Interconnecting Wi-Fi Devices with IEEE 802.15.4 Devices Without Using a Gateway

Shengrong Yin, Qiang Li, Omprakash Gnawali
Department of Computer Science
University of Houston
{syin, qiangli, gnawali}@cs.uh.edu

Abstract—In many wireless sensing and control application deployments, there is often a gateway device to bridge between the low power IEEE 802.15.4 network and the Internet. The bridge has at least two interfaces. One interface communicates with the 802.15.4 wireless. The other interface either communicates with WiFi or wired network. When a user wants to send a command to the wireless controller, lets say at a smart home, the user may use a smartphone and send command over WiFi to the gateway, often through a cloud service provider. Then gateway shuttles the message from the wired or WiFi chip to the 802.15.4 chip. Then the gateway transmits the messages over the 802.15.4 chip into the 802.15.4 network. In this work, we design a novel modulation technique that runs on the WiFi devices (e.g., smartphone) and demodulation technique that runs on 802.15.4 devices (e.g., a wireless controller in a smart home) to enable WiFi devices to directly communicate with 802.15.4 devices without any gateway. The key idea is to utilize cross-talk between 802.11 and 802.15.4 channels as the medium for communication. We implemented the proposed technique on multiple platforms and are able to successfully achieve a data rate of 2 bytes per second with above 90% accuracy in uncontrolled environments.

Keywords—wireless channel, data rate, coding system, wireless sensor network

I. INTRODUCTION

Wireless sensing and control applications are increasingly being deployed in our homes and environments to enhance comfort for the occupants [1], understand activities and energy use in a home [2], [3], increase energy efficiency [4], and allow better automation and control [5]. Many of these applications require users to interact with the sensor or control devices. For example, the user may want to control the light or thermostat in the house. The user may use a smartphone to perform such control actions. The control actions are conveyed to the wireless sensors or controls through the Internet. Some smart home automation applications are non-interactive. Yet, they require Internet access either to upload the data or download configuration information. Thus, in many scenarios, the wireless sensor and control devices in a smart home, office or environment require communication to or from the Internet.

The existing solution to enable such a communication is through a gateway or a bridging device. At a high level, the device is a router. It has one 802.15.4 interface to communicate with the 802.15.4 network. The other interface may be WiFi or wired. The device shuttles traffic between the two interfaces. A control message (e.g., to turn a light on) coming from a smartphone app, travels to the gateway (possibly through the Internet), is translated appropriately for 15.4 network, and is transmitted by the 15.4 radio. On the software side, the bridging may happen at the application layer (with custom application-specific messages) or at the network layer (with standardized network layer protocols). Recent IETF standards such as 6LoWPAN [6] support development of sensor networks with this architecture. This architecture, which we call gateway-oriented architecture, has served us well as evidenced by vibrant ecosystem of smarthome devices and companies that sell those products. Despite some application deployments use WiFi-based sensors and controllers, 802.15.4 or low-power low-rate radios occupy a unique point in the price and design space that they are likely to be a radio of choice for many years to come.

In this paper, we challenge the premise behind the gateway-oriented architecture: that to enable WiFi devices to send messages to 15.4 devices, we need to build a gateway with the two interfaces. While modern gateway devices provide additional consumer facilities such as local storage service, the gateways that ship with the 15.4 devices are primarily used to bridge between the Internet and the 15.4 network. We propose to eliminate the gateway from the network and enable WiFi devices to directly communicate with the 15.4 devices. If this is possible, we would significantly simplify the deployments and reduce the device and maintenance cost of the networks.

The core idea in our approach is to have WiFi devices transmit packets with special patterns representing the information to be conveyed to the 15.4 network. The transmission is done on a WiFi channel that overlaps with the 15.4 channel on which 15.4 devices are listening. The 15.4 devices sample the signal on the channel due to WiFi transmissions (which we would typically call cross-talk or interference and try hard to avoid) and interpret the information in the pattern.

Building such a modulation and demodulation scheme to enable direct communication between WiFi devices and IEEE 802.15.4 devices has two main challenges. First, WiFi channels and IEEE 802.15.4 channels are allocated for different frequency, though the frequency bands overlap partially. Transmissions on the overlapping channels result in cross-talk and interference rather than communication of data. Second, direct communication requires both the devices to perform modulation and demodulation, compared to the gateway-oriented solution, in which the gateway does the modulation or demodulation using the radios that operate in the specific channel of the frequency band. The demodulation, especially on the 15.4 devices has to be efficient in both power and computation. Any system we design must not only overcome these challenges but also offer at least a modest but usable data rate.
We have designed and implemented the proposed system on multiple WiFi devices (laptop with WiFi interface, and an OpenWRT-compatible wireless AP) and on two mote platforms (TelosB and Opal). We find that the proposed technique can be used to successfully send messages from WiFi devices to 15.4 devices. Even in uncontrolled environments with other APs and Bluetooth in a regular office or residential environment, we were able to achieve a data rate of up to 2 bytes per second with above 90% reliability.

We make these contributions in this work:

- We present the first cross-talk based primitive to enable communication between WiFi devices and IEEE 802.15.4 sensor nodes without a physical gateway. The primitive is the design of a novel modulation scheme that runs on WiFi devices and demodulation scheme that runs on the 15.4 devices.
- We implemented the proposed technique on real WiFi and 15.4 devices and perform experimental validation of the techniques in both controlled anechoic chamber and in uncontrolled environments. We achieve a data rate of 2 bytes per second with a reliability of above 90% in uncontrolled environments.

II. RELATED WORK

We briefly review work related to Internet connectivity to sensor networks and study of cross technology issues in wireless networks.

Connecting to the sensor and control devices from the Internet. Most interesting and useful sensor network and control applications require them to be connected to the Internet for configuration or data access. Some sensor networks use WiFi radios. These networks directly connect to the Internet. Many sensor and control networks use low power radios, such as the 802.15.4-compliant radios. Gateway devices are typically used to bridge to those networks. There have been two major efforts on this front. The first and slightly outdated method uses various application or other types of gateway devices built over serial, USB, or Ethernet hardware interface to a gateway device. Classic TinyOS serial forwarder protocol is an example of this approach. A more modern approach is to use a standardized protocol, such as 6LoWPAN [6]–[9], over the serial or other interface, so the gateway essentially becomes a network-layer routing device. Regardless of the layer at which the message switching occurs, the gateway device needs a 15.4 radio and a wired or WiFi interface where Internet-devices may connect. On research projects, it is common to connect a TelosB [10] or other mote to the computer and use the computer as a gateway between the Internet and the sensor network. In commercial products, the gateway often is a standalone device that connects either to the home router by Ethernet cable or by WiFi. In our work, we propose a technique to connect to the sensor and control devices from the Internet without using a separate physical gateway devices.

Wireless Interference. There has been a large body of work in understanding wireless interference, either within sensor networks or cross-technology interference. [11] presented a real time approach to detect and mitigate cross technology interference. [12] introduced an interference detector which was capable to distinguish different types of interference as well as Wi-Fi beacons. [13] also presented a system which can detect different interferers by observing the disrupted 802.15.4 packet. [14] investigated how to estimate bit error positions in a corrupted packet based on RSSI temporal variations. All the listed papers here assume Wi-Fi activity can corrupt bits in a 802.15.4 packet and design techniques to survive from such interference [15], [16]. There is another body of work that tries to understand the performance of links on different channels [17], [18]. Many such studies empirically studied the performance on channels that also overlap with WiFi thus quantifying the negative impact of WiFi traffic on packet transmission performance on the 15.4 links. In our work, rather than look at interference and cross-talk as a nuisance, we use it to enable communication between WiFi and 15.4 radios.

New Wireless Communication Channels. Recently, new types of wireless channels have been developed for use in sensor networks. For example, Liu et al. [19], [20] presented a design for communication using only ambient RF, by backscattering the ambient RF. There are also interesting work on developing Visual Light Communication channels for communication in wireless sensor networks. For example, Giustiniano et al. [21] and Wang et al. [22] created a visual light communication system with a fully functional Linux-based PHY and MAC layer implementation. Rajagopal et al. [23] enabled light communication for low power embedded devices by utilizing cameras on consumer devices. They achieved a data rate of 1.25 bytes per second. These are examples of research developing new medium for communication. In a similar spirit, in this paper, we develop a new communication channel between WiFi and 802.15.4 by utilizing the cross-talk between the two technologies.

III. SYSTEM OVERVIEW

In this section, we present the design of our system that allows direct communication between a WiFi device and 802.15.4 networks without using a physical gateway device.

A. System Architecture

Our goal is to allow WiFi devices to send messages to the devices that use the 802.15.4 radios. Thus, the users of our system consist of devices in these two networks. First, the components operate in 802.11 networks. For example, iOS and Android based phones, the wireless adapters used
in the laptop, or wireless access points operating in 2.4 GHz frequency band. Second, we also have the platforms deployed in 802.15.4 networks. These are typically low power devices with transceivers operating in 2.4 GHz frequency domain. Examples of such devices include TelosB as research platforms or smart gadgets in smart homes. As shown in Figure 1, the main difference between the prevalent approach and our approach is we enable this communication without the gateway device.

The basic idea of our approach is to make use of the cross technology interference to encode and decode information. Figure 2 shows the main components that makes this type of communication possible. Information is encoded as special timing patterns of IP packets. The packets are sent over WiFi interface of an AP or other wireless devices. The 802.15.4 receiver samples RSSI on the overlapping channel and decodes the timing pattern. The timing pattern represents the information, which is passed to the application. In the following sections, we describe each step in more details with the design nuances and tradeoffs.

B. Utilizing Cross-Talk Between 802.11 and 802.15.4

Our approach takes advantage of cross technology interference that exists between the 2.4 GHz channels and the 802.15.4 channels. The WiFi transmitter does not do anything special at the physical layer to encode information using the cross talk. At the physical layer, the transmission looks like transmission of any other packets. The 802.15.4 receiver, however, is not designed to receive packets from 802.11. So, a regular reception mode does not work. Instead, the receiver samples the RSSI on the channel at a few KHz. The signals transmitted in 802.11 channels can be received (even though the packets cannot be decoded) in the nearby 15.4 channels (Fig. 4). Such transmissions cause the 15.4 channels to be many times saturated with the signal. This leaked signal can be detected through background RSSI sampling on the 802.15.4 transceiver. We can modulate and demodulate these cross-talk signals based on the leaked signal characteristics. For example, in our system, we modulate the leaked signal to enable the communication from WiFi devices to IEEE 802.15.4 based sensor node (Fig. 3).

C. Modulation by WiFi Devices

WiFi devices modulate the cross-talk signal to send information to the sensor nodes. The information is encoded as timing patterns (on-off). The code itself is represented by controlling the presence and absence of high UDP traffic on the WiFi channel. The presence of high UDP traffic is defined as One. The absence of high UDP traffic is defined as Zero.

For accurate modulation, the timing of the traffic patterns needs to be accurate. In a general-purpose operating system, maintaining accurate timing on the outgoing WiFi interface requires accurate timestamping. For our experimentation, we use the iperf tool. We modified the iperf tool to generate different patterns and enable microsecond-level accuracy in packet generation and transmission timestamp.

Using the modified iperf tool, we can send back to back packets to achieve a maximum packet rate of nearly 3000 packets per second. Each packet has 1500 bytes. In the best case, if we send one packet to saturate the channel (indicating a ‘1’) and wait for one packet to indicate a ‘0’, we can theoretically achieve a data rate of 1.5kbit/s. However, sending one packet will only take 300 us. This symbol rate will typically be too fast for sensor nodes to decode successfully without errors. The decoder would need to be synchronized and perform high speed channel sampling. Thus, in our system we use much lower symbol rate so even a modest sensor platform such as TelosB or an Opal can decode the information correctly.

D. Demodulation by Sensor Nodes

We now describe how the sensor node detects the channel and decodes the information on that channel.

1) Channel Detection: There are two models for how the sensor node decides on the channel to use for reception. The first model is manual configuration. This approach is no different from how we configure many WiFi or sensor...

Fig. 2. Components of the proposed communication system that utilizes cross-talk between 802.11 and 802.15.4 channels.

Fig. 3. Transmission from WiFi devices to IEEE 802.15.4 sensor nodes on cross-talk channels. The sensor nodes can detect the presence or absence of signals on the channel even though they cannot receive the WiFi packets. These signals can be used to encode information. In this example, presence or absence of signal on the channel is used to decode the bit string “1010”.

Fig. 4. Map of 802.11 and 802.15.4 channels. These two sets of channels overlap with each other and cause cross-talk.
Fig. 5. Signal on the channel sampled by the sensor nodes with WiFi transmitters transmitting on all the channels in a residential building. Each cell represents an average from 11 rounds of 65,536 measurements.

Fig. 6. Signal on the channel sampled by the sensor nodes with WiFi transmitters transmitting on all the channels in an anechoic chamber. Each cell represents an average from 11 rounds of 65,536 measurements.

Fig. 7. Screen shot taken using WiSpy tool during the measurement study in the anechoic chamber. The figure shows quiet channels other than the ones used for WiFi transmissions (red).

devices. For example, when we program sensor devices, we set the radio channel. Similarly, in our system, we can manually configure the sensor device to listen for messages from the WiFi network on the 15.4 channel with the largest overlap with the WiFi channel.

The second model uses automatic detection of channel. We perform several experiments to collect data and provide heuristics to detect the channel used for communication. At a high level, WiFi transmitter sends a known pattern of signals on the channel. The sensor node receiver cycles through all the channels to receive the stated pattern. To test the feasibility of this technique, we perform RF experiments in an anechoic chamber. In the experiment, we had one laptop transmitting packets back to back on WiFI channels 1-11(Figure 7). We also had 16 TelosB motes tuned to 15.4 channels 11-26 sampling their respective channels at 4 KHz. Figure 6 shows the results from measurement study. It shows that whenever the WiFi transmits on a channel, the few channels overlapping with the WiFi channel can successfully sample the channel and detect the signal. The other channels are relatively quiet. Thus, if we cycle through the channels when there are known signal patterns, we may be able to detect channels to be used for reception at least in a controlled or a quiet environment. In an uncontrolled, this heuristics will not work reliably as demonstrated by our second round of measurement studies.

In the second study, we repeat the same measurements but in an apartment building. There are other WiFi and Bluetooth devices and hence may bleed into the channels used for cross-talk based communication. Figure 5 shows the results from these measurements. Although, the pattern has some similarity to the pattern from the controlled environment, there is one important difference: the blue vertical bands indicate certain channels are saturated (from the perspective of the 802.15.4 devices) regardless of the channel used by our WiFi transmitter. This is due to WiFi routers using channels 1 and 6 in the building. Thus, simple state-less channel scanning alone will not be able to detect the channel used for communication in this uncontrolled setting. A robust preamble or channel detection code will be required so the sensor receiver and the WiFi transmitter converge on a channel. An alternative is to perform aggregate analysis (e.g. CDF) of the signals sampled on the channel. Fig. 8 presents the CDF of sampled RSSI from channel 15 to channel 20 from the study in the uncontrolled environment with WiFi transmitter on channel 802.11 6.
find that more than 90% of RSSI samples on channel 15 and 20 are close to -100dBm or lower. The number is 70% for channels 15, 16, 17, 18. Thus, these four channels may be good candidates for cross-talk based communication with WiFi channel 6. Further measurements could narrow down the set of good channels, in this case channel 17 with the more interference. Thus, in our approach, we try to find the channels that offer the most interference (in contrast to interference avoidance work that tries to find the channels with the least interference). Empirically, we have also established that for manual configuration approach, we should use 802.11 channel N together with 802.15.4 channel N+11. This rule also matches the inference from standard channel maps such as the one in Figure 4.

Thus, with measurements in an anechoic chamber and an uncontrolled environment, we test the feasibility of channel scanning to find the channel for communication. We also note that manual configuration of channels is the most reliable way to synchronize the channels like how we configure many sensing systems today.

2) Signal Decoding: During a reasonably strong WiFi transmission, the 15.4 transmitter typically gets saturated. Thus, the RSSI samples show a pattern consisting of small values (when there are no WiFi transmissions) and high value (when there are WiFi transmissions). Figure 9 shows a sample RSSI trace captured at 4KHz by a TelosB mote during a WiFi transmission with our encoding scheme. We can identify the periodic on and off patterns in the raw RSSI values. Figure 10 plots the FFT of the time series signal. The largest peak corresponds to the periodicity of our WiFi-based modulation for cross-talk communication. This result provides evidence about the feasibility of detecting WiFi cross-talk modulation with a 15.4 radio. Given the feasibility, we now design two strategies to demodulate the cross-talk signals without incurring high memory overhead.

**Strategy 1: Percentage of Minimum RSSI:** This is a modification of the standard sliding window-based sampling approach. Assuming modulation rate and the RSSI sampling rate is known, the window size is configured to be:

\[
\text{window size} = \frac{\text{sampling rate}}{\text{bit rate} \times \text{steps within the window size}}
\]
Within each window, we first find the smallest RSSI value. Then, we calculate the percentage of RSSI values that are the smallest RSSI value. Intuitively, on a quiet channel, this percentage will be large, i.e., a symbol 0. On a busy channel, this percentage will be small, i.e., a symbol 1. Interestingly, the minimum RSSI value is recomputed at each window and thus it adapts to varying channel conditions. Figure 12 shows how this decoding scheme works. Algorithm 1 shows the details of this technique:

Algorithm 1 Decoding based on Percentage of Minimum RSSI
Input: RSSI Samples, Window Size
Output: Min. Percentage Samples

Initialization:
1: set window size
2: create buffer
3: preprocessing raw RSSI samplings

LOOP Process:
4: for item in RssiSamples do
5: counter increase
6: if (Counter < WindowSize) then
7: buffer increase
8: else if (Counter = WindowSize) then
9: index increase:
10: calculate the percentage of minimum RSSI.
11: clean buffer
12: end if
13: end for
14: return PercentageSamples of Min.RSSI

Strategy 2: Average Samples. Similar to percentage of the minimum RSSI, the average RSSI method also uses sliding window with the same size and computes the average RSSI in each window. Based on the average RSSI, we find the peak, i.e., “0”, and the valley, i.e., “1”, to decode the information. Figure 11 shows how this decoding scheme works. Algorithm 2 shows the details of this technique:

Algorithm 2 Decoding based on Average RSSI
Input: RSSI Samples, Window Size
Output: Average Samples

Initialization:
1: set window size
2: create buffer

LOOP Process:
3: for item in RssiSamples do
4: counter increase
5: if (Counter < WindowSize) then
6: buffer refill
7: else if (Counter = WindowSize) then
8: index increase:
9: calculate window average.
10: clean buffer
11: end if
12: end for
13: return AverageSamples

E. Bit Rate
With the proposed techniques, we have achieved a data rate of 2 Bytes per second. Theoretically, we can achieve a data rate of 1.5kbit/s through the maximum packet rate transmitted on our WiFi devices, however that will require RSSI sampling and decoding at much faster rate on the motes and we may also suffer from higher bit errors. Besides the challenge in high-speed RSSI sampling, symbol alignment also becomes challenging in a WiFi network with other traffic. Furthermore, the traffic generation must be real-time to ensure that the symbol duration is accurate, which means the traffic generation must be in real time. Otherwise, the decoding signal will not be synchronized with the encoding signal.

IV. System Evaluation
In this section, we evaluate the proposed communication technique.

A. Metrics and Settings
We use BER (Bit Error Rate) as the primary metric to evaluate the system reliability. We perform experiments in both residential and office-like environments since these areas are equipped with a lot of WiFi devices creating a challenging environment for our communication system. Fig. 13 shows the residential setting used in our experiment. This is a 3b2b apartment with Microwave utilities, wireless AP as well as portable devices such as cellphone, tablets, laptops and several Bluetooth speakers. All these WiFi devices are connected to the wireless AP for Internet access. By experimenting in this uncontrolled environment, we can test the robustness, and reliability the system.

B. Packet Rate
Our system uses packets to modulate signals, but the artificial traffic could negatively affect the normal use of WiFi network performance. So our goal is to generate the traffic with the minimum packet rate while maintaining high reliability. In order to achieve this goal, we evaluate our system in a real WiFi network scenario. We generated traffic when video streaming, web browsing, online game sessions were taking place by the residents. Fig. 14 shows the BER achieved at different packet rates, which is correlated with the symbol rate. During the experiment, we connect the WAN port of the wireless AP for Internet access, then start generating bit sequence from devices connected with this wireless AP. To control the environment settings, we allow only one associated
device. We run the wireshark packet capture tool on this associated device, which is a MacBook Pro with an Intel i5 CPU. We capture all the incoming WiFi packet from the Airport Interface. We surveyed the following packet rate indicated in Fig 14. As we can see in Fig 14, the packet rate can directly affect the system reliability. With 980 packets per second, it is possible to achieve a BER of less than 10%.

We configure the communication data rate as 16 bits per second. As we can see in the Figure 14, as the network bandwidth goes up, the Bit Error Rate will significantly decreased to less than 10% and tend to be near 0%. However, the bandwidth is not always good when it goes up. Using large bandwidth has a major influence on the network performance of WiFi network. We even test the network performance when setting the bandwidth up to 1600 packets per second. We recommend 1000 packets per second as our modulated traffic bandwidth. Compared to the BER in lower bandwidth, this setting provided stability and high decoding accuracy.

C. RSSI Sampling Rate

The sensor node samples the channel to interpret the symbols. Lower sampling is less costly in hardware resources and energy. Higher sampling rate makes the system more robust and potentially allows higher data rates but at hardware or energy cost. We performed experiments with different RSSI sampling rates on the motes under different environments to determine the right sampling rates. Table 1 shows the results. With no Internet users connected to the AP during the experiments (without Internet Access), low sampling rate, e.g., 2 KHz, is sufficient to achieve a low BER. When other users are accessing the Internet through the AP (with Internet Access), that was also modulating our information, we needed a higher sampling rate to achieve a low BER. Overall, higher sampling rates are better in a more noisy environment.

D. Decoding Approaches

We first experimented with the reliability of our system under the scenario of the correlation between data rate and BER. We modulated the WiFi signal into 5 continuously increasing data rate. For each modulated data rate, we decode them with different approaches to evaluate their performance. Fig. 15 presents the BER of the two decoding approaches. We evaluate the approach under different data rate we configured in the system. For data rate less than 16 bps, the two approaches have almost the same bit error rate, which is near 0. However, for data rate that is larger than 20 bps, using percentage of minimum RSSI decoding method results in lower BER.

E. Platform Independence

Next, we evaluate if our system works on multiple platforms both on the 802.11 and 802.15.4 networks. On the 802.11 side, we test a wireless AP and a laptop. For sensor nodes, we test with TelosB and Opal motes. We connect both TelosB and Opal motes to a 10 port USB hub which is connected to the laptop. The application provided both platforms 4 kHz RSSI sampling rate. The platform first samples RSSI and saves them to the local flash with fixed size, then we serialdump from the local storage to the laptop for data analysis. Fig. 16 compares the decoding performance under different platforms. For data rate within 16 bps, Opal provides reliable communication with BER less than 7% while TelosB can provide more reliable communication with BER less than 2%. However, the data rate can not be increased to more than 16 bps as that increases the BER.

In the second run, we configure the WiFi transmission device as an laptop. We simply run the iperf tool on this laptop, and initiate an UDP traffic between the laptop and another laptop which is associated with the same wireless AP, we compare the data rate with WiFi transmissions from different WiFi devices and plot it in Fig. 17.

Fig. 17 described the performance of the communication system with different types of WiFi devices as the transmit-
ters We find that the communication is more reliable if the modulation is conducted by wireless AP. Our guess is due to APs being specialized hardware for WiFi packet reception and transmission, they provide better control in timing and signal strength in packet transmissions.

F. Multiple Interferers

We now evaluate the system by exposing the system to different continuous interferers such as Bluetooth and WiFi traffic. We set three groups of interferers. The first is Bluetooth audio streaming. In this case, we connect keyboard, magic mouse and an JBL Bluetooth speaker to MacBook Pro. With this configuration, we change the packet rate to evaluate BER as a function of data rate with a modulation at 16 bps. In the second experiment, we use Bluetooth and Youtube streaming simultaneously on the laptop. In the third run, we use Bluetooth, Youtube streaming and a 3GB file downloading on the laptop to understand the robustness of the system under strong interference. We use the wireless AP as the WiFi transmission device. We connect three TelosB motes as the receiver in the sensor network. Fig. 18 shows the result. It is worth noting that that during file downloading, the WiFi nominal bitrate is always automatically adjusted by the AP. During the data collection, we also do web browsing, google search, online surfing activities. During packet rate change, Bluetooth speaker experience serious time delay. After some while, it recovers to normal state, we consider this as the channel hopping mechanism implemented in Bluetooth protocol. Under all circumstances, the communication achieved 10-15% BER with the highest packet rate of 1800/s. Since this is a very challenging environment, in which even a WiFi-WiFi or 15.4-15.4 communication would experience a lot of losses, it is not surprising that, with smaller traffic rate the BER goes up to 35%. Thus, we find that with appropriate modulation rate, even with heavy interference, the communication system can achieve less than 10% BER.

G. Communication Range

Next, we want to evaluate the performance of our communication system at different distances. It is important that the proposed communication system work up to certain distance. For example, a tablet may need to send a command to a smart device at the home. When tablet is moves around with people, we still need to be able to send the commands to the device. We designed an experiment in a residential apartment to evaluate the quality of communication at different distances. Fig 19 shows the results. In this residential apartment, there are multiple WiFi devices including cell phones, laptops, wireless printer, wireless access points and other reachable access points nearby. We deployed 5 TelosB motes every 7 feet in the living room. We wrote an TinyOS application for the motes to listen for WiFi traffic. We set the RSSI sampling rate as 4kHz. We run the experiment with different packet rate configurations using our modified iperf tool which was running on a commercial Buffalo router based on OpenWRT. The WiFi access point was configured as a transmission power of 17dBm. We plot the correlation between distance and BER under different data rate in Fig 19.

From Fig 19, we can see that data rate which is below 20 bps has a BER less than 10%. It is stable for the data rate under 20 bps within an area less than 35ft * 35ft. We argue that this space is enough for WiFi coverage in residential rooms. In this range, the communication from WiFi devices to IEEE...
802.15.4 sensor nodes achieves reasonable fidelity. With earlier architecture, communication between the wireless sensors and WiFi devices has to be enabled by a physical gateway which provides the protocol translation service. Our evaluation on the working distance suggests within a configured data rate of 2 bytes per second, we can reliably send several command from WiFi network to the low power wireless network up to 35ft. With lower bps, the BER can be close to 0 even at that distance.

V. DISCUSSIONS

The key technical challenge in modulation is precise control in the timing of the packet transmission. In the hardware platforms we experimented with, we were able to use near real-time control of packet timestamps after modifying the standard iperf tool. Currently, we only use a binary code to distinguish packet rate as high and low. We plan to explore other coding schemes as future work.

A bigger challenge to the system we are building may be the requirement that we need to generate extra packets to provide the “carrier” for the symbols. Study of other coding schemes may allow us to lower the packet generation rate without sacrificing data rate while maintaining low BER.

VI. CONCLUSIONS

We designed and implemented a Wi-Fi to 15.4 communication system without any physical gateway. We provide a detailed description of the modulation and demodulation schemes and their evaluations in controlled and uncontrolled environment. The results show our proposed system can provide a reliable wireless communication to interconnect Wi-Fi devices with IEEE 802.15.4 sensor nodes with an achieved data rate of 2 Bytes per second with more than 90% accuracy.

REFERENCES

Fig. 19. BER vs. Distance in terms of data rate.

