Cooperative Carrier Signaling: Harmonizing Coexisting WPAN and WLAN Devices

Xinyu Zhang and Kang G. Shin
Department of Electrical Engineering and Computer Science
University of Michigan
{xyzhang, kgshin}@eecs.umich.edu

***Abstract*—The unlicensed ISM spectrum is getting crowded by WLAN and WPAN users and devices. Spectrum sharing within the *same* network of devices can be arbitrated by existing MAC protocols, but the coexistence between WPAN and WLAN (*e.g.*, ZigBee and WiFi) remains a challenging problem. The traditional MAC protocols are ineffective in dealing with the disparate transmit-power levels, asynchronous time slots, and incompatible PHY layers of such heterogeneous networks. Recent measurement studies have shown moderate-to-high WiFi traffic to severely impair the performance of coexisting ZigBee.**

We propose a novel mechanism, called *cooperative carrier signaling* (CCS), that exploits the inherent cooperation among ZigBee nodes to harmonize their coexistence with WiFi WLANs. CCS employs a separate ZigBee node to emit a carrier signal (busy-tone) concurrently with the desired ZigBee's data transmission, thereby enhancing the ZigBee's visibility to WiFi. It employs an innovative way to concurrently schedule a busy tone and a data transmission without causing interference between them. We have implemented and evaluated CCS on the TinyOS/MICAz and GNURadio/USRP platforms. Our extensive experimental evaluation has shown that CCS reduces collision between ZiBee and WiFi by 50% for most cases, and by up to 90% in the presence of a high-level interference, all at negligible WiFi performance loss.

I. Introduction

The past decade has witnessed the proliferation of wireless MAC/PHY standards for establishing wireless local area networks (WLANs) and personal area networks (WPANs) on the ISM band. Allowing spectrum sharing among these networks will undoubtedly improve spectrum utilization. However, it also creates unprecedented challenges, especially the coexistence of incompatible MAC/PHY protocols. Two such networks, WiFi (IEEE 802.11 WLAN) and ZigBee (IEEE 802.15.4 WPAN), that operate in the 2.4GHz license-exempt band have received considerable attention. WiFi is designed for Internet access, video streaming, *etc.*, whereas ZigBee targets low duty-cycle monitoring and control applications such as health care and home/industrial automation. They are expected to run simultaneously in close proximity, *e.g.*, inside a residential or office or hospital building. However, recent measurement studies have shown that ZigBee's performance is severely degraded in the presence of moderate to high WiFi traffic. For example, in an enterprise WLAN and a co-located 90-node WPAN for building energy management, more than a half of the ZigBee links in the WPAN were observed to suffer connection loss during peak hours due to WiFi interference [1]. The harmful coexistence has also been observed in previous small- to medium-scale experimental deployments [2]–[6].

A straightforward way to avoid such interference is to allocate ZigBee devices to channels that are not or less used by WiFi devices [7], [8]. However, the prevailing frequency planning methods presume ZigBee to fall in the same management domain as WiFi, without resolution of the short-term collisions caused by unmanaged, bursty WiFi traffic. Moreover, there are only a limited number of orthogonal WiFi channels on the ISM band, and thus frequency separation is difficult to achieve in a densely-deployed WiFi environment [1], [9], where co-located APs tend to occupy different parts of the spectrum to avoid inter-cell interference. Therefore, it is important to devise a complementary approach that enables ZigBee to share the same frequency band with WiFi, but multiplex the channel over time.

The CSMA-style spectrum etiquette in such networks may seem to be an effective means to achieve this. However, system heterogeneity poses a serious challenge and may severely degrade CSMA-based coexistence schemes. Through link-level measurement of coexisting WPAN and WLAN, we find that the legacy ZigBee MAC experiences a 51% collision rate even when WiFi leaves the channel unused for 67% of the time. A further microscopic study has revealed the root cause of such harmful coexistence. First, ZigBee's transmit power is 20dB lower than WiFi's, yielding a smaller spatial footprint and hence, its poor visibility to WiFi. The ZigBee's MAC-layer time resolution is also 16 times coarser, and it can easily be preempted by WiFi in the middle of a rx/tx transition (*e.g.*, sensing-to-transmission or data-to-ACK transition), thus causing collision, even if they can sense each other. In addition, ZigBee allows for TDMA mode, which operates without carrier sensing, and may arbitrarily collide with an ongoing WiFi transmission. The disparate transmit power, time resolution and scheduling mode also make it difficult to deal with other coexisting networks, such as WiFi & Bluetooth [10], and WiFi & WiMax [11]. Therefore, by resolving the coexistence between ZigBee and WiFi, one could potentially extend the solution to other heterogeneous networks.

Based on the above observations, we propose a new mechanism, called *Cooperative Carrier Signaling* (CCS), to facilitate ZigBee's coexistence with WiFi. CCS builds atop the ZigBee MAC/PHY, but enhances it with a new coexistence management framework, making WiFi better aware of ZigBee's presence, and hence achieving better channel sharing. Unlike the traditional CSMA that relies on a data packet as an implicit carrier signal (busy-tone), CCS assigns a separate ZigBee node called *signaler*, as a proxy to perform the carrier signaling. The

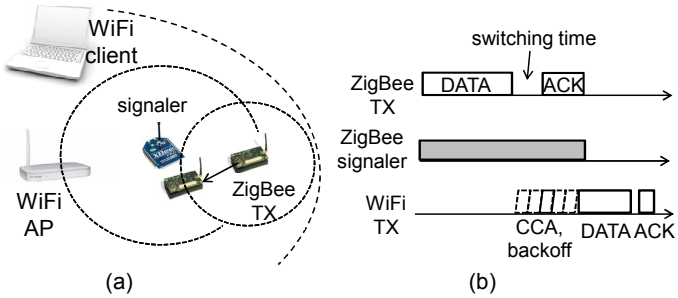

Fig. 1. The principle behind CCS: (a) spatial domain: allowing WiFi to indirectly sense weak ZigBee signals. (b) temporal domain: avoiding WiFi preempting in the rx/tx switching time of ZigBee.

signaler may have a higher power than the ZigBee transmitters, thus allowing the WiFi nodes to sense the ZigBee transmitter's presence indirectly by detecting the busy-tone (Fig. 1(a)). The busy-tone persists throughout the data and ACK roundtrip, thus preventing the WiFi's preemption in the rx/tx switching gap (Fig. 1(b)).

A key challenge to CCS is that the signaler's busy-tone must occur concurrently with the data transmission (without interrupting it). To overcome this difficulty, we design a *temporary channel-hopping* mechanism that separates the carrier signaling from the data transmission in frequency domain, yet ensures WiFi to sense the presence of ZigBee transmission. Furthermore, CCS must schedule the busy-tone to protect both the TDMA and CSMA packets of ZigBee, without adversely affecting WiFi's performance. We extend the ZigBee's built-in beacon synchronization mechanism to be a scheduler that synchronizes the signaler and the transmitter. The scheduler preserves the carrier sensing capability on the signaler, so that the busy-tone is emitted only when the channel is unoccupied by WiFi.

The cooperative signaling mechanism is triggered when WiFi interference is present and causes severe collisions. This adaptation is facilitated by a 2-dimensional carrier sensing scheme that estimates the WiFi's interference intensity and distinguishes it from ZigBee. We further introduce a simple *signaler configuration* scheme that configures the power and location of the signaler, so that its busy-tone may be sensed by multiple randomly located WiFi interferers. With signal configuration, CCS is applicable even when the WPAN nodes and nearby WiFi interferers are mobile.

We have implemented the CCS framework in TinyOS 2.0, which runs on a ZigBee-based WPAN. We also implemented a dedicated signaler on the GNURadio [12] software radio platform. Extensive experiments on the MICAz motes [13] and DC-powered USRP2 [14] have shown that under moderate to high WiFi traffic, CCS reduces the ZigBee's packet collision by more than 50% in most cases. This translates to significant reductions of packet delay and energy consumption. More importantly, when the WPAN runs low duty-cycle traffic, CCS does not degrade the performance of WiFi, compared to the legacy ZigBee. Since CCS does not require any hardware or firmware modification, it can be easily integrated into the ZigBee network stack.

The main contributions of this paper are three-fold.

- A microscopic analysis of coexisting WiFi and ZigBee signals via software radios, which identifies key factors that account for collision between them.
- A new cooperative framework, CCS, that allows ZigBee WPAN to avoid WiFi-caused collisions. CCS overcomes the inherent limitations of traditional CSMA, and can also be extended to enhance the coexistence of other heterogeneous networks.
- Implementation and evaluation of the CCS framework on ZigBee motes and a USRP2-based software radio.

The remainder of this paper is organized as follows. In Sec. II, we review the related work on ZigBee–WiFi coexistence, and introduce the differences of these two protocols. Sec. III analyzes the key factors causing the conflicting coexistence based on fine-grained measurements. Secs. IV and V detail the design and implementation of CCS to solve the coexistence problem. Sec. VI presents our testbed-based evaluation of CCS, while Sec. VII concludes the paper.

II. Background

A. Related work

1) Conflicting coexistence of ZigBee with WiFi: The interference between WiFi and ZigBee has been extensively studied in both the industry and the research communities. Under light WiFi traffic, ZigBee is known to suffer less from collision with WiFi and can recover loss via retransmission [15], [16]. However, under moderate to high WiFi traffic, ZigBee performance is severely degraded. In an indoor testbed with randomly-deployed nodes, Gummadi *et al.* [2] reported that ZigBee experiences a median packet-loss rate of 20%, and the loss could exceed 85% due to WiFi interference. This occurs even when the carrier sensing and packet retransmission are enabled. Pollin *et al.* [3] found that WiFi may interrupt ZigBee transmissions even when they are located close to each other. Similar results have been observed in other measurement studies and real-world applications [1], [4], [5]. With the proliferation of WiFi devices (*e.g.*, smartphones) and high-rate applications (*e.g.*, HD-videos), the amount of WiFi traffic in a typical home or enterprise environment will keep increasing, thus severely affecting the reliability of ZigBee WPANs for monitoring and control applications.

On the other hand, ZigBee seldom interferes with WiFi since it targets low duty-cycle applications with low channel occupancy (typically below 1% [17]). Moreover, WiFi has much higher transmit power, which easily forces ZigBee nodes to back off [2], [4], and can dominate the ZigBee interference. The testbed measurement in [2] has shown that WiFi experiences no packet losses when co-located with multiple ZigBee devices, except its TCP latency increases by about 5%. Controlled experiments reveal that ZigBee may also disturb the 802.11 packet decoding, but only with saturated traffic [3], or without MAC-layer sensing [10].

2) Alternative coexistence mechanisms.: A straightforward way of enabling ZigBee–WiFi coexistence is frequency planning. However, this approach is ineffective when WiFi WLANs are (i) unmanaged and may change channels unpredictably, or (ii) densely deployed, since 3 non-overlapping

channels (2 for the 802.11n 40MHz mode) can occupy the majority of 2.4GHz ISM bands. Strict frequency separation may also under-utilize bandwidth, a well-known problem already present in the TV spectrum. When WiFi traffic becomes intensive, ZigBee may adaptively switch to other idle channels [8]. However, this approach does not resolve bursty collisions—it responds only after collision had already occurred. Adaptive channel allocation also incurs long "blackout time" due to scanning and re-association [18], which can be on the order of several seconds and increases with the network size.

An alternative approach, called WISE [6], changes ZigBee frame size adaptively according to the estimation of the idle interval between WiFi transmissions. WISE needs to suspend ZigBee transmissions in each WiFi burst, and is unsuitable for TDMA packets or delay-sensitive applications. Liang *et al.* [1] proposed use of error-correcting code to recover partially-corrupted ZigBee packets due to WiFi interference. The mechanism is corrective in nature, whereas CCS is a preventive approach that avoids collisions.

In [2], a spectrum-survey-based method is introduced to improve WPAN–WLAN coexistence, by adjusting transmit power and carrier sensing threshold. This approach is suitable for static networks, and aims for long-term throughput fairness between heterogeneous networks. However, short-term performance metrics (*e.g.*, delay and collision rate) are equally important to the monitoring and control applications typically seen on ZigBee networks.

Rahul *et al.* [19] proposed an interference-nulling approach that forces wideband devices to allocate idle spectrum for narrowband devices. This approach requires hardware modification to perform customized signal processing. Moreover, the devices must be able to sense each other so as to determine which part of the spectrum should be nulled.

In contrast to the above approaches, CCS attempts to overcome the inherent limitations of traditional CSMA, and enable it to coordinate heterogeneous networks. Our key observation is that a sufficient idle channel time exists and can be exploited by ZigBee, but the WiFi's unawareness causes severe collisions. By enhancing the visibility of ZigBee to WiFi while preserving the CSMA-based spectrum etiquette, CCS can substantially improve ZigBee's channel utilization without compromising WiFi's performance.

Busy-tone-based signaling mechanism was first adopted by Tobagi and Kleinrock [20], as a solution to the hidden terminal problem in CSMA protocols. Many subsequent proposals extended the mechanism to enhance the performance of CSMA for ad-hoc and sensor networks [21], [22]. These protocols usually require two radios on each transceiver: one for data transmission, and the other emitting a dedicated narrowband signal as the busy-tone. In contrast, CCS adopts a cooperative busy-tone mechanism, which can be realized using off-the-shelf ZigBee devices. The objective of CCS differs from previous busy-tone mechanisms in that it enables the coexistence between *heterogeneous* MAC protocols.

We also proposed a protocol, called *Cooperative Busy-Tone* (CBT) [23], that adopts a mechanism similar to CCS, *i.e.*, a separate signaler is employed to make WiFi aware of ZigBee transmissions, thus reducing collision between ZigBee and WiFi. However, that work focused mainly on establishing a theoretical framework to analyze the performance of CBT and using simulation to validate the analysis. In this paper, we focus on detailed system design, testbed implementation and evaluation of the CCS mechanism. We perform detailed experiments to investigate the cause and effect of the conflicting coexistence between ZigBee and WiFi, and further redesign the signaler's functionalities (*e.g.*, CSMA scheduler, signaler configuration and signal classifier) to facilitate practical deployment.

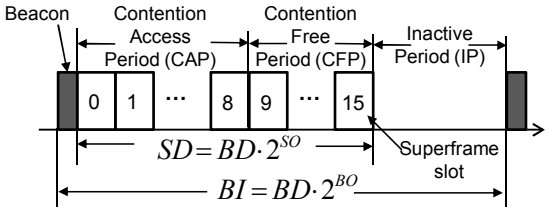

Fig. 2. A superframe in ZigBee. The beacon interval (BI) and superframe duration (SD) depend on beacon order (BO) and superframe order (SO), such that $0 \leq SO \leq BO \leq 14$. The BD value equals $15.36ms$.

B. ZigBee vs. WiFi: MAC/PHY layers

ZigBee specifies the MAC/PHY functionalities to establish low-power, low-rate WPANs. Each ZigBee WPAN assigns a unique *coordinator* to perform association control and beacon scheduling for *clients*. The coordinator schedules a mixture of TDMA and CSMA frames periodically. Each scheduling period is called a *superframe*, which starts with a beacon, followed by a number of CSMA slots (called *CAP*) and TDMA slots (called *CFP* or *Guaranteed Time Slot* (GTS)), and then an inactive period, as illustrated in Fig. 2.

The CFP slots are allocated to clients and deallocated on demand. In the CAP slots, ZigBee enforces slotted-CSMA access control, which differs from WiFi as follows. First, contention access must start from the boundaries of basic time units called *backoff slots*, each lasting 320 μs. In contrast, the WiFi backoff slot is only 20 μs (802.11b) or 9 μs (802.11a/g/n). Second, when sensing a busy channel, ZigBee resumes its backoff and CCA (clear channel assessment), and aborts after 5 consecutive attempts. In contrast, WiFi persists in backoff and sensing until it finds an idle slot for transmission. Third, each backoff in ZigBee consists of two contention windows, *i.e.*, a transmitter must ensure an idle channel for two slots (640 μs) before sending data, whereas WiFi only needs only one (20 or 9 μs) idle slot.

On the PHY layer, ZigBee's bit rate is limited to 250Kbps. Its CCA operation takes 128 μs, and the rx/tx switching time can be 192 μs, due to the hardware limitation. Both the long backoff time at the MAC layer and slow response at the PHY render ZigBee low priority when WiFi transmitters attempt to access the channel. Since the WiFi CCA duration is much shorter, a WiFi transmitter can easily preempt ZigBee in the middle of a ZigBee's rx/tx switching slot (thus causing collision), even when both can perfectly sense each other's presence. Besides, the maximum transmit power of ZigBee is only 0dBm, whereas WiFi typically transmits at 15dBm to 20dBm. Hence, ZigBee has a much shorter interference range

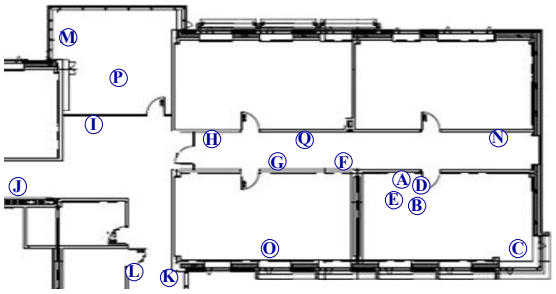

Fig. 3. An experimental testbed with WiFi and ZigBee nodes. By default, the WiFi link is $A \rightarrow C$. ZigBee location is varied in different experiments.

than WiFi [10]. Therefore, ZigBee may not be effectively sensed by WiFi when co-located.

It should be noted that ZigBee also produces special devices on the 868/915MHz band, but the rate (20/40Kbps) may not be suited for all applications. Some WiFi standards (*e.g.*, 802.11n) can operate over the 5GHz band that is orthogonal to ZigBee. However, for compatibility and extended range, 802.11n tends to be configured to the 2.4GHz band where the dominating 802.11b/g reside.

III. Cause and Effect of Conflicting Coexistence

In this section, based on extensive testbed experiments, we anatomize the inherent deficiency of the collision avoidance mechanism in WiFi and ZigBee, which accounts for their conflicting coexistence.

Our testbed includes both WiFi and ZigBee nodes, deployed in an office environment. The floor plan and measurement points are shown in Fig. 3. The WiFi transceivers adopt Atheros 5413 NIC with the MadWiFi v0.9.4 driver, default transmit power 15dBm. The ZigBee nodes are MICAz motes programmed with openzb [24], an open-source implementation of the ZigBee MAC/PHY. The motes uses default transmit power -5dBm. To isolate the ambient interference, the WiFi link is tuned to a channel least used by nearby WLANs, and all measurements are made in the night.

We found the ZigBee link loss rate strongly depends on WiFi's airtime usage Γ_w, defined as the fraction of time spent on transmission (approximately equal to the ratio of application-layer traffic rate to PHY-layer data rate). Among 8 randomly-selected transmitter/receiver pairs running ZigBee's TDMA mode, the median collision probability ranges from 9% when $\Gamma_w = 5.6\%$ and to 51% when $\Gamma_w = 33\%$. The CSMA mode performs only slightly better than TDMA. This link-level experiment implies that ZigBee's performance is severely degraded even though WiFi leaves sufficient idle time. The result is consistent with existing measurement studies [1]–[6]. More detailed results are omitted to avoid repetition. In what follows, we focus on more fine-grained experiments that reveal the root effects behind the conflicting coexistence, which are also the motivation behind the CCS principle.

A. Spatial collision hazards

1) Asymmetric interference: Due to their disparate transmit power levels, there exists a "gray region" where ZigBee can hear WiFi, but WiFi is oblivious of ZigBee and can arbitrarily interrupt its transmission, so called *asymmetric interference*. To profile this effect, we measure the region within the testbed where WiFi's signal strength exceeds ZigBee's carrier sensing threshold, but not vice versa. We use an interference classification scheme (which will be detailed in Sec. IV-D) to measure the WiFi signal power received by a MICAz mote. To verify if WiFi can sense ZigBee, we calibrate the power level of the USRP2 software radio [14] to that of MICAz[1], and allow it to send a continuous stream of ZigBee packets. When WiFi can sense the USRP2 transmitter, it keeps deferring the transmission, thus increasing the packet delay dramatically.

txpower	Nodes under asymmetric interference
0dBm	L, I
-5dBm	I, L, P
-10dBm	I, L, P, H, G, N, O
-15dBm	I, L, P, H, G, N, O, F, C
-25dBm	I, L, P, H, G, N, O, F, C

TABLE I
The asymmetric interference region in the testbed.

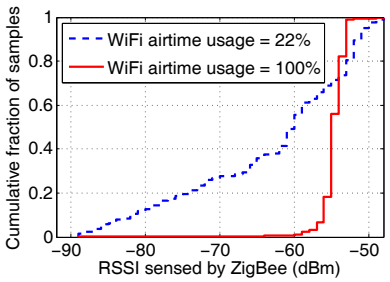

Fig. 4. Distribution of WiFi signal strength sensed by ZigBee.

Table I shows the set of ZigBee nodes vulnerable to asymmetric interference at various transmit power levels. At 0dBm, only nodes far away from the interferer experiences asymmetric interference. As their transmit-power level decreases, the number of vulnerable nodes increases. Below -10dBm, a majority of nodes suffer. This is because ZigBee transmitters implicitly use the data packet as a carrier signal (busy-tone), expecting it can reach the WiFi interferer. However, such carrier signaling becomes less effective as transmit power decreases, and when ZigBee is located farther away from the WiFi transmitter.

To combat asymmetric interference, a natural solution is to employ a proxy signaler with higher transmit power to send the busy tone. Whenever a node within the ZigBee WPAN starts transmission, the signaler would initiate the busy-tone simultaneously, so as to prevent WiFi interruption. This idea constitutes a key intuition behind CCS.

2) Hidden terminal: This problem has been studied extensively for 802.11, and remains a cause for spatial collision hazards in heterogeneous networks. It occurs when the WiFi and ZigBee transmitters cannot hear each other. This is the case for the location K and M, when the WiFi transmitter is

[1]Calibration is needed because the USRP2's output power level is unknown. For calibration, we place the MICAz and USRP2 transmitters near each other, and allow them to send ZigBee packets alternately to the same MICAz receiver. We tune the power level of the MICA transmitter and the PHY parameters of USRP2 (including signal amplitude and transmit gain), so that the MICAz receiver reads the same RSSI values when receiving from both of them. This way, we map the parameter configuration of the USRP2 transmitter to the power level of the MICAz transmitter (in dBm scale).

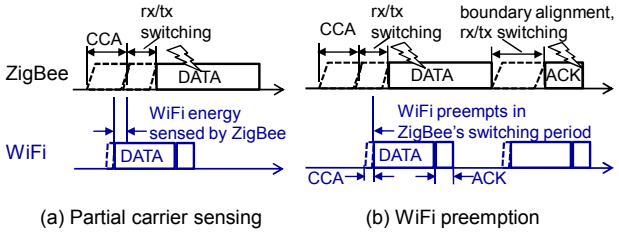

Fig. 5. Collision hazards in the temporal domain.

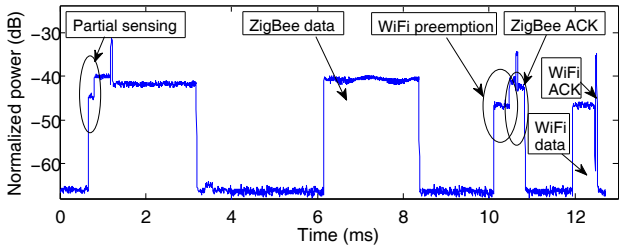

Fig. 6. Temporal collision hazards captured in the traces of a GNURadio/USRP2 oscilloscope.

at A, according to our measurement. Similar to the asymmetric interference, the hidden terminal effect can be alleviated using a proxy signaler visible to the WiFi transmitter.

B. Temporal collision hazards

1) Partial carrier sensing: In addition, collision hazards can occur in the time domain when a WiFi packet is partially sensed by ZigBee and is insufficient to trigger its backoff.

Intuitively, the RSS is independent of the transmitter's duty-cycle. However, through detailed measurement, we found that the WiFi signal strength sensed by a ZigBee receiver is stable when $\Gamma_w = 100\%$[2], but it varies from -50dBm down to the noise floor when $\Gamma_w = 22\%$ (Fig. 4). This implies that certain WiFi packets are partially sensed during the long sensing period of ZigBee, which occurs when WiFi starts transmission near the end of the ZigBee sensing duration, as illustrated in Fig. 5(a). Since the WiFi's energy level needs to be averaged over ZigBee's CCA duration (128 μs) to calculate the RSS, the resulting RSS is insufficient to trigger a ZigBee backoff, exposing it to harmful WiFi interference.

2) Non-CSMA transmission and WiFi preemption: Intuitively, the spatial collision hazards and partial carrier sensing should be less severe when WiFi and ZigBee transmitters are close to each other and RSS exceeds the CCA threshold. However, according to our measurement, even when they are $1m$ apart (*e.g.*, WiFi $A \rightarrow C$ and ZigBee $D \rightarrow E$), the collision rate can still exceed 30%. To capture the root cause, we use a software oscilloscope in GNURadio to log the sample-level traces of the channel (with 20 MHz sampling rate), and then calculate the power of the received signal. Fig. 6 illustrates a snapshot of traces when the above two links are transmitting using CSMA.

[2]We calibrate the USRP2 transmit power to that of WiFi, and use it to send a continuous stream of WiFi packets. The WiFi packets are generated using the BBN 802.11b module in GNURadio, which are compatible with legacy 802.11 packets.

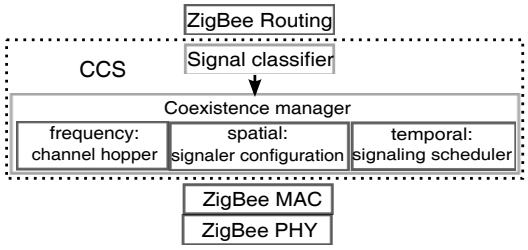

Fig. 7. Architecture of the CCS framework.

These traces reveal more deficiencies of ZigBee in avoiding WiFi interference. First, packets sent without sensing, such as GTS, ACK, and beacons, will be corrupted when encountering an ongoing WiFi session. Second, WiFi can preempt a ZigBee transmission when its carrier sensing falls in the rx/tx switching time of ZigBee transmitters (192 μs), or the time spent in waiting for the next CSMA slot boundary (up to a backoff slot, 320 μs [25]), as also illustrated in Fig. 5(b). These two cases are essentially due to the long response time of the ZigBee hardware.

Similar to the spatial conflict, temporal collision hazards can also be avoided if the carrier signaling operation is assigned to a separate ZigBee signaler. The signaler can notify WiFi before ZigBee's TDMA packets (or before CCA) by sending a busy-tone, and continue such signaling during the switching time of the co-located ZigBee sender, thereby preventing WiFi preemption. This establishes the key motivation and rationale behind CCS.

IV. Design of CCS

CCS harmonizes the coexistence of ZigBee WPAN with WiFi, via cooperation among the coordinator, clients, and a special ZigBee node designated as the signaler. Fig. 7 depicts the CCS architecture. CCS runs atop the ZigBee MAC/PHY layers. It incorporates a signal classifier that triggers the signaler by estimating WiFi's interference intensity; and a coexistence manager that coordinates the behavior of the ZigBee nodes to prevent WiFi interruptions. The coexistence manager consists of three components:

- Frequency domain: a *temporary channel hopper* that prevents the signaler from interrupting the ongoing transmission of a ZigBee data packet.
- Temporal domain: a *signaling scheduler* that ensures the busy-tone sent by the signaler protects the desired data packet, while conforming to the CSMA spectrum ettiqutte.
- Spatial domain: a *signaler configuration* framework that configures the location and power of the signaler, given a coarse estimation of network parameters, *e.g.*, the maximum link distance of ZigBee.

At a high level, CCS works as follows. It first performs the signaler configuration when the WPAN topology is established. Using the signal classifier, the ZigBee nodes obtains an estimation of the intensity of WiFi interference and packet-loss rate. Based on the estimation, the coordinator decides whether to trigger the signaling mechanism or not. When the signaling is activated, the signaler runs the temporary channel hopper to avoid interfering with data packets when sending busy-tones. The busy-tones are scheduled jointly by

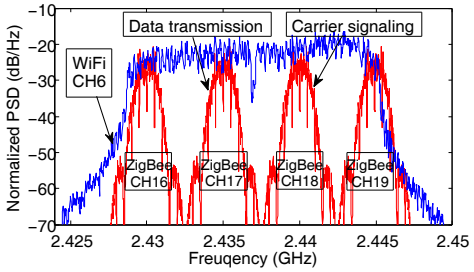

Fig. 8. CCS adopts *temporary channel hopping* to prevent mutual interference between carrier signaling and data transmission.

the coordinator, the clients, and the signaler, according to the MAC mode (TDMA or CSMA). If the WPAN runs delay-sensitive applications, CCS needs to be active persistently to guard against bursty WiFi interference. In such cases, a DC-powered ZigBee node (*e.g.*, the XBee Pro [26]) with similar power as WiFi is necessary as a signaler.

In what follows, we detail the design of CCS's components.

A. Temporary channel hopping

To make the signaling effective, the signaler must emit the busy-tone concurrently with the clients' or coordinator's data transmission. However, the signaler must not interrupt the ongoing or forthcoming transmission from the desired transmitter. A straightforward way to meet this requirement is to ensure sufficient spatial separation between the signaler and the transmitter. However, this solution is too restrictive, since the transmitter needs to fall in the same WPAN as the signaler, and likely within its interference range.

CCS addresses this problem with a *temporary channel-hopper* that leverages the inherent spectrum features of ZigBee and WiFi (Fig. 8). In the 2.4GHz spectrum, the width of each WiFi channel is 20 MHz and the i-th channel is centered at $(2.407 + 0.005i)$GHz, $i \in [1, 11]$. Adjacent channels partially overlap with each other. For ZigBee, the k-th channel is centered at $[2.405 + 0.005(k - 11)]$GHz, $k \in [11, 26]$ [25]. Each ZigBee channel occupies 4MHz, with 1MHz guard band between adjacent channels. As a result, each WiFi channel overlaps with four ZigBee channels.

When running the temporary channel-hopper, a ZigBee signaler switches to a nearby channel before its scheduled signaling, and returns to the original channel immediately after the busy-tone is sent. This approach ensures signaling is decoupled from ZigBee transmission in frequency domain, as adjacent ZigBee channels are orthogonal. However, the busy-tone still overlaps with the WiFi spectrum and can inform WiFi of a ZigBee transmission, as long as its power exceeds the WiFi's carrier sensing threshold.

Note that in practice, a ZigBee channel under interference may reside at the edge of a WiFi band, so CCS must decide on hopping to the left- or right-side adjacent channel. In Fig. 8, for example, when ZigBee runs on channel 19 and is interfered with by WiFi channel 6, the signaling would be ineffective if it hops to the right-side channel 20. To solve this problem, the ZigBee signaler first sends busy-tones on the left-side channel 18. If the packet loss is not alleviated, it switches to the right-side channel 20 instead for signaling. If packet loss persists, CCS recognizes that the current channel 19 is interfered with by two partially-overlapping WiFi bands (channels 6 and 9, spanning 2.426 to 2.448GHz and 2.441 to 2.463GHz, respectively). Such a problem occurs only when both bands carry intensive traffic, and can be resolved using two signalers that hop to channel 18 and channel 20, respectively.

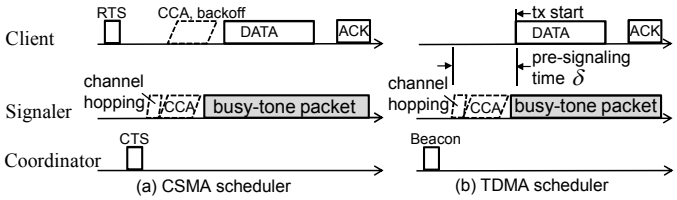

Fig. 9. Scheduling data transmission and carrier signaling.

The channel-hopper incurs channel-switching overhead to the signaler. However, the switching time is limited to 192 μs in ZigBee [25]. This overhead is equivalent to only 4 bytes of airtime, and is the same as the rx/tx switching time.

B. The Signaling Scheduler

CCS maintains the legacy scheduling protocol in ZigBee, but requires the signaler to dispatch the busy tone at an appropriate time, such that *(i)* it reduces the WiFi preemption over ongoing or forthcoming ZigBee transmissions and *(ii)* it minimizes the potential influence on WiFi performance. The signaling scheduler is designed to address this tradeoff. It allows both the CSMA and TDMA modes of ZigBee to coexist with WiFi.

1) CSMA scheduler: The CSMA scheduler is akin to the *indirect transmission* mode in ZigBee [25]. Specifically, a sender polls the receiver before delivering data, and the receiver returns a 5-byte confirmation packet when it is ready to receive (we refer to the polling and confirmation packet as *RTS*, *CTS*, respectively). Upon overhearing the CTS confirmation, the signaler starts the temporary channel-hopper and emits the busy-tone immediately. This handshake process is illustrated in Fig. 9(a).

The busy-tone duration equals the data packet length plus the ACK wait duration and a guard period. The data packet length is a one-byte field piggy-backed in the CTS. The ACK wait duration includes the airtime of ACK packet (352 μs), the rx/tx switching time (192 μs), plus a backoff slot (320 μs) that is needed to ensure slot-boundary alignment [25].

To avoid unnecessary interruptions to WiFi, the signaler also senses the channel after switching to the new channel. It starts signaling only if the channel is idle throughout one CCA slot, and aborts the signaling if detecting a busy channel over 5 consecutive CCA attempts. The other ZigBee transmitters are oblivious of the signaler's behavior and need to perform CCA independently. Since the signaler does not perform backoff and only needs one CCA slot to assess a clear channel, it tends to send the busy-tone before the data transmission. To compensate for this time offset, the busy-tone is extended by a guard period, which equals a CCA slot plus half of the maximum backoff window (4 backoff slots).

To reduce the overhead due to excessive CCA and backoff, the RTS/CTS packets are sent without carrier sensing. As a result of this, RTS/CTS may be lost due to collision with

WiFi packets. However, since the RTS/CTS packets are much shorter than data packets, the collision probability is lower. To further reduce such losses, each RTS packet is retransmitted for RETX times, and the receiver replies with a CTS whenever it receivers an RTS. The RETX value is piggy-backed as a one-byte tag in the CTS packet. The signaler schedules its signaling time according to the first tagged CTS packet it receives.

2) TDMA scheduler: In TDMA mode, CCS exploits the GTS mechanism in ZigBee to allocate fixed slots to clients, thus eliminating the need for per-packet handshake. The slot allocation information is carried in the coordinator's beacon message. Both the client and the signaler sends a confirmation packet via CSMA after receiving such a beacon. The beacon is retransmitted if the confirmatin is missing. Following a successful slot allocation, the signaler will send the busy-tone whenever a scheduled TDMA slot fires, as shown in Fig. 9(b). To reduce unnecessary interference to an ongoing WiFi transmission, the signaler starts CCA δ units of time earlier than the scheduled ZigBee transmission (δ is called *pre-signaling time*). It starts signaling on the first idle CCA, and cancels the signaling if the channel remains busy before the TDMA transmission. The pre-signaling time δ is a design knob serving two purposes: (i) it tolerates imperfect synchronization due to the clock skew between the signaler and transmitter; (ii) it can be used to raise the priority level of ZigBee when running delay-sensitive applications (*e.g.*, real-time monitoring). A larger value of δ allows the signaler to find an idle slot with a higher probability, but at the cost of extra channel time. In our current design, δ is set to the duration of 5 CCA attempts.

In addition to the data packets, the TDMA scheduler can protect beacon messages. Beacon protection is critical for reducing the packet delay, since ZigBee allows for TDMA packet transmission only after successful reception of a beacon for the corresponding superframe. Otherwise, the packet must be postponed to future superframes, implying a typical delay of 1 second. In CCS, the coordinator transmits a CTS packet immediately before the beacon, which will be used by the signaler as a sync message for starting a busy-tone to protect the beacon. Since beacons are short (11 bytes by default), sent infrequently, and tend to have high priority, the signaler is forced to send the busy-tone at the due time of each beacon, assuming that WiFi can promptly recover itself via backoffs and retransmissions, even if collision occurs and corrupts its data transmission.

C. Signaler Configuration

When using a dedicated signaler, one must carefully configure its location and transmit power so that the busy tone may be sensed by the potential WiFi interferers randomly located near the ZigBee WPAN or moving around. Consider a WPAN with one coordinator and multiple clients, with either uplink or downlink traffic. An optimal configuration method may place one signaler near each receiver. However, this will significantly increase the deployment cost. It might also be possible to adaptively place the signaler for the receiver that needs protection. However, it is difficult to realize this due to the random network topology, dynamic traffic pattern, and nodes' mobility.

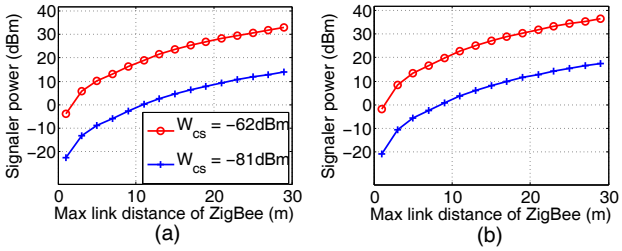

Fig. 10. Required signaling power to ensure the WPAN is sensible by WiFi: (a) uplink (b) downlink.

We adopt a more practical solution in CCS: a single signaler is placed near the coordinator so that all nodes in the WPAN may be equally protected. We verify this intuition based on the insights gained from an empirical propagation model. We show that with an appropriate level of signaling power, the signaler's busy-tone can be visible to any potential WiFi interferers with high probability, and can protect packets addressed not just to the coordinator (uplink traffic), but also to all clients within the WPAN (downlink traffic).

Let D_m be the maximum separation between the coordinator and any client; Λ_w, Λ_z and Λ_s be the transmit power of WiFi, ZigBee, and the signaler, respectively. The WiFi transmitter can cause packet loss (hence becoming a potential interferer) only if its signal power exceeds the desired ZigBee packet power by a capture threshold C_a. To model the signal attenuation, we use an empirical propagation model recommended by the IEEE 802.15 for 2.4GHz indoor environment [17]. At distance d, the signal's path-loss (in dB) is:

$$L_{dB}(d) = \begin{cases} 40.2 + 20\log_{10}(d), & d \leq 8m \\ 58.5 + 33\log_{10}(\frac{d}{8}), & d > 8m \end{cases}$$

Consider uplink first (*i.e.*, the coordinator is the receiver). Given D_m, there exists a maximum distance M_{wr} between the coordinator and potential interferer that can cause packet loss, which satisfies:

$$\Lambda_w[L(M_{wr})]^{-1}(\frac{4}{20}) = \Lambda_z[L(D_m)]^{-1}C_a^{-1} \quad (1)$$

where $L(d) = 10^{0.1L_{dB}(d)}$ is the path loss in normal scale. The term $\frac{4}{20}$ represents the effective power received by WiFi, since ZigBee and WiFi occupy 4MHz and 20MHz bandwidth, respectively. Note that M_{wr} is also the maximum separation between the signaler and any potential interferer. To make the busy tone sensible by WiFi, the signaler's transmit power Λ_s must satisfy:

$$\Lambda_s[L(M_{wr})]^{-1} \geq W_{cs} \quad (2)$$

where W_{cs} is the carrier sensing threshold of WiFi. From Eqs. (1), (2), we obtain the minimum signaler power for protecting the entire WPAN under arbitrarily located interferers:

$$\Lambda_s \geq \frac{\Lambda_w}{5\Lambda_z} W_{cs} L(D_m) C_a \quad (3)$$

For downlink traffic, consider WiFi interferers around a ZigBee client, again with the maximum separation M_{wr}. In the worst case, the distance between the interferer and the signaler is $M_{wr} + D_m$. Therefore, the required signaler power for protecting ZigBee in the worst case is:

$$\Lambda_s \geq \frac{\Lambda_w}{5\Lambda_z} W_{cs} L(D_m + M_{wr}) C_a \quad (4)$$

Fig. 10(a) shows the required power that makes the signaler visible to any potential interferers co-located with the WPAN. We configure the parameters to their typical values (converted to *dB* scale): $\Lambda_w = 15dBm, \Lambda_z = 0dBm, C_a = 10dB$. We consider two carrier sensing thresholds: $W_{cs} = -81dBm$, which is the default for many commercial WiFi cards [27]; and $W_{cs} = -62dBm$, which is the maximum allowable energy sensing threshold for 802.11 a/g/n [28, Sec. 17.3.10.5]. From these results, we observe that the downlink requires higher signaling power, since the signaler is located farther away from the clients than from the coordinator. However, the difference is insignificant, since the signal strength drops sharply with distance. The required signaling power increases with the ZigBee link distance D_m, and with the WiFi carrier sensing threshold W_{cs}. However, even when $W_{cs} = -62dBm$, a normal ZigBee signaler with $0dBm$ power can be effective for short range WPANs (*e.g.*, $D_m < 2m$, as in body-area networks). For longer ZigBee links, we need a dedicated signaler that has power comparable to WiFi's (above 15 *dBm*).

From the above analysis, we conclude that *by placing a single signaler near the coordinator, CCS can prevent the entire WPAN from being interfered with by randomly located (including mobile) WiFi transmitters*. This approach makes CCS deployable in mobile WPANs, as the signaler can always be co-located and moving together with the coordinator. In practice, due to the additional variations caused by small-scale fading (*e.g.*, multi-path and doppler effects), the empirical propagation equation cannot model all cases accurately. However, it allows us to flexibly explore the effects of all possible power-location configurations, and verify the effectiveness of CCS in the average cases. The propagation equation (1) can also be replaced by other empirical models, in order to obtain more accurate estimation for the required signaling power.

D. Signal classifier

Recall that the signaler needs a signal classifier to differentiate WiFi interference and estimate its intensity. CCS exploits the hardware CCA (clear channel assessment) capability of ZigBee devices to achieve this. ZigBee devices can perform three different modes of CCA: energy sensing (mode 1), feature detection (mode 2), and a mixed mode (mode 3). Mode 1 returns busy if the RSSI exceeds a carrier sensing threshold. Mode 2 returns busy only when a valid 802.15.4 signal with the specific spreading and modulation features has been detected [25]. Mode 3 performs a logical OR over the above two modes.

To classify the interference, CCS uses a 2-dimensional carrier sensing scheme that combines the RSSI measurement and CCA feature detection mode (mode 2). In particular, *signals that have high RSSI (i.e., exceeding the energy sensing threshold) but cannot be detected via CCA mode 2, are classified as WiFi interference*. In typical ZigBee-compatible hardware such as the MICAz mote, although only a single CCA mode is allowed at any time, the RSSI register can be read simultaneously with the CCA, thus making the signal classifier feasible.

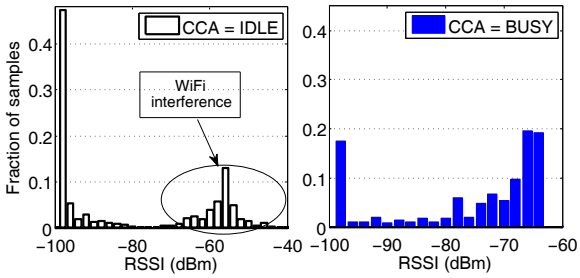

Fig. 11. Distribution of RSSI for different CCA (mode 2) decisions. CCS singles out WiFi interference based on the fact that WiFi signals' RSSI values may exceed the energy-sensing threshold, but CCA still returns "idle".

To verify this simple scheme, we use a MICAz mote E (Fig. 3) to collect the RSSI values and CCA decisions (as required in CCS), when a ZigBee D and WiFi node A are transmitting. We allow the MICAz sensor to read the RSSI register every 128 μs, the minimum time needed for a valid RSSI value [29].

Fig. 11 shows a histogram of RSSI values sampled over 60 seconds. Although partial sensing persists and RSSI varies, WiFi interference can still be singled out by combining the two CCA modes. The four different combinations between two logical decisions (if RSSI is larger than the sensing threshold, and if CCA returns idle) are clustered around different RSSI values and are separated by more than 20 dB from each other. Note that WiFi signals below ZigBee's CCA threshold may be mis-classified, but this does not fundamentally affect CCS since such signals are less likely to interfere with ZigBee transmissions.

After classifying WiFi interference, the clients periodically report the mean RSSI reading to the coordinator, as an indication of the interference intensity. The coordinator triggers the carrier signaling if the interference level exceeds the RSS of any link by the capture threshold C_a. Carrier signaling is also triggered if any link experiences a packet-loss rate that exceeds an application-defined threshold.

E. Application to other heterogeneous networks

CCS can be applied to the coexistence of different protocols that have heterogeneous PHY characteristics and incompatible MAC layers. Specifically, CCS can be used when two coexisting networks have heterogeneous transmit power, scheduling mode, or time resolution. To enable the temporary channel hopping, these networks must be able to tune to different bands. Moreover, at least one of the them must be able to defer its transmission when sensing a busy-tone. We briefly discuss one potential extension of CCS to heterogeneous networks consisting of Bluetooth and WiFi. A detailed study of this is left as our future work.

Bluetooth has been a popular means of establishing WPANs. It has a similar level of transmit power as ZigBee, but runs frequency-hopping based on a TDMA schedule. Existing measurement studies have revealed severe collision problems when a Bluetooth WPAN is co-located with WiFi WLANs (*e.g.*, [2], [10]). To prevent interference between Bluetooth and WiFi on the same hardware platform (*e.g.*, laptop), existing solutions mostly allow them to access the medium alternately

[11]. For nodes on different platforms, the IEEE 802.15.2 standard [30] proposed adaptive channel hopping which allows Bluetooth to hop within the spectrum unused by WiFi. The limitation of this approach has been discussed in Sec. II-A.

CCS can be used as a complementary approach to allow Bluetooth and WiFi to share the same spectrum, via a dedicated Bluetooth-compatible signaler. Since the hopping sequence is known to all nodes in the WPAN, the signaler can hop to the next frequency band before the clients, and perform CCA and signaling, similar to the TDMA mode for ZigBee (Sec. IV-B). Bluetooth has only 1 MHz bandwidth, and consecutive channels are orthogonal to each other, hence the signaling does not interrupt the data transmission. Note that Bluetooth radios can measure RSS and perform CCA directly, but enabling the deferral in the signaler requires firmware modification.

V. Implementation

A. Implementation on TinyOS/MICAz

We have implemented CCS on the TinyOS/MICAz platform using the nesC language. Our implementation builds atop openzb [24], an open-source implementation of the ZigBee MAC/PHY in TinyOS2.0. We have built the major components of CCS, including the CSMA/TDMA scheduler, temporary channel-hopper and interference classification algorithms.

We use the 32768Hz alarm clock in MICAz as an internal timer for the CCS scheduler. The resolution of this timer is 30.5 μs and the best approximation to the 320 μs slot resolution in ZigBee is 10 ticks (305 μs). Currently, we have not incorporated any sophisticated clock-calibration mechanism into the CCS scheduler. Using the USRP2 software radio as a sniffer, we found the clock jitter of the ZigBee timer (without calibration) is on the order of several milliseconds over one superframe (986 ms), which is comparable to the airtime of a data packet. Therefore, it is impractical to rely on the built-in timer to follow a fixed schedule to protect the beacon and GTS packets. We thus allow the coordinator to send 2 CTS packets before beaconing, and before each CFP slot, as synchronization pilots. Due to CTS losses, the actual performance of our implementation would be lower than one with well-calibrated timer.

B. Implementation on GNURadio/USRP2 and ns-2

We have also implemented a high-power *dedicated ZigBee signaler* based on the GNURadio/USRP2 software radio platform [12], [14]. One advantage in using such a dedicated signaler is that it does not restrict packet size, unlike the 127-byte limitation of typical ZigBee devices. A single USRP2 can emit a sufficiently long carrier signal that covers the data–ack roundtrip time of ZigBee packets. In addition, USRP2 has a maximum power level of 20dBm that is comparable to the WiFi transmitter. Hence, it can protect ZigBee WPANs with long link distance (Sec. IV-C).

The key challenge in realizing CCS on GNURadio/USRP2 is that it does not yet support delay-sensitive MAC operations because of its inefficient user-mode signal processing modules. For instance, our first implementation of the beacon protection mechanism was based on the GNURadio 802.15.4 library [31] (we modified the library so that it can support USRP2, and can transmit/receive packets compatible with the MICAz motes). This direct implementation results in a several milliseconds response time (from receiving a CTS packet to hopping channel and sending the busy-tone packet), whereas the response time of a MICAz mote is around 192 μs. Therefore, we performed the following optimization to reduce the USRP2 signaler's response time to a level comparable with a MICAz signaler.

We migrate relevant python components (used to glue the signal processing modules) to C++ level, and implement a radio controller that interfaces directly with the USRP2. We incorporate the transmit path and the receive path into a single module. These two paths are separated in GNURadio, and the context switching between them is a major source of latency. Further, since the signaling effectiveness only depends on the power level rather than the actual content of the signaling packet, we pre-process the signaling packet and log the digital samples corresponding to its modulated signals, which will be emitted by the USRP2 signaler as a busy-tone.

In CSMA mode, whenever the signaler receives a CTS packet, it first switches to the signaling channel, blocks waiting for the scheduled signaling time, and then feeds the prefetched signaling packet directly into the USRP2 transmit buffer via the radio controller. With this measure, we are able to reduce the mean response time of signaling to around 200 μs and jitter to 100 μs, which is even shorter than the slot resolution of ZigBee (320μs). An early version of our implementation also attempted to use a passband filter to separate the signaling channel from the data channel. However, the filter involves intensive floating point multiplication and the processing delay (4 to 5 ms) is unacceptably long for our purpose.

For TDMA mode, the PC host's timer is unable to achieve a microsecond scheduling accuracy, especially when running concurrently with the signal processing blocks. In our actual implementation, the MICAz transmitter needs to send a CTS message so as to sync with the USRP2 signaler on a per-packet basis. Since its switching delay between sensing and transmission is still on the order of milliseconds, the USRP2 signaler is unable to perform carrier sensing before emitting the busy-tone. Hence, it will affect WiFi performance more than a full-fledged implementation on carrier sensing based ZigBee nodes. Further evaluation of this approach will be discussed in Sec. VI-E. Note that both the CTS overhead and the lack of carrier sensing result from the limitation of the software radio platform. Implementation of CCS on a ZigBee-compatible high-power transceiver [26] would not suffer from these problems and is left as our future work.

Besides the testbed implementation, we have also implemented CCS in ns-2 (version 2.33) to perform trace-based simulation. Following the 802.15.4 specification, we developed a GTS management module in addition to the existing CSMA module in ns-2. We also implemented the core components of CCS, including the scheduler and channel-hopper schemes, in order to flexibley evaluate the energy consumption of CCS.

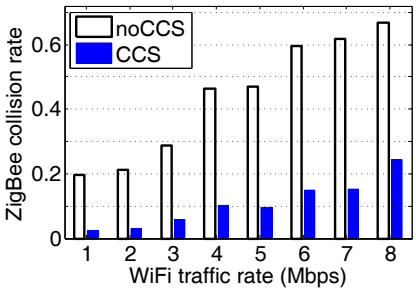

Fig. 12. Packet-loss rate under asymmetric interference.

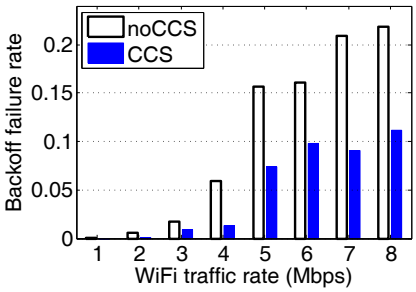

Fig. 13. Percentage of backoff failures in ZigBee CSMA mode.

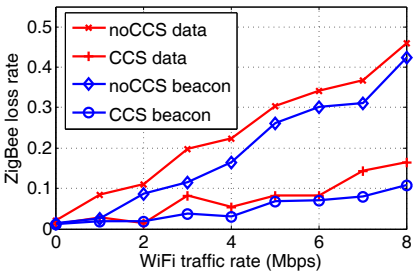

Fig. 14. ZigBee packet-loss rate in TDMA mode.

VI. Evaluation

In this section, we evaluate CCS's performance via testbed and simulation experiments. Our testbed configuration is the same as the measurement setup in Sec. III. The main findings from our evaluation are summarized as follows.

- CCS reduces the WiFi-caused collision by 42% to 90%, depending on the WiFi airtime usage, relative location, scheduling mode, *etc.*. Compared to the retransmission mechanism, it reduces not only packet loss but also the average packet delay by up to 63%.
- CCS can improve the performance of an entire WPAN with multiple ZigBee nodes, which coexist with randomly located WiFi transmitters. Compared to an ideal adaptive channel allocation protocol, it achieves a similar long-term throughput, but a 59.4% less disruption time.
- CCS may consume 10+% more energy when running the CSMA scheduler and a ZigBee signaler. In TDMA mode, however, it saves energy by 73.2% to 83.3% with a dedicated signaler.
- WiFi performance is degraded when running CCS or ZigBee in case of high airtime usage (60%), but unaffected under low duty-cycle traffic (below 10%).

A. Link-level performance improvement

We now quantify the effectiveness of CCS in alleviating collisions caused by spatial and temporal collision hazards, thereby improving ZigBee's link-level performance. In this set of experiments, two ZigBee signalers are placed near the coordinator, and configured to the maximum power level in order to protect the target link. The beacon and superframe orders (Fig. 2) are set to 4 and 6, respectively, resulting in 986 ms beacon interval.

1) Reducing spatial collision hazards: We select link G→H and set its transmit power to -10dBm, such that it experiences packet losses due to asymmetric interference (Table I). To focus on collision rate and isolate the temporal collision hazards (which will be evaluated separately), the transmitter operates in CSMA mode and sends uni-directional data without ACK. The packet size is 64 bytes and 2 packets are sent in each superframe, corresponding to airtime usage 0.48%. WiFi packet size is 1024 bytes. Its bitrate is fixed at 18Mbps, and the traffic rate is varied from 1Mbps to 8Mbps, corresponding to airtime usage (Γ_w) from 5.6% to 44.4%, which is consistent with existing measurement studies [1], [9].

Fig. 12 illustrates the ZigBee packet-loss rate due to WiFi-caused collisions. As Γ_w increases, the collision rate increases drastically. CCS significantly reduces the collision rate—the reduction ranges from 64% to more than 90%. Ideally, CCS should be able to fully protect the data packets. However, its packet-loss rate also increases under high WiFi interference, due to the increased CTS losses.

In CSMA mode, ZigBee may lose packets due to backoff failures, since its MAC protocol aborts the transmission if the sense-and-backoff fails after 4 retries. Fig. 13 illustrates the fraction of backoff failures among all transmission attempts. Using the signaling mechanism in the CSMA scheduler, CCS can reduce the failure rate by 50% in typical cases. Under high interference, the signaler has less opportunity to find idle slots, so the failure rate increases. In this sense, CCS provides best-effort signaling in CSMA mode, in order to avoid unnecessary interference to WiFi.

2) Reducing temporal collision hazards: To evaluate how CCS alleviates the temporal collision hazards that corrupt beacon packets and TDMA-scheduled packets, we focus on the nodes F→G close to the WiFi transmitter A, and set their transmit power to the maximum value, to isolate the effects of asymmetric interference.

Fig. 14 shows the loss rate of beacon and TDMA data packets. Without carrier sensing, these packets suffer from a high loss rate, especially as WiFi airtime usage increases. Data packets suffer a higher loss rate due to their larger size (while a beacon is only 11 bytes long). By notifying WiFi via the signaler's busy-tone, CCS reduces the packet-loss rate by 42 to 83%, without affecting the normal TDMA schedule.

Temporal collision hazards are also caused by WiFi preemption, which occurs when WiFi starts transmission within the tx/rx switching time between ZigBee data and ACK. We evaluate such an effect by enabling both data and ACK, log the packet sequences, and calculate the percentage of ACK losses due to WiFi preemption and overall packet-loss rate after retransmission. The results (Fig. 15) indicate that ZigBee ACK loss rate can be up to 63% when $\Gamma_w = 44.4\%$. By filling

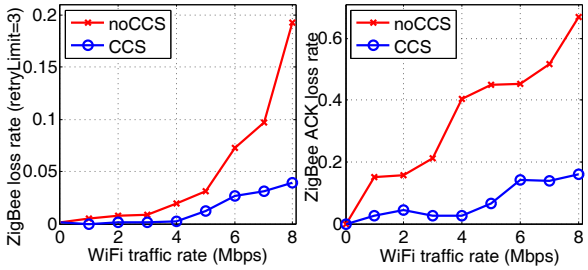

Fig. 15. Loss rate due to WiFi preemption.

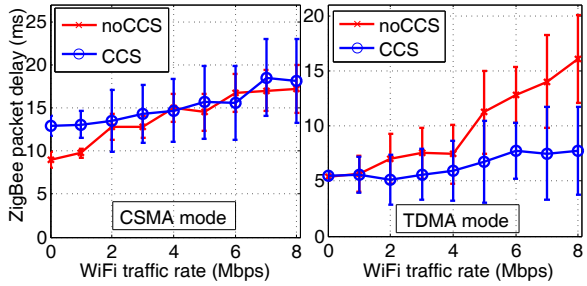

Fig. 16. Packet delivery delay (retryLimit=3).

in the switching time with a busy-tone, CCS reduces the ACK loss rate by more than 50% in most cases. Its packet-loss rate is consistently more than 55% lower than the non-CCS (legacy ZigBee). Although retransmission alleviates the non-CCS's packet loss and caps it below 19%, it should be noted that the experiment is still limited to link F→G which can be sensed by WiFi (as verified in Sec. III-A).

3) Reducing packet delay: When retransmission is enabled, WiFi interference can cause excessive retransmissions, increasing the delivery delay of each packet. In this experiment, we run a full featured ZigBee MAC that retransmits data until an ACK is received or retransmission attempts exceed the retryLimit (default set to 3). Fig. 16 plots the average delay between the generation of a CSMA packet and the corresponding ACK. Due to the additional cost of RTS/CTS packets, CCS extends the delay when the WiFi interference level is low, compared to the built-in retransmission scheme in ZigBee. However, this delay cost is offset by the retransmission time under high WiFi interference. The packets that are lost even after retransmissions are not counted in.

In TDMA mode, packets are sent without the RTS/CTS handshake. Without such overhead, CCS reduces packet delay by up to 63% when the WiFi airtime usage $\Gamma_w = 44.4\%$, as shown in Fig. 16. However, it should be noted that the additional delay due to beacon losses is not counted here. Beacon loss rate for non-CCS can be up to 42% (Fig. 14). This means that with probability 0.42, each TDMA packet sent from the application layer will be delayed by one additional superframe duration (0.986s). By preventing the TDMA packets from being delayed to the next superframe due to a corrupted beacon, CCS can further reduce the actual packet delay perceived by applications.

B. Operational validation for coexisting WPAN and WLAN

The above microbenchmarks showed the effectiveness of CCS in removing the collision hazards for a single link.

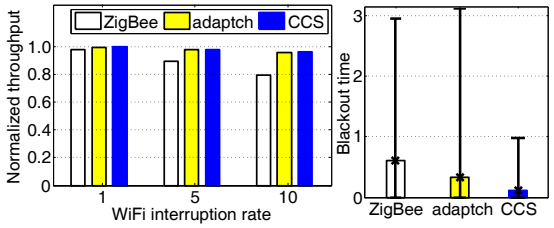

Fig. 18. Performance of CCS vs. ideal adaptive channel allocation (adaptch). (a) Normalized throughput (ratio of the number of uniquely received packets to those sent); (b) blackout time (the number of consecutive superframes with beacon losses) when WiFi interrupts 10 times in 10 minutes. Error bars show the maximum and minimum values.

We proceed to evaluate its performance in a WPAN running multiple ZigBee nodes and with the USRP2 signaler.

The WPAN coordinator is located in E and clients are F, G, H, N, Q. Each client requests one GTS slot and transmits one 64-byte data packet to the coordinator within each superframe, using transmission power 0*dBm*. The WiFi link's airtime usage is fixed at $\Gamma_w = 30\%$. Its AP location varies between A, C, K, and P, and client is fixed at B. Under different scenarios, collision can occur due to both spatial and temporal hazards. In all the tests, the USRP2 signaler is located close to the WPAN coordinator, in consistent with the signaler configuration (Sec. IV-C). Its transmit gain is set to half of the maximum gain, corresponding to an approximate output power of 10*dBm*.

Figs. 17(a) and 17(b) plot the collision rate (one-way packet loss ratio) for the data and beacon packets, respectively. The collision rate varies with the relative location between the WiFi transmitter and the ZigBee transmitter/receiver. The collision rate is also asymmetric. Data packet (uplink) collision is more severe when the WiFi interferer (location A and C) is closest to the ZigBee coordinator. The beacon (downlink) collision is more severe when the interferer (location P) is closer to the ZigBee clients. When the interferer (location A) is located in between the ZigBee nodes, the collision rate is lower since WiFi senses the ZigBee nodes more effectively. In all of the location settings, CCS reduces the collision rate for both data packets and beacons, with mean reduction 72.6% and 61.1%, demonstrating its effectiveness in removing the collision hazards.

Fig. 17(c) shows the packet-loss rate when retransmission and ACK are enabled. Even with retryLimit 3, the original ZigBee protocol (labeled as *noCCS*) experiences severe losses. By protecting the data and ACK packets, CCS reduces the loss rate by 37.7 to 87.5%, depending on the AP locations. By reducing retransmissions, CCS reduces the packet delay by 22.9 to 28.6% (Fig. 17(d)). These experiments justify that even though the signaler's power and location are fixed, it can facilitate the WPAN's coexistence with randomly located WLANs.

C. Resilience to bursty interference

To evaluate CCS under changing network dynamics, we generate bursty WiFi sessions, each running a UDP file transfer (airtime usage $\Gamma_w = 0.22$) for 20 seconds. As a benchmark comparison, we have also implemented an adaptive frequency-allocation protocol following [32] (referred to as *adaptch*).

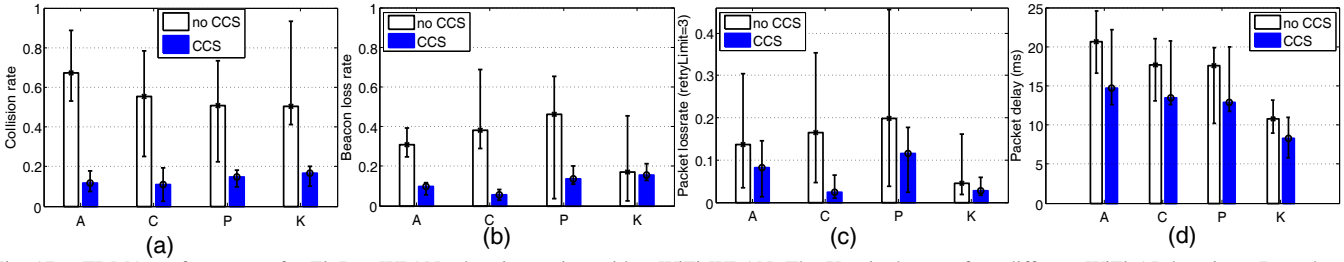

Fig. 17. TDMA performance of a ZigBee WPAN when it coexists with a WiFi WLAN. The X-axis denotes four different WiFi AP locations. Bars denote the performance averaged over all ZigBee links. Error bars denote the maximum and minimum values.

A coordinator in adaptch detects interference by observing packet losses during one beacon frame. It then scans other channels and notifies all clients to switch to a new channel when available.

In the experiments, the ZigBee WPAN is running in TDMA mode. WiFi session starts periodically and interferes with the current ZigBee channel. CCS maintains a dedicated USRP2 signaler, whereas adaptch hops to another channel. We assume an *ideal* scenario for adaptch, where it can always find an unoccupied channel. Fig. 18(a) shows the long-term throughput performance. As the number of WiFi interruptions (within 10 minutes) increases, the legacy ZigBee protocol suffers severe throughput degradation. Both CCS and adaptch can reduce packet losses and maintain more than 95% throughput. Their difference lies in the response time when WiFi interference occurs. As shown in Fig. 18(b), adaptch has larger average and maximum blackout time, due to the need for channel scanning, notification and reassociation. In comparison, CCS reduces the mean and maximum blackout time by 59.4% and 64%, so it is better suited for real-time applications that are sensitive to network outage.

D. Energy consumption

CCS's performance improvement comes at the cost of additional energy consumption at the signalers. Using the ns-2 simulator, we evaluate CCS's energy consumption averaged over the amount of data that is successfully delivered. We adopt the device-specific parameters for ZigBee from the TI CC2420 data sheet [29]. Specifically, we set transmit power to 0dBm, transmit current 17.4mA, receive current 19.7mA, idle current 0.426mA and supply voltage 2.4V. Other parameters, including carrier sensing threshold, packet size, and WiFi data rate, are configured consistently with the link-level experiments. The ns-2 802.15.4 module simulates packet-level energy consumption by transmission, idle listening, and reception. However, it does not model detailed PHY-layer interference and channel. Therefore, we use the measured packet collision probability over the links in the WPAN as the packet-loss rate model.

Fig. 19 illustrates the energy consumed per byte ($\mu J/B$) of the entire WPAN. In CSMA mode, CCS consumes more energy due to the additional overhead in synchronizing the signaling and data transmission. However, in TDMA mode, especially under intensive interference, the additional energy cost is offset by the savings in retransmissions, and the per-packet energy consumption is much lower than simple retransmission. In addition, when using dedicated DC-powered

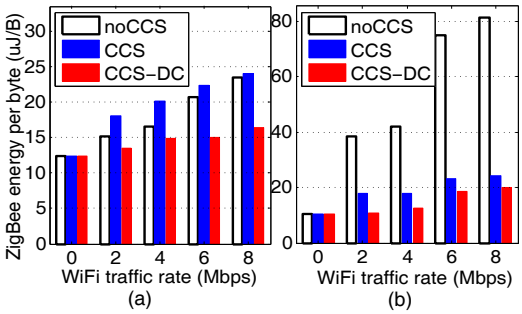

Fig. 19. Energy consumption in (a) CSMA and (b) TDMA mode.

signalers (CCS-DC), CCS reduces the network energy consumption consistently by 73.2% to 83.3%, without incurring any overhead to the low-power motes.

E. Impact on WiFi

Our experimental results have shown that CCS improves ZigBee's performance under moderate to high WiFi traffic. We now evaluate its influence on WiFi's normal operation. We focus on the ZigBee link D→E which is close to the WiFi link A→C. The ZigBee runs in TDMA mode, which is more aggressive in channel acquisition. We fix the WiFi airtime usage at $\Gamma_w = 0.22$ and vary the ZigBee airtime usage Γ_z. Fig. 20 shows that WiFi suffers from severe throughput degradation and longer latency when Γ_z approaches a high value, say 60%. However, this extreme case rarely occurs, since ZigBee targets low-rate applications with typical airtime usage lower than 1%, and up to 10% at its maximum [17]. In such common cases, WiFi performance is virually unaffected by ZigBee, as shown in the figure. More importantly, CCS incurs negligible additional delay or throughput loss compared to the legacy ZigBee, as it may reduce the ZigBee's channel occupancy by reducing retransmissions.

F. Discussion and future work

From the above experiments, CCS is found best applicable for ZigBee WPANs under moderate to high WiFi interference, especially when WiFi traffic is bursty. CCS makes greater performance improvements in TDMA mode, due to low signaling overhead and energy cost. In addition, a DC-powered ZigBee-compatible transceiver (*e.g.*, the XBee node [26]), is preferred as a dedicated signaler. Such a signaler can have comparable transmit power as WiFi, and can persistently protect the entire WPAN under bursty WiFi interference.

Throughout our experiments, the WiFi transmitter is running energy-detection-based CCA with the default threshold

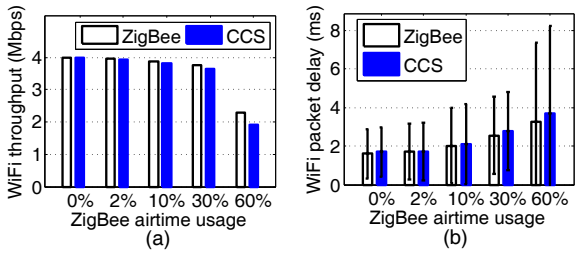

Fig. 20. Impact of ZigBee (TDMA mode) and CCS on WiFi performance.

(around -81dBm [27]). A lower threshold improves its sensitivity to ZigBee, but the threshold cannot be arbitrarily tuned (minimum -90dBm for typical devices [33]), and too low a threshold reduces the spatial reuse between WLAN cells. Also, note that the 802.11b allows for either energy detection or preamble detection mode [28, Sec. 18.4.8.4]. Since the CCS signaler cannot emit 802.11b preambles, it can only help coexistence with the former mode. However, for 802.11a/g/n (which uses OFDM modulation), both modes are mandatory [28, Sec. 17.3.10.5], albeit with different default thresholds (-62dBm and -82dBm, respectively).

Our current evaluation of CCS has left several issues as future work. For instance, how to tune the priority of ZigBee for delay-sensitive applications? CCS can use the pre-signaling time as a key control knob. If the busy-tone signal is emitted much earlier before the data transmission, the probability of sensing failure due to WiFi preemption would be even lower. However, early pre-signaling increases CCS's airtime usage, and may adversely affect WiFi performance, especially when WiFi is running bandwidth-intensive applications. Our current implementation of CCS does not allow for the evaluation of this delicate tradeoff, since the ZigBee timers are not calibrated and not synchronized at the μs level. The USRP2 incurs several milliseconds delay between CCA and data transmission—a much coarser resolution than the ZigBee nodes. We plan to explore the pre-signaling issue on more capable platforms such as the XBee module [26].

The current deployment of CCS is restricted to a single WPAN with star-topology. However, it can be extended to the multihop cluster-tree topology proposed in IEEE 802.15.4 [25]. By augmenting a signaler for each cluster-head, CCS can ensure the entire ZigBee network can coexist with nearby WLANs. Exploring this feasibility is part of our future work.

It should also be noted that we have focused on a single WPAN coexisting with randomly located WiFi WLANs. When multiple co-located WPANs are running CCS, they can simply use the 16 orthogonal ZigBee channels on the 2.4GHz ISM band. Since each WPAN needs two channels (one for the data transmitter and the other for the signaler), CCS can support a maximum of 8 co-located WPANs.

VII. Conclusion

Prior studies have revealed severe ZigBee performance degradation in the presence of WiFi traffic, even if WiFi leaves a sufficient idle time (*e.g.*, more than 67% airtime unused). In this paper, we presented fine-grained experimental evaluation to identify the causes—the spatial and temporal effects that fail the traditional CSMA mechanisms. We introduced CCS, a network-level framework to enhance CSMA and enable ZigBee-WiFi coexistence. CCS adopts a separate ZigBee signaler to emit carrier signals, which improves WiFi's awareness of, and prevents unnecessary interruptions to, ZigBee. CCS incorporates a scheduler to synchronize the signaling with ZigBee data transmission, and a temporary channel-hopping mechanism to avoid interference between the signaler and the transmitter. We have implemented CCS on the TinyOS and GNURadio platforms. Our experimental results have shown a more than 50% reduction of packet collisions when ZigBee operates in the presence of moderate to high WiFi traffic. Both CCS and the original ZigBee have negligible impact on WiFi's throughput and delay when running low duty-cycle applications.

There are two important implications of the CCS framework. First, CCS can be deployed in a real environment that needs both WiFi coverage for Internet access and a ZigBee network for environmental monitoring and control. It is particularly useful for real-time applications, such as medical sensing [34], which adopt ZigBee-based body area networks for patients' health monitoring in hospitals that are usually equipped with WiFi networks for data communications and Internet connections. Second, it represents a simple network-level rationale to assist the coexistence of different protocols that adopt a CSMA-style spectrum etiquette, but have heterogeneous PHY characteristics and incompatible MAC layers. We have briefly discussed how CCS can enhance the CSMA mechanisms in IEEE 802.15.2 (for Bluetooth–WiFi coexistence), which cannot prevent the harmful interference caused by unmanaged WiFi devices. The design philosophy behind CCS, *i.e.*, decoupling carrier signaling from data transmission, can be used to facilitate the coexistence of other heterogeneous wireless networks.

References

[1] C-J. M. Liang, N. B. Priyantha, J. Liu, and A. Terzis, "Surviving Wi-Fi Interference in Low Power ZigBee Networks," in *Proc. of ACM SenSys*, 2010.

[2] R. Gummadi, H. Balakrishnan, and S. Seshan, "Metronome: Coordinating Spectrum Sharing in Heterogeneous Wireless Networks," in *First International Workshop on Communication Systems and Networks (COMSNETS)*, 2009.

[3] S. Pollin, I. Tan, B. Hodge, C. Chun, and A. Bahai, "Harmful Coexistence Between 802.15.4 and 802.11: A Measurement-based Study," in *Proc. of CrownCom*, 2008.

[4] J-H. Hauer, V. Handziski, and A. Wolisz, "Experimental Study of the Impact of WLAN Interference on IEEE 802.15.4 Body Area Networks," in *Proc. of EWSN*, 2009.

[5] M. Petrova, Lili Wu, P. Mahonen, and J. Riihijarvi, "Interference Measurements on Performance Degradation between Colocated IEEE 802.11g/n and IEEE 802.15.4 Networks," in *International Conference on Networking (ICN)*, 2007.

[6] J. Huang, G. Xing, G. Zhou, and R. Zhou, "Beyond Co-existence: Exploiting WiFi White Space for ZigBee Performance Assurance," in *Proc. of IEEE ICNP*, 2010.

[7] S. Pollin, M. Ergen, M. Timmers, L. Van Der Perre, F. Catthoor, I. Moerman, and A. Bahai, "Distributed Cognitive Coexistence of 802.15.4 With 802.11," in *Proc. of CrownCom*, 2006.

[8] C. Won, J.-H. Youn, H. Ali, H. Sharif, and J. Deogun, "Adaptive Radio Channel Allocation for Supporting Coexistence of 802.15.4 and 802.11b," in *Proc. of IEEE VTC*, 2005.

[9] M. Rodrig, C. Reis, R. Mahajan, D. Wetherall, and J. Zahorjan, "Measurement-based Characterization of 802.11 in a Hotspot Setting," in *Proc. of SIGCOMM E-WIND*, 2005.

[10] R. Gummadi, D. Wetherall, B. Greenstein, and S. Seshan, "Understanding and Mitigating the Impact of RF Interference on 802.11 Networks," in *Proc. of ACM SIGCOMM*, 2007.

[11] J. Zhu, A. Waltho, X. Yang, and X. Guo, "Multi-Radio Coexistence: Challenges and Opportunities," in *Proc. of IEEE ICCCN*, 2007.

[12] "The GNU Software Radio," http://gnuradio.org/trac/wiki.

[13] Crossbow Technology Inc., "MICAz Datasheet," 2007.

[14] Ettus Research LLC, "Universal Software Radio Peripheral (USRP)," http://www.ettus.com/.

[15] Crossbow Technology Inc., "Avoiding RF interference between WiFi and Zigbee," Technical Report, 2004.

[16] Schneider Electrics, "ZigBee WiFi Coexistence," http://www.zigbee.org/LearnMore/WhitePapers.aspx, 2008.

[17] IEEE 802.15 Working Group, "Coexistence Analysis of IEEE Std 802.15.4 With Other IEEE Standards and Proposed Standards," 2010.

[18] F. Zhang, F. Wang, B. Dai, and Y. Li, "Performance Evaluation of IEEE 802.15.4 Beacon-Enabled Association Process," in *International Conference on Advanced Information Networking and Applications - Workshops (AINAW)*, 2008.

[19] H. Rahul, N. Kushman, D. Katabi, C. Sodini, and F. Edalat, "Learning to Share: Narrowband-Friendly Wideband Networks," 2008.

[20] F. Tobagi and L. Kleinrock, "Packet Switching in Radio Channels: Part II–The Hidden Terminal Problem in Carrier Sense Multiple-Access and the Busy-Tone Solution," *IEEE Transactions on Communications*, vol. 23, no. 12, 1975.

[21] Z.J. Haas and Jing Deng, "Dual Busy Tone Multiple Access (DBTMA)–a Multiple Access Control Scheme for Ad Hoc Networks," *IEEE Transactions on Communications*, vol. 50, no. 6, 2002.

[22] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC Protocols for Wireless Sensor Networks: a Survey," *IEEE Communications Magazine*, vol. 44, no. 4, 2006.

[23] X. Zhang and K. G. Shin, "Enabling Coexistence of Heterogeneous Wireless Systems: Case for ZigBee and WiFi," in *Proc. of ACM MobiHoc*, 2011.

[24] A. Koubaa A. Cunha, M. Alves, "An IEEE 802.15.4 protocol implementation(in nesC/TinyOS): Reference Guide v1.2 ," Tech. Rep., HURRAY-TR-061106, 2005.

[25] IEEE-SA Standard Board, "IEEE Standard Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)," 2003.

[26] Digi International Inc., "XBee-PRO 802.15.4 OEM RF Modules," http://www.digi.com/.

[27] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan, "Measurement-Based Models of Delivery and Interference in Static Wireless Networks," in *Proc. of ACM SIGCOMM*, 2006.

[28] "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std. 802.11*, 2007.

[29] Texas Instrument, "CC2420 Datasheet," 2004.

[30] "Coexistence of wireless personal area networks with other wireless devices operating in unlicensed frequency bands," *IEEE Std 802.15.2*, 2003.

[31] T. Schmid, "GNU Radio 802.15.4 En- and Decoding," Tech. Rep., UCLA NESL, 2006.

[32] R. C. Shah and L. Nachman, "Interference Detection and Mitigation in IEEE 802.15.4 Networks," in *Proc. of ACM/IEEE IPSN*, 2008.

[33] E. Perahia and R. Stacey, *Next Generation Wireless LANs: Throughput, Robustness, and Reliability in 802.11n*, Cambridge University Press, 2008.

[34] J. Hou, B. Chang, D-K. Cho, and M. Gerla, "Minimizing 802.11 Interference on ZigBee Medical Sensors," in *Proc. of the International Conference on Body Area Networks (BodyNets)*, 2009.