Zippy: On-Demand Network Flooding

Felix Sutton, Bernhard Buchli, Jan Beutel and Lothar Thiele
Computer Engineering and Networks Laboratory
ETH Zurich, Switzerland
{fsutton, bbuchli, beutel, thiele}@tik.ee.ethz.ch

ABSTRACT

In this paper, we tackle the challenge of rapidly disseminating rare events through a multi-hop network, while achieving unprecedented energy-efficiency. Contrary to state-of-the-art approaches, we circumvent the undesirable trade-offs associated with low-power duty-cycled protocols and backscatter technologies, and demonstrate a paradigm shift in low-power protocol design. We present Zippy, an on-demand flooding technique that provides robust asynchronous network wake-up, fine-grained per-hop synchronization and efficient data dissemination by leveraging low-complexity transmitter and receiver hardware. We are the first to demonstrate the on-demand flooding of rare events through a multi-hop network with end-to-end latencies of tens of milliseconds, while dissipating less than 10 microwatts during periods of inactivity. We present a prototype implementation of our proposed approach using a wireless sensor platform constructed from commercially available components. We extensively evaluate Zippy's performance in a laboratory setting and in an indoor testbed.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*wireless communication*

General Terms

Design, Experimentation, Performance

Keywords

Asynchronous rendezvous; on-demand; network flooding; time synchronization; wake-up radio; wake-up receiver; on-off keying; wireless sensor networks; cyber-physical systems

1. INTRODUCTION

Motivation. The past decade of wireless sensor network research has produced a plethora of robust and energy efficient protocols for disseminating *periodic* events through a network of resource-constrained motes arranged in a multi-hop topology. However, there are many real-world applications, such as industrial automation [13], structural health monitoring [19], medical alert systems [35], surveillance systems [43], and environmental monitoring [12], where events are not periodic, but occur only on *rare* occasions. Under this premise, state-of-the-art duty-cycled protocols are faced with an undesirable design trade-off between mote lifetime and end-to-end latency, and state-of-the-art backscatter technologies are limited by either the availability of infrastructure or low communication range. In this paper, we demonstrate a paradigm shift in low-power protocol design, where the aforementioned trade-offs are circumvented, thus enabling the rapid dissemination of rare events through a multi-hop network with unprecedented energy-efficiency.

Challenge. The key challenge is the dissemination of a rare event through a multi-hop network that is both rapid, *i.e.*, with low end-to-end latency, and energy efficient, *i.e.*, maximizes mote lifetime.

In order to exemplify the design trade-offs encountered when applying state-of-the-art techniques, we consider a concrete application scenario, whereby resource-constrained wireless gas sensors must rapidly alert a sink if the measured gas concentration exceeds a safe level. One approach is to apply a synchronous protocol, such as Glossy [8], to rapidly flood the event through the network to a sink. However, since the periodicity of the radio activity is directly linked to the end-to-end latency of the event, all motes in the multi-hop network must expend significant energy in order to achieve a low reporting latency of the event.

An alternative is to employ a Low-Power Listening [34] or Low-Power Probing [27] protocol. However, as demonstrated in [7], these duty-cycled protocols and their variants suffer the same undesirable design trade-off. That is, the need for low end-to-end latency increases the radio activity at each mote in the network, thereby reducing mote lifetime.

Yet another alternative is to employ backscatter technology where a dedicated [44] or ambient [24] high-power infrastructure provides motes with a medium for communication. However, such techniques are limited in their support for multi-hop connectivity as either the high-power infrastructure must be generated, ambient infrastructure may not always be available, or the achievable per-hop communication range is insufficient.

Proposed Approach. We present Zippy, a novel flooding technique that circumvents the aforementioned design trade-offs by leveraging low-complexity transmitter and receiver hardware. We exploit the ultra-low power dissipation

SenSys'15, November 1–4, 2015, Seoul, South Korea.

DOI: http://dx.doi.org/10.1145/2809695.2809705.

and unique timing properties of low-complexity radio hardware to perform on-demand flooding of rare events featuring end-to-end latencies on the order of milliseconds, while dissipating less than 10 microwatts during periods of inactivity. This level of energy efficiency makes it possible to support rare event dissemination for multiple years with low-capacity coin cell batteries.

ZIPPY combines four extensible components, which facilitate its adoption to a wide range of application domains. First, a robust *asynchronous network wake-up* ensures that all motes in the multi-hop network are awoken from a deep sleep state. Second, a novel *neighborhood synchronization* component achieves an average per-hop synchronization as low as 34μs. Third, an efficient bit-level *data propagation* component supports the dissemination of event packets with near constant end-to-end latency. Finally, a *carrier frequency randomization* component ensures that the destructive interference associated with concurrent transmissions are mitigated during a ZIPPY flood.

Contributions. In this paper, we make the following contributions:

- We present a robust scheme for performing on-demand wake-up of a multi-hop network using low-complexity radio hardware.
- We introduce a novel technique for per-hop synchronization and flooding of an event packet through a multi-hop network.
- We introduce the design of an ultra-low power wireless sensor platform supporting ZIPPY, while dissipating less than 10 microwatts during periods of inactivity.
- We detail a prototype implementation of ZIPPY, and extensively evaluate its performance both with controlled laboratory experiments and within an indoor testbed.

The paper is structured as follows: Sec. 2 presents on-demand rendezvous using low-complexity radio hardware. Sec. 3 presents an overview of ZIPPY, followed by a detailed design and analysis of its operation in Sec. 4. A prototype implementation of ZIPPY and its extensive evaluation is provided in Sec. 5. Finally, we conclude with a discussion of related work in Sec. 6, and a summary of conclusions in Sec. 7.

2. ASYNCHRONOUS RENDEZVOUS

ZIPPY exploits an asynchronous rendezvous mechanism to facilitate on-demand network flooding. In this section, we study how a mote can achieve on-demand rendezvous with its one-hop neighbors using low-complexity radio hardware.

2.1 Background

One of the fundamental challenges of low-power wireless communication is how to support energy-efficient packet exchange between single-hop neighbors, while minimizing the energy consumed due to idle listening. Given two arbitrary wireless motes within range, packet exchange is only facilitated when both motes have their radios turned on at the same time, which is termed a rendezvous [23]. If a mote does not know when its neighbor wants to send a packet, it must expend energy by powering its wireless receiver in anticipation of packet exchange. This behavior is termed idle listening [5], and is a significant source of energy waste due to the high power dissipation of commonly used receivers.

Wireless sensor network research has primarily addressed the problem of idle listening in the *time* domain [39]. That is, by duty cycling the radio at the appropriate time, rendezvous between one-hop neighbors is supported, while also reducing the mote's average energy consumption. However, under such duty-cycled schemes, there exists a fundamental trade-off between energy efficiency and how often the motes rendezvous. Either the radio duty-cycle is increased to achieve a high rendezvous rate at the cost of higher energy consumption, or the radio duty-cycle is decreased to reduce energy consumption at the cost of a low rendezvous rate. In the case where on-demand rendezvous is desired, *e.g.*, to rapidly exchange a rare event, a design decision must be made between the energy efficiency of the mote, and the latency for the packet exchange.

An alternative approach is to address the problem of idle listening in the *power* domain. Instead of duty cycling the radio hardware, the power dissipation of the radio is such that it becomes feasible to have it always turned on. There are two variants of this approach, based on the dependency on *infrastructure*, or not.

Backscatter technologies are an *infrastructure*-based approach, where a high-power signal source, *i.e.*, either dedicated in the case of RFID tags [44] or ambient in the case of ambient backscatter [24], provides a medium in which passive RFID tags or backscatter-enabled devices can communicate by reflecting the incident signal. Whilst there has been significant advancements of this novel communication scheme [30], the infrastructure for dedicated backscatter consumes significant energy resources, while realistic signal powers of ambient sources, *e.g.*, from TV transmissions, are likely to constrain indoor non-line-of-sight communication links to only a few meters.

In this paper, we turn our attention to an *infrastructure-less* power domain solution to idle listening. Specifically, we use low-complexity receivers to decode transmissions from low-complexity transmitters. In contrast to backscatter technologies, the receiver is an always-on active circuit specifically designed for ultra-low power operation, while the transmitter is only used when there is data to transmit.

This particular approach has been proposed in the literature [14] more than a decade ago, with the low-complexity receivers referred to as wake-up radios or wake-up receivers. However, recent advances in ultra-low power electronics and the commercial availability of components have made it feasible to achieve always-on idle listening while dissipating on the order of a few microwatts. This is in stark contrast to the order of tens of milliwatts dissipated by commodity radio hardware typically integrated into wireless sensor platforms.

We next present how such ultra-low power receiver structures are realized in practice.

2.2 Asynchronous Mote Architecture

The key to achieving ultra-low power on-demand rendezvous lies in the complexity of the radio hardware and the modulation scheme. Embracing low-complexity digital modulation schemes, such as On-Off Keying (OOK), drastically simplifies the receiver and transmitter circuitry, *i.e.*, by reducing the number and specificity of components, which leads to a significant reduction in power dissipation. In particular, ultra-low power receiver designs, *i.e.*, exhibiting sub-microwatt power dissipation, have been demonstrated in the literature for the acoustic [41], radio frequency [33], and optical [18] spectra.

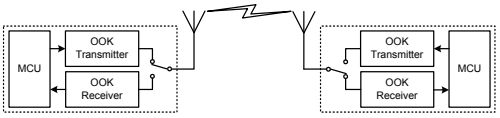

Figure 1: Proposed mote architecture with a microcontroller (MCU), OOK transmitter, and always-on ultra-low power OOK receiver.

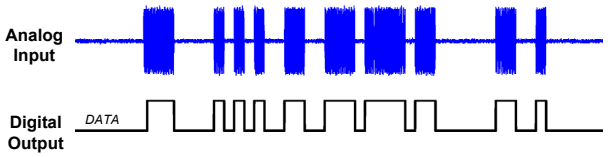

Figure 2: Example OOK signal at the input to the receiver and its corresponding demodulated output.

In this paper, we focus on low-complexity radio hardware for the ISM radio frequency band using the OOK digital modulation scheme. Fig. 1 illustrates the mote architecture assumed throughout this paper, where each mote consists of a microcontroller (MCU) provisioned with an OOK transmitter and an always-on OOK receiver. An RF switch controlled by the MCU selects the RF path from the antenna to either the input of the receiver, or the output of the transmitter. The OOK transmitter is designed such that its power dissipation is less than that of typical transceivers used for higher-complexity digital modulations, *e.g.*, the MAX7044 features a current drain of 14 mA compared to the CC430 which drains 29 mA with identical output transmission power. The OOK receiver is designed such that its power dissipation is on the order of microwatts, *i.e.*, several orders of magnitude below typical transceivers, thereby allowing always-on operation.

On-Off Keying is attributed as being the simplest of digital modulation schemes, where a 1 bit is represented by the presence of a carrier frequency over a bit period T_b, while a 0 bit is represented by the absence of a carrier frequency over a bit period T_b. Fig. 2 illustrates an example OOK-modulated bit sequence, both in its analog representation observed at the antenna of the OOK receiver, and its digital representation after decoding. The OOK receiver provides a single digital output, a *DATA* line, which is interfaced directly to the attached MCU for processing, which will be discussed next.

As illustrated in Fig. 2, the OOK receiver manipulates the digital level of the *DATA* line according to the envelope of the received OOK signal. Since the mote's MCU observes the state of the *DATA* line, it can perform two important functions, namely, *(i)* facilitate asynchronous rendezvous by the detection of the first rising edge of the *DATA* line, and *(ii)* decode an OOK modulated bit sequence by sampling the state of the *DATA* line within each bit period T_b. As discussed in Sec. 3, these two functions provided by the OOK receiver are pivotal to the operation of Zippy.

Despite the superior power-efficiency of low-complexity OOK receivers, they exhibit well-known practical limitations, as surveyed in [6]. We systematically evaluate the two most important limitations in Sec. 5.

We next describe how asynchronous rendezvous using low-complexity radio hardware is applied to a multi-hop network, while supporting tight per-hop time synchronization and energy efficient bit-level data dissemination.

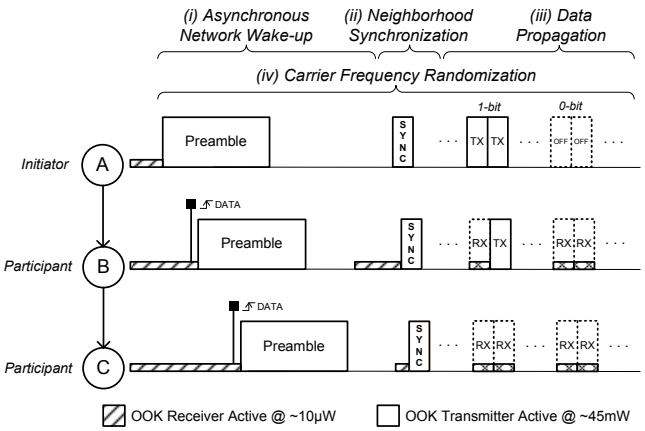

Figure 3: Overview of Zippy in an example 2-hop network, where the initiator mote has a rare event to disseminate through the multi-hop network.

3. OVERVIEW OF ZIPPY

In this section, we present an overview of Zippy, the first protocol to demonstrate the feasibility of on-demand multi-hop flooding using low-complexity radio hardware. Fig. 3 exemplifies the operation of Zippy, where an *initiator*, mote A, disseminates a rare event through a 2-hop network consisting of *participant* motes B and C. We next briefly describe the function of Zippy's four extensible components.

Asynchronous Network Wake-up. The initiator commences a Zippy flood by transmitting a preamble, *i.e.*, a sequence of 1 bits, using its low-complexity OOK transmitter. Since the OOK receiver at mote B is always on, the reception of the preamble will assert the receiver's *DATA* line, awaking its MCU from deep sleep. Mote B will then proceed to transmit a preamble so to wake-up its neighbor, mote C. The relaying of the preamble continues until all motes in the network are awake, and as a consequence, all motes achieve coarse-grained time synchronization, *i.e.*, on the order of milliseconds, to their nearest one-hop neighbors.

Neighborhood Synchronization. Shortly after the completion of the preamble transmission, the initiator transmits a single OOK 1 bit, termed the synchronization bit. Once the receiver of mote B detects the beginning of this bit, *i.e.*, by the rising edge of the receiver's *DATA* line, the mote immediately transmits a synchronization bit. The same process occurs at mote C. The time delay between the beginning of the synchronization bit transmission, and the instant it is detected at the MCU provides fine-grained per-hop time synchronization, *i.e.*, on the order of tens of microseconds.

Data Propagation. All motes now leverage the per-hop time synchronization in order to propagate a packet through the network with bit-level granularity. The packet is encoded using a repetition code, with the length of the code equal to the maximum hop count of the network. In the considered example, each bit of the packet is represented by two sub-bits, *i.e.*, a 1 is transmitted as 11 and a 0 is transmitted as 00. The initiator transmits all packet sub-bits, while each participant uses its OOK receiver to decode each sub-bit in sequence. If a sub-bit is decoded as a 1, the participant switches to its OOK transmitter and transmits any remaining 1 sub-bits, thereby propagating each bit to its neighbors.

Carrier Frequency Randomization. It is evident that transmissions during a Zippy flood will overlap. As shown in [40], the packet reception rate reduces as the number of concurrent transmissions increases. Furthermore, since we use low-complexity OOK receivers, we are not able to leverage the capture effect [20], which is associated with high-complexity receivers. Although it is shown in [2] that OOK receivers may be designed to exhibit the capture effect, the additional hardware increases the total power dissipation of the receiver, potentially negating the benefits of using it for asynchronous rendezvous. Therefore, during the execution of all aforementioned components of Zippy, we mitigate the impact of overlapping OOK transmissions by randomizing the carrier frequency used to represent OOK 1 bits.

While the mote architecture assumed throughout this paper targets the ultra-low power and low-complexity boundaries of the radio sub-system design space, we do not restrict the adoption of Zippy's four extensible components into existing motes and/or protocols. On the contrary, the fundamental components of Zippy may be combined to create innovative mote architectures and/or energy efficient extensions to established protocols. For example, asynchronous network wake-up and carrier frequency randomization provide a widely applicable and robust multi-hop wake-up service, while a state-of-the-art flooding protocol may benefit from the integration of asynchronous network wake-up, neighborhood synchronization and carrier frequency randomization components in order to reduce the energy footprint of disseminating rare events.

4. DESIGN AND ANALYSIS

We next present the design and analysis of Zippy's four extensible components. Where applicable, we characterize certain properties of the OOK receiver so to aid in the design and analysis of each component.

4.1 Asynchronous Network Wake-up

A precondition for communicating with all motes in a network is to ensure they are all turned on and listening, *i.e.*, the mote's MCU is active with a radio interface configured in receive mode. With the assistance of an always-on low-complexity OOK receiver integrated in each mote, the action of remotely waking-up a mote is achieved by the transmission of a preamble, *i.e.*, a fixed-length sequence of 1 bits. As introduced in Sec. 2, the always-on ultra-low power OOK receiver will detect the beginning of the preamble, activate its *DATA* line, and bring the attached MCU out of deep sleep using appropriate interrupt processing.

It follows that the fastest way to facilitate mote wake-up throughout an entire multi-hop network is to simply relay the preamble along each hop. That is, as soon as the *DATA* line is asserted by the OOK receiver, the attached MCU awakes from deep sleep, and proceeds to transmit the preamble. The duration of the preamble $T_{preamble}$ must therefore be longer than the time taken for the OOK receiver to detect the preamble and assert its *DATA* line. Fig. 4 illustrates the detailed timing relations between initiator and participant motes. In order to determine a suitable duration of the preamble, we must measure the delay T_{WAKE} from the beginning of the initiator's preamble transmission until the rising edge of the participant's *DATA* line. This timing characteristic is dependent on two factors: the OOK receiver hardware and the RF propagation effects. We next analyze

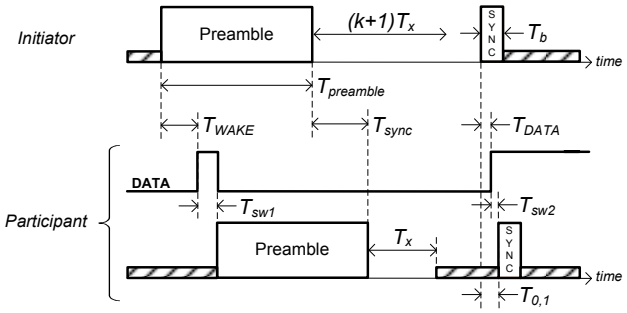

Figure 4: Timing diagram for Zippy's asynchronous network wake-up and neighborhood synchronization.

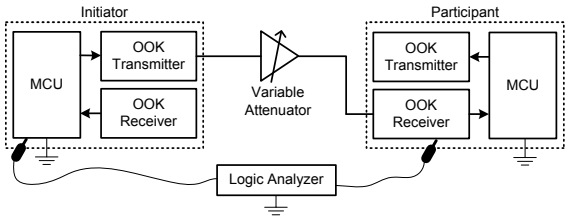

Figure 5: Experimental setup for the timing analysis of the OOK receiver's *DATA* line using a logic analyzer.

the timing properties of the OOK receiver used throughout this paper under ideal RF conditions in order to give a lower bound on the preamble length.

Using the experimental setup illustrated in Fig. 5, we measure the time taken for the receiver to assert its *DATA* line, T_{WAKE}, across the operating range of the receiver as controlled by a variable attenuator. A logic analyzer is used to measure the time between the start of the preamble transmission at the initiator's MCU, and the rising edge of the *DATA* line at the participant's OOK receiver. Fig. 6 plots the distribution of T_{WAKE} as the channel attenuation is varied such that the input signal power to the OOK receiver is between -31 dBm and -51 dBm.

The results indicate a very stable time distribution for T_{WAKE} across the operational range of the receiver. This is primarily due to the Automatic Gain Control (AGC) stage integrated in the OOK receiver detailed in Sec. 5.1. The AGC continuously adjusts a variable-gain amplifier to ensure that low input signals are able to be successfully decoded. Based on these experimental results, the preamble must be at least as long as the maximum observed T_{WAKE}, *i.e.*, $T_{preamble} > 370\,\mu$s.

Once the *DATA* line is asserted at a participant's MCU, the transmission of a preamble commences. The time taken to start the data radio transmission, T_{sw1}, is determined by the start-up time of the transmitter, and the software execution time needed to configure the hardware and appropriately fill the transmit buffer. This software delay can be made approximately constant, *e.g.*, the implementation detailed in Sec. 5.1 exhibits a T_{sw1} of approximately $350\,\mu$s.

At the end of the asynchronous network wake-up, the MCU of each wireless mote is awake, and has coarse-grained time synchronization to their nearest neighbor. Specifically, the per-hop synchronization, T_{sync}, is given by (1), where T_{prop} is the propagation delay of the carrier signal.

$$T_{sync} \geq T_{WAKE} + T_{sw1} + T_{prop} \quad (1)$$

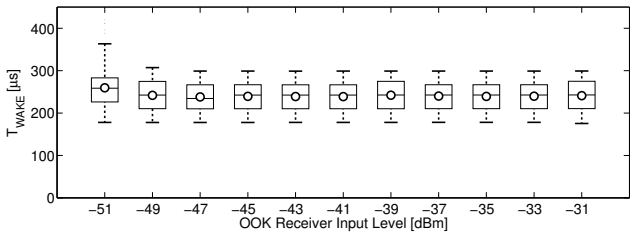

Figure 6: Delay to trigger the rising edge of the *DATA* line with respect to the receiver's input signal level.

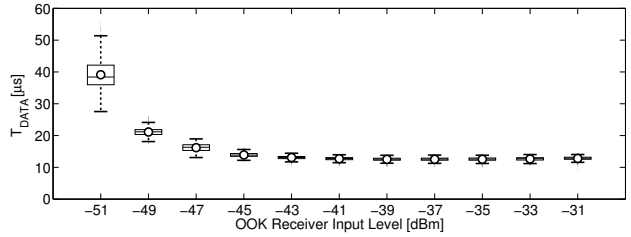

Figure 7: Delay to assert the *DATA* line in response to a neighbor's transmission of a 1 bit.

Based on the aforementioned characterization of the OOK receiver, the per-hop synchronization will be on the order of a millisecond. We next investigate how the per-hop time synchronization can be improved by further leveraging the timing properties of the low-complexity OOK receiver.

4.2 Neighborhood Synchronization

Since per-hop time synchronization is a prerequisite for rapid data dissemination through a multi-hop network, we aim to improve the per-hop time synchronization achieved by the asynchronous network wake-up. We take advantage of the fact that low-complexity OOK receivers decode on a bit-level, as opposed to a byte-wise or packet-wise reception offered by high-complexity receivers, *e.g.*, 802.15.4 compatible receivers. As introduced in Sec. 2, the OOK receiver provides a digital *DATA* line that represents the envelope of a received OOK signal. We now leverage the time it takes for the *DATA* line to assert, T_{DATA}, given that a neighboring mote has transmitted a 1 bit, in order to tightly synchronize motes to their neighbors. As indicated in Fig. 4, the delay T_{DATA} is expected to be smaller than T_{WAKE} since the receiver's AGC has settled to a steady state after the reception of the preamble.

Using the experimental setup illustrated in Fig. 5, we measure the time between the beginning of a 1 bit transmission from the initiator, until the rising edge of the participants *DATA* line using a logic analyzer. The time separating the preamble and the 1 bit transmissions is chosen to be longer than the receiver's AGC settling time. Fig. 7 illustrates the box plot of the delay T_{DATA}, as the OOK receiver's input signal level varies between $-31\,\text{dBm}$ and $-51\,\text{dBm}$. We observe a statistically stable delay with an average of approximately 13μs for low and medium signal strengths. The average increases by a factor of three as the input signal power approaches the receiver's sensitivity level of $-52\,\text{dBm}$, as evaluated in Sec. 5.1.1.

We leverage this advantageous timing property of the OOK receiver to achieve per-hop synchronization on the order of tens of microseconds. As illustrated in Fig. 4, once the asynchronous network wake-up is complete, the initiator transmits a synchronization bit, *i.e.*, a single 1 bit of duration T_b, using its OOK transmitter. The participant's OOK receiver will detect this bit, and assert its *DATA* line after an elapsed time of T_{DATA}. The participant's MCU then begins to transmit a synchronization bit using its OOK transmitter. The time to switch on the OOK transmitter, T_{sw2}, is a combination of hardware and software delays which are considered constant.

The relaying of the synchronization bit continues throughout the multi-hop network. In an ideal RF propagation environment, the time between the start of the synchronization bit transmission at hop i, and the start of the synchronization bit transmission at hop $i+1$ is at least $T_{i,i+1}$ seconds, as given by (2). The constant T_{prop} represents the carrier signal propagation time, which is considered negligible.

$$T_{i,i+1} \geq T_{data} + T_{sw2} + T_{prop} \tag{2}$$

The aforementioned neighborhood synchronization scheme has two caveats that necessitate further discussion. Firstly, once a participant has completed its carrier burst transmission, it must wait a minimum of T_x seconds before turning on its receiver in preparation for the relaying of the synchronization bit. If the receiver is turned on too early, the *DATA* line will activate in response to a neighbors preamble transmission, causing erroneous synchronization. Therefore, each participant must wait at least $T_x = \max(T_{WAKE}) + T_{sw1}$ seconds before reacting to the rising edge of its receiver's *DATA* line.

Secondly, the initiator must ensure that the synchronization bit is not transmitted before all motes within the multi-hop network have completed their preamble transmissions. Therefore, the initiator must know in advance the maximum number of hops the flood must propagate through. This poses no real practical limitation, as the maximum expected hop count can be estimated at the deployment time of the network. It follows that given a multi-hop network of k hops, the initiator must wait at least $(k+1)T_x$ seconds before transmitting the synchronization bit.

We show experimentally in Sec. 5 that a mean per-hop synchronization as low as 34μs is achievable in a 2-hop network, while only taking on the order of milliseconds to complete the Zippy flood.

4.3 Data Propagation

We next leverage the per-hop time synchronization in order to rapidly propagate a fixed length packet through a given multi-hop network. Since the OOK receiver provides bit-level granularity, we consider the bit-level propagation of a packet between single-hop participants.

The simplest method to propagate a bit through a multi-hop network is to transmit the bit as soon as it is received. This scheme is exemplified in Fig. 8(a), where the initiator transmits bit b_0, participant mote B receives the bit and transmits it in the next available bit period. This bit-level relaying continues throughout the multi-hop topology. Under this scheme, the initiator must know in advance the maximum hop count k of the network in order to determine when it can transmit the next bit b_1.

We therefore impose a structure where each bit consists of k sub-bits, where k is the maximum hop count of the network. All participant motes maintain a local counter j, which is initialized to zero. The counter represents the sub-

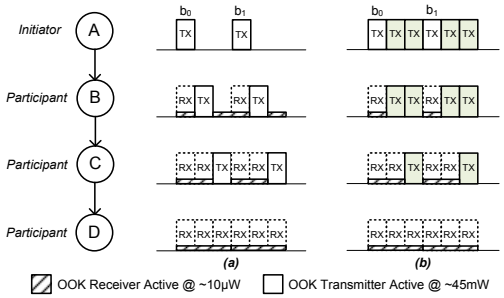

Figure 8: Bit-level propagation schemes using $k = 3$ sub-bits (a) without, and (b) with repetition.

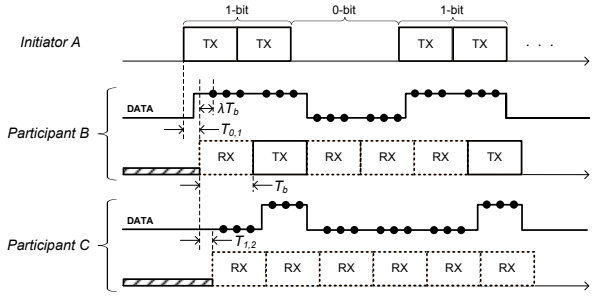

Figure 9: Bit-level propagation employed by Zippy with $n = 3$ majority vote sub-bit decoding.

bit, *i.e.*, between 0 and $k-1$, currently decoded by the participant. The counter is incremented by the participant at the end of each symbol boundary according to the following sequence:

$$j = (j+1) \bmod k \quad \forall j \in [0, k-1] \tag{3}$$

Once the sub-bit has been decoded, *i.e.*, the *DATA* line has been sampled and is deemed as either a 0 or a 1, the participant makes a decision on how to propagate the sub-bit based on the value of the counter j. If a sub-bit is decoded as a 0, the participant continues to listen for the next sub-bit. However, if a sub-bit is decoded as a 1, the participant will transmit a 1 during the next sub-bit if its counter value satisfies $j < k-1$. This behavior is illustrated in Fig. 8(a), where mote B decodes a 1 during sub-bit $j = 0$, and therefore transmits a 1 in the next sub-bit. Similarly, mote C decodes a 1 during sub-bit $j = 1$, and transmits a 1 in the next sub-bit. This scheme of bit-level data propagation ensures that all participants up to the k^{th} hop have an opportunity to receive each bit transmitted by the initiator.

The reliability of this scheme can be further improved by *(i)* incorporating a repetition code for the initiator and allowing participants to transmit during empty sub-bits, and *(ii)* performing a majority vote sub-bit decoding at all participants. We next detail these two techniques, which are both incorporated into Zippy.

As illustrated in Fig. 8(b), the initiator of a Zippy flood employs a k-repetition code, whereby a single bit is transmitted repeated $k-1$ times. In order to leverage the transmission of the additional sub-bits from the initiator, each participant may propagate all remaining 1 sub-bits. That is, if a participant decodes the j^{th} sub-bit as a 1, the remaining $k-j-1$ sub-bits are then transmitted as 1 bits. To exemplify the propagation scheme employed by Zippy, consider the example illustrated in Fig. 9, where initiator A transmits the first bit out of three, assuming a maximum hop count of $k = 2$. Participant B will receive the first sub-bit as a 1, and then transmit during the remaining sub-bit, whereas participant C decodes the first sub-bit as a 0, and therefore refrains from transmission, and instead decodes the second sub-bit as a 1. This described scheme gives participants residing up to $k-1$ hops away from the initiator more than one chance at successfully decoding each bit.

Majority vote decoding, a scheme originally proposed in [37], is performed at each participant by sampling the *DATA* line of its OOK receiver n times per bit period T_b, where n is a positive odd integer. The sub-bit is decoded as a 1 if the *DATA* line was sampled high at least $\lceil \frac{n}{2} \rceil$ times, otherwise, the sub-bit is decoded as 0. Fig. 9 illustrates the timing diagram for majority vote sub-bit decoding, with each sample per sub-bit separated by λT_b, where $\lambda = \frac{1}{n+1}$.

In order for the majority vote sub-bit decoding to function as intended, the sampling of the *DATA* line must not extend past the sub-bit boundary. This is prevented by ensuring the sub-bit period T_b is long enough, given the maximum per-hop synchronization delay $T_{i,i+1}$ experienced throughout the multi-hop network. This is analytically described by selecting the sub-bit period T_b such that the following inequality holds:

$$T_b > \frac{T_{i,i+1}}{\lambda} \quad \forall i \in [0, k-1] \tag{4}$$

It is evident that Zippy's data propagation is inherently biased toward 1 bits, since a 0 bit requires all k sub-bits to be decoded as 0, while a 1 bit requires only one of the k sub-bits to be decoded as 1. While this asymmetry may lead to packet corruption due to erroneous 1 bit propagation, Zippy leverages tight per-hop synchronization for aligning sub-bit boundaries, and majority sub-bit decoding for improved packet reception as evaluated in Sec. 4.4.

An important property of Zippy's data propagation scheme is that the end-to-end latency of a flood at each mote differs only by the small per-hop synchronization delays $T_{i,i+1}$. Specifically, the minimum end-to-end latency $L_{0,h}$ from the beginning of the initiator transmission until the participant at hop count h decodes the N_{data} bit packet, is given by:

$$L_{0,h} \geq T_{preamble} + (k+1)T_x + \sum_{i=0}^{h-1} T_{i,i+1} + (kN_{data}+1)T_b \tag{5}$$

We can therefore conclude that the theoretical minimum end-to-end latency $L_{0,h}$ increases linearly with the maximum hop count k, and with the number of packet bits N_{data}.

4.4 Carrier Frequency Randomization

We have so far only considered the operation of Zippy in a network of motes arranged in a line topology. We now extend its design to more realistic wireless networks where each hop may contain several participating motes.

The main challenge in extending Zippy into a dense network is interference. For example, when two motes simultaneously transmit an OOK-modulated 1 bit, the corresponding sinusoidal signals may constructively or destructively interfere at the antenna of the OOK receiver residing at the next hop. If the interference is constructive, then the receiver will decode the bit successfully without any adverse side effects. However, if the interference is destructive, the received signal level may be too low to be detected.

During a Zippy flood, destructive interference is undesirable as all aforementioned components may be affected.

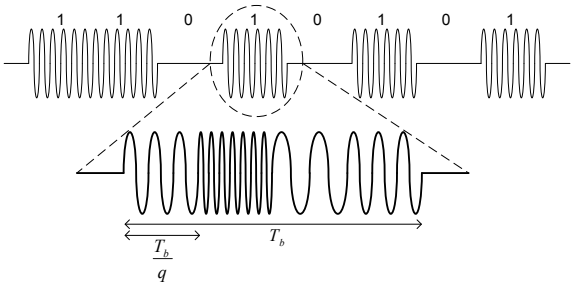

Figure 10: Example OOK transmission incorporating carrier frequency randomization with $q = 4$.

For example, destructive interference may result in a missed preamble reception, erroneous per-hop synchronization due to the late reception of the synchronization bit, or packet corruption due to erroneous decoding of sub-bits.

One solution to this problem is to avoid the interference by employing a collision avoidance mechanism, *e.g.*, Clear Channel Assessment (CCA) with random back-off. However, there are two severe problems that arise, making such mechanisms infeasible for Zippy. Firstly, the use of a random back-off would induce long delays between the relaying of each preamble. This will make it impossible to tightly bound the delay T_x, which is used by the initiator and all participants to correctly sequence the synchronization bit. Secondly, the induced delays of relaying the synchronization bit will drastically extend the per-hop synchronization bounds $T_{i,i+1}$, leading to an excessively long OOK bit period T_b in order to satisfy inequality (4) derived in Sec. 4.3.

Therefore, rather than trying to avoid collisions, Zippy embraces overlapping transmissions, but mitigates their effects by randomizing the carrier frequency used to generate OOK 1 sub-bits. Instead of transmitting each OOK 1 sub-bit using a single carrier frequency of duration T_b, each sub-bit is transmitted using a random sequence of $q > 1$ carrier frequencies, as depicted in Fig. 10. Since each mote randomly selects a different set of carrier frequencies per sub-bit, the probability of destructive interference during the entire sub-bit reduces.

In particular, each mote selects a carrier frequency $f = f_c + \beta\Delta$, at least q times per sub-bit period T_b, where f_c is the center carrier frequency, β is a discrete random variable uniformly distributed over a set of integers of size $M \geq 2$, and Δ is a minimum frequency offset. The parametrization of q, M, and Δ are dependent on the hardware of the OOK receiver, OOK transmitter, and the MCU used to control the two devices. Based on the developed prototype detailed in Sec. 5.1, together with extensive practical experiments, we support carrier frequency randomization with $q = 8$ random frequencies per sub-bit, with $M = 4$ random frequencies to choose from, centered about $f_c = 446.8\,\text{MHz}$, with each having a minimum offset of $\Delta = 135.41\,\text{kHz}$.

We next experimentally evaluate the effectiveness of carrier frequency randomization based on the aforementioned parameterization using the cabled setup as illustrated in Fig. 11(a). Using a 6-way RF power combiner, we combine the simultaneous transmissions of up to six participants, with the resultant signal cabled to an independent participant for decoding using its OOK receiver. An external signal generator is used to trigger the transmission of a randomly generated 32-bit packet from each of the attached participants. All participants share the same random

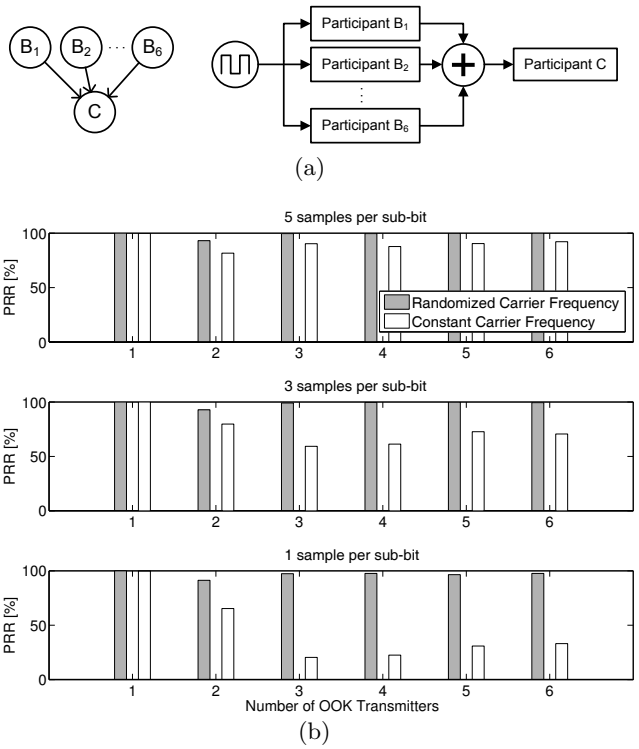

Figure 11: (a) Logical and physical experimental setup, and (b) results of evaluating carrier frequency randomization and majority vote sub-bit decoding.

seed, ensuring the packet is identical during each simultaneous transmission. Each participant randomly selects its own carrier frequency using a unique seed. An embedded CRC-8 within each packet is used to calculate the packet reception rate (PRR), with and without carrier frequency randomization enabled, and with and without majority vote sub-bit decoding. A total of a 1000 packets are generated for each test configuration. The transmission power of all participants is fixed at $-30\,\text{dBm}$. Shielded SMA cables interconnect all RF devices so to remove non-deterministic propagation effects.

The results of the experiment are illustrated in Fig. 11(b). As expected, a 100% PRR is achieved for a single transmitter, irrespective of the carrier frequency and majority decoding configuration. As the number of simultaneous transmitters is increased, the PRR drastically reduces when using a constant carrier frequency. This effect is slightly mitigated by increasing the number of samples per sub-bit from 1 to 5. In the case of only two simultaneous transmitters, we observed a slight reduction of PRR when using a randomized carrier frequency. Since it is a random process, we would expect the occasional bit error due to destructive interference. Nevertheless, the PRR remains higher than that achieved using a constant carrier frequency. In summary, carrier frequency randomization achieves almost 100% PRR with up to six simultaneous transmitters, thus making it a robust mechanism for Zippy to operate in dense networks.

5. EXPERIMENTAL EVALUATION

In this section, we present the design of a custom wireless sensor platform for the evaluation of Zippy. We first present the platform design, evaluate the sensitivity and false wake-up rate of its OOK receiver, and measure its total power

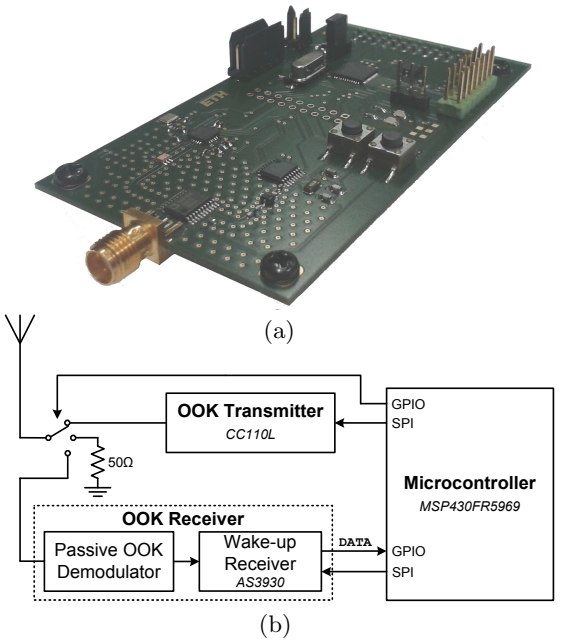

Figure 12: Prototype (a) and block diagram (b) of a wireless sensor platform supporting Zippy flooding.

dissipation. We then evaluate the performance of Zippy in a controlled laboratory setting and in an indoor testbed.

5.1 Prototype Wireless Sensor Platform

A prototype wireless sensor platform supporting Zippy is shown in Fig. 12(a), with its block diagram depicted in Fig. 12(b). The platform consists of a 16-bit MSP430FR5969 MCU interfaced to a CC110L transceiver, and an ultra-low power OOK receiver based on the AS3930 receiver. The antenna of the platform is connected to an ADG904 RF switch, which is controlled by the MCU. The antenna path can be fed into the OOK transmitter, the OOK receiver, or a 50Ω resistor. The components of the platform were selected based on their ultra-low quiescent supply current or ultra-low current drain during deep sleep mode.

OOK Receiver. The receiver is an adaptation of [11], where the impedance matching circuitry has been tuned for the 434 MHz ISM frequency band. The receiver consists of a passive OOK demodulator coupled with a commercially available AS3930 ASK receiver. We leveraged the *WAKE* line from the AS3930 to indicate the reception of a preamble, while the *DATA* line is used for detecting the synchronization bit and decoding packets. The OOK receiver supports a maximum data rate of 8.192 kbps, and features an ultra-low current drain of 2.7 μA measured at 3.0 V.

OOK Transmitter. In order to design a flexible wireless sensor platform for future research opportunities, we incorporated a multi-purpose wireless transceiver instead of a dedicated OOK transmitter module. Carrier frequency randomization is supported on the CC110L transceiver by configuring it for transmission using FSK-4 modulation. An OOK signal is produced by continuously generating random FSK-4 symbols, and using the antenna switch to generate either a 1 or a 0 sub-bit, as required. An OOK 1 sub-bit is produced by switching the antenna port to the CC110L transceiver for a period of T_b, while an OOK 0 sub-bit is produced by switching the antenna port to the 50Ω resistor for a period of T_b. A software-based pseudo-random number generator [26] is used to randomly select each FSK symbol from a uniform distribution, thus producing the desired randomization of the carrier frequency.

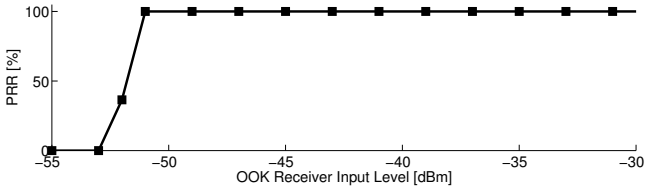

Figure 13: Sensitivity of the OOK receiver as measured experimentally with a variable attenuator.

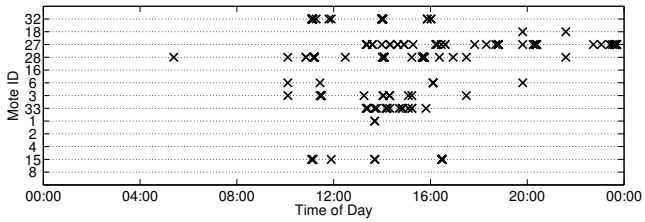

Figure 14: Sporadic false wake-ups observed during a 24 hour experiment on an indoor testbed.

5.1.1 Receiver Sensitivity

The sensitivity of a wireless receiver is defined as the minimum input signal power that supports successful decoding. While the receiver sensitivity has an influence on the overall reception range of the receiver, care must be taken not to infer reception range only from receiver sensitivity, as various techniques can be used to improve the RF link budget, *e.g.*, increasing the power of the transmitted signal and the use of high-gain antennas.

We measured the sensitivity of the OOK receiver using the setup described in Sec. 4.1, and illustrated in Fig. 5. By varying the attenuation between OOK transmitter and receiver, the sensitivity is determined by the minimum input signal level upon which the *DATA* line no longer asserts in the presence of an OOK 1 sub-bit transmission. As shown in Fig. 13, the receiver sensitivity of the OOK receiver is approximately −52 dBm. This sensitivity level is considered state-of-the-art compared to existing prototypes with a similar architectural design [10].

It is important to clarify that despite the apparent low sensitivity compared to high-complexity receivers, practical experiments have demonstrated ranges up to 15 meters non-line-of-sight in an office hallway, and up to 30 meters line-of-sight in an outdoor sports field. These experiments were carried out using quarter-wavelength omnidirectional monopole antennas with a ground plane of 25 cm^2, while being fixed 1.5 m above the ground.

5.1.2 False Wake-up Rate

Due to the low-complexity design of the OOK receiver, it is susceptible to external interference. In particular, we are interested in how often a false wake-up occurs, *i.e.*, when the *DATA* line asserts without an OOK preamble being transmitted. The receiver used in this work operates using subcarrier modulation [29], where a 434 MHz carrier produces the envelope of a 125 kHz carrier. While this design achieves

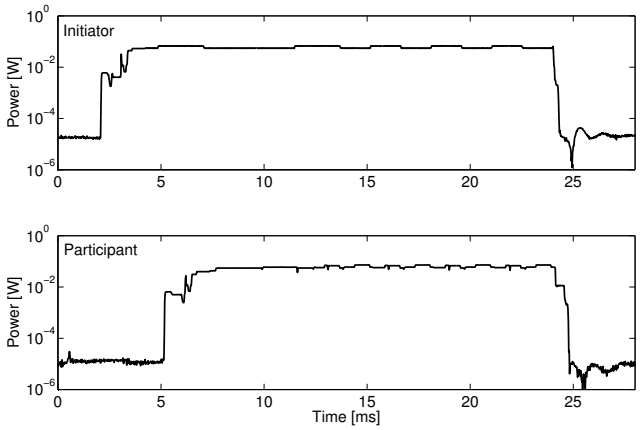

Figure 15: Power profile of initiator and participant motes during a Zippy flood.

ultra-low power dissipation, it is susceptible to strong noise sources operating around the 125 kHz frequency, *e.g.*, from switch-mode power supplies, and devices operating in the 434 MHz ISM band, *e.g.*, other wireless modules.

In order to quantify the false wake-up rate of our prototype, we deployed 13 motes into an indoor office environment, as described in Sec. 5.3. During a period of 24 hours, we recorded each time a false wake-up occurred. The results of the experiment are shown in Fig. 14, where a cross indicates a false wake-up. In summary, four motes did not exhibit any false wake-ups, while mote 28 experienced the maximum of 3166 false wake-ups. With the exception of mote 27, the false wake-ups were predominantly recorded during office hours. The root cause for all the false wake-ups are not yet known, however, it is anticipated that RF shielding and common mode noise rejection circuitry may reduce some false wake-ups.

5.1.3 Active and Idle Power Dissipation

The power profiles of an initiator and a participant mote are depicted in Fig. 15, as measured using an Agilent N6705A DC Power Analyzer. The two platforms were supplied with 3.0 V, while having their on-board low-dropout regulators bypassed. The power profile begins with both motes in an idle listening mode, whereby the MCU resides in a deep sleep mode, *i.e.*, LPM4. During this time, the CC110L transceiver is in standby mode, the antenna switch is active, and the OOK receiver is active. The initiator and participant measured a total idle power dissipation of 9.6μW during periods of inactivity.

Once the initiator is triggered to begin a Zippy flood (parameterized with a maximum of $k = 2$ hops), *i.e.*, at approximately 2 ms into the power profile, the initiator begins to transmit the preamble, followed by the synchronization bit, and an 8-bit packet containing the bit sequence 0x55. The power dissipation during all OOK transmissions is approximately 70 mW, which concurs with the CC110L datasheet.

5.2 Controlled Multi-hop Experiments

We first evaluate Zippy in controlled laboratory experiments before extending the evaluation to an indoor testbed. The motivation for controlled experiments is two-fold, firstly, to provide a baseline timing analysis of the per-hop synchronization achieved by Zippy, and secondly to evaluate

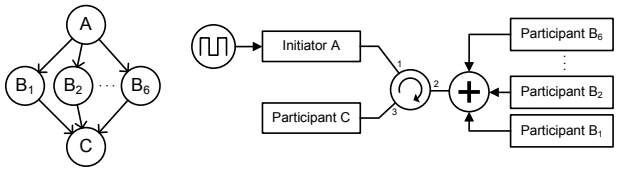

Figure 16: Logical (left) and physical (right) setup for the controlled laboratory experiments.

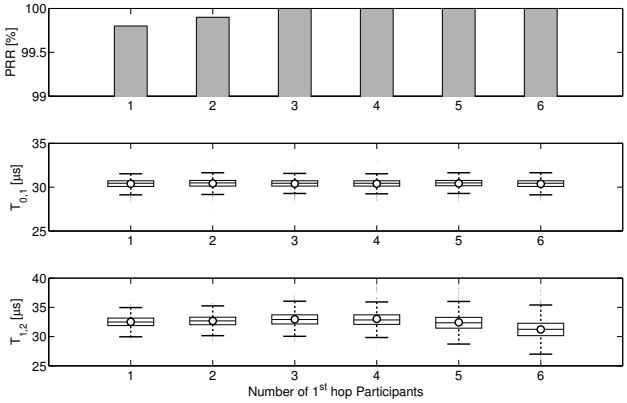

Figure 17: Packet reception rate and per-hop time synchronization of the cabled multi-hop network.

the robustness of Zippy in large, dense networks that would otherwise be infeasible to replicate in a testbed.

Experimental Setup. As illustrated in Fig. 16, a cabled 2-hop network is used to evaluate the timing and robustness of Zippy using a circulator. A circulator is a passive 3-port device that passes an RF signal between ports 1 and 2, but isolates the signal from port 3. Similarly, a signal may pass between port 2 and 3, but is isolated from port 1. Connecting a 6-port RF combiner at port 2 of the circulator enables the construction of sparse and dense 2-hop network topologies. A logic analyzer is connected to initiator A, and participants B_1 and C for the measurement of the per-hop time synchronization and evaluation of packet reception rate.

The initiator A starts the transmission of each Zippy flood according the the frequency of an external square-wave signal generator. Each flood consists of a 32-bit packet containing 24 randomly generated bits and an 8-bit CRC. Zippy is configured with $k = 3$ maximum hops, majority decoding with $n = 3$ samples per bit, and with carrier frequency randomization as described in Sec. 4.4. The transmission power of all motes is fixed at -30 dBm. The experiment begins with a single 1-hop participant, B_1, with all other combiner ports terminated with 50Ω SMA terminators. After the transmission of 1000 Zippy floods by initiator A, an additional 1-hop participant, up to a maximum of six, is connected to the cabled multi-hop network before restarting the experiment.

Results. The results of the controlled laboratory experiment are shown in Fig. 17. The robustness of Zippy is determined by the packet reception rate as measured at participant C. As more 1-hop participants are added, it is vital that the packet reception rate does not decrease. As we can see from Fig. 17, the packet reception rate remains at approximately 100% with up to six first-hop participants. This remarkable behavior is primarily attributed to the use of car-

Table 1: Zippy configuration used throughout all indoor testbed experiments.

Parameter	Description
$k = 2$ and 3 hops	Maximum hop count
$D = \frac{1}{T_b} = 1.364$ kbps	OOK receiver data rate
$N_{data} = 8$ and 16 bits	Packet length
$n = 3$ samples per sub-bit	Majority vote sub-bit decoding
$T_{preamble} = 1.4$ ms	Preamble duration
$T_x = 1.25$ ms	Participant wait delay

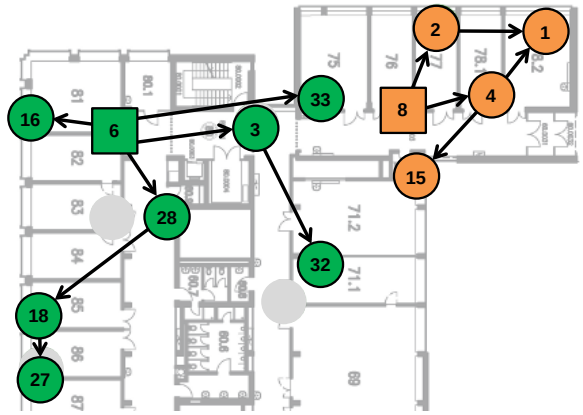

Figure 18: Indoor testbed deployment map, featuring large (left) and small (right) network topologies.

rier frequency randomization. As discussed in Sec. 4.4, the use of majority vote decoding slightly increases the packet reception rate under overlapping transmissions. This experimental result demonstrates an improvement on the packet reception rate achievable during concurrent OOK transmissions compared to using high-complexity receivers as presented in [40](Fig. 6), where simulations show that approximately 10% of all packets are corrupt when there is more than one concurrent transmission.

The per-hop time synchronization is determined by measuring the $T_{i,i+1}$ delay, as introduced in Sec. 4.2. The first-hop synchronization between the initiator A and participant B_1 is given by $T_{0,1}$, while the second-hop synchronization between participants B_1 and C is given by $T_{1,2}$. As shown in Fig. 17, the mean of the first-hop time synchronization is approximately 31 μs, and exhibits a very narrow distribution, irrespective of the number of additional participants. This result is not surprising, since the reception of the synchronization bit at B_1 is not affected by the reception of the synchronization bit by all other participants residing at the same hop. However, the mean of the second-hop time synchronization is slightly increased to approximately 33 μs, and has a wider distribution due to the colliding synchronization bit transmissions. As the number of 1-hop participants increases, the percentiles of the time distribution increase accordingly, however, the mean remains below 35 μs. This demonstrates that, under ideal conditions, Zippy provides fine-grained per-hop time synchronization, while also exhibiting robustness in sparse and dense network topologies due to carrier frequency randomization.

5.3 Testbed Multi-hop Experiments

We next evaluate Zippy's performance in an indoor testbed.

Experimental Setup. We deployed a total of 13 prototype wireless sensor platforms into the FlockLab [21] indoor testbed, which is configured with fine-grained tracing capabilities [22]. The location of each deployed mote is depicted in Fig. 18. We configured the motes into two independent networks, a 2-hop network with a topology having more than one participant at the first hop, and a 3-hop network offering substantial spatial coverage. All motes were configured with carrier frequency randomization, as detailed in Sec. 4.4. Motes 8 and 6 were configured as the initiators for the small and large networks, respectively. All experiments were performed with 8- and 16-bit randomly generated packets. As discussed in Sec. 5.4, support for longer packets with Zippy is indeed feasible.

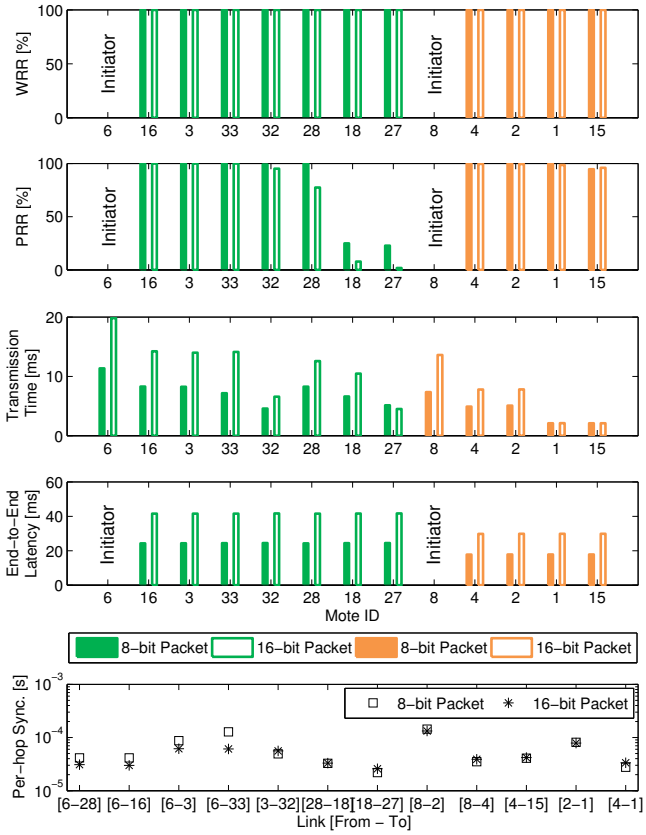

Figure 19: Results from indoor testbed experiments.

The experimental evaluation is based on the following four metrics: *(i) wake-up reception rate (WRR)* is the ratio of the number of motes that wake-up compared to the number of Zippy floods initiated, *(ii) packet reception rate (PRR)* is the ratio of the number of correctly received packets compared to the number of Zippy floods initiated, *(iii) transmission time* is the duration each mote had their OOK transmitter active per Zippy flood, *(iv) end-to-end latency* is the elapsed time between the start of the Zippy flood at the initiator until the end of the flood at each participant, and *(v) per-hop synchronization* is the mean time delay between the motes' nearest neighbor during all Zippy floods. We compute all metrics based on 500 Zippy floods using a static configuration, as listed in Table 1.

Results. The results of the testbed experiments are shown in Fig. 19, and are summarized as follows:

- We first observe a wake-up reception rate (WRR) of 100% for all motes deployed in the testbed. This is a remarkable result, given the challenging RF propagation environment

imposed by the testbed: motes are separated by physical obstacles including thick concrete walls reinforced with steel, and metal piping affixed to the ceiling for plumbing, heating and ventilation. No false wake-ups were observed during the experiments.

- The packet reception rate (PRR) of the motes residing in the large network range from 1.8% (mote 27) up to 100% (motes 16, 3, and 33), while all motes residing in the small network exhibited a PPR exceeding 94.6%. The higher PRR in the small network is attributed to the improved link quality between motes, which exhibit shorter link distances and less physical obstruction compared to the large network topology.
- The transmission times are approximately equal for all motes having the same hop count. This is due to the way in which Zippy propagates each sub-bit of the packet. In addition to the preamble and synchronization bit transmissions, each initiator (motes 6 and 8) transmit all packet bits using a repetition code, as detailed in Sec. 4.3. As the hop count increases, the number of sub-bits to be propagated decreases, resulting in a decrease in transmission time per hop. For example, in the small network, mote 8 initiates the Zippy flood and takes 7.3 ms to transmit the preamble, synchronization bit, and an 8-bit packet, while motes 2 and 4, residing at the first hop, experience a transmission time of approximately 5.0 ms for the preamble, synchronization bit and only half of the packet sub-bits. Finally, the leaf motes, *i.e.*, motes 1 and 15, require a transmission time of only 2.0 ms, as they transmit only the preamble and synchronization bit. A maximum of 19.8 ms was observed by the initiator of the large network for the transmission of a 16-bit packet.
- The end-to-end latency is nearly constant for each network topology and evaluated packet length. Due to the way in which Zippy propagates sub-bits through the network, all participant motes complete a flood at approximately the same time, with small differences attributed to the per-hop time synchronization to their neighbors. The small and large networks exhibit average end-to-end latencies of 17.8 ms and 24.4 ms for the 8-bit packet, and 29.8 ms and 41.6 ms for the 16-bit packet, respectively.
- The mean per-hop synchronization ranges from 21.9 μs between motes 18 and 27, and 143.5 μs between motes 8 and 2. Apart from the link between motes 6 and 33, there are only slight differences between the 8-bit and 16-bit packet experiments. This is to be expected as the per-hop time synchronization is dependent on the reception of the synchronization bit and not on the length of the packet. It is important to highlight that the minimum per-hop synchronization observed is less than what was measured in the cabled experiments in Sec. 5.2. This is indeed feasible, as all links in the testbed are not perfectly isolated. For example, mote 27 may on occasion receive the synchronization bit transmitted from mote 28, while the data propagation is decoded from mote 18.

Comparative Analysis. We next compare the energy efficiency of Zippy in flooding on-demand events compared to state-of-the-art flooding protocols, such as Glossy [8]. We consider a static network topology with a maximum of $k = 3$ hops and with a packet length of 16-bits. As we have shown in the testbed experiments, Zippy can disseminate an event through a 3-hop network with an end-to-end latency of approximately 41.6 ms. If instead Glossy was used to disseminate on-demand events with a reporting latency comparable to Zippy, all motes must perform periodic Glossy floods each separated by at most 41.6 ms. Assuming the Glossy parameterization presented in [45] for the TelosB and CC2420 radio with a maximum number of transmissions $N = 2$, this corresponds to maximum transmission time of 1.9 ms per flood every 41.6 ms. In contrast, the Zippy prototype only needs a maximum transmission time of 19.8 ms once per event. Despite the specific radio-on time depending greatly on the specific mote hardware and the protocol software implementation, we can conclude that Zippy will consume less energy than periodic Glossy floods provided the average event inter-arrival time is greater than approximately 434 ms.

5.4 Protocol Limitations and Challenges

We next discuss limitations and challenges associated with the operation, implementation and evaluation of Zippy.

False Wake-up Detection. As discussed in Sec. 5.1.2, the always-on low-complexity OOK receiver is susceptible to false wake-ups. Zippy is designed such that a preamble is always transmitted once a wake-up is detected. Therefore in the worst-case scenario, if a false wake-up is detected at one mote, all motes in the network will erroneously participate in a Zippy flood. After the preamble has been erroneously relayed, two scenarios are possible, namely, *(i)* a synchronization bit is detected and an erroneous packet is decoded and delivered to a higher-layer application, or *(ii)* a synchronization bit is not detected causing a protocol timeout after an inactivity timer expires. Erroneous packets can be filtered at higher-layers using forward error correction, however, in either case, a false preamble detection results in the misuse of energy reserves, the impact of which depends greatly on the specific parameterization of Zippy.

Erroneous Synchronization. If a participant mote receives the synchronization bit too early due to a false wake-up, or too late due to poor RF propagation, subsequent transmissions may corrupt the reception of all other participants within its 1-hop neighborhood. The quantitative impact of erroneous synchronization has yet to be investigated, however, extensive tests suggest an adequate parameterization of the preamble length $T_{preamble}$ and participant wait delay T_x reduce the occurrence of this erroneous behavior.

Noise Resilience. The performance of modulation schemes in the presence of noise are characterized analytically by their bit error vs. signal to noise ratio (SNR) curve [31]. It is well known that higher-order modulation schemes, such as BPSK, exhibit a lower bit error probability for the same SNR compared to OOK modulation. Based on this analytical result, there are two approaches to improve the operation of Zippy's data propagation with respect to noise, namely, *(i)* increase the SNR at the input of the OOK receiver's decoder, or *(ii)* reduce the bit error by employing a higher-order modulation scheme.

The first approach seeks to increase the SNR of the receiver, thereby reducing the probability of bit error. This is achieved in practice by selecting an advanced OOK receiver incorporating a low noise amplifier, which is directly associated with an increase in power dissipation of the radio hardware. However, if the advanced receiver supports a sufficiently high data rate, Zippy's data propagation can operate with higher energy efficiency.

An alternative approach is to adopt a new modulation scheme that is more robust against noise, *i.e.*, where a lower

SNR is needed for the same probability of bit error. However, the challenge is to manage the adverse effects of simultaneous transmissions using a modulation scheme other than OOK. As introduced in Glossy [8], constructive interference can be leveraged to reliably disseminate packets through simultaneous transmissions using O-QPSK modulation.

Data Rate Reduction. The data rate of the OOK receiver had to be reduced to 1.364 kbps for the implementation of Zippy due to instability of the OOK demodulator output. We suspect that this is due to the dynamic operation of the AS3930's integrated AGC. Since the design of the AGC is not in the public domain, it is difficult to identify the root cause of this behavior. However, in principle, there are no limitations in operating Zippy at higher data rates using an appropriately designed AGC.

Packet Length Support. The packet length supported by Zippy is primarily dependent on the quality of the RF links between motes. As exemplified in Sec. 5.3, motes with poor RF connectivity, *e.g.*, motes 18 and 27 in the large network, will experience higher bit errors, resulting in a lower PRR as the packet length is increased. However, when motes have good RF connectivity, longer packet lengths may be supported with a high PRR. For example, in the small testbed network, Zippy has been shown to support a packet length of 64-bits with an average PRR of 96.7%.

Network Scalability. As shown in the experiments presented in Sec. 5.3, Zippy's end-to-end latency is approximately constant for a given network configuration, and increases linearly with respect to the maximum hop count k, and the number of packet bits N_{data}. Furthermore, the transmission time of the initiator is determined by the maximum hop count k, while the transmission time of participants decreases linearly with respect to its hop count. It is anticipated that the advantageous properties of carrier frequency randomization, as validated in Sec. 4.4, will also apply in large network deployments with appropriate parameterization. The validation of these network scaling properties remain a future research opportunity.

Antenna Ground Plane. The ground plane and placement of the antenna adversely impacted the range of the low-complexity OOK receiver. Since each FlockLab target slot has a defined physical footprint, the size of the antenna ground plane is therefore limited. Furthermore, by affixing the antennas close to thick reinforced concrete walls, as is the case for all indoor FlockLab targets, the antenna radiation pattern of each mote is altered. These two combined effects reduced the operation range of some links within the testbed deployment, necessitating the installation of larger antenna ground planes at selected FlockLab observers.

6. RELATED WORK

Asynchronous network wake-up was first proposed as part of the PTW [42] protocol, where each mote immediately transmits a preamble as soon as it is woken up by the reception of a preamble. The relaying of preambles has been recently extended in FLOOD-WUP [32] by leveraging the addressable wake-up mode supported by modern ultra-low power receivers. Zippy builds on their work by addressing the problem of motes simultaneously transmitting. Leveraging carrier frequency randomization, as introduced in this paper, provides a robust, rapid, and topology-independent asynchronous network wake-up. In addition, Zippy provides tight per-hop synchronization and data dissemination.

Several protocols employing ultra-low power receivers have been proposed in the literature, for example E2RMAC [15], WUR-MAC [25], and RTWAC [1], which provide single-hop packet transfer by leveraging an asynchronous rendezvous. While these protocols achieve improved end-to-end latency and energy efficiency compared to synchronous and pseudo-asynchronous protocols, they do not support asynchronous multi-hop data dissemination as demonstrated by Zippy. Furthermore, the aforementioned protocols have only been evaluated empirically through simulation, in contrast to the extensive indoor testbed deployment of Zippy.

RFID tags for asynchronous rendezvous in wireless sensor networks have been proposed in [16] using active tags, and in [3] using passive WISP [38] platforms. While simulations of RFID-enabled multi-hop communication indicate promising energy efficiencies [17], it is unclear if the power dissipation of the RFID reader, coupled with limited operational range, prohibits the realization of energy-efficient multi-hop data dissemination using RFID tags.

Ambient backscatter [24] is a novel communication paradigm, however, despite recent advancements of these passive receivers [30], the communication range of ambient backscatter devices is constrained by the power of the ambient signal source. Realistic signal powers of ambient sources, *e.g.*, from TV transmissions, are likely to constrain the communication range to only a few meters. In this work, we have demonstrated Zippy operating in an indoor testbed with challenging RF propagation conditions, achieving multi-hop connectivity with per-hop links of up to 10 meters.

Dual-radio wireless sensor platforms incorporating a high-powered transceiver and an ultra-low power receiver have been proposed in [4], [36], [28] and [9]. Although these prototype platforms support asynchronous rendezvous and single-hop data dissemination, they do not support carrier frequency randomization, and have not been experimentally evaluated in a testbed. We have shown experimentally using our prototype wireless sensor platform that carrier frequency randomization is a prerequisite for robust multi-hop data dissemination using low-complexity OOK receivers.

7. CONCLUSIONS

This paper is motivated by the need for rapid and energy efficient dissemination of rare events through multi-hop wireless networks. We present Zippy, a paradigm shift in protocol design, where the fundamental design trade-off between mote life-time and end-to-end latency is decoupled. Zippy supports on-demand multi-hop flooding with end-to-end latencies of tens of milliseconds, while dissipating less than ten microwatts during periods of inactivity. We detail the design, analysis and prototype implementation of Zippy, and extensively evaluate its performance in a laboratory setting and in an indoor testbed.

Acknowledgements. We thank Marco Zimmerling, our shepherd Kamin Whitehouse, and the anonymous reviewers for their valuable feedback, Roman Lim and Balz Maag for their FlockLab feature extensions and support, and Reto Da Forno for his power optimizations on the Zippy prototype. This work was scientifically evaluated by the SNSF and financed by the Swiss Confederation and by Nano-Tera.ch.

8. REFERENCES

[1] J. Ansari, D. Pankin, and P. Mähönen. Radio-triggered wake-ups with addressing capabilities for extremely low power sensor network applications. *International Journal of Wireless Information Networks*, 2009.

[2] D. Ash. A comparison between OOK/ASK and FSK modulation techniques for radio links. Technical report, RF Monolithics Inc, 1992.

[3] H. Ba, I. Demirkol, and W. Heinzelman. Feasibility and benefits of passive RFID wake-up radios for wireless sensor networks. In *Global Telecommunications Conference (GLOBECOM)*. IEEE, 2010.

[4] J. Brown, J. Finney, C. Efstratiou, B. Green, N. Davies, M. Lowton, and G. Kortuem. Network interrupts: supporting delay sensitive applications in low power wireless control networks. In *Proceedings of the 2nd ACM Workshop on Challenged Networks*. ACM, 2007.

[5] I. Demirkol, C. Ersoy, and F. Alagoz. MAC protocols for wireless sensor networks: a survey. *Communications Magazine, IEEE*, 2006.

[6] I. Demirkol, C. Ersoy, and E. Onur. Wake-up receivers for wireless sensor networks: benefits and challenges. *Wireless Communications, IEEE*, 2009.

[7] P. Dutta, M. Grimmer, A. Arora, S. Bibyk, and D. Culler. Design of a wireless sensor network platform for detecting rare, random, and ephemeral events. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN)*. IEEE, 2005.

[8] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh. Efficient network flooding and time synchronization with glossy. In *Proceedings of the 10th International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 2011.

[9] G. Gamm, S. Stoecklin, and L. Reindl. Wake-up receiver operating at 433 MHz. In *Proceedings of the 11th International Multi-Conference on Systems, Signals and Devices (SSD)*, 2014.

[10] G. U. Gamm and L. M. Reindl. Smart metering using distributed wake-up receivers. In *Proceedings of the IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*. IEEE, 2012.

[11] G. U. Gamm, M. Sippel, M. Kostic, and L. M. Reindl. Low power wake-up receiver for wireless sensor nodes. In *Proceedings of the 6th International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*. IEEE, 2010.

[12] L. Girard, J. Beutel, S. Gruber, J. Hunziker, R. Lim, and S. Weber. A custom acoustic emission monitoring system for harsh environments: application to freezing-induced damage in alpine rock-walls. *Geoscientific Instrumentation, Methods and Data Systems Discussions*, 2012.

[13] W. Granzer, C. Reinisch, and W. Kastner. Future challenges for building automation: wireless and security. In *Proceedings of the IEEE International Symposium on Industrial Electronics (ISIE)*, 2010.

[14] C. Guo, L. C. Zhong, and J. M. Rabaey. Low power distributed MAC for ad hoc sensor radio networks. In *IEEE Global Telecommunications Conference*. IEEE, 2001.

[15] V. Jain, R. Biswas, and D. P. Agrawal. Energy-efficient and reliable medium access in sensor networks. In *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. IEEE, 2007.

[16] R. Jurdak, A. Ruzzelli, and G. O'Hare. Radio sleep mode optimization in wireless sensor networks. *IEEE Transactions on Mobile Computing*, 2010.

[17] R. Jurdak, A. G. Ruzzelli, and G. M. O'Hare. Multi-hop RFID wake-up radio: design, evaluation and energy tradeoffs. In *Proceedings of the 17th International Conference on Computer, Communications and Networks (ICCCN)*. IEEE, 2008.

[18] G. Kim, Y. Lee, S. Bang, I. Lee, Y. Kim, D. Sylvester, and D. Blaauw. A 695pW standby power optical wake-up receiver for wireless sensor nodes. In *Custom Integrated Circuits Conference (CICC)*. IEEE, 2012.

[19] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon. Health monitoring of civil infrastructures using wireless sensor networks. In *Proceedings of the 6th International Symposium on Information Processing in Sensor Networks (IPSN)*. IEEE, 2007.

[20] K. Leentvaar and J. Flint. The capture effect in FM receivers. *IEEE Transactions on Communications*, 1976.

[21] R. Lim, F. Ferrari, M. Zimmerling, C. Walser, P. Sommer, and J. Beutel. FlockLab: A testbed for distributed, synchronized tracing and profiling of wireless embedded systems. In *Proceedings of the 12th International Conference on Information Processing in Sensor Networks (IPSN)*. ACM, 2013.

[22] R. Lim, B. Maag, B. Dissler, J. Beutel, and L. Thiele. A testbed for fine-grained tracing of time sensitive behavior in wireless sensor networks. In *Local Computer Networks Workshops*, 2015.

[23] E.-Y. Lin, J. M. Rabaey, and A. Wolisz. Power-efficient rendez-vous schemes for dense wireless sensor networks. In *International Conference on Communications*. IEEE, 2004.

[24] V. Liu, A. Parks, V. Talla, S. Gollakota, D. Wetherall, and J. R. Smith. Ambient backscatter: wireless communication out of thin air. In *Proceedings of the ACM SIGCOMM Conference*. ACM, 2013.

[25] S. Mahlknecht and M. S. Durante. WUR-MAC: energy efficient wakeup receiver based MAC protocol. In *Proceedings of the 8th IFAC FET*, 2009.

[26] G. Marsaglia. The mother of all random generators, 1994.

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

[28] J. Oller, I. Demirkol, J. Casademont, and J. Paradells. Design, development, and performance evaluation of a low-cost, low-power wake-up radio system for wireless sensor networks. *ACM Transactions on Sensor Networks*, 2013.

[29] J. Oller, I. Demirkol, J. Casademont, J. Paradells, G. U. Gamm, and L. Reindl. Performance evaluation and comparative analysis of subcarrier modulation wake-up radio systems for energy-efficient wireless sensor networks. *Sensors*, 2013.

[30] A. N. Parks, A. Liu, S. Gollakota, and J. R. Smith. Turbocharging ambient backscatter communication. In *Proceedings of the 2014 ACM SIGCOMM Conference*. ACM, 2014.

[31] P. Z. Peebles Jr. *Digital communication systems*. Prentice-Hall, 1987.

[32] C. Petrioli, D. Spenza, P. Tommasino, and A. Trifiletti. A novel wake-up receiver with addressing capability for wireless sensor nodes. In *Proceedings of the International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2014.

[33] N. Pletcher, S. Gambini, and J. Rabaey. A 65 μW, 1.9 GHz RF to digital baseband wakeup receiver for wireless sensor nodes. In *Proceedings of the Custom Integrated Circuits Conference (CICC)*. IEEE, 2007.

[34] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys)*. ACM, 2004.

[35] C. V. Pollack Jr. Wireless cardiac event alert monitoring is feasible and effective in the emergency department and adjacent waiting areas. *Critical Pathways in Cardiology*, 2009.

[36] T. Prabhakar, N. Soumya, P. Muralidharan, and H. Jamadagni. A Novel Wake-Up Radio WSN Mote. In *Proceedings of the Texas Instruments India Educators' Conference (TIIEC)*, 2013.

[37] I. Reed. A class of multiple-error-correcting codes and the decoding scheme. *Transactions of the IRE Professional Group on Information Theory*, 1954.

[38] A. Sample, D. Yeager, P. Powledge, A. Mamishev, and J. Smith. Design of an RFID-based battery-free programmable sensing platform. *IEEE Transactions on Instrumentation and Measurement*, 2008.

[39] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava. Optimizing sensor networks in the energy-latency-density design space. *IEEE Transactions on Mobile Computing*, 2002.

[40] M. Wilhelm, V. Lenders, and J. Schmitt. On the reception of concurrent transmissions in wireless sensor networks. *IEEE Transactions on Wireless Communications*, 2014.

[41] K. Yadav, I. Kymissis, and P. R. Kinget. A 4.4μW wake-up receiver using ultrasound data communications. In *VLSI Circuits (VLSIC), 2011 Symposium on*. IEEE, 2011.

[42] X. Yang and N. Vaidya. A wakeup scheme for sensor networks: Achieving balance between energy saving and end-to-end delay. In *Proceedings of the 10th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2004.

[43] P. Zappi, E. Farella, and L. Benini. Tracking motion direction and distance with pyroelectric IR sensors. *IEEE Sensors Journal*, 2010.

[44] P. Zhang, P. Hu, V. Pasikanti, and D. Ganesan. Ekhonet: high speed ultra low-power backscatter for next generation sensors. In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking (MobiCom)*. ACM, 2014.

[45] M. Zimmerling, P. Kumar, F. Ferrari, L. Mottola, and L. Thiele. Energy-efficient real-time communication in multi-hop low-power wireless networks. Technical report, TIK Report No. 356, 2015.