Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Computer Engineering and
Networks Laboratory

ETH Zürich
Department of Information Technology and Electrical Engineering
Computer Engineering and Networks Laboratory
Computer Engineering Group

Semester Thesis

Wireless Cross-Platform Communication

Maciej Bednarek

Supervisor: Prof. Dr. Lothar Thiele
Advisor: Roman Lim

January 17, 2014

Abstract

More and more devices and standards start to operate simultaneously in the same 2.4 GHz ISM band. With the development of Internet and ubiquitous computing, a need to provide wireless connectivity to portable devices like laptops, smartphones or tablets has raised. WiFi standard turned out to be a perfect candidate for this application, providing within a reasonable range the end-user oriented reliable Internet connectivity to mobile platforms. On the other hand, Zigbee products basing on IEEE 802.15.4 played always significant role in industry and home applications. But now, other short range radio technologies enter the same radio band. Designers of these systems have to cope with the increasing interference coming from different communication devices. Many studies of mutual coexistence and interference between IEEE 802.15.4 (Zigbee), WiFi, Bluetooth, Wireless USB and other household interference sources have been conducted to understand their mutual impact and develop possible mitigation techniques. On the other hand, the fact that all these devices operate in the same frequency, gives an opportunity to create a cross-standard way of communication. This thesis investigates the similarities and differences between IEEE 802.15.4 and IEEE 802.11 specifications to find an approach of information exchange through energy sensing. We propose a communication scheme based on interference that induces a specific packet delay due to carrier sense backoff. A WiFi receiver sends multiple packets on the channel and is actively sensing Zigbee transmissions by observing the delay between injected frames. This energy based communication system is implemented on Zigbee Tmote sky node and WiFi laptop. The prototype evaluation is performed in few scenarios, in environments with different medium occupancy.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

There are many wireless standards operating in the 2.4 GHz industrial, scientific, and medical radio band (ISM), including WiFi, Zigbee and Bluetooth. They all serve different purposes and utilize different encoding, modulation techniques and frequency bands. However, because they are different protocols designed to communicate only within the same standard, they cannot interpret bits of packets generated by other standards and can pose significant interference to each other, even despite many interference mitigating methods being available. In the regular networks, successful information exchange between varying radio standards is guaranteed by intermediate translating devices operating in both standards.

To avoid this overhead, an idea of direct wireless cross-platform communication has been developed. Radio devices are capable of sensing the channel energy, which can be used to recognise specific interference patterns conveying information between different wireless standards. Data can be encoded using simple variations of On-off keying, having an alphabet (a set of symbols used for communication) represented by different lengths, patterns or signal strengths of transmission.

The scope of this Thesis is to analyse, design and implement an inter-standard wireless communication method on a WiFi device and a Tmote Sky Zigbee sensor node. The performance of the communication scheme will be evaluated on the prototype system. For the detailed task description see Appendix. Among many obstacles to overcome, interference from third devices operating in the shared medium, hardware constraints of platforms used and the closed nature of firmware running in wireless adapters are considered to be the greatest challenges for this thesis.

Chapter 2

Related work

Many papers [1], [2] have been published to show how much interference is created in the shared ISM band between different devices. A high congestion may lead to huge packet loss and as a result very low throughput, weak link quality and reliability. Huge effort [2], [3] has been put to find a working cross-standard solution to this problem, which will guarantee satisfactory operation range, energy consumption, throughput and reasonable cost. On the other hand, researches try to utilize this interference to enable the inter-standard information exchange. For example, WiFi interface, even when in power saving mode, still consumes substantial power [4]. Many solutions to this problem incorporated usage of secondary low power radios [5] to wake up main interface from power saving mode, but the short range of effective communication required additional deployment of multiple intermediate proxies and special servers to extend the sensing range. Nonetheless, the cross-standard communication based on the energy sensing has not yet been studied to that extent as interference mitigation, and therefore the physical modulation and encoding leave many possible scenarios of implementation.

2.1 Esense

Esense [5] is a protocol for communication through energy sensing between devices which has two different physical layers. It also works for devices with same physical layer, which are out of communication range, but able to sense the energy present on the wireless channel. Esense is implemented for IEEE 802.11 and 802.15.4 standards, using an alphabet - a complete set of symbols used in communication, of multiple packet sizes, derived from actual WiFi traces gathered in a measurement setup and basing on the current channel activity. This paper shows that it is possible to build large alphabet sizes, achieving efficient communication between devices using different physical layers.

The proof of this concept is implemented using unidirectional communication from 802.11

to 802.15.4 standard only. The main challenge of this approach is the limited resolution of energy sensing, especially when using high data rates of 802.11g. The accuracy of detecting channel occupancy in the Zigbee radio hardware is limited, as well as the inter-packet gaps in WiFi are unpredictable due to the CSMA/CA protocol. The measurement granularity of this setup is caused by the 32 kHz oscillator used as timer and the hardware which limits the platform to take measurements only every symbol. As a result, high inaccuracy in packet's length readings is observed. In extreme cases, a whole packet can be missed or two adjacent packets interpreted as one because of small time separation. Measurements showed that mote reports often values higher than the actual packet duration and the error margin is at most 2 clock ticks (61 μs). Next experiment showed that the length of inter-frame spacing plays huge role in precise channel occupancy measurement. The alphabet extraction algorithm produces alphabet size of 100-119. The validation test shows that it is possible to derive a large sized alphabet, which will guarantee effective and reliable communication.

The Esense framework is a MAC layer functionality and requires driver changes. The sender must be able to assemble arbitrary packet sizes and the receiver must be able to process interrupts from physical layer. It becomes a limitation for the off-the-shelf platforms, as usually driver do not have access to the radio. This solution requires also updates to pick the alphabet set appropriate for current scenario. Finally, the analysis of regular WiFi traces show the bimodal packet length distribution - majority of the packets are either very small (beacons, ACKs, management frames) or very long, corresponding to the Ethernet's maximum transmission unit (MTU). Because the size of the alphabet is tightly correlated with the bimodal packet length distribution and only transmission lengths occurring vary rare are used as communication symbols, it could be strongly limited in different traffic types, for example in video streaming.

Unlike in Esense, in this thesis main effort will be put to enable information exchange through a cross-platform communication protocol from Tmote Sky node to a laptop with WiFi interface. It appears to be more challenging than setting up the other way of communication, because of the closed nature of firmware running on most wireless chipsets. Moreover, this approach will recognise and measure channel occupancy, by observing the delay of injected packets to the medium, instead of directly measuring noise level on the channel. Therefore, this solution will require no changes to wireless card's firmware and, in order to optimise the performance, minor changes to the open source linux wireless driver. Experiments should prove the practical applicability of the prototype. The development process of the communication setup should be continued in further projects.

2.2 Interference

A good system to detect and classify non-WiFi interference on WiFi traffic is WiFiNet [2]. This framework is able to quantify the impact of each non-WiFi devices present in

the communication range on the actual WiFi throughput. It is also capable of providing exact physical location of the interference source. All these features are implemented on off-the-shelf WiFi cards, which perform regular carrier energy measurements. Second platform worth mentioning is SoNIC [3], a system which enables sensor networks to detect the interference sources they are exposed to. Moreover, it allows to select an appropriate mitigation strategy depending on the interference type, by observing the characteristic disruption of 802.15.4 packets. Contrary to WiFiNet, it doesn't rely on medium sampling to identify the interference origins.

Chapter 3

Background

WiFi and Zigbee are both specifications for wireless communication, based on IEEE standards, 802.11 [6] and 802.15.4 [7] respectively. IEEE 802.11 and 802.15.4 determine only physical and MAC layer for Wireless Local Area Networks (WLAN) and Wireless Personal Area Networks (WPAN), whereas WiFi and Zigbee are guidelines for higher communication protocols. They are managed and developed by WiFi Alliance [8] and Zigbee Alliance [9]. They both operate in the same 2.4 GHz ISM frequency band, offer flexibility and mobility for wireless devices, or enable to create networks of multiple nodes. Also the WiFi and the Zigbee based networks provide confidentiality for their users, implementing efficient encryption protocols. So why did those two wireless standards were developed almost in parallel? Despite many similarities, they also vary in other parameters, like operation range, modulation used, data rate or wireless channels. However, the most significant difference is in the intended application. Zigbee focuses on low-cost, low-speed ubiquitous communication between low-power devices, where no infrastructure is available and the electrical energy is the most scarce resource. Moreover, it supports mesh and self-organising radio networks. Zigbee finds application in traffic management systems, sensor networks, hospitals or home automation, like wireless light switches. The device operating in Zigbee standard must be able to communicate reliably, sometimes even for several years without battery exchange. On the other hand, WiFi has been developed more with an end-user oriented approach. It connects portable devices like laptops, smartphones, tablets, digital players to the Internet via wireless access points. Usual WiFi-hotspots are designed to cover small areas like single flats or offices with Internet connectivity. These two wireless standards were introduced more to complement each other in different applications, than compete for the wireless medium. For the scope of this thesis, only 802.11g protocol will be evaluated, as newer n/ac/ad releases utilise also the 5 GHz frequency band.

	Channels	Range	Data rate	Tx power	Modulation
WiFi	13 × 22 MHz	100 m	54 Mb/s (g)	20 dBm	DSSS/OFDM
Zigbee	16 × 2 MHz	30 m	250 kb/s	0 dBm	DSSS/Q-QPSK

Table 3.1: Differences between WiFi and Zigbee. WiFi outperforms Zigbee specification almost in all categories, however it cannot provide low-power and long-term connectivity in some applications.

3.1 Physical layer

Figure 3.1 presents main differences between WiFi and Zigbee. The same frequency band is divided in a different way. WiFi utilises 13 (in some countries 14) 22 MHz band overlapping wireless channels, whereas Zigbee uses 16, only 2 MHz wide, narrow non-overlapping channels. In both specifications channels are separated with 5 MHz spacing. Obviously, the maximum operation range depends on the environment and varies significantly between indoors and outdoors, but usually WiFi outranges Zigbee. It is mainly because most Zigbee devices have limited transmit power of only 1 mW. WiFi specification provides power limitation of devices with certification to 100 mW.

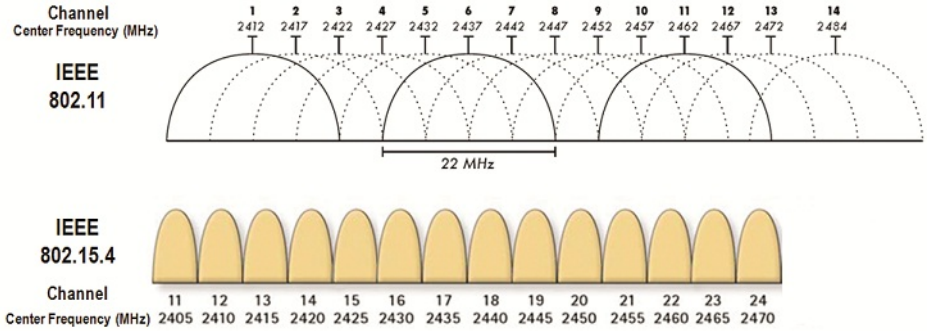


Figure 3.1: Division of 2.4 GHz spectrum into wireless channels for IEEE 802.11 and IEEE 802.15.4 specifications [10].

Both standards are also based on direct sequence spread spectrum modulation (DSSS), where the signal is spread over the whole available bandwidth, to mitigate the impact of narrow-band interference. However, the significance of interference caused by Zigbee to WiFi will be much smaller than WiFi interference at Zigbee device. It's mostly due to Zigbee having smaller channel bandwidth, which is treated as narrowband interference source by WiFi [11]. Additionally, the power conditions mentioned before put Zigbee in tougher situation. Studies on coexistence [12] have shown that the interference from devices operating in different standards may lead to considerable frame loss of even 90%. To minimize the mutual interference, techniques like non-overlapping channel selection, physical separation or dynamic channel allocation are applied. However, these solutions work only when free channels are available and the overall medium occupancy is low.

Moreover, not all environments are static, and usually the interference profile changes dynamically over time, which makes the channel management harder to realise.

3.2 MAC layer

In the MAC layer, Zigbee and WiFi use carrier sense multiple access with collision avoidance access scheme (CSMA). CSMA protocol itself guarantees that each node verifies the absence of other traffic in the channel, before transmitting on the shared medium. It means that always when there is any carrier wave present in the band, a station will wait with its own transmission to initiate, until the medium is sensed idle again, independently of the standard in which the current information exchange is taking place. CSMA/CA is used in many wireless networks, as nodes are not capable of detecting collisions at the receiver, and thus try to avoid them. It uses a binary exponential backoff to reschedule retransmissions after collisions. The backoff period is chosen from the contention window (CW), a QoS parameter determined by the IEEE 802.11 standard for each revision, ranging from 31 (15 for g) to 1023 time slots. CSMA/CA is used in IEEE 802.11 by the basic access scheme called distributed coordination function (DCF), where all stations and traffic are treated with the same priority [13]. The contention window is doubled every time the collision takes place, and reset to minimum whenever the transmission is successful. So whenever a node wants to send a packet - it senses the medium, if it is idle, it waits DCF Inter-frame space (DIFS) time and sends the data only when the channel is still unoccupied. In case when the medium is sensed busy by a node, the transmission is deferred for DIFS period and additional backoff interval, chosen randomly from [0,CW], where CW is the current contention window integer, multiplied by the slot time size.

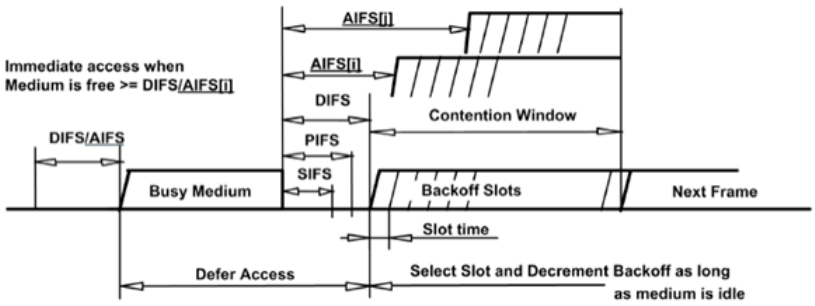


Figure 3.2: IEEE 802.11 DCF mechanism uses contention based CSMA/CA protocol. Inter-frame spacings serve to guarantee different priorities for different coordination schemes and traffic types [14].

Each successful transmission is confirmed by the receiver with ACK to the sender. When no ACK is received, a collision is assumed, the CW is doubled and a new backoff interval is chosen to retransmit the lost packet. The residual backoff decreases gradually only when

medium is sensed idle. Optional to the standard remains the implementation of point coordination function (PCF), where in the Infrastructure mode an access point performs central management and polls stations to transmit. The polled station has right to transmit after shorter PCF inter-frame spacing (PIFS) period, and hence gets higher priority over other network participants. Furthermore, to provide service differentiation Arbitration inter-frame space (AIFS) is used. It can guarantee specified quality of service for particular types of traffic, for example assign highest priority class and thus shortest waiting time to access the medium for real-time applications, like video streaming or VoIP. All inter-frame spacings are summarised in the Figure 3.2.

Table 3.2 compares MAC parameters for different 802.11 revisions. Usually, following dependencies apply: PIFS = SIFS + Slot time; DIFS = SIFS + 2 × Slot time; Arbitrary inter-frame spacing depends on the access category and equals: AIFS[AC] = AIFSN × Slot time + SIFS, where AIFSN is AIFS-number specifying access priority, usually ranging from 2 to 7 [16]. Because 802.11g standard is backward compatible to 802.11b, it can use two slot times.

	SIFS	PIFS	DIFS	AIFS	Slot time	CWmin	CWmax
802.11a	16 μs	25 μs	34 μs	$\geq$ 34 μs	9 μs	15	1023
802.11b	10 μs	30 μs	50 μs	$\geq$ 50 μs	20 μs	31	1023
802.11g	10 μs	19 (30) μs	28 (50) μs	$\geq$ 28 (50) μs	9 (20) μs	15	1023

Table 3.2: Comparison of MAC parameters for different IEEE 802.11 PHY revisions [15].

Chapter 4

Approaches

There exist different methodologies enabling cross-platform communication. They differ in throughput, performance and the extent of invasion into software necessary to make the system working. As the communication from WiFi to 802.15.4 has already been implemented in Esense, my work focuses on the information exchange from Tmote Sky node to laptop with WiFi. This chapter will describe few possible ideas for inter-standard communication based on signal strength measurements, some of which are novel and will be chosen for the prototype.

During carrier sensing, the wireless medium is considered idle or busy, based upon clear channel assessment (CCA) indicator reading. CCA checks the signal energy on the channel just before transmitting [17]. It assumes that any incoming packet will carry high enough signal strength intensity, to exceed the threshold, beneath which everything is considered as noise. Everything above will cause the node to postpone its transmission. CCA can be also used for energy based communication systems. The receiver is not interested in interpreting individual bits of the packet any more. Instead, it senses the energy patterns on the channel. Information can be encoded in signal patterns in various ways: by changing the intensity of the transmission energy, introducing silent gaps between energy bursts to create meaningful patterns, or changing the duration of energy bursts. Each of these approaches may be chosen to convey complete alphabet.

This method of encoding can be referred to as On-off keying (OOK) or Amplitude shift keying (ASK). A transmitter doesn't send packets with content, instead it turns on and off a radio generating carrier wave. This encoding scheme limits to only changing the lengths of signals. Usually binary one is represented by a specific duration of carrier presence, whereas zero is represented by the absence of any activity. However, as the silence gap is very hard to control in shared mediums, making OOK very sensitive to noise, various transmission durations are commonly used to convey additional information, or simply only two lengths are used to encode zeros and ones. ASK represents bits in variations in the amplitude of the transmitted carrier wave. Data are encoded using symbols represented

by particular amplitude. Arbitrary form of OOK or ASK encoding may be applied to following approaches.

4.1 802.15.4 communication

802.15.4 standard usually operates on low power devices with very limited resources. Despite its constraints, it is mainly used for running various mesh network applications on a small operating system like TinyOS or Contiki OS. The availability of open source platforms, guarantees access to all the system functionalities and offers a high degree of flexibility during application design.

Receiving: Zigbee application listening to WiFi transmissions should be able to indicate if there is "something" present on the channel above the noise level. Most of the devices offer clear channel assessment indicator for that purpose. Any signal above the noise threshold, independent of its origin, will trigger the CCA pin. In the cross-platform communication, the receiving application logs the durations of interference patterns using a timer, and compares it the the stored alphabet to decode potential information. In this approach, also the signal level of any transmission can be differentiated, offering additional dimension for encoding.

Sending: In order to broadcast messages, 802.15.4 node must be able to somehow create an active carrier present on the wireless channel. Many platforms offer possibility to introduce so called unmodulated carrier to the RF output. It can be freely used by the sending application to encode information bits into different transmission patterns, transmission lengths, or signal levels - depending on the modulation chosen. It's worth to mention that the outgoing signal doesn't have to abide the Zigbee standard, and can by freely modulated by any 802.15.4 radio. Zigbee frames won't be decoded by the regular WiFi receiver anyway, only the interference pattern they create.

4.2 Cross-standard communication from 802.15.4 to WiFi

WiFi devices offer great potential for rapid information exchange because of high data rates available for communication. But on the other hand, the methodology is constrained with many limitations due to closed source firmware running on wireless chips. As sending messages to Zigbee mote comes down to injecting packets from userspace through regular communication stack, receiving transmissions remains an open issue. Sampling the noise level reported by wireless card is a verified approach to recognise Zigbee messages [5], however do not apply to many off-the-shelf chipsets, as they do not expose the noise measurements to the kernel and upwards, or only with very limited time resolution.

Sending: Libpcap [18] is a portable C library for network traffic capture, and offers, among other functionalities, a possibility to inject arbitrary WiFi packets directly from the userspace. It can force WiFi interface to transmit any valid WiFi frame. Information desired to be exchanged with Zigbee platform may be encoded into different run lengths of transmissions, by adjusting the data rate or the number of bits in the frame. Also, the packets can be bound into specific signal patterns or the transmission power can be modelled and used for encoding if the receiving node is within the known range. Below are few approaches listed to receive Zigbee transmissions without major modifications at the WiFi node.

4.2.1 Communication using collisions

When WiFi node enters infrastructure or Ad-hoc mode, the signal level, noise level and link quality is reported to the kernel for the connection. These values are calculated for each frame and stored in the physical layer convergence protocol (PLCP) header. PLCP sublayer communicates with MAC layer to prepare protocol data units for transmission and to deliver incoming frames from the wireless medium. PLCP provides notification to MAC layer about the incoming packets together with the frame header, including information like data rate or RSSI, which is further passed to higher layers and converted to Signal-to-Noise (SNR) ratio, signal level or link quality. On the other hand, the CSMA/CA protocol guarantees that the WiFi transmission will take place only when the channel is sensed idle. However, it doesn't "detect" collisions, and whenever there is some interference during the ongoing transmission, the sending node won't stop generating signal (WiFi cards are half-duplex). This approach assumes that a WiFi node receives constantly new frames (with RSSI updates), for example by sending constant probe requests to the other node. In that scenario, Zigbee transmissions will collide with the responses. The superposition of signals should produce a detectable difference in reported signal strength of received messages. Hence, the patterns of changes between the real signal level, and the one influenced by Zigbee transmissions could be used to convey information. This method will work provided that WiFi frames remain readable at the receiver. The main challenge could be the frequency of signal level/link quality updates and achievable throughput. Furthermore, the RSSI is calculated only upon the stage of preamble receiving of an 802.11 frame and Zigbee transmissions even when colliding "correctly" with the rest of the packet won't influence the received signal strength.

4.2.2 Noise level

Sometimes wireless adapters enable to read out the noise level directly through driver. Then, a WiFi receiver would need only to probe the card's reading, for example in wireless statistics reported to kernel and log observed signal patterns. Having timestamps of noise level readings, a very simple application decoding messaged contained in energy bursts

could be developed and implemented. However this approach faces many limitations. Most of firmwares are closed-source and do not provide noise readings to upper layers. This could restrain the portability of the solution significantly. Also the update rates of signal/noise levels differ from one chipset to another, sometimes even being averaged out of last few samples.

4.2.3 Packet delays

This is an active method, which can be described as receiving by sending. It measures the delays between injected packets to derive the lengths and patterns of Zigbee transmissions. WiFi device floods channel constantly with arbitrary WiFi packets, and logs the timestamps of these packets reported from the network interface through a packet analyser. CSMA/CA protocol prevents frames to be sent when there is any other transmission ongoing in the channel, thus dummy frames will be injected only when the channel is free. Timestamp differences between the consecutive frames in the trace should indicate the lengths of Zigbee transmissions. The main advantage of this approach is that it requires minor changes to the receiving node and doesn't need any third devices. The open question remains how far away can be the transmitting Zigbee node located to make WiFi consider channel busy and stop from injecting packets. The main idea behind this approach is presented in the Figure 4.1 and we will call it an "active receiving".

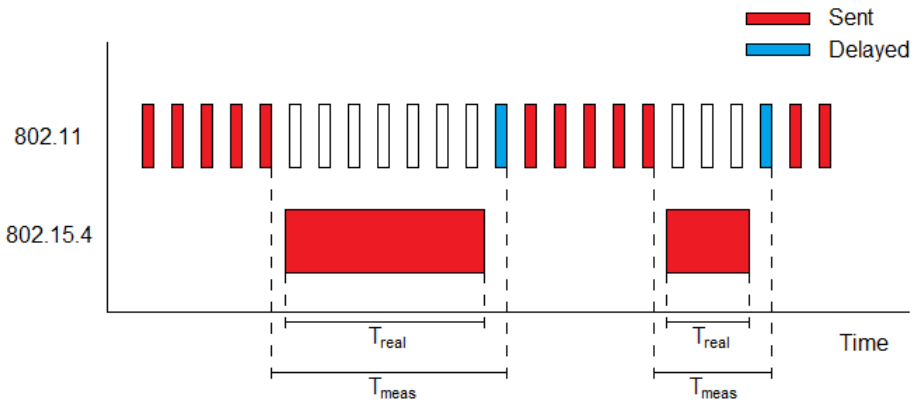


Figure 4.1: Packet delay measurement. During Zigbee transmissions WiFi packets get delayed by the transmission duration and additional measurement error $T_{err} = T_{meas} - T_{real}$

Chapter 5

Implementation

The **Packet delays** approach has been chosen for the prototype, as the method requiring no third devices and portable to other chipsets. It will be implemented on WiFi laptop with Atheros AR5B84 wireless network adapter and Tmote sky node operating in Zigbee standard, with Texas Instruments CC2420 radio. The Tmote sky node [19] is an ultra low power IEEE 802.15.4 compliant wireless sensor module with humidity, light, and temperature sensors. Laptop runs Ubuntu 12.04 Linux with kernel version 3.2.0.57, whereas Tmote sky got Contiki 2.7 operating system programmed.

5.1 Zigbee

Contiki [20] is an open source operating system, designed to connect low-cost, low-power microcontrollers to the Internet. It supports many low-power wireless standards and runs on a range of low-power wireless devices and platforms. Applications are developed in standard C language, which gives users a lot of flexibility during design. Open source nature of the software and a world-wide team of active developer working for the Contiki community makes Contiki perfect choice to implement the cross-platform communication applications on the Tmote sky node.

Receiving: The CC2420 Zigbee radio exposes the clear channel assessment pin through the microcontroller interface. [21] It is a signal based on the received signal strength indicator value and the CCA threshold. It informs whether in medium there is an ongoing transmission present above the CCA threshold (default -77 dBm). It offers much better resolution for measurements than reading out the RSSI register provided by CC2420, which reports an average value of last 8 symbols (128 μs). The best way to measure the durations of WiFi transmissions is to program an interrupt handler triggered by CC2420 radio interrupts, particularly by changes of the CCA pin. Tmote sky node offers two 32 kHz counter registers, which play a role of platform timers. They can be programmed to

be fired by the above mentioned events, and utilized to measure WiFi run lengths. If a valid message is decoded, any further corresponding action can be taken by the receiving application. It's may be useful to disable the watchdog timer running in the Contiki system, to prevent frequent system reboots in case when no events has been detected during the timeout period.

Sending: The CC2420 radio offers also two so called "Transmitter Test Modes", one to transmit an unmodulated carrier, and the second to send a shaped spectrum. The first one is sufficient to introduce dummy transmissions of arbitrary lengths, as Zigbee frames won't be decoded by the regular WiFi receiver anyway. In this mode the unmodulated single carrier is simply passed to the RF output pins, after setting two TX registers and issuing strobe command. Now, the sending application can encode arbitrary bits by turning on and off the carrier and creating signal patterns to be recognised by the WiFi receiver. It has been decided to use two transmission lengths as two symbols, binary zero represented by signal duration of 300 μs and binary one encoded to 600 μs duration. Such lengths provide enough distinguishability from packets, where delays were caused only by the arbitrary inter-frame spacing and packet processing in communication stack, and they doesn't limit performance significantly. Moreover 2.5 ms system delays were used for inter-symbol separations. However, as further observed during experiments, the speed of turning on/off the carrier in the radio is limited. The idea of encoding using bit transitions has been abandoned due to long turning on time of Zigbee radio from the receiving mode.

5.2 WiFi

Laptop with Atheros WiFi adapter is a standard desktop machine, running Ubuntu operating system. To inject WiFi packets, libpcap (packages: pcaputils, libpcap, libpcap-dev) interface for user-level packet capture will be used. It enables to open any network interface for live capture and use routine to send a packet. However it doesn't guarantee it will be emitted by the wireless card. Each injected packet need to have a proper frame formatting, in order to be passed to the air. It consists of IEEE header, radiotap header and payload. The choice fell on the format of probe requests, as they allow to build very short frames of 35 Bytes (37 Bytes on air after header reformatting) and receive highest priority in driver processing. Libpcap requires root privileges to use the network interface for injection. Moreover, the wireless interface must be put into the monitor mode to be able to send created frames.

To optimise chosen approach, minor driver modifications need to be introduced. Because WiFi protocol operates in CSMA/CA access scheme, the wireless adapter will always "backoff" with the frame to send, every time the medium is sensed busy. Because the delays between consecutive injected packets are used to derive Zigbee transmission durations, a new additional measurement error arises. As the MAC access scheme itself cannot be modified because of the closed firmware and consistency with IEEE standard, only driver

parameters will be adjusted to minimise this impact. Backports [22] project provides source code for latest stable ath9k and mac80211 driver releases. Changing the contention window parameters CW_{min} and CW_{max} to 0 will ensure that the injected packets are treated with higher priority - it's a know way of cheating in wireless networks to achieve better throughput. Moreover, all Arbitrary inter-frame spacing categories has been downgraded to 0, in case when packet get classified as some traffic type due to QoS mechanism.

Chapter 6

Evaluation

This chapter presents the evaluation of our proposed cross-platform communication protocol in several experiments. First, a short delay distribution analysis is performed. Then, the packet reception ratio of Zigbee transmissions in two different environments is calculated. Additionally, the portability of the packet delays approach to other chipsets i investigated. Finally, we conclude with a short discussion about obtained results and the limitations of the measurement method will take place.

6.1 Delay analysis

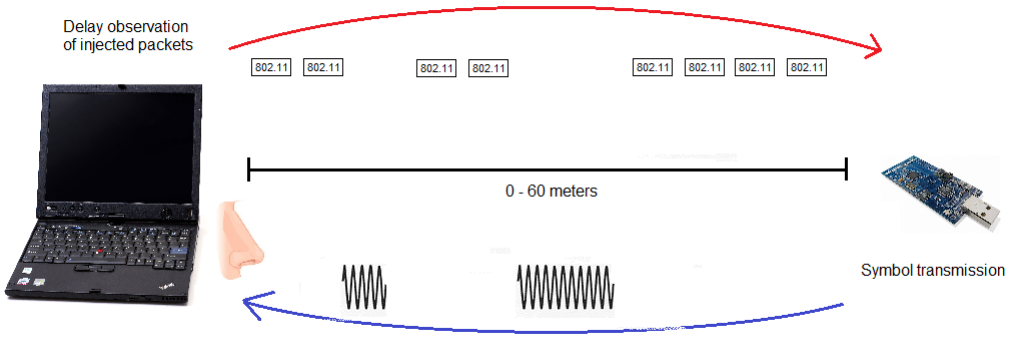


Figure 6.1: Setup for the experiment. WiFi receiver is actively receiving Tmote sky node transmissions, by injecting the probe requests to the medium. Delay at the receiver is observed and analysed to derive the durations of Zigbee transmissions.

6.2 Experiment setup

The experiment has been conducted in the ETZ building of ETH Zurich, on two floors. The A-floor is an underground garage with concrete walls and supports. During the experiment, no cars and no other obstacles where present in the area. Moreover, the analysis of activity in the wireless medium showed, that there were no transmissions detectable from other devices. Second floor chosen for the evaluation was the G one, a normal office environment with many stations and access points within the transmission range. It offers a long 2.5 meters high and approximately 3 meters wide corridor, with many metal boards on walls. Also on the G-floor, between 200 and 400 beacons a second coming from access points were sensed. In both cases, wireless WiFi channel 9 (2.452 GHz - more or less Zigbee equivalent of channel 20) has been chosen for the experiment. In the one end of the corridor (garage), a laptop with Tmote sky node has been place. This laptop was needed only to provide the power to the mote, where Zigbee application has been sending 1000 symbols, with 2.5 ms inter-symbol spacing. The symbol was either zero (300 μs transmission length) or one (600 μs). 1000 symbols were divided into 100 packets of same bitstream: 0100111010. The receiving WiFi laptop has been set up to inject dummy probe requests as fast as possible, in an infinite looping C program using libpcap. 37 byte packets have been transmitted with 54 Mb/s data rate. To log the packets delays, Wireshark network traffic analyser has been launched and configured to capture all packets in promiscuous mode. On each floor, a full trace of 1000 symbols has been sent and actively received by a WiFi laptop at each of the separation distances every 10 up to 60 meters. It took 4 seconds to send and receive all symbols, so on average 250 symbols every second were transmitted. The experiment setup is illustrated in the Figure 6.1.

6.3 Delay analysis

Flooding with short probe requests enabled to send as many as 22000 packets every second on a completely free medium, and around 20000/s when the sending laptop was hearing Zigbee transmissions. It means that on average, every 45 μs a packet has been sent. However, the duration of a 37 byte packet sent with 54 Mb/s should be around $\frac{38*8}{54*1000000} \approx 5.6\mu s$ only. The additional delay may come from the packet processing, as these probe requests are sent from the userspace and it may take some time to go through the complete communication stack. Additionally, according to the IEEE 802.11g specification (see Table 3.2), frames can be sent with at least DIFS separation, that is 28 μs. The reduction of contention window to 0 should prevent that frames are delayed by any further time slots. It is also not known how accurate the timestamp reported by the wireshark is. Figure 6.2 (left) shows distribution of frame delays not influenced by any transmissions. We assume that the general delay of injected packets is between 25 μs and 65 μ seconds, because in this scenario there were no other interference present. The same range of non-influenced delays will be assumed for the G-floor. We observe that the experiment environment also

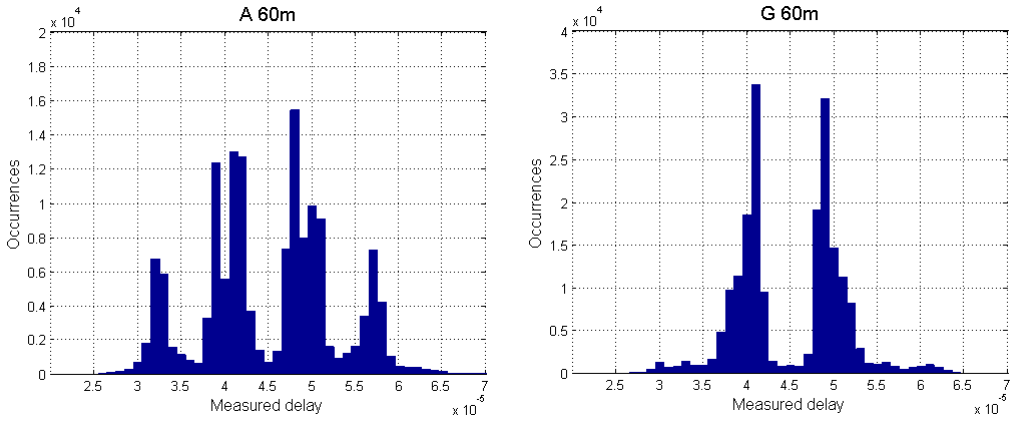


Figure 6.2: Delays between packets when no Zigbee transmissions were sensed. Traces from A (left) and G (right) floors gathered at 60 meters.

plays a role - but this delay should be only hardware dependent and the delay distribution should have been investigated further.

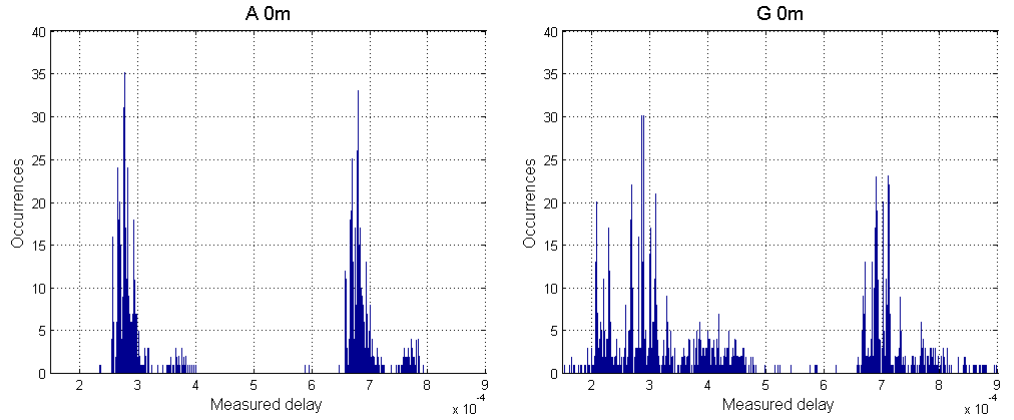


Figure 6.3: Distributions of delays caused by transmissions and packets present in the channel. Each transmission duration is represented by the main and a small side lobe.

Figure 6.3 shows the delay distributions when Zigbee transmissions are active. For the A-floor (left plot), it is easily distinguishable which set of values belongs to which transmission duration. The short symbol falls always in the interval between 240 μs and 400 μs, whereas the long symbol is greater than 650 μs and smaller than 800μs. Only single outliers are visible outside these intervals, hence these ranges will be used to assign symbols. Each transmission falling within this interval will be recognised as 0 or 1 bit, respectively. The symbol detection will be based on these delays. It is important to choose the interval as specific as possible, to filter out different lengths, not belonging to the communication alphabet. In other case, the number of false positives, that is the number of packets that were not sent by Zigbee transmitter, but were recognised by the receiver as valid symbols, could increase. This case is well visible in the right plot, where signal durations being

legitimate symbols coming from Tmote sky node are overlapping with packet lengths from other nodes which were within the transmission range. This brings the first experiment conclusion, that in order to avoid other transmissions being accidentally recognised as our alphabet, it is important to choose the symbols only of durations not used, or sensed very rarely, in the current environment.

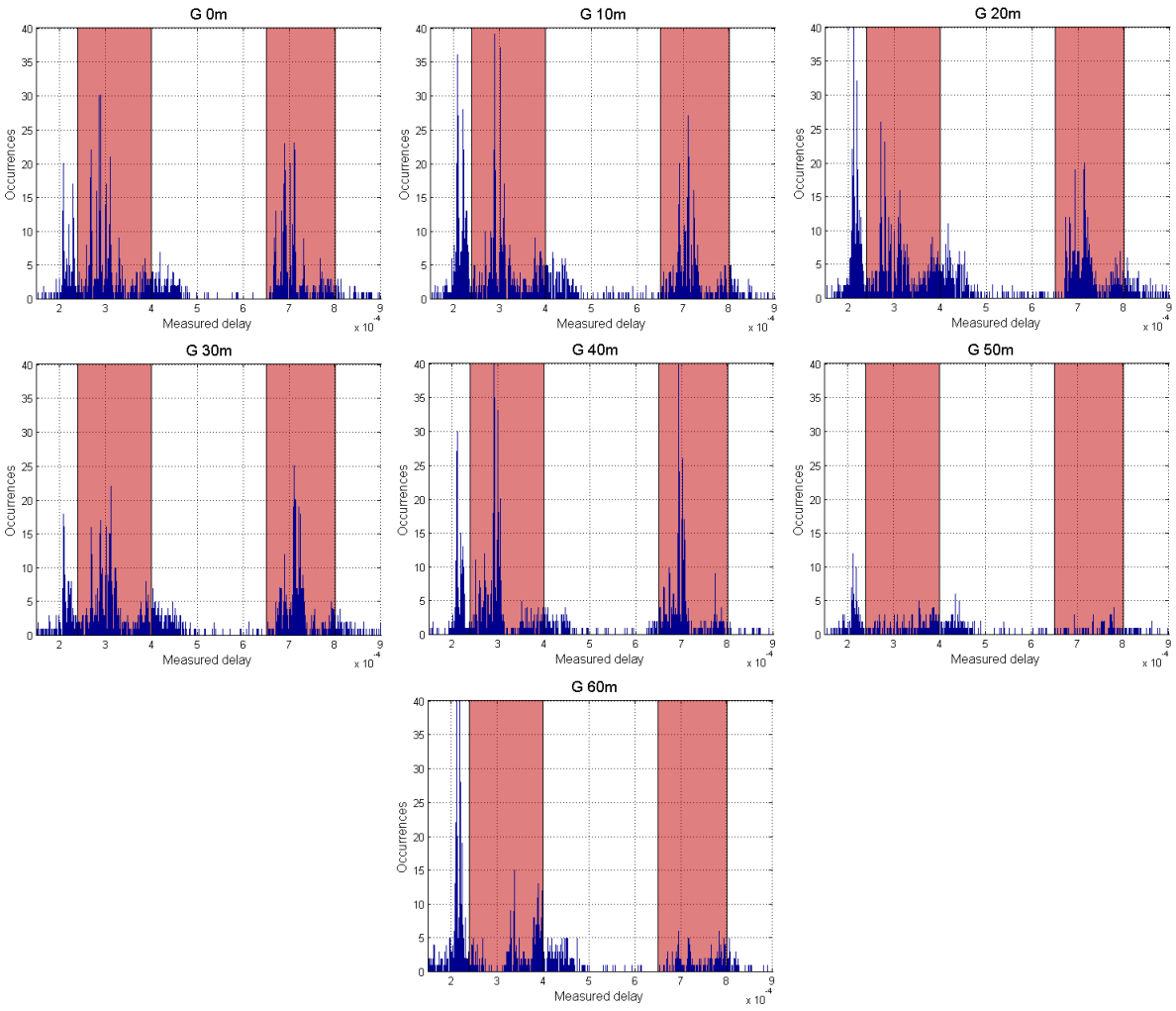


Figure 6.4: Delay distributions on G-floor at all distances. Shaded regions cover intervals where transmissions are decoded as symbols.

Taking average values of sensed durations for short and long (both with their side lobes) symbols, we realise that there are only run lengths of 285 μs and 690 μs. These numbers differ slightly from scheduled 300 μs and 600 μs transmissions. Normally, longer observations than scheduled can be explained with the addition of the inter-frame spacing to WiFi packets. However here the difference is almost 100 μs. Moreover, there should be no observations of durations shorter than actual transmissions, because of the added processing delay and inter-frame space. This error may come from the Tmote sky node and

the application scheduling transmissions, where system delay is used to postpone turning off the radio for a particular amount of time, after switching it on. The clock may not be accurate or the system time unit not equal to real one. Additionally, it may take different times to strobe the CC2420 radio to the transmitting or receiving mode. There is certainly a mismatch between configured and observed values, however it is not crucial for the approach and will not be investigated further. Also the analysis of the inter-symbol gap shows, that it has been measured around 3000 - 3200 μs instead of 2500 μs.

Figure 6.4 compares distributions observed on the G-floor at different distances from a Zigbee transmitter. Distance of 40 meters appeared to be the best place to listen to Zigbee transmissions, where a large number of signal durations fall in the interval for short and long symbols. It means that the performance of this approach is strongly dependent on the interference sensed and on the time of the experiment. The plot taken at 60 meters shows a packet length distribution of other WiFi devices present in the communication range. Some durations between 200-250 μs, around 400 μs and 700-800μs occur more frequently than other, interfering with the chosen alphabet. If prior analysis of this distribution from a regular trace took place, it could be possible to remap Zigbee symbols to unused run lengths, like for example between 500 μs and 650 μs. This would avoid recognition of many other station's transmissions as Zigbee signals, which certainly took place during this experiment. At some separation distances, the number of received zeros was even higher than the number of transmitted by the Tmote sky node.

6.4 Packet reception ratio

For the analysis of packet reception ratio, received delays have been converted into bitstream using intervals mentioned in previous section. Each transmission of duration falling within 240 - 400 μs interval is recognised as the same symbol and decoded into binary 0, whereas signals lasting from 650 μs to 800 μs are decoded as binary 1. Algorithm comparing bitstream to the transmitted pattern (0100111010) takes a window of 10 bits and looks for a match. If the packet is considered as received, the window is shifted to the next 10 bits. If there is no match, the window is moved by one to the next bit. This algorithm continues till the end of the bitstream. Outputs are presented in the Table 6.1.

	0 m	10 m	20 m	30 m	40 m	50 m	60 m
A-floor	94	85	63	0	0	0	0
G-floor	38	21	19	13	46	1	1

Table 6.1: Packet reception ratio: number of packets properly decoded out of 100 sent.

A-floor proves the expected behaviour, with increasing distance, packet reception ratio decreases. Above 20 meters Zigbee transmissions are not differentiable from the background noise to WiFi receiver and it pushes its frames constantly to the medium. The overall

packet reception ratio for G-floor is lower than for A, and it is because of the high medium occupancy there, and hence the chance of having some frames from third devices in between Zigbee symbols in a packet is higher. Though, in this floor, the WiFi laptop was able to receive Zigbee transmissions even at the distance of 40 meters. Very good packet reception ratio here may be justified by lower interference present at that particular place (note that during this experiment the WiFi receiver was moving). Also it is possible, that the longer communication range was induced by metal walls and ceiling in the corridor, which reflected waves and amplified the signal by creating constructive interference.

6.5 Stationary receiver

For this experiment, we have swapped the sender with the receiver and now the WiFi receiver has been placed in the fixed location, while Tmote sky node has been transmitting at different separations. Additionally, an analysis of wireless channels has been performed, to pick the less occupied medium. WiFi channel 3 with the center frequency of 2.422 GHz has been chosen, as there were least WiFi transmissions ongoing. Tmote sky node has been configured to send on the exact same frequency, which previously wasn't the case, as nearest neighbouring Zigbee channel has been chosen to overlap with WiFi's one. To avoid misinterpretation of other transmissions as own symbols, unused lengths of packets on the channel 3 have been adapted as Zigbee symbols. Durations of 500 - 680 μs were decoded as binary zeros, whereas run lengths of 1030 - 1220 μs were translated as ones. Also the inter-symbol gap has been slightly increased. This setup guarantees that most probably only Zigbee signals are decoded as valid symbols (no false positives). Table 6.2 shows packet reception ratio for the trace of that experiment on the G-floor. Now, the algorithm has firstly searched for the matches with the 10-bit pattern and then also for 5-bit packets (same 10-bit pattern considered as two separate). Same as previously, 100 10-bit packets has been transmitted (or 200 of two different patterns if treated as 5-bit).

	0 m	10 m	20 m	30 m	40 m	50 m	60 m
10-bit packets	59	3	4	0	1	1	0
5-bit packets	157	41	61	15	25	17	0

Table 6.2: Packet reception ratio at a stationary receiver. Same pattern was analysed in two ways, either as a single packet or as two different packets.

The results of this experiment show, that the selected channel was only good for the closest area of the WiFi receiver. There could have been very little interference at that place, however at farther separations there were other stations transmitting in that channel. Besides that, at 0 meters there was high floor noise CCA threshold, so only Zigbee signals and WiFi packets in the closest range were detectable. Farther apart, when Zigbee signal strength was much weaker, CCA threshold had been lowered and transmissions from other

stations could have been sensed. The overall packet reception ratio has increased when 10-bit pattern has been treated as two different packets. The numbers from first row (10-bit packets) have not only been doubled, but also additional frames were decoded. The shorter the packet, the less chances that at least one of the bits in the pattern is not sensed.

6.6 Bit error vs. packet error

For the same experiment setup, the number of decoded bits has been calculated (Table 6.3). Despite very high packet reception error at higher distances, many symbols have been properly decoded into bits. As high as 60% of transmitted zeros were decoded at 50 meter separation, where only 1% of 10-bit packets has been received and 9% of 5-bit patterns. Proper selection of channel at the receiver guarantees that no other packets in the air are taken as Zigbee transmissions. Furthermore, precise tuning of the sender's radio to the center frequency of the receiving WiFi station, extends the communication range up to 50 meters. The overall number of decoded ones is significantly lower as zeros, especially for higher separation distances. As 1 symbols were 1100 μs long constant transmissions, their presence on the channel could have been treated by some stations as background noise, and thus didn't make them backoff with their frames. As a result, many long symbols have been received as even longer transmissions, due to overlapping signals.

	0 m	10 m	20 m	30 m	40 m	50 m	60 m
0	491	390	348	311	340	303	1
1	482	246	359	106	185	170	10

Table 6.3: Number of decoded ones and zeros, each out of 500 transmitted.

Although, these figures cannot be interpreted as bit error rate because interference from other stations cannot be excluded, they are good approximation of number of bits correctly decoded. They also indicate that errors are spread over the whole pattern.

6.7 Portability to other chipsets

We have also verified the approach of measuring packet delays on other wireless adapters. Intel Centrino chipsets report very short timings between around 1 to 5 μs between injected packets, and then approximately every fifth packet a very long delay. We called it accumulated delay. Despite this limitation, symbols much shorter than the accumulated delay are detected. Figure 6.5 shows a time trace of delays, with short Zigbee transmissions in red circles. The last signal (circle to the right) has been "absorbed" by the accumulated delay, and is considered as lost. Furthermore, closed source firmwares must be treated as black boxes, that introduce additional uncertainty to measurements. Finally, not all drivers will

allow to change the contention window parameters or the number of time slots used for AIFS traffic categories, thus widening the distribution of received symbol lengths.

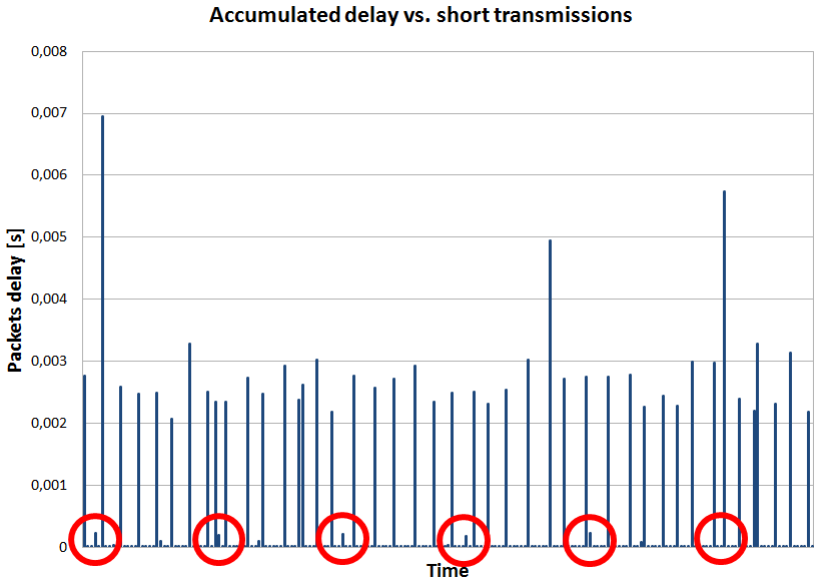


Figure 6.5: Accumulated delay observed for Intel chipsets. Approximately every fifth packet is sent with accumulated delay from previous packets. Even with this limitation, approach of measuring packet delays is able to distinguish between 200 μs long Zigbee transmissions.

6.8 Discussion

Besides good performance in interference free environments, the application of this approach is very limited in interference-rich environments. Uncontrolled collisions and frames from other devices are sensed and received between Zigbee symbols in a packet, making it unreadable. However, this is not the only source of errors, there are many further factors which influence the operation of packet delays method:

- **Inter-symbol transmissions:** The inter-symbol gap has been large enough to leave space for other stations to transmit. 3 ms silence period could even carry few frames from third devices. If the packet duration is in the chosen Zigbee alphabet, it will be decoded as a valid bit. It won't destroy the pattern, only if it falls on the Zigbee pattern boundaries, that is at the beginning or at the end. Otherwise, the whole packet with additional symbol in between is dropped, though the number of bits decoded increases. Inter-symbol transmissions also lead to the next problem of overlapping transmissions.

- **Overlapping transmissions:** Usually Zigbee transmission range is considered to be much shorter than WiFi's. Other devices may not be able to sense its transmissions and thus push their own packets even when there is Zigbee symbol currently present on the channel. If the receiver is within the communication range of both sending devices, it will observe much longer frames and drop them during processing, as they will lie outside the symbol interval. In some cases, short symbols may be decoded as long ones. Additionally, Zigbee sending application doesn't use CSMA/CA protocol, and pushes own frames in regular time intervals. So whenever some other station starts to transmit in the gap and doesn't end before next Zigbee transmission, an overlapping of signals will occur.

- **Sending frequency of the WiFi laptop:** Experiments have shown, that around 22000 frames every second can be injected with receiving station. Figure 6.2 shows that the average delay of uninfluenced packets varies between 25 μs and 65 μs. It limits the resolution of chosen approach, as no transmissions shorter than 45 μs could be distinguished. Moreover, the accuracy of reported timestamps is not known and the processing and injection delays couldn't been measured (it has been assumed that wireless chipset always obeys DIFS time).

- **Sending frequency of Tmote sky node:** As observed, hardware limitations doesn't allow to create arbitrary short transmissions on the unmodulated carrier. It is also meaningless to create signal durations shorter than the hardware delays between injected packets at the receiver, as they wouldn't be differentiated. It also takes some time for the radio to turn on again after being in the receive mode. Finally, the bimodal distribution of transmitted symbols (Figure 6.3) may be due to platform constraints and not the receiving node. This enforces the recognition intervals for symbols to be larger, and therefore creates a possibility that frames for other stations will fall in that interval.

- **Injected frames during Zigbee transmissions:** Above 30 meters on the A-floor, neither a single packet, nor even a bit have been received by the WiFi laptop, despite the other Tmote sky node adapted as transmission logger has been able to sense the packets. For WiFi, these frames couldn't probably exceed the CCA threshold and were classified as noise.

- **System delays:** On the A-floor there, were few longer non-Zigbee delays observed by the receiver, despite the fact that there were no other interfering devices within the communication range.

- **Frequency mismatch:** Tmote sky node supports Zigbee standard and enables to set arbitrary wireless channel. Even when WiFi and Zigbee channels overlap, they do differ in center frequency. If a mote transmission isn't tuned to the center of a WiFi channel, it may be treated as noise, especially at higher communication ranges. Moreover, the CC2420 can only perform 802.15.4 specific modulation. It is not

capable generating a 22 MHz wide signal, but can send on any frequency in 1MHz steps.

- **Varying interference:** When choosing a particular wireless channel for inter-standard communication, one must have in mind that the interference are not always periodic, and vary both between locations and also in time.

There are few possible ways to mitigate theses problems:

- **Decrease the inter-symbol interval:** The lower the inter-symbol interval, the higher are chances that a Zigbee packet will arrive at the WiFi receive, without any intermediate frames coming from third devices. This measure will surely improve performance for environments with periodic interference, like one rich in beacons from access points. However, during the evaluation, a limitation in setting an arbitrary inter-symbol gap in the Tmote sky node has been encountered. The CC2420 radio allowed to produce satisfactory short lengths of transmissions (even down to 200 μs), but as observed during the experiment the radio had to wait around 3 ms to turn on the carrier again.

- **Decrease the CCA threshold for WiFi:** Lowest possible clear channel assessment threshold will guarantee, that no frames are injected, even when the weakest Zigbee transmission is ongoing in the medium. This could increase the transmission range slightly and increase the packet reception ratio by decreasing the number of false negatives, when Zigbee symbol has been sent, however injected frames not delayed at the WiFi receiver. Unfortunately, this adjustment may not have any effect to some chipsets, as usually CCA threshold is adapted dynamically to the channel conditions and set to the floor noise level. The background noise is sensed periodically in the SIFS silence interval, after each packet's reception.

- **Forward error correction:** Channel coding may be used to control the errors in bits over the noisy communication channel. Some of the corrupted symbols could be extracted from the correctly received ones, increasing the number of detected patterns in the transmission. A good choice for forward correction are Reed-Solomon (RS) codes. With a proper way of encoding multiple random symbol errors could be corrected. It can detect any added erroneous symbols to the data, and correct some part of them. Moreover, RS codes can be used to find and correct erasures and perform well with a combination of both errors and erasures. They are widely used in WiMAX standard and could be implemented to increase the number of correctly received patterns in the energy based communication system.

Summing up, a cross-standard system designer must take into consideration many error sources which could even disrupt the communication completely. Some short prior analysis of the wireless channel and interference present within the communication range can

already improve the performance of system significantly. Also little investigation may help to choose non-overlapping symbol durations and avoid interpretation of false transmissions. In most cases, symbols longer then Ethernet's maximum transport unit would be easily recognisable by the receiver. However, longer transmissions block the shared wireless medium and decrease the communication throughput. A trade-off between system reliability and achievable throughput has to be considered.

Chapter 7

Conclusions

The cross-platform communication protocol has been successfully designed and implemented on Zigbee and WiFi devices. The presented prototype uses the approach of measuring packet delays and doesn't require any additional devices or complex changes to software. It enables to introduce precise sensing of wireless medium occupancy directly from the userspace. Observed delays between injected packets have been used to estimate transmission durations on the channel, decode them into alphabet symbols and reproduce the transmitted bitstream. The energy based communication system has been evaluated in experimental tests, showing high reception ratio for interference-free environments. A prior analysis of high occupied mediums may help to avoid interference from third devices and also achieve satisfactory communication reliability. Finally, measures to improve the protocol's throughput and minimise the impact of interference have been discussed. Future standards bring higher data rates and shorter packet durations, which open new possibilities for high precision delay measurements, used for the energy-based inter-standard communication.

Task description

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Autumn Semester 2013

SEMESTER THESIS

for
Maciej Bednarek

Supervisor: Roman Lim

Start date: September 30, 2013
End date: January 17, 2014

Wireless Cross-Platform Communication

Introduction

Nowadays there are several wireless standards using the 2.4 GHz industrial, scientific, and medical radio band (ISM) for communication. Among them are WLAN, Bluetooth, and ZigBee. Although they serve different purposes and use different modulation and encoding of the radio signal, transmissions of different protocol standards can interfere with each other, as can be seen in Figure 1. In order to mitigate interference, methods like carrier sensing, frequency hopping, or spread spectrum modulation are used.

Usually, information exchange between networks of different radio standards can only be achieved by using routing devices that provide a radio front-end for both radio standards, for example a laptop or a smart phone providing support for both Wi-Fi and Bluetooth.

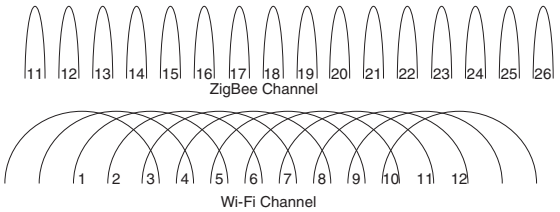


Figure 1: IEEE 802.15.4 and Wi-Fi channels are spread over the same range in the 2.4 GHz band.

Project Goals

In this thesis we are going to explore a different approach, namely to achieve wireless communication directly. Most of these radio devices can sample the energy in a specific channel without receiving valid data. Using that feature it is possible to generate specific interference patterns to convey information between different radio standards.

The devices used in this semester project are a Wi-Fi device (laptop) and a wireless sensor node (Tmote Sky). Apart from employing a wireless transceiver, they do not have much in common: ZigBee devices like the Tmote Sky node are devices with very scarce resources (energy, processing power, memory) and mostly intended to run low-power applications. On the other hand, personal computers are very powerful, as they run CPUs with multiple cores and provide high bandwidth radio communication.

Communication using interference patterns is inherently slower because no sophisticated modulation can be employed. This makes it also more vulnerable to interference induced by other devices. It is therefore crucial to apply appropriate measures for error detection and/or error correction like cyclic redundancy check sums.

Once a bi-directional communication between Tmote Sky nodes and the Wi-Fi device is implemented, a thorough evaluation of the communication scheme is needed to characterize the performance and the limits of this approach.

Task Description

1. Create a project plan and determine milestones, both time wise and topic wise. Care should be taken to allow for ample time for the final presentation and the report.
2. Make yourself familiar with the relevant work related to the wireless communication standards of Wi-Fi and IEEE 802.15.4, interference in wireless communication [1][2] and encoding of information [3][4]. Do a literature research. Search specifically for relevant publications. Examine which of these ideas/concepts can be applied to your project.
3. Download the latest Contiki OS release [5] and get familiar with this operating system that can be used with Tmote Sky.
4. Compile an overview of Wi-Fi hardware that is suitable for energy sensing.
5. Based on the constraints of the available hardware, find a suitable encoding scheme to transfer information using interference patterns.
6. Design an operational prototype and implement it, both on Tmote Sky nodes and the Wi-Fi device.
7. Create a testbed that allows for repeatable tests of the prototype.
8. Evaluate the performance of your communication system (data rates, robustness against interference, range).
9. Document the work done thoroughly by means of a presentation and a written report.

Thesis Organization

- Duration of the work:
  The master thesis starts on September 30, 2013 and ends on January 17, 2014.
- Project plan:
  A project plan with milestones is held and updated continuously. Unforeseen difficulties that change the project plan are documented and discussed with the supervisors in a timely manner.
- Weekly meetings:
  In regular (weekly) meetings with the supervisor, the current state of the work, potential difficulties, as well as future directions are discussed. No later than the morning of the day of the meeting a brief status report should be emailed to the supervisor commenting on these issues, in order to allow for an adequate meeting preparation for the student and the supervisor.
- Initial presentation:
  Two to three weeks after the start of the thesis, the student presents the objectives of the work as well as some background on the topic. The presentation should not exceed 5 minutes (the shorter the better) and consist of no more than three slides.

- Thesis report:
  At the end of the project, no later than January 17, 2014, the student provides a written report to the supervisor (PDF via email suffices, no printing required). Together with the developed software, this report constitutes the main outcome of the project. Code should contain meaningful comments to allow for a follow-up project.
- Final presentation:
  After handing in the report, usually within two weeks after the end of the project, the student presents the achieved results. The presentation should not exceed 20 minutes.
- Evaluation of the work:
  The criteria for grading the work are described in [6].
- Finishing up:
  The used resources (e. g., laptop, desk, keys) should be cleaned up and handed back in. The complete material collected and developed throughout the thesis (design docs, code, thesis sources, etc.) must be committed to the provided subversion repository (CrossCom).

Further information about carrying out a master thesis in our group can be found in [7].

References

[1] J.-H. Hauer, A. Willig, and A. Wolisz, "Mitigating the effects of rf interference through rssi-based error recovery," in *Proceedings of the 7th European conference on Wireless Sensor Networks*, EWSN'10, (Berlin, Heidelberg), pp. 224–239, Springer-Verlag, 2010.

[2] C.-J. M. Liang, N. B. Priyantha, J. Liu, and A. Terzis, "Surviving wi-fi interference in low power zigbee networks," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, (New York, NY, USA), pp. 309–322, ACM, 2010.

[3] K. Chebrolu and A. Dhekne, "Esense: communication through energy sensing," in *Proceedings of the 15th annual international conference on Mobile computing and networking*, MobiCom '09, (New York, NY, USA), pp. 85–96, ACM, 2009.

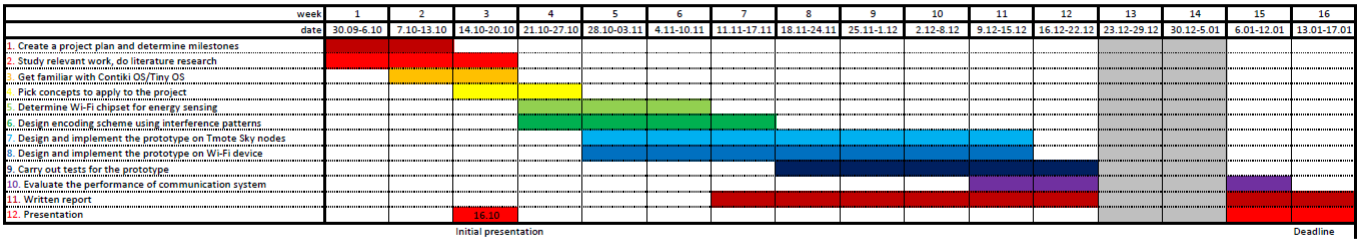
[4] "Manchester coding basics." http://www.atmel.com/Images/doc9164.pdf.

[5] "Contiki: The open source os for the internet of things." http://www.contiki-os.org/.

[6] TIK, "Notengebung bei Studien- und Diplomarbeiten." Computer Engineering and Networks Lab, ETH Zürich, Switzerland, May 1998.

[7] TIK, "Semester and Master Projects: Guidelines for Students and Advisors." Computer Engineering and Networks Lab, ETH Zürich, Switzerland, Apr. 2009.

Workplan


week
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
date
30.09-6.10
7.10-13.10
14.10-20.10
21.10-27.10
28.10-03.11
4.11-10.11
11.11-17.11
18.11-24.11
25.11-1.12
2.12-8.12
9.12-15.12
16.12-22.12
23.12-29.12
30.12-5.01
6.01-12.01
13.01-17.01
1. Create a project plan and determine milestones
2. Study relevant work, do literature research
3. Get familiar with Contiki OS/Tiny OS
4. Pick concepts to apply to the project
5. Determine Wi-Fi chipset for energy sensing
6. Design encoding scheme using interference patterns
7. Design and implement the prototype on Tmote Sky nodes
8. Design and implement the prototype on Wi-Fi device
9. Carry out tests for the prototype
10. Evaluate the performance of communication system
11. Written report
12. Presentation
16.10
Initial presentation
Deadline

Bibliography

[1] C.-J. M. Liang, N. B. Priyantha, J. Liu, and A. Terzis, "Surviving wi-fi interference in low power zigbee networks", SenSys10

[2] S. Rayanchu, A. Patro, S. Banerjee, "Catching Whales and Minnows using WiFiNet: Deconstructing Non-WiFi Interference using WiFi Hardware", NSDI 2012

[3] F. Hermans, O. Rensfelt, T. Voigt, E. Ngai, L. Norden, P. Gunningberg, "SoNIC: Classifying Interference in 802.15.4 Sensor Networks", IPSN 2013

[4] Y. Agarwal, R. Chandra, A. Wolman, P. Bahl, K. Chin, and R. Gupta, "Wireless Wakeups Revisited: Energy Management for VoIP over Wi-Fi Smartphones", MobiSys 2007

[5] Kameswari Chebrolu, Ashutosh Dhekne, "Esense: communication through energy sensing", MobiCom '09

[6] IEEE P802.11, The Working Group for Wireless LANs, http://grouper.ieee.org/groups/802/11/

[7] IEEE 802.15 WPAN Task Group 4 (TG4), http://www.ieee802.org/15/pub/TG4.html

[8] WiFi trade association, http://www.wi-fi.org/

[9] Zigbee Alliance, http://www.zigbee.org/

[10] "Five Factors to Consider When Implementing a Wireless Sensor Network (WSN)", National Instruments 2012

[11] "Atmel AT02845: Coexistence between ZigBee and Other 2.4GHz Products", Atmel MCU Wireless, Atmel Corporation

[12] A. Sikora, "Compatibility of IEEE802.15.4 (Zigbee) with IEEE802.11 (WLAN), Bluetooth, and Microwave Ovens in 2.4 GHz ISM-Band", Technical report, Steibeis-Transfer Centre 2004

[13] Azade Khalaj, Nasser Yazdani, Maseud Rahgozar, “Effect of the contention window size on performance and fairness of the IEEE 802.11 standard”, Wireless Personal Communications 2007

[14] Marc Kuhn, “Mobile Communications: Technology and Quality of Service”, ETH FS 2013

[15] IEEE 802.11 standard, IEEE Standards Association 2012

[16] Marcus Burton, “802.11 Arbitration White Paper”, Certified Wireless Network Professional 2009

[17] J. Collin Engstrom, Chase Gray, and Srihari Nelakuditi, “Clear Channel Assessment in Wireless Sensor Networks”, ACM 2008

[18] Packet Capture library, http://www.tcpdump.org/

[19] Tmote sky datasheet, http://www.eecs.harvard.edu/ konrad/projects/shimmer/references/tmote-sky-datasheet.pdf

[20] “Contiki: The open source os for the internet of things” , http://www.contiki-os.org/

[21] CC2420 Texas Instruments 2.4 GHz IEEE 802.15.4 transceiver, http://www.ti.com/product/cc2420

[22] Linux kernel backports, http://backports.wiki.kernel.org/index.php/Main_Page

[23] Bluetooth Special Interest Group, http://www.bluetooth.org/