Fourth, a receiver-initiated link layer that offers the conventional services required of the link, integrates the first three primitives within a unified framework, and interoperates with and can support the communication patters of low-power stationary networks, as provided by HotMac. Fifth, a delay-tolerant transport that offers a message queue abstraction to applications that can buffer application traffic for delivery upon opportunistic connectivity, as provided by TinyDTN.

Chapter 3

3P's Hardware Platform Architecture

In this chapter, we present the design and implementation of three very different platforms that support our low-power mobile sensing architecture. These platforms include Irene, a wearable wireless sensing node targeted at motion sensing applications with full support for the MOV metric; the Common Sense Badge, a handheld air quality monitor with multiple radio interfaces; and the 6LoWPAN border router, an IPv6 router that connects sensor networks to the Internet. All three platforms are built around a common module – the Epic Core – and they are designed using the 3P's building block approach to hardware platform design.

A major focus of this chapter is presenting the 3P's approach to platform architecture that supports the *prototype*, *pilot*, and *production* stages of hardware design, and preserves the artifacts and learnings accumulated in moving quickly through these design stages. This approach, based on a decade of collective experience, arrives at an architecture in which general-purpose *modules* that require expertise to design and incorporate commonly-used functionality are integrated with application-specific *carriers* that satisfy the unique sensing, power supply, and mechanical constraints of an application.

We present heuristics for partitioning functionality between modules and carriers, and identify guidelines for their interconnection. Our approach advocates exporting a wide electrical interface, eliminating the system bus, and supporting many physical interconnect options for modules and carriers. We evaluate this approach by constructing a family of general-purpose modules and application-specific carriers that achieve a high degree of reuse despite very different application requirements. We show that this approach dramatically shortens platform development time and has been broadly adopted beyond the scope of this dissertation. More than thirty application-specific platforms have been developed following the 3P's approach, many by novice graduate students building their first platform, showing the utility of the approach.

3.1 Overview

Sensornet hardware designs, like most other embedded systems, are tightly coupled to their applications. Constraints including form factor, sensor suite, and power supply are often particular to an application, making it difficult for general-purpose platforms to address these application-specific needs. This forces platform designers to accept either suboptimal solutions or to repeatedly reimplement functionality. We propose a third way that composes platforms from a two-layer hierarchy consisting of compact, general-purpose *modules* which provide the common functionality and application-specific *carriers* which glue together these modules and also incorporate the sensors, power supplies, and mechanical constraints unique to the application.

Of course, sensornet platforms and modular approaches are widespread, but past efforts have failed to produce an architecture that supports the 3P's. Despite the diversity of prior platform design efforts, none of the available options meet all of these goals, as Section 3.2 examines in detail. In this chapter, our focus is on an overall platform architecture for supporting the three phases of sensornet development – *prototype*, *pilot*, and *production*. This focus acknowledges the tensions among design tradeoffs in a rapidly changing field. The desire to tackle new, unexplored problems means that rapid prototyping and "try it and see" experimentation are very important. The wide diversity of valuable applications make realistic pilot studies at modest scale and modest investment essential, and these have to be well-enough executed to gain unprecedented measurements. And the maturing of the field means bringing the technology into production state, reducing cost, optimizing performance, improving manufacturability, and obtaining high reliability, all while preserving the learnings and artifacts accumulated along the way in moving rapidly through these phases of development.

This chapter presents a building block approach to sensornet platform design represented by the Epic family which we believe is the first to support all three phases of sensornet platform development well enough for rapid forward going innovation. The key ideas behind this approach include systematically partitioning functionality, exporting a wide electrical interface for modules, eliminating the system bus, and supporting multiple ways of physically interconnecting modules and carriers, from hand-soldering to machine-assembly. The specifics of this approach are presented in Section 3.3.

At the heart of the Epic family is a core module that integrates a state-of-the art microcontroller, IEEE 802.15.4 radio, and flash memory onto a small, inexpensive, single-sided board with excellent RF characteristics. Following the architectural principles of exporting a wide electrical interface and minimizing logical interface constraints, the core exposes essentially all the pins that might possibly be useful, including internal signals, and does not hide any of these signals behind a multiplexed system bus. The core can be snapped into a standard socket for prototyping, easily soldered to routine carrier boards for pilots, or inlined for production, according to the principle of supporting many physical interconnect options. Despite this architectural focus, we recognize that modules can be only ϵ-suboptimal if they are to be enthusiastically adopted. Therefore, module designers must go to some lengths to ensure that the basic building blocks exhibit competitive performance. Section 3.4 describes the Epic core module and its internal subsystems, introducing the key characteristics and revisiting part selection with these in mind, looks at new alternatives since the core was designed, discusses manufacturing and mechanical considerations, provides a quantitative analysis of core module performance, and outlines recent developments and future directions for the core.

The case for a core module is clear: effective RF engineering requires deep expertise to design high-frequency circuits and specialized equipment to assemble, test, and tune them. These reasons are not limited to the core module, however. For example, a solar harvesting circuit can present a range of design options and subsystem choices that a designer unfamiliar in the art would find difficult to navigate. There are other reasons to build modules as well. Some functions are so common that reuse in modular form is inevitable. Many platforms, for example, require a USB host interface or battery charger, so this is an obvious module candidate. Finally, sometimes it is simply more convenient to group a set of related chips together on a board, like a handful of different memory technologies to create a memory hierarchy module. Collectively, these principles provide some guidance for partitioning functionality between modules and carriers. Complementing the core module is a supporting cast of specialized peripheral modules that offer a handful of choices for complete systems, and a framework for forward going innovation, as Section 3.5 describes.

The glue for these modules are breakout and development boards or application-specific carriers. For prototyping, breakout and development boards expose a wide array of pins and allow modules to be socketed, enabling novice system builders to compose platforms using simple jumper wires in a "try it and see" fashion and module developers to debug otherwise complex systems with complete freedom to access all exposed and intermediate signals. Section 3.8 explains our overall vision and approach for prototyping using the Epic family and presents some development hardware designed to support such prototyping efforts.

For pilots, inexpensive two layer carriers are typically designed to fit a particular enclosure and a set of mechanical constraints with Epic modules being treated just like chips. For production, the modules are eliminated by incorporating their contents directly into the underlying board through *hardware inlining*. Section 3.9 evaluates the architecture by illustrating how these building blocks are used to build several simple, cost-effective, and application-specific carriers are designed using freely available CAD tools, inexpensively manufactured, and hand-assembled by novice graduate students. Carrier board design is so simple that it can be used even in an undergraduate classroom setting where students do application-specific design, fabricate the boards, and assemble a final solution in just weeks.

The final sections reflect on how effectively the Epic approach meets the various contraposing design goals of the three phases of sensornet platform development. Our experience shows that the building block approach leads to greater reuse, more compact designs, increased simplicity, and lower overall part count. Not only do modules become true building blocks, but so do other components created like CAD parts and scripts. An important benefit of viewing hardware in this way is that modules capture working hardware designs. In the future, we envision others will create many new modules make them available to the wider research community.

3.2 Related Work

In the early stages of wireless sensor network research, the architecture and the form factor of the platform were wide open questions. The UCLA WINS project developed WinCE-based devices about the size of a shoebox [113]; USC developed PC/104 devices and proposed a tag that would have a small motherboard with slots for a radio board, a power board, and sensor boards [116]; the UCB SmartDust project developed the WeC mote with two microcontrollers, a radio, and a couple of sensors on a disk the size of a half-dollar [4]. Numerous other projects developed a variety of ARM-based systems. The Berkeley René mote [73] began a sea change by integrating the core elements of the low-power WeC design into a simple board with an array of common analog and digital interfaces organized like a conventional system bus on a 51-pin connector.

The René design reflected a key understanding that the common elements across sensor network applications are sampling, processing, storage, and communication, while the parts that are application-specific are the sensor suite, the power subsystem (which can support the application's sample and communication rate), and the mechanical design which holds the three together, exposes the sensors to the phenomena they need to sense, and protects the rest. This 51-pin "AT Bus" of the sensornet world carried forward to the MICA [73], MICA2 [28], MICAz [31], IRIS [29], and many, many other designs. Numerous sensor boards and power boards were designed to stack on it. In many ways, it shaped sensor network research activities for over five years.

Unfortunately, the 51-pin connector proved to be unsound for long-term deployments in harsh conditions, and it was expensive relative to the other components in the system. It began to fail the Goldilocks test – instead of being "just right" it was often too general for simple applications and too limited for demanding

ones. New microcontrollers, new radios, and new flash chips led to a variety of new mote designs, such as the Mica2Dot [30], Telos [112], iMote [79], BTnode [16], Eyes, TIP [100], TinyNode [38], Sensinode [118], IRIS [29], MICAz Stamp [31], and kMote. Each with different form factor, connectors, power requirements, and interfaces.

In hindsight, this chaos was a symptom of an underlying tension among design tradeoffs. The rapidly changing nature of the field and the desire to explore novel applications meant that prototyping and experimentation were very important. Meanwhile, realistic pilot studies at modest scale were essential to gain unprecedented measurements, leading researchers to either use commercial offerings that were often not quite right or design their own platforms from scratch at great opportunity cost. And while the maturing of the field meant bringing the technology into production state, none of the commercial offerings addressed the unique challenges in moving through the design phases, critical for preserving the accumulated learnings and artifacts.

Despite the diversity and scope, prior approaches remain inadequate because they fail to address the spectrum of needs for prototype, pilot, *and* production usage. We classify these approaches into three broad categories, *bus-based*, *highly-integrated*, and *assembly-optimized*, and explore their drawbacks.

The modular, stackable approach of bus-based architectures like WINS [113], MICA [73], iMote [79], PASTA [116], Stack [15], MASS [51], and mPlatform [96] make prototyping mechanically simple but their busses present barriers to interfacing peripherals and also result in signal conflicts if not multiplexed, and their backplanes and board stacks are too bulky, expensive, or fragile for realistic pilot or production use, as many deployments discovered.

The highly-integrated approach, advocated by Telos [112], bundles a mote core with sensors, antenna, and host interface into a single circuit board, which makes software development and desktop experimentation easy. However, with this approach, realistic prototypes and pilots are strained because too few I/O lines are exported, production costs are too high since many unnecessary features are integrated, and onboard sensors are not useful for many scientific purposes.

To address the various shortcomings of the bus-based and highly-integrated approaches, vendors began to offer new, assembly-optimized module versions of their core platforms, like the MICAz [31], IRIS [29], and Tmote Mini [119]. These modules, while well-suited to high-volume surface-mount assembly failed to eliminate duplicated expertise, are challenging to integrate into prototype and pilot projects because their packaging makes hand-assembly and socketing difficult or expensive, and their relatively narrow interfaces hide many internal signals useful for research.

3.3 Building Block Approach

This section presents the architecture and principles that support the prototype, pilot, and production stages of platform design, and preserves the artifacts and learnings accumulated in their implementation. At the heart of our approach are two architectural elements: the *module* and the *carrier*. Modules are reusable, self-contained subsystems in a multi-chip module (MCM) package. Modules are composed of one or more packaged ICs and other electronic components typically found on a system board. Carriers are custom circuit board substrates that glue together general-purpose modules with application-specific sensors, power supplies, and mechanical constraints. Heuristics for partitioning functionality between modules and carriers are an important aspect of the building block approach. The value of these heuristics, and their effectiveness in creating reusable platform components, are discussed in Section 3.10. A sensornet platform constructed using the 3P's building block approach is shown in Figure 3.1.

Figure 3.1: A sensornet platform designed according to the 3P's building block-approach. A general-purpose module (square circuit board) is attached to an application-specific carrier (rectangular circuit board). The carrier includes and accelerometer (small black square near the upper left-hand corner) and vibration switches (cyclindrical devices on the backside), hosts a coin cell battery (backside) and power supply (along the top, next to the card edge connector), and conforms to a standard enclosure (footprint and two mounting holes).

Several principles focus on the interface between modules and carriers. First, we observe that a bus adds cost and complexity but that effective modularity does not really require one. Therefore, we *eliminate the system bus* from a module's interface specification. This allows modules to be flexibly wired together in whatever way a designer sees fit, rather than being encumbered by the constraints of a generic system bus since it uses precious circuit board space, requires costly or bulky or fragile connectors, complicates integration of peripherals, and reduces generality. Extending this line of thought, modules should *export a wide electrical interface* to maximize generality and reuse potential. Finally, to support prototype, pilot, and production purposes, modules should *support many physical interconnect options* ranging from socketing to hand-soldering to machine-assembly.

3.4 Core Module

Epic platforms are organized around a general-purpose core module as well as optional peripheral modules. This section describes the core module, shown in Figure 3.2, which is essentially the guts of a mote without the constraints on how it can be used. The core module integrates a state-of-the-art microcontroller, IEEE 802.15.4 radio, flash memory, a 48-bit unique serial identifier, and a U.FL RF connector, all attached to a four-layer, 1 mm thick, LCC-68 form factor circuit board one inch on a side, as Figure 3.3 shows.

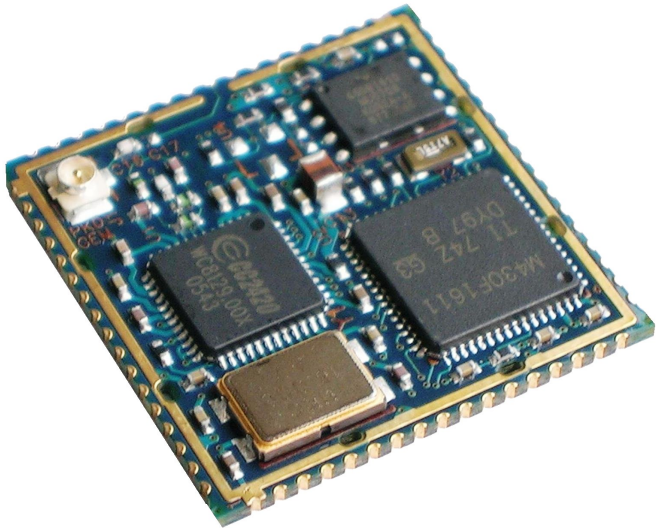


Figure 3.2: The Epic Core module: a wireless sensornet node ("mote") core that integrates a microcontroller, radio, and flash memory into a square inch form factor that can be hand-soldered, socketed, or machine assembled.

Architecturally, the core is very similar to earlier mote designs like Telos [112] and MICAz, but its design is part of a larger framework that seeks to enable the 3P's methodology that supports the prototyping, piloting, and production of sensornet platforms. To be useful for prototyping, the core module must be easy to use, debug, and profile, and it must provide good performance, sufficient storage, and ample I/O lines. To be useful for pilots, the core module must be easy to design-in at the CAD level, simple to hand solder at bench scales, cheap at cm scale, and flexible when it comes to antenna choices. The core must also be easy to program in-circuit and debug *in situ*, both at the hardware and software levels. To be viable for production, the core must provide performance comparable to commercial modules, have an attractive cost profile, satisfy regulatory constraints like RoHS and FCC, and be open source to allow unforeseen innovation, adaptation, and hardware inlining.

3.4.1 Component Choices Revisited

When this study was started in 2007, a handful of new microcontroller and radio options were available that did not exist when earlier platforms were designed, and today this list has grown even longer. This situation raises the question of whether earlier component choices still hold given today's offerings. The short answer is that when the core was designed, the answer was still yes. Today, the answer is less clear, and this has led to the design of a pin-compatible follow-on core. Moving forward, the answer depends largely on application drivers like performance and cost. The rest of this section articulates the long answer to this question.

The opportunity to revisit the core's design raises another architectural question: *what changes are needed regardless of component choice to effectively support prototype, pilot, and production designs?* Addressing this question is a central contribution of this work.

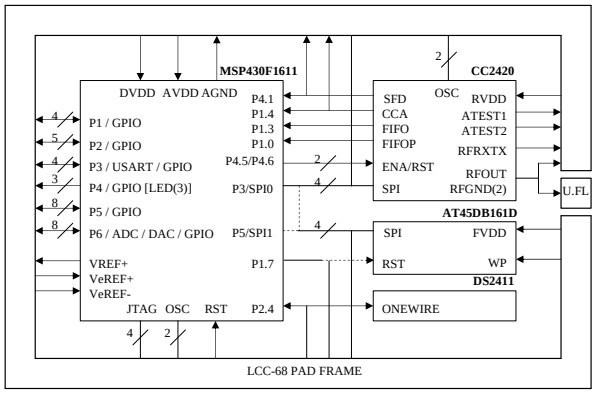


Figure 3.3: The Epic Core architecture. A Texas Instruments MSP430F1611 microcontroller and CC2420 radio sit at the heart of the core module. An Atmel AT45DB161D NOR flash provides 16 Mbit of storage. A Maxim DS2411 provides a globally unique serial identifier. Nearly all MCU peripherals are exported, including GPIO lines, ADC inputs, ADC voltage references, DAC outputs, USART lines, and the JTAG module. Many internal connections between components are exported as well.

Microcontroller

The microcontroller market includes many new offerings that were not available when earlier generation mote platforms were designed, as Table 3.1 summarizes. Some of these products, including the TI MSP430F2618 and MSP430F5437 are product line extensions of existing microcontrollers like the MSP430F1611 that offer more memory, better performance, or new features. Other products, like the Jennic JN5139 or Atmel ATmega1281, were not available for consideration until recently. Given these new choices, it is worth revisiting why the MSP430F1611 still makes sense. Several factors influenced the decision to use this microcontroller, but most of the reasons are the same as the ones articulated in the Telos mote design [112]. These include low active current, wide operating voltage range, a 16-bit sleep timer, fast wakeup from sleep, a large amount of RAM, and three direct memory access (DMA) channels that can operate while the CPU sleeps.

By these metrics, the Atmel ATmega1281 (and larger ATmega2561) look more competitive than their predecessor, the ATmega128L. The active current has remained approximately constant at 0.9 μA at 1 MHz, only about twice that of the MSP430F1611. Since the microcontroller does not dominate the system power budget, this difference is not likely to have a large impact on lifetime. The operating voltage of the ATmega1281 matches the MSP430F1611 on the low end with a minimum voltage of 1.8 V and exceeds the MSP430F1611 on the high end at 5.5 V, providing a full 1.9 V wider operating range. This can be beneficial for systems that are directly connected to a lithium battery, which supplies between 2.6 V and 4.2 V, depending on its state of charge. This benefit only accrues if all system components can be operated over this range, which is usually not the case.

The ATmega1281 offers 8 KB of RAM, only 2 KB less than the MSP430F1611. The memory requirements for many sensornet applications make the 4 KB available on the ATmega128L untenable. Embedded networked devices can use significant amounts of RAM to store message buffers while data collection applications can buffer sensor data in RAM for processing or prior to writing to flash. Therefore, RAM size is an important consideration for mote-class devices. With its 10 KB of RAM, the most among microcontrollers

Mfg	Device	Year	Arch	GCC (y/n)	VCC (V)	RAM (kB)	Flash (kB)	Active (mA)	Sleep (μA)	Wake (μs)	Timer (bits)	DMA (y/n)	Area (mm^2)
Atmel	ATmega128L	2002	RISC/8	yes	2.7-5.5	4	128	0.95	5	6	8	no	81
	ATmega1281	2005	RISC/8	yes	1.8-5.5	8	128	0.9	1	6	8	no	81
	ATmega2561	2005	RISC/8	yes	1.8-5.5	8	256	0.9	1	6	8	no	81
Ember	EM250	2006	XAP2b/16	no	2.1-3.6	5	128	8.5	1.5	>1000	16	yes	49
Freescale	HC05	1988	8-bit	no	3.0-5.5	0.3	0	1	1	>2000	16	no	180
	HC08	1993	8-bit	no	4.5-5.5	1	32	1	20	4	16	yes	305
	HCS08	2003	8-bit	no	2.7-5.5	4	60	7.4	1	10	16	yes	144
	MC13213	2007	HCS08	no	2.0-3.4	4	60	6.5	35	10	16	yes	81
Jennic	JN5121	2005	RISC/32	yes	2.2-3.6	96	128	4.2	5	>2500	16	yes	64
	JN5139	2007	RISC/32	yes	2.2-3.6	192	128	3.0	3.3	>2500	32	yes	64
TI	MSP430F149	2000	RISC/16	yes	1.8-3.6	2	60	0.42	1.6	6	16	no	81
	MSP430F1611	2004	RISC/16	yes	1.8-3.6	10	48	0.5	2.6	6	16	yes	81
	MSP430F2618	2007	RISC/16	yes	1.8-3.6	8	116	0.5	1.1	1	16	yes	49
	MSP430F5437	2008	RISC/16	yes	1.8-3.6	16	256	0.28	1.7	5	16	yes	196
	CC2430	2007	8051	no	2.0-3.6	8	128	5.1	0.5	4	8/16	yes	49
ZiLOG	eZ80F91	2004	ez80/16	no	3.0-3.6	8	256	50	50	3200	16	yes	169

Table 3.1: Comparison of modern microcontrollers potentially suitable for sensornet platforms. The release year provides a sense of the underlying technology trends. The processor architecture and GCC support affect the cost and complexity of the toolchain. Key design considerations include RAM and flash memory size, active current (at 3 V and 1 MHz if possible) and sleep current, wakeup time from sleep, DMA support, largest width low-power sleep timer, mechanical package, and required circuit board area. For cases in which a manufacturer offers multiple products that are very similar, this table lists those parts with the largest RAM and flash. For cases in which a microcontroller comes in many packages, this table lists only the smaller (or smallest) package.

in its size and performance class, the MSP430F1611 remains a competitive choice. And yet, despite this significant amount of RAM, it still has among the lowest of sleep currents (with RAM retention). Today, we see fewer complaints about RAM since many systems with greater RAM requirements use members of the Telos family. We do observe that some applications, like TinyDB [97], require more flash memory than the MSP430F1611 offers, and since the ATmega1281 offers 128 KB and the ATmega2561 offers 256 KB, they are better choices for applications requiring a large code footprint, but the newer MSP430 make this a moot point.

Despite the ATmega1281's many improvements over the ATmega128L, there are two important drawbacks that tipped the scale in the MSP430F1611's favor. First, the ATmega1281 low-power mode timer is only 8 bits wide, meaning it has to wakeup every 7.8 ms (using a 32 kHz clock) to service a timer overflow in sleep mode. Second, the ATmega1281 does not provide DMA support, important for collecting low-jitter samples [65] and high-throughput peripheral communications.

Today, there are many other low-power microcontroller and integrated microcontroller/radio choices available, so we briefly outline them and identify their strengths and weaknesses. The Freescale and ZiLOG microcontrollers are not supported by the GCC toolchain, making them less attractive for a research platform. In addition, the rather high active and sleep currents, long wakeup time, narrow operating voltage range, large footprint, and lack of GCC support make the ZiLOG eZ80F91 especially unattractive as a modern research platform.

Several of the microcontrollers also integrate a radio peripheral. The Ember EM250 integrates a 16-bit XAP2b microprocessor core and radio into a single chip package. An interesting feature of this product is its sleep timer which can operate from either a 32 kHz crystal or a calibrated 1 kHz clock, coupled with a prescaler (up to 2^{10} clock divider), which would let the node sleep for over 18 hours without a clock overflow. Unfortunately, a smaller RAM, higher active current, long wakeup, and uncertain GCC support make this device less appealing as a research platform.

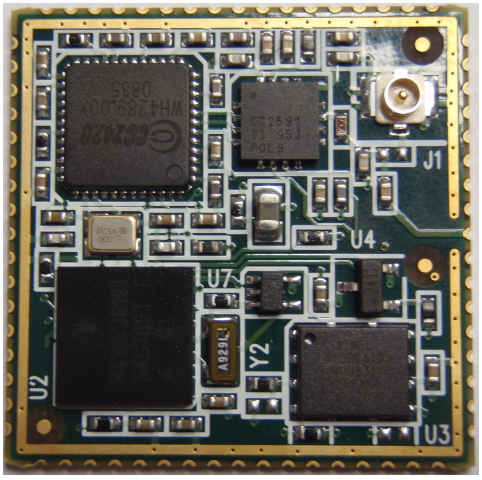


Figure 3.4: The Epic Core 2 module, currently under development in conjunction with Vectare, is fully pin-compatible with the Epic Core, allowing backward compatibility will all Epic-based platforms. The Core 2 is built around an MSP430F2618, which increases the program memory to 116 KB. The module keeps the same radio and flash as the Epic Core but adds a CC2591 RF power amplifier, increasing output power from 0 dBm to 20 dBm and improving receive sensitivity by 6 dB.

The Jennic JN5121 and JN5139 also integrate a microprocessor and radio into a single package. Their large RAM and flash sizes are attractive but they come with a high cost: a wakeup time of 2.5 ms + 1 ms/kB of program memory when entering and exiting certain sleep states. The CC2430 provides a highly-integrated microcontroller with excellent across-the-board numbers, however its major drawback is a lack of native GCC support due to its 8051-based core.

The MSP430F2618 improves upon the already excellent MSP430F1611 performance numbers, is nearly pin-compatible, and addresses the major weakness of the F1611: limited flash memory. The recently announced MSP430F5437 adds still more flash and RAM but with a slightly lower active and higher sleep current than the F2618. However, the F5437 is much larger, which makes building a similar-sized core module around it impossible. Neither the F2618 nor the F5437 were available when the Epic core was designed but had they been, we would have chosen one, especially since their flash memories can be programmed down to 2.2 V. Recent evolutionary efforts have moved in this direction, as Figure 3.4 shows.

Radio

Lacking relevant industry standards, early mote designs used a host of narrowband and wideband radios for their wireless interface. For example, designs employed radios that modulate the signal using on-off keying (OOK), amplitude shift keying (ASK), frequency-shift keying (FSK), and phase-shift keying (PSK). More recently, with widespread consensus on the IEEE 802.15.4 standard, at least at the physical layer, and a number of vendors now offering compliant radios, this choice is a natural one. The diversity in 802.15.4 radio choices, shown in Table 3.2, once again opens up the design space and warrants a reexamination of the available options with the benefit of hindsight. Although our specific design point focuses on standards-based radios, we do not believe the architectural choices would be different if a another standard (or none at all) were chosen.

Mfg	Device	Year	Wake (ms)	VCC (V)	RxSens (dBm)	TxPwr (dBm)	Rx (mA)	Tx (mA)	Sleep (μA)	FIFO (Rx/Tx)	SCLK (MHz)	SFD (y/n)	CCA (y/n)	AES (y/n)	Area (mm^2)
Atmel	RF230	2006	1.1	1.8-3.6	-101	+3	15.5	16.5	.02	128	8.0	no	no	no	25
Ember	EM260	2006	1	2.1-3.6	-99	+2.5	28	28	1.0	128	5	yes	yes	yes	36
Freescale	MC13192	2004	7-20	2.0-3.4	-92	+4	37	30	1.0	128/256	8.0	yes	yes	yes	25
	MC13202	2007	7-20	2.0-3.4	-92	+4	37	30	1.0	128/256	8.0	yes	yes	yes	25
	MC13212	2005	7-20	2.0-3.4	-92	+3	37	30	1.0	128/256	8.0	yes	yes	yes	81
Jennic	JN5121	2005	>2.5	2.2-3.6	-93	+1	38	28	<5.0	16	16.0	yes	yes	yes	64
	JN5139	2007	>2.5	2.2-3.6	-95.5	+0.5	37	37	2.8	16	16.0	yes	yes	yes	64
TI	CC2420	2003	0.58	2.1-3.6	-95	0	18.8	17.4	1	128/128	10	yes	yes	yes	49
	CC2430	2005	0.65	2.0-3.6	-92	0	17.2	17.4	0.5	128/128	4	yes	yes	yes	49
	CC2520	2008	0.50	1.8-3.8	-98	+5	18.5	25.8	.03	128/128	8.0	yes	yes	yes	25

Table 3.2: Comparison of modern IEEE 802.15.4-compatible radios. The release year provides a sense of the underlying technology trends. The wakeup time (wake) is the time required to transition the radio from sleep to listen. The receive sensitivity (RxSens) is a measure of the minimum signal strength needed for successful reception. The transmit power (TxPwr) is the output power of the radio. Rx, Tx, and Sleep are the receive, transmit, and sleep current draws. The amount of the receive and transmit data path buffering is available (FIFO). The speed of the data bus (SCLK) limits the rate of data input/output to/from the radio from the host microcontroller. The start-of-frame-delimiter (SFD) is a hardware handshake signal that toggles at a well-defined point during packet transmission or reception. The clear-channel-assessment (CCA) is a hardware handshake signal that indicates whether the channel power exceeds the clear channel threshold. The advanced encryption system (AES) indicates if encryption is support by the radio hardware.

For many systems, radio idle listening dominates the system power budget, so receive power is an obviously important metric. By this standard, the Atmel RF230 would be the best radio option since it offers the lowest receive current and best receive sensitivity. However, for CSMA systems employing low-power listening [111], the key to reducing the idle listening cost is to minimize the cost of channel polling since this time establishes the lower bound on duty cycle. The channel polling time is the sum of several factors: the startup time of the radio's crystal oscillator, the time to sample the channel for energy, and the time to convey this information to the microcontroller. Using a low-resistance crystal, the CC2420 is reported to start in 580 μs and detect channel energy in 128 μs [112]. Since the CC2420 exports the clear channel assessment (CCA) signal using a dedicated pin, this allows the host microcontroller to determine if there is channel activity without having to poll the radio over the SPI bus, reducing channel polling time. Since the CC2420 can wake up in about half the time of the RF230 and convey the channel status to the host using hardware lines, the energy cost of polling the channel is much lower on the CC2420 than the RF230.

Another advantage of the RF230 comes from its ultra-low sleep current but this logic is deceiving on two counts. The reasoning would go, since the node is asleep most of the time, sleep current matters a great deal. While this may be true in theory, in practice the constant factors dominate. First, the sleep cost must consider sleep currents aggregated over all components, and the lowest microcontroller current is 25 times larger at 500 nA. Second, for systems that operate around 1% duty cycle, but use a radio whose active current to sleep current is 10000:1 or higher like the RF230, energy consumed in the sleep state pales in comparison to energy consumed in the active state. Recent research has demonstrated radio operation at permille (0.1%) duty cycles, making sleep currents more important, but not the most important of factors.

The RF230 also offers better receive sensitivity than the CC2420 (-101 dBm vs -95 dBm) and higher transmit power (+3 dBm vs 0 dBm), so its link budget is about 9 dB higher than the CC2420. This translates to either longer-range or lower-power communications since transmit power is adjustable. Finally, a shared send/receive FIFO and the lack of hardware support for AES means this cryptographic function must occur in MCU software, rather than in optimized hardware.

Today, there are many other 802.15.4-compliant radio choices, so we briefly outline some of them and identify some of their strengths and weaknesses. The EM260 appears to offer excellent receive sensitivity and transmit power, at the expense of higher current draws and a constrained development environment. The Freescale family of radios offer an order of magnitude longer wakeup times, in the range of 7-20 ms, than the CC2420 as well as much higher current draws. The Jennic JN5121 and JN5139 are attractive because of their large RAM and 32-bit core, but their 2.5 ms minimum wakeup latency is long, and still longer if RAM retention is disabled and the program must be copied to RAM from flash on each wakeup. The CC2430 appears to be an excellent, highly-integrated system with ample RAM and flash. The only downsides are low receive sensitivity and a lack of GCC toolchain support. Finally, the CC2520 offers nearly all of the benefits of the CC2420 and RF230. If this radio had been available when the core was designed in early 2007, we would have selected it.

For these reasons, the CC2420 still provided the best overall power profile at the time of the Epic core design, cementing our decision to use it in the core module. To ensure a low radio wakeup similar to Telos, the core's radio oscillator circuit is built around a Hong Kong Xtal's C5M family 16 MHz crystal. This decision was inspired by observations that showed the benefits of choosing a crystal with a low series resistance, namely allowing the radio to start up quickly [112]. This crystal's lines are also exported using short traces to allow oscillator quick start circuits to be explored using this module [17]. If such a circuit is added, care must be taken to ensure that capacitive loading of the crystal does not exceed the manufacturer's recommended tolerances. Our evaluation of the Epic core in Section 3.4.4 shows that its wakeup performance tracks that of Telos.

The CC2420 also provides a pair of test lines, ATEST1 and ATEST2. These lines can be programmed to output a range of internal signals at various stages of the signal processing pipeline. Although normally intended for production testing, these signals can provide the low-level access needed to implement analog network coding [85] or interference cancellation [67]. The radio SPI bus, CCA, and SFD lines are also exported from the module, simplifying external probing and allowing external hardware to count both the number of times these signal are asserted as well as the amount of time they remain asserted. These are important indicators of channel activity, availability, and interference.

Flash

The core uses an AT45DB161D NOR flash [12] that provides 16 Mbit of non-volatile storage. Although this chip has a higher sleep current than the ST M25P80 [130] used in Telos, the dual RAM buffers simplify driver software and allow data to be accessed from one buffer over the SPI interface while the other buffer is busy reading from or writing to non-volatile storage.

There are two core module designs that only differ in the way the flash memory is connected to the MCU. In one configuration, the radio and flash are on the same bus (SPI0), preferable for workloads where the node is connected with another serial device, like a host computer or a sensor with an RS-232 port. In the other configuration, the flash and radio are on different buses, SPI0 and SPI1, respectively, desirable for nodes that do not use their UART, like routers in a mesh network, since resource contention will not occur and SPI bandwidth does not have to be shared.

The flash memory has a write-protect line that is exported because there is no broadly appropriate default. According to one school of thought, a "boot sector" should always be write-protected unless the module is being reprogrammed through physical connection to a programming board or host computer; however, there is no simple and fool-proof way for the module to determine this unambiguously. According to another school of thought, the default behavior of the module should be to allow the flash to be repro-

grammed in its entirety. The issue boils down to a policy decision, so in the interest of end-user flexibility, this line is neither driven nor pulled high or low – the platform developer has the option to pull-up this line by populating a resistor on the core module.

3.4.2 Implementation Decisions

This section presents several implementation choices that focus on component interconnections, I/O exports, and supply circuitry that have architectural motivations like "export everything" and "minimize constraints."

Component Interconnections and Exports

The MCU communicates with the radio using an SPI bus (USART0), receives status information (CCA, FIFO, and FIFOP) from the radio using three interrupt-capable input lines and packet transmission/reception timing (SFD) from the radio using one timer capture register, and controls and resets the radio using a pair of output lines. The MCU communicates with the flash memory using SPI on either USART0 or USART1 and communicates with the serial identifier chip using a single, interrupt-capable, GPIO line with pull-up to implement Dallas Semiconductor's 1-wire protocol. The MCU exports a byte-wide port to simplify the interface to devices with a byte-wide bus interface like NAND flash memory, FIFOs, and high-speed parallel ADCs.

In addition to the communications and control interfaces shared with the MCU, the radio also exports a wireless interface and some useful test lines. The wireless port passes through a balun and is routed to both a 50-ohm RF port on the LCC-68 module as well as a U.FL connector onboard the module circuit board. A single capacitor selects which way the RF signal goes – LCC-68 pad or U.FL connector. This flexibility allows developers to choose either an external antenna with a U.FL-terminated pigtail – now common because of 802.11 b/g radios – or a board-integrated antenna like a chip antenna or a planar-inverted F-antenna (PIFA). The first choice eliminates low-level RF engineering while the second choice allows for a more compact solution.

Power, Ground, and References

The core exports four different power supply lines for the four major power domains: DVDD supplies the microcontroller core and serial identifier, AVDD supplies the ADC core and reference, RVDD supplies the radio, and FVDD supplies the external flash memory. These signals may be tied together externally, connected to different supplies with slightly different voltages, or individually passed through current sense resistors to allow current profiling per power domain. All of the supply lines are internally decoupled using 0.1 μF capacitors. If long external power traces are used, larger external capacitors should be used. The core also exports several references used by the ADC. The VREF+ line allows the internal ADC reference to be used by external circuitry (with appropriate buffering). The VeREF+ and VeREF- lines allow externally-generated high and low references to be used by the ADC.

In addition to the four supply lines, the core exports four different ground lines. Although three of these ground lines are internally connected, they individually provide the preferential ground return for the microcontroller, radio, and flash memory. The fourth ground line, AGND, connects to an isolated ground plane section and provides the return for the analog section of the microcontroller. The AGND can be connected to the digital grounds externally, but care must be taken to reduce digital noise from coupling

with AGND. Finally, the radio ground is divided into a digital section and an analog section with a separate ground, RFGND. The radio digital section shares a common ground with the microcontroller and flash while RFGND provides the return for the RF path. The point where the RFGND lines are exported from the module is the only place where the analog and digital grounds are connected together – the proverbial "ground mecca" – situated on the ground ring along the module perimeter, providing a convenient solder point for an RF shield.

3.4.3 Mechanical Design

A question that every module designer must confront at some point is *what form factor and connector interface should the module use?* There are nearly as many different answers to this question as there are mote platforms. The Epic core module uses an industry-standard LCC-68 (68-pin leadless chip carrier) form factor that places all parts on one side of the module circuit board and exposes nearly every signal that might possibly be useful along the board edge via perimeter pads. This packaging wastes no connector space since the board edge is otherwise unused, allows a seamless transition from prototype to production since modules can socketed, hand-soldered, or machine-assembled, and a single-sided board makes signal probing easy.

Several considerations played a role in the choice of perimeter pads. First, since the package is leadless, no costs are incurred on connectors. Second, since the package land pattern is essentially JEDEC-compliant (except for pin numbering), an off-the-shelf prototype or production socket can be used to program the device or break out the signal lines for debugging. Third, since the 68 pads around the module perimeter are actually plated-through semi-holes (also known as castellations or routed vias), they are easy to solder by hand which greatly simplifies prototyping. Fourth, since the plated-through semi-holes are concave, an oscilloscope or voltmeter probe tip rests easily in them, making debugging just a bit easier. Fifth, since the plated-through semi-holes are actually vias that connect all layers of the circuit board, they reduce the number of vias that might otherwise be necessary, potentially reducing cost and providing more circuit board real estate.

Superficially, the Epic core's LCC-68 footprint might seem similar to the the MICAz [31] and IRIS [29] OEM modules or the Tmote Mini [119], but there are some important differences that make Epic well-suited to pilot studies: it can be hand-soldered when others cannot, it has a wide interface that exports nearly every internal signal, and it can be socketed. This design consideration raises an important architectural question: *should the number of pins a module exports grow linearly with its area or as the square root?* A ball grid array (BGA) allows a linear relationship between area and pin count while the perimeter pins of a leadless chip carrier (LCC) grows as the square root of the area. We chose an LCC-68 package with plated-through semi-holes to allow hand assembly, but a side-effect of the decision is that modules are more limited in their I/O width. We also experimented with different module thicknesses and found that an 0.5 mm board was too flimsy (without a structural shield) and that the standard 1/16 in circuit board was unnecessarily thick, so we settled on a module thickness of 1.0 mm.

Other mote designs, like the MICA family including the MICA, MICA2, and MICAz often waste circuit board real-estate unnecessarily making them too large to comfortably design into enclosures, require expensive and fragile connectors, and do not export many I/O lines useful for research and experimentation. The highly-integrated Telos suffers from many of these same problems. The MICA2Dot [30] is more space-optimized and integrates the core pieces better, buts its limited I/O lines reduce choice, its connector is difficult to attach, its "crown-of-thorns" pin array is expensive to machine assemble, and its antenna connector is poorly matched.

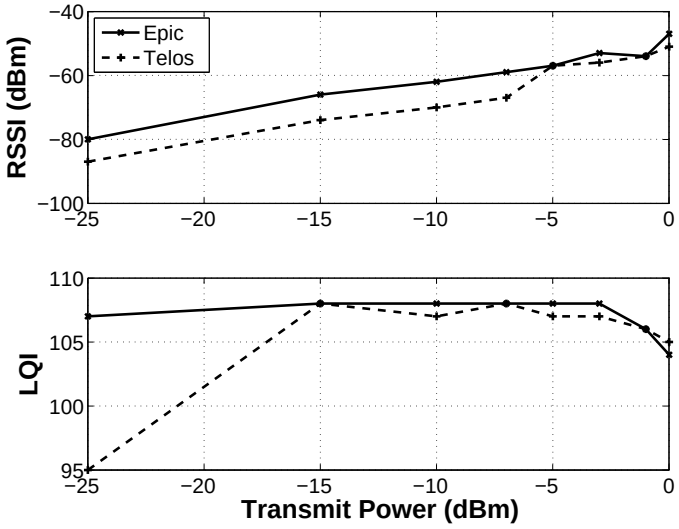


Figure 3.5: Radio reception performance (RSSI and LQI) of Epic and Tmote Sky over the same channel as the transmit power is swept from -25 to 0 dBm.

3.4.4 Performance Microbenchmarks

Modules will only be adopted if their performance is at most ϵ-suboptimal to other alternatives, and we show here that epic compares favorably to earlier work. One of the key metrics for a platform is the radio wakeup time. We measured the wakeup time of both Epic and Telos by monitoring the state of the CC2420's CCA pin in the same way that the TinyOS 1.x and 2.x stacks use to determine when the oscillator has stabilized. In our experiments, Epic wakes up in $629 \pm 3\mu$s while the Telos wakes up in $619 \pm 3\mu$s (95% confidence).

Sleep current is another important performance metric which for Epic is 7 μA at 3 V. In comparison, we measured the Telos sleep current to be 6 μA at 3 V when running the TinyOS `Null` application. Although the Epic sleep current is comparable to Telos, the constituent currents are different: most of the Epic current draw comes from the flash chip while most of the Telos current draw comes from its host interface, which Epic removes for reasons of generality.

To evaluate radio reception, a transmitter node (Telos B) is positioned 3 m from a fixed antenna. In the first experiment, a Sentilla Tmote Sky [120] is connected to the antenna. In the second experiment, an Epic is connected to the same antenna. During each experiment, 20 packets are transmitted from the sender to the receiver. The received signal strength indicator (RSSI) and link quality indicator (LQI) are logged. This experiment is repeated at eight different power levels. These results, along with tests over a range of channels and distances, confirm that the RF performance of Epic is commensurate with a mature commercial system.

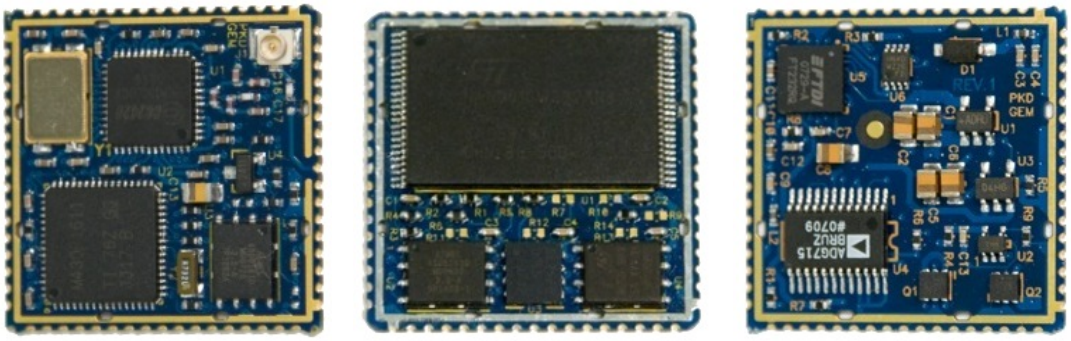
As a cautionary note, we point out that achieving this performance required months of design, evaluation, tuning, and redesign. This work was carried out using expensive test and measurement equipment including high-speed digital oscilloscopes, spectrum analyzers, and network analyzers. In the final analysis, ten different RF section layouts, three different inductor choices, and two different RF ports were evaluated. All of our designs are open-sourced and available online.

3.4.5 Future Directions

In hindsight, the choice of the MSP430F1611 microcontroller and CC2420 radio have stood the test of time, and product line extensions like the MSP430F26x, MSP430F54x, and CC2520 promised a simple migration pathway forward. Indeed, the obvious next-generation core module – an evolutionary one – that integrates these much improved but is still fully backward-compatible has been designed and is now under active development. Going forward, this path will allow the community to leverage existing investments in software yet allow new research efforts by moving more functionality into the radio, and making the processor-radio interface richer and more flexible. At the same time, new products from other vendors are quickly closing, or have already closed, the gap in wakeup latency, RAM size, low-power timer support, direct memory access, and operating voltage range. Perhaps the most important developments are the newly-announced, but not yet shipping, products tht integrate an 802.15.4-compliant radio as a memory-mapped peripheral in the same package as a high-performance ARM processor.

3.5 Expert Peripheral Modules

Complementing the core module are a family of peripheral modules that provide specific functions, such as power supply conditioning, high speed host communication interfaces, bulk storage, or analog signal conditioning. Figure 3.6 shows the modules currently in the Epic family.



(a) Core (b) Storage (c) USB

Figure 3.6: Epic Core, Storage, and USB modules. All modules have a one square inch form factor that is compatible with an LCC-68 footprint.

Since a key aspect of the architectural approach is a systematic partioning of functionality between modules and carriers, we identify four cases when modules make sense: when their design requires *deep expertise*, when their assembly or tuning requires *specialized equipment*, when their function is so common that *reuse in modular form* is inevitable, and when it is *simply more convenient* to group a set of related components. Collectively, these principles provide some guidance for partitioning functionality between modules and carriers and they address the question: *where do modules come from?* Following these heuristics, Table 3.3 traces the genesis of the modules currently in the Epic family, and the remainder of this section discusses their functions.

Module	Deep Expertise	Special Equip.	Modular Reuse	Simple Convenience
Core	yes	yes	yes	no
USB	no	no	yes	yes
Storage	no	yes	no	yes

Table 3.3: The genesis of core and peripheral modules in the current Epic family. Modularizing a component is beneficial if it demands deep expertise to design, requires specialized equipment to assemble or tune, is general enough that reuse in modular form is inevitable, or just as a way to group together related parts into a subsystem as a matter of simple convenience.

3.5.1 USB/Power Module

The USB module provides four functions: host interface, reprogramming, JTAG over USB (requires additional host software), and battery charging and management. The first three offer the same functionality as the Telos [112] in that the host interface and reprogramming functions are multiplexed using the same I/O lines and JTAG over USB is possible (but not supported). The battery charging and management can recharge a Lithium battery whenever the module is plugged into a USB port and arbitrate between USB power and an attached Lithium (or alkaline) battery. This module was built because it was perceived to be quite useful to a number of platforms in modular form and was a convenient container for related functionality.

3.5.2 Bulk Storage Module

The storage module integrates four different non-volatile memory chips – a 1 Gbit NAND flash, two 16 Mbit NOR flashes, and one 512 Kbit FRAM. These memory chips have very different read, write, and erase characteristics and so they represent a useful collection of chips integrated on a single module for simple convenience when researching storage systems. Additionally, some of the included flash chips are in packages that are either leadless or with extremely small pitch, making them difficult to hand solder and warranting specialized manufacturing equipment. This module was built both for experimentation and as a storage subsystem for motes.

3.6 Expert Circuit Subsystems: Implementing the MOV Metric

There are many forms of the MOV metric and in this section we describe the circuits and devices used to implement shock, vibration, acceleration, gestures, displacement, and activity recognition. The key constraint is that the MOV cure must not be worse than the mobility symptom. In other words, the benefits that accrue from having knowledge of motion must be greater than the power cost to deliver this knowledge. In this section, we show how some members of the MOV family of metrics can be provided for as little as 1.2 μA while others can cost ten to twenty times that figure. However, based on technology trends and consumer market forces, this gap is closing, and within a year or two, silicon accelerometers are expected to provide the MOV metric in all its various forms at near-nanopower budgets.

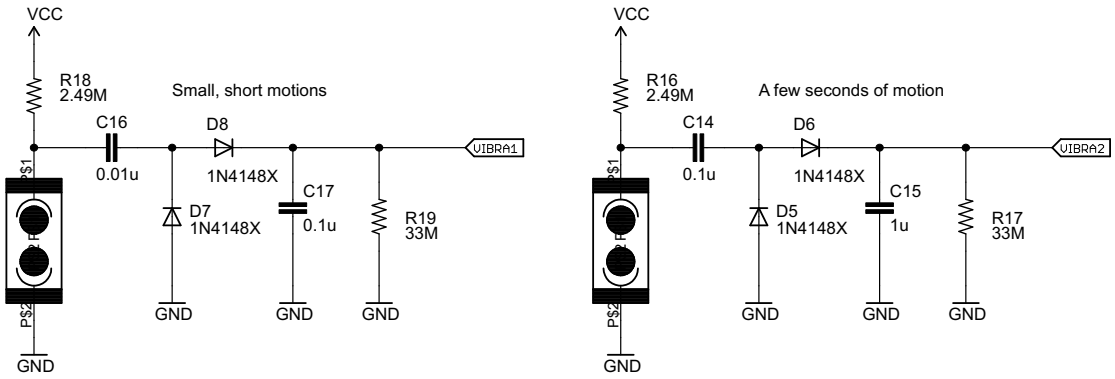


Figure 3.7: Shock and vibration sensing circuits using a vibration switch, AC-coupled one-shot, and integrator. The VIBRA1 and VIBRA2 outputs trigger after minor shock/bumps and prolonged vibration/motion. This circuit draws approximately 1.2 μA at 3 V during quiescent sensing.

3.6.1 Detecting Shock and Vibration

To detect shock (bumps) and vibration, we use the vibration dosimeter shown in Figure 3.7. The sensor is an omni-directional vibration switch [123] that is nominally closed at rest but chatters open and closed in response to movement [121]. The switch is connected to ground on one terminal and in series with a pullup resistor to power. The 2.49 MΩ pullup resistor sets the quiescent current draw of the circuit. At rest, the circuit draws 1.2 μA at 3 V. A capacitor AC-couples the output of the sensor, a first diode steers negative voltage transients to ground, and a second diode steers positive transients to a capacitor that integrates these signals. A resistor in parallel with the integration capacitor slowly discharges the capacitor so that in the absence of motion, the capacitor voltage goes to zero. The choice of capacitors allows us to set the time constant of these two circuits from near-instant reaction to several seconds of delay [122].

3.6.2 Detecting Acceleration, Gestures, and Displacement

To detect acceleration, gestures, and displacement, we use the Analog Devices ADXL345 accelerometer [8]. This sensor provides a digital interface, low-power operation down to 25 μA at 3 V at a 25 Hz data rate, and built-in threshold and gesture detection. The output of the accelerometer can be buffered in a 32-element FIFO that can be operated in a variety of modes including bypass, FIFO (tail drop), stream (head drop), and trigger (store 32 samples after trigger event). A simple displacement-after-trigger mode can be implemented draining the FIFO after a trigger and summing the elements. Newer accelerometers push the envelope even further. The ST Microelectronics LIS331HH [129] offers a 10 Hz data rate at 10 μA and with 6-bits resolution, showing that digital accelerometers are closing the gap with analog circuits.

Figure 3.9 shows the shock and vibration motion detector circuit in operation overlaid with acceleration samples from a conventional accelerometer. Tri-axial acceleration samples taken at 200 Hz are shown with their bias removed and amplitude scaled. The output of the motion detection wakeup circuit can been seen as a pulse that alternates between zero and one as the sensor transitions from rest to motion. At time $t = 0.5$ s, a node is picked up and moved and at time $t = 1.33$ s, the motion detector circuit wakeup triggers, waking up the sleeping microcontroller using an interrupt line. At time $t = 3.09$ s, the node stops moving and time $t = 4.3$ s, the motion detector output indicates movement has stopped. This process repeats at time t =7.5 s.

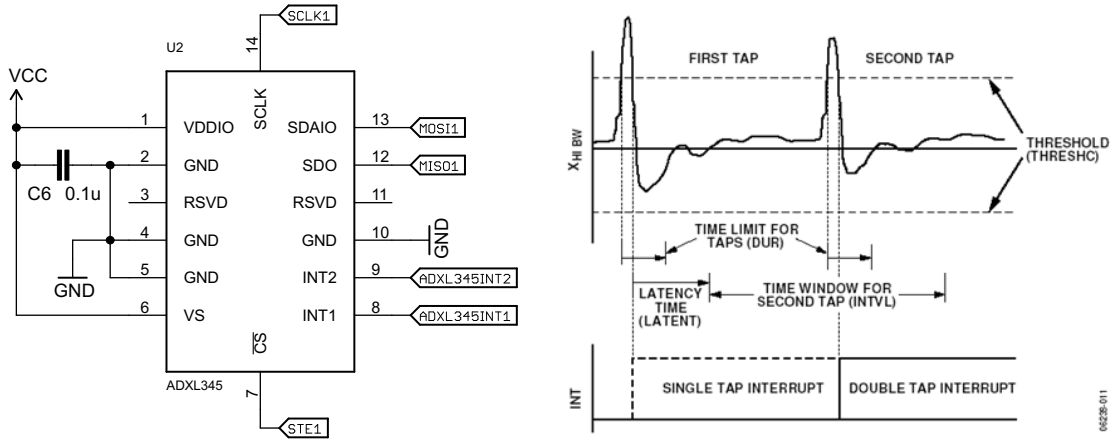


Figure 3.8: Acceleration, gesture detection, and displacement monitoring with the ADXL345. This circuit draws approximately 25 μA at 3 V during quiescent sensing at 25 Hz (left). The sensor can trigger on simple threshold excursions or more complex gestures with programmed acceleration thresholds and time components (right, image credit: Analog Devices, Inc.).

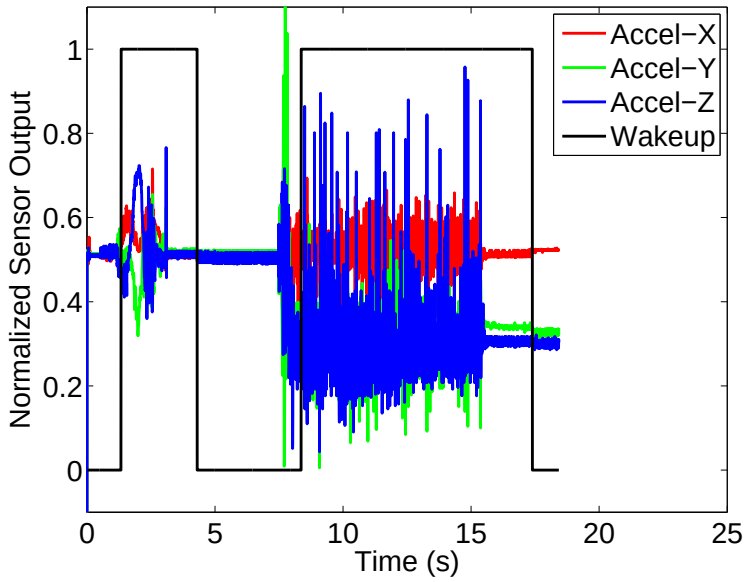


Figure 3.9: The shock/vibration motion detection circuit in operation overlaid with acceleration readings. Acceleration bias is removed and the readings are scaled.

3.6.3 Detecting General Activity Recognition

To detect general motion and generate an activity coefficient, we use the circuit shown in Figure 3.10, inspired by recent work on body-worn sensors for personalized building comfort control [60]. This circuit uses a piezo sensing element [105] that is typically used as a 3-axial hard disk shock sensor. The output of

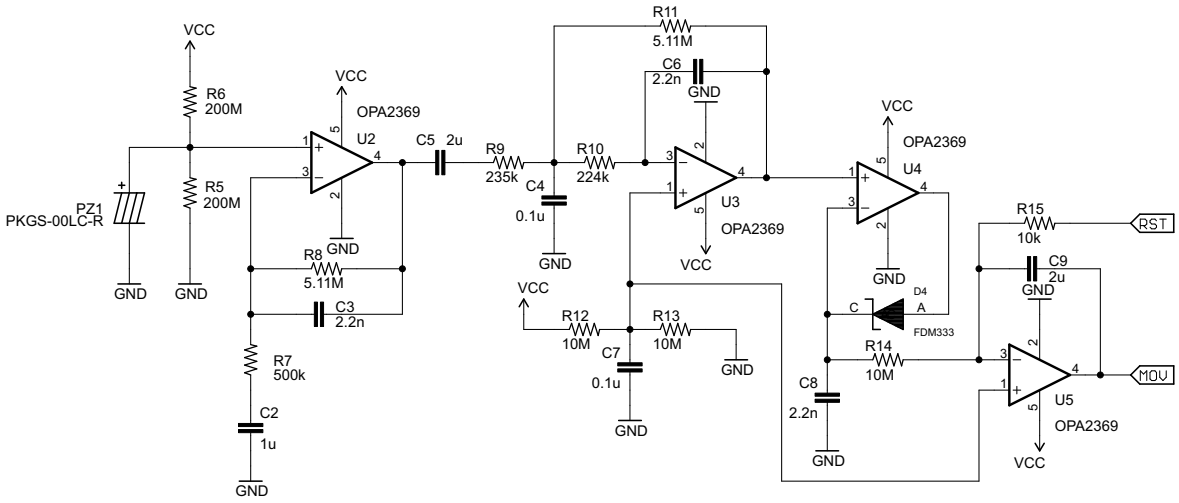


Figure 3.10: General activity detection using a piezo sensor and ultra low-power filtering, amplification, peak detection, and integration. The circuit draws approximately 3μA at 3 V.

the piezo sensor is low-pass filtered and amplified, and the resulting signal is fed to a peak detector whose output is integrated, but can also be reset. The circuit draws approximately 3μA at 3 V and offers an output that is roughly proportional to level of recent movement.

3.7 Expert Circuit Subsystems: Measuring Nodal Energy Consumption

Mobile sensors exhibit widely varying energy consumption due to their unpredictable link volatily and intermittent connectivity. To adapt to the varying workload, it is useful for nodes to be able to introspect their own energy consumption. In this section, we present iCount, a new energy meter design that can provide energy meter for free. For many systems that have a built-in switching regulator, adding a single wire between the regulator and the microcontroller enables real-time energy metering. iCount measures energy usage by counting the switching cycles of the regulator. We show that the relationship between load current and switching frequency is quite linear and demonstrate that this simple design can be applied to a variety of regulators. Our particular implementation exhibits a maximum error of less than $\pm$20% over five decades of current draw, a resolution exceeding 1 μJ, a read latency of 15 μs, and a power overhead that ranges from 1% when the node is in standby to 0.01% when the node is active, for a typical workload. The basic iCount design requires only a pulse frequency modulated switching regulator and a microcontroller with an externally-clocked counter. Additional details about this design are available elsewhere [45].

The iCount design is motivated by the simple observation that many switchers exhibit a nearly linear relationship between switching frequency and load current over a wide dynamic range. Figure 3.11 shows how the switching frequency changes with load current for several different commercial switching regulators. The basic iCount design follows directly from the data in Figure 3.11. Counting the switching cycles of the switching regulation signal is all that is required to accumulate energy usage. Since many battery-operated systems include a microcontroller, and since most microcontrollers support counters that can be externally-clocked, simply adding a single wire between the switcher and a the microcontroller's counter

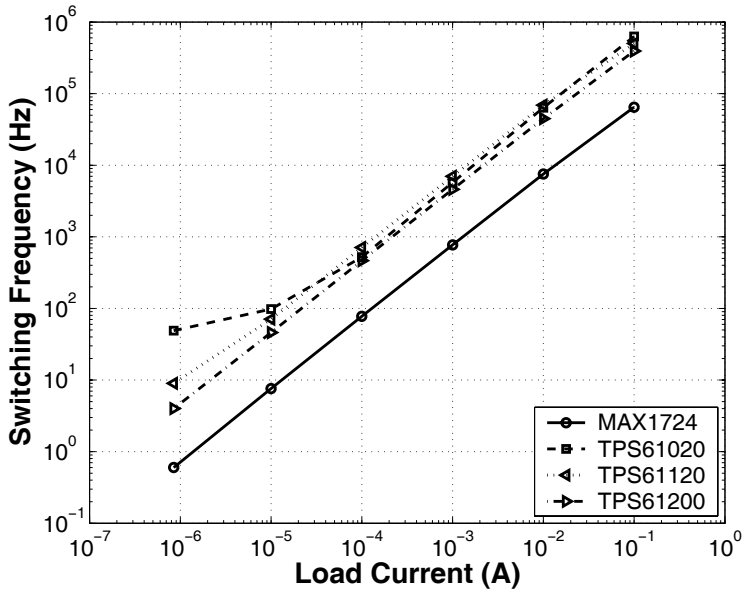


Figure 3.11: The relationship between load current and switching frequency for several switchers, after bias compensation. Some switchers are more linear than others (and sometimes over a wider dynamic range). Plotted on a log-log scale.

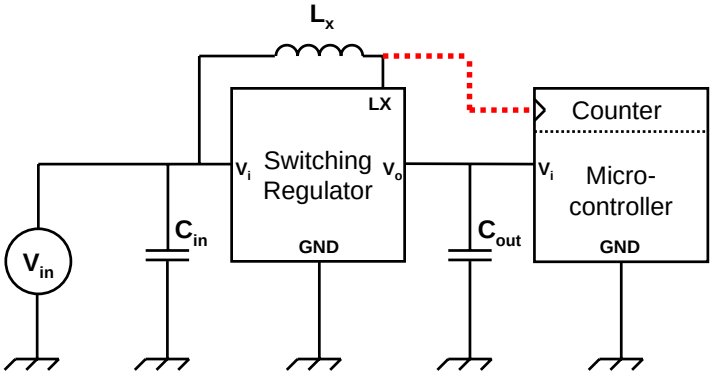


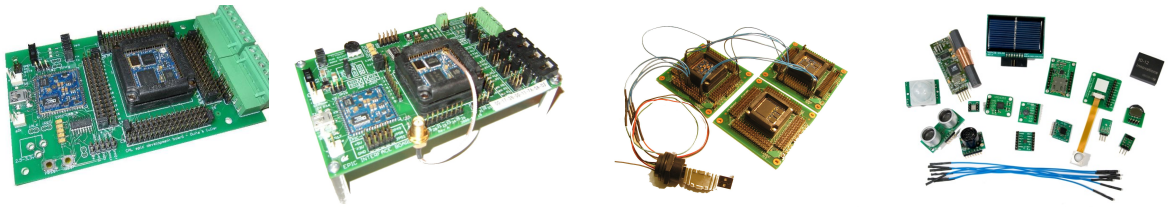
Figure 3.12: A typical circuit with a switching regulator and microcontroller. Adding a single wire (the dashed line) enables real-time energy metering.

enables real-time energy metering. Figure 3.12 shows the needed circuit.

Since switching regulators are common in many battery-operated systems, and iCount requires no additional hardware beyond the existing regulator and a spare microcontroller counter, this approach or its slight adaptations should be simple to incorporate into a range of platforms. iCount should simplify hardware power profiling and enable applications to introspect their own energy consumption in real-time and with low overhead, providing the metering framework for software energy profiling and runtime adaptation.

3.8 Prototyping

In our vision for prototyping, platform developers are able to pick a handful of components including sensors, motes, storage, battery packs, and solar harvesting modules, and literally wire them together in whatever way is most appropriate for a given study. Writing the corresponding system software would follow a similar pattern; most components would have associated drivers that could simply be declared and wired to the hardware resources they use, like GPIO lines, ADC channels, or an SPI bus. We envision the emergence of platform construction kits that include an assortment of building blocks, their associated driver software, and the glue to assemble a wide variety of prototype nodes. In this section, we examine how Epic supports prototyping approaches for both novice and advanced system designers. We used this prototyping support during the early design stages of all non-trivial platforms developed for this dissertation.



(a) Development Board (b) Interface Board (c) Breakout Board (d) COTS Parts

Figure 3.13: The Epic family includes hardware specifically designed for (a) making platform prototyping possible in a classroom setting by novice designers (b) interfacing with the popular Phidgets analog and digital sensors (c) empowering module designers to construct, probe, and debug intricate circuits on-the-fly, both only using (d) off-the-shelf parts such as jumpers, sensors, solar power packs, and surfboards.

3.8.1 Try It And See with Agile Platform Prototyping

Many projects begins with experimentation and rapid prototyping inspired by a "try it and see" attitude. The goal is to demonstrate a basic implementation that showcases an important capability, enables some exploratory data to be collected, or reduces perceived implementation risk through an existence proof. At this stage of the game, maximum impact demands a narrow focus on the essential elements of the system, but the other parts must be good enough to evaluate the prototype. The metric of merit is time-to-result.

Unfortunately, several factors increase time-to-result. Issues like sensor and power supply selection, electrical wiring, and device driver development dominate engineering efforts while more novel aspects like application software, performance characterization, and end-user data collection are routinely back burnered during the initial stages. To improve productivity, we created a Development Board that can be easily and inexpensively integrated with off-the-shelf sensors, displays, and solar packs to improve time-to-result.

Figure 3.13(a) shows the Development Board, which benefits from the choice of an industry-standard LCC-68 footprint by including an off-the-shelf socket for easily swapping modules. Adhering to the principle that all signals should be available to the platform designer, breakout pins allow access to every signal, simple shorting shunts allow each signal to be individually connected to power or ground, and jumper wires allow a signal to be easily connected to off-the-shelf parts like the ones shown in Figure 3.13(d).

The Development Board also incorporates a USB module for programming, alkaline and lithium battery connections for supplying power, and LEDs and buttons for feedback, debugging, and control. This

flexible platform enables quick prototyping and exploration of novel development elements while circumventing the complexities of module and carrier design. The board has already been used by undergraduate students to develop application-specific platforms and a second version, shown in Figure 3.13(b), was used to teach a summer school on wireless embedded systems.

3.8.2 Debugging

Debugging is an often frustrating aspect of prototyping and pilot development. Effective debugging requires the developer to probe signal voltages to verify circuit operation and measure currents to identify unexpected draws and verify expected ones. Unfortunately, many systems can make probing signals and debugging painfully difficult: signals are buried under chips, routed through to intermediate layers of the printed circuit board, and never exposed through any header. Measuring currents can be still more challenging since it requires breaking a circuit to take the measurement. In most systems, directly measuring the individual draws of the microcontroller, radio, flash, or other peripherals is impossible since the individual power supply lines are buried in the circuit board and a single, global power supply line is exposed. The result is that developers must write test code that isolates different functions, rather than being able to directly observe the system running application code.

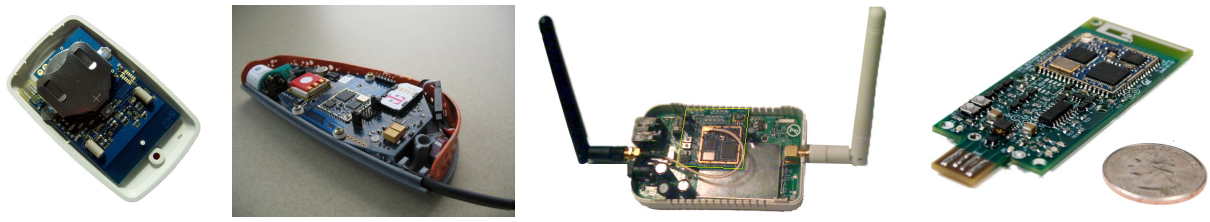
To address these challenges of hardware debugging, we developed a breakout board, shown in Figure 3.13(c), that includes an LCC-68 socket, pins for easily accessing and jumpering each signal, and an Epic programming port. With access to the full array of signals, hardware developers can easily probe every point in a design, connecting the circuit, multimeters, oscilloscopes, and other monitoring equipment as they see fit.

3.9 Carrier Board Case Studies

Carriers are circuit boards that act as substrates to glue together one or more general-purpose modules with application-specific *sensors*, *power supplies*, and *mechanical constraints*. To evaluate the utility of our proposed architecture, we designed and implemented several different pilot-stage carrier boards. These case studies illustrate how our decomposition allows new platforms to be designed quickly (usually in a few days), fabricated inexpensively on typically two-sided circuit boards (for a few hundred dollars), and easily hand-assembled (in hours, by graduate students). Table 3.4 summarizes these carriers and their differences and Figure 3.14 shows their pictures.

Carrier	Modules	Sensors	Power	Mechanical
Irene	Core	A, L, V	button cell	off-the-shelf enclosure
Badge	Core	C, G, H, L, N, O, P, T	Li+ poly pack	custom enclosure
Router	Core	n/a	AC adapter	Open-Mesh connector
BenchMark	Core, USB	T, H, L	USB, AA pack	Telos-like

Table 3.4: Despite their unique application requirements, all carriers incorporate the same mote core. Sensors: acceleration (A), carbon monoxide (C), GPS (G), humidity (H), light (L), nitrogen oxides (N), ozone (O), position/orientation (P), temperature (T), and vibration (V).



(a) Irene Mote (b) Common Sense Badge (c) IPv6/LoWPAN Router (d) BenchMark Board

Figure 3.14: Platforms for different applications have been built to both evaluate the Epic architecture and provide concrete design points to evaluate our low-power mobile sensing architecture: (a) the Irene mote incorporates multiple motion sensors and a rechargeable coin cell battery in an enclosure that can be wrist or lanyard worn, (b) the Common Sense Badge integrates a wide range of air quality and environmental sensors with a rechargeable lithium battery and a custom handheld enclosure, (c) an IPv6/LoWPAN router that connects 802.3 and 802.11-based IP networks to 6LoWPAN-based sensor networks, and (d) a platform for sensornet testbeds with a USB interface, application energy metering, and a FIFO buffer for collecting and streaming high-frequency data. Each platform was designed in days or weeks using the same generalized core module while satisfying the specific requirements of the application. Photo credits: (b) Mazzarello Media and Arts.

3.10 Discussion

Component reuse is a basic aspect of the building block approach to platform construction and carriers are no exception. The motivation for reuse comes from a desire to preserve the accumulated learnings and artifacts in moving through the phases of development, but this section also traces our experience with unplanned, organic reuse at the level of CAD parts and schematics.

We demonstrated the viability of this approach by building a handful of application-specific carrier boards from a collection of modules but, in the process, we discovered two curious things. First, reuse occurs at the CAD parts, schematic, and parts inventory level as well as at the module level. Designers use parts and circuits created by their colleagues or stocked in the lab rather than create new CAD parts themselves or choose parts that must be ordered from distributors. This suggests that we should encourage greater reuse by sharing our niche part libraries more broadly and creating platform development kits that bundle many of these common pieces.

A second observation is that there is little overlap in electronic parts between modules and carriers. Even discrete parts like 10 kΩ pull-up resistors or 0.1 μF decoupling capacitors are different. The module designs, driven by space constraints and anticipating machine assembly (of the modules themselves but not necessarily the carriers), use smaller surface mount parts (e.g. 0402). The carrier board designs, constrained far less by space and anticipating hand assembly (at least for pilot runs) use larger surface mount parts (e.g. 0603 or 0805). This limited overlap in part usage provides some evidence that our modularity hits a design sweet spot; modules and carriers appear well-optimized for their particular purpose. Indeed, the first article of every carrier board presented in this chapter was hand-assembled while almost exactly the opposite is true for the modules.

The development of many systems proceeds through the familiar phases of prototype, pilot, and production and while the engineering activities undertaken in each phase are very different, accruing the experiences and intellectual property through the phases is important. The modular architecture proposed in this

chapter supports such a fluid development model and we believe this approach to sensornet platform design is the first to support all three phases of sensornet development well enough for rapid progress.

3.11 Summary

This chapter argues for a building block approach to hardware platform design that partitions functionality between general-purpose modules and application-specific carriers. A key principle of this approach is for modules to export as wide an electrical interface as possible rather than a narrowly-defined system bus. Lowering the hardware abstraction level "below the bus" facilitates greater module reuse, more compact designs, increased integration simplicity, and lower overall part count. And, by supporting many physical interconnect options for modules including socketing, soldering, and hardware inlining, this approach supports prototype, pilot, and production system development well enough for rapid progress. An important benefit of decomposing platforms in this way is that modules capture working hardware designs, making hardware libraries a natural extension. In the future, we envision others will create many new modules – like solar harvesting, signal conditioning, or high-precision clocks – and share them broadly to support rapid forward going innovation.

Chapter 4

Asynchronous Neighbor Discovery

The asynchronous neighbor discovery problem is important for all mobile networks but it presents a particular challenge for micropower systems: how can two or more nodes operate their radios at low duty cycles (e.g. 1%) and yet still discover and communicate with one another during infrequent, opportunistic encounters without requiring any prior synchronization information? This chapter presents Disco, an asynchronous neighbor discovery and rendezvous protocol that allows nodes to operate the radio at a low duty cycle but still ensure that discovery is fast, reliable, and predictable over a range of operating conditions. Disco translates a target duty cycle into a pair (or triplet) of prime numbers that are then used to schedule the radio. This protocol ensures that two nodes will have overlapping radio on-time within a bounded number of periods, even if nodes independently set their duty cycles. Once a neighbor is discovered, and its wakeup schedule known, rendezvous is a matter of scheduling a wakeup during the neighbor's subsequent wakeup.

4.1 Overview

The interaction of things – energy-constrained mobile objects with other mobile and static objects – provides a fertile ground for application-driven research. However, the low-power, asynchronous neighbor discovery problem poses one challenge for these applications: how can two systems that are awake infrequently and perhaps rarely co-located discover each other without any prior knowledge of their potential encounters, and without external assistance? The key issue is that nodes must operate their radios at low duty cycles to maximize lifetime, and yet be actively vigilant to detect the emergence of new links and the attrition of old ones. These two requirements – low-power operation and active vigilance – are at odds with each other since optimizing for one may come at the expense of the other.

This chapter presents *Disco*, a practical solution to the asynchronous neighbor discovery and rendezvous problem that works by scheduling radio wake times at multiples of prime numbers, ensuring deterministic pairwise discovery and rendezvous latencies without requiring global coordination of duty cycles or a superframe structure. The algorithm selects a pair of prime numbers such that the sum of their reciprocals is equal to an application's desired radio duty cycle. Each node increments a local counter with a globally agreed-upon period and, if this local counter is divisible by either of the primes, the node turns on its radio for one counter period. This protocol, a simple adaptation of Sun Zi's two-millenia old Chinese Remainder Theorem [107], ensures discovery in bounded time, even if nodes independently set their own duty cycle. Section 4.3 presents the Disco design starting with a simplified version of the algorithm to show correctness, and then relaxes the simplifying assumptions to flesh out a protocol that works in practice.

To use Disco, an application first chooses a desired duty cycle or discovery latency, and optionally a node class (nodes in the same class do not need to minimize discovery latency between themselves). Disco automatically selects primes that match the desired duty cycle or discovery latency and then turns on the radio at every multiple of the chosen primes. During each such wakeup, a node can listen, beacon, or do both, depending on application requirements. Section 4.4 presents the details of our Disco implementation.

Disco performs well on key performance metrics like discovery latency and rendezvous frequency as a function of duty cycle. Disco also offers great flexibility for applications: nodes can independently select their own duty cycles and still ensure discovery, or nodes can be assigned to different classes such that inter-class discovery times are guaranteed to be much faster than without class assignments, or nodes can adjust duty cycles to achieve a particular discovery latency, or nodes can choose to beacon, listen, or do both, during their wake times. Disco can also regulate discovery in response to mobility, by adapting behavior based on the MOV metric, for example by increasing or decreasing the discovery rate or duty cycle. Such mobility-aware regulation of discovery can greatly improve system lifetime by constraining rapid discovery to times during or after high mobility, while still supporting lower discovery rates and power draws at other times. Section 4.5 compares Disco's performance with earlier work and sensitivity to several parameters through a simulation study and Section 4.6 presents empirical data collected from our implementation.

The flexibility afforded by Disco is motivated by the different needs, duty cycles, and interaction patterns of emerging applications. For example, in some applications, nodes must exchange data (talking) with one another during infrequent, unpredictable, and opportunistic encounters (docking). In other applications, a group of nodes must establish membership and maintain connectivity as the group moves together, even as individual members churn and the group diameter changes (flocking). And, in some cases, static nodes must maintain connectivity in the midst of an adversarial radio environment.

4.2 Related Work

The asynchronous neighbor discovery problem has been explored in many prior contexts. For wall-powered or rechargeable nodes, the neighbor discovery problem has a simple solution: a node periodically beacons its presence and any always-on neighbor that receives the beacon considers the first node to be a neighbor. The problem is also simple with some external assistance: a node attempts to discover and join a network just after being powered on or reset by a human operator or just after being initially deployed [101]. Discovery is also aided greatly by external synchronization: a node only beacons and listens for neighbors for just a brief period after each minute, quarter-hour, or hour, for example, if nodes can synchronize their clocks using GPS [93]. The problem is also simple if the nodes are deployed in a static network and expect to have one or more neighbors at all times: nodes maintain time synchronization [138], send packets with long preambles [111], or repeatedly send the same packet until it is acknowledged [19].

These simple techniques work because encounters are predictable: a sender has a reasonable expectation that the receiver is nearby, that its duty cycle is known, and that it will be awake soon. The discovery problem becomes more challenging in energy-constrained, mobile environments since a node may not know whether any neighbors are present, and what duty cycles those neighbors might be operate at currently, given the widely varying energy availability and usage observed in practice for both mobile [126] and static nodes [133]. Since idle listening often dominates the system power budget [44], the most expedient – perhaps only – way to balance the power supply and demand is to reduce the listen duty cycle. Another reason asymmetric duty cycles are useful is that they allow nodes with different roles or capabilities, like cattle collars and static data sinks [144], to interact.

Once listen periods must be adjusted to reflect the available energy or differing workloads, sampling protocols that employ low-power listening and assume a fixed listen period, like B-MAC [111] and X-MAC [19], become less appealing for neighbor discovery. This occurs because the required preamble length to ensure discovery is no longer a network-wide constant. Likewise, slotted protocols like S-MAC [148] that periodically listen for a whole synchronization period to discover neighbors [91] assume global agreement on the length of this period, and therefore on the minimum duty cycle. In contrast, Disco does not require such global agreement on duty cycle, instead allowing each node to independently choose a duty cycle.

Prior work in asynchronous neighbor discovery has employed stochastic, quorum, and combinatorial techniques. McGlynn and Borbash proposed "birthday protocols" for asynchronous neighbor discovery in static ad hoc networks [101]. They considered the problems of energy conservation during node deployment and energy-efficient neighbor discovery following deployment using a scheme in which nodes listen, transmit, or sleep with different probabilities. Their work concludes that mobile ad hoc networks would not use discovery, but that discovery would be useful in static ad hoc networks. Even in static networks, however, they schedule an explicit discovery phase that quickly terminates. Our proposal differs in a few key ways. First, we note that discovery can be quite valuable in mobile networks when the nodes move slowly or have modest dwell times near peers. Second, we suggest that discovery should be a fundamental and continuous service in both mobile *and* static networks, rather than a one-time event. Finally, we note that probabilistic discovery leads to aperiodic and unpredictable rendezvous latencies, and long tails on discovery probabilities, reducing its appeal.

Tseng et al. propose a quorum-based protocol for multihop ad hoc networks [139]. Their protocol divides time into a sequence of beacon intervals which are grouped into sets of m^2 contiguous intervals, where m is a global parameter. In each group, the m^2 intervals are arranged as a two-dimensional $m \times m$ array in a row major manner. A node arbitrarily picks one column and one row of entries to transmit and receive, respectively, for a total of $2m - 1$ intervals, in each group of m^2 intervals. Since m is a global parameter, all nodes use the same duty cycle, which limits flexibility. Jiang et al. generalize this result to any quorum protocol that satisfies a rotation closure property [82]. We note that both sampling protocols, like B-MAC and X-MAC, and scheduled protocols, like S-MAC, implicitly use quorum-based neighbor discovery. Their use differs only in the details of what happens during each interval and whether transmissions are row major and listening is column major in each group of m^2 intervals as is the case for sampling protocols, or the reverse for slotted ones.

Zheng et al. apply optimal block designs using difference sets to the problem of asynchronous neighbor discovery [152]. Their solution addresses the symmetric duty cycle problem, when all node duty cycles are uniform throughout the network. They conclude that for asymmetric duty cycles, their approach reduces to an NP-complete minimum vertex cover problem requiring a centralized solution. These limitations are at odds with the requirements of our problem.

Herman et al. explore the temporal partition problem in sensornets where two or more groups of nodes with differing schedules become unaware of each other [69]. The paper presents several self-stabilizing protocols to solve the problem of temporal partition; starting from an arbitrary temporally partitioned state, these protocols lead the network to a state in which all nodes have aligned sleep schedules. Their approach uses randomly chosen relatively prime sleep periods and occasional, and possibly random, probing of extra time slots. This approach constrains a node to only two duty cycle choices and the paper further states that since deterministically guaranteeing that two groups make distinct choices is difficult, the protocol resorts to randomization. Disco addresses a more general set of neighbor discovery problems and avoids the need for a randomized protocol by using *pairs* of primes.

4.3 Disco Design

This section presents the design of the Disco neighbor discovery and rendezvous protocol. Neighbor discovery (or rediscovery) allows two nodes with independent duty cycles and no prior (or current) synchronization information to discover each other in bounded time when the nodes are within radio range of each other. Rendezvous allows nodes to deliver messages to previously discovered neighbors with predictable and controllable latencies. We begin with a simplified version of the Disco algorithm that makes proving Disco's correctness straightforward. We then relax the simplifying assumptions to flesh out a protocol that works in practice.

4.3.1 Simplified Algorithm

Discovery is the process by which nodes learn about their current one-hop neighbors. The idea behind the discovery algorithm is simple. Two nodes, i and j, pick two numbers, m_i and m_j, such that m_i and m_j are relatively prime (coprimes) and $1/m_i$ and $1/m_j$ are approximately equal to i and j's desired duty cycles, respectively. Time is divided into fixed-width *reference periods* and consecutive periods are labeled with consecutive integers. Nodes i and j start counting the passage of these periods at times a_i and a_j, with their respective counters, c_i and c_j, initialized to zero, and with i and j counts synchronized to the reference period (we will relax this last assumption in later sections). If $c_i|m_i$ (c_i is divisible by m_i), then i turns on its radio for one period and beacons (or listens, or does both, depending on application requirements). Similarly, if $c_j|m_j$, then j turns on its radio for one period and beacons. When both i and j turn on their radios during the same period, they can exchange beacons and discover each other.

It is easy to see that there is exactly one such overlapping period every $m = m_i m_j$ periods. Letting x represent the reference period number, we have

$$\begin{aligned} c_i &= x - a_i \\ c_j &= x - a_j. \end{aligned}$$

Our goal is to find an x such that $c_i|m_i$ and $c_j|m_j$. We can express this as a pair of simultaneous congruences

$$\begin{aligned} x &\equiv a_i \pmod{m_i} \\ x &\equiv a_j \pmod{m_j}. \end{aligned}$$

Such a set of congruences are known to have a common solution by the Chinese Remainder Theorem [107]. This theorem states that if x_0 is one such solution, then an integer x satisfies the congruences if and only if x is of the form $x = x_0 + km$ for some integer k. One x_0 is

$$x_0 = a_i b_i m_j + a_j b_j m_i.$$

where the solution is unique $\pmod{m}$ for $m = m_i m_j$, and where b_i and b_j must satisfy the congruences

$$\begin{aligned} b_i m_j &\equiv 1 \pmod{m_i} \\ b_j m_i &\equiv 1 \pmod{m_j}. \end{aligned}$$

Let us consider a concrete example. Let node i select $m_i = 3$ (so i's duty cycle is: $DC \approx 33\%$), start counting at reference period $x = 1$ (so that $a_i = 1$), with counter values c_i. Similarly, let node j select

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
c_i	-	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
c_j	-	-	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

Figure 4.1: An example discovery timeline. Two nodes, i and j, start their counters, c_i and c_j, at times $x = 1$ and $x = 2$, with periods $m_i = 3$ and $m_j = 5$, and duty cycles of approximately 33% and 20%, respectively. The dark cells indicate times when the nodes i and j turn on their radio. Both nodes are awake at times $x = 7$ and $x = 22$. This pattern repeats when $x = 7 + 15k$, for all $k \in \mathbb{Z}^+$.

$m_j = 5$ (so j's duty cycle is: $DC \approx 20\%$), start counting at reference period $x = 2$ (so that $a_j = 2$), with counter values c_j. Figure 4.1 illustrates this timeline and counter values. Dark entries in the c_i and c_j rows indicate $c_i | m_i$ and $c_j | m_j$, respectively. Columns where both c_i and c_j are dark indicate values of x for which both i and j have overlapping on slots, and can therefore communicate. In this example, when $x = 7$ and $x = 22$, both i and j are turned on and can discover each other.

We can express this example as the following simultaneous congruences

$$\begin{aligned} x &\equiv 1 \pmod 3 \\ x &\equiv 2 \pmod 5, \end{aligned}$$

and see that when $x = 7$, both congruences are solved

$$\begin{aligned} (1-7)|3 \\ (2-7)|5 \end{aligned}$$

An analytic solution requires finding b_i and b_j

$$\begin{aligned} 5b_i &\equiv 1 \pmod 3 \\ 3b_j &\equiv 1 \pmod 5. \end{aligned}$$

We see that values of $b_i = 2$ and $b_j = 2$ satisfy these congruences and hence one solution x_0 is

$$\begin{aligned} x_0 &= a_i b_i m_j + a_j b_j m_i \\ x_0 &= 1 \cdot 2 \cdot 5 + 2 \cdot 2 \cdot 3 \\ x_0 &= 22. \end{aligned}$$

Since all solutions are unique $\pmod{15}$, we have

$$x_0 = 22 \pmod{15} = 7.$$

which agrees with our solution from Figure 4.1 and gives $x = 7 + 15k$, for all $k \in \mathbb{Z}^+$.

The preceding analysis sidesteps a number of practical considerations. Since, for example, the Chinese Remainder Theorem requires the moduli m_i and m_j be coprimes to guarantee a solution to the simultaneous congruences, these values cannot be independently chosen by the nodes, which is limiting. We also required that nodes be able to express their desired duty cycle as the reciprocal of a positive integer (e.g. $1, 1/2, 1/3, \ldots, 1/k$, where $k \in \mathbb{Z}^+$), which is restrictive. We assumed that nodes i and j synchronized their counting with the reference phase, which aids analysis but is unlikely to hold in practice. The preceding analysis also fails to explore the effect of clock drift on discovery, ignores radio startup time and energy overhead, and assumes that communications jitter is negligible. In the remainder of this section, we progressively relax these assumptions, and evolve our protocol, to flesh out one that works in practice.

4.3.2 Coprimes are not Enough

The Chinese Remainder Theorem requires the moduli m_i and m_j be coprimes to guarantee a solution to the simultaneous congruences. This restriction raises some challenges. First, the moduli cannot be chosen independently by the nodes since such choices could lead to values of m_i and m_j that are not coprimes. It would, of course, be preferable to let nodes choose the moduli that best satisfy their individual duty cycle requirements rather than require a static or central assignment. Second, restricting the moduli to coprimes is not scalable since there are only a handful of numbers that can satisfy both the target duty cycle (typically 1-5%) and coprime requirement. Third, if $m_i = m_j$, then nodes i and j may never discover each if they wake up with the same period but different phase.

One way to allow each node to select the best duty cycle for itself while still ensuring discovery occurs is to require each node i to pick two primes, p_{i_1} and p_{i_2}, such that $p_{i_1} \neq p_{i_2}$ and the sum of their reciprocals (approximately) equals the desired duty cycle

$$DC \approx \frac{1}{p_{i_1}} + \frac{1}{p_{i_2}}.$$

Each node increments a local counter and if a node's local counter is divisible by either of its primes, the node turns on its radio for a single interval, whose length is the only global parameter. This approach ensures that no matter what duty cycles are independently selected at different nodes, for every pair of nodes i and j, there will be at least one pair in the set $\{(p_{i_1}, p_{j_1}), (p_{i_1}, p_{j_2}), (p_{i_2}, p_{j_1}), (p_{i_2}, p_{j_2})\}$ that are relatively prime, satisfying the Chinese Remainder Theorem. Note, however, that simply requiring p_{i_1} and p_{i_2} to be coprimes does not satisfy the requirements of the Theorem, and therefore cannot ensure discovery. For example, letting $(p_{i_1}, p_{i_2}) = (30, 77)$ and $(p_{j_1}, p_{j_2}) = (35, 66)$, ensures that intra-node pairs are coprime

$$\begin{aligned} \gcd(p_{i_1}, p_{i_2}) = \gcd(30, 77) &= 1 \\ \gcd(p_{j_1}, p_{j_2}) = \gcd(35, 66) &= 1, \end{aligned}$$

however, the inter-node pairs are not

$$\begin{aligned} \gcd(p_{i_1}, p_{j_1}) = \gcd(30, 35) &= 5 \\ \gcd(p_{i_1}, p_{j_2}) = \gcd(30, 66) &= 6 \\ \gcd(p_{i_2}, p_{j_1}) = \gcd(77, 35) &= 7 \\ \gcd(p_{i_2}, p_{j_2}) = \gcd(77, 66) &= 11, \end{aligned}$$

which means discovery may fail. Consider the case in which node i starts counting at time $x = 0$, so node i's radio is on at times $x = 30k$ and $x = 77k$, for all $k \in \mathbb{Z}^+$, and node j starts counting at time $x = 1$, so node j's radio is on at times $x = 35k + 1$ and $x = 66k + 1$, for all $k \in \mathbb{Z}^+$. There is no x for which both i and j have their radios turned on simultaneously, ensuring discovery will fail. Therefore, to ensure correctness, Disco uses prime pairs rather than coprimes.

4.3.3 Choosing Primes

The choice of primes can have a large impact on discovery latency. For example, a target duty cycle of 2% can be achieved in several ways. One combination of primes that achieves this duty cycle is 97 and 103 ($1/97 + 1/103 = 2\%$) but another combination is 53 and 883 (1/53 + 1/883 = 2%). Which combination

is better? Assume that both nodes picked the same pair of primes. In that case, with high probability, the worst-case discovery latency will be $97 \times 103 = 9,991$ periods vs $53 \times 883 = 46,799$ periods, more than a factor of four difference. Now, assume that one node picked 53 and 883 while the other node picked 57 and 409. In this case, the worst-case discovery latency becomes $53 \times 57 = 3,201$, *fifteen times faster* than $53 \times 883 = 46,799$.

The ratio of the worst-case discovery latency between an auspicious and an unfortunate set of prime pairs is bounded by the duty cycle. For example, a 10% duty cycle results in no worse than a 1:10 ratio, a 2% duty cycle is bounded by a 1:50 ratio, and a 1% duty cycle results in at worse a 1:100 ratio. If we let $c = 1/DC$, we have

$$\frac{1}{c} \approx \frac{1}{p_{i_1}} + \frac{1}{p_{i_2}} \tag{4.1}$$

rearranging and solving for p_{i_2}, we have

$$p_{i_2} \approx \frac{p_{i_1} c}{p_{i_1} - c}. \tag{4.2}$$

The limit of the ratio between the auspicious and unfortunate worst-case latencies is

$$\lim_{p_{i_1} \to c+1} \frac{p_{i_1}^2}{p_{i_1} p_{i_2}} = \frac{(c+1)^2(c+1-c)}{(c+1)(c+1)c} = \frac{1}{c} = DC. \tag{4.3}$$

These observations suggest that picking the prime pairs requires care: a good choice can result in low discovery latency but a poor choice can result in much longer worst-case discovery latency. Low discovery times are possible if one of the primes is very close to the reciprocal of the duty cycle while the other prime is a much larger number. If this approach is taken, then it becomes important to randomize the choice of prime pairs to reduce the chance that two nodes will have picked the same pair if they both select the same duty cycle.

If nodes can be assigned to different classes such that members of a class do not need to discover or communicate with each other, then it is easy to ensure good pairs are selected. For each possible duty cycle, every node running Disco employs a deterministic algorithm to generate an ordered list of prime pairs that can satisfy the duty cycle. A prime pair's position in the ordered list, taken modulo the number of distinct classes defined by the application, dictates the particular class to which a pair is assigned. A node chooses at random one of the prime pairs assigned to its class. This algorithm ensures that nodes in different classes are assigned distinct pairs, which, as we will show later can greatly improve discovery latency. The policy of class label assignment is left to the application, but the mechanism to ensure good inter-class pairs are chosen is handled by Disco.

4.3.4 Slot Non-Alignment

We now relax the assumption that slots are aligned and delve into the details of slot construction. In practice, slots will rarely be aligned since nodes are run independently and do not adjust clock skews or set up a global time reference. Since we now assume slots are not aligned, we need to ensure that two nodes will still discover each other regardless of how their slots overlap. *To maximize the likelihood that overlapping slots result in discovery, Disco transmits a beacon at both the beginning and end of a slot when beaconing*.

Even if slots are generally non-aligned, nodes that are in-phase may come into contact with each other from time to time. When this happens, both nodes may attempt to transmit their beacons at the same time, causing collisions or receiving each others beacons during every slot. Although this situation can occur,

Disco leaves it to the application to decide how to respond at a coarse grain by, for example, changing the duty cycle or changing to a listen-only mode. Part of the reason to leave this to the application is that if nodes have even small variations in the clocks, nodes are likely to fall out of phase naturally. One API design question is whether there should be explicit support for letting the application shift phase to explicitly deal with synchronization?

An important question is what a node should do if the channel is busy. There are a handful of options. A node may blindly transmit regardless of channel contention, it could enter a channel contention phase, it could forgo transmission altogether, or it could wait until the channel is clear and then transmit. Blindly transmitting when the channel is busy is hardly scalable and would lead to channel contention from colliding beacons. Entering channel contention could introduce a long delay, effectively throwing off the timing of the slot. Forgoing transmission also throws off the timing of the slot. Disco currently transmits as soon as the channel is clear, provided it does not expect any of its neighbors to do so.

4.3.5 Duty Cycle from Discovery Latency

In many applications, it will be necessary to compute the duty cycle or beacon rate required to satisfy a particular discovery latency. The application will specify the worst-case discovery latency tolerable and the minimum duty cycle will need to be computed. The procedure to convert maximum discovery latency, t_{disco}, to duty cycle is relatively simple. Two nodes operating with primes p_{i_1} and p_{j_1} will discover each other in at most $p_{i_1}p_{j_1}$ counter periods, where each counter period is of length t_{slot}. For discovery to occur in the required time, the following inequality must hold

$$p_{i_1}p_{j_1}t_{slot} \leq t_{disco}.$$

Without loss of generality, we assume the primes are equal so that $p = p_{i_1} = p_{j_1}$, giving us the following constraint for choosing the primes

$$p \leq \sqrt{\frac{t_{disco}}{t_{slot}}}.$$

Recall, however, that Disco requires a *pair* of primes to ensure discovery when duty cycles are independently chosen. Therefore the minimum duty cycle, DC, must satisfy the following inequality

$$DC \geq \frac{1}{p} + \frac{1}{p} = \frac{2}{p}.$$

Since each prime p results in a beacon slot every p slots, the minimum required beacon rate is given by

$$f_{beacon} \geq \frac{2}{p \cdot t_{slot}} = \frac{2}{\sqrt{t_{disco}t_{slot}}} \text{ Hz},$$

and although the beacon rate *increases* with smaller t_{slot} values, the effective duty cycle *decreases*

$$DC \geq \frac{2}{p} = 2\sqrt{\frac{t_{slot}}{t_{disco}}},$$

Which provides a way to compute the minimum duty cycle given t_{slot} and t_{disco}.

4.3.6 Duty Cycle Granularity

A side effect of allowing only those duty cycles that can be expressed as the sum of the reciprocal of two primes is that many large duty cycles cannot be specified with fine granularity. For example, the only legal duty cycles between 57.6% and 100% are shown in Table 4.1.

p_{i_1}	1	2	2	2	2	2	3	2
p_{i_2}	0	3	5	7	9	11	4	13
DC (%)	100	83.3	70	64.3	61.1	59	58.3	57.6

Table 4.1: Legal duty cycles (DC) between 57.6% and 100%. Many duty cycles cannot be realized and the distribution of duty cycles is not uniform when only two primes are used.

To allow a more flexible and fine-grained assignment of duty cycles, Disco supports a third parameter, p_{i_3} that can assume any prime number. With this addition, the duty cycle becomes

$$DC \approx \frac{1}{p_{i_1}} + \frac{1}{p_{i_2}} + \frac{1}{p_{i_3}}. \tag{4.4}$$

4.3.7 Robustness to Clock Skew

Clock skew presents a challenge for many synchronized protocols and requires nodes to estimate and adjust for neighbors' clock drift to maintain synchronization. Some time synchronization protocols use linear regression to perform this function [99, 125] but since Disco generally operates in an unsynchronized manner, it only updates information about clock offsets and does not compute skews. Over short timescales, this approach works well, but over longer timescales, nodes that have significantly different skews are likely to have difficulty with rendezvous, but asynchronous discovery should still work. Whether or not skew compensation is needed is unclear but one way to explore this question is to assume clock skew is $r_{skew} = \Delta f / f$. Under normal circumstances, a cycle repeats every $m = p_{i_x} \cdot p_{j_y}$ slots. If the skew during a cycle must be less than some slot fraction α, to ensure overlapping slots, then we must ensure

$$t_{slot} \cdot p_{i_x} \cdot p_{j_y} \cdot r_{skew} \leq \alpha \cdot t_{slot}. \tag{4.5}$$

For typical values of $p_{i_x} = 97$, $p_{j_y} = 103$ (a 2% duty cycle), and a conservative skew assumption $r_{skew} = 50$ ppm, $\alpha \approx 0.5$, suggesting that with typical crystals operating at the very extremes of their temperature specification, there could be up to 1/2 of a slot-width's phase shift every m slots when running at a 2% duty cycle. This may be the case for some of the nodes used in our earlier experiments. Because of the design of our slots – a beacon at the beginning and one at the end – a phase shift simply means that a *different* pair of slots will overlap, quite possibly sooner than predicted by the Chinese Remainder Theorem, since every offset between zero and $max(p_{i_x}, p_{j_y})$ occurs between overlapping slots.

For example, in Figure 4.1, we see that between time $x = 7$ and time $x = 22$, node i and node j have awake slots that are offset from each other by every value between zero and $max(3, 5) = 5$. Offsets of zero ($x = 7$), one ($x = 12$ and $x = 13$; $x = 16$ and $x = 17$), two ($x = 10$ and $x = 12$; $x = 17$ and $x = 19$), three ($x = 7$ and $x = 10$), four ($x = 12$ and $x = 16$; $x = 13$ and $x = 17$) and five ($x = 7$ and $x = 12$; $x = 17$ and $x = 22$) all appear. This suggests that even with clock skew, there may be overlaps that continue to occur. Understanding and characterizing this phenomenon may allow use of discovery at duty cycles below 1% and even at the extremes of the temperature operating range.

4.4 Disco Implementation

To evaluate the feasibility and performance of our design, we implemented Disco using the nesC programming language [63], TinyOS operating system [72], and Telos [112] and Epic sensor nodes [48]. Figure 4.2 shows the radio on time, beacon transmissions, and current draw profile during a 25 ms slot. We experimented with a range of t_{slot} values and found that discovery performance degrades when $t_{slot} < 5$ ms due to the jitter introduced by the TinyOS timer library and radio stack. Therefore, we use $t_{slot} = 10$ ms in the rest of this chapter, but we note that with a dedicated hardware timer and `async` calls, which would reduce latency and jitter, the slot time could be reduced. Such improvements would require changes to the radio driver internals and interfaces, both to remove context switches and expose internal control points, which our current implementation does not require but from which it would certainly benefit.

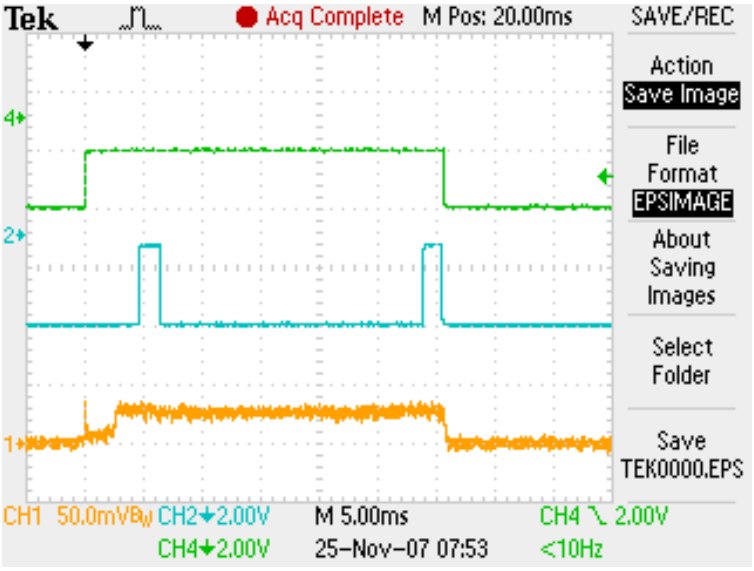


Figure 4.2: Beacon slot details ($t_{slot} = 25$ ms). The green line (top) indicates the on-time envelope of the radio. The blue line (middle) shows beacon transmissions at the beginning and end of the slot, and the orange line (bottom) shows the current draw in mA (not mV) using a 1 Ω sense resistor in series with the power supply.

The `Discovery` interface, shown in Figure 4.3, allows an application to control discovery parameters, policy, and traffic tunneling. A duty cycle between 0 and 100% can be requested and the service will indicate the duty cycle actually used. Alternately, an application can explicitly choose the primes and local counter value. The worst-case discovery latency can be limited by assigning different nodes to different classes. The application can control the beaconing policy and piggy-back packets over the discovery channel.

The conversion from duty cycle to primes occurs as follows. Given a duty cycle, c, let

$$p_{1_{min}} = \left\lfloor \frac{1}{c} \right\rfloor + 1, \quad \text{and} \quad p_{1_{max}} = \left\lfloor \frac{2}{c} \right\rfloor,$$

and for each prime, p_1, between $p_{1_{min}}$ and $p_{1_{max}}$, a prime pair, p_2, is selected from a range starting at

$$p_{2_{min}} = \left\lceil \frac{1}{c - 1/p_1} \right\rceil$$

```
interface Discovery {
  // Request a duty cycle between 0 and 100 percent
  command uint8_t  setDutyCycle(uint8_t dutycycle);
  command uint8_t  getDutyCycle();

  // Get/set the primes explicitly
  command error_t  getPrimes(uint16_t *p1, uint16_t *p2 uint16_t *p3);
  command error_t  setPrimes(uint16_t p1, uint16_t p2 uint16_t p3);

  // Get/set the local counter explicity
  command uint32_t getLocalCounter();
  command error_t  setLocalCounter(uint32_t counter);

  // Set/get the node class to reduce inter-class latency
  command uint8_t  getNodeClass();
  command error_t  setNodeClass(uint8_t classid);
  command uint8_t  getNodeClassCount();
  command error_t  setNodeClassCount(uint8_t count);

  // Select beacon-and-listen or listen-only mode
  command beacon_t getBeaconMode();
  command error_t  setBeaconMode(beacon_t mode);

  // Request, event, callback for app-specific payload
  command error_t  requestBroadcast();
  event error_t    fetchPayload(void *buf, uint8_t *len);
  event message_t  received(message_t* msg, void* buf,
    uint8_t len);
}
```

Figure 4.3: The discovery programming interface. An application can control discovery parameters, policy, and traffic tunneling. A duty cycle between 0 and 100% can be requested and the service will indicate the (integral) duty cycle actually used. The choice of primes can be set explicitly, as can be the value of the local counter. The worst-case discovery latency can be limited by assigning different nodes to different classes (the total number of node classes must be specified). The application can control the beaconing policy (listen only or beacon on any combination of the primes), and piggy-back packets over the discovery channel.

and increasing. The prime pairs, (p_1, p_2) are labeled sequentially. For example, a 5% duty cycle generates the ordered mapping of labels, $\ell \in \mathbb{L}$, to prime pairs: $\{1, 2, 3, 4\} \rightarrow \{(23, 157), (29, 67), (31, 59), (37, 43)\}$. If the number of node classes, n, equals one, then Disco selects a random element $\ell \in \mathbb{L}$. If $|\mathbb{L}| \geq n > 1$, then Disco selects a prime pair randomly from the subset of $\mathbb{L}$ that is divisible by the node class number, k, where $1 \leq k \leq n$. Finally, if $n > |\mathbb{L}| > 1$, then Disco selects a prime pair as follows. The prime pair with label $\ell = (k-1) \pmod{|\mathbb{L}|} + 1$ is chosen. This last equation maps more than one node class to the same prime pair, but with fewer prime pairs than node classes, the options are limited.

Protocol	Parameters	Duty Cycle	Latency CDF(n)	Asymm
Disco	$(p_{i_1}, p_{i_2}), (p_{j_1}, p_{j_2})$	$\frac{p_{i_1}+p_{i_2}-1}{p_{i_1} \cdot p_{i_2}}, \frac{p_{j_1}+p_{j_2}-1}{p_{j_1} \cdot p_{j_2}}$	No closed form	Yes
Birthday	$0 \leq p_{tx}, p_{rx} \leq 1$	p_{tx}, p_{rx}	$1-(1-p_{tx} \cdot p_{rx})^n, \forall n \in \mathbb{Z}^+$	Yes
Quorum	$m \in \mathbb{Z}^+$	$\frac{2m-1}{m^2}$	$1-(1-\frac{n}{m^2})^2, \forall n \leq m^2$	No
Combin.	$k = p^q; p \in \mathbb{P}, q \in \mathbb{Z}^+$	$\frac{k+1}{k^2+k+1}$	$1-\frac{n}{k^2+k+1}, \forall n \leq k^2+k+1$	No

Table 4.2: Comparison of asynchronous neighbor discovery techniques including Disco, Birthday [101], Quorum [139], and Combinatoric [152]. The duty cycle and discovery latency of these techniques are parameterized by primes (Disco), transmit and receive probabilities (Birthday), the rank of a square matrix (Quorum), and powers of a prime (Combinatoric). The latency cumulative distribution function describes the probability of discovery after n trials or slots as a function of the protocol parameters. An asymmetric protocol allows each node to choose a duty cycle independently of other nodes and still ensure discovery (with high probability in case of the Birthday protocol). $\mathbb{P}$ is the set of primes. $\mathbb{Z}^+$ is the set of positive integers.

4.5 Simulation Study

In this section, we evaluate the performance of Disco with earlier work through simulation and we study the relationship between slot length, beacon rate, discovery latency, discovery rate, and duty cycle. In particular, we evaluate the sensitivity of the protocol to the choice of primes.

We use the term *balanced primes* to refer to the case in which the intra-node primes are approximately equal (e.g. 37 and 43). The term *unbalanced primes* refers to the case in which the intra-node primes are significantly different (e.g. 23 and 157). We use the term *symmetric pairs* to refer to the case in which both nodes choose the *identical* pair of primes. The term *asymmetric pairs* refers to the case in which both nodes choose a *different* pair of primes.

4.5.1 Simulation Models

Table 4.2 shows the three most closely related asynchronous neighbor discovery services. We developed the closed form expression of latency CDFs to speed up simulation, but we verify their output against a random set of brute force simulations. Even though Disco is compared against both Birthday and Quorum, neither of the two actually meets application needs. Birthday is based on a randomized algorithm that does not provide predictable rendezvous times and exhibits a long tail for discovery while Quorum specifies a global constant that all nodes use for their duty cycle.

4.5.2 Discovery Latency Comparison

Discovery latency refers to the delay between the moment two nodes are within communications range to the moment when they first discover each other. The distribution of discovery latencies, as well as the tail of this distribution (*i.e.* worst-case discovery latency), are both important metrics for a neighbor discovery protocol. The worst-case discovery latency determines the minimum amount of time two nodes need to be in communications range to ensure discovery. The distribution provides insight into the average case, or median, behavior.

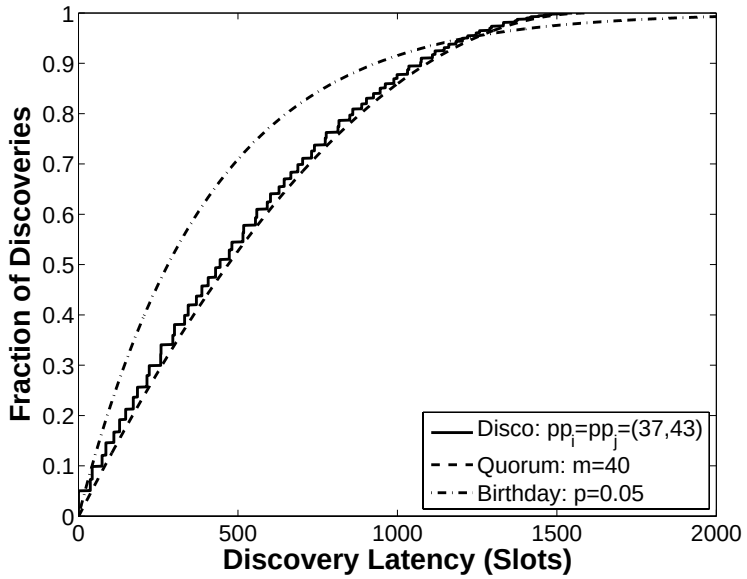


Figure 4.4: The distribution and worst-case discovery latency of Disco closely matches the Quorum protocol. Both Disco and Quorum trail the Birthday protocol, which achieves the lowest latency 95% of the time, but its probabilistic nature leads to a long tail. The CDF of discovery latency for Disco, Quorum, and Birthday protocols operating at a 5% duty cycle is shown. The two Disco nodes use balanced primes and symmetric pairs (37,43); the Quorum system uses $m = 40$; and the Birthday protocol nodes both turn on their radio with probability $p = 0.05$.

In this section, we compare the discovery latency of the Disco approach with those of the probabilistic [101] and quorum [139] approaches. We do not compare Disco with the combinatoric approach that uses difference sets because that approach addresses the symmetric problem, when node duty cycles are the same, but concludes that for asymmetric duty cycles, the approach reduces to an NP-complete minimum vertex cover problem requiring a centralized solution [152]. These limitations are at odds with many of the requirements of our problem.

For Disco, the discovery latency is a function of the particular prime pairs being used as well as the offset in the node counters. For the grid quorum system, this latency is a function of the particular row and column choices being used as well as the group size, m^2, where these m^2 intervals or slots are arranged as a 2-dimensional $m \times m$ array in row-major manner [139]. For birthday protocols, the discovery latency is a function of the beacon/listen probability of each node [101].

Figure 4.4 shows the cumulative distribution of discovery latencies for Disco using the (37,43) balanced primes and symmetric pairs, Quorum using $m = 40$ (value of m that makes $\frac{2m-1}{m^2} = 5\%$), and Birthday using probability $p_{tx} = p_{rx} = 0.05$, with all protocols operating at 5% duty cycle. The 50-th percentile discovery latency is 444 slots for Disco, 470 slots for Quorum, and 281 slots for Birthday. The distribution and worst-case discovery latency of Disco closely matches the Quorum protocol, although Disco performs just slightly better. Both Disco and Quorum trail the Birthday protocol, which achieves the lowest discovery latency 95% of the time, but its underlying probabilistic nature leads to a long tail.

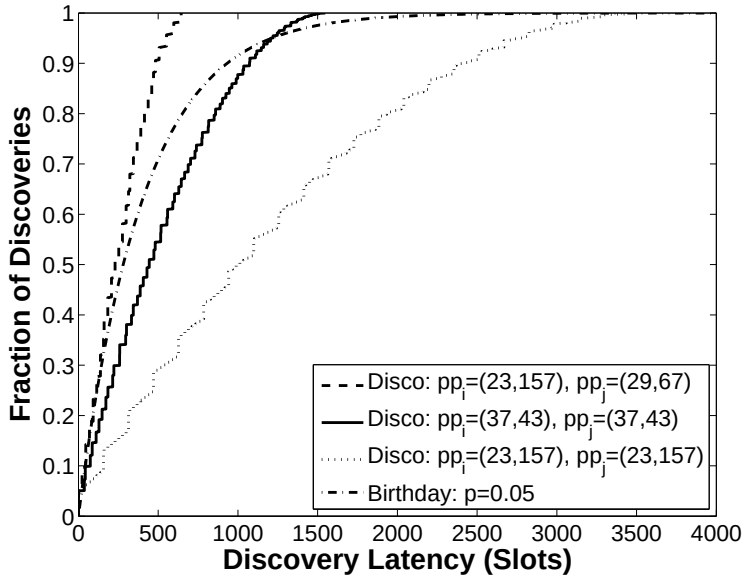


Figure 4.5: The choice of prime pairs significantly affects the latency distribution and worst-case discovery latency. The CDF of discovery latency for three different Disco prime pairs as well as the Birthday protocol, all operating at a 5% duty cycle, is shown. Unbalanced primes in asymmetric pairs provide the best overall behavior and offer the lowest worst-case discovery latency – better than all other approaches. In contrast, unbalanced primes in symmetric pairs provide the worst average-case behavior and the highest worst-case discovery latency.

4.5.3 Discovery Latency: A Deeper Look

The discovery latencies in Figure 4.4 for Disco reflect a particular choice of prime pairs. In general, if two Disco peers select balanced primes and symmetric pairs, their discovery latency will closely track that of the Quorum protocol. If, on the other hand, nodes choose unbalanced primes and asymmetric pairs, then the discovery latency could be reduced significantly, or increased considerably, as Figure 4.5 shows. Three Disco cumulative distributions are plotted, showing the range of potential discovery latencies. The Birthday distribution for the same duty cycle is plotted as a baseline. Unbalanced primes in asymmetric pairs of (23,157) and (29,67) provide the best average-case behavior, 230 slots at the 50-th percentile mark, and they also offer the lowest worst-case discovery latency of 644 slots – better than all other approaches. In contrast, unbalanced primes in symmetric pairs of (23,157) and (23,157) provide the worst average-case behavior, 1012 slots at the 50-th percentile mark, and the highest worst-case discovery latency of 3454 slots. Balanced primes in symmetric pairs of (37,43) and (37,43) and Birthday using probability $p_{tx} = p_{rx} = 0.05$ is shown for comparison.

The ratio of the worst-case discovery latency between an a good and bad pairing is bounded by the duty cycle. For example, a 10% duty cycle results in no worse than a 1:10 ratio, a 5% duty cycle is bounded by a 1:20 ratio, and a 1% duty cycle results in at worse a 1:100 ratio. While these numbers paint a grim picture of the worst-case downside, they fail to capture *how likely* that downside is to occur. Since unbalanced primes in symmetric pairs result in the worst discovery latency, it is worth exploring how often such bad pairs occur and what is lost if more conservative pairings are used.

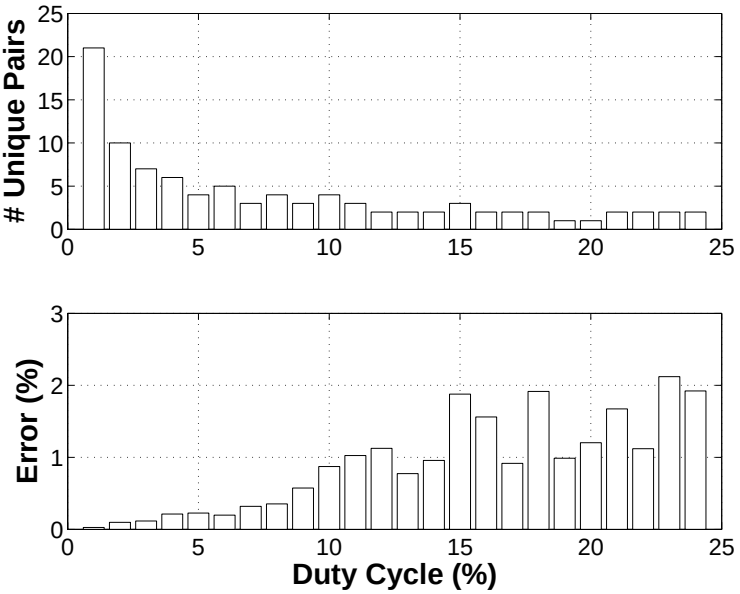


Figure 4.6: The number of unique prime pairs that generate a particular duty cycle and largest error across all of the pairs for each duty cycle. Error is measured as the magnitude of the deviation from the desired duty cycle, e.g. for primes a and b, and duty cycle c, the error is $\left|\frac{1}{c} - \frac{a+b-1}{a \times b}\right|$.

Bad pairings occur whenever two nodes pick the same prime pairs. The key question is how often this happens. Assume that two nodes operate at the same duty cycle and they choose their prime pairs independently and uniformly randomly from a set of k prime pair choices. Then, the chance that they pick the same pair is $1/k$. If k is large, then this chance is small. Figure 4.6 shows the number of unique prime pairs for each possible integral duty cycle from 1% to 25% as well as the largest duty cycle error across all pairs for a given duty cycle. The number of unique pairs possible for each integral duty cycle value grows quickly as the duty cycle falls below 5%. Since 5% appears to be near the elbow of the curve in the number of unique pairs in Figure 4.6, we next explore the cumulative distribution of discovery latencies across all 16 prime pair combinations for a 5% duty cycle.

Figure 4.7 shows the distribution of discovery latencies of all sixteen pairings possible with each node choosing one of the following prime pairs: (23,157), (29,67), (31,59), (37,43). The dark line with a worst-case latency of 1,591 slots highlights the case when both nodes select the (37,43) pair. All lines to the right of the (37,43) line are cases when both nodes choose the same pair while all lines to the left of the (37,43) line are cases when nodes choose dissimilar pairs. The conclusion is clear: asymmetric pairs dramatically reduce discovery latency, by 30 – 50%.

Figure 4.8 shows the cumulative distribution of the median (50-th percentile) and worst-case (100-th percentile) discovery latency *across all possible prime pair combinations* for a 5% duty cycle. These values are taken directly from the 50-th and 100-th percentile data points from the 16 CDFs shown in Figure 4.7. The horizontal line labeled "Balanced/Symmetric" identifies the median and worst-case discovery latencies when both nodes select the (37,43) pair. The key observation is that the data show excellent average-case performance: 75% of all pairwise combinations result in median discovery latency of less than 261 slots (substantially less than the median discovery latency for the balanced/symmetric pairs) and over 93% of all

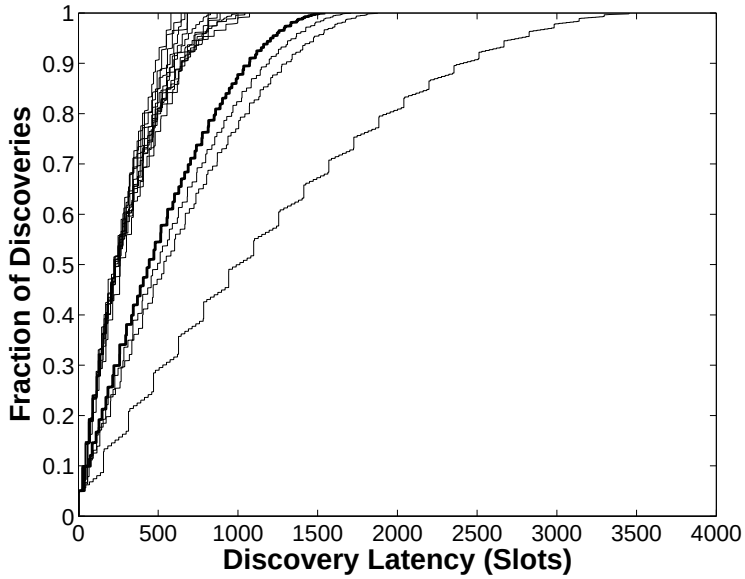


Figure 4.7: The cumulative distribution of discovery latency across all 16 possible prime pair values for a 5% duty cycle. Although the worst-case discovery latency is 3,611 slots when both nodes choose the (23,157) pair, the median discovery time is much lower.

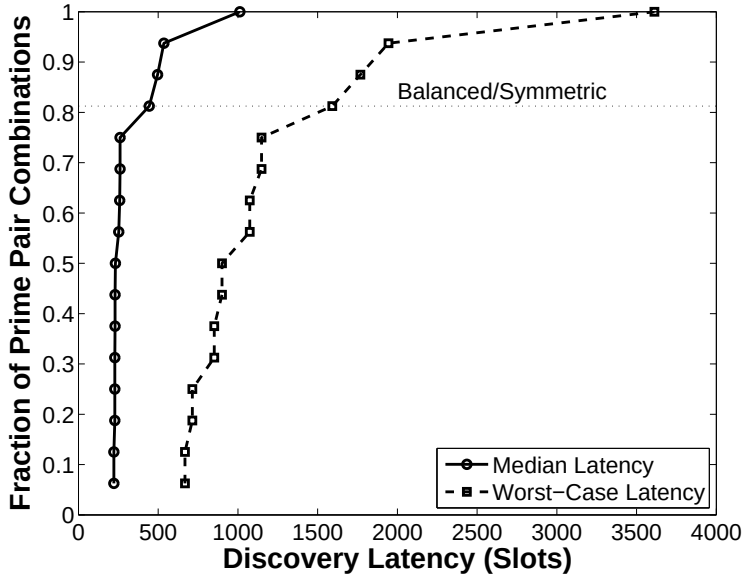


Figure 4.8: The cumulative distribution of the median (50-th percentile) and worst-case (100-th percentile) discovery latency values across all 16 possible prime pair combinations for a 5% duty cycle. The dotted line intersects the CDF of the balanced primes (37,43) in symmetric pairs.

combinations result in a median discovery latency of less than 536 slots, 2.61 seconds and 5.36 seconds, respectively, using our implementation slot length of 10 ms.

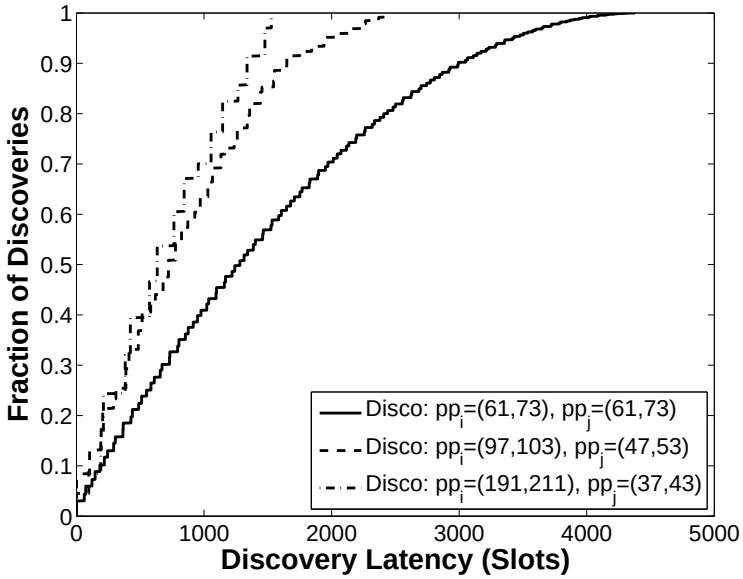


Figure 4.9: Discovery latency decreases with increasing asymmetry in pairwise duty cycles for a fixed average duty cycle. The CDF of discovery latency for an average duty cycle of 3% is shown. This 3% average is achieved in three ways: (3%+3%)/2 using prime pairs (61,73) and (61,73), (2%+4%)/2 using prime pairs (97,103) and (47,53), and (1%+5%)/2 using prime pairs (191,211) and (37,43).

4.5.4 Impact of Duty Cycle Asymmetry

In some docking applications, discovery occurs between nodes with dissimilar energy supplies [144]. In other applications, a fixed amount of energy may be allocated between two or more nodes [109]. And, in a network of equal-energy nodes, operating nodes at different duty cycles makes sense [68]. Figure 4.9 shows that inter-pair asymmetry reduces discovery latency for a fixed pairwise-average duty cycle. This suggests that more powerful beacons combined with less powerful mobile tags is feasible and beneficial, and well supported by the algorithm.

4.5.5 Latency-Driven Discovery with Small Encounter Windows

In some applications, the encounter window of two nodes is short, and it becomes necessary to configure the duty cycle, DC, or beacon rate, f_{beacon}, to ensure that discovery occurs within this short window [21, 75]. The duty cycle and beacon rate depend on the maximum tolerable discovery latency, t_{disco}, and the length of a beaconing slot, t_{slot}, as derived in Section 4.3.5. The duty cycle required to ensure discovery in time t_{disco} is

$$DC \geq 2\sqrt{\frac{t_{slot}}{t_{disco}}},$$

and beacon slot rate required to ensure discovery is

$$f_{beacon} \geq \frac{2}{\sqrt{t_{disco}t_{slot}}} \text{ Hz}.$$

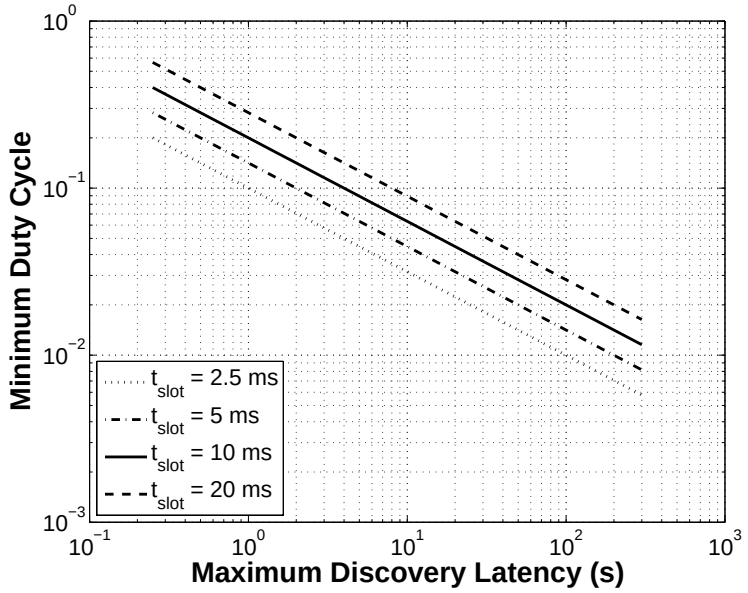


Figure 4.10: The minimum duty cycle required to ensure a maximum discovery latency.

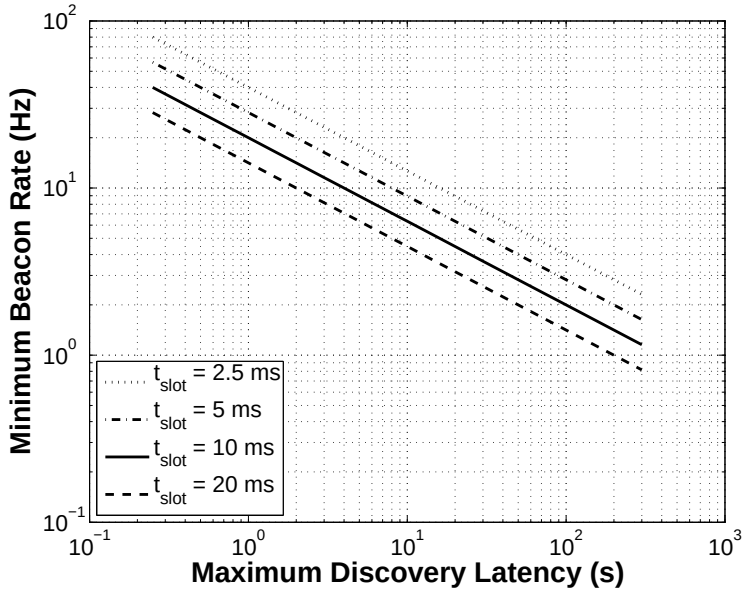


Figure 4.11: The minimum beacon rate required to ensure a maximum discovery latency.

Figures 4.10 and 4.11 show the minimum duty cycle and beacon slot rate, respectively, required to ensure a maximum discovery latency across a range of t_{slot} values. For example, to ensure discovery in 100 seconds using t_{slot} =10 ms, at least a 2% duty cycle or a 2 Hz beacon rate would be needed. Of course, the two values are complementary ways of specifying the same underlying discovery workload since

$$2 \text{ Hz } \times 10 \text{ ms} = 2\%.$$

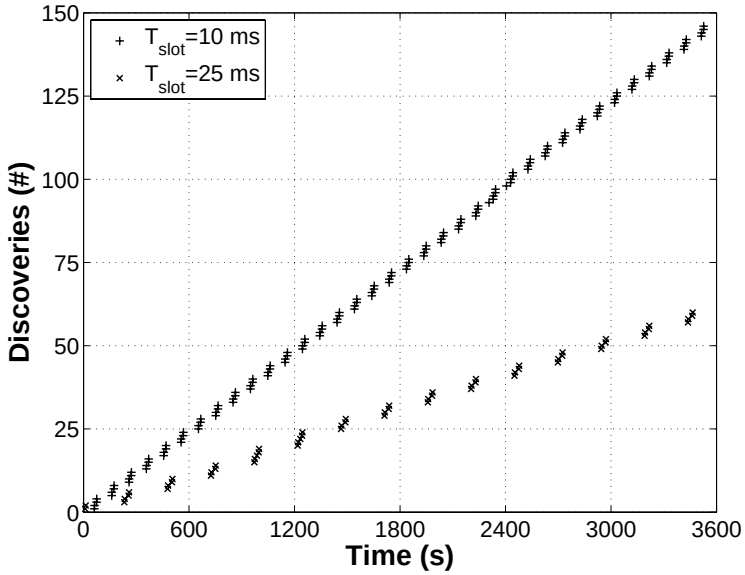


Figure 4.12: Empirical discovery timeline for two different t_{slot} values (10 ms and 25 ms) using balanced primes and symmetric pairs (97, 103). Rendezvous is stable and predictable over the one hour experiment.

4.6 Empirical Performance Evaluation

In this section, we evaluate the performance of Disco empirically, based on our TinyOS implementation that runs on Telos and Epic motes. We study the sensitivity of Disco to slot length, real-world effects like clock skew and jitter, and node density.

4.6.1 Discovery Rate

Discovery rate refers to the number of discovery beacons received per unit time. Figure 4.12 shows an empirical timeline of discoveries between a pair of Telos nodes for two values of the slot period, t_{slot}, collected over two one hour periods. For a t_{slot} value of 10 ms, approximately 150 discoveries or rendezvous occur over a one hour period, which translates to an average discovery period of about 24 seconds. For a t_{slot} value of 25 ms, 60 discoveries or rendezvous occur over a one hour period, which translates to an average discovery period of about 60 seconds. The ratio of these two numbers matches the ratio of the two different t_{slot} values, as expected, showing that the average discovery rate scales linearly with slot length, even though duty cycle and beacon rate do not. Note that with a 2% duty cycle using *symmetric* pairs (97,103), the worst-case discovery latency is 9,991 slots or about 100 seconds for the 10 ms slot and 250 seconds for the 25 ms slot.

4.6.2 Discovery Latency in Clusters

Figures 4.13 and 4.14 compare how simulated and empirical results compare in clusters. These two figures present the same underlying data in different ways. To collect this data, seven nodes were programmed to operating at a 2% duty cycle using balanced primes and symmetric pairs (97, 103). The unique neighbor discoveries of one particular node (the "test" node) were logged. Each time the test node discovered all of

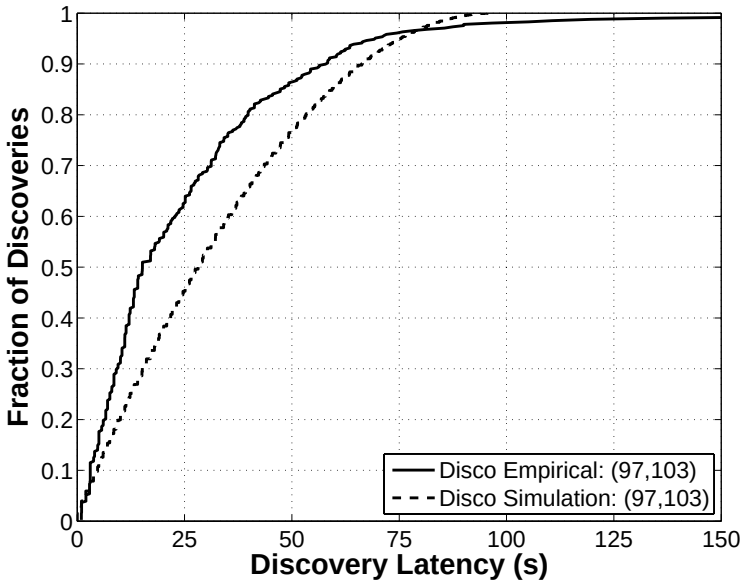


Figure 4.13: The empirical discovery latency is lower than the simulated discovery latency in 95% of trials ($N = 408$) but in 2% of the trials, the empirical discovery latency exceeds the worst-case discovery latency obtained through simulation. The nodes are operating with a $t_{slot} = 10$ ms and at a 2% duty cycle using balanced primes and symmetric pairs (97, 103) for both empirical and simulation results.

its neighbors, it was held in reset. Each of the other nodes were randomly reset while the test node was held in reset. The test node started running (i.e. was released from reset) a random amount of time after the other nodes were reset. The discovery latency of each neighbor (time to first discovery from reset) was logged.

The distribution of empirical discovery latencies is shown in Figure 4.13 along with the simulated latencies. The empirical discovery latency is lower than the simulated discovery latency in 95% of trials ($N = 408$) but in 2% of the trials, the empirical discovery latency exceeds the worst-case discovery latency obtained through simulation. This difference is due to two main factors. First, the slots are rarely aligned in practice while in simulation, and for worst-case analysis, they are aligned. This slot non-alignment generally leads to both lower discovery latencies and more frequent discoveries. It may also be due to clock skew and jitter, both of which contribute to small variations in slot times.

Figure 4.14 shows the time to discover the 1st, 2nd, 3rd, 4th, and 5th neighbor when a node joins a cluster (simulated by being reset). Note that the discovery latencies are for the first through fifth neighbors discovered regardless of the actual neighbor identifier. Discovery latency times for the sixth neighbor is not shown because of a long tail (the tail is, however, shown in Figure 4.13). The long tail may be an artifact of channel contention or collisions, especially since all nodes are using the same prime pairs. The data suggest that beaconing rate adaptation may very well be necessary for node clusters of modest density.

Despite the long tail in Figure 4.14, the median discovery latency for the first (of six) neighbors is far lower than the median discovery across all of the neighbors shown in Figure 4.13. This suggests that sharing neighbor table information may decrease neighbor discovery latency quite significantly (by nearly an order of magnitude in this example, since all nodes are within radio reach of all other nodes). More generally, striking a balance between sharing neighbor table entries and streamlining the beaconing process is required, and will reflect application policies.

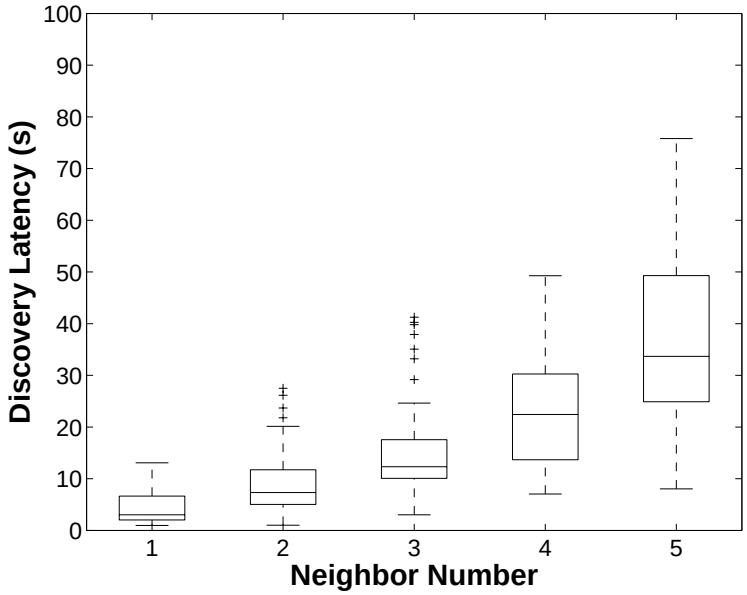


Figure 4.14: The discovery latency of the 1st, 2nd, 3rd, 4th, and 5th neighbors (regardless of their actual ids) when a node joins a cluster with six nodes. The nodes are operating with a $t_{slot} = 10$ ms and at a 2% duty cycle using balanced primes and symmetric pairs (97, 103) for both empirical and simulation results. The sixth node is not shown because of a long tail, suggesting beaconing adaptation may be needed in dense node clusters.

4.7 Discussion

Having presented and evaluated the Disco design both empirically and through simulation, we now revisit some open issues that merit further study and discuss some possible extensions to this work.

4.7.1 Beacon Rate Adaptation

As node density increases, beaconing consumes an increasing fraction of channel activity. For Disco to be able to scale to high densities, the beaconing rate must be reduced when nodes are in high density clusters. Disco currently allows the application to control duty cycle, beaconing mode (beacon and listen, or listen only), and other service parameters. Factoring out these policies from their underlying mechanisms makes sense since they are application-specific and no single adaptation policy is likely to satisfy a wide range of application needs.

Still, we can envision some approaches to beacon rate adaptation. In one scenario, nodes could track how often they hear neighbors and they probabilistically adjust whether to both beacon and listen during their on slots or just to listen. If nodes are just listening, then they would send beacons during their normally scheduled rendezvous slots with nodes in their neighbor table to ensure that routing links remain connected and synchronized, but they would not send other beacons. In this scenario, a neighborless mobile node would beacon while nodes in the cluster would (largely) listen. Upon discovering a new node, members of the cluster would gossip about their own neighbors with the new node, which perhaps would help the new node discover its neighbors more quickly.

4.7.2 Accelerating Discovery with Gossip

The current implementation does not make effective use of gossip even though beacons do include a neighbor count field (which indicates the number of one-hop neighbors in the neighbor table). We envision that in the future, if a node i receives a beacon from a node j with a neighbor count less than some threshold (e.g. two), then node i could send a longer beacon with (a random subset of) the entries of i's own neighbor table to help node j probabilistically speed up its own neighbor discovery attempts. We suggest that the speedup may be probabilistic in nature because i's neighborhood may or may not overlap j's neighborhood. The rationale for keeping beacons small normally, and sending long beacons infrequently, is that it optimizes for the common case: sending long beacons most of the time wastes bandwidth and energy as most beacons would not be heard in mobile, low-power, or low-density networks.

4.7.3 Secure Discovery

In some applications, it may not be advisable for mobile nodes to normally beacon (i.e. they only listen), and only respond if they first receive a peer's message that can be authenticated. A motivating application would be wireless sensors mounted on military vehicles or personnel. Beaconing would be undesirable because opposing forces might be able to detect and track these transmissions. Peer authentication, however, is tricky. It is not enough for a peer to digitally sign a beacon because such a beacon could be captured and replayed across an entire city using a high power transmitter. Including a timestamp in the beacon doesn't protect against a replay attack either. Identity, time, and location must be authenticated to ensure that a node only replies when it is verifiably close in both space and time to an authenticated peer. The Disco application programming interface needed to support secure discovery may require a tighter coupling or greater application-layer influence over the beacons.

4.8 Summary

This chapter presents a practical solution to the low-power, asynchronous neighbor discovery problem that arises in almost all low-power, mobile sensing applications. The solution is conceptually simple and easy to implement: nodes pick a pair of dissimilar primes such that the sum of their reciprocals is equal to the desired radio duty cycle. Each node increments a local counter and if a node's counter is divisible by either of its primes, the node turns on its radio for a single interval, whose length is the only global parameter. This simple protocol achieves discovery (or overlapping radio on-time) faster than other discovery protocols for a given duty cycle, allows nodes to independently select their own duty cycle, offers a provable upper bound on discovery latency, and performs better than expected in practice. These features provide a great deal of flexibility and provide support for both the symmetric and asymmetric interaction patterns that arise naturally during talking, docking, and flocking.

Disco, foremost, is a scheduling algorithm that ensures overlapping time slots for independently chosen duty cycles. What occurs during those slot is a different question. In this chapter, we evaluate Disco with duty cycles between 1% and 5%, using two slot lengths, 10 ms and 25 ms, and show that Disco works for a slot design in which a beacon is transmitted at both the beginning and the end of the slot. This particular design ensures bi-directional discovery between nodes, even when their slots are poorly-aligned. In the next chapter, we explore Backcast, an alternate link layer mechanism for synchronizing nodes by streamlining the slot design. Instead of bi-directional beacons, Backcast uses a uni-directional probe coupled with an automatically-generated hardware acknowledgement, to determine whether a neighbor is present.

Chapter 5

A Link Layer Synchronization Primitive

In this chapter, we present Backcast, a link layer synchronization primitive that allows a node to efficiently determine if any neighbors are present or whether they have any pending traffic. A backcast is a frame exchange in which a single *probe* frame may be acknowledged by zero or more identical *acknowledgement* frames (sent by zero or more neighbors). Although the acknowledgement frames may collide, as we discussed in Chapter 4, we show through analysis and experimentation that for certain physical layer modulation schemes, they are likely to interfere non-destructively, and so their superposition can still be successfully decoded by the radio. Backcasts are useful in a mobile or static context because they sidestep the ACK implosion problem that can occur when multiple neighbors respond to a single probe. Essentially, backcast provides the illusion of acknowledged anycast – an acknowledgedment ensures that at least one but possibly multiple neighbors successfully received the probe. In this chapter, we show that backcast is feasible using a commodity radio, both accurate and fast since it conclusively determines neighbor presence within 544 μs after a probe, efficient since it runs in constant time independent of the number of neighbors, and scalable since it works with many interfering acknowledgments. In Chapter 6, we build a backcast-based, receiver-initiated link layer that provides unicast, broadcast, discovery, wakeup, and pollcast.

5.1 Overview

The radio dominates the power budget in many low-power wireless systems, even when no communication occurs [44]. This happens because in most modern low-power radios, the power needed to listen for incoming packets, known as *idle listening*, is nearly the same as the power draw while actively transmitting or receiving (although there are some exceptions [26]). Since energy efficiency is the primary design consideration in these systems, reducing the cost of idle listening is a basic design goal.

Low-power link layers deal with this problem by aggressively duty cycling, or turning on and off, the radio to reduce average power. Synchronous link protocols coordinate their sleep and wake times, and a node only communicates when it knows the recipient will be awake and able to communicate [148, 149, 39]. Asynchronous link protocols, in contrast, do not coordinate their sleep and wake times and instead use one of two basic techniques, low-power listening (LPL) [71] or low-power probing (LPP) [106], to synchronize communications. An LPL transmitter sends a long preamble or repeatedly retransmit a frame; an LPL receiver samples the channel to detect the presence or absence of energy, and then stays awake if channel energy is detected. An LPP receiver transmits a request-to-receive (RTR) probe and listens for a response; an LPP transmitter first listens for an RTR probe, and after receving one, transmits a data frame.

The existing link layer synchronization techniques are not ideal for low-power, mobile sensing applications for a variety of reasons. Synchronous protocols are quite energy efficient when traffic workloads are known *a priori* or the network topology is relatively static. However, they are ill-suited to the unpredictable data traffic and topology dynamics that result from mobile nodes. Asynchronous LPL techniques also have two major drawbacks. First, the long preambles occupy the channel for an extended duration which can run afoul of regional or national regulations (e.g. in Japan) and they scale poorly to low duty cycles since their length is inversely proportional to the radio duty cycle. Second, in a crowded spectrum which is increasingly common both indoors and out, LPL samples can register many false alarms. Although asynchronous LPP techniques can address both of the problems that LPL faces, LPP has its own problems when dealing with low-power, mobile networks. First, a mobile node may be unaware of which neighbors, if any, are within range at any particular time, so it cannot efficiently address each individual neighbor. Second, even if the neighbors are known, it is inefficient to query each neighbor individually, and querying them all at once with a broadcast (a sort of "acknowledged anycast") could lead to data (or acknowledgement) implosion.

In this chapter, we show how acknowledged anycast can be implemented quite efficiently by using superposition. An initiator transmits a probe frame to a multicast or broadcast address, all nodes that match the destination address acknowledge the frame concurrently, and the initiator correctly decodess the superposition of the multiple acknowledgment frames to learn that at least one node received the probe despite the collision. We term such an exchange a *backcast* and show that it offers a wireless boolean-OR service abstraction: a node can pose a true or false question to its neighbors, and each neighbor votes false by ignoring the packet or votes true by acknowledging it. In a disconnected mobile setting, no neighbors will be in range, so the node can turn off its radio after the ACK timeout has passed. In pairwise communications, an acknowledgement will be followed immediately by data. And for the many-to-one case in a single-hop topology, the acknowledgement will be successfully received, so the node will know to remain awake while its neighbors contend to send data traffic. Such a primitive enables efficient, robust implementations of LPP.

Apart from the acknowledged anycast semantics, a backcast allows a node to robustly poll the channel for neighbors or pending traffic. The most consequential decision that a low-power link layer makes after polling the channel is whether to stay awake or go back to sleep. If the link decides that traffic is pending when none exists – *a false positive* – then the radio will remain on, wasting the receiver's energy. If the link decides that no traffic is pending when some is – *a false negative* – then the transmitter's energy is wasted, communications latency increases, and packet goodput drops. Making a good decision about whether to stay awake or go back to sleep is a critical one, but it is not always an easy one for LPL protocols that sample the channel for energy, drawing a still sharper distinction between LPL and backcast-based LPP. First, for LPL systems, *interference* from external systems (*e.g.* an 802.11 network, a cordless phone, or a microwave) might be mistaken for legitimate radio activity. Second, a receiver might *overhear* a partial packet sent to a different node, and stay awake until it can conclude that the packet is destined elsewhere. Third, hidden terminals might cause packet *collisions* or the MAC carrier sense might mistake external interference for a packet collision, forcing the radio receiver to stay awake longer than required. A backcast-based synchronization primitive does not suffer from any of these problems, as we discuss in this chapter.

Furthermore, a reliable, efficient, and scalable acknowledged anycast service could enable or improve multiple link and network layer services. For example, a low-power, network wakeup service would be possible [106]. A low-power, receiver-initiated unicast service [131] that eliminates the long preambles common in today's LPL protocols [111] would also be feasible. Single-hop collaborative feedback [35] would benefit from the OR semantics of acknowledged anycast. Group testing protocols could be implemented using the backcast primitive. Finally, asynchronous neighbor discovery could be streamlined.

This chapter explores the degree to which an acknowledged anycast service can be implemented efficiently using off-the-shelf hardware – via a range of experiments based on the IEEE 802.15.4-compliant CC2420 [134] and AT86RF230 [11] radios. We implement backcast using the hardware automatic acknowledgments (auto-acks) available in all IEEE 802.15.4-compliant radios as follows. An initiator transmits a packet to a unicast, multicast, or broadcast address. All nodes that match the destination respond with identical acknowledgment frames automatically generated by the hardware. Although these auto-acks interfere, they usually do so non-destructively, so the initiator can decode their superposition.

The results show that a commodity radio can decode the superposition of a large number of identical acknowledgments with high probability and they suggest that an efficient and robust single-hop anycast service that does not suffer from ACK implosions is possible with at least the O-QPSK modulation scheme used in 802.15.4. Our results also suggest some important relationships between the signal strength and quality of acknowledgments, number of responders, and delay variation. In a controlled experiment with equal path loss and round trip times between the initiator and responders, we find that the two-responder case exhibits slightly worse signal quality and reception rates than all other cases. We also find that if path delay differences exceed one physical layer symbol time for otherwise equal power and equal path loss communications between three nodes, backcast success rate drops precipitously due to intersymbol interference, as expected. Backcast lays the groundwork for the link layer we present in Chapter 6.

5.2 Related Work

Backcast is a link layer synchronization primitive that is related to a range of physical layer concerns including beamforming and fading, and variety of link layer techniques including time synchronization and channel polling. Backcast can also serve as the basic synchronization primitive for receiver-initiated media access control protocols and it has applications to a broad set of link and network layers services including unicast, multicast, broadcast, discovery, wakeup, pollcast, and others.

Beamforming is a common technique in which two or more transmitters coordinate their communication, typically by adjusting the amplitude and phase of a signal while keeping the frequency constant, allowing transmitters to actively shape a beam to enhance or cancel the signal at particular points in space. In contrast, backcast does not control amplitude or frequency, and only loosely controls baseband signal phase (but not the carrier) so interactions can occur between superimposed signals that the beamforming literature does not consider. The channel fading literature does consider frequency variations – fading models assume a superposition of signals of varying amplitude and phase, and Doppler-spread induced frequency shifts – providing a more representative analytical model. Barriac et al. proposed a distributed extension to the beamforming technique, in which n nodes coordinate their RF transmissions to create steerable "beams" towards different receivers, leading to an n-fold gain in signal-to-noise ratio [14].

Hill [73] and Ringwald and Römer [114] showed that radio packet collisions are not always destructive, but instead they can be used to compute Boolean operations (e.g. bitwise OR) of the transmissions of multiple synchronized senders. Their scheme requires time coordination and a simple modulation scheme – on-off keying (OOK), making it less robust than FSK or PSK schemes. However, Whitehouse showed capture using mote-class FSK radios [147]. Backcast takes a step further, showing that it is possible to coordinate the transmissions of multiple nodes so that the superposition is non-destructive, both statistically and empirically. Coordination occurs when nodes synchronize their transmissions to the radio probes they receive. In this respect, backcast is similar to Reference Broadcast Synchronization (RBS), which uses packet broadcasts to synchronize node clocks [55].

Low-power, asynchronous, transmitter-initiated protocols often use low-power listening (LPL), originally proposed by Hill and Culler [71], to wake up the intended receivers. LPL-based protocols are very efficient under low traffic workloads since nodes turn their radios on only to sample the channel – an action that requires only few milliseconds. However, since nodes use channel energy levels to detect activity, LPL suffers from the overhearing problem when a node that happens to sample the channel during a frame transmission or interference erroneously turns on the radio. On the other hand, because backcast requires a node to correctly decode an acknowledgement to keep its radio on, it does not suffer from interference in the same way. On the other hand, sending a probe is inherently slower than sampling the channel and therefore LPL checks will always be more efficient. Finally, channel usage of backcast-based MAC protocols increases with node density even when no traffic is sent due to the periodic probes.

Some MAC protocols use receiver-initiated communications [62] to minimize the medium contention from transmitters, especially in the case of hidden terminals and for low duty-cycle networks in which the sleep interval is large. In RI-MAC, for example, nodes broadcast a probe frame on every wake up [131]. Nodes with pending data wait for the probe from the destination node. Musăloiu-E. et al. presented *low-power probing* (LPP) [106], a MAC protocol that implicitly uses the backcast primitive to provide fast feedback on whether pending data packets exist. Specifically, LPP nodes take advantage of the acknowledgments generated by the lower layers, such as the hardware acknowledgement from the 802.15.4-compliant radios. The IEEE 802.15.4e standard, under development in 2009, has explored receiver-initiated transmissions to reduce the long channel occupancy time of LPL better to conform to government regulations.

Demirbas et al. recently proposed *pollcast*, a primitive in which a node broadcasts a poll about the existence of a node-level predicate P and all nodes for which P holds reply simultaneously [35]. The poller detects one or more positive answers by reading its radio's Clear-Channel-Assessment (CCA) signal. While pollcast offers an ingenious approach for calculating predicates rapidly, the proposed mechanism has a number of limitations: simultaneous pollcasts within a 2-hop neighborhood would cause false positives as would interference from other networks (e.g. 802.11). Backcast provides a more robust primitive for implementing pollcast which can be used to implement the applications outlined in [35]. In the context of ad-hoc networks, Carter et al. proposed the *manycast* group communication primitive in which one client communicates simultaneously with k out of m group members [20]. Manycast requires a solution to the ACK implosion problem to efficiently discover available group members. Therefore, the synchronous superposition of acknowledgments we describe can be used as a building block of the manycast service.

Network-wide wakeup is an important primitive in low-power wireless networks. One approach to waking up a low-power network proposed by Lu and Whitehouse is to flood a wakeup command using LPL transmission techniques [95]. Unfortunately, this approach makes the network unusable for the duration of the wakeup. Dutta et al. proposed a different approach to this problem [44]. In their scheme, every node periodically transmits a beacon and then briefly listens for channel activity. If any channel activity is detected, the node remains awake, but if no activity is detected, the node goes back to sleep. To wake up the network, the initiator listens for a time equal to the beacon period to identify all one-hop nodes. Then, during the next such period, the initiator contacts each of its one-hop neighbors in turn. These neighbors then repeat this process for the two-hop neighbors, and so on. If two or more nodes attempt to contact the same node in a lower tier, the work conjectured that the concurrent transmissions may collide, but that the receiver would detect channel energy, remain awake, and give the transmitters a chance to enter backoff and compete. Using backcast, the superposition of the concurrently transmissions would be non-destructive, making the wakeup robust and not susceptible to external interference. Musăloiu-E. et al. proposed *low power probing* (LPP) as another solution to the wakeup problem [106].

5.3 Backcast Design

A backcast is a link-layer frame exchange in which a single radio frame transmission triggers zero or more acknowledgment frames that interfere non-destructively at the initiator. Figure 5.1 illustrates a backcast exchange involving three nodes.

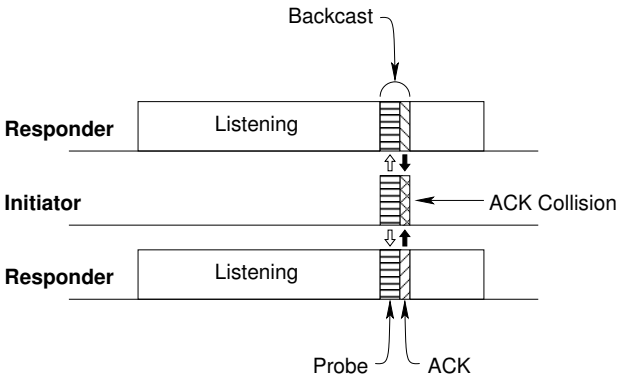


Figure 5.1: A backcast exchange involving three nodes. The backcast initiator transmits a probe frame that two responders acknowledge using identical ACK frames. Although the ACKs collide, they do so non-destructively, so the initiator can correctly decode their superposition.

The two responders have their radios configured to automatically acknowledge any received frames. The backcast exchange begins with the initiator transmitting a probe frame to the hardware broadcast address. Both responders receive the probe and they both transmit *identical* acknowledgments. Although these two acknowledgments collide at the initiator, as long as certain conditions are met, this collision is non-destructive, allowing the initiator to correctly decode the acknowledgment frame and conclude that at least one of its neighbors responded.

In addition to the broadcast address, a backcast probe can be sent to a multicast or unicast address, to which only a subset of the initiator's neighbors might respond. The choice of the destination address of a backcast probe depends on the radio's capabilities as well as the needs of the communications service using backcast. For example, the hardware broadcast address might be appropriate when waking up an sleeping network while a unicast address would be appropriate for communications with a single node.

The key to a successful backcast is that ACK collisions are non-destructive. This condition can hold due to power capture if one ACK frame has a higher power than the sum of the remaining ACK frames [10], or delay capture if one ACK frame arrives some period of time before the rest [33], or message retraining capture – a "message in message" model – where the radio attempts to resynchronize mid-packet if it detects a suddenly elevated energy level [104], or trivially if the radio uses an on-off keying (OOK) [114].

In this chapter, we suggest that backcast is possible under a much wider range of conditions than what capture alone would predict. In particular, we hypothesize that backcast is possible using minimum shift keying (MSK) and orthogonal quadrature phase shift keying (O-QPSK) modulation schemes for certain radio designs provided that: (i) inter-symbol interference resulting from different path lengths is limited, (ii) concurrent ACK frames do not cancel each other at the physical layer, (iii) the radio can automatically generate an ACK frame with an accurate and precise turnaround time, and (iv) the superposition of multiple ACKs is semantically meaningful (e.g., the ACKs are identical). Despite these constraints, we show that backcast works in practice under both controlled and realistic conditions using commodity radios.

The remainder of this section presents the backcast theory and design. We ground our discussion in the context of the IEEE 802.15.4 standard for Low-Rate Wireless Personal Area Networks (LP-WPAN) operating in the 2.4 GHz ISM band. Standards-compliant radios are available from several sources [134, 11] and used in popular sensor platforms [103, 29, 48]. We begin with a simple model where all nodes have identical transmission power, path loss, phase delay, and oscillator frequency. This simple model allows us to show that backcast is theoretically feasible. We then progressively relax these assumptions to show backcast can work in practice.

5.3.1 Simplified Model

Figure 5.2 shows the critical time constants of an 802.15.4 hardware automatic acknowledgement (auto-ack or HACK). In this figure, the backcast probe, labeled **P**, is a simply a standard data frame sent by the backcast initiator with the acknowledgement request bit set. The responder, upon receiving this probe frame, generates an auto-ack, labeled **A**. The 802.15.4 standard stipulates that the auto-ack must be generated precisely 12 symbol periods (192 μs) after the end of **P**. The auto-ack frame is 11 bytes long[1] and requires 352 μs to transmit. Typically, a responder would transmit a **DATA** frame with a short random delay after sending the **A** frame.

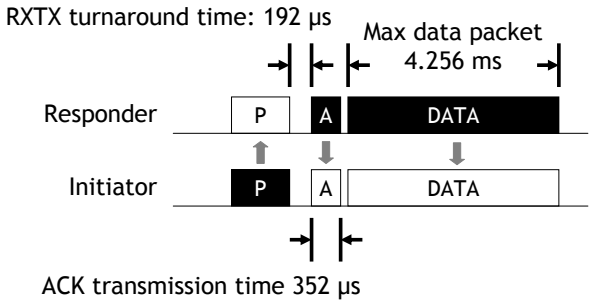


Figure 5.2: IEEE 802.15.4 acknowledgment timing details.

In the 802.15.4 standard, the data stream is divided into bytes, and each byte is divided into two *symbols* that are 4 bits each. Each symbol is mapped to one of 16 pseudo-random sequences that are 32 chips each [78]. The chip sequence is coded using *orthogonal quadrature phase shift keying* (O-QPSK) with half-sine pulse shaping, modulated with a carrier wave in the 2.4 GHz band, and transmitted at 2 Mchips/s (or 250 kbps data rate). This particular modulation technique employs continuous-phase frequency shift keying, has been well-studied in the literature, and is also known as *minimum shift keying* (MSK) [66].

The O-QPSK modulated signal, $s(t)$, that is transmitted by the radio at time t can be expressed as

$$s(t) = a_I(t)\cos\frac{\pi t}{2T}\cos(2\pi f_c t) + a_Q(t)\sin\frac{\pi t}{2T}\sin(2\pi f_c t)$$

where $a_I(t)$ is the in-phase (odd) chips of the data stream, $a_Q(t)$ is the quadrature-phase (even) chips of the data stream, $2T$ is the length of a half-sine pulse ($T = 0.5$ μs), and f_c is the carrier frequency (2.405 to

[1] An ACK frame is 11 bytes long: preamble (4), start-of-frame delimiter (1), length (1), frame control (2), sequence number (1), frame check sequence (2).

2.480 GHz in 5 MHz steps). The right-hand side of this equation can be simplified to

$$s(t) = \cos\left(2\pi f_c t + b_k(t)\frac{\pi t}{2T} + \phi_k\right)$$

where b_k is $+1$ when a_I and a_Q have opposite signs, b_k is -1 when a_I and a_Q have the same sign, and ϕ_k is 0 or π when $a_I = 1$ or -1, respectively [108].

Factoring out the $2\pi t$ makes the frequency of the transmitted signal much more clear

$$s(t) = \cos\left[2\pi t\left(f_c + b_k(t)\frac{1}{4T}\right) + \phi_k\right].$$

Since the only values of $b_k(t)$ are ± 1, the frequency of the transmitted signal alternates between $f_h = f_c + 1/4T$ and $f_\ell = f_c - 1/4T$, illustrating the continuous-phase, binary frequency shift keying nature of the modulation scheme. The frequency separation between these two tones is $1/2T$, or 1 MHz for 802.15.4 transmissions.

Consider a scenario in which two responders have identical transmission power, path loss, path delay, oscillator phase, and oscillator frequency. Since we assume that the ACK frames are identical, two responders that ACK the same probe will generate the same waveforms. Under these assumptions, the superposition of two (or more) ACK frames will simply be the superposition of multiple FSK signals, or simply the sum of two equal frequency sines (which is just another sine). For each doubling in the number of responders, a 3 dB increase in the received signal strength would be observed at the receiver.

5.3.2 Phase Offset

Now, consider a scenario in which two ACK frames arrive at the initiator out of phase. Such phase offsets in the received ACKs can occur for several reasons: responders may not be phase-synchronized, the path length between different responders and the initiator may not be equal, or the path lengths may not be an integer multiple of the carrier wavelength. In these cases, the signal received by the initiator will be the sum of two sines with an arbitrary phase offset. We continue to assume the signals have equal frequency. For the superposition of the ACK frames to be decoded, the sum of the sines will need to exceed the receive sensitivity (and SNR threshold) of the radio.

To determine the fraction of time the radio will be able to correctly receive the ACK frame, we need to consider the amplitude distribution for the sum of two sines, each with unit amplitude and random phase. In particular, we seek to address the question, for what fraction of possible phase offsets between two sines is their sum above the radio receive sensitivity? The complementary cumulative distribution function (CCDF) of the sum of sines provides the answer and is given by

$$f(x) = \frac{2}{\pi}\arccos\left(\frac{x}{2}\right).$$

over the interval $0 \leq x \leq 2$. The amplitude of the sum exceeds 1 (0 dB) two-thirds of the time, exceeds 1/2 (-3 dB) 84% of the time, exceeds 1/4 (-6 dB) 92% of the time, and exceeds 1/8 (-9 dB) 96% of the time, as Figure 5.3. If nodes pick neighbors with a few dB of link margin, then in the great majority of cases in which two responders are observed with equal power at the initiator, their arbitrary phase offsets will not result in the received signal cancelling or falling below the receive threshold. Conversely, this observation has implications for neighbor selection: if two neighbors are both operating *at* the receive sensitivity of the initiator, and they both ACK the same probe, then with non-trivial probability (1/3), the ACK will be lost.

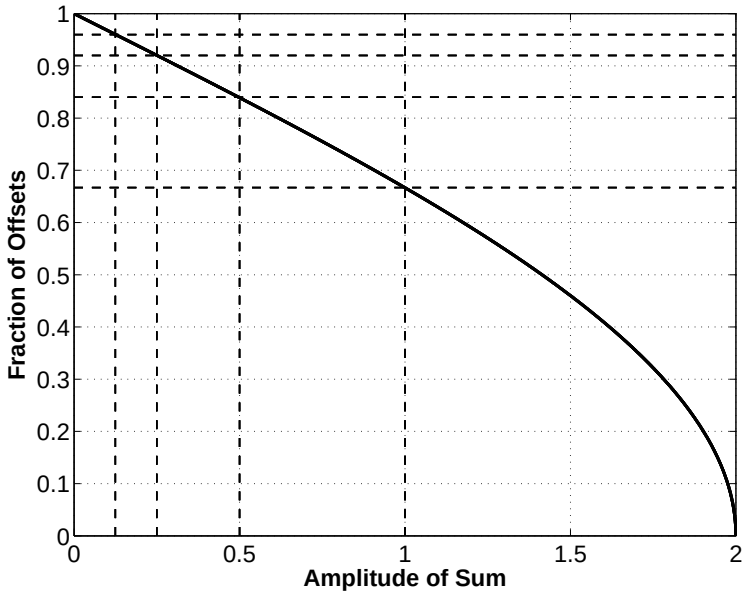


Figure 5.3: Cumulative CDF of the amplitude of the sum of two sines, each with unit amplitude and uniformly random phase. The amplitude of the sum exceeds 1 (0 dB) two-thirds of the time, exceeds 1/2 (-3 dB) 84% of the time, exceeds 1/4 (-6 dB) 92% of the time, and exceeds 1/8 (-9 dB) 96% of the time. These results suggest that colliding ACK frames with direct line-of-sight, equal frequency and power, but dissimilar phase, will rarely cancel. More importantly, if a few dB of link margin is reserved when selecting neighbors, ACK collisions will usually be decodable.

We briefly sketch the derivation of the cumulative CDF of the sum of sines. Let Φ be random variable representing the phase offset between a pair of sinusoids and be drawn from a uniform distribution over the interval $[0, \pi]$

$$\Phi = \mathcal{U}[0, \pi].$$

Let $f(t, \phi)$ be the sum of the two sines where t is time, ω is angular frequency, and ϕ is the phase offset

$$f(t, \phi) = \cos(\omega t) + \cos(\omega t - \phi).$$

Without loss of generality, let $wt = \phi/2$, which gives the following for the amplitude of the sum

$$f(\phi) = \cos\left(\frac{\phi}{2}\right) + \cos\left(-\frac{\phi}{2}\right).$$

Since cosine is a symmetric function, we can further simplify

$$f(\phi) = 2\cos\left(\frac{\phi}{2}\right).$$

We note that $f(\phi)$ is monotonic (in fact, strictly so) and defined only over the interval $[0, \pi]$. We use this observation to help derive the analytic formulation of the PDF, CDF, and cumulative CDF of the sum in the remainder of this subsection.

For a monotonic function $f(x)$ defined on the interval $[a, b]$, if $f(x)$ is strictly increasing (or decreasing), then $f(x)$ is invertible and the CDF (or complementary CDF) of $f(x)$ is given by $F(y)$ (or $F_c(y)$) as follows, where $f(a) \leq y \leq f(b)$

$$F(y) = \frac{1}{b-a} f^{-1}(y).$$

We start by inverting the domain, $[a, b]$, and range, $[f(a), f(b)]$, of the strictly increasing function, $f(x)$, and then mapping from the new domain, $[f(a), f(b)]$, to the new range, $[a, b]$, with $g(y) = f^{-1}(y)$. We can compute the probability that y falls in the interval $[m, n]$, where $f(a) \leq m \leq n \leq f(b)$ as

$$P\left[m \leq y \leq n\right] = \frac{1}{b-a}\left(g(n) - g(m)\right).$$

To determine the probability *density*, we normalize by the interval $n - m$, giving

$$p\left(y \mid m \leq y \leq n\right) = \frac{1}{b-a}\frac{g(n) - g(m)}{n-m}.$$

Substituting $n = m + \epsilon$, we have the following equation

$$p\left(y \mid m \leq y \leq m+\epsilon\right) = \frac{1}{b-a}\frac{g(m+\epsilon) - g(m)}{\epsilon}.$$

In the limit, the right-hand side can be reduced to the derivative of $g(y)$, giving the PDF

$$p(y) = \lim_{\epsilon \to 0} \frac{1}{b-a}\frac{g(m+\epsilon) - g(m)}{\epsilon} = \left(\frac{1}{b-a}\right)\frac{d}{dy}g(y).$$

The CDF, $F(x)$, is simply the indefinite integral of the PDF

$$F(x) = \int_{-\infty}^{x} p(y)dy = \frac{1}{b-a}g(x).$$

In the case of the amplitude of the sum of sines, $a = 0, b = \pi, f(x) = 2\cos(\phi/2), f(0) = 2, f(\pi) = 0$, and $f(x)$ is strictly decreasing. Inverting $f(x)$, we have

$$g(y) = f^{-1}(y) = 2\arccos\left(\frac{y}{2}\right).$$

Since $f(x)$ is strictly decreasing, the complementary CDF, $F_c(y)$ is given by

$$F_c(y) = \frac{2}{\pi}\cos\left(\frac{y}{2}\right).$$

and the CDF, $F(y)$, is given by

$$F(y) = 1 - \frac{2}{\pi}\cos\left(\frac{y}{2}\right).$$

The preceding analysis holds for short phase offsets in the carrier signal but not for long path delays, or rather differences in the path delays. The problem with long path delay differences is that they can lead to intersymbol interference. The chip rate of 2 Mchips/s, coupled with orthogonality properties of the pseudo-random sequences, governs the maximum phase difference permissible without destructive intersymbol interference. A path length difference of $(3 \times 10^8) \times T = 150$ m is needed to delay the signal by one chip time, T, and the delayed path would have to also be nearly equal in power – and unlikely scenario in practice.

5.3.3 Power Differences

The received signal strength at the initiator from two responders may not be equal because of differences in the radios or the RF channel due to attenuation or reflection. When this situation occurs, regardless of the phase offset or path delay of the signals, the result is the sum of two sines and the analysis is similar to the preceding discussion. The key difference is that with unequal signal power, complete cancellation does not occur. If one signal is stronger than the other by margin that exceeds the radio signal to interference and noise ratio (SINR) threshold (typically about 3 dB), then the stronger signal will be received.

5.3.4 Frequency Skews

Next, we consider the case in which the oscillator frequencies of the responders are not equal. The main challenge with this situation is the resulting amplitude modulation and beat frequency that occurs when two sines with slightly different frequencies mix. If the original sinusoids are equal-amplitude, then beats occur with frequency $f_b = |f_{c1} - f_{c2}|$. In practice, the frequencies f_{c1} and f_{c2} are limited to a tolerance range, $\pm f_\Delta$, around a nominal carrier frequency f_c. This limit on f_{c1} and f_{c2} implies

$$f_b \leq 2 \cdot f_\Delta$$

and the range of possible beat periods, T_b, is

$$T_b \geq \frac{1}{2 \cdot f_\Delta}.$$

The IEEE 802.15.4 specification requires radio crystals to offer ± 40 ppm stability over manufacturing variation and temperature range. This translates to a range of beat frequencies up to 192 kHz and beat periods no shorter than 5.2 μs, at the nominal carrier frequency of 2.4 GHz. The shortest beat interval is more than ten times longer than the chip time of 0.5 μs. However, because the length of an 802.15.4 acknowledgement frame is approximately 1 ms, a modest range of realizable beat frequencies could cause destructive interference on time scales larger than the chip time but smaller than the frame time.

5.3.5 Capture

We have argued that the key to a successful backcast is that ACK collisions are non-destructive. The preceding analysis progressively relaxed the constraints of equal phase, power, and frequency, and argued that backcast would fail in a narrow set of cases. We close this section with a discussion of capture, another important backcast enabler. ACK collisions can be non-destructive due to power capture if one ACK frame has a sufficiently higher power than the sum of the remaining ACK frames [10]. Delay capture might account for a successful backcast if one ACK frame arrives some period of time before the rest [33]. Message retraining capture – a "message in message" model – where the radio attempts to resynchronize mid-packet if it detects a suddenly elevated energy level could also enable a successful backcast [104]. A trivial form of backcast is possible if the radio uses an on-off keying modulation scheme [114] and the radio simply detects energy in the channel.

5.3.6 Analytical Performance Model

We now consider the case in which n responders send ACK frames that collide at the initiator. We assume that each responder transmits a sinusoidal carrier wave with the frequency and phase chosen at i.i.d.

randomly. The initiator receives the superposition of these of these signals, which when normalized, gives

$$f_n(t) = \frac{1}{n}\sum_{i=1}^{n}\cos(w_i t + \phi_i).$$

Since w_i and ϕ_i are random variables, $f_n(t)$ is a random scalar, and f_n is an empirical process. For now, let us assume that $p(w)$ and $p(\phi)$ are uniform distributions

$$p(w) = \mathcal{U}[w_0 - \delta, w_0 + \delta]$$

and

$$p(\phi) = \mathcal{U}[-\pi/2, \pi/2].$$

We wish to determine the shape of f_n as n grows. By the law of large numbers, as $n \to \infty$, for a fixed value of t, $f_n(t)$ converges to its expectation, $\mathbb{E}f_n(t)$. Under more restricted conditions, one can also show that the *function* f_n converges to the *function* $\mathbb{E}f_n$ (in, say, an RMS sense). In either case, for our purpose, it suffices to examine $\mathbb{E}f_n(t)$

$$\mathbb{E}\, f_n(t) = \mathop{\mathbb{E}}_{w,\phi}\cos(wt + \phi).$$

We can abstract away the influence of the phase. By iterated expectation,

$$\begin{aligned}
\mathbb{E}\, f_n(t) &= \mathop{\mathbb{E}}_{w}\mathop{\mathbb{E}}_{\phi}\cos(wt+\phi) \\
&= \mathop{\mathbb{E}}_{w}\frac{1}{\pi}\int_{-\pi/2}^{\pi/2}\cos(wt+\phi) \\
&= \mathop{\mathbb{E}}_{w}\frac{1}{\pi}\sin(wt+\phi)\Big|_{\phi=-\pi/2}^{\phi=\pi/2} \\
&= \mathop{\mathbb{E}}_{w}\frac{1}{\pi}\left[\sin(wt+\pi/2) - \sin(wt-\pi/2)\right] \\
&= \mathop{\mathbb{E}}_{w}\frac{1}{\pi}\left[\cos(wt) + \cos(wt)\right] \\
&= \frac{2}{\pi}\mathop{\mathbb{E}}_{w}\cos(wt),
\end{aligned}$$

which shows that randomizing the phase simply scales the signal in expectation, as one would expect based on our prior analysis.

Now, to identify the effect of the random frequency

$$\begin{aligned}
\mathbb{E}\, f_n(t) &= \frac{2}{\pi}\mathop{\mathbb{E}}_{w}\cos(wt) \\
&= \frac{2}{\pi}\frac{1}{2\delta}\int_{w_0-\delta}^{w_0+\delta}\cos(wt) \\
&= \frac{1}{\pi\delta}\frac{1}{t}\sin(wt)\Big|_{w_0-\delta}^{w_0+\delta} \\
&= \frac{1}{\pi\delta}\frac{1}{t}\left[\sin((w_0+\delta)t) - \sin((w_0-\delta)t)\right] \\
&= \frac{1}{\pi\delta t}\left[\sin(w_0 t)\cos(\delta t) + \cos(w_0 t)\sin(\delta t) - \sin(w_0 t)\cos(\delta t) + \cos(w_0 t)\sin(\delta t)\right] \\
&= \frac{2}{\pi}\frac{\sin(\delta t)}{\delta t}\cos(w_0 t).
\end{aligned}$$

The preceding analysis shows that the sum of sinusoids with uniformly random phase and frequency converges to a sinc modulated at the carrier frequency w_0. Intuitively, this makes sense. The Fourier transform of sinc is the symmetric rectangular function in the frequency domain. Modulating the sinc with a cosine function simply shifts its Fourier transform, the rectangular pulse, to $-w_0$ and w_0.

The analysis assumes that w was drawn from a uniform random distribution, but for many physical phenomena, like crystal oscillator frequencies, the distribution may not be uniform; rather it may be Gaussian. We now show that the prior analysis generalizes to a symmetric distribution. Suppose that ϕ is drawn uniformly as before, but w is drawn from a distribution $p(w)$ that is a symmetric function about some w_0. In other words, p is an even function shifted by w_0: $p(w) = q(w - w_0)$, and $q(\delta) = q(-\delta)$ for all δ. Then

$$\begin{aligned}
\mathbb{E}\, f_n(t) &= \frac{2}{\pi}\, \mathop{\mathbb{E}}_{w} \cos(wt) \\
&= \frac{2}{\pi} \int_{-\infty}^{\infty} p(w) \cos(wt)\, dw \\
&= \frac{2}{\pi} \int_{-\infty}^{\infty} p(w_0 + \delta) \cos((w_0 + \delta)t)\, d\delta \\
&= \frac{2}{\pi} \int_{-\infty}^{\infty} q(\delta) \left[\cos(w_0 t) \cos(\delta t) - \sin(w_0 t) \sin(\delta t)\right]\, d\delta \\
&= \frac{2}{\pi} \cos(w_0 t) \int_{-\infty}^{\infty} q(\delta) \cos(\delta t)\, d\delta - \frac{2}{\pi} \sin(w_0 t) \int_{-\infty}^{\infty} q(\delta) \sin(\delta t)\, d\delta.
\end{aligned}$$

The second integral is 0 because q is even and $\sin$ is odd. The first integral is the Fourier transform of q, which we denote $\mathcal{F}[q]$. So, in general, we have

$$\mathbb{E}\, f_n(t) = \frac{2}{\pi} \mathcal{F}[q] \cos(w_0 t),$$

which shows that the superposition of the random signals rides $\mathcal{F}[q]$ at the carrier frequency w_0. This shows that our analysis generalizes to symmetric distributions of carrier frequencies (e.g. a normal distribution).

5.4 Backcast Implementation

To evaluate the feasibility and performance of our design, we implemented Backcast using the nesC programming language [63], TinyOS operating system [72], and Epic [48], Telos [112], and Iris [29] sensor nodes. The Epic and Telos include the Texas Instruments CC2420 [134], while Iris includes the Atmel AT86RF230 [11]. Both of these radios are IEEE 802.15.4-compliant. The 802.15.4 protocol is ideal for implementing backcast because it provides hardware support for generating automatic acknowledgement.

The 802.15.4 MAC defines a frame control field that includes an acknowledge request flag. If a receiver is configured for automatic acknowledgments, then an acknowledgment frame is transmitted after twelve symbol periods (192 μsec) for all incoming frames that meet three conditions: they (i) have the acknowledge request flag set, (ii) are accepted by the radio's address recognition hardware, and (iii) contain a valid CRC. Acknowledgments are transmitted without performing clear channel assessment and have the following fields: preamble, start-of-frame delimiter, length, frame control, sequence number, and frame check sequence. Notably absent from this list is a source address, ensuring that all ACKs for a given sequence number are identical.

5.4.1 Application Programming Interface

```
interface Backcast {
  // Turn on the radio
  command error_t start();
  event   void    startDone(error_t err);

  // Turn off the radio
  command error_t stop();
  event   void    stopDone(error_t err);

  // Enable hardware automatic acknowledgements
  command void    enable(bool enable);

  // Set local hardware address and sync with the radio
  command void    setShortAddr(uint16_t addr);
  command error_t sync();
  event   void    syncDone(error_t err);

  // Perform a backcast poll and receive a yes/no response
  command error_t poll(uint16_t addr, message_t* msg);
  event   void    yes(message_t* msg);
  event   void    no(message_t* msg);
}
```

Figure 5.4: Backcast programming interface.

5.4.2 CC2420 Radio

If the CC2420 is configured to automatically generate a hardware ACK frame, then an ACK frame is transmitted 192 μs after all incoming frames satisfying three conditions: they (i) have the acknowledgement request flag set, (ii) are accepted by the radio's address recognition hardware, and (iii) contain a valid CRC.

The CC2420 hardware address recognition accepts only those frames where the destination addresses matches the local address or the destination address is the hardware broadcast address (0xFFFF).

5.4.3 CC2520 Radio

The CC2520 can automatically compare the source address of a received frame with entries in an on-chip table. The CC2520 can be configured to automatically generate an ACK frame if a match is found. The radio can automaticallly set the *frame pending* bit in the ACK frame to indicate pending traffic if the radio's AUTOPEND function is enabled.

5.5 Evaluation

This section evaluates the reliability, efficiency, and performance of Backcast under both carefully-controlled laboratory settings and more realistic indoor settings. We also characterize the energy profile of Backcast, integrate it into low-power probing (LPP), and compare its performance against low-power listening (LPL). We show that backcast works on two different radios from two different vendors, has a narrow range of failure cases, provides high energy- and channel-efficiency, and provides a strong foundation upon which to build a range of important link layer services for low-power networks.

In the experiments that follow, signal strength is measured by the radio over the first eight symbols of an acknowledgement (ACK) frame and reported as the received signal strength indicator (RSSI) in dBm. Signal quality (LQI) is also measured by the radio over the first eight symbols and is reported as a 7-bit unsigned value that can be viewed as the average correlation value or chip error rate (values near 100 indicate an excellent link).

5.5.1 Performance in a Controlled Two-Node Case

We first explore how both delay and loss differences in the signal path affect ACK reception rate.

Methodology

Figure 5.5 presents the setup for this experiment. Two nodes, an *initiator* and a *responder* (both Telos B) are connected to each other through a pair of circulators and a wireless channel emulator. A circulator is essentially an RF splitter that provides a low-loss RF path between some terminals (1-to-2, 2-to-3, and 3-to-1) but a very high-loss path between other terminals (1-to-3, 2-to-1, and 3-to-2). Circulators are used to split a single bi-directional RF path into two unidirectional paths. We use the D3C2060 circulator from DiTom Microwave. A wireless channel emulator allows a complex RF environment, including attenuation, delay, fading, Doppler shift, and multipath, to be evaluated in a laboratory setting. We use the Spirent SR5500 wireless channel emulator in these experiments. The SR5500 allows each channel to be composed of several independent paths, each with its own delay and attenuation.

Effects of Dissimilar Path Delay

To evaluate the effect of path delay on destructive intersymbol interference, the ACK channel from the responder to the initiator (Channel 2) is split into two equal paths inside the channel emulator. The delay in the second path is swept from 0 to 1 μs in 10 ns steps. For each delay step, the initiator transmits 100 packets to the hardware broadcast address, at 125 ms intervals, and logs the RSSI, LQI, and sequence number of the resulting acknowledgments. The results are shown in Figure 5.6 and indicate intersymbol interference becomes destructive between 500 and 600 ns, as expected. Note that a delay of 500 ns corresponds to a path difference of 150 m. Normally, such differences in path lengths would correlate with significant differences in received power as well.

Effects of Dissimilar Path Attenuation

To explore the effect of power capture [10] on backcast performance, the second path component in Channel 2 is delayed by 8000 ns (1/2 of the 802.15.4 symbol time). The attenuation for this second path is swept from 0 to 3.5 dB in 0.1 dB steps. As a result, the initiator receives the superposition of two (identical)

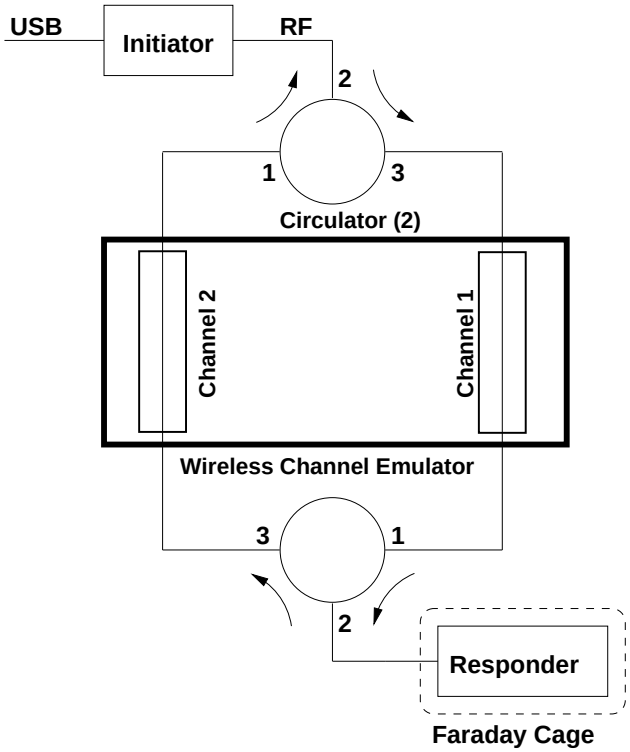


Figure 5.5: The experimental setup used to measure the effect of path delay and pass loss differences on backcast packet reception rates. A Spirent SR5500 wireless channel emulator allows path delay and loss to be finely controlled. A pair of circulators are used to split the transmit and receive channels and a faraday cage prevents over-the-air leakage between the initiator and responder.

frames, delayed by 8000 ns, over a range of SINR values. Figure 5.7 shows the results. When the first frame arrives with 3 dB or higher power, it will be decoded consistently. This figure establishes that power capture dominates (and explains) the backcast phenomenon when the strongest responder's power exceeds the sum of the remaining nodes by more than about 3 dB.

5.5.2 Performance in a Controlled Multinode Setting

Our next experiment explores how different responder configurations affect the acknowledgments' signal strength and quality.

Methodology

We first explore how the RSSI and LQI of acknowledgment frames are affected as the number of responders increase in a controlled setting. Figure 5.8 presents the setup for this experiment. Eight nodes are sequentially turned on so that the number of responders monotonically increases from one to eight. In each of the eight trials, the initiator transmits 100 packets to the hardware broadcast address, at 125 msec intervals, and logs the RSSI and LQI values of the resulting acknowledgments.

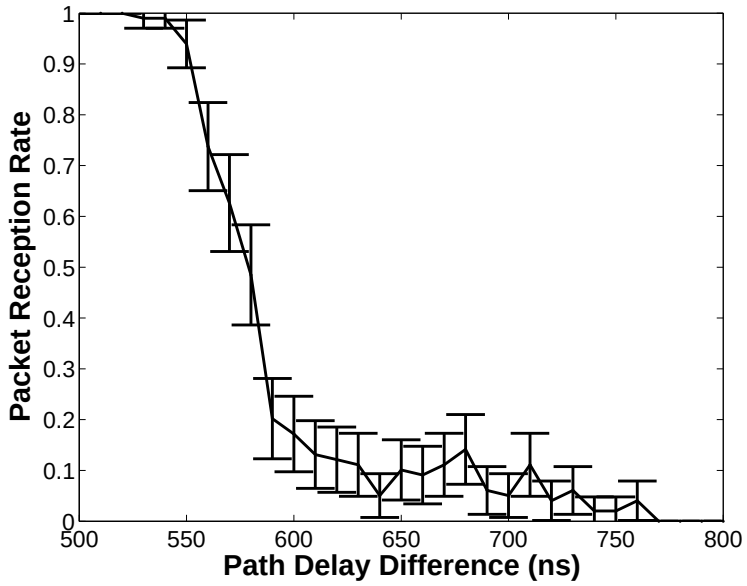


Figure 5.6: The onset of destructive inter-symbol interference. Packet reception rate falls sharply as the delay difference in two paths exceeds 0.5 μs.

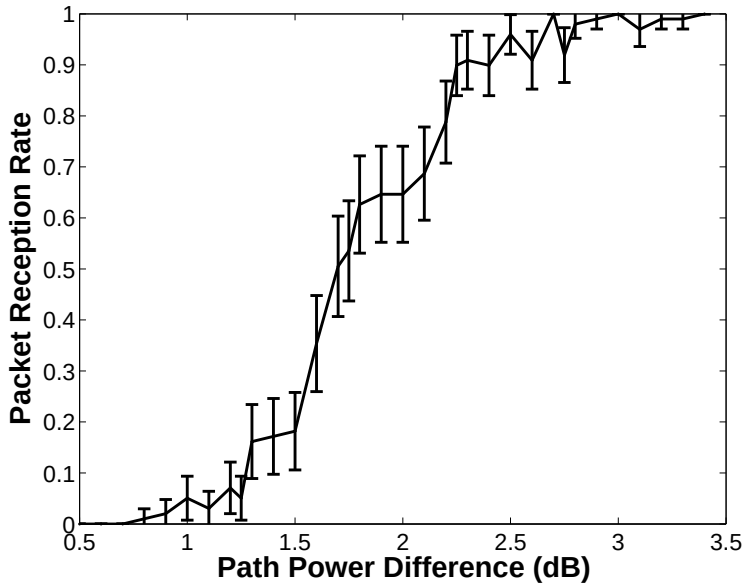


Figure 5.7: The effect of power capture. When two frames collide, the first frame to arrive will be decoded correctly if its receive power is 3 dB higher than the second frame.

The results are shown in Figure 5.9 and indicate that median RSSI values increase, median LQI values stay nearly constant, both values show greater variance, and few acknowledgments are lost. The next two subsections discus these results in detail.

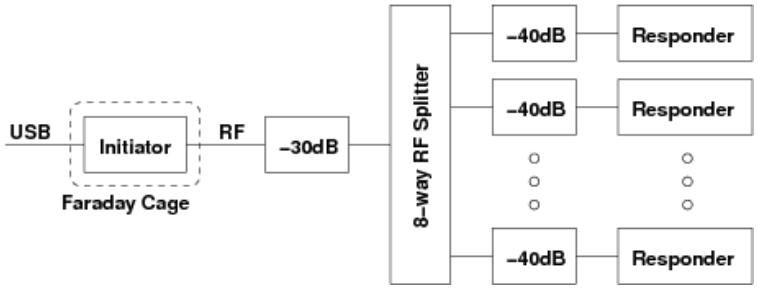


Figure 5.8: Experimental setup for the controlled tests. An initiator is connected via a 30-inch, 50 Ω RF cable and a 30 dB attenuator to the common port of an 8-way RF splitter. The other splitter ports are connected via 6-inch, 50 Ω RF cables and 40 dB attenuators to responders. A Faraday cage around the initiator limits over-the-air RF leakage.

Effect on Received Signal Strength

As the number of colliding acknowledgments increases, so does the median RSSI value. This trend is not surprising since for every doubling of nodes, an additional 3 dB of power is injected into the channel (assuming nodes transmit at nearly equal power levels and are equally distanced from the initiator). What is slightly more surprising is that RSSI variance is substantial and spans a range of 10-20 dB, which is both below and above the single node case, and that the distribution of values in the two-node case has many outliers. These results suggest that elements of both constructive and destructive interference of the carrier signal may be at play. When three or more acknowledgments collide, both the outliers and RSSI variance *decrease*, suggesting that the statistical superposition of an increasing number of signals diminishes destructive interference, possibly due to the central limit theorem.

Effect on Link Quality Indicator

As the median LQI value is largely independent of the number of nodes in the controlled setting (except for the two node case) and it shows a slight decrease in the more realistic setting (computed, but not shown). Since LQI is inversely correlated with chip error rate, the data show that most acknowledgments are decoded with relatively few chip errors, even when a dozen acknowledgments collide. The data suggest that acknowledgment collisions are rarely destructive and in most cases not particularly corrupting either. LQI values show a lower median value for two responders than they do for either one or more than two responders, suggesting once again that elements of both constructive and destructive interference of the carrier signal may be at play. The RSSI distributions are largely symmetric with few outliers but the LQI distributions are left-tailed. This observation suggests that although collisions rarely improve the chip error rate, they can make it worse.

5.5.3 Performance in a More Realistic Setting

Next, we explore how backcast performs in a more realistic university testbed setting. The testbed is located in an office building and contains 47 Telos motes. For this experiment however, we used only 12 nodes approximately situated at the same distance from the initiator. These experiments compare the performance of hardware-generated acknowledgments (HACKs), software-generated acknowledgments (SACKs), and

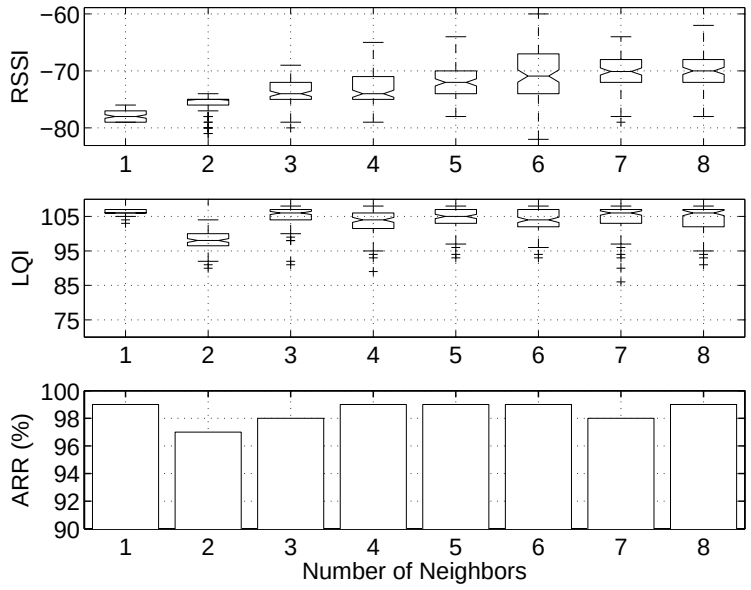


Figure 5.9: Results of the controlled experiments. The received signal strength (RSSI), link quality indicator (LQI), and acknowledgment reception rate (ARR) are shown for each trial. The median RSSI value rises but it also exhibits greater variance. The median LQI value stays nearly constant or falls slightly, but the LQI values exhibit a longer left-tail. ACK reception stays largely unchanged.

HACKs with randomized preamble lengths of between 3 and 16 bytes (VP-HACKs) that start at the same time but may end at different times. HACKs are automatically generated by the radio, while SACKs require the host processor to intervene, introducing considerable delay and jitter. We introduce VP-HACKs to explore how acknowledgments with smaller delay variations than SACKs interfere at the initiator. Note that while SACKs have non-uniform delays due to software processing, the VP-HACKs are all delayed by an integer multiple of the symbol time (composed of 32 chips) and the symbols themselves are orthogonal codes.

In each experiment, 500 packets are transmitted at 125 msec intervals. This procedure generates a gradual increase in the number of colliding ACK frames. The LQI and acknowledgment reception rates are shown in Figure 5.10. The results show that HACK and SACK LQI values exhibit higher variance and volatility as responders increase. Both HACKs with random preambles and SACKs exhibit quickly decreasing LQI and ARR values, while HACKs incur practically no loss.

While the results of the prior section suggest some important relationships between signal strength and quality of acknowledgments, number of responders, and delay variation. Further analysis, described below, supports our hypothesis that the capture effect alone cannot explain the surprisingly robust performance of backcast.

The data show that hardware acknowledgments exhibit negligible loss rates with no fewer than twelve concurrent packets, while software acknowledgments approach very high loss rates with just six or seven concurrent acknowledgments, as well as a substantial decline in link quality with just three or four acknowledgments. In between these two extremes are the variable-length preamble HACKs (VP-HACKs). The two distinctions between SACKs and VP-HACKs are in timing and composition. First, SACKs are delayed by

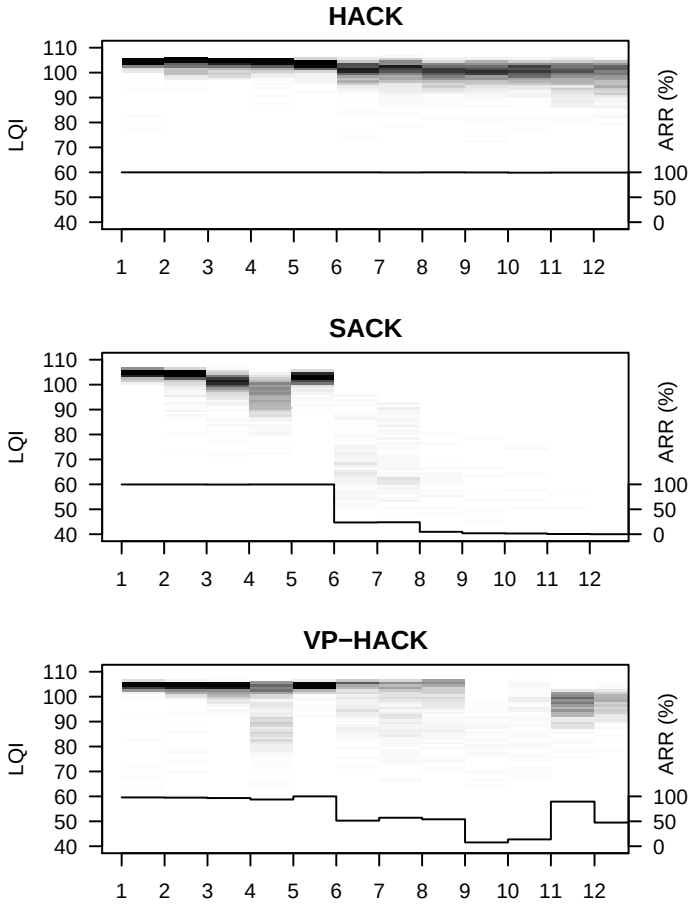


Figure 5.10: Backcast in a realistic environment, using hardware and software acknowledgments. The data are shown as a two-dimensional histogram; darker areas indicate a higher density of LQI samples. The lower line shows the average acknowledgment reception rate (ARR).

multiples of the CPU clock cycle since a SACK requires software processing, but a VP-HACKs are delayed by an integer multiple of the symbol time. Since the symbols are chosen from an orthogonal set, this may explain the better performance of VP-HACKs compared with SACKs, despite the fact that VP-HACKs collide more frequently and are not even identical. Since these three types of acknowledgments differ in the delay and jitter of their transmissions, we argue the capture effect alone cannot explain the surprisingly robust performance of HACK-based backcasts.

5.5.4 Large Scale Performance in a More Realistic Setting

We now explore how backcast performs in a more realistic setting *at scale.* The testbed consists of Telos B nodes and it is located in an office building with a typical RF environment. For this experiment, 94 nodes within radio range of an initiator are programmed to automatically acknowledge all backcast probes. In this experiment, 500 frames are transmitted at 125 msec intervals. This procedure generates a gradual

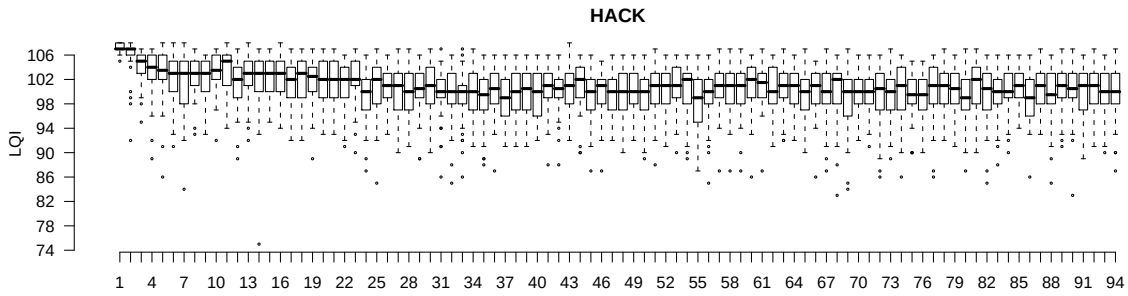


Figure 5.11: The effect on LQI as the number of concurrent ACKs increases from 0 to 94 in a typical indoor deployment setting. The median value of LQI falls quickly for the first six nodes and then falls slowly. Beyond approximately 30 nodes, the LQI values stabilize at approximately 100. The data suggest that even in the presence of a large number of ACK collisions, the receiver can successfully decode the ACK frame. Note the y-axis ranges from 74 to 106.

increase in the number of colliding ACK frames. The LQI values are shown in Figure 5.11 and are inversely related to the chip error rate of the received ACK.

The results show that the median value of LQI falls quickly for the first six nodes and then falls slowly. Beyond approximately 30 nodes, the LQI values stabilize at approximately 100. The data suggest that even in the presence of a large number of ACK collisions, the receiver can successfully decode ACK frames, even when no single ACK frame's power dominates. These data suggest that elements of both constructive and destructive interference of the carrier signal are at play. This result is not surprising since the carrier signals are neither synchronized in phase nor frequency across these 94 nodes. Rather, the carriers are generated locally by each node from a free-running crystal oscillator. These data show that the statistical superposition of an increasing number of signals does not lead to destructive interference and that backcast provides a robust link layer primitive.

5.5.5 Energy Microbenchmarks

All Backcast-based services are built using the *probe* primitive. Figure 5.12 shows the trace of this primitives as well as its energy costs. These data are collected by capturing the voltage drop across a 10 Ω resistor in series with a 3 V power supply using a Tektronix TDS3014 digital storage oscilloscope. Figure 5.13 shows the trace of the clear channel assessement (CCA) used in low-power listening protocols. Although backcast is more expensive, it does not suffer from false alarms like the CCA primitive does.

5.6 Discussion

This section reflects on how standards and radio hardware might evolve to better support backcast-enabled services. We also discuss some fundamental limitations of this approach.

5.6.1 Standards Implications

IEEE 802.15.4 ACK frame is considered a poor design by many because the ACK frame is not encrypted and, since no source or destination address field exists, the frame could be misinterpreted in the case

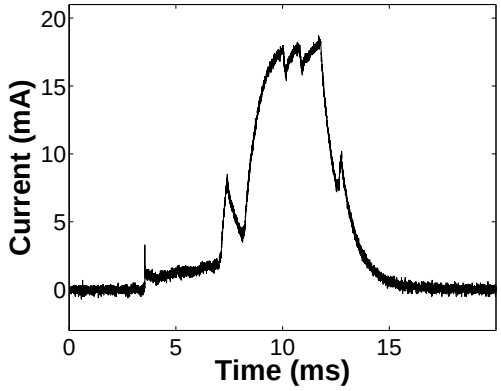


Figure 5.12: Backcast probe current profile. A probe uses 253 μJ to conclusively determine whether a neighbor is present or traffic is pending.

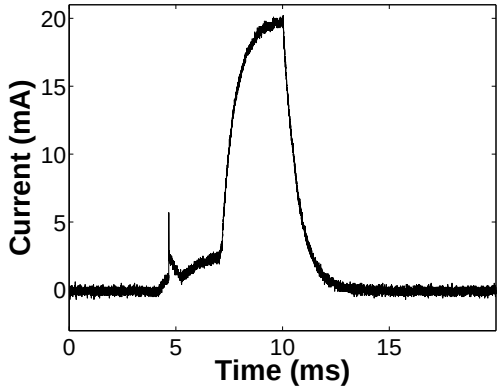


Figure 5.13: Clear channel assessment (CCA) current profile. A CCA is the fundamental link primitive in low-power listening protocols.

of hidden terminals both transmitting a packet with the same sequence number but with the receiver only able to decode one of the them, but appearing to acknowledge both. As a result, many MAC implementations avoid using the hardware auto-ack feature of the standard. This paper argues that there are situations where such anonymous ACKs are useful, namely when identical ACKs must be concurrently transmitted as backcast requires. We do note, however, that such anonymous ACK frames are not necessary if the radio can: (i) quickly change its hardware address, (ii) generate deterministic ACK frames, and (iii) enter a promiscuous mode in which it acknowledges all data frames (needed for broadcast).

As an aside, one reason that ACK frames are not encrypted is because this decision allows a simpler radio design: all AES encryption operations can be offloaded to the host processor, making the radio less expensive. For example, the Atmel AT86RF230 radio does not provide hardware support for AES [11]. Hardware auto-acks, in general, cannot be offloaded to the host processor because of the strict timing needed between a data frame and its auto-ack, and so the ACK frame is sent in the clear. In the future, this standard might evolve to allow ACK frame encryption. If this does occur, nodes can still generate identical encrypted ACK frames provided that the same seed is used. This provides one challenge and suggests that the initiator might consider adding the seed (or challenge) to the probe frame and requiring the hardware to be aware of its presence.

5.6.2 Limitations

There are two fundamental limitations to this approach. First, since backcast is a primitive for receiver-initiated communications, the receiver must probe the channel periodically for transmissions. This makes backcast-based services fundamentally less channel efficient under no data conditions than transmitter-initiated ones that listen quietly when no traffic is present. Despite the channel inefficiency for probe traffic, we note that many protocols beacon periodically for neighbor discovery or routing updates and that these transmissions incur their own channel inefficiencies when using low-power, transmitter-initiated schemes.

Backcast may be unsuitable for networks with high node density, short communication latency, or low-probability of detection requirements. The first two issues can be somewhat mitigated by using a different channel for the probes and auto-acks than for the actual data transmissions, since the initial probes can sent on a hailing or pilot channel. The latter issue is more severe: backcast is fundamentally at odds with stealthy networks since nodes cannot just listen quietly. Rather, *listening* really means probing which requires transmissions.

The second fundamental limitation to this approach is that backcast's primitive operation, a channel *probe*, is inherently more expensive than the channel *sample* primitive in transmitter-initiated protocols. Sending a probe frame and listening for an ACK will always require more time than sampling the channel. However, the benefits of backcast, namely its fixed energy cost, low false alarm rate, and efficient multiplexing ability, underscore a familiar theme in systems and networking research: optimal solution that work over a narrow range often perform more poorly over the diversity of workloads seen in practice.

Despite these drawbacks, the major benefit of backcast probes over channel samples is that probes do not suffer false alarms in the way that channel samples do. Indeed, a recent paper reported that the effective duty cycle of a low-power listening protocol was 2.5 times higher than expected due to overhearing 802.11 b/g beacons. Although nodes were programmed to operate at a 2.2% duty cycle, which they maintained while operating on a different channel than a nearby access point, when the 802.11 and 802.15.4 channels overlapped, the effective duty cycle increased to 5.6%, making efforts to predict battery life difficult [61]. Our findings concur with this observation. Conversely, long preambles that congest the channel with LPL are replaced by listening, which improves channel efficiency.

5.7 Summary

The central message of this chapter is that while it is relatively easy to optimize for a narrow range of operating conditions, like modern low-power listening protocols do, it is more difficult to find general solutions that work well across a broad spectrum of conditions. This chapter argues that the backcast primitive is better suited to the range of services needed by modern low-power wireless systems: low duty cycles, predictable operation, channel efficiency, and support for bursty workloads. This work paves the way for new research in the design of radio hardware, MAC sub-layer primitives, MAC-layer services, and performance studies to more deeply assess the utility and performance of this approach in meeting emerging application needs like mobile sensing or ultra-low duty cycles.

Chapter 6

Mobility-Aware Data Link Control

In this chapter, we present HotMac, a receiver-initiated link layer that offers unicast, broadcast, wakeup, discovery, and pollcast services built around the backcast link layer synchronization primitive. Using backcast, HotMac is able to provide services that have predictable duty cycles, high channel efficiency, and good support for bursty workloads, while multiplexing the many demands of different services on top of a single mechanism. HotMac works well across a broad spectrum of workloads, supporting both mobile-to-mobile and mobile-to-static communications, and offers many attractive features including ultra-low duty cycle support, high channel efficiency, and mobility-aware, power-proportional operation. In particular, HotMac supports mobility by beaconing and listening on motion triggers, integrating asynchronous neighbor discovery, and offering agile link prediction. We find that the backcast-based HotMac services are robust across a range of environmental conditions, provide a balance between false alarms and false negative, and provide competitive performance on a range of both low-level microbenchmarks and high-level macrobenchmarks. Although originally designed to support ultra low-power operation, especially in mobile networks, we show that HotMac has much to offer traditional static workloads.

6.1 Overview

The vision of a future in which sensing, communication, and actuation are densely deployed has received much attention in the past decade and holds the promise of dramatically improving industrial efficiencies. However, the radio dominates the energy budget of many current low-power wireless systems *while doing nothing*, and this represents a significant barrier to long term deployment. This is due to the fact that in most modern low-power radios, the power necessary to listen for incoming packets, known as *idle listening*, is nearly as large as the power draw while transmitting or receiving. Since energy is the primary constraint in these systems, reducing idle listening is an important design goal.

Low-power MAC protocols deal with this problem by aggressively duty cycling – turning on and off – the radio to reduce power [111]. Synchronous MAC protocols coordinate their sleep and wake times, and a node only transmits a packet when it knows the receiver will be awake to receive it. Asynchronous MAC protocols, in contrast, do not coordinate their sleep and wake times and instead transmitters either (i) send a long preamble, (ii) repeatedly retransmit a packet, or (iii) wait for a request-to-receive message. Receivers periodically *poll* the channel, either by (i) sampling the received signal strength (RSSI) to detect a preamble using low-power listening (LPL), (ii) checking for a packet transmission, or (iii) transmitting a request-to-receive (RTR) probe and listening for a response, respectively.

The most consequential decision that a low-power MAC makes after polling the channel is whether to stay awake or go back to sleep. Since this decision must be made on the order of one hundred thousand times per day in a typical low-power MAC, being indecisive or incorrect can get very costly very quickly. If the MAC decides traffic is pending when none exists – *a false positive* – then the radio will remain on, wasting energy. If the MAC decides no traffic is pending when some is – *a false negative* – then the transmitter's energy is wasted, communications latency increases, and packet goodput drops.

Clearly, making a good decision about whether to stay awake or go back to sleep is a critical one, but it is not an easy one for many reasons. First, *interference* from external systems (*e.g.* an 802.11 network, a cordless phone, or a microwave) might be mistaken for legitimate radio activity. Second, a receiver might *overhear* a partial packet sent to a different node, and stay awake until it can conclude that the packet is destined elsewhere. Third, hidden terminals might cause packet *collisions* or the MAC might mistake external interference for a packet collision, forcing the radio receiver to stay awake longer than required. Furthermore, a multitude of conflicting requirements like support for ultra-low duty cycles, asynchronous network wakeup, asynchronous neighbor discovery, efficient bursty traffic delivery, and handling hidden terminals make balancing energy, latency, false alarms, and false negatives difficult.

This paper claims that a new synchronization primitive is needed to reliably and efficiently balance these conflicting needs. We present backcast, a new link layer primitive that allows a node to probe all of its neighbors in parallel and reliably distinguish the case of zero replies (indicating no pending traffic) from the case of one or more replies (indicating pending traffic). In the former case, the MAC can turn off the radio and quickly return to a sleep state. In the latter case, the MAC would leave the radio on to receive traffic. We implement backcast using the hardware automatic acknowledgments (auto-acks) available in all IEEE 802.15.4-compliant radios as follows. An initiator transmits a packet to a unicast, multicast, or broadcast address. All nodes that match the destination respond with identical acknowledgment frames automatically generated by the hardware. Although these auto-acks interfere, they usually do so non-destructively, so the initiator can decode their superposition. We call this particular exchange of a single probe packet and the potential superposition of multiple ACK frames a backcast.

6.2 Related Work

Given the large impact of Medium Access Control (MAC) protocols on a node's energy consumption, it is not surprising that a wide range of MAC protocols have been proposed for low-power wireless networks. Based on which side of a connection initiates a communication, we classify the MAC as either *transmitter-initiated* or *receiver-initiated*.

MAC protocols can also be classified as scheduled or asynchronous. Protocols in the first category synchronize the duty cycles of nodes with the same broadcast domain [148, 141]. This way, a node can schedule its wakeups to correspond with its neighbors' wakeup schedules. Transmissions among all active nodes are arbitrated through a contention protocol such as CSMA. While scheduled protocols can achieve low duty cycles by controlling a node's sleep interval, they have the inherent overhead of maintaining the nodes' clocks synchronized. Moreover, in the case of S-MAC, nodes keep their radio on throughout the active period irrespective of actual traffic [148]. T-MAC reduces this idle listening period by adaptively adjusting the length of the radio's on period based on the level of traffic [141]. Backcast-based techniques do not require nodes to synchronize their sleep intervals. Furthermore, due to the non-destructive interference of acknowledgments sent by multiple responders, backcast offers a very efficient mechanism for nodes to determine the existence of any pending traffic.

Asynchronous transmitter-initiated protocols on the other hand do not require nodes to synchronize their sleep intervals. Instead, transmitters use the *low power listening* (LPL) technique, originally proposed by Hill and Culler [71] and adapted by Polastre et al. [111], to wake up the intended receivers. According to LPL, receivers periodically turn their radios on to sample the medium. If the level of channel energy is above a predefined threshold the node keeps its radio on to receive a packet. Then, a transmitter that generates a preamble at least as long as the receiver's sleep interval will wake up its intended receiver. While originally proposed in the context of bit-level, low-power radios [71], LPL has been adapted to byte-level [111] and packet-level [19] radios such as the ones defined by the IEEE 802.15.4 standard.

LPL-based protocols are very efficient under low traffic loads as nodes need to turn their radios on only to sample the channel—an action that requires only few milliseconds. On the other hand, because nodes use channel energy levels to detect activity, LPL suffers from the *overhearing* problem. Specifically, nodes that happen to sample the channel during a packet transmission will erroneously turn their radios on. Perhaps more alarmingly, interference from other RF devices (e.g., WiFi networks, cordless radios, microwave ovens, etc.) can also induce nodes to keep their radios on. The results from Section 6.4.3 indicate that LPL indeed suffers from interference, significantly increasing a node's duty cycle. On the other hand, because backcast requires a node to correctly decode an acknowledgement to keep its radio on, it does not suffer from interference. On the other hand, sending a probe is inherently slower than sampling the channel and therefore LPL checks will always be more efficient. Finally, channel usage of backcast-based MAC protocols increases with node density

Many low-power MAC protocols rely on receiver-initiated communications [62] to minimize the medium contention from transmitting preambles, especially in low duty-cycle networks where the sleep interval is large. In RI-MAC [131], nodes broadcast a probe packet every time they wake up. Nodes with pending data packets wait for the probe packet from the destination node. Musăloiu-E. et al. presents *low power probing* (LPP) [106], a MAC protocol that implicitly uses the backcast primitive to provide fast feedback on whether pending data packets exist. Specifically, LPP nodes take advantage of the acknowledgments generated by the lower layers, such as the hardware acknowledgement from the 802.15.4-compliant radios. In this regard, Pollcast [35] also offers a way to quickly calculate predicates. However, as the paper acknowledges: simultaneous pollcasts within a two-hop neighborhood would cause false positive. Backcast can provide a robust primitive for pollcast. The poll initiator first transmits the predicate packet, which contains an ephemeral identifier. Upon receiving, nodes change their hardware address to match the identifier in the case of positive predicate evaluation. Then, the poll initiator transmits the poll packet to the identifier address and awaits an acknowledgement.

Network-wide wakeup is an often used primitive in low-power wireless networks, used to synchronize the network or to flood data to the whole network. Dutta et al. proposed one approach to this problem [44]. In their scheme, every node periodically transmits a beacon and then briefly listens for channel activity (either a packet or increased energy). If any channel activity is detected, the node remains awake, but if no activity is detected, the node goes back to sleep. To wake up the network, the initiator listens for a time equal to the beacon period to identify all one-hop nodes. Then, during the next such period, the initiator contacts each of its one-hop neighbors in turn. These neighbors then repeat this process for the two-hop neighbors, and so on. If two or more nodes attempt to contact the same node in a lower tier, the paper conjectured that the concurrent transmissions may collide, but that the receiver would detect channel energy, remain awake, and give the transmitters a chance to enter backoff and compete. Musăloiu-E. et al. proposed *low power probing* (LPP) as another solution to the wakeup problem [106].

6.3 Link Layer Communication Services

The section presents the design of several receiver-initiated link layer communications services built using backcast. In particular, we show how low-power, high-performance link services including unicast, broadcast, wakeup, and discovery can be constructed using the probe mechanism. We also discuss the benefits and drawbacks of using backcast-based implementations compared with more traditional approaches that use channel sampling. To ground our design, we focus on the IEEE 802.15.4 standard [78] and two commercially-available radios that support this standard, the TI CC2420 [134] and the Atmel AT86RF230 [11], and we map the backcast frame exchange to 802.15.4 data and ACK frames.

6.3.1 Unicast Communications

In a receiver-initiated implementation of unicast communications, a transmitter first listens for a ready-to-receive (RTR) packet and only then transmits a data packet in response to the receiver's RTR. The transmitter often jitters the data packet transmission with a small, random delay to avoid collisions if multiple transmitters are waiting. Protocol processing overhead can introduce additional delays in generating the data packet: the transmitter must receive the RTR packet, copy it from the radio to the processor memory, signal an interrupt, dispatch the packet to the link layer, determine if the packet is indeed an RTR from the intended receiver, if so then possibly jitter the transmission, and finally copy the data packet into the radio's transmit buffer. Meanwhile, the receiver must wait patiently with its radio turned on, using precious energy.

Our basic unicast design diverges from traditional receiver-initiated designs by first acknowledging the probe with a fast and deterministic radio-generated frame (a backcast frame exchange), and only then sending the data packet. This approach has many benefits. First, the receiver only has to wait marginally longer than the radio's RX/TX turnaround time before concluding that no inbound traffic is present, saving considerable energy on every probe. In the IEEE 802.15.4 standard, a turnaround occurs in 192 μs, nearly 20 times faster than the 3.75 ms beacon-data turnaround time that RI-MAC requires with its software-based protocol processing [131]. Second, our approach distinguishes between collisions and interference, whereas RI-MAC cannot. In RI-MAC, as with LPL channel samples, interference can lead to extended listening. With a backcast-based approach, interference is easily distinguished from an ACK superposition: the former appears as just channel energy; the latter appears as a real frame reception. Therefore, HotMac does not suffer from a range of interference-based false alarms.

The rest of this section describes the design of our backcast-based unicast service. To implement unicast, we use two key features of IEEE 802.15.4-compliant radios: hardware-based address filtering and hardware-generated auto-acks. The radio hardware can be programmed to automatically generate an auto-ack if an incoming frame's destination address matches the local hardware address (the 16-bit or 64-bit MAC address). The design question, then, is what source and destination addresses to use in the probe frame? One option is for the probe to be sent to the broadcast address requesting an auto-ack, and all nodes with pending traffic enable auto-acks for broadcast frames.

There are three problems with this approach. First, a receiver will auto-ack *every* probe it receives, including ones for which no traffic is pending, causing all but one neighbor to stay awake and waste energy. We call this the *overreacting* problem. Second, the 802.15.4-2006 standard specifically prohibits this behavior: § 7.5.6.4, "... any frame that is broadcast shall be sent with its Acknowledgment Request subfield set to zero." Third, because this behavior is prohibited, it enjoys poor radio support: while CC2420 [134] radio and AT86RF230 [11] Rev A silicon both support broadcast auto-acks, the AT86RF230 Rev B silicon "fixes" this standards non-compliance and does not auto-ack broadcast frames.

We avoid the overreacting problem and design a standards-compliant unicast protocol as follows. When a sender S has pending traffic for a receiver R, S enables hardware address recognition, enables its hardware auto-acks, and *sets its hardware address* to R+0x8000.[1] Instead of sending a probe to the broadcast address, receiver R sends its probe to destination address R+0x8000 and requests an auto-ack. Sender S (as well as any other nodes with pending traffic to R) respond to the probe (multiple auto-acks interfere non-destructively). If its probe is acknowledged, R remains awake to receive a packet while sender S does not succumb to the overreacting problem. Figure 6.1 illustrates a unicast exchange (left) and unicast collision recovery (right). The remainder of this section describes the details of this exchange.

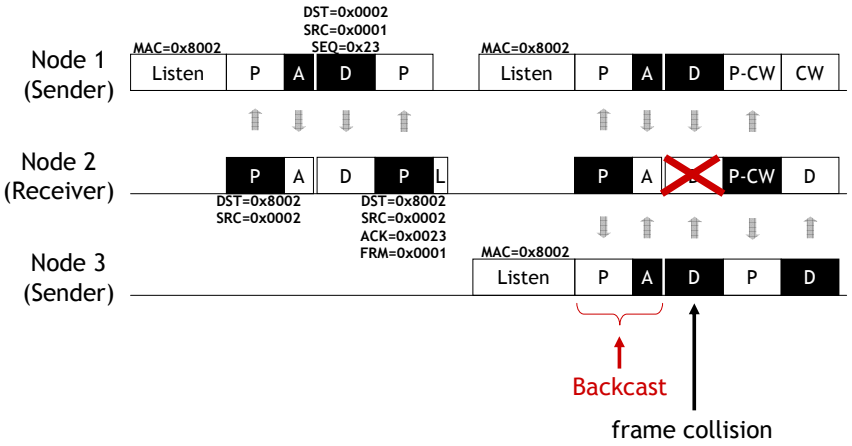


Figure 6.1: Unicast communications using backcast. A contention-free transfer (left) and a collision (right).

Figure 6.1 (left), shows a sender (Node 1) with traffic pending for the receiver (Node 2). The sender turns on its radio, sets its hardware address to 0x8002, enables hardware auto-acks, and begins to listen. At some later time, the receiver wakes up and sends a backcast probe with a source address of 0x0002 and a destination address of 0x8002, and requests an acknowledgment. When the sender receives the probe frame, its radio automatically generates an ACK frame. Upon detecting the beginning of the ACK frame, the receiver decides that an ACK frame may be incoming, so it continues to listen for at least 352 μs (or possibly less if the data appear garbled) before turning off the radio. If a valid ACK is received, the receiver concludes there is pending traffic for it, and it remains awake to receive this data. At the same time, the sender transmits a data packet (after a short random delay) with a source address of 0x0001, a destination address of 0x0002, and a locally-selected sequence number of 0x23, which the receiver successfully receives. The sender does *not* change its radio hardware address for this transmission. The receiver then prepares its next probe which explicitly acknowledges the preceding data packet by source address (0x0001) and sequence number (0x23).[2] The sender turns off auto-acks if it has no further data pending (or repeats this process if it has more data), letting the receiver's second probe go unacknowledged, which allows the receiver to return to sleep after a short wait (of just over 192 μs).

[1] We reserve addresses with the high-order bit set for this purpose.

[2] As an optimization, the receiver ACKs the sender's data packet, which the sender uses as a "hint" that its transmission was successful (since hardware auto-acks only have sequence numbers but not source or destination addresses). This optimization allows the sender to disable auto-acks prior to the receiver's next probe transmission, eliminating a race condition in which sender has to check the contents of the receiver's second probe to decide whether to ACK it. The sender still waits for the receiver's second probe to verify the hint by checking that the second probe's sequence number *and* source address match sender's previous packet.

Figure 6.1 (right), shows two senders (Node 1 and Node 3) with traffic destined for a receiver (Node 2). Both senders auto-ack the receiver's probe. Although the ACKs collide, they do so non-destructively, so their superposition is correctly decoded as an ACK frame, as we showed it Chapter 5. In this example, the senders are hidden terminals so they cannot detect each other's transmission, so introducing the small, random jitter between the ACK and the data does not lead to collision avoidance (although it likely would if both transmitters were in the same collision domain). Unaware of one another, both senders begin transmitting their data packets, but these packets collide at the receiver, and are lost. After having received an ACK, and having waited the maximum packet transmission time, the receiver concludes that the transmission was lost and sends a second probe which increases the size of the contention window. In our example, Node 2 selects the first contention window and successfully transfers its packet while Node 1 selects the second window and does the same.

6.3.2 Broadcast Communications

Broadcast is a fundamental operation used by a wide range of upper layer services and applications. Neighbor discovery, routing updates, and data dissemination, all depend on a robust broadcast service for operation. Backcast-based broadcast is identical to unicast communications with one important difference. A sender S, instead of using the radio hardware address recognition to filter probes based on a particular receiver R's probe address, $R+$`0x8000`, simply disables hardware address recognition altogether but keeps hardware auto-acks enabled.

When a higher-layer service wants to send a broadcast packet, it sets the destination address of the packet to the broadcast address, e.g. `0xFFFF`, and submits the packet to HotMac for delivery. When this packet is ready for transmission, the HotMac disables the hardware address recognition function of the radio for at least as long as the probe period of it neighbors (or the longest of its neighbors' probe periods, if different neighbors are operating with different periods). During this time, S will auto-ack *every* probe it receives, regardless of the probe's actual destination address, and proceed to send the data packet like in the unicast case. Although this design does not violate the IEEE 802.15.4 standard, it clearly abuses the standard in support of physical and link layer primitives that the standard was not originally designed to support. Our goal is to show that our design can work with existing hardware, not that it is necessarily standards-compliant (although the latter is preferable, to allow it to be implemented with standards-compliant hardware).

A common case that arises with this design is what to do if, while the broadcaster is listening for neighbors' probes, the broadcaster's own probe timer fires. Should it send the probe and then return to listening or should it forgo the probe entirely and simply continue listening? The HotMac design chooses the first approach: a probe is transmitted when the probe timer fires. Doing so avoids a scenario we call the *broadcast standoff* in which two or more nodes that attempt to broadcast a packet wait patiently for the other(s) to transmit a probe. This situation is avoided in the HotMac design, but it raise two issues. First, the transmit and receive state machines become more complex and cross-coupled. Second, while probing, a broadcaster may miss other neighbors' probes, thereby reducing broadcast reliability.

One potential issue with our design is that if hardware auto-acks were used to acknowledge data packets as well as probes, then a broadcaster would inadvertently acknowledge every single data packet it received, signaling that the data packet was successfully received by its sender when in fact it may not have actually been received. Our unicast implementation avoids this problem by reserving hardware auto-acks exclusively for acknowledging probes. Data packets are acknowledged by including the acknowledgment information in the next probe. We do explore, as an optimization, auto-acking data packets but in this case, the ACKs are only hints that a transmission succeeded. The next probe contains the conclusive acknowledgment.

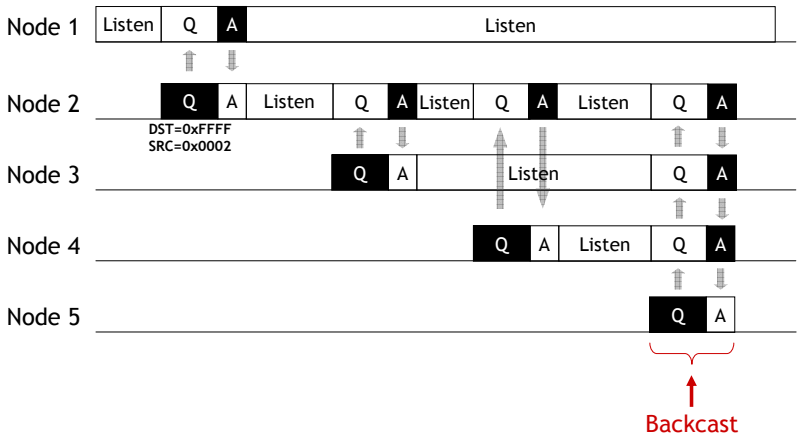


Figure 6.2: Asynchronous network wakeup using backcast. Although Nodes 2, 3, and 4 all ACK Node 5's query probe, the ACK collision is non-destructive, and Node 5 remains awake to communicate.

6.3.3 Asynchronous Network Wakeup

Waking up a multihop network of duty cycled nodes is a fundamental problem in sensor networks. Applications as diverse as interactive data collection, exceptional event detection, and target tracking require nodes to wake up neighbors or even the entire network in response to an asynchronous event. In many such applications, nodes will remain asleep for long periods of time and so they are likely lose synchronization. Ideally, the nodes would wake up only in response to external events or user queries, but would otherwise remain asleep. In the mobile case, although nodes may only need to communicate when then have data to upload, it is still useful to be able to wake up a mobile node to issue it a command or query.

Several techniques have been proposed for asynchronous network wakeup in a low-power setting including various forms of flooding and dissemination, but these techniques have poor channel efficiency, exhibit logistic-like performance in that they start and end slowly, or are designed with the assumption that nodes are synchronized. As a result, none of these techniques are ideally suited to the low-power, asynchronous network wakeup problem. In this section, we discuss two approaches to designing a backcast-based wakeup service – one that can work with standards-compliant radios and one that cannot. They exhibit high channel efficiency, achieve the lower bound on wakeup time, and do not assume unsynchronization.

Figure 6.2 shows the first approach. In this figure, all nodes cease periodic communications like routing beacons and instead operate at a very low duty cycle. The nodes wake up infrequently, perhaps once every ten seconds or each minute, to check if any of their neighbors requires them to stay awake, by sending a probe to the broadcast address. Node 1 initiates an asynchronous network wakeup by configuring its radio to acknowledge all frames. After some time, Node 2 sends a query probe. Node 1 ACKs this query and Node 2 stays awake. This process repeats with Node 2 waking up Node 3 and Node 4. However, when Node 5 wakes up, its neighbors – Nodes 2, 3, and 4 – are already awake and they all simultaneously ACK Node 5's query. Although the ACKs collide, they do so non-destructively, ensuring Node 5 stays awake.

Transmitting to the broadcast address with the acknowledgement request bit set does not comply with the 802.15.4 standard (and hence only works with the CC2420). One way to sidestep the issue is to send the probe query to a reserved wakeup address rather than the broadcast address. This leads to a wakeup phase, in which a node first performs wakeup for one cycle, and then engages in normal communications.

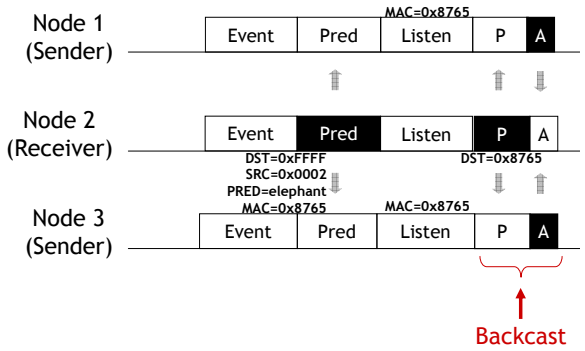


Figure 6.3: Pollcast based on the backcast primitive. All nodes observe an "elephant sighting" event. Node 2 wishes to corroborate this observation with its neighbors. It uses backcast to efficiently determine if any neighbor also observed this event.

6.3.4 Pollcast Neighborhood Queries

Demirbas et al. recently proposed *pollcast*, a two-phase primitive in which a node broadcasts a poll about the existence of a node-level predicate P and then all nodes for which P holds reply simultaneously [35]. The poller detects one or more positive replies by sampling its radio's Clear Channel Assessment (CCA) signal. While pollcast offers a novel approach to quickly calculate predicates, the proposed mechanism has some drawbacks, as the paper acknowledges: simultaneous pollcasts within a two-hop neighborhood would cause false positives (as would interference from external sources). Backcast provides a more robust primitive for implementing pollcast, as Figure 6.3 shows.

To leverage the backcast primitive, pollcast is modified to first transmit the predicate, then transmit the poll, and finally listen for an acknowledgment. Figure 6.3 shows all nodes observing an event. Node 2 wishes to corroborate an "elephant sighting" event with its neighbors so it transmits a predicate describing the event, including a locally-generate ephemeral identifier. The destination address of the predicate is `0xFFFF` (broadcast), the source address is `0x0002`, the predicate is `'elephant'`, and the ephemeral identifier is `0x8765`. Node 2 then waits for some time to allow Nodes 1 and 3 to receive and evaluate the predicate. Node 1 then sends a probe destined to the ephemeral identifier `0x8765`. Since both Node 1 and Node 3 observed the same event, they both ACK the probe, indicating the predicate was true. Note that although the predicate is sent to the broadcast address, it does not need to be automatically acknowledged, so this approach is compatible with 802.15.4. standards-compliant radios.

One drawback with this approach to implementing pollcast is the need for *two* packet transmission by the receiver: the first packet sends the predicate and the second packet sends the probe to the ephemeral identifier. Ideally, the predicate could be piggybacked onto to the probe, eliminating the separate predicate transmission and its associated delay. The two challenges with this approach include choosing the destination address of the probe and ensuring that the predicate can be evaluated quickly enough (by the processor) to generate a properly timed auto-ack. One option that we explore is to send the probe to the broadcast address, piggyback the predicate on the probe, and pad the probe with a large payload. This approach allows a node to detect the beginning of the probe, read and evaluate just the predicate while the rest of the packet is being received, and still enable hardware auto-acks before packet reception completes. The pad bytes provide sufficient buffer time to evaluate the predicate before the 192 μs auto-ack timer fires.

6.4 Evaluation

This section evaluates the viability, efficiency, and performance of a backcast-based link layer architecture. We implement all of the primitives and services described in this paper and evaluate our implementation against a range of performance metrics. We evaluate both low-level microbenchmarks and high-level macrobenchmarks to characterize critical details and overall performance under realistic workloads. The goal of these experiments is to explore the degree to which a backcast-based approach to low-power link layer services is viable, identify areas in which its performance is not competitive with competing approaches, and quantify areas in which it outperforms other techniques.

6.4.1 Methodology

We evaluate HotMac using the TinyOS 2.x sensor network operating system [72]. We use the Moteiv Tmote [103], Crossbow Iris [29], and Berkeley Epic [48] motes for these experiments. The Tmote and Epic platforms are based on the TI CC2420 radio [134] while the Iris uses the Atmel AT86RF230 radio [11]. Both radios are IEEE 802.15.4 standards-compliant and can interoperate at a 250 kbps data rate. We verified that HotMac implementation on different radios are compatible, but do not provide additional details about their interoperability.

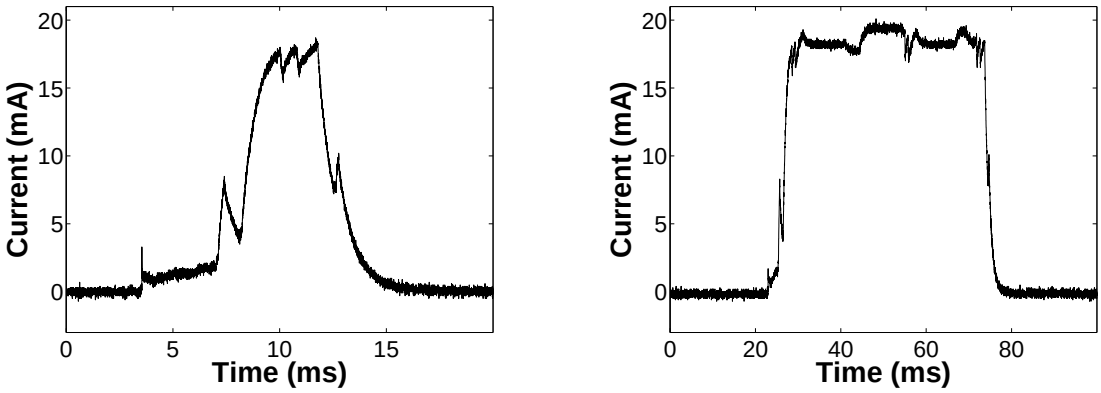
6.4.2 Link Energy Microbenchmarks

Backcast-based services are built by combining a small set of link primitives including *probe*, *receive*, *transmit*, and *idle* (listening for a probe). The probe involves sending a probe frame, performing an TX/RX switch, and listening for an ACK frame. The receive primitive encapsulates the probe and additionally involves continuing to listen after an ACK frame is received, receiving a data packet, loading a probe that acknowledges the data packet, and performing another probe operation. The transmit primitive is more complex and involves configuring the radio address and hardware auto-acks, listening for (and automatically acknowledging) a probe, loading and transmitting a data packet, and listening for a probe that acknowledges the prior data packet. An idle operation largely involves listening, like the transmit operation, but without the probe, data, and ACK packet exchanges.

Table 6.1 summarizes the energy cost of each basic primitive, including a clear channel assessment, the fundamental primitive for low-power listening protocols. In all cases involving packet transmission or reception, we use the IEEE 802.15.4 link MTU frame size (127 byte payload). Figures 6.4 show a trace of the probe and receive primitive current draws and Figures 6.5 show a trace of the transmit and idle (listening for a probe) primitive current draws, both for the Epic mote. These data are collected by capturing the voltage drop across a 10 Ω resistor in series with a 3 V power supply using a Tektronix TDS3014 digital storage oscilloscope.

Primitive	Cost (μJ)
Probe	253
TX only	1578
RX only	2670
CCA	194

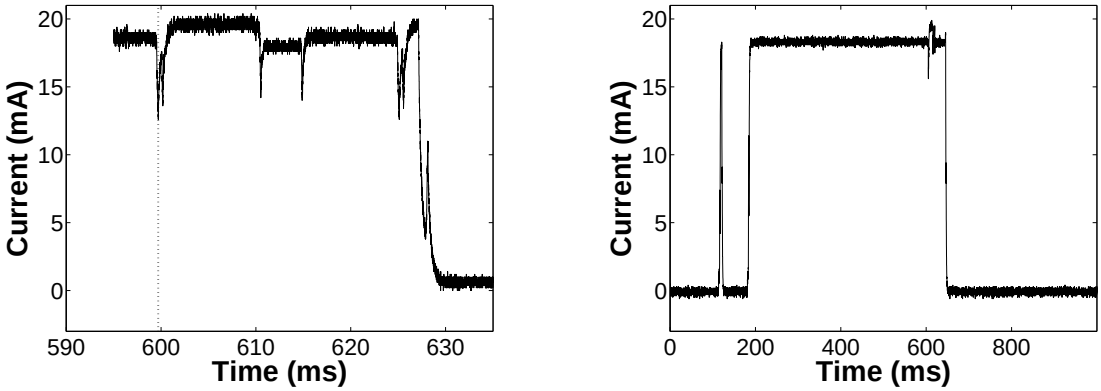
Table 6.1: Primitive energy costs.



(a) Probe

(b) Receive (Length=127 bytes)

Figure 6.4: Link current trace. The current draw for (a) probe and (b) receive primitives. These figures show the Epic mote's instantaneous current draw for the representative asynchronous link primitives.

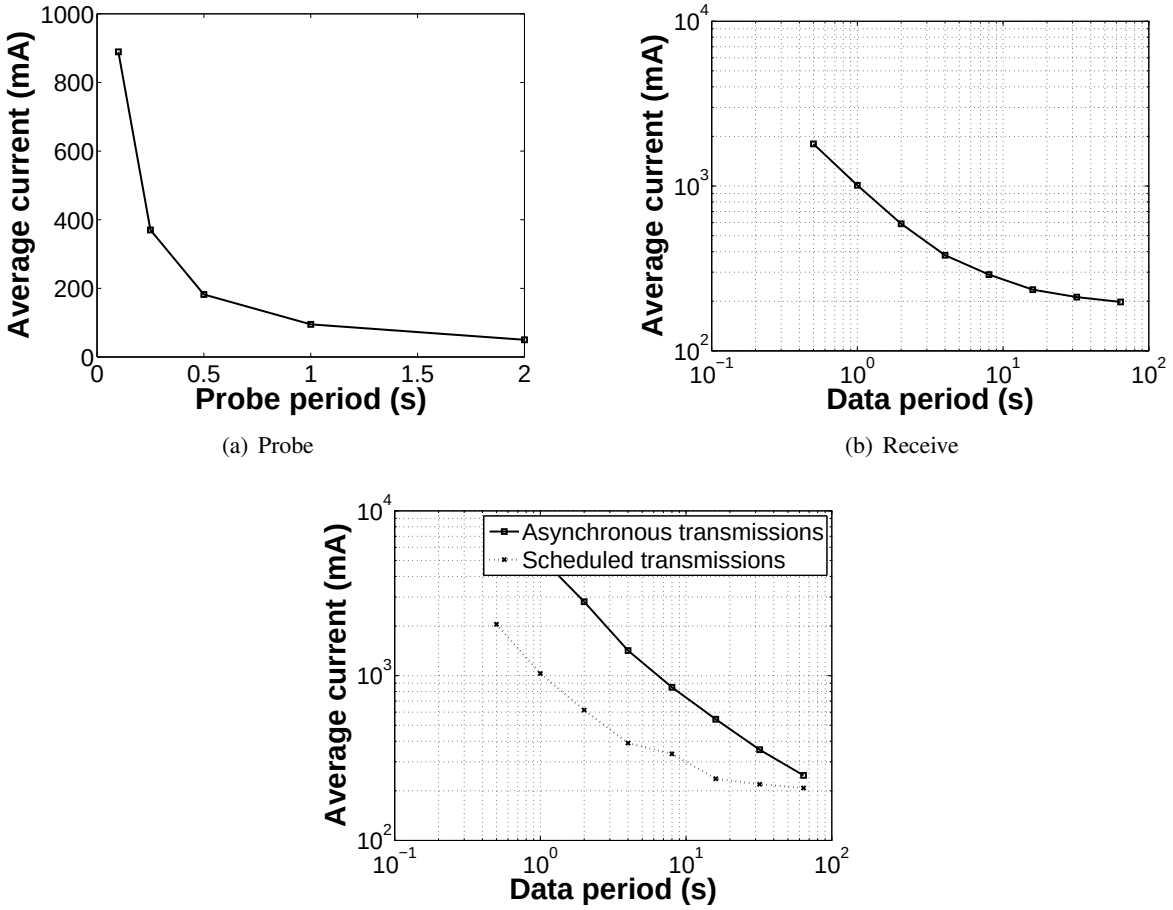


(a) Transmit (Length=127 bytes)

(b) Idle (500 ms wait)

Figure 6.5: Link current trace. The current draw for (a) transmit and (b) idle (listening for a probe). These figures show the Epic mote's instantaneous current draw for the representative asynchronous link primitives.

Figures 6.6 show the average current draw of the probe, receive, and transmit for various probe and data transmission periods. Figure 6.6(a) shows how the average current due to probing cost scales with the probe period. Figure 6.6(b) shows the how the receive cost scales with data rate. Figure 6.6(c) shows the how the transmit cost scales with data rate for both *asynchronous* communications (when the transmitter does not know the receiver's probe schedule) and *synchronous* communications (when the transmitter does know the receiver's sleep schedule). Collectively, the data in These figures show that the link *primitives* in backcast-based services are *more* expensive than in an optimized, commercial-grade LPL implementation using a transmitter-initiated approach [77] but comparable or better than most other research systems that offer similar functionality.



(a) Probe

(b) Receive

(c) Transmit

Figure 6.6: Link power model. The average current for probing, receiving, and transmitting, respectively, as a function of the probe period (f) and data period (g) and (h), with $T_{probe} = 0.5$ s.

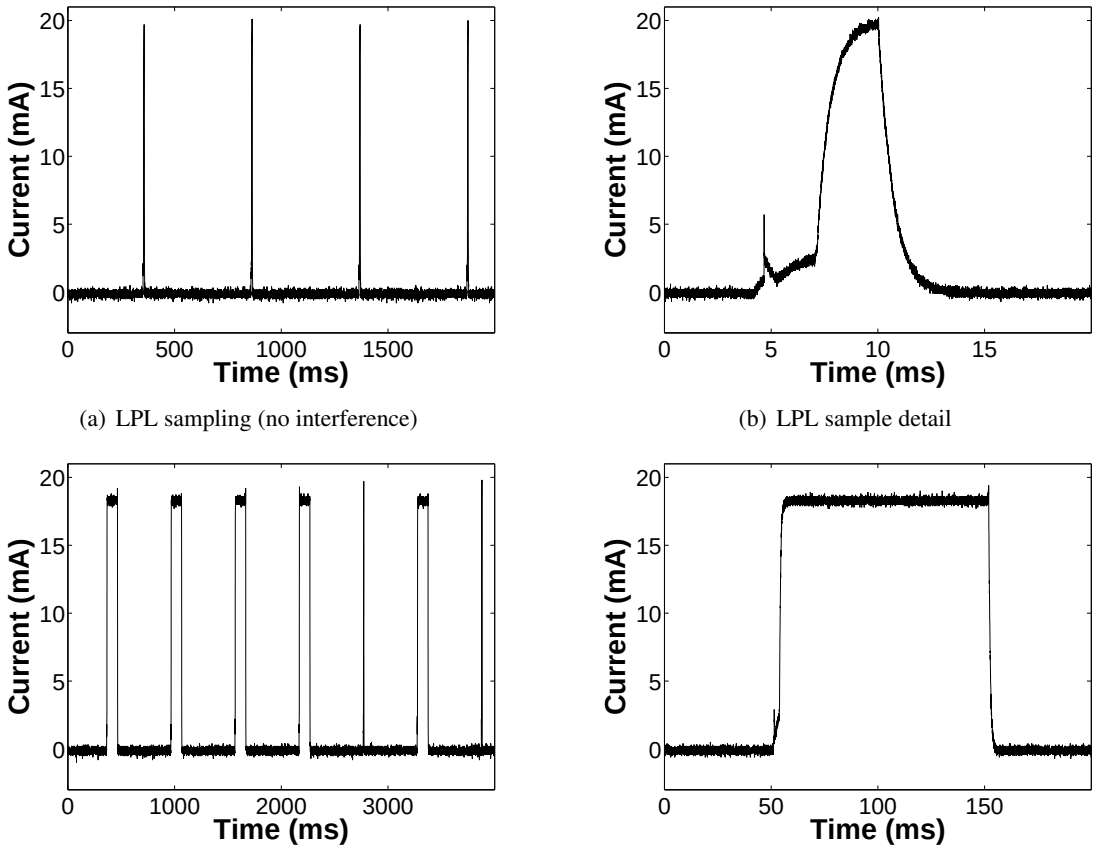
6.4.3 Susceptibility to Interference

A basic problem with LPL-based systems is that they suffer from many sources of false alarms. First, *interference* from an external system (e.g. an 802.11 network, cordless phone, or microwave) might be mistaken for legitimate radio activity. Second, a node might *overhear* a partial packet sent to a different node, and stay awake until it can conclude that the packet is destined elsewhere. Third, hidden terminals might cause packet *collisions* or the MAC might mistake external interference for a packet collision, forcing the radio receiver to stay awake longer than required. Recent research has demonstrated the cost of external interference on the effective duty cycle of LPL protocols – for example increasing from a programmed 2.2% to actual 5.6% when an LPL-based link operates near an 802.11 access point [61].

We repeat similar experiments and find the results more surprising than anticipated. Table 6.2 shows the results of an experiment in which we measure the receiver's idle listening current in an office environment on two channels and with and without a file transfer in progress. Although the sampling technique performs better under ideal conditions, it degrades dramatically in the presence of interference, increasing the average

Channel	802.11 Interference?	Backcast	Sampling
26	no	203 μA	175 μA
26	yes	221 μA	2785 μA
17	no	206 μA	2461 μA
17	yes	230 μA	3030 μA

Table 6.2: The average current draw of our backcast-based scheme and the LPL preamble-sampling scheme used in TinyOS 2.1 under no-load conditions and a 500 ms probe or check period. Although the sampling technique performs better under ideal conditions, it quickly degrades under interference. Results are the average of five samples, each one minute long.



(a) LPL sampling (no interference)

(b) LPL sample detail

(c) LPL sampling (w/ interference)

(d) LPL overhearing detail

Figure 6.7: LPL preamble sampling techniques leave receivers susceptible to noisy wireless environments, such as those caused by 802.11 interference. Figures (a) and (b) show the macroscopic and microscopic behavior of the TinyOS 2.1 sampling algorithm when the channel is clear: the receiver immediately returns to sleep. Figures (c) and (d) show the macroscopic and microscopic behavior while a file transfer is in progress using a nearby 802.11 access point. Of the seven channel samples visible in this trace, five are unnecessarily lengthened due to channel noise.

power by more than an order of magnitude compared to the idle listening case. In contrast, backcast exhibits less than 15% increase in its average power draw.

Figure 6.7 illustrates how the preamble sampling techniques used in LPL protocols leave receivers susceptible to noisy wireless environments, such as those caused by 802.11 interference during beaconing, file transfers, or audio/video streaming. Figures 6.7(a) shows the current draw over time when the channel is clear and Figure 6.7(b) shows the detailed current draw of one channel sample. Figure 6.7(c) shows the current draw of the same system while a file transfer is in progress using a nearby 802.11 access point. Of the seven channel samples visible in this trace, five are unnecessarily lengthened due to channel noise. Figure 6.7(d) shows the details of an extended sample.

Although the overhearing problem is well-known in the design of low-power link protocols, these results suggest that the response to interference and overhearing deserve further study. It is not clear how duty cycles, delivery ratios, false positives, false negatives, and latency are affected as the channel sample time is adjusted after a "busy" channel assessment. These result also underscore the challenge of predicting the lifetime of a sensor deployment based only on models of data workloads, but without a good model of the environmental factors.

6.4.4 Unicast Performance

Our evaluation thus far has focused on several important microbenchmarks comparing the time, energy, and false alarms of backcast and LPL-based primitives. We now turn our attention to how well backcast supports several common services. It is well known that receiver-initiated MAC schemes handle contending flows and hidden terminals much better than low-power, transmitter-initiated ones [62, 25, 131]. Our results confirm that backcast-based unicast communications handles contending flows well. Table 6.3 shows four senders contending to transmit to a single receiver. In this experiment, the receiver sends a probe, the senders all ACK the probe concurrently, and then they contend for the channel. The receiver resends a probe after either each successful transmission or after receiving an ACK, but no data. The receiver sends up to a total of five probes before stopping. Each probe doubles the size of the contention window. The base contention window size is 20 jiffies (610 μs).

Number of Senders	Packet Delivery Ratio
1	99.1%
2	98.8%
3	97.5%
4	90.9%

Table 6.3: Packet delivery ratios for 1 through 4 distinct senders sending to a single receiver. $T_{probe} = 0.5$, and senders attempt to send on each probe to stress the contention algorithms, for 1000 packets. The largest difference between the maximum and minimum success rates was found with three senders and was 2.6% $(96.7 - 99.3)$

6.4.5 Asynchronous Network Wakeup Performance

A wakeup is special case of network flood or dissemination in which the goal is to ensure that every node in the network receives a wakeup message. Prior work has shown that LPL-based flooding techniques

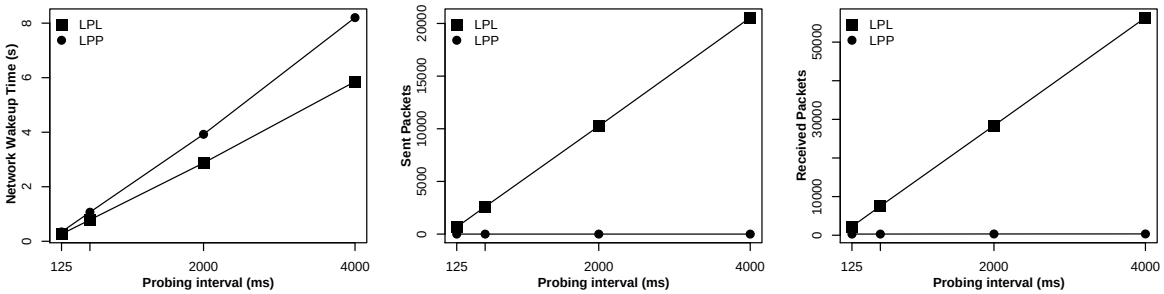


Figure 6.8: Wakeup latency for LPL and backcast for several sleep periods (0.125s, 0.5s, 2s, 4s). LPL is about 30% faster at waking up the network than Backcast. Backcast (labeled LPP) can provide slightly higher mean wakeup times with a fraction of the packets (and channel usage) required for LPL.

can cause significant contention and can use the radio channel (a scarce resource) for a long time to complete a flood [95]. Figure 6.8 explores how well the TinyOS 2.1 LPL and a backcast-based wakeup implementation compare. In this experiment, the LPL wakeup algorithm is a simple flood: the source of the flood repeatedly resends a wakeup packet for slightly longer than the sleep interval. Every node that receives the packet also retransmits it, after it detects a clear channel. The backcast flooding algorithm is a recursive broadcast without subsequent data packet transmissions.

Figure 6.8 shows the mean wakeup time of 47 nodes in a multihop testbed environment. The results show that LPL is about 30% faster at waking up the network than backcast. However, this performance comes at the expense of a far larger number of packet transmissions for LPL while backcast-based wakeup requires exactly one probe per node per sleep interval and at most one acknowledgment per neighbor per sleep interval. In other words, backcast exhibits dramatically better channel efficiency, allowing other concurrent communications and greater regulatory compliance.

6.4.6 Pollcast Performance

We next explore a question that was raised in our earlier discussion of pollcast: how large of a payload padding is needed to allow a node to read and evaluate a predicate contained within a packet while the packet is in transmission, but before it has been received in its entirety. The benefits of such processing is that it would allow arbitrary predicates to be included in a packet and evaluated by its recipient – that is, it allows software-in-the-loop backcast – while still retaining the timing benefits of hardware auto-acks.

Figure 6.9 shows the effect of varying the payload size on the mean inter-packet delay for auto-acks. The frame size includes the 802.15.4 header starting with the frame control field (FCF) and includes the frame check sequence (FCS). The mean shows that the difference between hardware auto-acks (HACKs) and the default software-generated acknowledgments (SACKs) implementation in TinyOS 2.1. Software-optimized "Fast-SACKs" can close this gap. The standard deviation graph provides more details about the variability of SACKs and shows that Fast-SACKs can achieve HACK timing starting from a payload of 36 bytes (36-11=25 if we exclude the 802.15.4 header used in TinyOS). If an application needs to selectively ACK, but also preserve the timing properties of a HACK, the application may wish to use a packet length of at least 36 bytes, adding at most 800 μs (at a 250 kbps data rate) to the probe packet. Figure 6.10 confirms that Fast-ACK collisions are non-destructive and function identically to radio-generated HACKs.

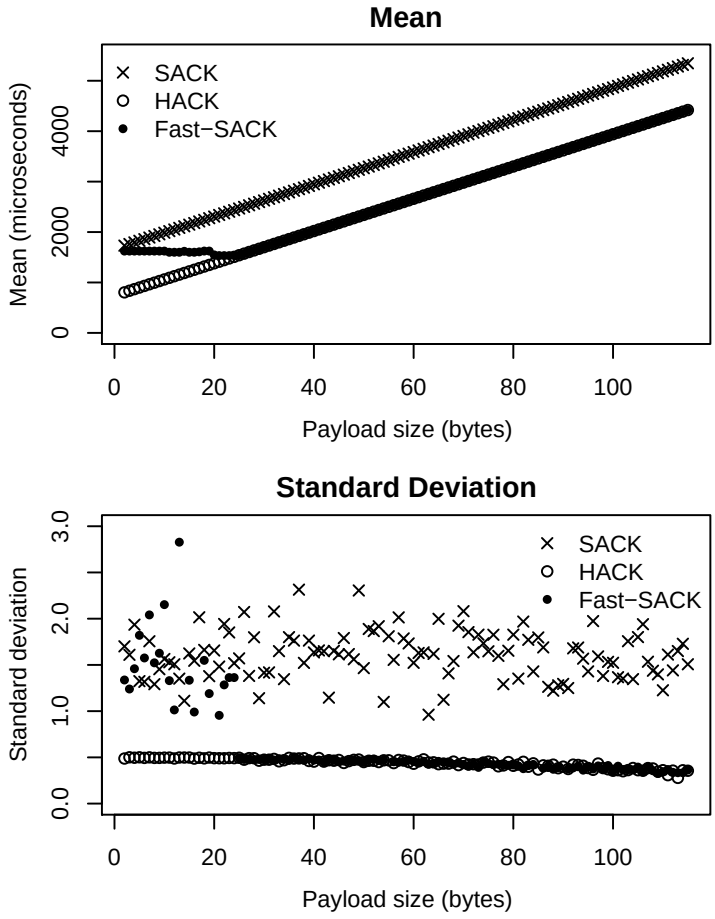


Figure 6.9: The impact of payload size on the mean and standard deviation in the inter-packet delay for hardware auto-acks (HACK), software-generated acknowledgments (SACK), and pollcast-optimized SACKs (Fast-SACK). The timing data are provided by the the CC2420 Developers Kit Packet Sniffer.

6.4.7 Collection Tree Protocol Performance

Collection (or convergecast) is *the* communications workload in data collection-oriented sensornets and any link layer that is to be competitive must perform well on collection. We now evaluate how our backcast-based low-power probing (LPP) implementations of unicast and broadcast support the TinyOS Collection Tree Protocol (CTP) [64] and how well these implementations perform compared to the standard TinyOS low-power listening (LPL)-based link layer.

Figure 6.11 shows the results of this experiment on a 137 node test network. The results show that (i) the distribution of node duty cycles are similar, but that backcast-based LPP exhibits about a 1% higher mean value than LPL; (ii) the average number of packets generated by each LPL node is between 3 and 24 times higher than backcast, and (iii) the inter-arrival time of packets at the base station (a measure of delivery latency) from each node is approximately similar. These results show that backcast can provide similar energy-efficiency and delivery latency as LPL-based systems with much higher channel efficiency. These

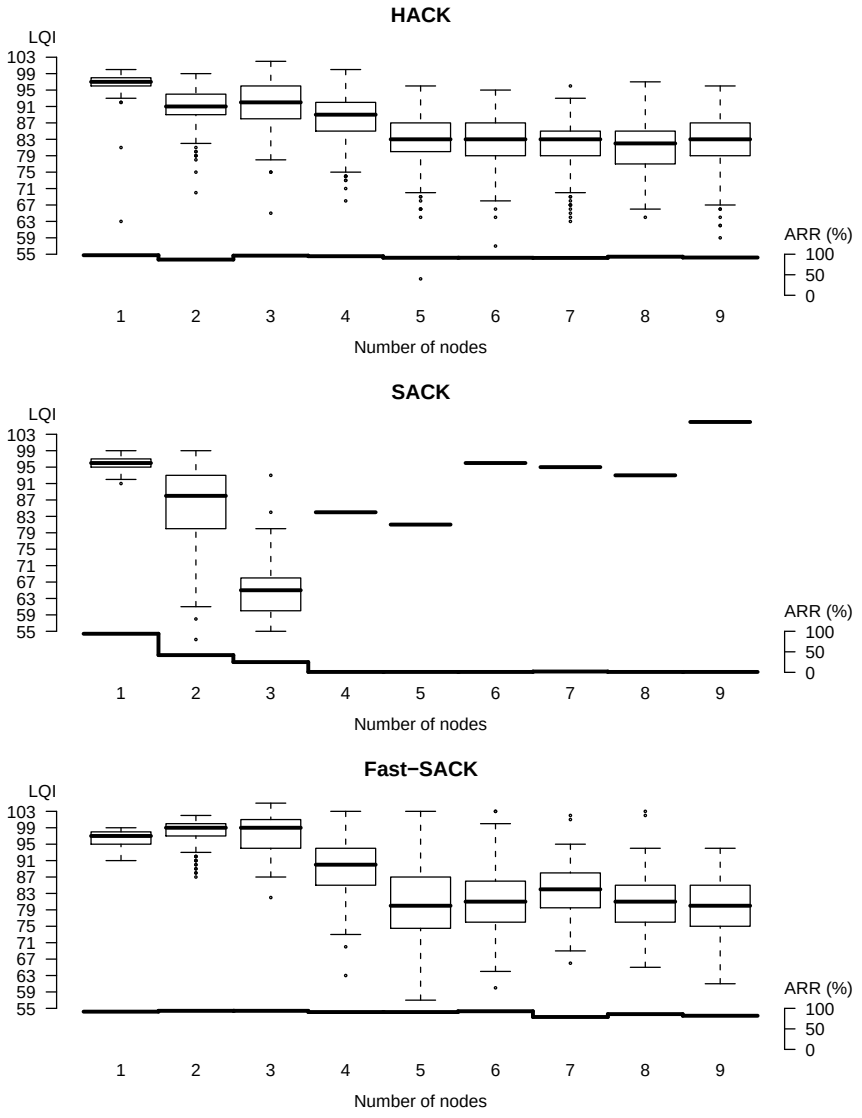


Figure 6.10: The effect on LQI and ARR as the number of concurrent ACKs goes from one to nine for HACK, SACK and Fast-SACK. These figures show the impact on LQI and ARR as a function of the number of concurrent ACKs. We see that SACKs collide destructively very quickly – after just four concurrent nodes in this experiment – but that Fast-SACK continues to work. The timing properties of the Fast-SACKs indeed translates to comparable performance as HACKs.

data support our thesis that a solution that handles the mobile case well can also support static networks as a special case.

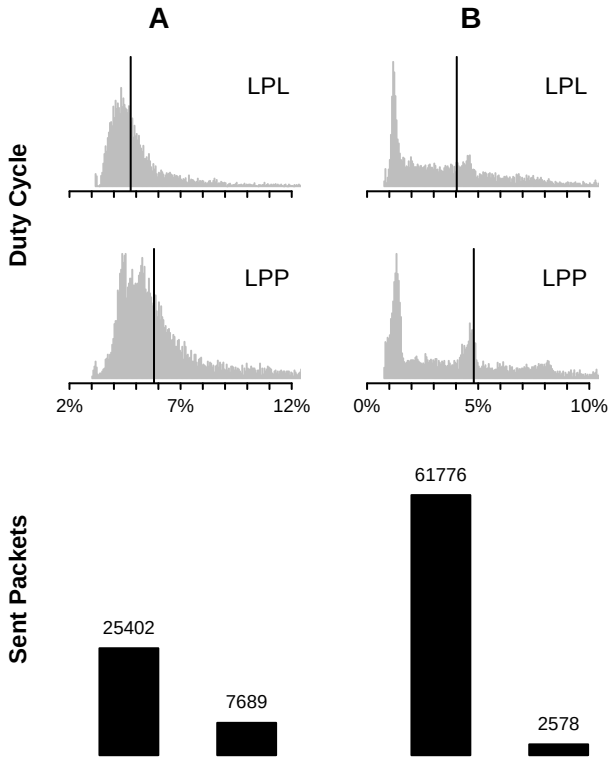


Figure 6.11: CTP performance over backcast-based LPP and the TinyOS 2.1 LPL implementation on a 137-node testbed. Both sub-figures have a sampling interval of 60 seconds and a delay-after-receive of 20 ms. However, subfigure A uses a shorter sleep interval, 512 ms, while subfigure B uses 2048 ms. Although the mean duty cycle of backcast-based LPP is slightly higher, it exhibits much greater channel efficiency, between 3 and 24 times.

6.5 Discussion

We briefly discuss, in this section, how protocol standards and radio hardware could evolve to better support backcast-enabled services.

6.5.1 Standards Implications

To send a packet to receiver R using the unicast protocol described in this paper, a sender S sets its own hardware address to R+`0x8000` and a receiver sends a probe to this same address. This scheme is used to ensure that an auto-ack is only generated for the intended recipient (to avoid the overreacting problem). A better design might have been to have R send the probe to the broadcast address and let a sender filter on the source address of the probe, which is R, and only then auto-ack the probe. Unfortunately, there are three issues with this approach. First, the IEEE 802.15.4-2006 specifically prohibits acknowledging broadcast frames: § 7.5.6.4 reads, "... any frame that is broadcast shall be sent with its Acknowledgment Request subfield set to zero." Second, some radios do not support acknowledging broadcast frames [11]. Third although some newer radios provide hardware support for *source* address filtering [136], others do not and

instead only support *destination* address filtering [134, 11]. Because of these constraints, our unicast design reserves addresses with the high-order bit set for probe packets.

6.5.2 Hardware Implications

There are a few radio enhancements that could improve the performance and energy efficiency of the backcast primitive and low-power services built using this primitive. The main bottlenecks in our current design occur from the limited processor-radio bandwidth. Since backcast-based communications requires multiple loads and unloads of the transmit and receive FIFOs, respectively, they are often the critical path operations. Since these data transfers occur over a relatively slow serial bus, they affect both the time and energy cost of communications. If either hardware support for backcast existed, or the primitive and MAC services were implemented in a processor with a memory-mapped radio [135, 81], these operations could be made more competitive relative to current transmitted-initiated approaches.

Under the current unicast design, a sender S sets it local address to R+`0x8000`, where R is the receiver's hardware address. As a result, S cannot concurrently acknowledge probes from a different receiver, R', for which it also has pending traffic. Richer support for hardware address recognition in the radio would allow a sender to multiplex listening for a probe. For example, a radio could filter for multiple source or destination addresses in parallel. Some radios, like the TI CC2520 [136], can already filter frames on up to twelve different source addresses but these frames must be sent to a unicast destination address (meaning the approach outlined in this paper will not benefit).

6.6 Summary

The central message of this chapter is that while it is relatively easy to optimize for a narrow range of operating conditions, like modern low-power listening protocols do, it is more difficult to find general solutions that work well across a broad spectrum of conditions. This chapter argues that the backcast primitive is better suited to the range of services needed by modern low-power wireless systems: low duty cycles, predictable operation, channel efficiency, and support for bursty workloads. This work paves the way for new research in the design of radio hardware, MAC sub-layer primitives, MAC-layer services, and performance studies to more deeply assess the utility and performance of this approach in meeting emerging application needs like mobile sensing or ultra-low duty cycles.

Chapter 7

Evaluation

This chapter evaluates our architecture by presenting three different applications and discussing how each leverages the work we present in this dissertation. The three applications – AutoWitness, SleepTrack, and CommonSense – are representative of asset tracking, health and fitness, and participatory sensing, and they each represent a distinct point in the system design space. AutoWitness occupies a point that is extremely energy-constrained but experiences rare mobility. CommonSense occupies a point of relatively few energy constraints but experiences frequent and unpredictable mobility. SleepTrack occupies a midpoint between these two applications with modest energy constraints and frequent but short motions. These applications embody the same overall architecture but stress different aspects of it, as show in this chapter.

- Our first application, AutoWitness, uses small, low-cost, and low-power tags affixed to household objects to monitor and track theft. It is, in many ways, a LoJack-like, near-nanopower application that aims to deter, detect, and track theft using a combination of mobile tags and a wide area network of anchors. AutoWitness is representative of a much larger class of asset tracking applications.
- Our second application, SleepTrack, explore the causal effects of periodic limb movements during night time rest, stressing neighbor discovery, motion detection, data filtering, and disruption tolerant networking. It represents a possible blueprint for home-based health and fitness applications, and it is representative of data collection from the home.
- Our third application, CommonSense, demonstrates a truly bottom-up approach to wide-area air quality data collection. It uses a network of handheld mobiscopes carried by people to collect timestamped and geotaggd sensor readings, present this data to its host by tethering to a cell phone, and uploading the data to a central server for larger scale visualization and analysis. CommonSense presents many tradeoffs that allow the system to be tuned from being a high-power, near real-time sensing system to a low-power, intermittently-reporting system. This application is representative of emerging "citizen science" applications in which people collect and share data about their surroundings.

7.1 Autonomous Theft Detection and Tracking with Sleepy Tags

According to the 2007 FBI Uniform Crime Reporting Program, burglary accounted for an estimated 22.1 % of all property crimes, and burglary losses totaled $4.3 billion [140]. Due to the difficulty and expense of investigating such crimes, most go unsolved, and burglary continues to be pervasive. Unfortunately, most

traditional home- and office-based security systems attempt to deter or detect burglary through increased vigilance – security cameras, motion detectors, and alarm systems – but they cannot help track or recover objects once they are stolen.

Existing asset tracking products like Brickhouse [1] and Liveview [2] can track stolen objects. They use GPS to obtain location fixes and cellular infrastructure to communicate this data but unfortunately, the size, cost, and lifetime of such devices is not suitable for use in tracking everyday objects like televisions and microwaves. The LoJack vehicle tracking system [3] uses a small device hidden inside a vehicle to transmit homing beacons when stolen. When a LoJack-equipped vehicle is reported stolen, a network of high-power wireless transmitters send an activation signal to the device. Once activated, devices transmit periodic beacons that can be tracked using police car-mounted LoJack receivers. Unfortunately, this approach requires a device to be powered continuously to receive the activation signal, and requires frequent, high-power transmissions once activated, making it unsuitable for long-term, battery-powered operation (only about three days of operation without access to vehicle power is possible). In addition, a $695 price tag makes the cost prohibitive for tracking everyday objects.

In this section, we present a burglar tracking system called *AutoWitness* that addresses the drawbacks of earlier systems and better matches the cost, size, and power requirements of everyday theft detection and tracking. The AutoWitness system consists of battery-powered *tag* nodes that are attached to everyday objects and a city-wide network of *anchor* nodes that enable energy-efficient tracking in real-time. Although the goal of the system is to lead to the arrest of burglars, rather than merely deterring them to attack more vulnerable targets, installation of theft detection systems are known to have deterrent effects.

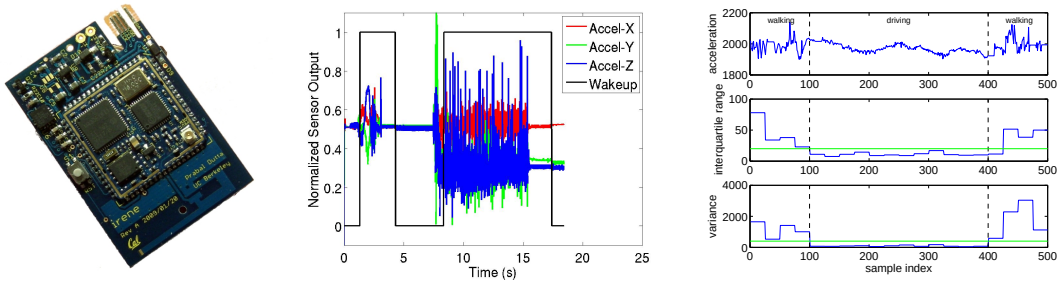
7.1.1 Tag Node

The key hardware design challenges for the tag node revolve around the size, power, and cost requirements of this application. Tags must be small and unobtrusive if they are to be hidden inside of everyday objects like computers, TVs, and stereo equipment. Once deployed, tags must operate unattended for many years without maintenance. To be viable, tags must cost substantially less than the assets they protect. More concretely, the long-term design goal is to build tags that are smaller than 10 cm^3, can last for 10 years, and eventually cost less than $10.

Figure 7.1 shows Irene, a prototype tag mote based on the Epic Core [48], and its operation. Since we are currently using this platform for evaluating design tradeoffs, it has several additional features that would not be part of the final production design, such as rechargeable Li+ battery, dual linear voltage regulators, dual pushbuttons, a secondary wakeup circuit, and 2 MB external flash memory. A production design would also use an integrated microcontroller and radio unit to lower costs. The tag node detects theft autonomously using a hierarchical wakeup system of passive and active vibration sensors. The radio is turned off until motion is detected, a theft signature is classified, and the stolen asset is presumably being transported by a vehicle. The vibration sensor serves other purposes including arming/disarming the system.

7.1.2 Battery

The battery plays an important role in this application, so its selection criteria is critical, and they include self-discharge rate (which affects shelf-life), energy density (which affects size), and cost (which affects viability). The common lithium mangenese dioxide ($LiMnO_2$) primary cell is a good fit for this application. These batteries exhibit a shelf-life of over 10 years at room temperature and are often used



(a) Tag Node (b) Motion Detection (c) Theft Classification

Figure 7.1: Prototype tag hardware. (a) The tag mote integrates an Epic Core, dual vibration switches, an accelerometer, and a rechargeable Li+ battery in a 51 mm x 34 mm x 10 mm footprint. (b) The motion detection circuit in operation. Acceleration bias is removed and the readings are scaled. (c) Acceleration measurements for a walking-driving-walking sequence (top), inter-quartile range (middle) and variance (bottom) of first difference with zeros removed. Horizontal lines show possible classification thresholds.

as a permanent component for the entire lifetime of a system. Their bulk volumetric energy density is approximately 600 mWh/cm^3, although for some small batteries like photo/coin cells, the effective volumetric energy density can be lower due to packaging overhead. Commonly-available lithium coin cells in the CR family, like the CR2032, are widely-used in consumer products, making them relatively inexpensive.

Although alkaline primary cells also have low self-discharge rates, their volumetric energy density is half of the lithium primary cells, which increases size, and their terminal voltage drop makes voltage regulation more important. Common secondary (rechargeable) cells like NiCad, NiMH, Li+, and LiPoly chemistries have higher self-discharge rates, less than half the energy density, and a higher cost per watt-hour than lithium, making them ill-suited to this application.

As a concrete design point, the Energizer CR2032 has a 10+ year shelf-life (losing only 15-20% of its capacity at room temperature), provides an energy density of 653 mWh/cm^3 (supplying over 200 mAh in a 1 cm^3 package), and is available for less than $1 through retail channels (and substantially less in bulk) [56]. These figures translate to approximately 2.5 μA-decade/cm^3 charge density which implies that the average current draw must be less than 2μA to achieve a 10-year lifetime.

7.1.3 Motion Detection

The basic detection problem is to distinguish an object at rest from an object in (prolonged) characteristic motion while drawing less than 2 μA. We use a *vibration dosimeter*, shown in Figure 7.2, to perform this function. The sensor is an omni-directional vibration switch that is nominally closed at rest but chatters open and closed in response to movement [123]. The switch is connected to ground on one terminal and in series with a pullup resistor to power. The 2.49 MΩ pullup resistor sets the quiescent current draw of the circuit. At rest, the circuit draws 1.2 μA at 3 V. A capacitor AC-couples the output of the sensor, a first diode steers negative voltage transients to ground, and a second diode steers positive transients to a capacitor that integrates these signals. A resistor in parallel with the integration capacitor slowly discharges the capacitor so that in the absence of motion, the capacitor voltage goes to zero.

Figure 7.1(b) shows the motion detector circuit in operation. Tri-axial acceleration samples taken at 200 Hz are shown with their bias removed and amplitude scaled. The output of the motion detection wakeup circuit can been seen as a pulse that alternates between zero and one as the sensor transitions from rest to

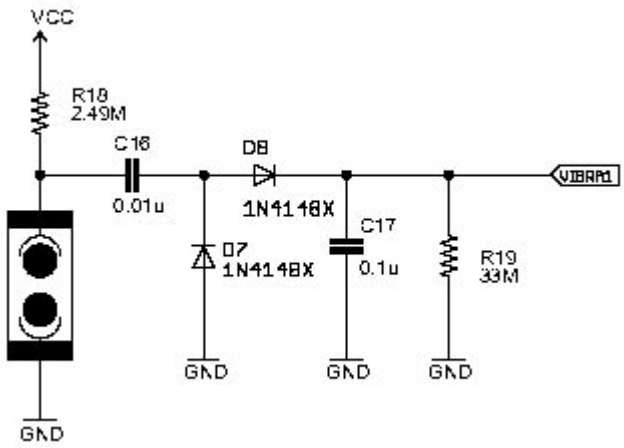


Figure 7.2: Motion detection circuit. A dosimeter integrates the output of a vibration switch and trips after a brief period of continuous motion.

motion. At time t =0.5 s, a tag is picked up and moved and at time t =1.33 s, the motion detector circuit wakeup triggers, waking up the sleeping microcontroller using an interrupt line. At time t =3.09 s, the tag stops moving and time t =4.3 s, the motion detector output indicates movement has stopped. This process repeats for a second, longer and more significant motion starting at time t =7.5 s.

Today, many applications use accelerometers for detecting the onset of motion but commercially-available accelerometers draw too much power for our needs, even when duty-cycled, and they provide a higher resolution than needed. In addition, the accelerometer output would need to be integrated over time, either in hardware or in software, to detect prolonged motion. Because of these drawbacks, our design does not use an accelerometer as the motion detection sensor, but we do use an accelerometer for classifying the theft signatures and interpreting user gestures. In the future, we plan on exploring ways to accomplish these functions without an accelerometer. Another potential motion detection sensor is the piezo-electric vibratab, like the one used in the CargoNet node [98]. Vibratabs and their processing circuits can operate on nanopower budgets but these sensors and their active integrator circuits add size and cost, both of which are at a premium in our design.

7.1.4 Anchor Network

The anchor node network provides the communications infrastructure needed to route beacons from tags to first responders. Anchors also task moving tags with wake/sleep information and they communicate with tag nodes using a low-power, short-range 802.15.4 radio interface provided by a Telos B mote [103]. Anchors associate with nearby 802.11 access points, if available, or they communicate amongst themselves using high-power, long-range proprietary radios that can form a multihop mesh network and are similar in design to DieselNet ThrowBoxes [13]. Anchor nodes are powered by solar cells, which can simplify their deployment at roadway intersections. The solar cells must supply sufficient average power to keep the 802.15.4 interface running at all times and ready to receive packets from any passing tag nodes.

Each anchor node keeps an estimate of the minimum travel time to its nearest neighbor. Upon receiving a theft report beacon from a tag, the anchor responds with this travel time estimate, allowing the tag to sleep

for a substantial fraction of this time. Although not currently implemented, the anchor's travel time estimate should also include a digitally-signed message of the travel time plus a nonce supplied by the tag. Prior work has shown that signature verification using the RSA algorithm using a 1024 bit signature is possible on tag-class nodes [50].

7.1.5 Network Operation

Initially, a Tag node is disarmed and sleeping. When purchased, it can be shipped to a user without raising any theft alarm. Once a user receives a Tag node, he/she arms it before hiding it in an item that is likely to be taken in the event of a burglary. The process of arming/disarming uses a novel mechanism of entering upto 6 digit password (chosen by the user while purchasing the Tag node online/or generated automatically by the system) using the tilt sensor (or accelerometer) in the Tag node. Once armed, the tag mote goes in a deep sleep mode (with just a passive vibration switch active). It wakes up when interrupted by the vibration switch (as a result of significant movement, e.g., jerk, displacement, etc.). Once awake, it collects further readings of the movement using an accelerometer and runs a simple and efficient classification algorithm to determine whether it is being carried in a vehicle. If not, it returns to the deep sleep mode. Otherwise, it enters into the stolen mode and starts looking for an anchor node to notify the law enforcement agency of it being stolen, and its most recent encounter with an Anchor node. During this tracking phase, it runs on a 5% duty cycle, while ensuring rendezvous, with high probability, with anchor nodes on its way. In between its encounters with successive anchor nodes, the Tag node enters a deep sleep mode after having received an estimate of travel time to reach the next Anchor node on its way (saving further energy and enhancing the trackable lifetime by 5-10 times). Figure 7.3 shows the state transitions of the Tag node in the various states together with events that cause the transition among states.

7.2 Tracking of Periodic Limb Movements and their Causal Effects

Digital healthcare represents another application area of interest for mobile sensing. In this section, we present a representative healthcare application that demonstrates many aspects of our low-power mobile sensing architecture. Our goal is to detect night-time periodic limb movements, track and log their causal effects, and upload this data to a central site for monitoring. More specifically, data should only be logged when movement appears to be correlated – when one bed partner appears to moves shortly after the first.

Periodic limb movements affect 29% of population aged 50 to 65 and 44% of those over 65, and contributes to the inability to sleep in 20% of people who have been diagnosed with insomnia. Figure 7.4 shows the node-level state machine for collecting data from nodes.

Nodes are shipped from the factory in the SLEEP state. Upon receipt, a user first plugs in the IPv6 border router into their home network. Then the user presses the START pushbutton which brings the device out of deep sleep and into the READY state. When the integrated light sensor detects darkness (a proxy for night time rest), the accelerometer begins to SENSE data at 25 Hz data rate. If MOV is detected, the node enters in the DISCO TALK state during which time it engages in low-power asynchronous neighbor discovery. If no neighbors are discovered with some TIMEOUT value, the node returns to a SENSE state. If, on the other hand, a rendezvous (RNDVZ) occurs with another node, then the two nodes exchange data (XCHNG DATA) and STORE this data to non-volatile memory. Eventually, the sensor returns to the SENSE state, where it persists, until it is LIGHT out, and the sensor returns to the READY state. It maintains this state the both MOV is detected and DATA is stored and needs to be transmitted. Once that occurs, a node

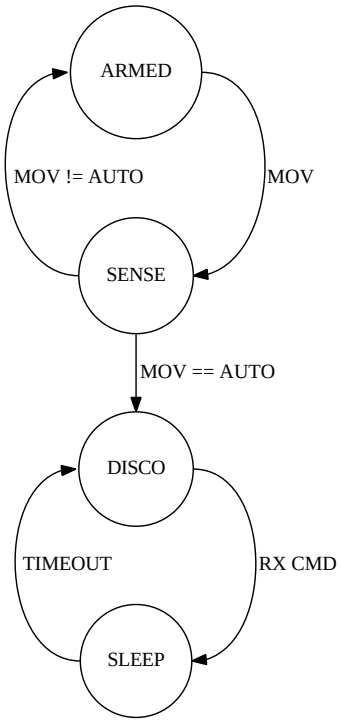


Figure 7.3: The tag node state transition diagram. Tags hibernate in an ultra-low power state until they detect motion and then they operate in a low-power beaconing mode that may be further modulated by sleep commands and motion.

enters the DISCO DOCK state where it scans for a network connection. If none is found, a TIMEOUT occurs and the sensor goes back to the READY state. If a neighbor is discovered, a rendezvous (RNDVZ) occurs, and the cached data is uploaded.

7.3 Distributed Air-Quality Sensing with Handheld Monitors

Poor air quality is a global health issue, causing serious problems like asthma, cancer, and heart disease around the world. Earlier this decade, the World Health Organization estimated that three million people die each year from the effects of air pollution [88]. Unfortunately, while variations in air quality are significant, today's air quality monitors are very sparsely deployed. To address this visibility gap, the Common Sense project is developing participatory sensing systems that allow individuals to measure their personal exposure, groups to aggregate their members' exposure, and activists to mobilize grassroots community action. This system collects environmental data from the Oakland/Berkeley neighborhoods and displays it in real-time [41].

7.3.1 Overview

Data about air quality could enable important advances in medicine, science, and policy. However, only limited air quality readings are currently available. At present, air quality readings are typically taken by official organizations in a relatively small number of fixed locations. These readings are made with carefully

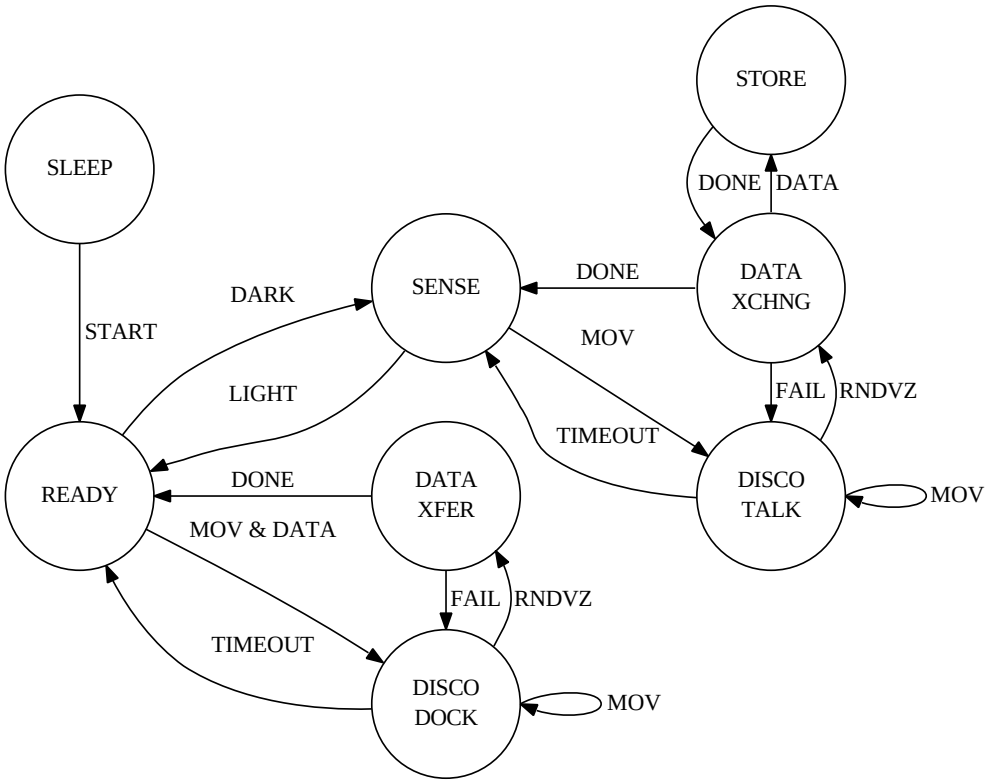


Figure 7.4: The sleep track state machine.

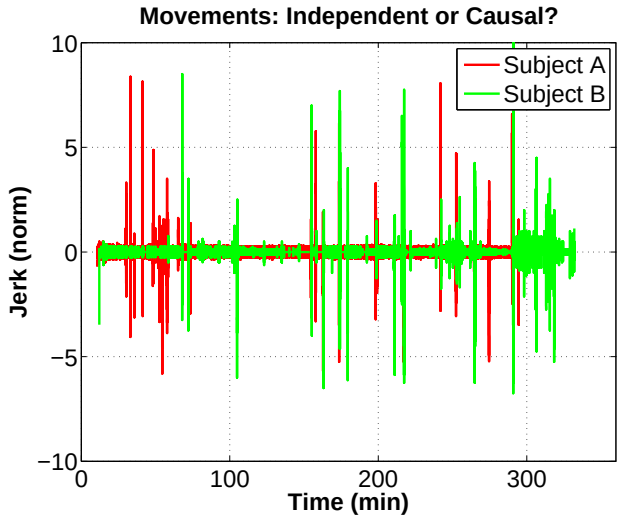


Figure 7.5: An example of night time movements. Dozens of major movements and hundreds of smaller movements are observed. The challenge is capturing the movements that are correlated in time in a statistically meaningful manner.

calibrated professional equipment, and therefore often have the advantage of being highly accurate. This system offers many benefits and meets regulatory specifications. However, it has nonetheless been critiqued as being limited in the number of monitoring sites (for example, there is only one site for the City of San Francisco), as well as for failing to represent the air that citizens breathe on a day-to-day basis (due in part to the fact that regulations require monitoring intakes to be located high above the ground and well away from highways, railroads, and other identified pollution sources to ensure that monitoring sites collect "representative" values rather than "peak" values).

In this application, we employ a mobile participatory sensing [6] approach to collecting air quality data. Mobile participatory sensing uses consumer electronics (e.g., mobile phones) to capture, process, and disseminate sensor data, complementing alternative architectures (e.g., wireless sensor networks) by "filling in the gaps" where people go but sensor infrastructure has not yet been installed. While some types of sensors are already commonly present in consumer devices (e.g., geolocation, motion, sound, etc.), other kinds of compact, low-power sensors (e.g., air quality) are not yet commonly included but offer the ability to collect additional data of individual and social interest. Our work is in the spirit of "citizen science" [80] or "street science" [27], in the hopes of enabling everyday citizens to collect politically relevant data and participate in the decision making process.

7.3.2 Implementation

If the kind of sensing-based applications we envision prove out, we would expect the required sensors to be integrated directly into mobile devices. For prototyping, however, we have developed a suite of board designs and embedded software that can be deployed with associated mobile devices or in a stand-alone configuration, as Figures 7.6(a) and 7.6(b) show. In our current implementation, sensor readings and GPS space-time stamps are sent to a database server using a GPRS radio.

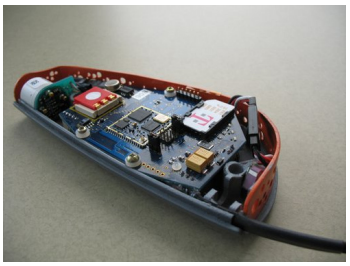
We are also developing mobile and Internet-based visualization tools and community features to support collaborative online interpretation of interesting phenomena and collective development of strategies for action, as Figures 7.6(c) and 7.6(d) show.

The current design takes sensor readings every 5 or 10 minutes and transmits these readings in real-time. The samples are taken, and the data are transmitted without regard to their value or the mobility of the node. With this strategy, we are able to operation the device for about 16 hours using a 1250 mAhr lithium polymer battery. This allows the device to converge with daily patterns of human activity, including battery recharge.

Going forward, the design and implementation of handheld air quality monitors raise a range of research challenges. Since monitor "deployment" is not a carefully controlled process – different users will carry or mount the monitors in different ways – there may be time-varying biases in the readings that must be detected and corrected. Sensor data can be acquired on different timescales, triggered by different actions, and delivered in a multitude of different ways. For example, sensor data may be collected periodically, randomly, or non-uniformly. Sensor data acquisition, and especially GPS position fixes, may be triggered in response to detected motion – the MOV metric – rather than continuously, to save battery drain and extend lifetime. Similarly, sensor readings may be delivered via Bluetooth, 802.15.4, or GPRS radios, depending on either user interest or data entropy.

Striking a balance between the needs of the individual and the needs of the group – between longer sensor life and higher global data fidelity or between privacy and accountability – remains a dynamic challenge but contextual information in general, and mobility and environmental cues in in particular, can improve battery life. How much could contextual information help? Assuming the sensor is attached to a purse

(a) Enclosure



(b) Electronics

(c) Web Interface

(d) Web Interface – detail

Figure 7.6: The current hardware can be selectively populated with commercial carbon monoxide, nitrogen oxides, and ozone gas sensors as well as light, temperature, relative humidity, and orientation sensors. This sensor data is acquired and processed locally using the Epic Core module [48], which also provides an 802.15.4 interface suitable for connecting with other sensors or low-power networks. The handheld monitor can be tethered to a mobile phone using the handheld's integrated BlueSmiRF Bluetooth module from SparkFun which allows sensor readings to be easily visualized on a mobile phone. Finally, a Cinterion GPRS radio and GPS receiver enable the sensor samples to be space-time stamped and uploaded to a hosted server for dissemination, visualization, and analysis over the web. Photos (a) and (b) by Mazzarello Media and Arts.

or bookbag, and a person spends one hour a day commuting outdoors and the rest of the time indoors, mobility-aware regulation of sensing and reporting could improve lifetime by more than a factor of fifteen. Or consider the case in which sensing occurs continuously, but reporting occurs via Bluetooth tethering to a mobile phone (when in range) and via the GPRS radio at other times. The ratio between the average GPRS power and Bluetooth power would determine the lifetime improvement, especially since the sensor suite (other than GPS) draws relatively little power. Even the GPS average power can reduced substantially by using the low-power MOV circuits presented in this thesis.

Chapter 8

Conclusion

This dissertation presents a landscape of mobile sensornet applications, discusses the many challenges that mobility raises, presents an architecture for micropower mobiscopes, and focuses on a handful of key concerns across several layers of the system stack. We explore the prevailing technology trends and observe that motion sensing technology is approaching near-nanopower budgets, that high-density storage (in the gigabytes) is now less costly than microcontrollers and radios, that energy harvesting technologies are maturing to the point of commercial availability, and that thin-film batteries are available as surface mount components. These enabling technologies, we claim, portend a bright future for micropower mobiscopes.

Within this new context, we elaborate on our earlier claim: although *mobility changes everything*, the mere *knowledge of mobility can help address the ensuing challenges*; in other words, we can turn the mobility *bug* into a *feature*. Accordingly, we suggest that real-time knowledge of motion should be made available by the platform and fully exploited. We propose a new metric that indicates node-level movement, called MOV, and we explore how it can address many mobility-induced challenges in micropower mobiscopes.

We develop a modular hardware platform architecture, built on a decade of experience, that incorporates all essential components. These include the mote core (including processor, radio, and storage), peripheral modules (including motion sensing, storage, and energy harvesting), carrier boards (that provide the platform "glue," include addtional functionality, and satisfy application-specific concerns), and a modular, building block methodology to support overall platform composition and synthesis. Following this approach, we present several concrete design points suitable to mobile sensing, including the Irene mote, Common Sense Badge, and Open Mesh border router.

We develop a mobility-aware network architecture that defines the key services, their interfaces, and their interactions. They include, at the platform layer, the MOV metric, a node-level metric that captures movement in various forms, including shock, vibration, acceleration, and displacement. At the link layer, we develop asynchronous neighbor discovery, link quality prediction, a synchronization primitive, and a receiver-initiated medium access control protocol. At the network and transport layers, we adapt existing IPv6-based solutions to incorporate mobility into routing decisions and develop a delay-tolerant network stack for data delivery over an intermittently-connected network.

We present the implementation and evaluation of all the key elements of the architecture. These include *MOV*, a family of motion detection triggers; *Irene*, a mote that embodies many of the key platform ideas supporting mobility, including hardware support for various forms of motion detection, timekeeping, storage, optional energy metering, and a form factor suitable for mobile sensing applications; *Disco*, an asynchronous neighbor discovery protocol that allows nodes to discover neighbors quickly and efficiently;

Backcast, a link layer synchronization primitive that allows a node to efficiently poll multiple neighbors in parallel; *HotMac*, a receiver-initiated, mobility-aware MAC protocol based on Backcast that supports unicast, broadcast, discovery, wakeup, pollcast; and *TinyDTN*, a mobility-aware, disruption-tolerant network stack that uses transport-layer storage and opportunistic forwarding. We also evaluate the system using complete applications that incorporate the essential elements of the architecture. This implemention shows that micropower mobiscopes are indeed feasible, that knowledge of mobility can vastly improve the power and performance of an application, and that many of the techniques are useful for static sensornets and may be generalized to other systems.

Bibliography

[1] Brickhouse security gps tracking system. http://www.brickhousesecurity.com/gps-tracking-system.html.

[2] Live View GPS Asset Tracker. http://www.liveviewgps.com/all+gps+tracking+products.html.

[3] LoJack Security System for Stolen Vehicle Recovery. http://www.lojack.com/car/Pages/car-works.aspx.

[4] UC Berkeley SmartDust Project. http://robotics.eecs.berkeley.edu/~pister/SmartDust/.

[5] Watts Up? .NET Electricity Meter. https://www.wattsupmeters.com/secure/products.php?pn=0&wai=32&spec=2.

[6] Tarek Abdelzaher, Yaw Anokwa, Peter Boda, Jeff Burke, Deborah Estrin, Leonidas Guibas, Aman Kansal, Samuel Madden, and Jim Reich. Mobiscopes for human spaces. *IEEE Pervasive Computing*, 6(2):20–29, 2007.

[7] Jude Allred, Ahmad Bilal Hasan, Saroch Panichsakul, William Pisano, Peter Gray, Jyh Huang, Richard Han, Dale Lawrence, and Kamran Mohseni. Sensorflock: an airborne wireless sensor network of micro-air vehicles. In *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*, pages 117–129, New York, NY, USA, 2007. ACM.

[8] Analog Devices, Inc. Adxl345: 3-axis, $\pm 2g/\pm 4g/\pm 8g/\pm 16g$ digital accelerometer, 2009. http://www.analog.com/static/imported-files/data_sheets/ADXL345.pdf.

[9] Paul M. Aoki, R. J. Honicky, Alan Mainwaring, Chris Myers, Eric Paulos, Sushmita Subramanian, and Allison Woodruff. A vehicle for research: using street sweepers to explore the landscape of environmental community action. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 375–384, New York, NY, USA, 2009. ACM.

[10] J. Arnbak and W. van Blitterswijk. Capacity of slotted ALOHA in rayleigh-fading channels. *IEEE Journal on Selected Areas in Communications*, 5(2):261–269, Feb 1987.

[11] Atmel Corporation. AT86RF230 Datasheet. Available at: http://www.atmel.com/dyn/products/product_card.asp?part_id=3941.

[12] Atmel Corporation. ATMega AT45DB161D Flash Memory. http://www.atmel.com/dyn/products/product_card.asp?part_id=3772.

[13] Nilanjan Banerjee, Mark D. Corner, and Brian Neil Levine. An Energy-Efficient Architecture for DTN Throwboxes. In *Proceedings of IEEE Infocom*, pages 776–784, May 2007.

[14] G. Barriac, R. Mudumbai, and U. Madhow. Distributed beamforming for information transfer in sensor networks. In *Proceedings of the 3^{rd} International Conference on Information Processing in Sensor Networks (IPSN)*, 2004.

[15] Ari Y. Benbasat and Joseph A. Paradiso. A compact modular wireless sensor platform. In *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*, page 56, Piscataway, NJ, USA, 2005. IEEE Press.

[16] Jan Beutel, Oliver Kasten, Friedemann Mattern, Kay Roemer, Frank Siegemund, and Lothar Thiele. Prototyping Wireless Sensor Network Applications with BTnodes. In *Proceedings of the 1st European Workshop on Wireless Sensor Networks (EWSN 2004)*, 2004.

[17] Shane Blanchard. Quick Start Crystal Oscillator Circuit. In *Proceedings of the IEEE 15th Biennial University/Government/Industry Microelectronics Symposium*, pages 78–81, 2003.

[18] Gaetano Borriello, Waylon Brunette, Matthew Hall, Carl Hartung, and Cameron Tangney. Reminding about tagged objects using passive rfids. In *Ubicomp*, pages 36–53, 2004.

[19] Michael Buettner, Gary V. Yee, Eric Anderson, and Richard Han. X-MAC: a short preamble mac protocol for duty-cycled wireless sensor networks. In *SenSys '06: Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, pages 307–320, New York, NY, USA, 2006. ACM.

[20] C. Carter, S. Yi, P. Ratanchandani, and R. Kravets. Manycast: exploring the space between anycast and multicast in ad hoc networks. In *Proceedings of the 9^{th} annual international conference on Mobile computing and networking (MOBICOM)*, 2003.

[21] Kameswari Chebrolu, Bhaskaran Raman, Nilesh Mishra, Phani Kumar Valiveti, and Raj Kumar. Brimon: a sensor network system for railway bridge monitoring. In *MobiSys '08: Proceeding of the 6th international conference on Mobile systems, applications, and services*, pages 2–14, 2008.

[22] Yuen Hui Chee, M. Koplow, M. Mark, N. Pletcher, M. Seeman, F. Burghardt, D. Steingart, J. Rabaey, P. Wright, and S. Sanders. Picocube: A 1cm3 sensor node powered by harvested energy. In *DAC '08: The 45th ACM/IEEE Design Automation Conference*, pages 114–119, jun 2008.

[23] Tanzeem Khalid Choudhury. *Sensing and modeling human networks*. PhD thesis, Massachusetts Institute of Technology, 2004.

[24] B. N. Chun, P. Buonadonna, A. AuYoung, Chaki Ng, D. C. Parkes, J. Shneidman, A. C. Snoeren, and A. Vahdat. Mirage: A Microeconomic Resource Allocation System for Sensornet Testbeds. In *Proceedings of the 2nd IEEE Workshop on Embedded Networked Sensors (EmNets '05)*, 2005.

[25] Dennis P. Connors and Gregory J. Pottie. Response Initiated Multiple Access (RIMA), a Medium Access Control protocol for satellite channels. In *IEEE GLOBECOM 2000*.

[26] B. W. Cook, A. Berny, A. Molnar, S. Lanzisera, and K. S. J. Pister. Low-power 2.4-ghz transceiver with passive rx front-end and 400-mv supply. *Journal of Solid-State Circuits*, 41:2757–2766, dec 2006.

[27] J. Corburn. *Street Science: Community Knowledge and Environmental Health Justice*. MIT Press, Cambridge, MA, 2005.

[28] Crossbow Technology. Crossbow MICA2 Mote. `http://www.xbow.com/products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf`.

[29] Crossbow Technology. IRIS OEM Module Datasheet. `http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/IRIS_OEM_Datasheet.pdf`.

[30] Crossbow Technology. MICA2Dot Datasheet. `http://www.xbow.com/products/Product_pdf_files/Wireless_pdf/MICA2DOT_Datasheet.pdf`.

[31] Crossbow Technology. MICAz OEM Module Datasheet. `http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz_OEM_Edition_Datasheet.pdf`.

[32] Crossbow Technology. MICAz Datasheet. `http://www.xbow.com/Products/Product\pdf\files/Wireless\pdf/MICAz\Kit\Datasheet.pdf`, 2006.

[33] D. Davis and S. Gronemeyer. Performance of slotted ALOHA random access with delay capture and randomized time of arrival. *IEEE Transactions on Communicationss*, 28(5):703–710, May 1980.

[34] Douglas De Couto, Daniel Aguayo, John Bicket, and Robert Morris. A high-throughput path metric for multi-hop wireless routing. In *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 134–146, 2003.

[35] M. Demirbas, O. Soysal, and M. Hussain. A singlehop collaborative feedback primitive for wireless sensor networks. In *Proceedings of the 27th IEEE Conference on Computer Communications (INFOCOM)*, 2008.

[36] Digi Corporation. XTend OEM RF Modules. `http://www.digi.com/products/wireless/long-range-multipoint/xtend-module.jsp`.

[37] Richard Draves, Jitendra Padhye, and Brian Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 114–128, 2004.

[38] H. Dubois-Ferriere, R. Meier, L. Fabre, and P. Metrailler. TinyNode: A Comprehensive Platform for Wireless Sensor Network Applications. In *Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN'06)*, 2006.

[39] Dust Networks, Inc. Time Synchronized Mesh Protocol. Available at `http://www.dustnetworks.com/docs/TSMP_Whitepaper.pdf`, 2006.

[40] P. K. Dutta and D. E. Culler. System software techniques for low-power operation in wireless sensor networks. In *ICCAD '05: Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design*, pages 925–932, 2005.

[41] Prabal Dutta, Paul M. Aoki, Neil Kumar, Alan Mainwaring, Chris Myers, Wesley Willett, and Allison Woodruff. Common sense: participatory urban sensing using a network of handheld air quality monitors. In *SenSys '09: Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, pages 349–350, 2009.

[42] Prabal Dutta and David Culler. Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications. In *SenSys '08: Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 71–84, 2008.

[43] Prabal Dutta and David Culler. Mobility changes everything in low-power wireless sensornets. In *HotOS-XII: Proceedings of the 12th Workshop on Hot Topics in Operating Systems*, may 2009.

[44] Prabal Dutta, David Culler, and Scott Shenker. Procrastination Might Lead to a Longer and More Useful Life. In *The 6th Workshop on Hot Topics in Networks (HotNets VI)*, 2007.

[45] Prabal Dutta, Mark Feldmeier, Joseph Paradiso, and David Culler. Energy metering for free: Augmenting switching regulators for real-time monitoring. In *IPSN '08: Proceedings of the 7th International Symposium on Information Processing in Sensor Networks*, 2008.

[46] Prabal Dutta, Mike Grimmer, Anish Arora, Steven Bibyk, and David Culler. Design of a wireless sensor network platform for detecting rare, random, and ephemeral events. In *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*, page 70, 2005.

[47] Prabal Dutta, Răzvan Musăloiu-E., Ion Stoica, and Andreas Terzis. Wireless ACK collisions not considered harmful. In *Proceedings of the Workshop on Hot Topics in Networks (HotNets-VII)*, 2008.

[48] Prabal Dutta, Jay Taneja, Jaein Jeong, Xiaofan Jiang, and David Culler. A building block approach to sensornet systems. In *Proceedings of the Sixth ACM Conference on Embedded Networked Sensor Systems (SenSys'08)*, November 2008.

[49] Prabal K. Dutta, Anish K. Arora, and Steven B. Bibyk. Towards radar-enabled sensor networks. In *IPSN '06: Proceedings of the 5th international conference on Information processing in sensor networks*, pages 467–474, 2006.

[50] Prabal K. Dutta, Jonathan W. Hui, David C. Chu, and David E. Culler. Securing the deluge network programming system. In *IPSN '06: Proceedings of the 5th international conference on Information processing in sensor networks*, pages 326–333, 2006.

[51] N. Edmonds, D. Stark, and J. Davis. Mass: modular architecture for sensor systems. In *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*, pages 393–397, apr 2005.

[52] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G-S. Ahn, and A. T. Campbell. The bikenet mobile sensing system for cyclist experience mapping. In *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*, pages 87–101, New York, NY, USA, 2007. ACM.

[53] Shane B. Eisenman and Andrew T. Campbell. Skiscape sensing. In *SenSys '06: Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, pages 401–402, 2006.

[54] J. Elson, S. Bien, V. Bychkovskiy, A. Cerpa, D. Ganesan, L. Girod, B. Greenstein, T. Schoellhammer, T. Stathopoulos, and D. Estrin. EmStar: An Environment for Developing Wireless Embedded Systems Software. *UCLA CENS Technical Report No. 9*, 2003.

[55] J. E. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. In *Proceedings of the* 5^{th} *Symposium on Operating Systems Design and Implementation (OSDI)*, 2002.

[56] Energizer. Energizer cr2032 product datasheet, 2009. `http://data.energizer.com/PDFs/cr2032.pdf`.

[57] Jakob Eriksson, Lewis Girod, Bret Hull, Ryan Newton, Samuel Madden, and Hari Balakrishnan. The pothole patrol: using a mobile sensor network for road surface monitoring. In *MobiSys '08: Proceeding of the 6th international conference on Mobile systems, applications, and services*, pages 29–39, New York, NY, USA, 2008. ACM.

[58] Emre Ertin, Anish Arora, Rajiv Ramnath, Vinayak Naik, Sandip Bapat, Vinod Kulathumani, Mukundan Sridharan, Hongwei Zhang, Hui Cao, and Mikhail Nesterenko. Kansei: A Testbed for Sensing at Scale. In *Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN '06)*, 2006.

[59] Deborah Estrin, Ramesh Govindan, John Heidemann, and Satish Kumar. Next century challenges: Scalable coordination in sensor networks. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 263–270, 1999.

[60] Mark C. Feldmeier. *Personalized Building Comfort Control*. PhD thesis, Massachusetts Institute of Technology, 2009.

[61] Rodrigo Fonseca, Prabal Dutta, Philip Levis, and Ion Stoica. Quanto: Tracking energy in networked embedded systems. In *8th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2008, December 8-10, 2008, San Diego, California, USA, Proceedings*, pages 323–338.

[62] J. J. Garcia-Luna-Aceves and Asimakis Tzamaloukas. Reversing the collision-avoidance handshake in wireless networks. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 120–131, 1999.

[63] David Gay, Phil Levis, Rob von Behren, Matt Welsh, Eric Brewer, and David Culler. The nesC language: A holistic approach to networked embedded systems. In *PLDI'03: Proceedings of the SIGPLAN Conference on Programming Language Design and Implementation*, June 2003.

[64] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection tree protocol. In *Proc. of the 7th ACM Conf. on Embedded Networked Sensor Systems (SenSys), Berkeley, CA*, 2009.

[65] Ben Greenstein, Christopher Mar, Alex Pesterev, Shahin Farshchi, Eddie Kohler, Jack Judy, and Deborah Estrin. Capturing High-Frequency Phenomena Using a Bandwidth-Limited Sensor Network. In *Proceedings of the 4th ACM Conference on Embedded Networked Sensor Systems (SenSys'06)*, 2006.

[66] Steven A. Gronomeyer and Alan L. McBride. MSK and offset QPSK modulation. *IEEE Transactions on Communications*, 24(8), 1976.

[67] Daniel Halperin, Josephine Ammer, Thomas Anderson, and David Wetherall. Interference Cancellation: Better Receivers for a New Wireless MAC. In *The 6th Workshop on Hot Topics in Networks (HotNets VI)*, 2007.

[68] Tian He, Sudha Krishnamurthy, John A. Stankovic, Tarek Abdelzaher, Liqian Luo, Radu Stoleru, Ting Yan, Lin Gu, Jonathan Hui, and Bruce Krogh. Energy-efficient surveillance system using wireless sensor networks. In *MobiSys '04: Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pages 270–283, 2004.

[69] Ted Herman, Sriram V. Pemmaraju, Laurence Pilard, and Morten Mjelde. Temporal partition in sensor networks. In *SSS '07: 9th International Symposium on Stabilization, Safety, and Security of Distributed Systems*, pages 325–339, 2007.

[70] Ted Herman, Sriram V. Pemmaraju, Alberto M. Segre, Philip M. Polgreen, Donald E. Curtis, Jason Fries, Chris Hlady, and Monica Severson. Wireless applications for hospital epidemiology. In *WiMD '09: Proceedings of the 1st ACM international workshop on Medical-grade wireless networks*, pages 45–50, New York, NY, USA, 2009. ACM.

[71] Jason Hill and David Culler. Mica: A Wireless Platform for Deeply Embedded Networks. *IEEE Micro*, 22(6):12–24, November 2002.

[72] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David E. Culler, and Kristofer S. J. Pister. System Architecture Directions for Networked Sensors. In *ASPLOS-IX: Architectural Support for Programming Languages and Operating Systems*, pages 93–104, 2000.

[73] Jason L. Hill. *System Architecture for Wireless Sensor Networks*. PhD thesis, University of California, Berkeley, 2003.

[74] R.J. Honicky, E. Brewer, E. Paulos, and R. White. N-SMARTS: Networked suite of mobile atmospheric real-time sensors. In *Proceedings of the Second ACM SIGCOMM Workshop on Networked Systems for Developing Regions (NSDR'08)*, August 2008.

[75] Jyh-How Huang, Saqib Amjad, and Shivakant Mishra. Cenwits: a sensor-based loosely coupled search and rescue system using witnesses. In *SenSys '05: Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, pages 180–191, New York, NY, USA, 2005. ACM.

[76] Jonathan W. Hui. *An Extended Internet Architecture for Low-Power Wireless Networks - Design and Implementation*. PhD thesis, University of California, Berkeley, 2008.

[77] Jonathan W. Hui and David E. Culler. IP is dead, long live IP for wireless sensor networks. In *Proceedings of the 6th International Conference on Embedded Networked Sensor Systems, SenSys 2007, Raleigh, NC, USA, November 5-7, 2008*, pages 15–28.

[78] IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks. Specific requirements – Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs). Available at `http://www.ieee802.org/15/pub/TG4.html`, May 2003.

[79] Intel Corporation. Intel iMote. `http://www.intel.com/research/exploratory/motes.htm`.

[80] A. Irwin. *Citizen Science: A Study of People, Expertise and Sustainable Development*. Routledge, 1995.

[81] Jennic. Wireless Microcontrollers: JN5121 and JN513x. Available at `http://www.jennic.com/products/`, 2007.

[82] Jehn-Ruey Jiang, Yu-Chee Tseng, Chih-Shun Hsu, and Ten-Hwang Lai. Quorum-based asynchronous power-saving protocols for IEEE 802.11 ad hoc networks. *Mobile Networks and Applications*, 10(1-2):169–181, 2005.

[83] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li-Shiuan Peh, and Daniel Rubenstein. Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '02)*, 2002.

[84] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Next century challenges: mobile networking for "smart dust". In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 271–278, New York, NY, USA, 1999. ACM.

[85] Sachin Katti, Shyamnath Gollakota, and Dina Katabi. Embracing Wireless Interference: Analog Network Coding. In *Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '07)*, 2007.

[86] B.S. Kerner, C. Demir, R.G. Herrtwich, S.L. Klenov, H. Rehborn, M. Aleksic, and A. Haug. Traffic state detection with floating car data in road networks. In *Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE*, pages 44–49, Sept. 2005.

[87] Sukun Kim, Shamim Pakzad, David Culler, James Demmel, Gregory Fenves, Steven Glaser, and Martin Turon. Health monitoring of civil infrastructures using wireless sensor networks. In *Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN '07)*, 2007.

[88] Alex Kirby. Pollution: A life and death issue. http://news.bbc.co.uk/2/hi/science/nature/4086809.stm.

[89] Branislav Kusy, Prabal Dutta, Philip Levis, Miklos Maroti, Akos Ledeczi, and David Culler. Elapsed time on arrival: A simple and versatile primitive for canonical time synchronization services. *International Journal of Ad hoc and Ubiquitous Computing*, 2, 2006.

[90] Jeremie Leguay, Timur Friedman, and Vania Conan. Evaluating mobility pattern space routing. In *INFOCOM'06: Proceedings of the 25th IEEE Conference on Computer Communications*, 2006.

[91] Yuan Li, Wei Ye, and John Heidemann. Energy and latency control in low duty cycle MAC protocols. In *IEEE WCNC '05: Proceedings of the IEEE Wireless Communications and Networking Conference*, 2005.

[92] Joshua Lifton, Mark Feldmeier, Yasuhiro Ono, Cameron Lewis, and Joseph A. Paradiso. A Platform for Ubiquitous Sensor Deployment in Occupational and Domestic Environments. In *Proceedings of the 6th international Conference on Information Processing in Sensor Networks (IPSN '07)*, 2007.

[93] Ting Liu, Christopher M. Sadler, Pei Zhang, and Margaret Martonosi. Implementing software on resource-constrained mobile sensors: experiences with Impala and ZebraNet. In *MobiSys '04: Proceedings of the 2nd International conference on Mobile Systems, Applications, and Services*, pages 256–269, New York, NY, USA, 2004. ACM.

[94] Konrad Lorincz, Bor-rong Chen, Geoffrey Werner Challen, Atanu Roy Chowdhury, Shyamal Patel, Paolo Bonato, and Matt Welsh. Mercury: a wearable sensor network platform for high-fidelity motion analysis. In *SenSys '09: Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, pages 183–196, New York, NY, USA, 2009. ACM.

[95] Jiakang Lu and Kamin Whitehouse. Exploiting the capture effect for low-latency flooding in wireless sensor networks. In *INFOCOM*, 2009.

[96] Dimitrios Lymberopoulos, Nissanka B. Priyantha, and Feng Zhao. mPlatform: A Reconfigurable Architecture and Efficient Data Sharing Mechanism for Modular Sensor Nodes. In *Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN '07)*, 2007.

[97] Samuel R. Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. TinyDB: An Acquisitional Query Processing System for Sensor Networks. In *ACM Transactions on Database Systems*, 2005.

[98] Mateusz Malinowski, Matthew Moskwa, Mark Feldmeier, Mathew Laibowitz, and Joseph A. Paradiso. Cargonet: a low-cost micropower sensor node exploiting quasi-passive wakeup for adaptive asychronous monitoring of exceptional events. In *SenSys '07: Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*, pages 145–159, New York, NY, USA, 2007. ACM.

[99] Miklós Maróti, Branislav Kusy, Gyula Simon, and Ákos Lédeczi. The flooding time synchronization protocol. In *SenSys '04: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pages 39–49, 2004.

[100] Maxfor Technology, Inc. TIP Datasheet. `http://maxfor.co.kr/sub5_1.html`.

[101] Michael J. McGlynn and Steven A. Borbash. Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks. In *MobiHoc '01: Proceedings of the 2nd ACM International Symposium on Mobile Ad hoc Networking & Computing*, pages 137–145, 2001.

[102] Somnath Mitra, Zizhan Zheng, Santanu Guha, Animikh Ghosh, Prabal Dutta, Bhagavathy Krishna, Kurt Plarre, Santosh Kumar, and Prasun Sinha. An affordable, long-lasting, and autonomous theft detection and tracking system. In *SenSys '09: Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, pages 351–352, New York, NY, USA, 2009. ACM.

[103] MoteIV Corporation. Tmote Sky. Available at: `http://www.sentilla.com/moteiv-endoflife.html`.

[104] R. Mud, J. Boer, A. Kamerman, H. Van Driest, W. Diepenstraten, R Kopmeiners, and H. Von Bokhorst. Wireless LAN with enchanced capture provision. US Patent No. US5987033, 19919.

[105] muRata. PKGS-00LC-R: Piezoelectric Ceramic Sensors (PIEZO TITE) Shock Sensors, 2009.

[106] R. Musăloiu-E., C.-J. Liang, and A. Terzis. Koala: Ultra-low power data retrieval in wireless sensor networks. In *Proceedings of the 7^{th} International Conference on Information Processing in Sensor Networks (IPSN)*, 2008.

[107] Ivan Niven, Herbert S. Zuckerman, and Hugh L. Mongomery. *An Introduction to the Theory of Numbers*. John Wiley & Sons, 1991.

[108] Subbarayan Pasupathy. Minimum Shift Keying: A spectrally efficient modulation. *IEEE Communications Magazine*, 1979.

[109] Shyamal Patel, Konrad Lorincz, Richard Hughes, Nancy Huggins, John H. Growdon, Matt Welsh, and Paolo Bonato. Analysis of feature space for monitoring persons with parkinson's disease with application to a wireless wearable sensor system. In *Proceedings of the 29th IEEE EMBS Annual International Conference*, August 2007.

[110] E. Paulos, R.J. Honicky, and B. Hooker. *Citizen Science: Enabling Participatory Urbanism*. IGI Global, 2008.

[111] Joseph Polastre, Jason Hill, and David Culler. Versatile low power media access for wireless sensor networks. In *Sensys'04: Proceedings of the Second ACM Conferences on Embedded Networked Sensor Systems*, 2004.

[112] Joseph Polastre, Robert Szewczyk, and David Culler. Telos: enabling ultra-low power wireless research. In *IPSN '05: Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*, page 48, 2005.

[113] G. J. Pottie and W. J. Kaiser. Wireless Integrated Network Sensors. *Communications of the ACM*, 43(5):51–58, 2000.

[114] M. Ringwald and K. Romer. BitMAC: a deterministic, collision-free, and robust MAC protocol for sensor networks. In *Proceedings of the 2^{nd} European Workshop on Wireless Sensor Networks*, 2005.

[115] Janos Sallai, Branislav Kusy, Akos Ledeczi, and Prabal Dutta. On the scalability of routing integrated time synchronization. In *EWSN'06: Proceedings of the 3rd European Workshop on Wireless Sensor Networks*, pages 115–131, 2006.

[116] Brian Schott, Michael Bajura, Joe Czarnaski, Jaroslav Flidr, Tam Tho, and Li Wang. A modular power-aware microsensor with >1000x dynamic power range. In *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*, page 66, Piscataway, NJ, USA, 2005. IEEE Press.

[117] L. Selavo, A. Wood, Q. Cao, T. Sookoor, H. Liu, A. Srinivasan, Y. Wu, W. Kang, J. Stankovic, D. Young, and J. Porter. LUSTER: Wireless Sensor Network for Environmental Research LUSTER: Wireless Sensor Network for Environmental Research. In *Proceedings of the 5th ACM Conference on Embedded Networked Sensor Systems (SenSys'07)*, 2007.

[118] Sensinode. NanoSensor 2.4 GHz. `http://www.sensinode.com`.

[119] Sentilla Corporation. Tmote Mini Datasheet. `http://www.sentilla.com/pdf/eol/Tmote_Mini_Datasheet.pdf`.

[120] Sentilla Corporation. Tmote Sky Datasheet. `http://www.sentilla.com/pdf/eol/tmote-sky-datasheet.pdf`.

[121] SignalQuest Precision Microsensors. SQ-SEN-200 Application Note: Application Circuits, 2009. `http://www.signalquest.com/sq-sen-200.htm`.

[122] SignalQuest Precision Microsensors. SQ-SEN-200 Application Note: Sensitivity Control, 2009. `http://www.signalquest.com/sq-sen-200.htm`.

[123] SignalQuest Precision Microsensors. SQ-SEN-200 Omnidirectional Tile and Vibration Sensor, 2009. `http://www.signalquest.com/sq-sen-200.htm`.

[124] Pavan Sikka, Peter I. Corke, Philip Valencia, Christopher Crossman, Dave Swain, and Greg Bishop-Hurley. Wireless Adhoc Sensor and Actuator Networks on the Farm. In *Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN'06)*, 2006.

[125] Hoi-Sheung Wilson So, Giang Nguyen, and Jean Walrand. Practical synchronization techniques for multi-channel mac. In *MobiCom '06: Proceedings of the 12th Annual International Conference on Mobile Computing and Networking*, pages 134–145, 2006.

[126] Jacob Sorber, Alexander Kostadinov, Matthew Garber, Matthew Brennan, Mark D. Corner, and Emery D. Berger. Eon: a language and runtime system for perpetual systems. In *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*, pages 161–174, 2007.

[127] Kannan Srinivasan, Prabal Dutta, Arsalan Tavakoli, and Philip Levis. Some implications of low power wireless to ip networking. In *In Proceedings of the Fifth Workshop on Hot Topics in Networks (HotNets-V)*, nov 2006.

[128] Kannan Srinivasan, Prabal Dutta, Arsalan Tavakoli, and Philip Levis. An empirical study of low power wireless. *ACM Transactions on Sensor Networks*, 2010.

[129] ST Microelectronics. Lis331hh: MEMS digital output motion sensor: ultra low-power, high full-scale 3-axes, nano accelerometer, 2009. .

[130] STMicroelectronics. STM25P80 Flash Memory Datasheet. http://www.st.com.

[131] Yanjun Sun, Omer Gurewitz, and David B. Johnson. RI-MAC: a receiver-initiated asynchronous duty cycle mac protocol for dynamic traffic loads in wireless sensor networks. In *Proceedings of the 6th International Conference on Embedded Networked Sensor Systems, SenSys 2007, Raleigh, NC, USA, November 5-7, 2008*, pages 1–14.

[132] Robert Szewczyk, Alan Mainwaring, Joseph Polastre, and David Culler. An analysis of a large scale habitat monitoring application. In *Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys'04)*, November 2004.

[133] Jay Taneja, Jaein Jeong, and David E. Culler. Design, modeling, and capacity planning for micro-solar power sensor networks. In *IPSN '08: Proceedings of the 7th international Conference on Information Processing in Sensor Networks*, pages 407–418, 2008.

[134] Texas Instruments. CC2420: 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver. Available at http://www.chipcon.com/files/CC2420_Data_Sheet_1_3.pdf, 2006.

[135] Texas Instruments. CC2430: System-on-Chip Solution for 2.4 GHz IEEE 802.15.4 / ZigBee. Available at http://www.ti.com/lit/gpn/cc2430, 2007.

[136] Texas Instruments. CC2520: Second generation 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver. Available at http://www.ti.com/lit/gpn/cc2520, 2007.

[137] J. Thiele, O. Osechas, J. Bitsch, and K. Wehrle. Smart sensors for small rodent observation. In *Sensors, 2008 IEEE*, pages 709–711, Oct. 2008.

[138] Gilman Tolle, Joseph Polastre, Robert Szewczyk, David E. Culler, Neil Turner, Kevin Tu, Stephen Burgess, Todd Dawson, Phil Buonadonna, David Gay, and Wei Hong. A macroscope in the redwoods. In *Sensys'05: Proceedings of the Second ACM Conferences on Embedded Networked Sensor Systems*, pages 51–63, 2005.

[139] Yu-Chee Tseng, Chih-Shun Hsu, and Ten-Yueng Hsieh. Power-saving protocols for IEEE 802.11-based multi-hop ad hoc networks. In *INFOCOM'02: Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Comm. Soc.*, 2002.

[140] FBI UCR. Burglary - Crime in the United States - 2007. Website, 2007. http://www.fbi.gov/ucr/cius2007/offenses/property_crime/burglary.html.

[141] Tijs van Dam and Koen Langendoen. An adaptive energy-efficient MAC protocol for wireless sensor networks. In *SenSys*, pages 171–180, 2003.

[142] Roy Want, Veronica Falcao, and Jonathan Gibbons. The active badge location system. *ACM Transactions on Information Systems*, 10:91–102, 1992.

[143] Tim Wark, Chris Crossman, Wen Hu, Ying Guo, Philip Valencia, Pavan Sikka, Peter Corke, Caroline Lee, John Henshall, Kishore Prayaga, Julian OGrady, Matt Reed, and Andrew Fisher. The Design and Evaluation of a Mobile Sensor/Actuator Network for Autonomous Animal Control. In *Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN'07)*, 2007.

[144] Tim Wark, Wen Hu, Pavan Sikka, Lasse Klingbeil, Peter Corke, Chris Crossman, and Greg Bishop-Hurley. A model-based routing protocol for a mobile, delay tolerant network. In *SenSys '07: Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*, pages 421–422, 2007.

[145] Geoff Werner-Allen, Konrad Lorincz, Jeff Johnson, Jonathan Lees, and Matt Welsh. Fidelity and yield in a volcano monitoring sensor network. In *OSDI'06: The 7th USENIX Symposium of Operating System Design and Implementation (OSDI'06)*, 2006.

[146] Geoffrey Werner-Allen, Patrick Swieskowski, and Matt Welsh. MoteLab: A Wireless Sensor Network Testbed. In *Proceedings of the 4th international Conference on Information Processing in Sensor Networks (IPSN '05)*, 2005.

[147] K. Whitehouse, A. Woo, F. Jiang, J. Polastre, and D. Culler. Exploiting the capture effect for collision detection and recovery. In *Proceedings of the 2^{nd} IEEE workshop on Embedded Networked Sensors (EmNets)*, 2005.

[148] Wei Ye, John Heidemann, and Deborah Estrin. An energy-efficient mac protocol for wireless sensor networks. In *INFOCOM'02: Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies*, June 2002.

[149] Wei Ye, Fabio Silva, and John Heidemann. Ultra-Low Duty Cycle MAC with Scheduled Channel Polling. In *SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 321–334, New York, NY, USA, 2006. ACM.

[150] Pei Zhang, Christopher M. Sadler, Stephen A. Lyon, and Margaret Martonosi. Hardware Design Experiences in ZebraNet. In *Proceedings of the 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys'04)*, 2004.

[151] Xiaolan Zhang, Jim Kurose, Brian Neil Levine, Don Towsley, and Honggang Zhang. Study of a bus-based disruption-tolerant network: mobility modeling and impact on routing. In *MobiCom '07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 195–206, New York, NY, USA, 2007. ACM.

[152] Rong Zheng, Jennifer C. Hou, and Lui Sha. Asynchronous wakeup for ad hoc networks. In *MobiHoc '03: Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking & Computing*, pages 35–45, 2003.