DISTRIBUTED SPECTRUM SENSING IN UNLICENSED BANDS USING THE VESNA PLATFORM

Zoltan Padrah

Master Thesis
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia, October 2012

Evaluation Board:

Asst. Prof. Dr. Mihael Mohorčič, Mentor, Chairman, Jožef Stefan Institute, Jamova 39, SI-1000 Ljubljana, Slovenia

Prof. Dr. Gorazd Kandus, Member, Jožef Stefan Institute, Jamova 39, SI-1000 Ljubljana, Slovenia

Asst. Prof. Dr. Tomaž Javornik, Jožef Stefan Institute, Jamova 39, SI-1000 Ljubljana, Slovenia

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL

Zoltan Padrah

Distributed spectrum sensing in unlicensed bands using the VESNA platform

Master Thesis

Porazdeljeno zaznavanje radijskega spektra v nelicenčnih frekvenčnih pasovih z uporabo platforme VESNA

Magistrsko delo

Supervisor: Asst. Prof. Dr. Mihael Mohorčič

Ljubljana, Slovenia, October 2012

Index

Abstract

Some of the concepts and methods for more efficient utilisation of radio spectrum depend on better knowledge of the changing availability of radio resources with time and location, which can be provided by various spectrum sensing approaches. This thesis presents the design and implementation of a spectrum sensing testbed, based on low-cost and low-complexity VESNA platform, aimed to support experimental-driven research in cognitive radio and networking.

First, the state of the art in spectrum sensing methods has been surveyed. Based on the requirements of different sensing algorithms, it has been concluded that the energy detection sensing method is the most suitable for our purposes. Following this decision, a stand-alone spectrum sensing prototype has been developed, which uses a specific radio module and software application. This prototype consists of two parts: the sensing part implemented on VESNA and the control and data collection part implemented on a PC. This stand-alone spectrum sensing prototype has been calibrated and integrated in a heterogeneous sensing system. In this system, it has been used to collect data about received signal power at multiple locations in a laboratory, for several locations of a continuously transmitting signal generator. By presenting obtained experimental results, correct functioning of the calibrated sensing prototype is demonstrated. In a subsequent experiment in more controlled environment, the accuracy of the stand-alone prototype has been determined by measuring a known continuous signal with multiple devices at multiple locations and comparing the results of different device types by using a very simple propagation model.

Multiple VESNA based spectrum sensing prototypes have then been integrated in a distributed spectrum sensing testbed. The testbed consists of two parts: the remote outdoor deployment of multiple VESNA devices and the central management and control system. In order to select the locations of the VESNAs forming the testbed, extensive wireless network design including on the spot measurements has been carried out to ensure that the deployed system will function correctly. The complex communication systems and the requirement for the testbed to function without the need of physical intervention have resulted in the development and use of a custom protocol, because the standardized alternatives would have required more effort to implement.

The thesis presents an example of planning and executing a spectrum sensing experiment in the deployed testbed. In order to support easier and more correct experiment planning, radio link measurements have been performed to determine the coverage areas of individual transceivers. Next, an example experiment, consisting of an emulated interference avoidance scenario, has been planned and performed with a preselected set of nodes. The obtained experimental results are in line with the expected values, proving the correct functioning of the testbed.

Povzetek

Nekateri koncepti in metode za učinkovitejšo rabo radijskega spektra so v veliki meri odvisni od čim boljšega poznavanja spremenljive razpoložljivosti radijskih virov v času in prostoru. V ta namen se lahko uporabijo različni postopki za zaznavanje zasedenosti radijskega spektra. V magistrskem delu predstavljamo zasnovo in izvedbo senzorskega omrežja z uporabo enostavne in poceni platforme VESNA za zaznavanje zasedenosti radijskega spektra. Namen takega senzorskega omrežja je podpora eksperimentalnim raziskavam na področju kognitivnega radia in omrežij.

V magistrskem delu najprej naredimo pregled tehnik zaznavanja radijskega spektra. Na podlagi zahtev različnih algoritmov za zaznavanje smo kot najprimernejšo za naše potrebe izbrali metodo detekcije energije signala. Po tej izbiri smo zasnovali prototip samostojnega vozlišča za zaznavanje radijskega spektra, ki temelji na namensko razvitem radijskem modulu in programski aplikaciji. Prototip sestoji iz senzorskega dela, ki je zasnovan na senzorski platformi VESNA, ter iz dela za zajem podatkov in nadzor, ki je realiziran na osebnem računalniku. To samostojno vozlišče smo umerili in integrirali v heterogen sistem za zaznavanje radijskega spektra, kjer smo v laboratorijskem okolju zbirali podatke o sprejeti jakosti signala za različne lokacije oddajnika in sprejemnika. Predstavljeni eksperimentalni rezultati kažejo na pravilno delovanje umerjenega prototipa vozlišča. V naslednjem eksperimentu v bolj nadzorovanih okoliščinah smo delovanje samostojnega vozlišča primerjali še z vrsto drugih naprav za zaznavanje zasedenosti radijskega spektra in z enostavnim modelom razširjanja valovanja.

V nadaljevanju smo več VESNA vozlišč integrirali v porazdeljeno eksperimentalno senzorsko omrežje za zaznavanje zasedenosti radijskega spektra. Omrežje sestoji iz oddaljene zunanje postavitve VESNA vozlišč in osrednjega sistema za upravljanje in nadzor. Za izbiro lokacij VESNA vozlišč smo izvedli obsežno načrtovanje omrežja, ki je vključevalo tudi meritve signala na terenu, da bi zagotovili pravilno delovanje eksperimentov. Zaradi zapletenih komunikacijskih sistemov ter zahteve, da eksperimentalno omrežje deluje brez fizične intervencije na terenu, smo zasnovali namenski protokol, saj bi standardizirane rešitve zahtevale več truda za realizacijo v omrežju.

Magistrsko delo predstavlja tudi primer načrtovanja in izvedbe poskusa za zaznavanje spektra v eksperimentalnem omrežju. Da bi podprli lažje in bolj pravilno načrtovanje poskusov, smo izvedli meritve radijskih povezav in s tem določili območja pokrivanja posameznih primopredajnikov. Nato smo na izbranem naboru senzorjev izvedli scenarij izogibanja medsebojni interferenci. Dobljeni eksperimentalni rezultati so v skladu s pričakovanimi vrednostmi, kar dokazuje pravilno delovanje eksperimentalnega omrežja.

Abbreviations

ADC	=	Analog-to-Digital Converter
AGC	=	Automatic Gain Control
ARM	=	Advanced RISC Machines
ASCII	=	American Standard Code for Information Interchange
ASK	=	Amplitude Shift Keying
CCC	=	Common Control Channel
CINR	=	Carrier-to-Interference-plus-Noise Ratio
COTS	=	Commercial Off-The Shelf
CRC	=	Cyclic Redundancy Check
DAC	=	Digital-to-Analog Converter
DHCP	=	Dynamic Host Configuration Protocol
DSA	=	Dynamic Spectrum Access
FCC	=	Federal Communications Commission
FPGA	=	Field Programmable Gate Array
FSK	=	Frequency Shift Keying
GCC	=	GNU Compiler Collection
GFSK	=	Gaussian Frequency Shift Keying
GNU	=	GNU is Not Unix
HLP	=	HTTP-like Protocol
HTTP	=	Hypertext Transport Protocol
HTTPS	=	Hypertext Transfer Protocol Secure
I2C	=	Inter-Integrated Circuit
IDE	=	Integrated Development Environment
IETF	=	Internet Engineering Task Force
IrDA	=	Infrared Data Association
ISM	=	Industrial, Scientific and Medical
JTAG	=	Joint Test Action Group
LQI	=	Link Quality Indicator
MSE	=	Mean Squared Error
MSK	=	Minimum-Shift Keying
NAT	=	Network Address Translation
OFDM	=	Orthogonal Frequency-Division Multiplexing
OOK	=	On-Off Keying
PSD	=	Power Spectral Density
RISC	=	Reduced Instruction Set Computing
ROC	=	Receiver Operating Characteristic
RSSI	=	Received Signal Strength Indicator
SDR	=	Software Defined Radio
SNC	=	Sensor Node Core
SNR	=	Sensor Node Radio

SPI	=	Serial Peripheral Interface Bus
SSL	=	Secure Sockets Layer
TCP	=	Transmission Control Protocol
UART	=	Universal Asynchronous Receiver/Transmitter
USB	=	Universal Serial Bus
USRP	=	Universal Software Radio Peripheral
VESNA	=	Versatile platform for Sensor Network Applications

1 Introduction

In this introductory chapter, we provide the motivation behind this thesis and the formulation of the problem addressed. Next, we defined the goals of this thesis, and we provide a brief overview of the thesis and of related work.

1.1 Motivation

Many radio systems have been developed since the discovery that electromagnetic waves can be used to transmit information and many of them are still in operation. All these systems use the same shared resource: the radio spectrum. The radio spectrum is defined as the part of the electromagnetic spectrum with frequencies ranging from 3 Hz to 300 GHz. In conclusion, the total available radio spectrum is considered a limited resource calling for more efficient use.

1.1.1 Usage of the radio spectrum

To allow the numerous radio systems to coexist in the limited radio spectrum, the usage of the radio spectrum has been regulated by regulators at the international and national levels. By regulations, frequency bands are allocated for different systems and the users of a given frequency bands have to get a license for that band. There exist also bands for which no license is required, for example the Industrial, Scientific and Medical (ISM) frequency bands. Currently most of the radio spectrum has been allocated. As an example, the frequency allocation chart of the United Kingdom is shown in Figure 1.1. Allocated spectrum blocks are represented by colored rectangles, while unallocated spectrum is represented with white rectangle. The colored band under the rectangles shows if a given frequency is allocated for civil use (blue), military use (orange) or shared civil and military (black). As it can be observed, only the beginning (3-9 kHz) and the end (275-300 GHz) of the spectrum is not allocated. Multiple rectangles in the same frequency band represent the fact that the given frequency band has been allocated to multiple systems.

It is important to remark that Figure 1.1 refers to allocated radio spectrum, not to the actually used spectrum. Studies and actual measurement campaigns have shown that most of the allocated spectrum is not actively used (Valenta *et al.*, 2010) (Cabric *et al.*, 2004) in time and/or location. Based on activity, a frequency band in a given geographic area can be considered fully utilized, partially used, for example a few times per day, or completely unused. The unused and allocated frequency bands can exist due to various reasons. For example, systems that could use a given frequency band in an area might be inactive or even not installed; in other cases the local geographic conditions might stop the signal from reaching an area.

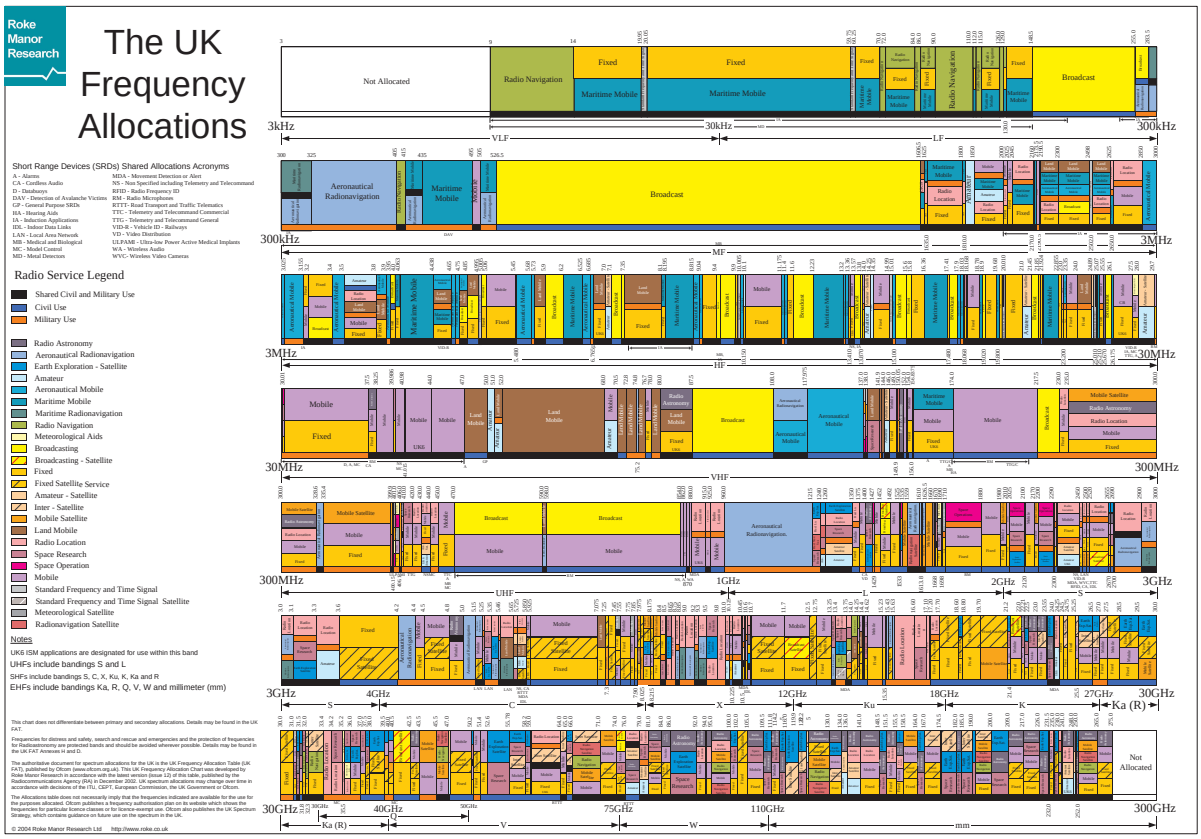


Figure 1.1: Allocation of radio spectrum in the UK (Roke Manor Research, 2004)

1.1.2 Spectrum sensing

Having mostly allocated radio spectrum with many unused or lightly used frequency bands, the following question has arisen: how would it be possible to reuse the allocated but unused frequency bands for other purposes than the ones they were allocated for? One possible answer to this question has been provided by the cognitive radio paradigm (Mitola & Maguire, 1999). In this paradigm, a radio device can be considered cognitive if it has the following properties:

- it observes its environment;
- it has knowledge about radio related concepts and about itself, including its internal state;
- it can change its radio parameters;
- it is able to perform reasoning and make decisions based on its observations and knowledge.

Such radio device should be able to automatically adjust all of its transmitting and receiving parameters in order to minimize interference with other users of the radio spectrum. For example, it could change the frequency it uses, or it could lower its transmitting power.

Federal Communications Commission (FCC) gives the following definition for cognitive radio (FCC, 2003): "A cognitive radio (CR) is a radio that can change its transmitter parameters based on interaction with the environment in which it operates. This interaction may involve active negotiation or communications with other spectrum users and/or passive sensing and decision making within the radio".

As it can be seen, the concept of cognitive radio is very general; it should be able to work either in its own frequency band, or coexist in a radio band with other systems. The topic of coexistence with other radios is particularly interesting, because, as it has been presented before, most of the radio spectrum is allocated, but only a very little portion of it is used at a given moment of time and location.

Hereby we need to distinguish two approaches. The first one called dynamic spectrum access (Zhao & Sadler, 2007) is targeting licensed bands and allows using allocated but unused parts of the radio spectrum on the secondary basis, without causing harmful interference to the existing active systems called primary users of a given frequency band. The second approach is designed for unlicensed bands where no primary users exist and is termed spectrum sharing (Peha, 2005). It assumes that systems using unlicensed bands will establish some sort of collaborative behavior to minimize mutual interference.

For more efficient use of radio spectrum both approaches rely on accurate detection of other systems operating in the same frequency band, in order to avoid or minimize interference. One way of detecting primary users is to scan the radio spectrum for the activity of such systems. In case there is no active primary user, the dynamic spectrum access system can freely use the verified frequency band, whereas in case the primary user is or becomes active, the dynamic spectrum access system has to search for alternative frequency band.

In a spectrum sharing scenario in unlicensed bands the users search for other users of the same frequency band and in case when two users detect each other, they start establishing collaboration in order to minimize their reciprocally induced interference and both continue their operation at a good performance level.

For the detection of other users of a frequency band, many algorithms and methods have been developed. These methods can be divided into two categories: (i) sensing based methods, in which physical radios are used to analyze frequency bands, and based on the analysis, a decision can be taken; and (ii) database driven detection, in which devices or systems look up frequencies and locations in a database and act based on the returned results. The database-driven approach can work well in some licensed bands, where the primary users have well-known location and frequency bands, however in license-free bands the mobility and/or temporal activity of devices makes the database-driven approach unfeasible, hence sensing needs to be performed. In case of sensing, results can be shared between devices, leading to cooperative sensing, or the results can only be processed by the device, which is called local or device-centric sensing. Device-centric sensing has the disadvantage of not being able to detect hidden nodes, while cooperative sensing devices can detect a transmitter which tries to communicate with a hidden node, so cooperative sensing approach is preferred.

Cooperative sensing can be divided in multiple subcategories based on the type of data shared and on the location where decisions about spectrum occupancy are taken. Such subcategories include centralized sensing and distributed sensing. In the first case, the raw spectrum measurements are collected at a central entity, which takes decision about the occupancy of frequency bands, while in the second one, the spectrum measurements are shared between devices, but decisions are taken on each device individually. The latter approach is more robust against device failure and it is better suited for ad-hoc networks, because there is no central entity in the network formed by the sensing devices, however it may suffer from the same problems as distributed routing, i.e. synchronization and convergence related problems.

1.1.3 Improving state of the art

The development of new communication or sensing systems requires theoretical analysis, then creation of a prototype, evaluation of the prototype in the environment as similar as possible

to the targeted one, and improvement of the prototype until it satisfies all of the specified requirements. Finally, the new system to be used in wide audience can be implemented in real operating environment.

In the areas of dynamic spectrum access and spectrum sharing, as parts of broader area of cognitive radio and networking, a lot of theoretical research has been carried out, but most of this work has not been tested and validated since the number of physical prototypes is still very limited. Development of sensing algorithms has been carried out on multiple device categories, ranging from high-end and expensive spectrum analyzers, Universal Software Radio Peripherals (USRPs), Wireless Open-Access Research Platforms (WARPs) or other software defined radio platforms to low-cost and simple wireless sensor nodes such as Telos-B. Also, it is worth noting that it is hard to access data collected from the existing experiments, since most of this data is either not publicly available or not suitable documented and the respective experiments are generally not repeatable. This calls for setting up appropriate testbeds to support carrying out supervised and repeatable experiments, in which the developers can perform realistic evaluation of their algorithms or systems and validation of results obtained by simulation tools.

1.2 Problem formulation

It is generally agreed that spectrum sensing is the core functionality of dynamic spectrum access and spectrum sharing systems, as it is providing context awareness necessary to support decision making about the interference levels between operating systems and availability of radio resource in a given frequency band. As such, it is essential for experimental-driven research in cognitive radio and networking, as well as for validation and further improvement of spectrum sensing algorithms and procedures to set up a suitable testbed. Thus, the purpose of this thesis is to present and explain the design procedures for setting up a sensor network based testbed for spectrum sensing that can be used in various experiments on dynamic spectrum access and spectrum sharing and to demonstrate its implementation.

For the operation of the testbed we assume that either (i) a radio communication experiment is prepared in an ISM radio frequency band or (ii) there exists a region where the radio activity in an ISM band is of interest; in both cases there might exist also an external interfering communication.

The hypothesis is that in both cases mentioned above, by using the testbed, we are able to accurately measure the activity in time in selected frequency band at the locations of the sensing devices in the testbed and thus provide input data for the construction of Radio Environment Maps (REMs) or decision making in the dynamic spectrum access (DSA) and spectrum sharing systems.

1.3 Goals of the thesis

There is a significant body of existing work that addresses problems in dynamic spectrum access and spectrum sharing systems from the theoretical point of view. Many approaches and algorithms have been validated by simulations, but only few were actually implemented and experimented on real devices. Also, there are only a few testbeds available to date, most of them in controlled indoor laboratory environments.

To the best of our knowledge the amount of practical experiments carried out with real radios in outdoor environment is limited. Hence, we are presenting and explaining the design procedures for setting up a sensor network based testbed. We have developed required extension modules, software libraries and application for spectrum sensing using VESNA

platform, calibrated it and used it in the deployment of a testbed. The result of this work is new knowledge that will indirectly improve the state of the art in the field, by allowing realistic evaluation of spectrum sensing algorithms and systems in outdoor environment, but also by providing experimentation facility for future research. The operation of the testbed itself has been evaluated using a reference experiment scenario.

A significant amount of radio communication research targets the Industrial, Scientific and Medical (ISM) bands, one of the reasons being that using these bands does not require specific license. Also, numerous end-user devices are operating in the ISM bands, making it a good target for sensing experiments. Hence, the focus of this thesis is on the ISM bands and hence spectrum sharing approach.

The following main goals have been achieved in this thesis:

- Defining the system architecture for a testbed where multiple VESNA devices can be used for distributed spectrum sensing.
- Developing software libraries and an application that allows a VESNA device equipped with the appropriate radio module to perform spectrum sensing in a selected ISM band.
- Calibration of multiple VESNA devices for spectrum sensing, evaluation of their performance and performing experiments with them.
- Implementation of the functionalities needed to integrate multiple VESNA devices in a testbed and designing the communication system supporting experiments in the testbed.
- Experimental evaluation of the performance of a VESNA-based spectrum sensing testbed.

1.3.1 Contributions

From the goals of the thesis presented above, it can be seen that for the achievement of the goals in a timely manner, a team of people is needed. Indeed, the setup of the spectrum sensing testbed is the achievement of the SensorLab (SensorLab, 2012a) team. This subsection lists the contributions to the testbed of the author of this thesis. For the contributions of other members of the team, please refer to the Acknowledgements chapter.

The main contributions of the author are related to spectrum sensing in ISM bands. These contributions are the following:

- implementation of spectrum sensing on the radios used for sensing and initial definition of sensing profiles for these radios;
- performing of experiments with stand-alone spectrum sensing prototype in two types of scenarios, involving single and multiple sensing locations;
- integration of the developed spectrum sensing prototype in a heterogeneous spectrum sensing system and contribution to the experiment performed with the integrated system;
- initial implementation of the general command parser module and of the spectrum sensing interface employed in sensing in ISM bands on the nodes in the newly established testbed;
- definition and execution of a spectrum sensing experiment in the newly established testbed and processing of the collected data.

The minor contributions are the following:

- contributions to the definition of the custom protocol used in the newly created testbed;
- initial implementation of a radio device debugging tool, used during the development of sensing applications;
- contributions to the implementation of packet transmissions on the radios used in sensing experiments;
- initial implementation of a hardware-accelerated cyclic redundancy check (CRC) calculating software module;
- updates to the build system used to generate the firmware of VESNA nodes.

1.4 Overview of the thesis

This section summarizes the content of each chapter in this thesis.

Chapter 1 presents the motivations of this thesis, the formulation of the problem, the proposed goals and the related work to experimentation in dynamic spectrum access and spectrum sharing.

Chapter 2 summarizes the study of spectrum sensing methods. First it presents and compares the most relevant approaches, then presents possible implementation platforms for spectrum sensing functionalities. Finally, this chapter gives an introduction to several standards that incorporate spectrum sensing functions.

Chapter 3 introduces the VESNA platform chosen for the development of the testbed. VESNA is a modular platform; first the central, core module of this platform is presented. Next the different modules used in experiments and in the testbed are described: the radio modules suitable for sensing, the modules responsible for communications inside the testbed and the Ethernet module used for the communication of the testbed with its associated control and management system. Finally, the software support of the VESNA platform is described.

Chapter 4 details the implementation of the stand-alone sensing prototype demonstrating the viability of low-cost and low-complexity hardware as a spectrum sensing platform. An experiment performed with the prototype is presented in the second part of the chapter.

Chapter 5 presents two experiments performed with the stand-alone spectrum sensing prototype. First the results of an experiment carried out with a heterogeneous spectrum sensing system are presented. In this system, multiple spectrum sensing devices have been integrated, including the VESNA-based sensing prototype. Next, the performances of the prototype are quantified in the second sensing experiment involving the measurement of the received power of a continuous signal at multiple locations.

Chapter 6 first presents the implementation of the spectrum sensing testbed. Starting from the requirements and constraints, this chapter presents the architecture, the performed network planning, the used protocols, the implemented functions of the deployed VESNAs and the control and management part of the testbed.

Chapter 7 details the implementation and execution of an experiment in the testbed. In a predefined scenario, the interaction of two terminals is being observed; in the scenario, one terminal switches to a different frequency when it detects interference from the second terminal, and thus it avoids further interference. Several other nodes in the testbed simultaneously observe the scenario.

Chapter 8 presents the conclusions of this work and identifies future improvements related to spectrum sharing experiments and to the testbed.

1.5 Related work

As it has been mentioned in Section 1.3, there is only a limited number of dynamic spectrum access and spectrum sharing testbeds in operation and most of them are not available for use by other groups of researchers. Furthermore, most of the existing testbeds only consist of a few devices. In order to observe spectrum usage in an area, more than one detector is needed because of the hidden terminal problem, detailed in Chapter 2. However, deploying many spectrum analyzers or USRPs involves significant costs.

In (Yan *et al.*, 2008) a testbed, consisting of five USRP devices, has been used for a dynamic spectrum access experiment. Two USRPs have been programmed to behave as primary users, two as secondary users and one as observer. The primary users have employed single carrier modulation for transmissions, while the secondary users have adapted their orthogonal frequency-division multiplexing (OFDM) transmissions in order to avoid interference with the primary users.

A genetic algorithm is used to optimize a USRP's radio transmission parameters in (Sokolowski *et al.*, 2008). After a few iterations of the genetic algorithm, the radio is able to select appropriate modulation, transmission power and frequency. The goal of the optimization can be the minimization of the bit/packet error rate, the maximization of the throughput, the minimization of the transmission power or a combination of the previous goals. For the combination of the goals, the relative importance of the three basic goals has to be specified.

The coexistence of two cognitive radio architectures in the same frequency band is investigated in (Nolan *et al.*, 2007). Both architectures are based on the USRP platform. During the experiments, one cognitive radio architecture emulated the primary user, while the other the secondary user. Results show that interference-free coexistence is possible, and that the detection of the primary user is very important.

In (Mishra *et al.*, 2005) the BEE2 high-performance cognitive radio platform is presented. This platform, similarly to USRP, uses field programmable gate arrays (FPGAs) for processing. However, instead of directly connecting the radio transceiver modules to the main FPGA board, optical connections are used between the main board and the transceivers. This way it is simpler to set up an experiment with multiple transceivers and the processing power of the FPGAs can be used more efficiently. The same platform is used in (Cabric, 2006) for investigating the performance of the energy detector.

There have been attempts to use low cost commercial devices for cognitive radio, but it proves to be a difficult task, as it has been summarized in (Kim *et al.*, 2007), where energy detection has been performed by using IEEE 802.11 devices: "Energy detection is the only PHY-layer detection scheme supported by 802.11. Even so, a key difficulty in implementing incumbent detection with the Atheros platform is that energy detection mechanism is hidden in the hardware. For this reason, we developed our incumbent detection method based on counting PHY/CRC errors.".

In (Iyer *et al.*, 2011) a distributed sensing platform has been proposed. In this platform a number of spectrum analyzers are placed in a geographic area, each processing commands from a central server. The central server accepts queries asking for signal strength in a given frequency band at a given location. These queries are translated into commands for spectrum analyzers around the location specified in the query, starting the scan of the frequency band specified in the query. The results from more spectrum analyzers are combined by the central server, and returned to the user.

In (Mercier *et al.*, 2010) and (Akan *et al.*, 2009) the usage of wireless sensor network platforms for cognitive radio experiments is proposed. The architecture and plans are presented in the papers, but no large-scale testbed deployment has been presented.

Several members of the FP7 - IP CREW project (Cognitive Radio Experimentation World (CREW) project, 2012) have existing large scale cognitive radio testbed deployments up to several hundreds of sensing agents. This project proposes to establish an open federated test platform, where cognitive radio experiments can be set up remotely on the currently available five testbeds.

2 Spectrum sensing

One of the most important functionalities of a radio communications device that participates in dynamic spectrum access or in spectrum sharing scenarios is the detection of other radio communications devices operating in the same frequency band. Dynamic spectrum access refers to the reuse of a licensed frequency band by secondary users, which use the licensed frequency band for communication only if the primary user of the frequency band is not active (Zhao & Sadler, 2007). In spectrum sharing, multiple users of an unlicensed frequency band try to detect each other and use the frequency band for communications in a way to avoid interference with each other (Peha, 2005). Hence, the devices that should be detected by spectrum sensing can be primary users or other secondary users in a given frequency band at a given location, or other equally threated users in case of unlicensed frequency bands. There are three major types of radio detection techniques:

- *spectrum sensing*, in which a device extracts information from its surroundings by using its radio to analyze the activity on various frequency bands in its region,
- *beacons*, usable when primary users of a given frequency band transmit special signals to indicate their presence,
- *geolocation* or *database-based* approach, in which the information about the availability of a frequency band at a given location is extracted from a database. In this case the device only needs communication with a database, and it does not need to perform accurate measurements.

The most flexible way of detecting other radios is spectrum sensing. However, it must be noted that there are situations when one radio, without cooperating with others, is not able to detect interfering primary users or opportunities to transmit. These situations are presented in Section 2.1. Section 2.2 presents the main spectrum sensing methods, Section 2.3 describes platforms suitable for easy implementation of various sensing methods and Section 2.4 presents standards incorporating spectrum sensing. Finally, Section 2.5 concludes this chapter with a short discussion of the presented topics.

2.1 Hidden terminal and exposed terminal situations

In some cases one device scanning the radio spectrum might perform accurate measurements and consider a frequency band free, even though that frequency band is not free. Such situations are the *hidden transmitter problem*, the *hidden receiver problem* and the *exposed transmitter problem*. These situations are depicted in Figure 2.1. Terminal A tries to communicate with terminal B, and terminal X with terminal Y. A and X are transmitting, B and Y are receiving. Ellipses denote the transmission and reception range of the transmitters A and X.

The hidden transmitter problem is depicted in subfigure (a). Terminal X transmits at the same time as terminal A, because it cannot detect terminal A's transmission. At terminal

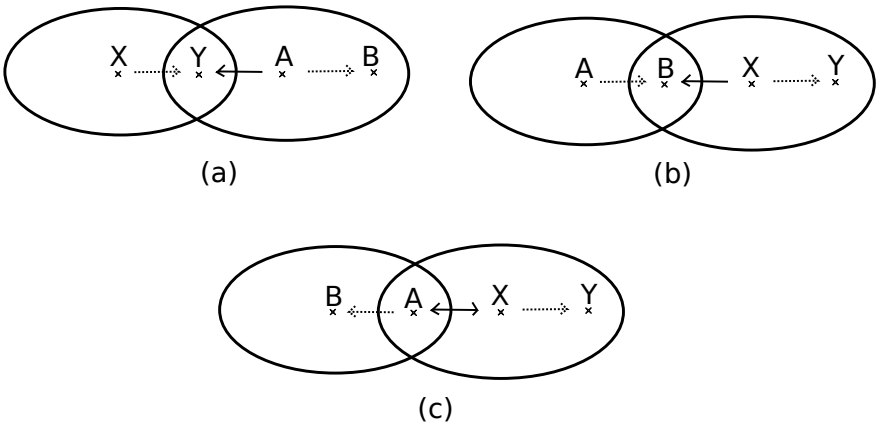


Figure 2.1: Hidden transmitter (a), receiver (b) and exposed transmitter (c) situations

Y collision occurs, hence the reception is unsuccessful. Note that this situation happens because terminal A is not able to detect the receiving terminal Y.

The hidden receiver problem, illustrated in subfigure (b) is very similar to the hidden transmitter problem, but instead of terminal Y, terminal B cannot successfully receive the data addressed to it. In this case terminal A considers that the used frequency band is free, so it tries to send data to terminal B. Terminal B might have been able to detect terminal X, but without coordination between terminal A and B, terminal A is not able to find out that terminal X is active.

The exposed terminal problem is shown in subfigure (c). Terminal X detects the transmission of terminal A, and it does not transmit. Because terminal B is out of the range of terminal X and terminal Y is out of range of terminal of terminal A, a transmission from terminal X would not have caused collision. This situation affects less the total performance of all users because in this case only one transmission opportunity has been "lost", not two actual transmissions, as in the previous situations.

The hidden terminal situations illustrate that for achieving maximum efficiency some coordination is needed between multiple devices. However, this coordination supposes communication between devices. One of the proposed approaches has been the use of a Common Control Channel (CCC) which can be used for local communication between dynamic spectrum access / spectrum sharing devices (Akyildiz *et al.*, 2006).

2.2 Spectrum sensing methods

Having presented the main limitations of spectrum sensing, the following subsections describe various sensing methods. These sensing methods are: *energy detection*, *cyclostationary feature detection*, *matched filter detection*, *eigenvalue based detection* and *cooperative detection methods*. Some of the above mentioned methods, like eigenvalue based detection, or cooperative detection methods can be implemented in more than one way, hence it might be more correct to call some methods method categories. The last two subsections of this section present comparison metrics of the sensing methods and compare the presented methods, respectively.

2.2.1 Energy detection

Energy detection is based on measuring the received power in a frequency band and comparing it with the power of the noise floor. Figure 2.2 illustrates the cases when there is no signal in the frequency band and when there is a signal. The power of the signal (P_S) is

greater than the power of the noise floor (P_N). By comparing the two values, the signal can be detected.

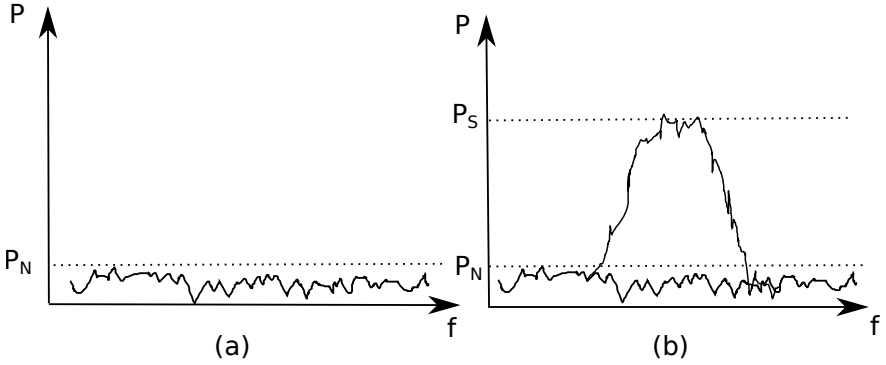


Figure 2.2: Power in a frequency band: (a) only noise is present (b) signal is present

The implementation of the energy detector requires the least computational power. By using this sensing method, any kind of narrowband signal can be detected, but this detection method cannot distinguish between different types of detected signals. Also, energy detection cannot accurately detect spread spectrum signals, because spread spectrum techniques distribute the power of the signal in a large frequency band, so there is very little difference in spectral power between a spread spectrum signal and noise (Cabric *et al.*, 2004).

The main advantage of energy detection is simplicity; its main disadvantage is that this detector needs an accurate value of the noise level. In practice it is difficult to determine the noise level accurately.

2.2.2 Eigenvalue Based Detection

Same as energy detection, eigenvalue based detection (Zeng & Liang, 2008) also does not need any knowledge about the signal to be detected. This method exploits the fact that noise has different statistics than modulated signals. For calculating the relevant statistics, the sample covariance matrix of the received signal is used. The elements in this matrix show how much the samples of the received signal vary with respect to each other. The statistical difference between noise and modulated signals can be found in the eigenvalue distribution of the matrix. Eigenvalues of a matrix A are the scalars λ, for which a nonzero vector v exists such that $Av = \lambda v$.

There are two types of eigenvalue based detectors: maximum-minimum eigenvalue (MME) detector and energy with minimum eigenvalue (EME) detector.

In the case of maximum-minimum eigenvalue detector the decision about the presence of a signal is based on the ratio of the maximum and minimum eigenvalues of the covariance matrix. This ratio is compared to a threshold, and if greater, then a signal is considered to be present.

The energy with minimum eigenvalue detector bases its decision on the ratio between the energy and the minimal eigenvalue of the covariance matrix. This ratio is compared to a threshold value.

Advantages of the eigenvalue based detector include:

- no information is needed about the signal that should be detected; all types of signals can be detected and the same threshold can be used for all of them.
- robustness to multipath propagation, in contrast to coherent detection methods.
- no synchronization is required.

- noise uncertainty problem does not affect this method.

The main disadvantage is the large computational complexity.

2.2.3 Cyclostationary Feature Detection

Cyclostationarity is a property of signals that have periodicity in their statistics, mean and autocorrelation. The periodicity is usually intentionally introduced in the signal to help the receivers in synchronization to the transmitter. Examples of sequences that cause cyclostationary behavior in signals are the synchronization sequences of packet based transmissions, sine wave carriers, pulse trains, spreading or hopping sequences or the cyclic prefix used in OFDM. Cyclostationary feature detection uses the periodicity property of the signals for detection (Yucek & Arslan, 2009) (Cabric *et al.*, 2004).

This method first calculates the spectral correlation function of the received signal, then applies feature detection on the correlation function for decision making. The spectral correlation function shows how much the frequency components of the signal change in time relatively to each other. Different signal types have different spectral correlation functions. In (Cabric *et al.*, 2004), an illustration of this method is presented. The power spectral density (PSD) and spectral correlation function is shown for the same signal, in good (SNR=10dB) and bad (SNR=-20dB) conditions. In the first case the signal can be clearly distinguished from noise in the power spectral density plot, while in the second case the noise dominates. However, the spectral correlation function remains very similar in both cases.

The main advantage of cyclostationary feature detection is the robustness to noise. Its main disadvantages are its large computational complexity and long observation time.

2.2.4 Matched Filter Detection

A matched filter detector tries to identify a known sequence from a radio transmission. Effectively, this detection technique tries to demodulate the received signal and calculates a similarity measure between the received signal and the known signal sequence (Yucek & Arslan, 2009) (Cabric *et al.*, 2004). Such sequences can be synchronization sequences, preambles, pilot signals or spreading sequences. The similarity measure is compared to a threshold value and if greater, then the signal is considered to be present, otherwise it is considered absent.

The similarity measure between the received and the known signal sequences is given by the correlation of the two sequences. The calculation of the similarity measure is equivalent to the filtering of the received signal, where the output of the filter is the similarity measure. From this comes the name matched filter.

The main advantage of this method is the short time needed for performing detection. The main disadvantages of this method are: it needs the parameters of the transmission that should be detected and a signal sequence from that transmission. Also synchronization to the received signal is needed.

2.2.5 Cooperative Sensing

For cooperative detection, more than one radio is needed, each collecting data for the detection process at its own location but not taking a final decision in the detection. The collected data is centralized and a final decision is made (Akyildiz *et al.*, 2006). As mentioned in Section 2.1, the hidden terminal problems cannot be solved by performing detection at only one location. By using cooperative sensing, these problems can be solved.

This method cannot be used alone, because it does not perform any detection. Cooperative sensing centralizes partial detection results, so at least one of the previously presented detection methods, or a new detection method has to be used together with it. It is possible to use more than one detection method, by centralizing the partial results of several detection methods.

This method is able to detect transmitters from outside the detection range of a single cognitive radio, but communication between cognitive radios is needed. The main disadvantage is that the communication for centralizing the partial decisions adds overhead to the radio network.

2.2.6 Metrics for Evaluating Sensing Methods

In order to be able to compare different sensing methods, their performance must be quantified. Because the goal of sensing methods is to detect a signal, the following situations are possible:

Real situation	Output of the sensing method	Notes
Signal not present	Signal not present	Detection
Signal not present	Signal present	False alarm
Signal present	Signal not present	Missed detection
Signal present	Signal present	Detection

The first and the last cases occur at the correct functioning of a sensing method, while false alarm and missed detection are errors. So for the comparison of sensing methods it is of interest to find the occurrence probability of the detection errors. An important parameter on which the error probabilities depend is the signal to noise ratio (SNR) of the signal that should be detected. Signals more differentiated from the noise floor are easier to detect than the ones that are very close to the noise floor.

In an ideal case, a detector would have zero probability of false alarm and missed detection. Theses two parameters are often presented on a Receiver Operating Characteristic (ROC) curve. Such curves are created by plotting the signal detection probability as a function of false alarm probability. An example of such curve is presented in Figure 2.3. The closer the curve to the upper left corner of the plot, the better is the characterized detector.

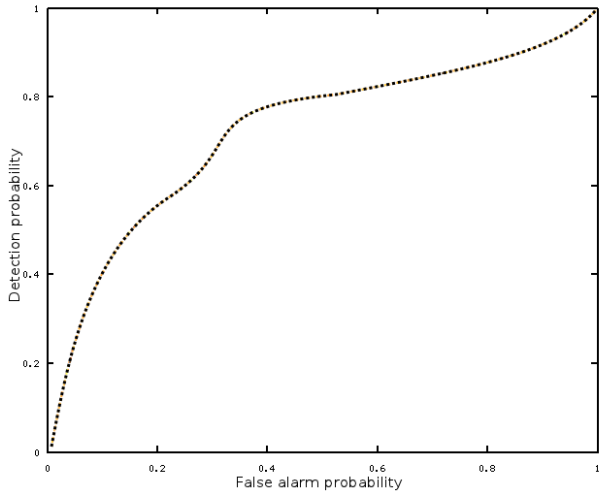


Figure 2.3: Example of a receiver operating characteristic curve

Sensing method	Advantages	Disadvantages
Energy detection	Detects any type of signal; Simple to implement	Needs good estimation of the noise floor; Makes no difference between interference, noise or modulated signal; Weak performance in detecting spread spectrum signals
Eigenvalue based detection	Detects any type of signal; Robustness against multipath propagation; It does not require synchronization;	Cannot differentiate between different types of transmissions; Large computational complexity
Cyclostationary feature detection	Needs less samples than the energy detector; Robustness to noise	Needs knowledge about the signal to be detected; Large computational complexity
Matched filter detection	Needs less signal samples than the energy detector; Fast detection time	Needs the parameters of the transmission to be detected; Needs a part of the signal to be known; Requires synchronization

Table 2.1: Comparison of sensing methods

2.2.7 Qualitative Comparison of Sensing Methods

The above presented sensing methods are compared in Table 2.1. Cooperative sensing is not included in the table because it relies on any of the other methods to perform partial sensing.

As it can be seen, each detector type has advantages and disadvantages. Based on their goals, the methods can be divided in two groups: general detection methods and specialized detection methods.

The first two methods, the energy detection and eigenvalue based detection, are more general, but their performance is not as good as the other methods'. These general methods aim to differentiate any signal from noise.

The last two methods are more specialized for detecting one signal, so in their case the tradeoff between performance and generality is biased towards performance. In this case information is needed about the signal that should be detected. Practically the matched filter detector requires all the information needed to demodulate a given signal.

2.3 Implementation platforms for spectrum sensing

For implementing devices that are capable of performing dynamic spectrum access or spectrum sharing, including the implementation of the sensing methods enumerated above, at least some kind of radio hardware is needed. Radio transceivers usually are not very flexible: they can change their transmitting and receiving frequency within certain limits, set their

transmitting power, and sometimes use different modulations. With respect to information about radio channels, most of the commercial off-the-shelf (COTS) hardware only provides the received signal strength indicator (RSSI) and the link quality indicator (LQI) values. These values can be used as input for an energy detector because the RSSI value is proportional to the received signal power. The other detection methods need samples of the received signal, so they cannot be used when only RSSI and LQI information are available.

The last three signal detection methods presented in this chapter require the processing of radio signal samples in order to function. Because few COTS hardware devices provide access to radio signal samples, a different category of existing radio hardware and software platforms has been targeted with the implementation of new detection methods and dynamic spectrum access/spectrum sharing functions. This category of platforms is called software defined radio, presented in the next subsection.

2.3.1 Software Defined Radio

The goal of the software defined radio (SDR) platform is to be as flexible as possible, in order to allow prototyping of a wide range of functions or change between the used radio standards on run-time. While on most radios signal processing is performed on dedicated hardware because of performance and efficiency reasons, software defined radio platforms employ minimal amount of dedicated hardware and implement the majority of their functions on general purpose processors, in easily replaceable software. Practically the radio frequency part of a transceiver, including radio frequency filtering, local carrier generation, mixing, down- and up-conversion, digitization and signal generation need fixed hardware parts, while the rest of a transceiver chain can be implemented in flexible software.

Software defined radios offer suitable implementation platforms for terminals supporting dynamic spectrum access and spectrum sharing, because these radios have to be as flexible as possible. On SDR platforms different detection, transmission and reception algorithms can be activated from software, without changing the hardware - changing these operating parameters might be necessary for the above mentioned radios. Also, developers can quickly implement and test multiple new signal processing techniques on the same hardware.

One of the popular software defined radio platforms is the Universal Software Radio Peripheral (USRP) (Ettus Research, 2011). Its block diagram is depicted in Figure 2.4.

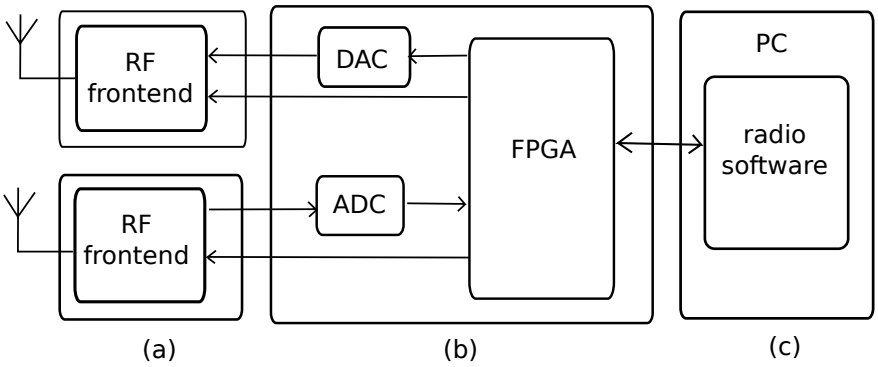


Figure 2.4: Block diagram of the USRP platform

The main module of the USRP, marked with (b) in Figure 2.4, hosts an FPGA and digital to analog and analog to digital converters (DAC and ADC, respectively). The ADC and DAC are connected to separate boards, which contain the radio frequency circuits of the receiver and of the transmitter (a). There are more options for radio frequency boards, each working in a different frequency range. The signal that will be transmitted first is converted to analog signal by the DAC, then fed into the radio-frequency part that shifts the signal

to the desired carrier frequency and finally the resulting signal is transmitted. The signal coming from the receiver is converted to low frequency, in order to allow for the ADC to convert it into a digital signal. This digital signal is processed by the FPGA. The FPGA performs part of the signal processing, and it is communicating with the radio software running on a PC, which performs the rest of the processing.

Other software defined radio platforms were developed in the Open Air Interface project (Open Air Interface project, 2012) and at Rice university (Rice University, 2012). Both platforms have an architecture very similar to the USRP's.

An example of software running on PC that can be used for software defined radio experiments is GNU Radio (GNU Radio project, 2012). This software platform abstracts functionality in blocks having inputs, outputs and parameters. By connecting multiple blocks, a flow-graph representing a given radio architecture can be created. Other notable software platform is Iris (Sutton *et al.*, 2010).

2.4 Standards

Spectrum sensing is not just a functionality of selected hardware and software platforms, because it has been included in industry standards, as presented in this section. Spectrum sensing functionality has been integrated in several modern wireless technologies, such as IEEE 802.11 or Bluetooth. In the case of IEEE 802.11, the IEEE 802.11k (IEEE, 2008) amendment to the standard adds the capability of reporting physical and media access control statistics of the mobile units to the access points (AP). The statistics include the received signal strength and signal quality, the list of available access points and their signal strength. The implementation is not specified in the standard, but the requirements can be fulfilled by employing energy detection. For Bluetooth the added feature is called adaptive frequency hopping (AFH), and it is meant to avoid collisions with IEEE 802.11 systems. A sensing algorithm is used to determine if a given channel is used or not. This algorithm is based on collected statistics, including packet-error rate, BER, received signal strength indicator (RSSI), carrier- to-interference-plus-noise ratio (CINR) (Golmie *et al.*, 2003).

Another standard, IEEE 802.22 (IEEE 802 LAN/MAN Standards Committee, 2012), has been adopted for cognitive radios using the recently reclaimed television bands. It describes a framework is that integrates spectrum sensing algorithms, in order to detect TV signals (Shellhammer, 2008). The standard does not specify any sensing algorithms, just a set of inputs and outputs for the algorithm. Illustration of the framework can be found in Figure 2.5. The input signal type defines what kind of signal should be detected. Possible values

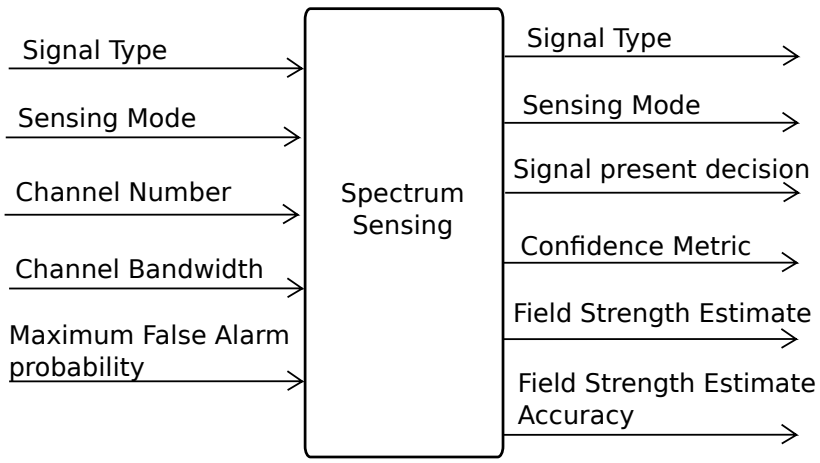


Figure 2.5: Spectrum Sensing Framework in IEEE 802.22 draft

are various TV signals (NTSC, ATSC, etc.), wireless microphone signals, and others. The sensing mode input selects between the three possible modes that can be used. These modes will be detailed together with the outputs corresponding to them in the next paragraph. The channel number and channel bandwidth specify the channel and its bandwidth, respectively. Channel width depends on the country, so it has to be specified. The maximum false alarm probability defines the maximal probability of false detection, when only noise is present in the channel. The outputs signal type and sensing mode are identical to the inputs with the same names.

Depending on the specified sensing mode, outputs are active in the following way:

- sensing mode 0 - only signal present output is active, containing the decision of the spectrum sensing algorithm;
- sensing mode 1 - the signal present output is active, together with the confidence metric of the result (this mode is optional);
- sensing mode 3 - the field strength estimate and field strength estimate accuracy are active (this mode is optional).

The confidence metric shows how reliable is the decision about present signal. For example, if the detected signal is very strong, then the signal present output is true, and the confidence metric is high; but if the detected signal just barely passes the detection level, then the confidence metric is low. The field strength estimation shows the estimated electromagnetic field strength, while the field strength estimation accuracy presents the accuracy of the measurement.

2.5 Discussion

From the point of view of implementation complexity, devices and sensing methods can be divided into two categories:

- simple and general sensing, applying energy detection on measured RSSI values and generally using low-cost COTS hardware
- specialized and high performance sensing, applying advanced detection methods on radio signal samples, and generally using mid- and high-end hardware

As the goal of this thesis is to investigate the usage of multiple radio devices by using low-cost and low-complexity hardware, it can be concluded that the energy detection method suits best the proposed goals. This means that by using COTS hardware and collecting RSSI and LQI values, accurate measurements of signal power values should be carried out. As future work, it might be of interest to investigate the extraction of signal samples from simple radios and to evaluate the limits of signal processing capabilities of low-complexity hardware platforms.

3 VESNA platform

For proof of concept applications or for testbed building, instead of high end or dedicated radio boards also low-cost wireless sensor network platforms can be used with application specific radio modules and chips. For developing the spectrum sensing testbed the multi-purpose and low-complexity VESNA platform (SensorLab, 2012b) has been selected. In the spectrum sensing application many VESNA devices are planned to form a tested, so the low complexity and price of the VESNA platform make it a good fit.

This chapter presents the VESNA platform and the communications and sensing modules used in the rest of this thesis. First an overview of the VESNA platform is given in Section 3.1, next the VESNA platform's features relevant to spectrum sensing are presented in Section 3.2. The communication modules used to connect nodes inside the testbed and to connect the testbed to the management and control systems are presented in Section 3.3 and Section 3.4, respectively. Next, software development tools for the VESNA platform are described in Section 3.5. Finally, Section 3.6 concludes this chapter with a discussion.

3.1 VESNA platform overview

The VESNA platform has been primarily designed for sensor network applications, although its modularity and expandability make it useful for other purposes, too. Due to its modular design and the availability of expansion connectors, it can be easily adapted to many applications. The main module of the node hosting the microcontroller is called the Sensor Node Core module (SNC). The block diagram of this module is depicted in Figure 3.1. VESNA employs a STM32F103Rx microcontroller, having a 32-bit ARM Cortex-M3 processor core. The microcontroller has 64 kB of RAM and 512 kB of flash memory, and supports a variety of interfaces: USB, RS232, UART, IrDA, SPI, I2C, SD/MMC, 12 bit ADC, DAC. The core module of the VESNA platform has two expansion connectors: one designed for radio modules, and one for various purposes. As radio expansion modules, sub-gigahertz and 2.4 GHz-band transceivers are available (e.g. using CC1101 and CC2500 radio chips), but alternatively other modules can be used as add-on boards connected to the radio connector. Examples of various expansion boards are the debugging and programming board, Ethernet to serial converter, WiFi to serial converter, protoboarding modules and the additional power supply module. As it can be seen in Figure 3.1, a VESNA can be powered from a variety of sources: external DC input, USB, battery, solar cell and also the RS-232 connector can be used as power input. This feature is useful for development, because the multitude of power input options allow developers to easily keep a node powered while debugging the firmware under development.

Software development for VESNA can be performed by using standard debug protocol (JTAG) and open-source tools (OpenOCD, CodeSourcery G++ lite and Eclipse).

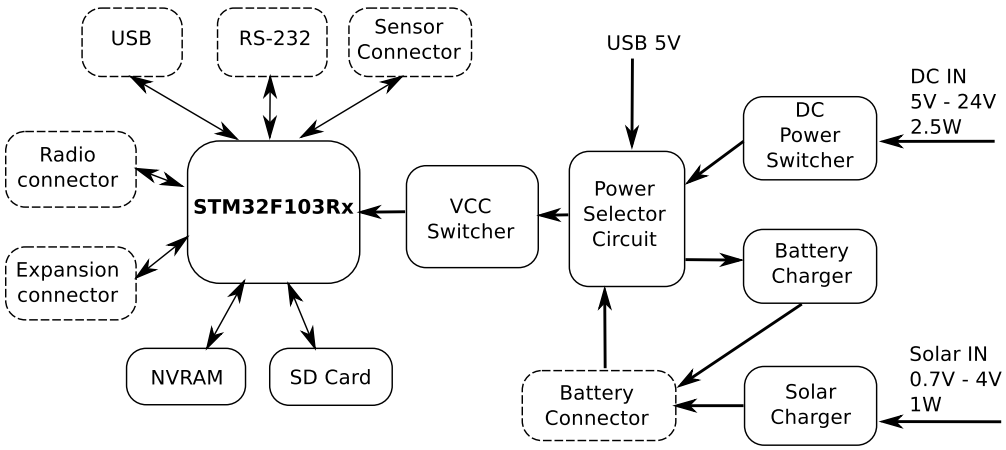


Figure 3.1: Block diagram of the Sensor Node Core

3.2 Spectrum sensing on VESNA

For performing spectrum sensing operations, the following features of VESNA are relevant:

- Exchangeable radio and expansion modules, because multiple types of radios are needed during development, and also because an individual spectrum sensing device will use different radio module than a device integrated into a complex testbed.
- RS-232 interface for outputting data: although in practically all applications an RS-232 connection is needed at least for sending debug messages, for spectrum sensing with a single device the RS-232 connection is used as the main control and data collecting interface. Details of the used communication protocol are presented in Chapter 4.
- SD card interface: the process of collecting RSSI samples from a radio can generate data at a higher rate than the maximum data transmission capability of the network in a testbed, so the collected data might need a storage location before it gets downloaded from the device. Hence, an SD card can be used as a large capacity storage of spectrum sensing measurement results.

Any low-power radio using an SPI or GPIO interface can be used on a radio board. For spectrum sensing applications, the CC1101 (Texas Instruments, 2011a) and CC2500 (Texas Instruments, 2011b) radios have been selected for the 868 MHz and 2.4 GHz ISM bands, respectively. These radios are well suited for sensing applications because in contrast to other popular low-power radios, like CC2420 or AT86RF212, they allow access to many low-level settings of a radio, including the channel bandwidth, modulation type, base frequency and channel spacing. The channel bandwidth can be set from 58 kHz to 812 kHz in discrete steps, the modulation can be Frequency Shift Keying (FSK) with 2 or 4 frequency values, Gaussian Frequency Shift Keying (GFSK), On-Off Keying (OOK)/Amplitude Shift Keying (ASK) or Minimum-Shift Keying (MSK). The base frequency can be set to values in the 433 MHz, 868 MHz and 902 MHz frequency bands; officially values outside these bands are not supported. Channel spacing can be set from 25 kHz to 400 kHz, in discrete steps. These radios also have useful features like unconditional continuous reception, output for data obtained by unconditionally demodulating anything received, and autonomous pseudo-random noise generation. It should be noted that the CC1101 and CC2500 transceivers do not support the IEEE 802.15.4 standard.

For sensing with a standalone VESNA, the CC1101 or CC2500 radios have been located on a radio board attached to the radio connector, depicted in Figure 3.2. In this case the

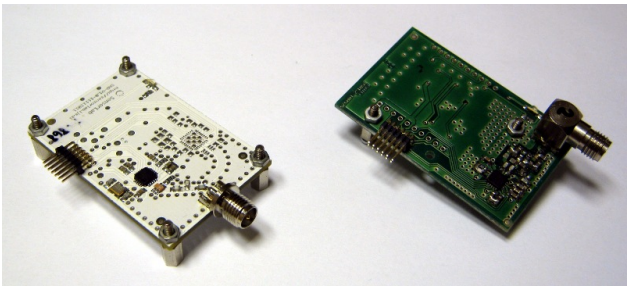


Figure 3.2: CC1101 and CC2500 based radio boards for VESNA

expansion board has been mostly used for debugging the software under development. For the final testbed, the design of the CC radio board has been integrated into the specialized spectrum sensing expansion board, in order to free up the radio connector for the inter-VESNA radio communication module used in the tested configuration. The spectrum sensing expansion board, depicted in Figure 3.3, hosts the following radios:

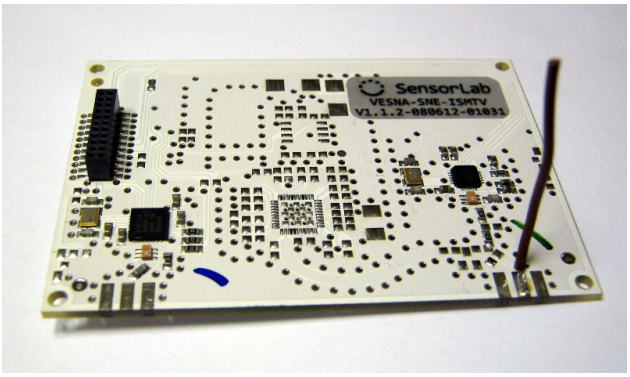


Figure 3.3: Spectrum sensing expansion board for VESNA, equipped with CC2500 radio

- One of the following devices:
  - CC1101 transceiver, operating in the 868 MHz ISM band
  - CC2500 transceiver, operating in the 2.4 GHz ISM band
  - TDA18219 TV tuner from NXP Semiconductors, operating in UHF TV bands
- AT86RF212 transceiver from Atmel, operating in the 868 MHz ISM band.

Because this expansion board hosts two radios independent from the management network of the testbed, various sensing scenarios can be developed by using the available features of the hardware. Only two radios are installed on the sensing board, because the installation of more than two sensing and one communication antennas on a VESNA would add a lot of complexity.

3.3 Modules used for communications within of an integrated testbed

For the operation of a testbed consisting of numerous devices, it is necessary to have communication between the devices and the users of the testbed. In case the testbed is located at a different location than its users, then a two-part communication channel is needed:

- Communication from the users to the entry point of the testbed (gateway); any point-to-point communication channel can be used;
- Communication from the entry point of the testbed (gateway) to all of the devices of the testbed; this type of communication supposes star or mesh topology with possibility of multi-hop.

This section presents the hardware used for implementing the communication network inside the testbed. This network is used for dispatching commands from the users of the testbed to the target devices, for collecting experiment results and for the management of the testbed. An important function carried out through the management network is the remote update of the software running on the devices of the testbed. This function is not discussed in this thesis.

Due to the fact that the testbed has been planned to operate in an outdoor environment, with distances of 25 to 50 meters between devices, the cost and complexity of employing a wired network are too high, hence a radio-based management network has been planned for the testbed. The goal of the management network is to form a robust multi-hop network with all of the devices in the testbed and offer reliable data transfer between any two nodes. For simplifying the development, stand-alone modules have been chosen which carry out the network management independently and offer a high-level interface towards its users. Because most of the experiments in the testbed are expected to be targeting the 2.4 GHz ISM band or TV-UHF band, for avoiding interference between the management network and the radio experiments, the 868 MHz ISM band has been chosen for the operation of the management network.

The ATZB-900-B0 ZigBit™ modules (Atmel, 2012a) from Atmel have been selected for use in the management network. These modules implement the ZigBee specification with proprietary extensions. When the modules are started, they form a mesh network with a central coordinator, zero or more routers and zero or more end devices. This device is depicted in Figure 3.4, while a VESNA equipped with this communication board can be observed in Figure 3.5. The modules receive AT commands through their universal asynchronous

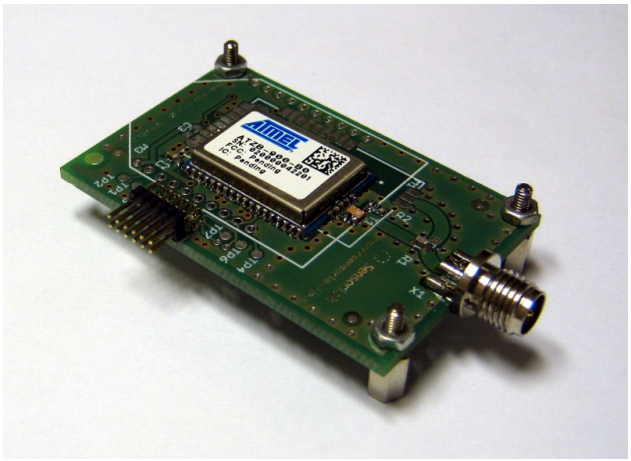


Figure 3.4: ZigBit based radio board for VESNA

receiver/transmitter (UART) interface, and send data through their radio interface. Data received from radio is transferred to the same UART interface. On VESNAs, the ZigBit modules are connected to the central microcontroller. In the testbed, the coordinator of the ZigBit network is the VESNA node with Internet access. After some initial testing, it has been found that modules configured as end devices have much smaller throughput than modules configured as routers, presumably because of radio duty-cycle limitation of the end

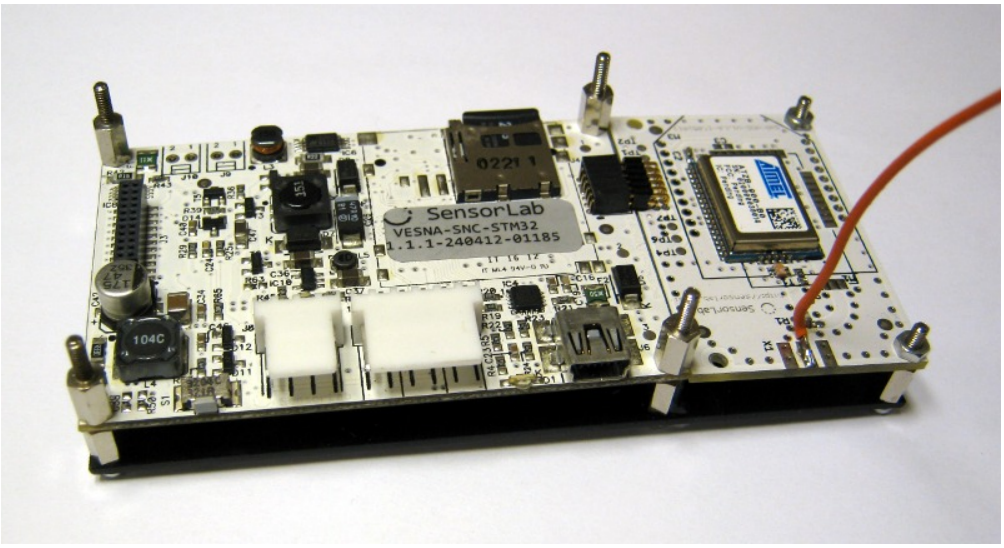


Figure 3.5: Picture of a VESNA equipped with a ZigBit based radio board

devices. Hence, all the modules used in the testbed have been configured as routers, except for the two coordinators/gateways of the testbed. The maximum amount of data that can be transmitted in a ZigBit frame is 94 bytes when security is disabled and 64 bytes when security is enabled (Atmel, 2012b). Enabling security means that the traffic in the network is encrypted. This feature is useful for preventing various types of attacks on the testbed. As no application level security has been planned, enabling security in the ZigBit network is necessary for preventing packet injection in the network.

3.4 Modules used to connect a remote testbed with its control systems

As previously mentioned, two types of connectivity are necessary for an outdoor testbed: connectivity inside the testbed and connectivity between the gateway of the testbed and the users or management and control system of the testbed. The hardware chosen for the implementation of the first type of connectivity has been presented in the previous section, while this section presents the hardware used to connect the testbed to its users.

For supporting point-to-point connectivity in a generic manner, an Ethernet interface has been used for providing connectivity. This Ethernet interface is implemented on an expansion board with the Digi Connect ME® 9210 Ethernet to serial line converting module (Digi International Inc., 2012). This module features an Ethernet interface and a universal asynchronous receiver/transmitter (UART) interface; it can be configured through both of its interfaces. The main function of the module is to transfer data between its UART interface and transmission control protocol (TCP) connections. The module is shown in Figure 3.6. The following features of the module are used in the spectrum sensing testbed:

- Connecting to an IPv4 network through the Ethernet interface; the network settings are obtained by using the dynamic host configuration protocol (DHCP).
- Establishing a secure sockets layer (SSL) connection to a predefined host and port.
- Acting as a transparent tunnel between the UART interface and the SSL connection. Hence, data sent by the microcontroller on the sensor node core (SNC) board to Digi Connect ME module through UART, appears as data in the SSL connection; when

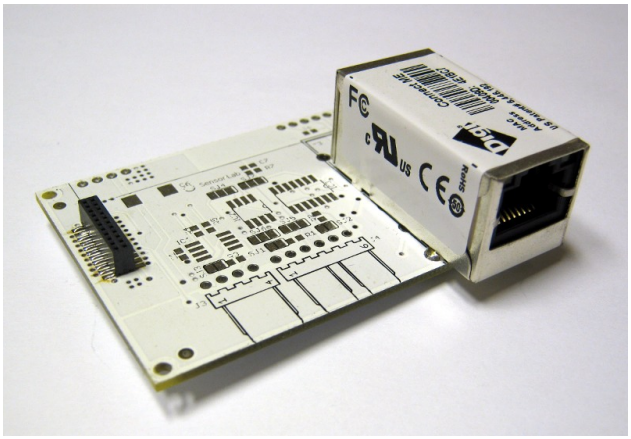


Figure 3.6: Ethernet module expansion board for VESNA

the other peer in the SSL connection sends data towards the module, the same data is forwarded to the microcontroller by the UART lines.

The SSL connection is used for securing the testbed against unauthorized access, because the testbed has been planned to be accessible through the Internet. With unencrypted connection the testbed would be exposed to various attacks.

3.5 Software development for the VESNA platform

This section presents the available tools and libraries for the creation of applications using the previously listed functions and devices.

One of the strong points of the VESNA platform is the openness and wide availability of the necessary software development tools. All of the required development tools are available under open-source license for multiple operating systems, including Windows and all Linux distributions. The required tools are:

- GNU[1] compiler collection (GCC) based cross-compiler toolchain for Advanced RISC Machine (ARM[2]) devices; a binary distribution is the CodeSourcery Lite toolchain;
- OpenOCD on-chip debugger (OCD) used for uploading and debugging firmware on the microcontroller of VESNA;
- GNU make utility used for building applications;
- Cygwin Unix-like environment required only for Windows operating system; it is needed by the build-system used in the software libraries.

The following optional applications are also available under open-source license:

- Eclipse integrated development environment (IDE) with EGit and Zylin Embedded CDT[3] plugins, used for most of the development-related tasks, including editing source

[1] GNU stands for GNU is Not Unix. In this context we are referring to the free software project called GNU Project, which develops numerous popular development tools, including a compiler collection and a debugger.

[2] It is debatable if ARM is an acronym or not. The designer of ARM processors had been named as Advanced RISC Machines Ltd before 1998, but since then it has changed its name to ARM Holdings.

[3] In Eclipse's terminology, CDT stands for C development tooling. In this particular case, Zylin Embedded CDT is the proper noun name of a specific Eclipse plugin.

files, compiling applications, uploading applications to VESNA, debugging and working with version control systems. Other IDEs or editors can be used for software development, depending on personal preferences.

- CuteCom, moserial, Termite or other serial terminal, used to interact with a VESNA through RS-232 connection.

For Windows and for Linux distributions, the VESNA development manual (SensorLab, 2012c) presents the setup of the development environment in detail.

3.5.1 Debugging

For uploading firmware and debugging, the ARM-USB-OCD debugger is used by connecting it with a joint test action group (JTAG) cable to the debug board attached to a VESNA. The ARM-USB-OCD debugger is connected with USB connection to a PC and it offers JTAG, RS-232 and power supply interfaces. On most Linux distributions, this device works out-of-the box, while on Windows, it requires driver installation.

Figure 3.7 shows a screenshot of the Eclipse IDE running on Linux. The empty space in the middle is reserved for opened source code editors. On the left side of the window the directory structure of the VESNA drivers project can be observed, on the right side the Make Targets View, with the VESNA drivers demo application selected. In Make Targets

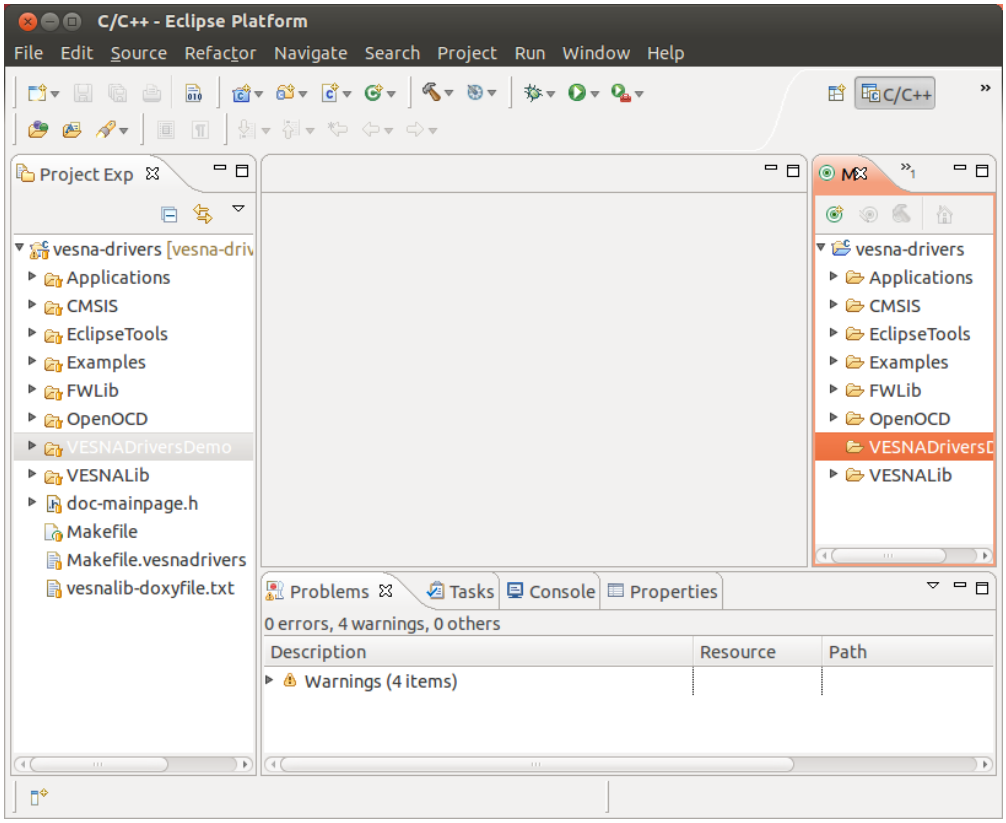


Figure 3.7: Make targets view

View the building process of applications can be started. The bottom of the main window shows Eclipse's diagnostics about the VESNA drivers project.

The debugging features of Eclipse IDE are presented in Figure 3.8. The marked elements in the figure are as follows:

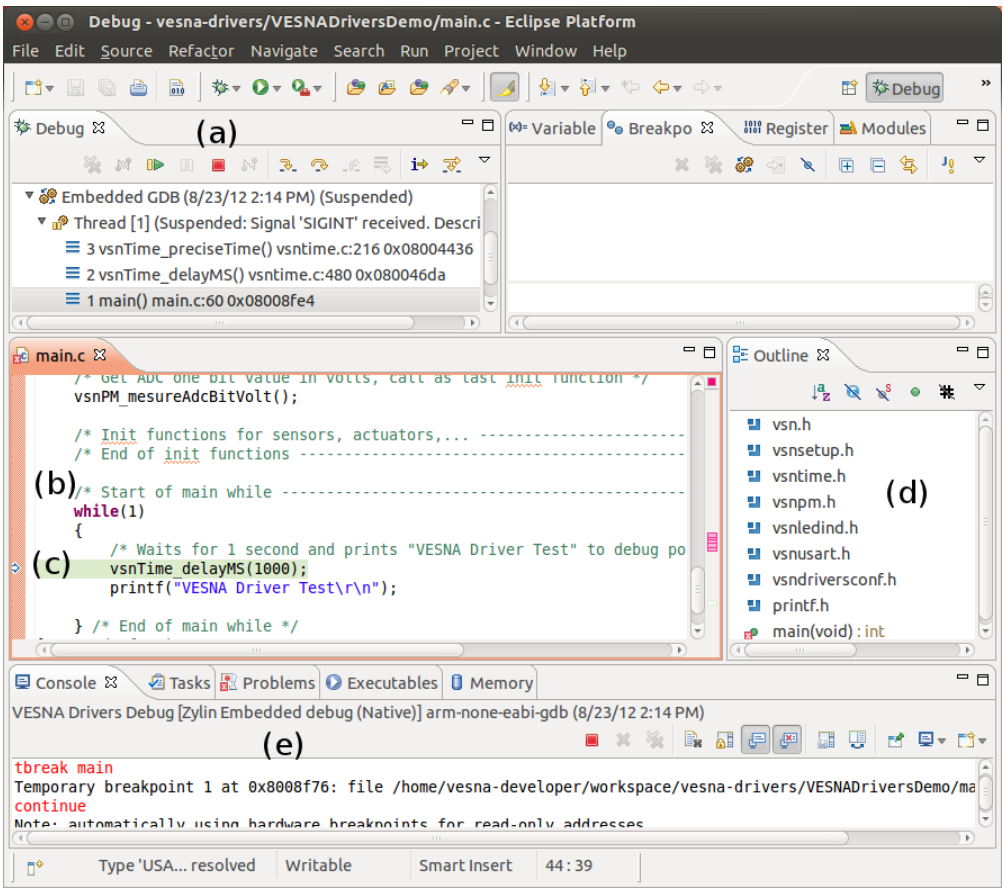


Figure 3.8: Debug perspective

(a) Program flow control and the running processes. Note that the list also contains OpenOCD, the debug server, and the program being debugged, including the stack(s). The OpenOCD service needs to be started before any debugging can be performed.

(b) Source code view

(c) The line being executed.

(d) The outline of the current source code. In the upper region there are the lists of breakpoints, register values, variables.

(e) Debugging console. There is also the Memory view at the right part of the bottom.

At the end of debugging, the executed program and also the debug server should be stopped. As it can be seen, Eclipse IDE allows rapid program development and efficient debugging. This way the development effort for new applications can be greatly reduced.

3.5.2 Libraries

An extensive set of software libraries has been developed for VESNA, consisting of drivers for available hardware and common building blocks of applications. Most of the building blocks have associated demo applications showing their usage. The VESNA libraries are based on STMicroelectronic libraries provided for the STM32F103Rx microcontrollers; the latter ones providing abstractions to the microcontroller's low level functionalities.

For the CC1101 and CC2500 modules the low-level driver has been mostly developed before the start of this work; this driver has allowed reading and writing to the radio's

registers, but it did not implement other high-level functionality. While working on spectrum sensing, three high-level drivers have been created for the CC1101 and CC2500 radios:

- one oriented towards testing packet transmissions,
- one for using these radios for spectrum sensing,
- one for generating signals with these radios.

While developing these high-level drivers the low-level interface has been progressively improved. Finally the sensing and signal generation drivers have been plugged into more generic sensing and signal generation interfaces. These general interfaces are providing unified access to the sensing and transmitting capabilities of VESNAs and allow the simple use of multiple signal generating or sensing radios.

The ZigBit and Ethernet modules are controlled through UART interfaces; the UART functionality has been available in VESNA libraries, while the drivers for the two modules have been developed by other members of SensorLab.

3.6 Discussion

Before using the presented VESNA platform for spectrum sensing, we carried out some testing of the Ethernet expansion board. While testing the Ethernet module, the following undocumented behavior has been observed:

- when the module reestablishes its SSL connection, it sometimes sends a 0x01 byte on its serial line towards the microcontroller, while some other times it sends a 0x00 byte. This byte can create confusion for higher layers in the communication stack.
- Other unexpected limitation of the Ethernet module is the incompatibility of hardware flow-control implementation of the microcontroller and the Ethernet module. For an unclear reason, when hardware flow control is enabled on the USART interfaces of both the microcontroller and the Ethernet module, duplicated characters appear in the data. This behavior has been worked around by not using flow control on the mentioned USART connection.

The VESNA libraries have been developed inside SensorLab, so it is possible for SensorLab to license these libraries under any license it chooses. There have been plans for releasing the libraries as open-source software and publishing technical documentation about VESNA, so a community can be formed around this platform. At the time of writing, tight deadlines and the unclear legal status of STMicroelectronics' libraries are slowing down this process. Possibly in the future a part of the libraries will be open-sourced and the VESNA platform may become popular development platform with an active community.

4 Stand-alone spectrum sensing

The first step in developing a spectrum sensing testbed is to experiment with one device and develop basic functionalities on it. The main goal of the planned testbed is spectrum sensing in unlicensed frequency bands, so the first step in the development of the testbed is to implement spectrum sensing functionality with one VESNA device.

As it has been discussed in Chapter 2, the most suitable detection method for low-end devices is the signal power based energy detection. As input for such detectors many radios can provide RSSI readings as estimations of the current signal power received. Because the CC1101 and CC2500 radios are highly configurable, they have been selected for collecting radio spectrum information.

This chapter presents the steps needed to carry out spectrum sensing experiments with the radios planned to be used in the testbed. First, in Section 4.1 a framework is presented which can be used for collecting information about radio spectrum usage, the spectrum sensing implementation is detailed and a spectrum sensing experiment in the 2.4 GHz ISM band is described to demonstrate the capabilities of the framework. Section 4.2 presents the calibration of the radios and its results. Finally Section 4.3 provides discussion about this chapter.

4.1 Implementation

For implementing spectrum sensing functionality, first a system architecture has been designed. After the architecture has been defined, all parts of the system have been developed, and finally spectrum sensing experiments have been carried out. The next subsection presents the system architecture, while subsequent subsections describe the implementation of components.

4.1.1 Sensing system overview

The proposed spectrum sensing architecture is depicted in Figure 4.1. It consists of two parts: the sensing and data collection and control part. For the communication between the two parts of the system an RS232 connection is used.

The sensing part performs the radio spectrum measurement, applies optional preprocessing of the collected data and sends the data to the data collection and control part of the framework. It is capable of changing the sensing parameters by applying different sensing profiles; switching between these profiles is triggered by commands received on its RS232 interface. The sensing part has been implemented on the VESNA platform by developing special software application for the device. The software application running on VESNA is presented in detail in Section 4.1.2.

The data collection and control part performs the control of the sensing part and it stores and processes the measurement data. It is implemented by software modules running on a personal computer (PC). These modules receive the collected data via the RS232 connection,

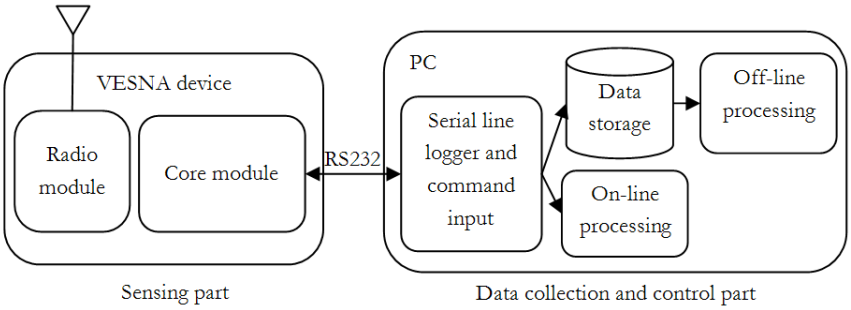


Figure 4.1: Overview of the VESNA based spectrum sensing framework

process and display it, and also store it for later use. The real-time display of the data allows on-site inspection of measurements, while the stored data can be converted to formats that allow importing of the measurement data into various data processing tools, for instance for building the Radio Environmental Maps (REMs) (Atanasovski *et al.*, 2011) or to support collaborative sensing and spectrum sharing algorithms. The data collection and control part also provides the user interface for selecting the active sensing profile for the sensing part. This way, radio spectrum measurements with different parameters can be easily carried out.

4.1.2 Software support for VESNA based spectrum sensing

In order to use VESNA platform for spectrum sensing, an application has been developed, that sets up different sensing profiles for the radio located on the SNR module, collects the measurements from the radio, processes the raw measurement data and sends the measurement results to the data collection and control part of the framework. This subsection details the implementation of the spectrum sensing application where we distinguish between sensing and communication parts.

The developed application works with both the CC1101 and CC2500 radios. These radios show a very large degree of similarity between themselves, including pin- and software compatibility. Practically the only significant difference between the two radios is the frequency band in which they operate. Hence, the two radios will be referred to as the CC radios.

Sensing part of application

In order to obtain valid RSSI values from the CC radios, the radios have to be properly configured. The configuration is performed by setting up the radios' registers with appropriate values. Texas Instruments provides a tool called Smart RF Studio (Texas Instruments, 2012a) for defining the values of various configuration registers of the radios based on the desired transmission and reception settings. Figure 4.2 shows a screenshot of this tool while it is used to configure a CC1101 radio.

On the top left part of the window the available predefined settings for the radio are listed, in the middle there are controls for manually setting up most the radio's parameters. At the bottom left are controls for selecting the use case of the radio, which include continuous transmission and reception, packet transmission and reception and packet error rate (PER) testing. These controls work for evaluation boards from Texas Instruments. At the right part of the window the generated register settings can be inspected, modified and exported. For usage with VESNA, register settings have been created for predefined and testbed transmission and reception profiles and exported for both CC radios, for subsequent use with the spectrum sensing application.

Sensing profiles contain register settings for the radio and human-readable specification

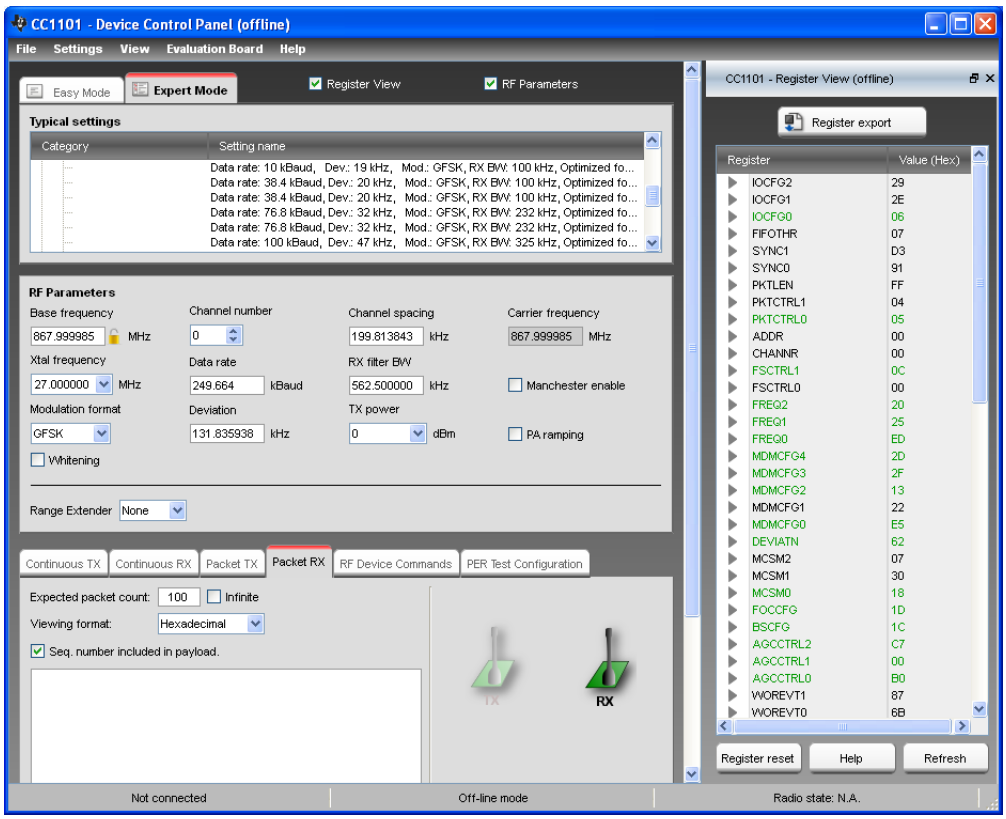


Figure 4.2: Smart RF Studio from Texas Instruments

of the settings, including the frequency band in which the sensing should be performed, the channel bandwidth on which the radio should operate, the list of frequencies on which the sensing band should be applied and the number of samples that should be averaged in order to obtain one data point. By using different sensing profiles, trade-offs can be made between the parameters of the sensing, such as resolution, accuracy, bandwidth, speed of sensing and minimal signal level that can be detected. The selection of sensing profiles is controlled by the data collection and control part of the sensing framework, by sending commands to the sensing part. These commands are received on the RS232 interface on the SNC, and processed in software.

The most basic steps used for getting RSSI data from a CC radio are the presented in Algorithm 1.

Algorithm 1: *Getting RSSI samples from one channel of a CC radio.* Short overview of setting up a CC radio and reading RSSI samples in an infinite loop.

Reset the CC radio
Load the profile settings generated by Smart RF Studio into the CC radio
Set the radio into receiving (RX) mode
while true do
    Read the RSSI register of the CC radio
    Convert the register value to signal power in dBm
end

The steps listed above get one RSSI value from the radio for one channel. Both CC radios provides RSSI readings with 0.5 dB resolution; the conversion procedure from register value to dBm is presented in Algorithm 2, as described in the datasheets of the CC radios

(Texas Instruments, 2011a) (Texas Instruments, 2011b).

Algorithm 2: *Converting the value of the RSSI from a CC radio to dBm.* Input value is an 8 bit integer, output value is expressed in dBm.

```
input: rssi_reg: the RSSI register value
constant: device_offset: RSSI offset of the particular CC radio  device, expressed in dBm
result = -70 + device_offset
if rssi_reg ≥ 128 then
                result = result + (rssi_reg - 256)/2
        else
                result = result + rssi_reg/2
fi
return result
```

One channel of the CC radios can be at most 840 kHz wide. For sensing in a wider frequency band several adjacent channels can be used. The radio can be repeatedly set to one of the multiple channels, get RSSI samples and then move to the next channel. Although only a part of the frequency band on interest can be observed instantly with this method, long-enough transmission bursts or periodic packet transmissions can be easily observed. For sensing on multiple channels, the necessary steps are presented in Algorithm 3.

Algorithm 3: *Getting RSSI samples from multiple channels of a CC radio.* Short overview of setting up a CC radio and reading RSSI samples from multiple radio channels in an infinite loop.

```
Reset the CC radio
Load the profile settings generated by Smart RF Studio into the CC radio
Set the radio into receiving (RX) mode
while true do
    for each channel do
        Set the radio to idle mode, so the analog part of the radio is turned off
        Set the channel register to the current channel
        Set the radio into receiving (RX) mode
        Wait until the radio's automatic gain control (AGC) stabilizes
        Read the RSSI register of the CC radio
        Convert the register value to signal power in dBm
    end
end
```

Getting RSSI values from multiple channels can be achieved by changing the current channel of a CC radio. Both CC radios support 255 channels. Before changing the channel value, the radio's analog part has to be turned off by setting the radio's state to idle mode; otherwise undefined behavior occurs and is most cases the radio stops responding to commands. After setting up the current channel value, the radio can be put back into receiving mode. When starting to receive, first the CC radio's automatic gain control system sets up the internal amplifiers of the radio to correct values and during this operation the RSSI values given by the radio are incorrect. This is why before reading the RSSI register a certain period of waiting is necessary.

The accuracy of the RSSI readings can be increased by reducing the noise in the reading with averaging. When reading RSSI values for a given channel, multiple values can be

read, then the average of the obtained values can be used as a single data point. It is possible to calculate other statistics of the obtained RSSI values, for example minimum and maximum values or standard deviation. This sensing algorithm performing averating is presented in Algorithm 4. Note that dBm values need to be converted into Watts, before their average value can be calculated. The general formulas used in this calculation are $x[W] = 1mW \times 10^{X[dBm]/10}$ and $X[dBm] = 10 \times log_{10}(x[W]/1mW)$.

Algorithm 4: *Getting averaged RSSI samples from multiple channels of a CC radio.* Short overview of setting up a CC radio and reading averaged RSSI samples from multiple radio channels in an infinite loop.

```
Reset the CC radio
Load the profile settings generated by Smart RF Studio into the CC radio
Set the radio into receiving (RX) mode
while true do
    for each channel do
        Set the radio to idle mode, so the analog part of the radio is turned off
        Set the channel register to the current channel
        Set the radio into receiving (RX) mode
        Wait until the radio's automatic gain control (AGC) stabilizes
        for all samples to be averaged do
            Read the RSSI register of the CC radio
            Convert the register value to signal power in W
        end
        Output the average value of the above collected samples
    end
```

Communication part of application

The data collected by the CC radios is sent to the data collection and control part organized in lines; each line describes the power level detected at each of the frequencies specified in the active sensing profile. Besides measurement data, lines contain a timestamp of the measurement and markers for line start and line end used for corruption detection at the end of a sensing activity. If sensing is interrupted, the successfully transmitted lines can be easily recovered based on the line start and line end markers. The generic form of one output line is as follows:

```
TS timestamp DS value1 value2 ... valueN DE
```

Example of data sent from the sensing part to the data collection and control part as follows:

```
TS 110 DS -85.45 -65.45 -76.46 -83.48 DE
TS 163 DS -75.54 -79.63 -84.36 -81.56 DE
```

This data is processed by the data collection an control part presented in the next subsection.

4.1.3 Data collection and control part

The data collection and control part of the framework (i) allows the user to select the active sensing profile, (ii) displays the spectrum measurement data in real time, in order to allow

the monitoring of the measurements and (iii) stores the spectrum data on the PC, in order to allow off-line processing.

The selection of the current sensing profile is performed by manually typing the command that selects a given sensing profile. The list of available sensing profiles and a short description is provided by a "help" command. The real-time data monitoring interface is shown in Figure 4.3. It presents the received signal power versus time, in the frequency band defined by the active sensing profile. The same data is saved on the PC, in order to be processed offline. The format of the saved data is identical to the data transmitted on the RS232 connection; however, it is guaranteed that the saved data contains only valid lines. In order to import the data into MATLAB or Octave, scripts have been developed, which load the saved files and store the available data in data structures specific to the programs mentioned. After this importing procedure, the data can be freely processed.

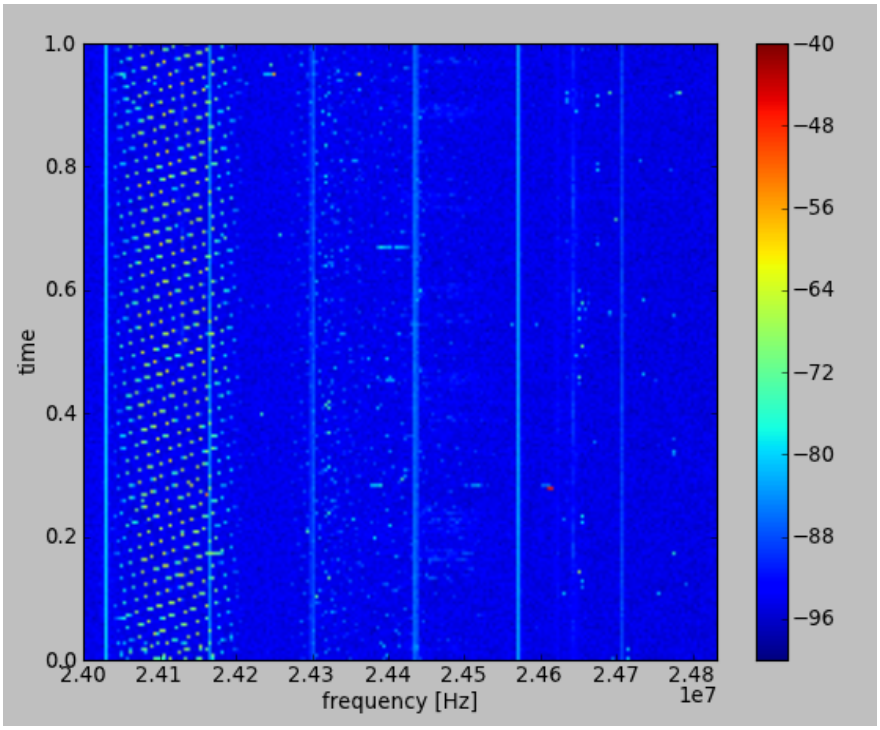


Figure 4.3: Real-time data monitoring interface for the system

4.1.4 Initial experiments

The spectrum sensing framework presented in this chapter has been used to analyse the radio spectrum usage in the 2.4 GHz ISM band. Figure 4.3 presents the measurement results obtained by the real-time data analyzer. The frequency bands used by Wi-Fi (2400–2483 MHz) have been scanned for several minutes. The periodic vertical lines on the plot are the result of a known limitation of the CC2500 radio. This background signal is likely caused by internal interference inside the radio device, at frequencies which are integer multiples of the half of the clock frequency of the radio. The patterns appearing on the left side of the plot indicate the Wi-Fi activity. Based on the plot, it can be concluded that the Wi-Fi devices have been transmitting on the center frequency of 2412 MHz, which corresponds to the Wi-Fi channel 1. Also, a weak signal can be observed around 2431 MHz, corresponding to Wi-Fi channel 6.

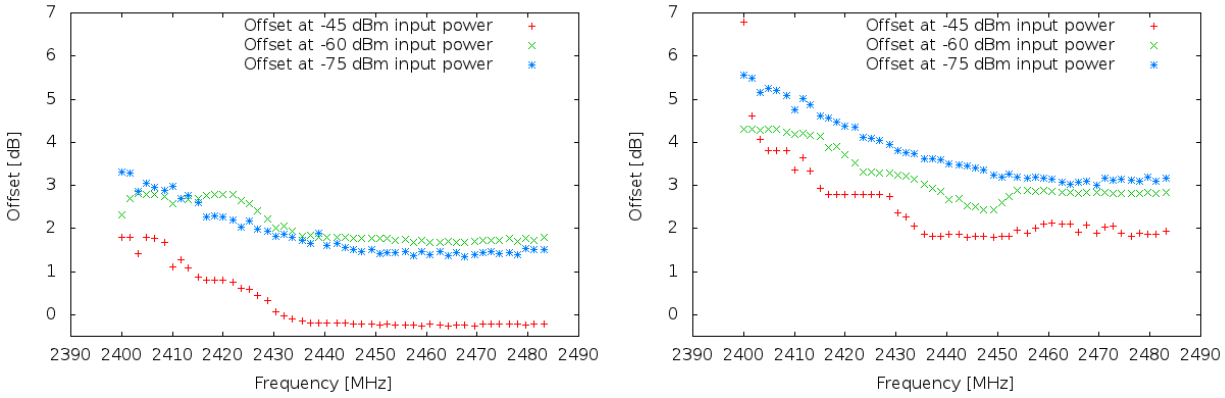
As it can be seen from the results, the initial experiments quoted show that the sensing system is functioning as planned. The next step has been the calibration of sensing devices,

presented in the following section.

4.2 Calibration

For accurate measurements, the CC radios used for spectrum sensing should be calibrated. Based on the datasheet (Texas Instruments, 2011a) (Texas Instruments, 2011b), the linearity of the radios is good, just they might have a constant offset added to the reported values which needs to be determined in controlled environment.

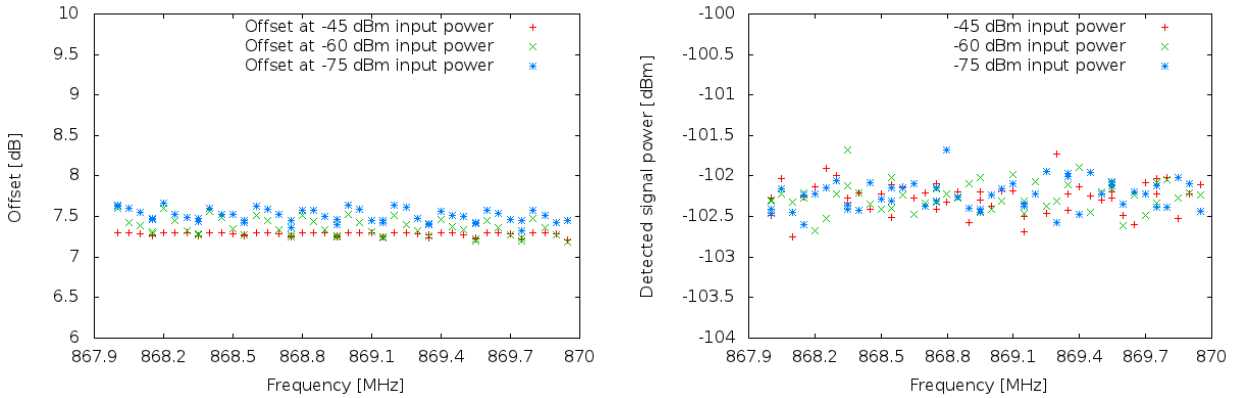
For calibration the CC radios mounted on VESNAs have been connected with a coaxial cable to a signal generator. As an example we used four VESNA platforms, two equipped with CC2500 radios and two equipped with CC1101 radios. Signals have been generated with different power levels and different bandwidths. The signal power values measured by VESNAs have been recorded. Example results are depicted in Figures 4.4 and 4.5.



(a) CC2500 radio device number 1

(b) CC2500 radio device number 2

Figure 4.4: Variation of error in detected power with frequency and input power level for CC2500 radios



(a) CC1101 radio device number 3

(b) CC1101 radio device number 4

Figure 4.5: Variation of error in detected power with frequency and input power level for CC1101 radios

As it can be observed, the offset of the CC2500 radio is in the order of a few decibels; the variation of the detected signal level with the frequency is approximately 2 dB at constant signal level, while the variation of the measured signal level with the input level is also approximately 2 dB. With proper calibration, these offsets can be easily compensated. The

CC1101 radios produce results similar to the ones depicted in Figure 4.5a. Note that the total variation of the offset is up to 0.5 dB only. Because of this small variation, the output of the CC1101 can be considered practically independent of the signal power level and the frequency. At lower power levels the variation of the offset increases, suggesting that the noise of the receiver affects the offset value more with the decrease of the input power level.

Figure 4.5b presents the results obtained from a malfunctioning CC1101 radio. The values on the OY axis are not the offset of the device, but the absolute value of the measurement results. As it can be seen, all of the values are around -102 dBm regardless of the input power level. This means that the receiver is not able to detect any signal, it is just showing its own noise. This radio board has been marked as not functioning, and the CC1101 needs be replaced.

Based on the results of the calibration measurements, the offset of each device has been determined. These offset values are to be taken into account when data from experiments is processed.

4.3 Discussion

In this chapter a VESNA platform based spectrum sensing framework has been presented which is capable of collecting information about radio spectrum usage. This information could be used in optimization of radio networks, implementation of dynamic spectrum access or as a sensing component for cognitive radio systems. Related to the programming of the CC radios and using them for spectrum sensing, several remarks are presented in the rest of this section.

Texas Instruments provides evaluation boards for their CC radios. This boards can be used together with their Smart RF software tool, so developers can quickly test the CC radios. For the development targeting VESNA, the evaluation boards have not been very useful because they use a different clock frequency (26 MHz) than VESNA's radio boards (27 MHz), so the settings tested with the evaluation modules are not compatible with the used radio boards of VESNA. Hence, the evaluation modules have not been used for testing radio settings for VESNA and the time needed to create correctly functioning profile settings for the CC radios has been a longer process than expected.

Based on the datasheet of CC radios an implementation of automatic profile settings generation has been tried, but for multiple registers Texas Instruments recommends to use their Smart RF tool for generating valid profile settings. For most of the automatically generated profile settings the CC radios have only been able to receive noise at the noise floor level. Because of this complication, predefined and tested profiles have been created for the CC radios, and the predefined profiles are used for sensing.

As presented in Section 4.1.2, sensing the radio spectrum in a wider frequency band than the channel bandwidth of the CC radios involves constantly switching the radio channel on which the CC radio receives. Due to this constant channel switching, it is possible to miss very short transmissions during sensing, because the radio might be receiving on a different frequency band than the frequency band of the transmission. Due to the time needed for the CC radios to switch between channels, Wi-Fi transmissions generally appear as a periodic pattern of dots in the waterfall plots of the sensing data, as it can be observed in Figure 4.3. For most radio profile settings, a full sensing sweep takes one second, so continuous transmissions longer than one second are guaranteed to be detected.

Averaging multiple RSSI values has the benefit of increased accuracy because of less noise in the measurements, however averaging increases the sensing time. Generally, the time spent sensing one channel can be written as:

$t_{channel} = t_{channel-switching} + N \times t_{RSSI-update}$
where

- $t_{channel}$ is the time spent on sensing one channel;
- $t_{channel-switching}$ is the time needed for the radio to change its channel from a different channel to the current channel and to provide the first correct RSSI value;
- N is the number of averaged RSSI readings;
- $t_{RSSI-update}$ is the update interval of the RSSI value.

Both CC radios have the following values for these parameters: $t_{channel-switching}$ is approximately 50 ms and $t_{RSSI-update}$ is at most 1 ms (Texas Instruments, 2010). In conclusion, 52 RSSI values have to be averaged ($N = 52$, $t_{channel} = 102\,ms$) in order to double the sensing time compared to the case without averaging ($N = 1$, $t_{channel} = 51\,ms$). In the profile settings the value of N is at most 10, and the performance difference between profiles with averaging and without averaging is small.

5 Distributed spectrum sensing

This chapter presents two distributed sensing experiments performed with the spectrum-sensing prototype. First, distributed sensing is qualitatively demonstrated in Section 5.1, then the performance of multiple sensing device is quantitatively analyzed in Section 5.2. Finally, discussions in Section 5.3 conclude this chapter.

5.1 Demonstration of distributed sensing

For demonstrating the correct functioning of VESNA together with other devices capable of spectrum sensing a distributed sensing experiment has been carried out with multiple device types. The VESNA low-cost spectrum sensing platform has been integrated (Padrah *et al.*, 2012) in the existing heterogeneous sensing system developed at the Wireless Networks Group of the Faculty of Electrical Engineering and Information Technologies - Institute of Telecommunications at Ss. Cyril and Methodius University - Skopje. After the integration in the heterogeneous sensing system three types of sensing devices have been used: eZ430-RF2500, USRP2 and VESNA. The first two devices are briefly presented in the following paragraphs.

The eZ430-RF2500 wireless development tool (Texas Instruments, 2012b) from Texas Instruments (TI) is a low-complexity, low-cost device. It consists of an MSP430F2274 microcontroller and a CC2500 radio, the same as the one used also on the VESNA platform. Its 16-bit RISC architecture CPU works on 1KB of RAM and 32KB of Flash memory. The CC2500 radio is a general-purpose highly configurable device from TI, working in the 2.4 GHz ISM band, and it has a 0.5 dB resolution for signal strength indicator. It has a maximum bandwidth of 840 kHz.

USRP2 (Ettus Research, 2012a) is a powerful, high-end computer-hosted software radio device. It collects radio signal samples from its RF interface and forwards the samples on its Ethernet port towards a PC; the PC can do all the signal processing in software, for example, by using GNU Radio (GNU Radio project, 2012). The USRP2, being a version of the USRP hardware platform presented in Chapter 2, equipped with the SBX daughterboard, has been receiving on a 40 MHz wide frequency band with its radio interface.

5.1.1 Experiment Setup

The experiment has consisted of measuring the power of a signal at preselected locations in a laboratory, for a set of different transmitter locations. A signal generator has been set up in turn at 17 locations to transmit a continuous signal at -20 dBm and in total 17 sensing devices with fixed locations have been sensing the signal level. For each location of the signal generator the received signal level has been recorded by each device for 20 minutes before the signal generator has been moved to the next position. The positions of the transmitter and the spectrum sensors are depicted in Figure 5.1. The $\circ$ signs represent the locations of the 10 eZ430-RF2500 (TI) devices, the + signs show the locations of the 5 USRP2s, the $\times$

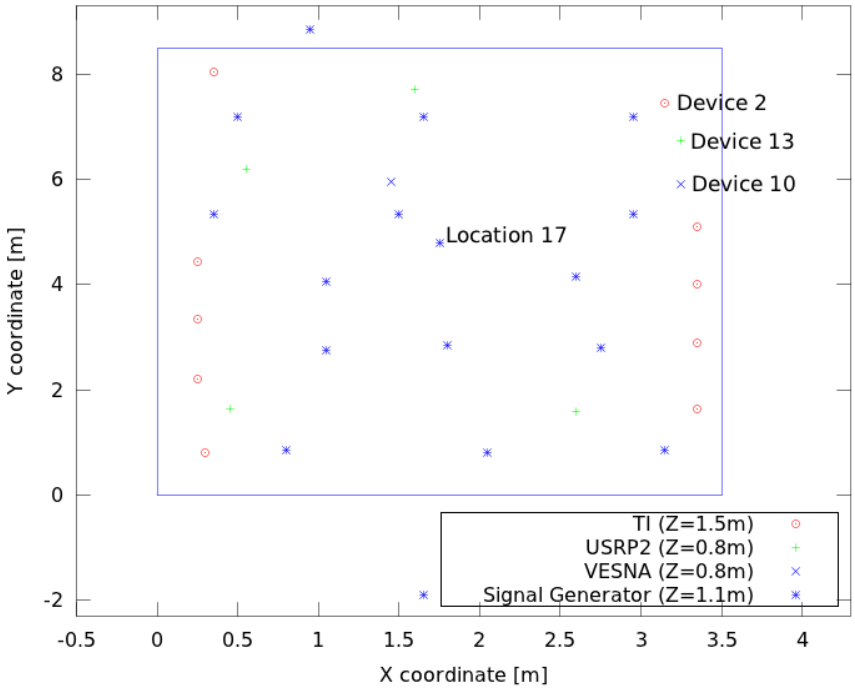


Figure 5.1: Positions of the signal generator and positions of sensing devices

signs mark the locations of the 2 VESNAs and the $\star$ signs show the locations at which the signal generator has been placed. Sensing devices have been placed on tables (USRP2 and VESNA) at height of 0.8 m and shelves (TI) at height of 1.5 m, whereas the signal generator was on a trolley at 1.1 m height. The rectangle on the figure represents the room's walls in which the experiment has been carried out. Note that two locations of the transmitter are outside of the room.

The frequency band 2400 MHz – 2420 MHz has been used in the experiment. The devices have recorded on average one spectrum sweep per second. The USRP2s have employed FFT-based signal power measurement, while eZ430-RF2500s and VESNAs have scanned the frequency band of interest using 840 kHz wide subchannels. Before starting the experiment, all of the devices have been calibrated with a signal generator.

5.1.2 Results

In Figure 5.2 representative measurement results can be observed from the experiment. For three selected devices of different type the variation of received signal power with time has been plotted. The devices have been located close to each other, at the locations marked Device 2, 10 and 13 in Figure 5.1. The transmitter has been placed in the middle of the room, at the Location 17 in Figure 5.1. From the plot of the signal power it can be seen that all devices have been functioning correctly during the measurements, and for each device the average value of the measured signal level is constant in time. Variations in the signal level like the one at the beginning of the measurement occur at the same time in all three traces, which confirms the correct time synchronization of the devices.

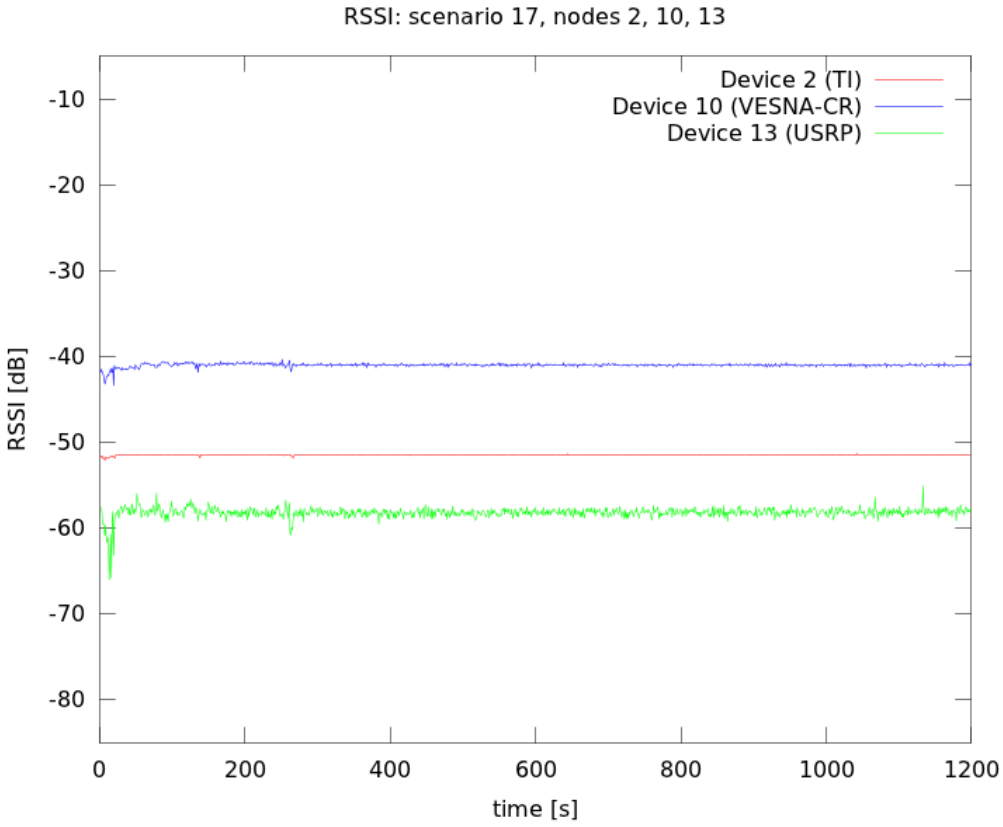


Figure 5.2: Signal power received by three devices, all belonging to different device types

5.2 Comparison of multiple devices

Having a spectrum sensing application developed for VESNA, it is of interest to characterize its performance and compare it with other types of devices. This section presents a sensing experiment carried out with VESNAs, presented in detail in (Van Wesemael *et al.*, 2012). In the presented experiment the signal power has been measured at multiple locations in an indoor environment and the collected data has been analyzed. The goal of the analysis has been to determine the influence of device and location types on the accuracy of the results.

5.2.1 Experiment setup

The experiments have been performed in the room depicted in Figure 5.3. The calibration of the sensing devices has been carried out with a signal generator and cables. At the position marked with $\times$ a signal generator has been placed. During the experiment it has been continuously transmitting a 20 MHz wide OFDM signal on the WiFi channel 8 with the transmit power equal to 3 dBm. All of the used sensing devices have been used to measure the signal power of the continuous OFDM transmission in turn at the 23 measurement locations, denoted with 1 to 23 in Figure 5.3. The devices used in the experiment have been:

- Metageek Wi-Spy 2.4x with Kismet Spec-tools for Linux OS (Metageek, 2012);
- Crossbow/Memsic TelosB (MEMSIC, 2012) (Polastre *et al.*, 2005); with CC2420 transceiver (Texas Instruments, 2007); and TinyOS application (Levis *et al.*, 2005);
- Fluke Airmagnet Spectrum XT (Fluke Networks, 2012);

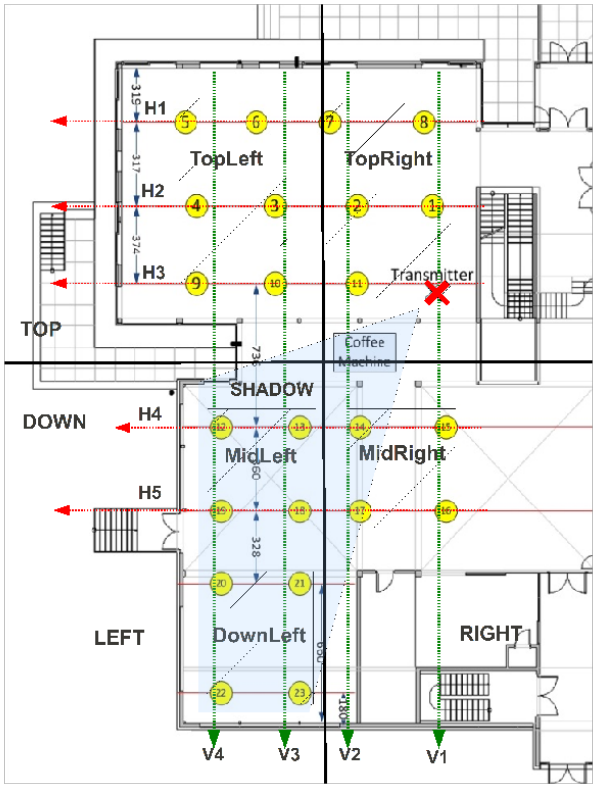


Figure 5.3: Sensing locations and locations of the transmitter

- Ettus Research USRP 2.0 (Ettus Research, 2012a) (Ettus Research, 2012b) with XCVR 2450 daughterboard and Iris (Sutton *et al.*, 2010);
- VESNA sensor node (SensorLab, 2012b);
- Imec sensing engine (Ingels *et al.*, 2010) (Pollin *et al.*, 2010).

For signal generation the Rohde & Schwarz lab equipment (SMIQ06 vector signal generator and AMIQ02 I/Q modulation generator) has been used.

At each location each device has been sensing for at least 30 seconds. For representing the data from all of the different device types, the CREW common data format (Cognitive Radio Experimentation World (CREW) Project, 2011) has been used. This data format defines a device-independent representation of spectrum sensing data. During experiments each device has been outputting measurement data in its own format and then all of the collected data has been converted to the CREW common data format. In this way the postprocessing of the data has been significantly simplified.

5.2.2 Results of the experiment

Experiment results have been obtained by different types of devices at different locations used to characterize a propagation path model for the considered operating environment.

The following very simple path loss model has been fitted over the collected data:

$$PL(d)[dB] = 20 \cdot \alpha \cdot log10(d) + \beta$$

where: PL is the path loss, d is the distance between the transmitter and the receiver, α and β are parameters of the propagation model. Least squares regression and robust fitting has been used to find the values of α and β. The least squares regression analysis method estimates values of parameters by minimizing the sum of squared difference between the

actual measured values and the expected values given by the model for all measurement data points. Mean squared error is the error metric that from multiple error values creates one error value by summing the squared values of all input error values. Robust fit also performs estimation of parameters, as the the least squares regression method, but its results are not so much affected by outliers in the processed measurement data as the results of the least squares regression.

First, fitting has been applied to data from each device type individually and the resulting path loss models have been considered device type references. Then all of the collected data has been used for fitting, defining the all devices reference path loss model. Because measurements have not been performed with a signal analyzer, the device reference has been considered the true value of the measurement values.

Using least squares regression fitting method, the values obtained for α and β have been 2.32 and 46.4, respectively, resulting in path loss model

$PL_{RSR}(d)[dB] = 46.4 \cdot log10(d) + 46.4.$

Similarly for robust fitting we obtained $\alpha = 2.29$ and $\beta = 46.8$, resulting in path loss model

$PL_{RF}(d)[dB] = 45.8 \cdot log10(d) + 46.8.$

The data used for fitting and the results for both methods are graphically represented in Figure 5.4 for all collected data. Figure 5.5 shows least square regression results for individual device types. In case of all devices reference, the fitted curve is located approximately at the center of the collected data. Between the individual device references, the USRP2 device shows unexpected results because it fitted curve is not parallel with the fitted curves of other devices.

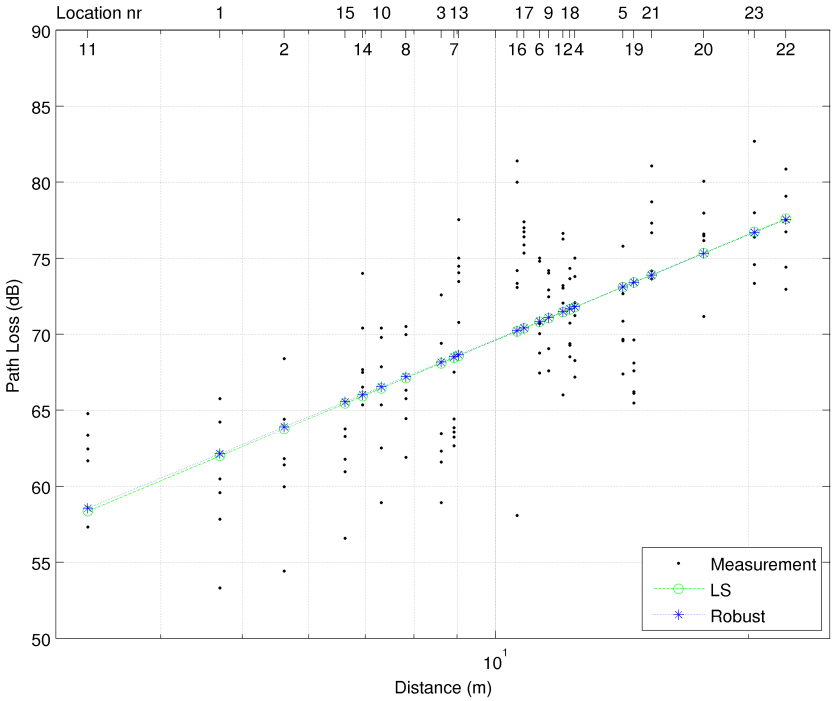


Figure 5.4: Least squares and robust fit

After the fitting, the error between collected data and fitted path loss model was calculated. As a metric, the mean squared error (MSE) has been used. For each device type, the data from the experiments has been compared with two fitted path loss models: the respected device reference, derived from its own data, showing the consistency of the device

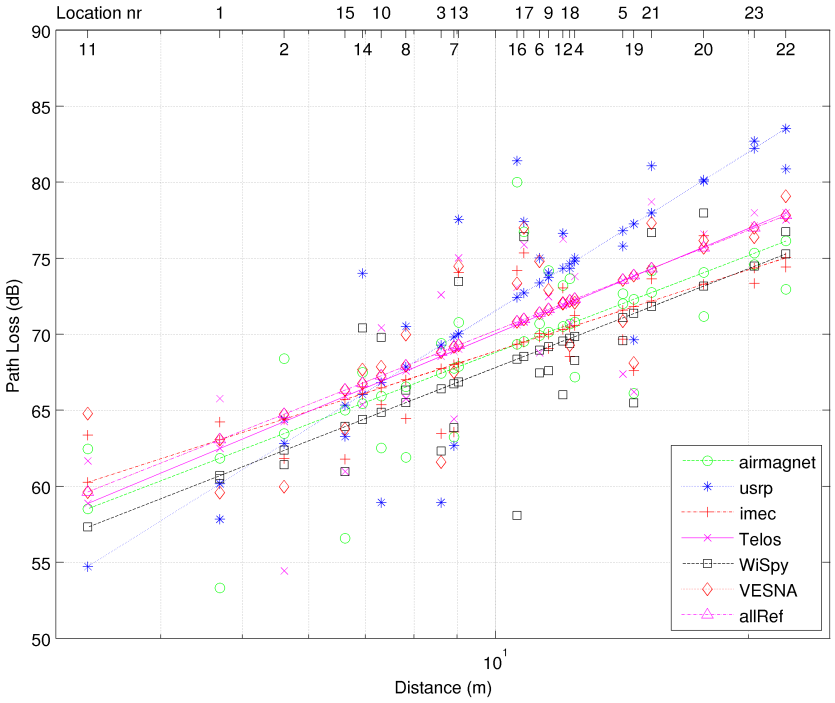


Figure 5.5: Least squares regression for individual device types

Device Name	MSE	
	Compared to devRef	Compared to allRef
allRef		19.9883
Airmagnet	31.8376	20.7657
USRP	33.9482	28.8005
imec	14.8554	21.5957
Telos	28.5254	20.1824
WiSpy	25.9246	23.2951
VESNA	15.6993	20.4692

Table 5.1: Mean Squared Error for individual device types

with its own measurements, and the all devices reference, calculated from the data gathered by all devices. The latter error shows the device's consistency with the average value of the measurements.

The results are presented in Table 5.1. From the second column of the table it can be concluded that imec and VESNA devices produce the most consistent measurements, while from the third column of the table it can be seen that the Telos and VESNA devices are the closest ones to the average value of the measurements.

5.3 Discussion

This chapter has presented two spectrum sensing experiments in which VESNA devices have taken part. In Section 5.1, three spectrum sensing devices and their integration into a heterogeneous sensing system have been presented. This system collects data from all of the sensor device types in a unified format. After the integration of the VESNA device into the sensing system, an experiment has been carried out. Based on the data collected

during the experiment, a representative combination of signal traces has been presented, which incorporates data from all three integrated device types. The developed prototype has been integrated into a spectrum data collecting system and it functioned properly.

In the second experiment, presented in Section 5.2, based on the data collected at multiple locations, a very simple path loss model has been fitted to the scenario. By calculating the difference between the measured values and the values predicted by the fitted path loss model the consistency of the collected data has been characterized. The data collected by VESNA devices has been the second most self-consistent between the used spectrum sensors. This self-consistency shows that measuring radio frequency signal power with VESNA device gives relevant results, so VESNAs are well suited for usage in a testbed.

Another important conclusion of this chapter is the need for a well-defined data format in multi-device experiments. The use of CREW common data format has allowed the processing of data coming from six different device types. In the VESNA-based testbed the data format of measurements will be clearly defined in order to allow easy access to the users and improved interoperability.

6 Spectrum sensing testbed

Having a well-performing simple spectrum sensing prototype ready, the next step of development is the integration of the spectrum sensing functionality into a testbed and the development of the functionalities required for the operation of the testbed. In comparison with the spectrum sensing prototype, VESNAs in the testbed have to accept commands from remote operators of the testbed, not just from a directly connected PC; also the data collection mechanism in the testbed needs to collect data from multiple nodes and it needs to function over multiple, potentially lossy communication channels. Also, the testbed has to operate reliably without the need of physical intervention in it.

This chapter presents the design of the LOG-a-TEC testbed. First, the proposed architecture of the testbed is presented in Section 6.1. The goals of the testbed are described in Section 6.2. The requirements and constraints are discussed in Section 6.3. Section 6.4 describes the procedure of physically designing the testbed by selecting the hardware and locations of the devices. Section 6.5 shortly presents the communication protocol used in the testbed, with focus on spectrum sensing. Section 6.6 describes the management and access control system supporting the deployed VESNAs and presenting a friendly interface to the users of the testbed. Section 6.7 presents how the testbed can be used and how experiments can be automated.

6.1 Testbed architecture

The LOG-a-TEC testbed was planned to be located in the city of Logatec, Slovenia. This testbed has been planned to consist of two major parts: (i) the remote field deployment of VESNA sensor nodes, performing experiments and (ii) the central management and access control system, used for managing the deployed VESNAs, setting up experiments and collecting the experiment results. Inside the remote deployment, nodes use a sensor network form communication and this sensor network is connected through a suitable gateway to the management and control system.

The initial phase of the testbed has been specified to contain 50 VESNA devices. Because of the limited number of locations, where devices could be installed, the 50 devices have been split into two clusters, each cluster being composed of 25 VESNAs. One cluster is located in the industrial zone, while the other is located in the city center. The two clusters have different characteristics with respect to radio communications. The locations of the two clusters inside Logatec are presented in Figure 6.1. The markers represent the locations of the clusters. The marker with a black dot corresponds to the city center, while the marker without a dot shows the location of the industrial zone. The devices are installed on public light poles.

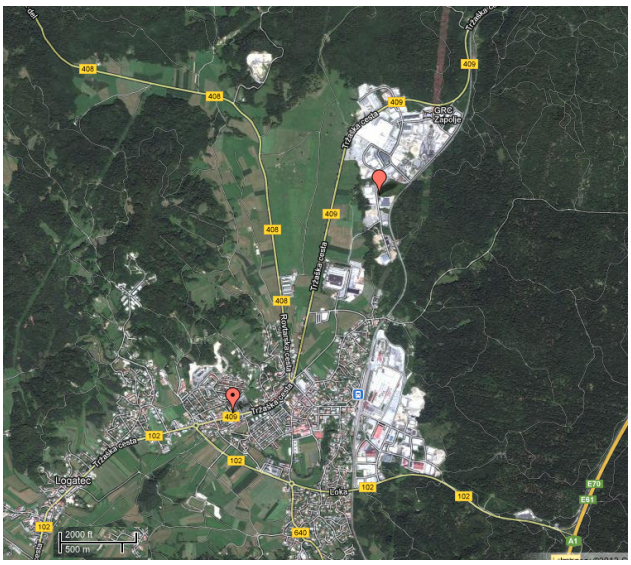


Figure 6.1: Locations of the device clusters inside Logatec

6.1.1 Building blocks

Each deployed VESNA for spectrum sensing in LOG-a-TEC testbed is equipped with multiple radio modules: one module is used for communications on the management network and additional radio modules are used in experiments. The communication module is located on a radio board, while the radios used in the experiment can be found on an expansion board. Both of these boards have been presented in Chapter 3. The modules used in experiments include the following:

- Transceivers operating in the license-free 868 MHz European ISM and upper channels of the licensed UHF TV band based on CC1101 from Texas Instruments. The primary use of these transceivers is spectrum sensing and signal generation with the possibility of carrying out communication experiments.

- Transceivers operating in the license-free 2.4 GHz international ISM (industrial, scientific, medical) band, based on CC2500 from Texas Instruments. The primary use is spectrum sensing and signal generation with the possibility of communication experiments.

- IEEE 802.15.4 compliant radios, in the form of AT86RF212 and AT86RF230 from Atmel. The first radio operates in the license-free 868 MHz European ISM band, while the second one operates in the 2.4 GHz international ISM band. These radios are to be used for communication experiments.

- Custom designed spectrum sensing receivers operating in the UHF TV band based on TV tuner TDA18219.

Each VESNA is equipped with a subset of the radios presented above. There are three types of nodes, based on the equipped radio types:

- AT86RF212 + CC1101 for 868 MHz sensing

- AT86RF212 + CC2500 for 2.4 GHz sensing

- AT86RF212 + TV TDA18219 receiver for UHF sensing

The AT86RF230 radios are not used because they have smaller range than AT86RF212s and it would be hard to create outdoor multi-hop mesh networks with it. Moreover, as most of license-free experiments are planned for the 2.4 GHz ISM band it would not be very suitable to have the management network in the same frequency band. The optimal configuration of the testbed, including the exact location and configuration of each device is determined in Section 6.4.

6.1.2 Services and protocols

VESNA platforms as embedded devices have limited computational capability, determined by the CPU and memory of their microcontroller, thus part of the functionality of the testbed has to be implemented on the management and access control system serving the testbed. For example, experiment scheduling, user access control and collection of all data from the experiment have to be performed by more powerful computers than the microcontroller of VESNA. Hence, a set of services for the testbed have been developed for computers located at JSI, which are tightly coupled with the VESNAs in the testbed. For security reasons the deployed VESNAs should not be directly accessible so the users of the testbed have to use the interface exposed by the management and access control system.

For communicating with a VESNA in the testbed, the transmitted and received data has to pass through at least two communication channels: the ZigBee based network between the targeted VESNA and the coordinator of the network, and the TCP connection between the coordinator and the management and access control system. Because of the needed technology-independent data transmission and possible data loss, custom transport and application layer protocols have been designed for the testbed. As a standardized alternative to custom protocols, the IETF CoAP (Internet Engineering Task Force, 2012) has been considered, but its implementation for microcontrollers is part of Contiki OS (Dunkels *et al.*, 2004). This means that either Contiki OS had to be ported first to VESNA and all of the functionalities of the testbed would have to be integrated with Contiki OS, or the CoAP protocol had to be reimplemented. Because Contiki has been considered insufficiently stable for use in the testbed and because porting would have required a significant amount of effort, the development of a custom-protocol has been selected instead. This way the protocol could be designed for the needs of the testbed, and its development could proceed faster.

The new protocol is similar to the hypertext transfer protocol (HTTP) and in functioning principle to CoAP, but it does not claim compatibility with any of them. In this protocol data can be requested from abstract resources (GET) residing on nodes in the testbed, or data can be posted to the abstract resources (POST). For preventing data corruption, the requests that post data always contain a cyclic redundancy check (CRC) value of the entire request. The custom-developed protocol defines similar GET and POST requests as HTTP and CoAP. A significant difference between HTTP and the custom protocol is that the custom protocol allows posting data and arguments to a resource with the same request. This functionality is important for request forwarding by the coordinator of a ZigBee network. In the future, when appropriate CoAP implementation is ready for the VESNA platform, it should be possible to migrate the system to CoAP.

The approach of abstracting functionality with getting and posting data allows a very modular firmware structure, because multiple functionalities can be developed independently by implementing different resources. For example, remote reprogramming of the devices, spectrum sensing and signal generation have been independently developed, and later integrated. Also this approach allows easy extension of functionalities: a new resource can be defined on the nodes, without modifying the preexisting ones. Another important benefit

of the resource-based approach is the possibility to reduce the complexity of the algorithms running on VESNAs. If a functionality is not time critical, a few low-complexity resources can be defined for supporting it on VESNAs, and a more complex algorithm that interacts with the resources can run on the management and access control system side. For example, this is how spectrum sensing and remote reprogramming of the nodes in the testbed are implemented. Only the services running on the management and access control system part have information about the exact status of each node; the devices have only information about themselves, and they only react to the commands sent by the management and access control system.

The requests coming from the management and access control systems are dispatched to their destination by the gateway node. A simple addressing scheme has been defined: for nodes different than the gateway of the cluster, the requests are sent to a special resource named `nodes`. A field in the requests specifies the destination address inside the cluster; based on this address the gateway forwards requests to their destinations. The responses of the requests are forwarded by the gateway to the management and access control system. For example, for getting the list of sensing radios from node number 5, the following request should be sent to the gateway: `GET nodes?5/sensing/deviceList`.

For connecting the clusters of the testbed to the management and access control system, the Ethernet expansion board, presented in Chapter 3, has been mounted on the coordinator of the ZigBee network. This board has been programmed for automatically connecting to a fixed IP address and port number, though an encrypted TCP connection (SSL). The scheme of coordinator connecting to the management and access control system has been chosen in order to avoid possible limitations of the Internet access at the testbed. For example, if the coordinator of a cluster is behind a firewall or network address translation (NAT), then it can only accept incoming connections if the router or other networking device is reconfigured. Because the equipment that connects the coordinators to the Internet is typically provided by external parties, the effort needed to allow the Ethernet board to accept connections might be high, without any significant benefit. Another reason why this connecting scheme has been chosen is that the Ethernet board's assigned IP address might change in time, and then we would need a way to find the coordinator of a cluster on the Internet. Because the servers at JSI have fixed IP addresses, the used connecting scheme avoids this problem.

The logical block diagram of one cluster of the testbed is depicted in Figure 6.2. The cluster has two parts. The first part, where the experiments are performed, is located in Logatec and it is illustrated in the upper part of the figure. This part is connected to the Internet through a Gateway and internally uses a ZigBee network for communication. The second part of the testbed, presented in the lower part of the figure, is the management and access control system serving the experimental part. This part consists of a server connected to the Internet and communicating with the Gateway. Users of the testbed connect to the PC on the management and access control system part of the testbed.

6.2 Goals for spectrum sensing testbed

Compared to the individual spectrum sensing prototype, the testbed configuration has more requirements because it has to operate without physical intervention on it. The individual spectrum sensing prototype is generally residing at a few meters distance from its operator, so it can be manually supervised, reset, configured or even reprogrammed as needed. In contrast to the above, the spectrum sensing testbed is only accessible through its network interface and there is no visual feedback about its operation. Because of these constraints, a set of goals have been defined for the testbed, in order to ensure its proper functioning,

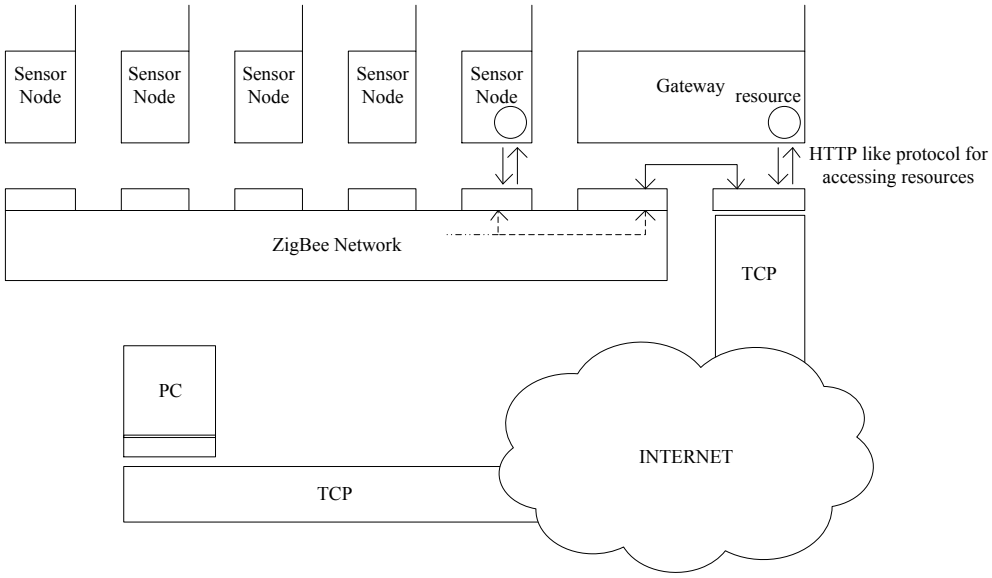


Figure 6.2: Logical block diagram of the LOG-a-TEC testbed

reduce the effort of developing it and make the testbed easier to operate.

The following features and goals have been proposed:

- Everything required for experiments should be configurable remotely through the control network of the testbed. Because the devices forming the testbed are not physically accessible, the setup has to be done from a remote location.
- Unified control interface: All nodes in the testbed should be accessed through a consistent addressing scheme. This makes the usage of the testbed easier. The addressing scheme is necessary because the coordinator of each cluster has to do request forwarding between an SSL connection and the ZigBee network.
- Centralized control and data collection: For simplicity and in order to avoid uncontrolled situations in the ZigBee network, all communication in the ZigBee network should happen between the coordinator of the network and one node, or for broadcast data between the coordinator and all of the nodes. Direct node-to-node requests are not allowed. Also, nodes should just wait for requests from the management and access control system; they should not initiate requests.
- Possibility of easily adding functionality in the future. The initial setup of the testbed might not cover all possible experiments and in the future new important experiment types might appear. Two features of the testbed allow the addition of new experiments: (i) the possibility of remotely reprogramming the VESNAs in the testbed and (ii) the possibility of adding new resources to the nodes with minimal effort.

6.3 Requirements and constrains of spectrum sensing testbed

This section focuses on the requirements and constraints of the ZigBee based management network in the testbed. The requirements are imposed by the use cases of the testbed,

namely spectrum sensing, while the constraints are related to the placement of devices and their capabilities.

Communications for network management are carried out in the 868 MHz ISM band, while experiments are done primarily in the 2.4 GHz ISM band, but possibly also in the 868 MHZ ISM band; in UHF TV band usually only spectrum sensing is performed by the deployed devices although some transmission can be done in upper UHF channels using 868 MHz radio transceivers. One of the constraints of the network design is the availability of frequency bands where the ZigBee network can operate. There are two choices for this network: either the 2.4 GHz ISM band or the 868 MHz ISM band. Because the 868 MHz ISM band offers better propagation characteristics and it does not interfere with the crowded 2.4 GHz ISM band, where the majority of the experiments are expected to be performed, the 868 MHz ISM band has been chosen as operating band for the ZigBee network. The locations for devices are constrained by the following factors:

- Availability of Internet access: the Gateway of each cluster has to be connected to the Internet; for maximizing the performance, wired connection is preferred.
- Location of light poles: devices are mounted on light poles belonging to the public lighting system. Possible exceptions are the Gateway devices.
- Power connections to the light poles: the devices in LOG-a-TEC have been planned to have always available power source. Light poles are powered in groups, so economically the most efficient solution is to equip all light poles of the selected light pole groups.
- Connectivity: because all the devices are accessed through the Gateway, for maximum performance it is important to have minimum number of hops from any device to the Gateway.
- Possibilities for experiments: the clusters of the testbed should allow experiments that involve different types of network topologies, including multi-hop scenarios and mesh networks.

Because the first two constrains drastically reduce the number of possible locations for the devices, the problem of selecting the device locations for both clusters can be modeled by the abstract problem of selecting N=25 locations from the available M possible locations. This model is used in later sections.

Besides constraints for the network, it is important to evaluate the network performance required for the operation of the testbed. The usual management activity for the network is expected to generate small amount of traffic while the following activities are expected to require the transfer of large amounts of data:

- Transmission of firmware images for devices, in order to run custom applications during experiments: to allow the testing of custom protocols, their implementation has to be uploaded to the devices. Because a local storage for firmware images is implemented on each device, the uploading operation is not time-critical. The amount of transferred data is expected to have a few hundreds of kilobytes for each different firmware type. Because the firmware distribution mechanism is planned to take advantage of the broadcast nature of the radio communications, by a single transmission multiple nodes can receive a firmware image instantly.
- Collection of spectrum sensing data: during experiments, a significant amount of data can be collected. The data can consist of received signal strength indicator (RSSI) values taken at a given frequency and at a given time, or it can consist of various

performance metrics like packet loss rate or average network throughput. The spectrum sensing data can be similarly saved to a local storage on the devices as the firmware images, but the resulting amount of data can be in the order of megabytes. If the required sample rates of the data allow it, real-time data collection can be implemented; in this case the measurement results are transmitted as soon as they are collected. The feasibility of the real-time data collection depends on the latency and throughput of the ZigBee network and on the performance of the sensing radios.

For the management network operating in the 868 MHz ISM band, ATZB-900-B0 ZigBit™ modules (Atmel, 2012a) from Atmel are used. The modules implement the ZigBee specification with proprietary extensions. When modules are started, they form a mesh network with a central coordinator, zero or more routers and zero or more end devices. In our deployment the coordinator of the ZigBit network is placed on the VESNA node with the Internet access. After some initial testing, it has been found that modules configured as end devices have much smaller throughput than modules configured as routers, most likely because of radio duty-cycle limitation of the end devices. Hence, all the modules used in the testbed were decided to be configured as routers, except for the two coordinators of the clusters. The maximum amount of data that can be transmitted in a ZigBit frame is 94 bytes when security is disabled and 64 bytes when security is enabled (Atmel, 2012b). Enabling security means that the traffic in the network is encrypted. This feature is useful for preventing various types of attacks on the testbed. Because no application level security is implemented, enabling security in the ZigBit network is necessary for preventing packet injection in the network. Tests have shown that maximum 10 packets per second can be transmitted before significant packet loss happens in the ZigBit network. This means that the maximum point to point throughput in the network is $10 \times 64 = 640$ bytes per second. For calculations, this number can be used as an approximate measure of the network performance. For the transmission of firmware images, the performance of the network can be increased by broadcasting the firmware image in the network; in this case the redundant transmissions to each node can be eliminated.

6.4 Network Planning

Given the requirements of the testbed, the limitations of the devices, and physical possibilities for the deployment presented in the previous section, it is of interest to select exact locations in Logatec where VESNAs can be installed. This section presents the steps taken for selecting device locations. A set of measurements have been carried out in two zones in Logatec, and based on the results, locations have been selected for the devices that form the testbed.

In the following subsections first the methods used for designing the testbed and for performing necessary measurements are described, then the measurements performed at JSI and at the planned location of the testbed are presented.

6.4.1 Measurement methodology

The behavior of the management network has been determined experimentally, by measuring radio link quality between possible locations of devices. For the experiments, two VESNAs have been used, both equipped with the ATZB-900-B0 radio modules that are used in the LOG-a-TEC testbed. One VESNA has been acting as an echoing device denoted with (E), while the other VESNA has been transmitting requests to the first one, and receiving the responses denoted with (T/R). This setup is depicted in Figure 6.3. The VESNA node (E)

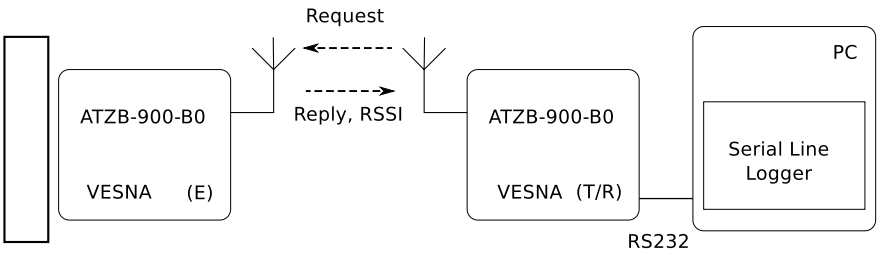


Figure 6.3: Setup used for measuring link quality

is responding to the requests sent from the VESNA node (T/R). The VESNA node (T/R) is connected with an RS232 connection to a PC and the PC saves the measurement logs. The PC and the VESNA connected to it have been moving during the measurement campaign while the VESNA node (E) has been fixed to a location. The VESNA node (T/R) with PC is depicted in Figure 6.4 and the fixed VESNA node (E) is presented in Figure 6.5. The

Figure 6.4: VESNA node (T/R) and a PC on a mobile measurement platform

Figure 6.5: VESNA node (E) at a fixed location

saved output from the VESNA node (T/R) consists of the following:

- Sequence number of the reply; this number allows detection of lost requests or other unexpected problems.

- Round trip time: the time elapsed between the moment of sending the request and the reception of the reply.
- Received Signal Strength Indicator (RSSI): the power level of the received reply, measured in dBm.
- Link Quality Indicator (LQI): value showing the power and the amount of distortion in the frames received by the radio (Ergen, 2004). Because the ATZB-900-B0 modules are internally implemented with an AVR microcontroller (ATmega1281) and an AT86RF212 radio, the LQI obtained from the modules is expected to correspond to the LQI of the AT86RF212 radio, as presented in the radio's datasheet (Atmel, 2010).

For every 30 requests, the VESNA connected to the PC has been printing statistics about packet loss, time needed for measurements, average round trip time and RSSI. The application running on the VESNAs has been developed by using the VESNA libraries; in the PC an off-the shelf serial line logger application has been used. A small representative part of the output from the VESNA is presented below:

```
reply from 5051: seq=266 time=108 ms
   lqi=252 rssi=-54 dbm
reply from 5051: seq=267 time=105 ms
   lqi=252 rssi=-52 dbm
reply from 5051: seq=268 time=106 ms
   lqi=252 rssi=-53 dbm
30 pkts transmited, 30 received,
   0% packet loss, time 30023 ms
   avg 109 ms rtt, -52 dbm rssi
```

During the measurements, first the VESNA node (E) has been mounted on a fixed location, then the VESNA node (T/R) connected to a PC and mounted on a mobile measurement platform has been moved to each measurement location and link measurement has been performed for approximately two and a half minutes. The collected data has been saved for subsequent processing.

The GPS coordinates of each measurement location have been recorded. The GPS receiver incorporated in many smartphones proved to have low accuracy, typically 50m, while a dedicated GPS receiver could reach up to 3m accuracy. During the experiments, a GPSMAP 60CS device produced by Garmin has been used for marking the coordinates of the measurement locations. This has been necessary because no complete maps containing the locations of the light poles have been available for the city of Logatec.

6.4.2 Testing measurement procedure

By having the measurement equipment ready, a set of preliminary measurements has been carried out in the premises of Jožef Stefan Institute, in order to test the measurement equipment and procedure.

Measurements at JSI have followed the same procedure as the measurement performed in Logatec. This procedure is the following:

1. start the GPS receiver;
2. install the echoing device at a fixed position;
3. note the GPS location of the echoing device;

4. for each selected measurement location:

   (a) move the mobile measurement platform to the location and verify that the mobile VESNA node's antenna is pointing upwards;

   (b) start the data collection;

   (c) wait until the GPS receiver shows the current location with the maximum possible accuracy;

   (d) note the current GPS location;

   (e) wait until the data collection finishes;

   (f) stop the current measurement;

The measurement procedure has proved to work well during the test-measurements. Manually writing the GPS data in a writing pad was not practical, because of a need to hold both the GPS receiver and the writing pad in one hand. Hence, the GPS locations have been manually collected in a text file on the computer. Computer was also programmed to automatically remind experimenters to check the position of the antenna on the mobile measurement platform before taking measurements as it tended to drop from the vertical position during movement.

The locations where the measurements have been performed are depicted in Figure 6.6 while the measurement results obtained at each of the locations are presented in Figure 6.7.

Figure 6.6: Measurement locations at JSI, satellite view

Figure 6.7: Measurement locations and results at JSI, map view

On both figures the red marker represents the location of the echoing device and the blue markers represent the measurement locations. In Figure 6.7 the plotted bars represent the received RSSI value (yellow left bar) in dBm and the packet loss in percents (red right bar). Error bars show the minimum and maximum RSSI value obtained during the measurement.

6.4.3 Measurements in Logatec

In Logatec, two clusters of the testbed are planned. One cluster is located in the industrial zone, while the other cluster is in the center of Logatec. Because public light poles are

powered in groups, the light poles on which VESNAs could be installed have been preselected based on the power group to which they belong. The measurements carried out in the two clusters are presented in separate subsections.

Industrial zone

The locations of the light poles in the industrial zone are depicted in Figure 6.8 and in Figure 6.9, on a satellite and on a map view of the zone, respectively. Figure 6.9 also shows the numbers assigned to the light pole locations. Location (C) represents the position of the coordinator; this location has been fixed because only that location provides an Internet connection.

Figure 6.8: Measurement locations in the industrial zone of Logatec, satellite view

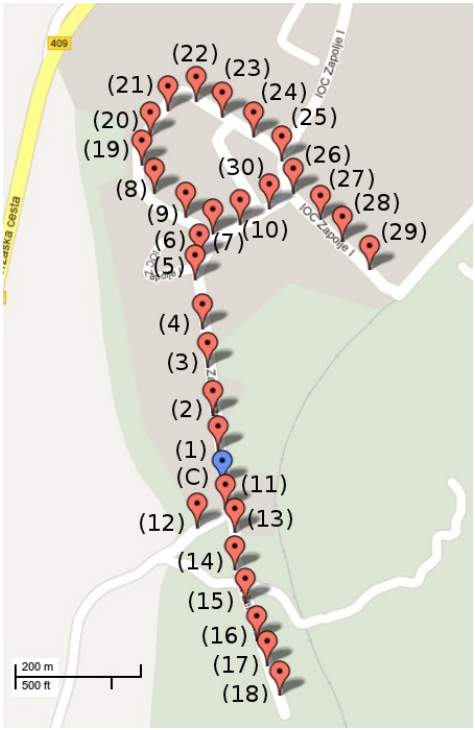


Figure 6.9: Numbers of the measurement locations in the industrial zone of Logatec, map view

Two sets of measurements have been carried out in this zone: first the link quality between the position of the coordinator and one subset of the light poles has been measured, and second the link quality between location number 5 and another subset of the light poles has been measured. The two subsets of measurement locations partially overlap, and together they cover all possible device locations. The results from the first set of measurements are presented in Figure 6.10 and the results from the second set of measurements can be seen in Figure 6.11.

From the results it can be seen that for locations farther away from the coordinator than location 5, over 10 % of the transmitted packets are lost. Hence, it has been considered that the nodes that are closer than location 5 are reachable from the coordinator with one hop, and the rest of the nodes need more than one hop to reach the coordinator. Because of the bad connection quality the second set of measurements has been carried out with the echoing device placed at location 5. The results from the second set of experiments show that most of the positions are reachable with two hops, although some locations, for instance 20, 25, 26, 28 and 29 might need a third hop for reaching the coordinator. From

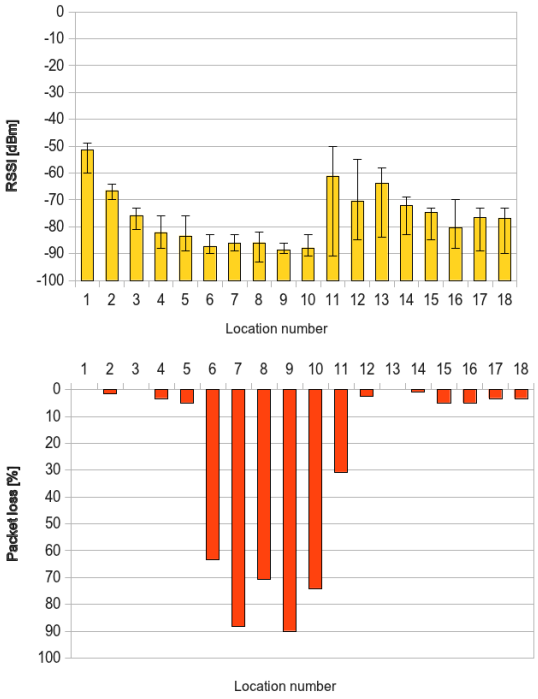


Figure 6.10: RSSI and packet loss in the industrial zone, measured between public light poles and the coordinator's location

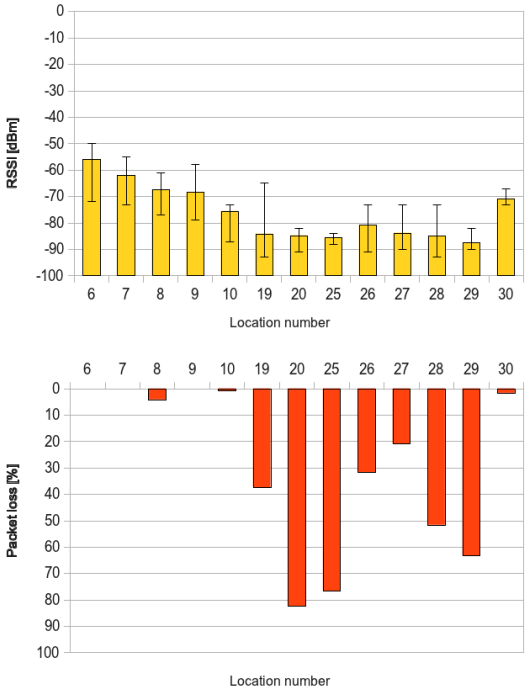


Figure 6.11: RSSI and packet loss in the industrial zone, measured between public light poles and location number 5

the measurements it has been concluded that the ZigBit modules operating in the 868 MHz ISM band can reach nodes at least at 5 light poles away. These estimates are pessimistic because the measurement have been carried out at the ground level and on the public light poles, some 4 to 5 meters above the ground, the propagation conditions are expected to be better.

City center

The locations of the light poles in the center of Logatec are presented on a satellite view in Figure 6.12 and on a map in Figure 6.13. The numbers on the map represent the numbers

Figure 6.12: Measurement locations in the center of Logatec, satellite view

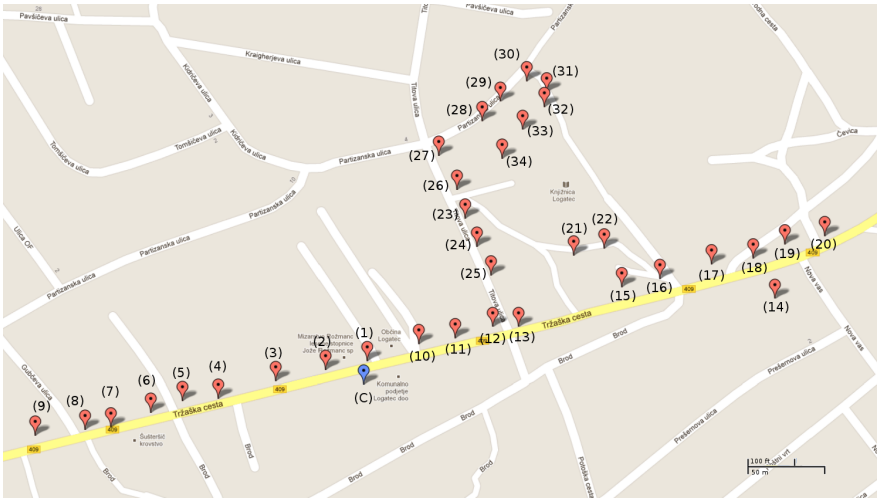


Figure 6.13: Numbers of the measurement locations in the center of Logatec, map view

given to the public light poles in the center of Logatec. The location marked with (C) indicates the location of the coordinator; at that position Internet access is available for the testbed. The public light poles in the center are installed more densely than in the industrial zone.

As in the industrial zone, in the center of Logatec two sets of measurements have been carried out. In the first set the link quality from the location of the coordinator towards most of the other possible locations has been estimated, and in the second set the link from location number 25 towards the rest of the public light poles in the zone has been estimated.

The results of the two measurement sets are shown in Figure 6.14 and Figure 6.15, respectively. From the first set of measurements it can be seen that the coordinator has

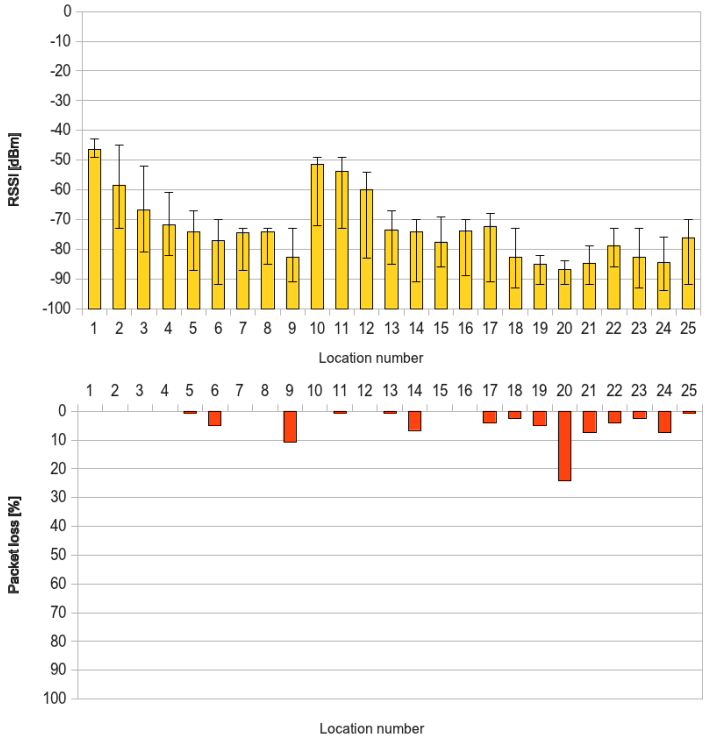


Figure 6.14: RSSI and packet loss in the center of Logatec, measured between public light poles and coordinator

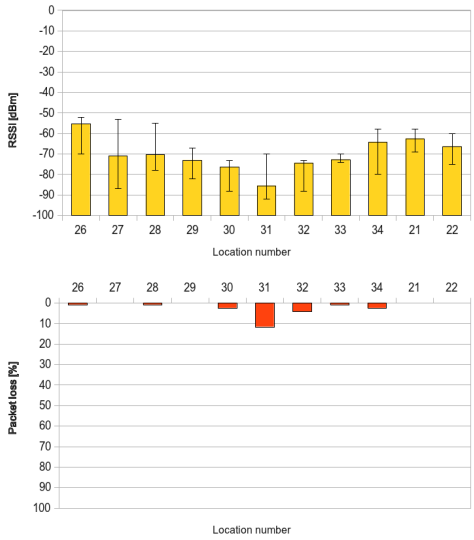


Figure 6.15: RSSI and packet loss in the center of Logatec, measured between public light poles and location 25

a good quality link to all of the devices installed along the central street of Logatec. The only location where more than 10 % packet loss has been measured, is location 20. After performing the measurement of links pointing to the coordinator's location, the echoing device has been moved to location 25, from where good link quality was expected towards the rest of the possible locations of devices. The results shown in Figure 6.15 confirm the good link quality towards locations 26-34. Based on these measurements, all nodes from the cluster in the center of Logatec are expected to be at most two hops distance from the coordinator of the cluster.

6.4.4 Selecting device locations

ZigBit modules form a network with a tree topology starting from the coordinator as the root of the tree. The maximum hop count in network when communicating with the gateway is expected to be 3 or 4 hops in the case of the industrial zone and 2 hops for the city center of Logatec, so the proper functioning of the ZigBit modules and the automatic build-up of the ZigBit network is expected.

From the measurements it can be seen that the communication range in the 868 MHz ISM frequency band is at least 5 light poles. This means that the sniffing devices operating in the 868 MHz ISM band can be installed on every third light pole, without affecting the sensing performance significantly. Sniffers operating in the 2.4 GHz ISM band have been installed every second light pole or closer, because they have smaller reception and transmission range. TV transmissions in the UHF frequency band are constant and the locations of the transmitter towers are known, so it's enough to install only a few UHF band TV receivers in the two clusters of the testbed.

Based on the observations from above, the light pole allocations depicted in Figure 6.16 and Figure 6.17 have been created for the industrial zone and for the center, respectively. For each node, the associated marker shows the type of installed sensing radio on the top of the marker and the nodes' ZigBee address at the bottom. The markers with blue upper part and 868 written on them show the locations of the 868 MHz ISM band sniffers, the markers with red upper part and 24 written on them show the locations of the 2.4 GHz ISM bands sniffers, the markers with green upper parts with TV written on them represent the UHF sniffers and the white markers show the location of the gateway/ZigBee coordinator

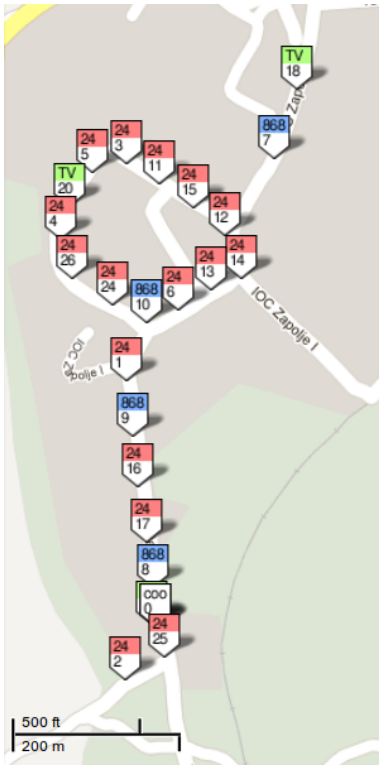


Figure 6.16: Device types deployed in the industrial zone

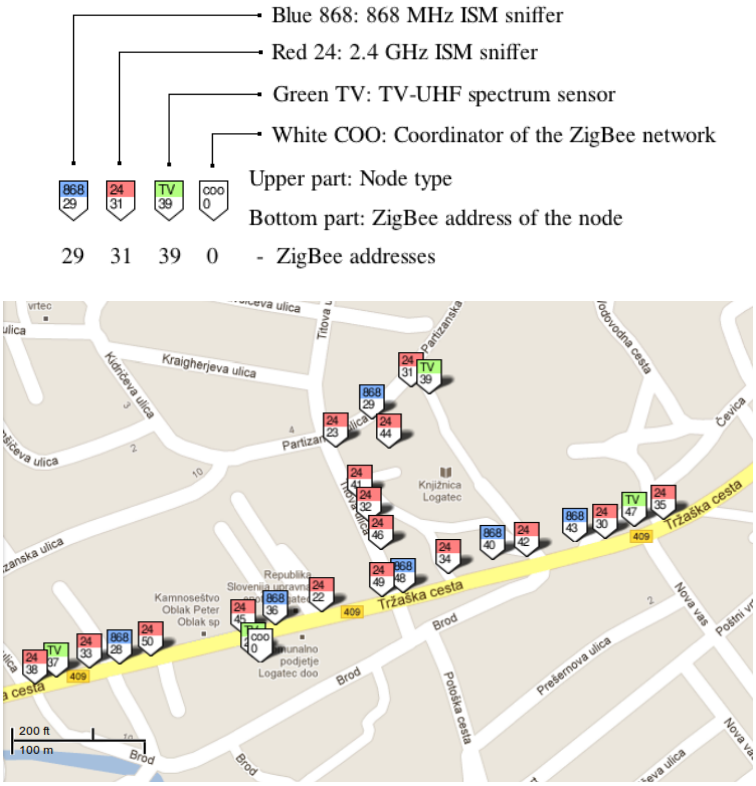


Figure 6.17: Device types deployed in the city center

of the clusters. In both clusters these locations have been chosen to be collocated with the TV-UHF band spectrum sensors; these markers are only partially visible. The ZigBee address of the coordinators is always 0, while the address of the other nodes is written on the bottom part of their corresponding markers. Note that the ZigBee addresses of the nodes are independent of the public light pole locations; the addresses identify a VESNA node, while the public light pole numbers identify locations.

By comparing the figures depicting the location of the installed nodes (Figure 6.16 and Figure 6.17) with the figures of the measurement locations (Figure 6.13 and Figure 6.9), it can be observed that the set of measurement locations and the set of installed location does not completely overlap. There are measured locations where no nodes have been installed and also nodes installed at not measured locations. This is caused by two factors: first, some locations are reserved for future expansion of the testbed, so at those locations no VESNAs have been installed; second, the public light poles are powered in groups and some locations belong to different groups than originally expected, and other, not measured locations are in the same groups as the measurement locations. A subset of the not measured locations belonging to the same light pole powering groups as the measurement locations have been used in the testbed.

As it can be seen from Figure 6.16 and Figure 6.17, three UHF band TV receivers are planned for the industrial zone, and four for the center. The locations of the TV receivers are evenly distributed along the area of the testbed. For the rest of the locations, in the industrial zone the pattern of one sensor for 868 MHz ISM band and two sensors for 2.4 GHz ISM band is repeated, while in the center the 868 MHz and 2.4 GHz ISM band sensors are assigned to light poles in an alternating manner. This way 14 sensors for 2.4 GHz ISM band have been installed in the industrial zone and 15 in the center, while for 868 MHz ISM

band sensing 9 and 7 sensors have been installed in the industrial zone and in the center, respectively. The reserved locations are marked on the maps because the public light poles are powered in groups, so the reserved locations have to be powered in any case. Knowing that power is already available at some locations is useful for the future expansion of the testbed.

6.5 Protocols

As it has been presented in Section 6.1, custom-built transport and application layer protocols similar to HTTP are used in the testbed. In the rest of this thesis we refer to the collection of these protocols as the HTTP like protocol (HLP). These protocols are presented in this section. Two types of requests can be sent to resources abstracting various functions: GET for reading data and POST for writing data. The format of requests is specified by the transport layer, while the name of resources and the format of data handled by the resources is defined by the application layer.

6.5.1 Transport Layer

On the transport layer the format of requests is defined. The generic form of a GET request is the following:

```
GET resource?arg1=val1&arg2=val2&...&argN=varN\r\n
```

where

- `resource`: abstract resource identifier, for example, `firmware/version` or `sensors/temperature`.
- `arg1`: parameter 1 name
- `val1`: value of parameter 1
- `argN`: parameter N name
- `valN`: value of parameter N
- `\r`: Carriage return character, ASCII value is decimal 10
- `\n`: Line feed character, ASCII value is decimal 13

Sending a GET request to resource generally means that data is requested from the specific resource. For example, sending a GET request to the firmware/version resource could return the value `1.0`. The GET request does not have data integrity check defined on the transport layer, because it should not alter the state of the node, so any GET request should be repeatable. Later on this design decision was not considered the best because in many resource the data returned as answer to a GET request had to contain CRC value. It would have been simpler to add full transport layer error control from the beginning.

The second request defined, the POST request, has the following general form:

```
POST resource?arguments\r\n
Length=len\r\n
<data, having len bytes length>\r\n
crc=crc_value\r\n
```

where:

- `resource`: abstract resource, for example: `firmware`
- `arguments`: arguments given to the handler of the POST request, example: `12.34/firmware`
- `len`: length of the data that will be written to the specific resource
- `data`: possibly binary data, to be transmitted
- `crc_value`: value of CRC calculated on all the previous content, except the line starting with `crc=`; its value is represented as an unsigned decimal number

As it can be seen, the POST request sends data and optional arguments to a resource; the integrity of the request can be verified by using the CRC field in the request. Because the POST requests are expected to change the state of a resource, subsystem or node, it was considered important that only valid requests are accepted.

The generic format of a response is the following:

```
plain-text response to a specific request\r\n
OK\r\n
```

The output coming from a node is considered a response to the request until the sequence `\r\nOK\r\n` is received. After the terminating mark the node is considered to be ready to accept new requests.

In case an error occurs, the line before the terminating mark should signal the cause of the error. On the transport layer two errors are defined: incorrect input formatting and CRC mismatch. The first type of error signals that the received request does not match any of the two request formats presented above. This can happen, for example, if the first line of the request gets corrupted or if the user tries to send a request to a not existing resource. After such error, five new lines (`\r\n\r\n\r\n\r\n\r\n`) have to be sent before a new request. This feature has been added to prevent the execution of posted data as requests. The CRC mismatch error can be generated as a response to a POST error, if the CRC in the request does not match expected CRC value.

6.5.2 Application Layer

On the application layer, several resources have been defined. This subsection briefly lists the resources relevant for spectrum sensing.

- `sensing/deviceList`: lists the radio devices installed on the specific node in the testbed.
- `sensing/deviceConfigList`: lists the available configurations and the parameters of the configurations for all of the installed radios or for one radio.
- `sensing/deviceStatus`: returns the status of all or one radio devices. Can be used for debugging VESNAs in the testbed.
- `sensing/program`: Resource in which sensing activities can be POST-ed. A sensing activity is defined by the radio which should be used for sensing, the used configuration of the radio, the channels on which sensing should be performed, the timing of the sensing and the location on the SD card where the results should be saved. Sending GET request to this resource returns the currently active sensing activities.

- `sensing/activeProgram`: Returns the currently active sensing activity, if there is any.
- `sensing/lastSweepBin`: Returns the lastly completed sweep in the currently active sensing activity in binary representation; useful for querying the status of a long-running sensing scenario.
- `sensing/lastSweepText`: Returns the lastly completed sweep in the currently active sensing activity in textual, human-readable representation.
- `sensing/quickSweepBin`: When a valid sensing configuration is POST-ed to this resource, the node performs one sensing sweep with the specified configuration and returns the results of the sensing in binary representation.
- `sensing/quickSweepText`: Same as above, but it returns the sensing results in human-readable textual form.
- `sensing/slotInformation`: Returns the status of a measurement result storage slot. A slot can be empty or it can hold up to 1 MB of data.
- `sensing/slotDataHeader`: Returns the metadata associated with spectrum sensing results. This data is composed of the parameters of the sensing activity that resulted the measurements saved in the specific slot.
- `sensing/slotDataBinary`: Returns the saved measurement results from a given data storage slot in binary form. Up to 512 bytes can be returned as an answer to one request. The request can specify the starting position and length of the data of interest, so larger amounts of data can be downloaded in multiple requests. A CRC value is appended to the returned data.
- `sensing/freeUpDataSlot`: The data slot specified in the argument is marked as empty if the value 1 is POST-ed in this resource. For avoiding potential data loss, new measurement results can be only saved to empty slots, so freeing up slots is necessary.

6.6 Management and access control system

The management and access control system part of the testbed, located in the premises of JSI, provides the web based user interface of the testbed. It has the following functions:

- Maintains an SSL connection with both clusters of the testbed;
- Performs testbed user authentication and experiment scheduling;
- Provides high-level experiment-oriented interface;
- Provides direct command access user interface for interacting with resources located on VESNAs in the testbed;
- Provides performance monitoring for all of the deployed nodes.

The functional block diagram of the management and access control system is depicted in Figure 6.18. The management and access control system is listening on a predefined set of ports and it waits for the coordinators of the deployed clusters to connect. When the coordinator is connected, the established SSL connection is used for sending requests to nodes. The HTTP to HTTP like protocol (HLP) translator block translates HTTPS

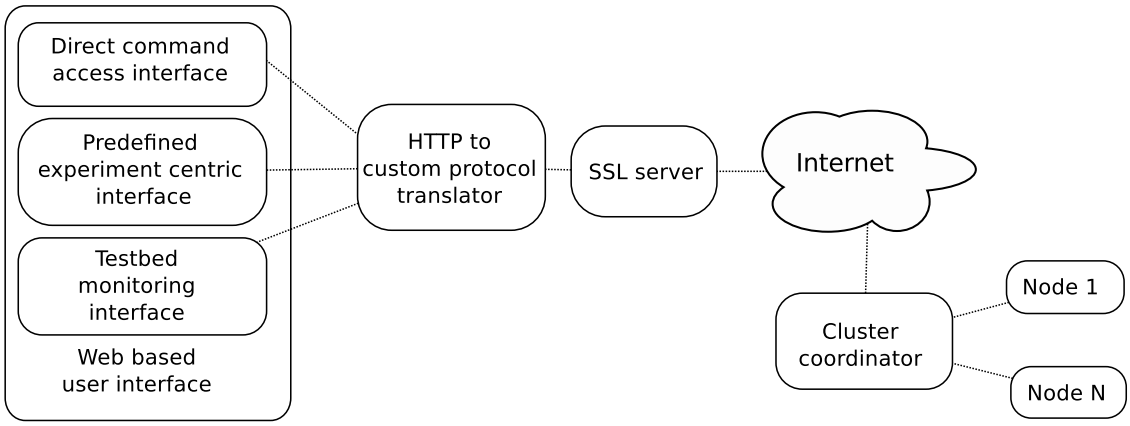


Figure 6.18: Block diagram of the management and access control system part of the testbed

requests to HLP protocol requests and returns the request results as HTTPS responses. The resources and the format of the transferred data is the same as in the HLP protocol. The HTTPS protocol is HTTP protocol over transport layer security (SSL/TLS). This means that the connection to the HTTP to custom HLP protocol translator is encrypted and user authentication is possible. The protocol translator block takes into account the scheduling of the users and only allows the scheduled users to access the testbed.

The web based user interface in composed from three major parts: the direct command access interface, the predefined experiment centric interface and the testbed monitoring interface. These interfaces are presented in the following.

The direct command access user interface of the testbed allows users to directly interact with the resources of the nodes in the testbed in a web-based graphical environment. This interface is presented in Figure 6.19. This user interface is divided into two parts. The left

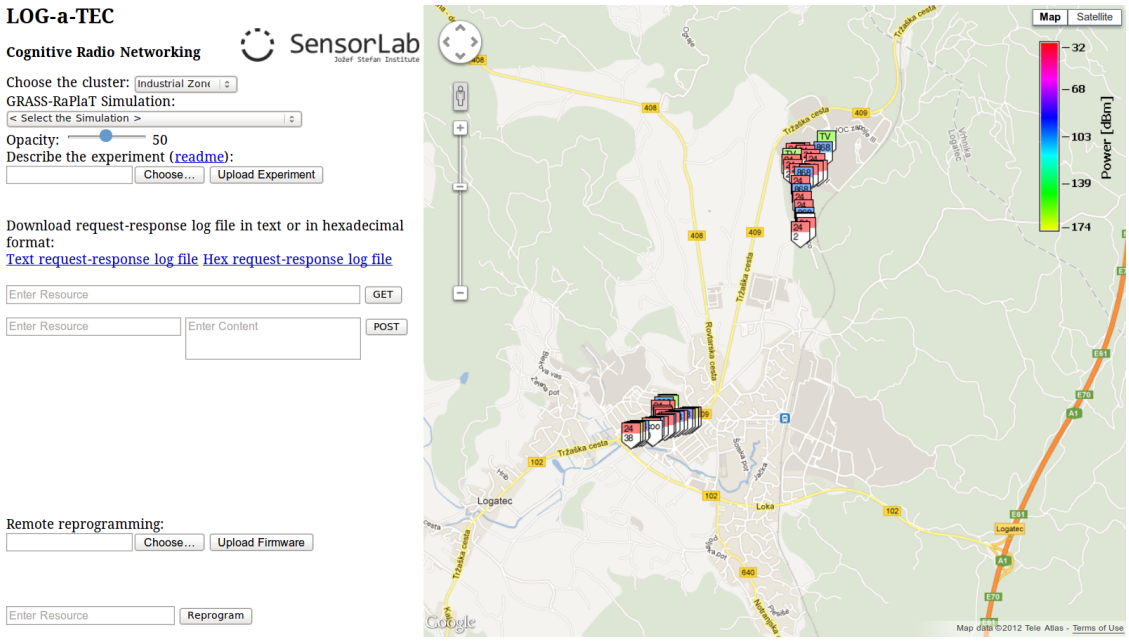


Figure 6.19: Web-based testbed user interface

part contains controls for interacting with the testbed, including the cluster selector, the simulation result viewer, a very basic experiment automation interface, fields for directly accessing resources and partial reprogramming automation. The right part shows a map of Logatec, based on Google Maps, with markers showing the location and type of the deployed nodes. In the future, the markers are planned to have more functionality. The scale for signal

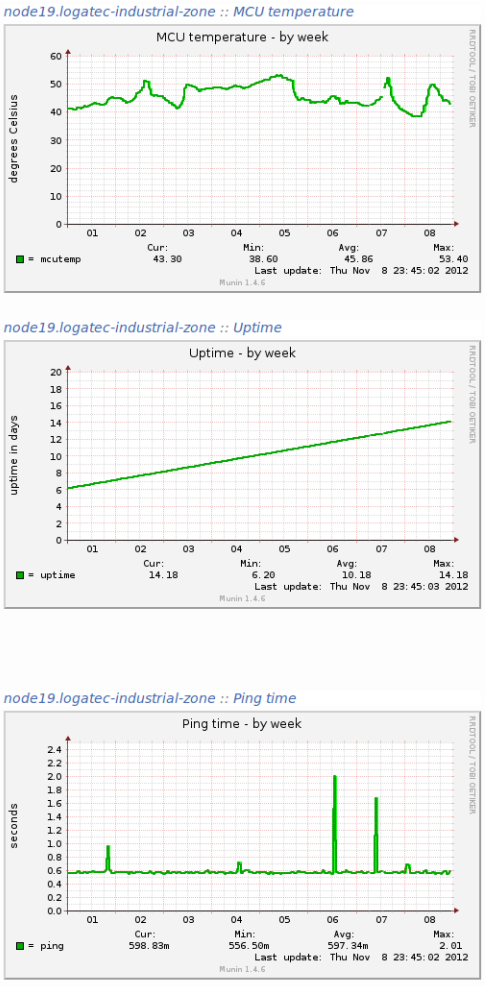
power value is used when viewing simulation results provided by additional tools which are beyond the scope of this thesis.

In contrast to the direct command access interface, the predefined experiment centric user interface does not allow low-level access to the resources related to remote reprogramming. This predefined experiment centric user interface only allows the selection between a set of experiments and the adjustment of a few parameters of the experiments. Predefined experiments include sensing with nodes in a given band, or transmitting with a subset of nodes and receiving with another subset.

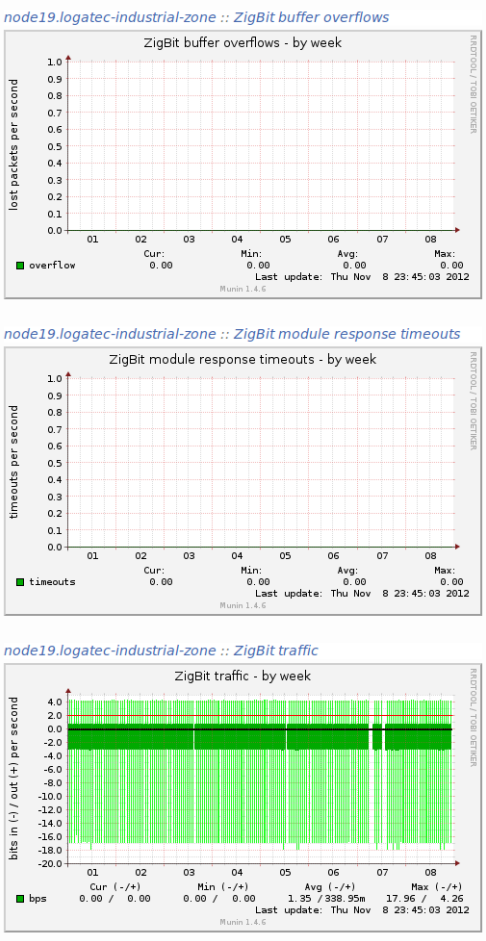
The testbed monitoring user interface shows the data collected by the testbed monitoring system running in the background. This system collects statistical data about the functioning of the testbed.

This data includes:

- temperature of the microcontroller of the node;
- the time since the last restart of the node;
- ping time of the node;
- number of errors occurred while transmitting data through the management network;



(a) Microcontroller temperature, time since last restart and ping time of node 19



(b) Number of error while transmitting data, number of timeouts in the management network and amount of data transmitted and received

Figure 6.20: Monitoring system interface for node 19

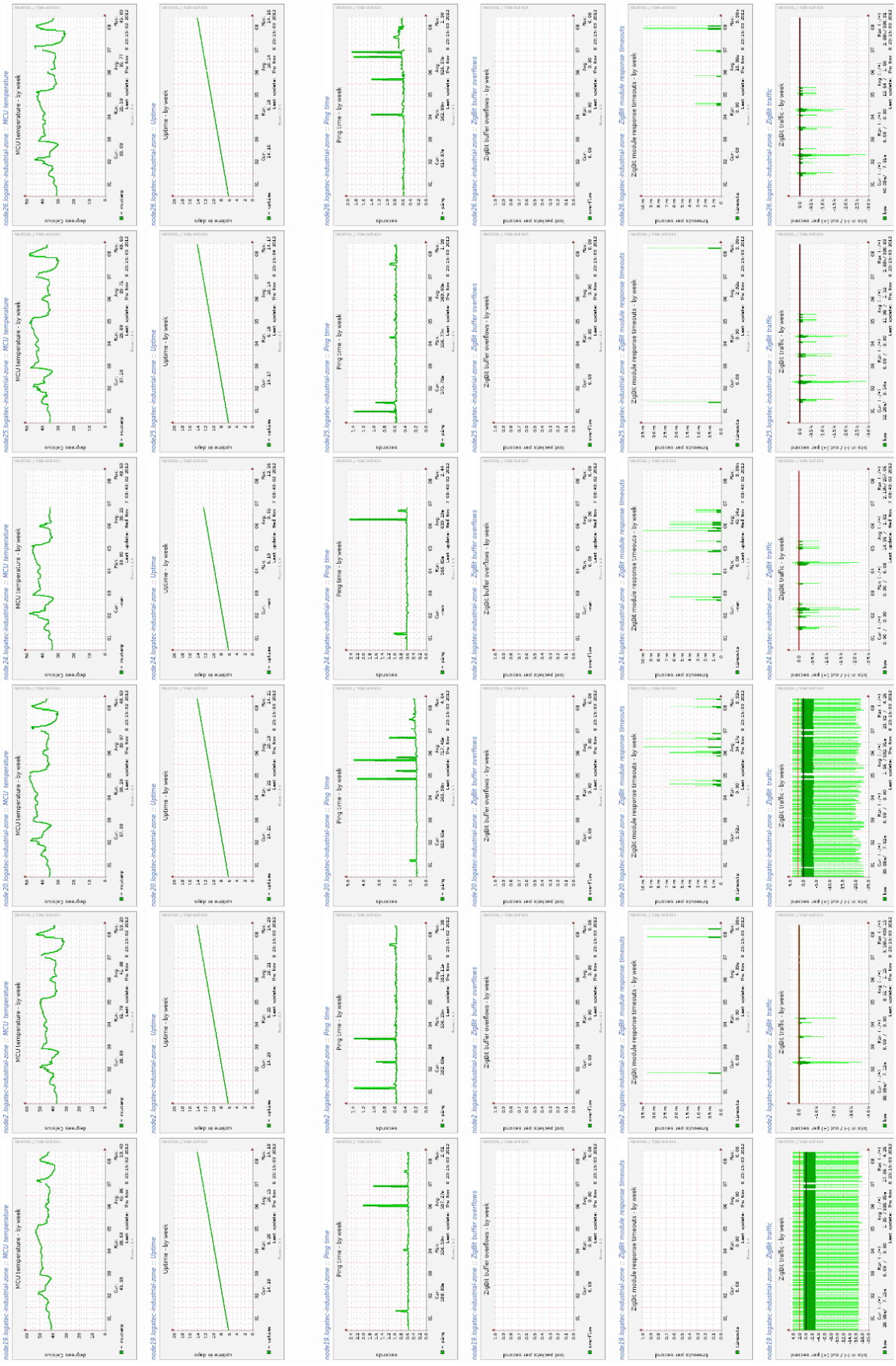


Figure 6.21: Monitoring system interface

- number of timeouts in the management network;
- amount of data transmitted and received on the management network.

Examples of the collected statistics for one node are presented in Figure 6.20a and Figure 6.20b.

The collected monitoring data can be used for evaluating the status of the testbed. Internally, the monitoring system requests the values stored in a set of resources and stores it. The system is based on the open source Munin monitoring tool (Munin monitoring, 2012). The overview of the interface showing various statistics from the deployed testbed can be observed in Figure 6.21. Columns in the figure belong to different nodes, while rows correspond to the collected data types enumerated above, in the same order. The user interface has been zoomed out until all collected data types become visible, hence the letters on it are very small.

6.7 Experimenting

Earlier sections have presented the internal structure of the testbed; this section presents how the testbed can be used. There are two types of experiments that can be performed with the testbed: predefined experiments and advanced experiments. Predefined experiments can be set up and run easily, but they do not offer much flexibility to the user. For example, sensing in a specified frequency band with a set of nodes can be predefined as a simple experiment. In advanced experiments users can directly access a subset of the resources available on the nodes through the HTTPS to HLP translator, so they have more control but they have to know details about the testbed's internal functioning. For example, cognitive radio experiments testing various algorithms, where nodes react to changes in their environment, have to be implemented as advanced experiments, because the tested algorithm on one hand has to access the spectrum sensing results, and on the other hand it has to be able to command the nodes. Because HTTPS is implementation-independent, the programs communicating with the low-level HTTPS interface can be developed in any technology.

Currently we have predefined several experiments with the aim to test the entire setup and to showcase the functioning of the testbed. For advanced experiments there exists a reference implementation of spectrum sensing in Python programming language, which can be used as a starting point for further work.

6.7.1 Spectrum sensing in advanced experiments

This subsection presents the reference implementation of spectrum sensing experiments using the advanced low-level testbed interface. The reference implementation consists of a library and several sample applications. The reference implementation has been developed in Python programming language. The library abstracts the functionality of the nodes in the testbed in an object-oriented manner; requests can be issued to a node by calling methods of node type objects. The library facilitates the development of spectrum sensing experiments by providing general spectrum sensing classes, which set up the radio available on nodes, run the sensing process and collect the measurement results.

The provided applications built on the top of the bundled libraries include:

- several predefined spectrum sensing experiments; the data collection is performed automatically and the results are provided in an easily reusable format;
- application to reprogram a given node with a given firmware image;

- application to measure the link quality in the management network between pairs of nodes;
- application providing an interactive terminal with the possibility of directly interacting with the nodes from the testbed with a high level interface.

By running the sample applications, experiments can be easily performed. For small changes in the scenarios the samples can be adjusted as needed.

6.8 Discussion

This chapter has presented:

- the architecture of the newly setup testbed;
- the differences between sensing with a stand-alone device and a group of distributed devices with common control;
- the network planning, defining the locations and types of the deployed device;
- the communications protocols used in the testbed;
- the management and access control system;
- the example experiment implementation.

While setting up the testbed, several unexpected difficulties have appeared. A part of them is related to the used ZigBee modules. These modules perform well in theory, but in practice their 1 kB/s effective throughput is less than what has been expected; this low throughput is the main limiting factor for the number of requests that the testbed can process. Another unexpected feature of the ZigBee mesh network is that if the network is used at its maximum capacity, some ZigBee nodes randomly lose connectivity with the network and cannot join back until the network becomes idle. The biggest problem with the ZigBee modules is that under certain circumstances they provide incorrect output format on their UART interface. When a radio packet is received by the ZigBee module at the exact same time when a command is sent to the module on its UART interface, then the module fails to send the termination characters for response to the command received on its UART, and it starts sending the data received from the radio. In some cases messages have to be dropped because they are not recoverable from the resulting output. A lot of time and effort has been spent searching for the cause of issues with the ZigBee network and developing workarounds in software. The support team from the manufacturer of the ZigBee modules, Atmel, did not prove very helpful in this process.

On higher protocol levels we realized that it would have been simpler to include CRC values in all requests and responses, because for many resources we needed to add CRC value on the application level in order to ensure data integrity. In the design of the transport protocol a clearer response format and error signaling specification would have reduced the development effort and would have increased the maintainability of the firmware source code.

Overall, the testbed is functioning correctly. Because flexible design approaches have been taken, the major problems mentioned above have been successfully worked around.

7 Experiments with the testbed

In this chapter the usage of the LOG-a-TEC testbed is demonstrated by performing an experiment. The nodes participating in the experiment do not take decisions, but just follow the predefined script of the experiment. The goal of this experiment is to demonstrate the use of the testbed for distributed spectrum sensing. Given the current status of the testbed, other types of experiments can also be developed, but they are beyond the scope of this thesis.

Section 7.1 presents how this scenario has been implemented in the LOG-a-TEC testbed, Section 7.2 details the experiment results and finally Section 7.3 discusses the experiment procedure.

7.1 Experiment implementation on LOG-a-TEC testbed

This section presents the steps taken for implementing the proposed experiment. In the experiment, a spectrum sharing scenario is emulated in the 2.4 GHz ISM band. The industrial zone has been selected as the location to perform the experiments, because in that region less interference is expected in the 2.4 GHz band than in the city center.

In the spectrum sharing scenario, two terminals are participating. The first terminal has cognitive radio capabilities, while the second one does not. By cognitive radio capabilities we understand that a terminal is able to scan the radio spectrum for unused frequency bands and if needed it can take decision to use one. Initially the first terminal is transmitting a continuous signal in a given frequency band. Then the second terminal starts transmitting in a frequency band that overlaps with the one used by the first terminal, thus the two transmissions interfere. The first terminal detects the interference, scans the radio spectrum for a usable frequency band and starts using a different frequency band, which does not overlap with the one used by the second terminal. This scenario is observed by multiple nodes in the testbed.

First, in the 2.4 GHz frequency band, the propagation between nodes of the testbed has been empirically characterized. Based on the results from the characterization, transmitting and receiving nodes have been chosen for the experiment. Next, the timing of the experiment scenario has been defined and the scenario has been implemented by using the Python libraries presented in Chapter 6. After creating the script for the experiment scenario, the experiment could be performed by running the script and analyzing the collected data.

7.1.1 Radio wave propagation at 2.4 GHz in the testbed

For setting up the testbed, radio link quality has been measured between possible locations of the deployed nodes in the 868 MHz ISM band (see Section 6.4 in Chapter 6). The range of 2.4 GHz links have been only estimated at that point. For performing spectrum sensing during an experiment in the 2.4 GHz band, it is necessary to find the pairs of nodes in the

testbed, for which a transmission started at the first node of the pair can be detected by the second node of the pair. By having this data, roles of nodes in experiment can be assigned easier.

For determining if the transmission of one node can be detected by another node, a set of measurements have been performed. For each operational node, a testing scenario has been performed in which the selected node has been transmitting a signal, and all the other nodes have been receiving. The transmitted pattern has been the following: continuous transmission at 2422 MHz for 25 seconds, followed by 5 seconds silence and finally a continuous transmission at 2445 MHz for 20 seconds. The bandwidth of the transmitted signal has been 200 kHz and the transmission power has been set to 0 dBm.

In these testing scenarios we used a subset of 10 nodes, so the total number of obtained traces has been $10 \times 9 = 90$ traces. For simplicity, only summarized results are presented below. Based on the signal power of the received test pattern, three categories of links have been defined:

- good link quality: the test pattern is the most clear signal pattern in the received data; this translates to approximately at least 20 dB of signal power above the noise floor. An example of results from such link is presented in Figure 7.1.
- weak link: the test pattern is easily observable, but it is not very clear; the received pattern is approximately 10 dB over the noise floor. Example results can be observed in Figure 7.2.
- very weak link: the test pattern is barely distinguishable from the noise floor. An example of measured data is plotted in Figure 7.3.

Using the defined radio link categories, Table 7.1 summarizes the measurement results. Note that the numbers of nodes used in this chapter are different from the location numbers used in Chapter 6. The transmission of each node can be detected by at most four other

Number of transmitting node	List of receivers by reception quality		
	Good	Weak	Very Weak
2	25	17	6
4	26		17
6	13	17	2, 26
11			
13	6, 17		15, 11
15		11	
17		6, 13, 25	2
24			
25	17	2	
26	4	6, 25	11

Table 7.1: Quality of received test transmissions for operational nodes

nodes. In worst case, the transmission of a node cannot be detected by any other node, for example, in case of nodes 11 and 24. Figure 7.4 shows that the closest neighbors of these nodes are 15 and 26, respectively. The transmissions of the latter two nodes are detected by the former two nodes, but instead of receiving strong signals, only weak signals have been detected. From this fact we realized that transmission of nodes 11 and 24 was malfunctioning and required repair. Nonetheless, as shown in Table 7.1 there were sufficient number of other node combinations to carry out the experiment.

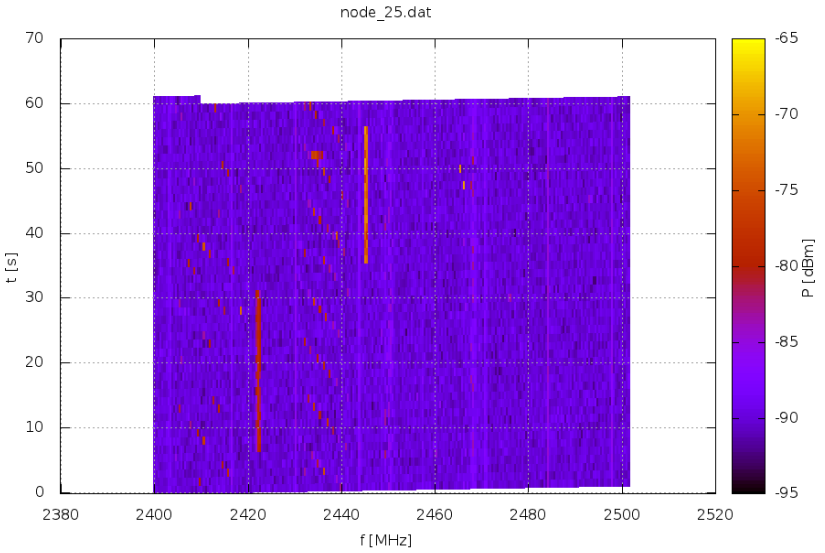


Figure 7.1: Measurement results for a good radio link quality

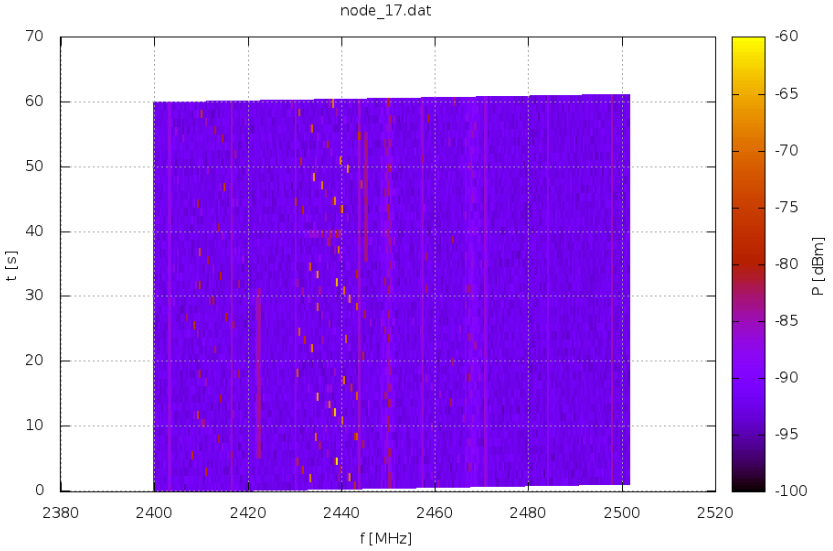


Figure 7.2: Measurement results for a weak radio link quality

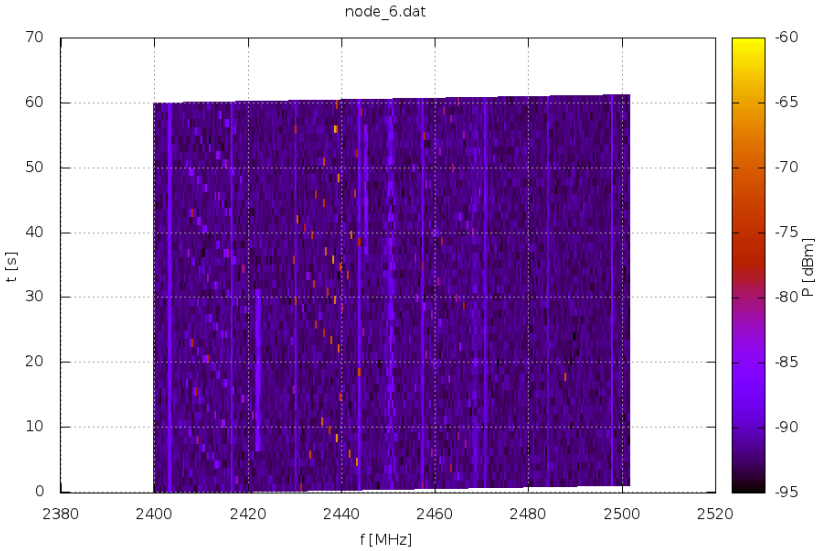


Figure 7.3: Measurement results for a very weak radio link quality

7.1.2 Selecting nodes for the experiment

Based on the measured link quality summarized in Table 7.1, nodes can be selected for usage in the spectrum sharing experiment. As it can be seen from the table, nodes 2 and 17 can reciprocally detect each other's transmission. Node 25 can observe the transmission of both node 2 and node 17, while node 6 can observe node 2, and node 13 can observe node 17. There are other possible configurations for the experiment, but in other cases each of the receiving nodes can observe the activity of one transmitter, but not the two transmitters at the same time. During the experiment the rest of the operational nodes, the ones not mentioned above, have been set up as receivers, because they might capture useful data.

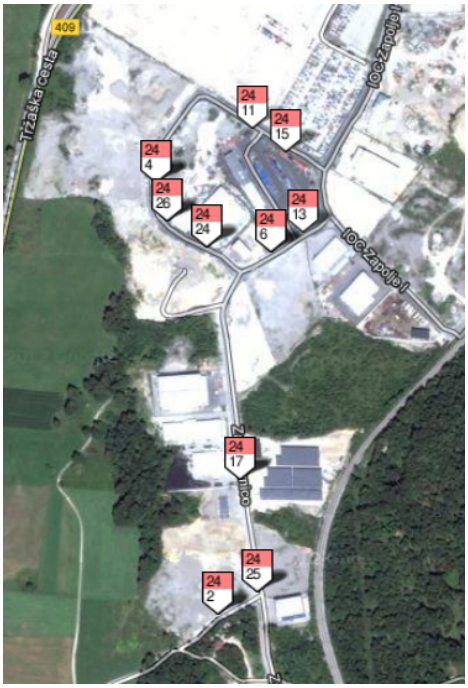


Figure 7.4: Locations of the nodes selected for experiment

In the experiment, node 17 was defined to act as the terminal with cognitive radio capabilities and node 2 to act as the terminal without cognitive radio capabilities. The positions of the nodes selected for the experiment are depicted in Figure 7.4. Nodes 2 and 17 are located in the southern part of the selected group, close to node 25. Tests have shown that in the direction defined by nodes 2, 17, 6 and 13, the radio propagation has small losses, possibly because the installed nodes have a line of sight connection. Hence, it is expected that nodes 25, 13 and 6 will be able to clearly receive the transmissions of nodes 2 and 17.

During preliminary tests, Wi-Fi activity has been detected on the Wi-Fi channels 1, 6 and 11, so for avoiding interference with those transmissions, the center frequencies of 2422.0, 2422.8 and 2445.0 MHz have been used in the experiments. The used channel bandwidth has been 200 kHz. The difference of 800 kHz has been used between the two interfering transmissions because (a) the difference is small enough to generate significant interference between the two and (b) the two central frequencies have enough distance between them for being easily differentiable in the results.

7.1.3 Scripting the scenario

The diagram in Figure 7.5 graphically presents the conceptual timing of the planned events in the experiment. First the terminal with the cognitive radio capabilities starts transmitting 5 seconds after the start of the experiment. Then, 20 seconds later, a terminal without

cognitive radio capabilities starts a transmission in a very close frequency band, thus starting to cause interference. As a result of interference, the first terminal stops transmitting after 5 seconds since the start of the interference and then it moves its transmission to another unoccupied frequency channel after another 5 seconds. In second 55 of the experiment both terminals stop transmitting. Sensing is performed from the beginning of the experiment for 60 seconds, then the experiment ends.

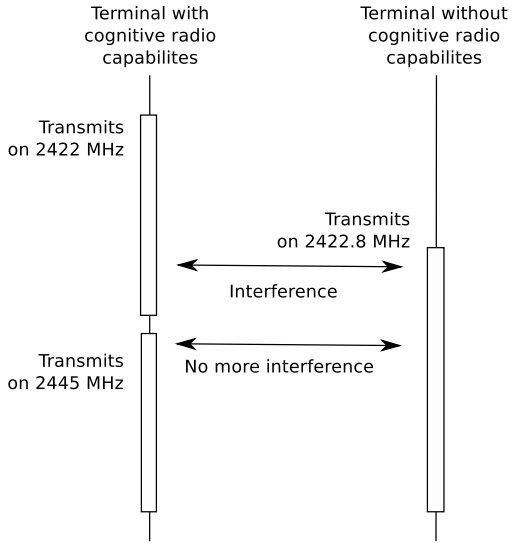


Figure 7.5: Timing diagram of the experiment

For implementing the experiment a python script has been created. This script initiates communication with the testbed, sets up the experiment, runs the experiment and finally downloads the collected data from the nodes. The part that is setting up the experiment is listed below.

```
def setup_experiment(time_start_arg, nodef):
    # cognitive terminal, default frequency
    SignalGenerationRun(                                # (1)
            alh = nodef(17),
            time_start = time_start_arg + 5.0,
            time_duration = 25,
            device_id = 0,
            config_id = 0,
            channel = 110,
            power = 0).program()

    # cognitive terminal, moved frequency
    SignalGenerationRun(                                # (2)
            alh = nodef(17),
            time_start = time_start_arg + 35.0,
            time_duration = 20,
            device_id = 0,
            config_id = 0,
            channel = 225,
            power = 0).program()

    # non-cognitive terminal's frequency
```

```
    SignalGenerationRun(                                # (3)
            alh = nodef(2),
            time_start = time_start_arg + 25.0,
            time_duration = 30,
            device_id = 0,
            config_id = 0,
            channel = 114,
            power = 0).program()

    return MultiNodeSpectrumSensingRun(                 # (4)
            nodes = [nodef(4), nodef(6), nodef(11),
                    nodef(13), nodef(24), nodef(25),
                    nodef(26) ],
            time_start = time_start_arg,
            time_duration = 60,
            device_id = 0,
            config_id = 0,
            ch_start = 0,
            ch_step = 1,
            ch_stop = 255,
            slot_id = 6)
```

In the script three signal generation activities and one sensing activity have been defined. The transmission power in all transmissions is 0 dBm. Two signal generation activities, marked with `(1)` and `(2)` in the listing, set up node 17 for transmitting as the terminal with cognitive radio capabilities. The transmission in performed with the transmission device 0, the CC2500 radio, with configuration 0, corresponding to the configuration with 2.4 GHz base frequency, channel spacing and channel bandwidth of 400 kHz. From the settings on the configuration it follows that channel 110 has center frequency of $2400 + 110 \times 0.2 = 2422$ MHz, while channel 225 has the center frequency of $2400 + 225 \times 0.2 = 2445$ MHz. The third signal generation activity, `(3)`, sets up node 2 for transmission on channel 114, corresponding to the center frequency of $2400 + 114 \times 0.2 = 2422.8$ MHz.

The sensing activity defined in the script, `(4)`, uses nodes 4, 6, 11, 13, 24, 25 and 26 for sensing with their CC2500 radio in the 2400 - 2502 MHz frequency band for 60 seconds. The collected data should be stored in slot 6 of the SD cards installed on the nodes. Device 0 and configuration 0 for sensing corresponds to the CC2500 radio, set up with base frequency of 2.4 GHz, channel spacing and channel bandwidth of 400 kHz. Hence, channel 0 and 255 have the center frequencies of $2400 + 0 \times 0.4 = 2400$ MHz and $2400 + 255 \times 0.4 = 2502$ MHz, respectively.

In the implementation the timing of the experiment is relative to the `time_start_arg` argument given to the presented function. Sensing is started at the moment specified by the argument and it lasts for 60 seconds (`time_duration` argument). The starting moment of transmissions are specified by the values of the `time_start` arguments, while their duration is given by the `time_duration` arguments.

7.2 Experiment results

By running the script of the experiment, the proposed experiment has been performed, and results have been collected.

As expected based on testing experiments, the proposed scenario has been best observed by node 25. The data collected by this node is plotted in Figure 7.6. In this figure both

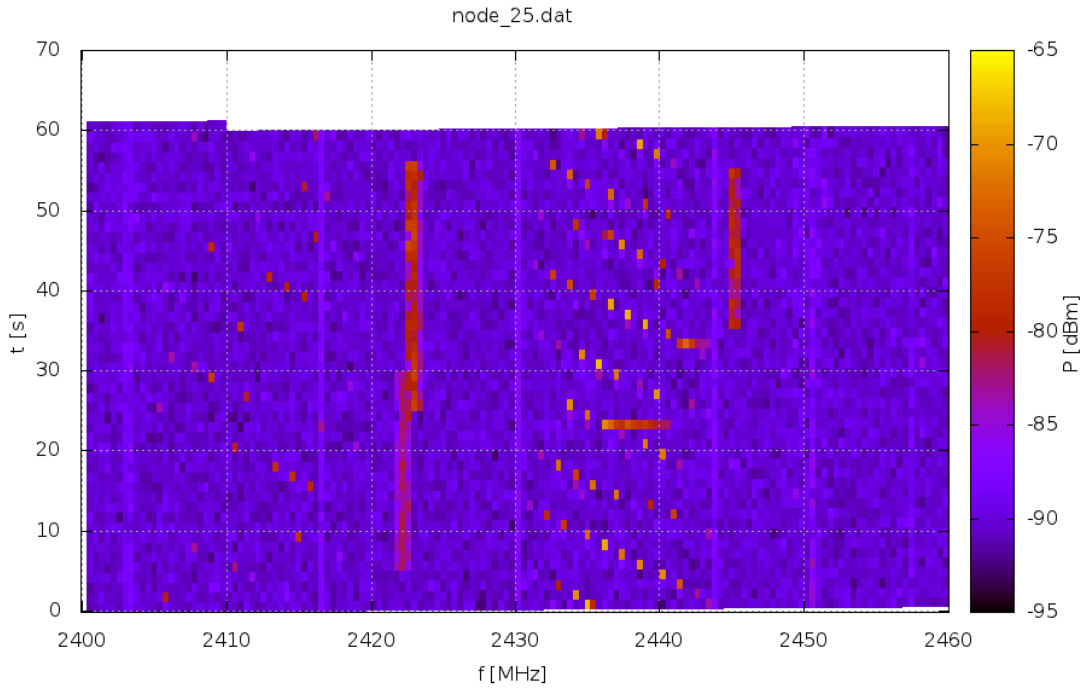


Figure 7.6: Experiment results from node 25

the transmission of the terminal with cognitive capabilities, node 17, and the transmission of the terminal without cognitive capabilities, node 2, can be easily observed. The periodic frequency hopping transmissions at around 2435 MHz are very likely Wi-Fi transmissions. Note that time flows from the bottom towards the top of the figure. First, the terminal with cognitive radio capabilities starts transmitting on 2422 MHz in second 5. Next, in second 25, a second transmission appears very close to the initial one, at 2422.8 MHz. This transmission has a slightly larger signal power than the original transmission and together with the first transmission it creates a thicker red line in the figure, because its central frequency is greater than the central frequency of the first transmission. This second transmission is coming from the terminal without cognitive radio capabilities.

In second 30 the first terminal stops its transmission, because it has detected the transmission of the second terminal. Due to the stopped transmission the red vertical line in the figure at 2422 MHz becomes as thin as the initial transmission, but its center is moved towards slightly higher frequencies.

Next, in second 35 the first terminal starts transmitting on 2445 MHz, so it does not interfere with the second terminal. The detected power of the transmission is very similar to the original transmission power, and it is smaller than the signal power of the second transmission.

Nodes 6 and 13 have recorded similar data to node 25, plotted in Figure 7.7 and Figure 7.8. These nodes have been able to receive only the transmissions of the terminal with cognitive radio capabilities, implemented with node 17 in the testbed. Because there is no interfering transmission, the behavior of node 17 can be easily observed. The short bursts around 2410, 2435 and 2465 MHz are likely Wi-Fi transmissions.

Some nodes have not been able to receive any of the two transmissions. As example the results from nodes 24 and 26 are presented in Figures 7.9 and 7.10. Both nodes have detected transmissions looking like Wi-Fi, although the results from node 24 show surprisingly little

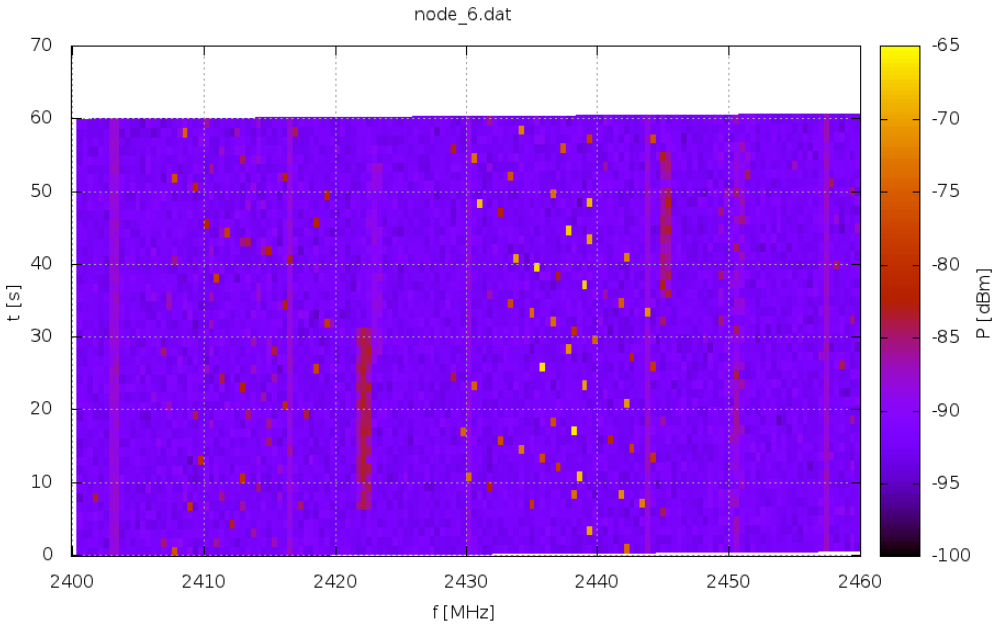


Figure 7.7: Experiment results from node 6

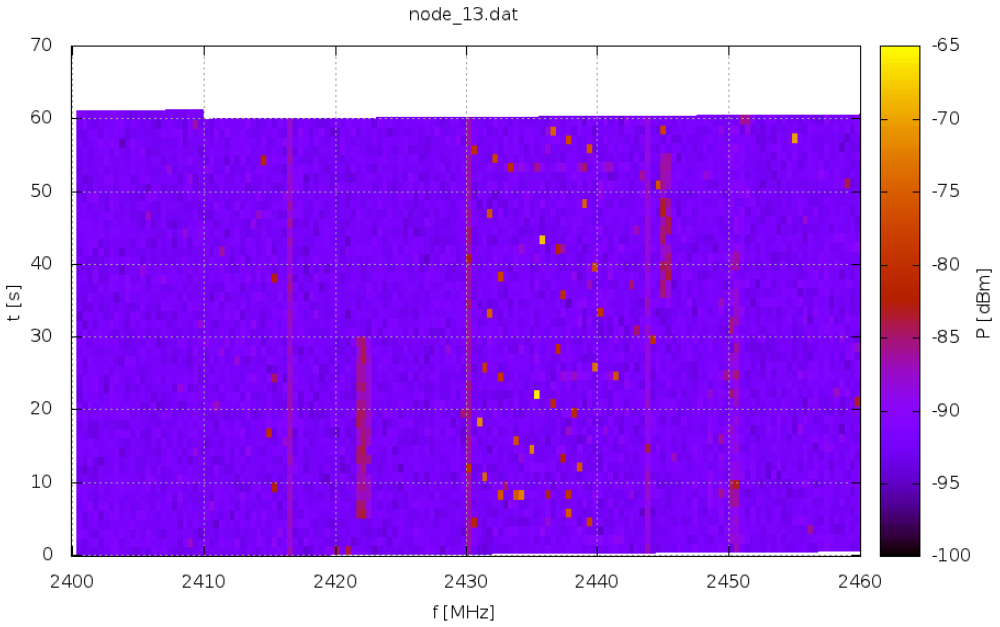


Figure 7.8: Experiment results from node 13

noise. As has been mentioned earlier in this chapter, node 24 had to be investigated, because its results differ from the results of the other nodes, and because its transmissions cannot be detected by any other node.

7.3 Discussion

This chapter has presented the implementation of an experiment in the spectrum sensing testbed. The experiment mimics a spectrum sharing scenario, in which a terminal with cognitive radio capabilities detects a second, not so advanced terminal, and it starts using a

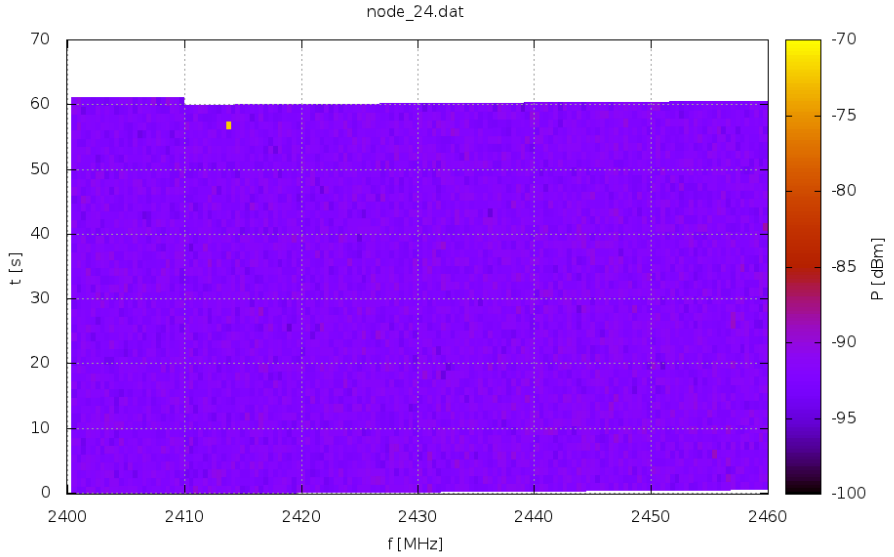


Figure 7.9: Experiment results from node 24

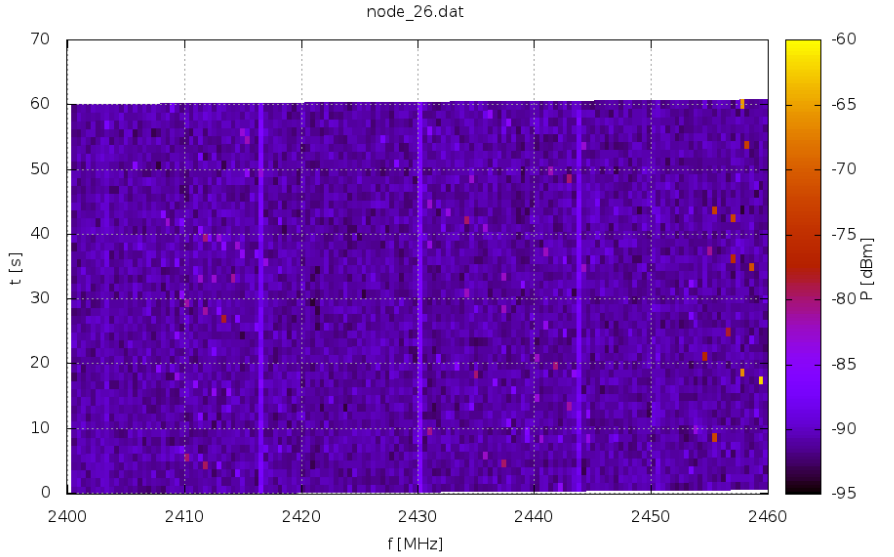


Figure 7.10: Experiment results from node 26

different frequency band in order to avoid interference.

Running an experiment in a newly established spectrum sensing testbed requires a very cautious approach, because radio propagation is affected by many factors, and estimating propagation is sometimes difficult. While preparing an experiment, first a link measurement experiment had to be performed for finding the link quality of the CC2500 radios between nodes. While this activity could be easily automated, it still took one day to run all link measurements. Having a secure interface accessible from the entire Internet proved to be very useful in this task because experiments could be performed from any location and at any scheduled time.

Because the testbed is located in an outdoor uncontrolled environment, it would be interesting to investigate the relation between the weather conditions the testbed location and the radio propagation. The presented experiments have been performed in rainy and cloudy conditions, in which the large amount of water in the air could have created significant fading.

The buildings around nodes 26 and 15 have an easily observable effect on radio propagation, because the mentioned nodes can only detect transmissions from their neighbors. In contrast, transmissions of the distant nodes 2 and 6 are reciprocally observed by each other, although the two nodes being located several public light poles away from each other.

The presented experiment has shown that the testbed is correctly functioning. During the experiment the ongoing transmission could be observed from multiple locations. Based on the foundations of this experiment, more complex scenarios can be implemented in the future.

8 Conclusions

This thesis has presented the design and implementation of a spectrum sensing testbed based on low-cost and low-complexity VESNA platform. This chapter summarizes the conclusions of the thesis and briefly describes directions for future work.

First, the state of the art in spectrum sensing methods have been surveyed. Based on the requirements of different sensing algorithms, it has been concluded that the energy detection sensing method is the most suitable for our purposes. Other detection techniques involve direct processing of received signal samples, which is not possible with the employed radio hardware.

Next, a stand-alone spectrum sensing prototype has been developed, which uses a specific radio module and software application. This prototype consists of two parts: the sensing part implemented on VESNA and the control and data collection part implemented on a PC. This architectural pattern has also been used when designing the testbed consisting of multiple distributed sensing devices, because it allows a flexible development procedure in which functionality is gradually added to the sensing part of the system.

This stand-alone spectrum sensing prototype has been calibrated and integrated in a heterogeneous sensing system. In this system, it has been used to collect data about received signal power at multiple locations in a laboratory, for several locations of a continuously transmitting signal generator. By presenting obtained experimental results, correct functioning of the calibrated sensing prototype has been demonstrated. In a subsequent experiment in more controlled environment, the accuracy of the stand-alone prototype has been determined by measuring a known continuous signal with multiple devices at multiple locations and comparing the results of different device types by using a very simple propagation model. The prototype has been the second most close to the average of all measurements and also its results have been the second most self-consistent.

The architecture used for integrating multiple VESNA devices in a distributed spectrum sensing testbed shows similarity with the architecture of the sensing prototype. It consists of two parts: the testbed part composed of multiple VESNA devices and the management and control system part. While in the sensing prototype, direct communication was established between a PC and the VESNA, in case of the testbed deployment, a more complex communication chain was necessary to transmit data between the VESNAs deployed in the testbed and the management and control system running on the remote server.

In order to select the locations of the VESNAs forming the testbed, extensive network design including on the spot measurements has been carried out for ensuring that the deployed system will function correctly. This task had to take into account a set of requirements and constraints imposed by the actual operating environment, such as the communication protocol used for establishing the network between deployed devices, distribution of locations of public infrastructure used for deployment of nodes, availability of power connections and availability of the Internet access. Eventually, the network design resulted in two separate clusters of the testbed with 25 VESNAs covering three different frequency bands, the UHF TV band and two ISM bands, one at 868 MHz and one at 2.4 GHz.

The complex communication systems and the requirement for the testbed to function

without the need of physical intervention have resulted in the development and use of a custom HTTP-like protocol (HLP), because the standardized alternatives would have required more effort to implement. All of the functions of the testbed are accessible through the HLP protocol; practically all experiments can be defined as a set of requests in the HLP protocol.

The thesis also presented an example of planning and executing a spectrum sensing experiment in the deployed testbed. In order to support easier and more correct experiment planning, radio link measurements have been performed to determine the coverage areas of individual transceivers. Next, an example experiment consisting of an emulated interference avoidance scenario has been planned and performed with a preselected set of nodes. The obtained experimental results are in line with the expected values, proving the correct functioning of the testbed.

8.1 Future work

The goal for setting up the distributed spectrum sensing testbed has been achieved. This section presents guidance for future improvements of the testbed and for future experimenters.

The main limiting factor of the testbed's performance is the throughput of the used Zig-Bee based mesh network between the devices. Upgrading this network to a faster technology could significantly reduce the time needed for performing experiments, especially downloading measurement results. Given a fast enough network, some collected data could be retrieved in real-time during experiments.

For improving long-term maintainability of the testbed, the currently used custom HLP protocol could be migrated to CoAP and eventually an embedded operating system could be used on the VESNAs in the testbed.

On the experimentation side, future work includes testing various algorithms usable in spectrum sharing and dynamic spectrum access scenarios. By implementing algorithms and systems that directly interact with the external interface provided by the testbed, it should be possible to perform real-time cognitive radio experiments.

Acknowledgements

This thesis would not have been possible without the support of many people.

First, I offer sincerest gratitude to my supervisor, Asst. Prof. Dr. Mihael Mohorčič, for his support and guidance throughout my master studies. I could not have wished a better supervisor.

Next, I would like to thank my working supervisor, Carolina Fortuna, for the valuable discussions we had and for helpful guidance.

In addition, I would like to acknowledge the members of my thesis committee, Asst. Prof. Dr. Tomaž Javornik and Prof. Dr. Gorazd Kandus for their support and constructive comments.

It is a pleasure to thank my colleagues and collaborators at the Jožef Stefan Institute, particularly SensorLab, who made this thesis possible.

I want to express my gratitude to the people who have contributed to the work presented in this thesis:

- Tomaž Šolc for numerous contributions:
  - the development of the spectrum sensing expansion board used in the testbed;
  - implementation of the data collection and control part of the stand-alone sensing prototype;
  - calibration of the CC1101 and CC2500 radios;
  - definition of the general software interface for the sensing radios;
  - implementation of the continuous signal generation function on the CC1101 and CC2500 radios;
  - contributions to the network design of the testbed;
  - implementation of the monitoring interface of the testbed;
  - reference implementation of the scripting application and library used in experiments;
  - many bug-fixes applied to practically all parts of the testbed.
- Matevž Vučnik for contributions to the design and development of the custom protocol used in the testbed and implementation of the management and access control part of the testbed.
- Marko Mihelin for the following:
  - development of the CC1101 and CC2500 radio boards used in the stand-alone prototype;
  - development of the Ethernet module;
  - development of ZigBee module based radio boards, used in the testbed;

  - initial implementation of the software support for the ZigBee modules;
  - implementation of general software support libraries for VESNA.

- Marko Kastelic for feedback and implementation contributions during the development of the custom protocol used in the testbed and for implementing the remote reprogramming functionality to the sensing testbed.
- Bostjan Mikelj for developing software support for the Ethernet modules used in the testbed.
- Adnan Bekan for contributions to the web-based user interface of the testbed.

I would like to express my gratitude to Patricia Oniga for the support, patience and understanding during these years.

Finally, I would like to thank my parents for their support during my studies.

References

Akan, O.; Karli, O. & Ergul, O. Cognitive radio sensor networks. *Network, IEEE* **23**, 34–40 (2009).

Akyildiz, I. F.; Lee, W.-Y.; Vuran, M. C. & Mohanty, S. NeXt generation/dynamic spectrum access/cognitive radio wireless networks: A survey. *Computer Networks* **50**, 2127–2159 (2006).

Atanasovski, V. *et al.* Constructing radio environment maps with heterogeneous spectrum sensors. In: *2011 IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*. 660–661 (IEEE, Aachen, Germany, 2011).

Atmel. ZigBit 900MHz Module with Balanced RF Output. `http://www.atmel.com/tools/ZIGBIT900MHZMODULEWITHBALANCEDRFOUTPUT.aspx` (accessed October 2012).

Atmel. AVR2050: Atmel BitCloud Developer Guide. `http://www.atmel.com/Images/doc8199.pdf` (accessed October 2012).

Atmel. AT86RF212, Low Power 700/800/900 MHz Transceiver for IEEE 802.15.4-2006, IEEE 802.15.4c-2009, Zigbee, 6LoWPAN, and ISM Applications. `http://www.atmel.com/Images/doc8168.pdf` (accessed October 2012).

Cabric, D. Experimental Study of Spectrum Sensing based on Energy Detection and Network Cooperation. In: *Proc. of the ACM 1st International Workshop on Technology and Policy for Accessing Spectrum (TAPAS)*. 12–12 (ACM, Boston, USA, 2006).

Cabric, D.; Mishra, S. M. & Brodersen, R. W. Implementation issues in spectrum sensing for cognitive radios. In: *Proc. the 38th Asilomar Conference on Signals, Systems, and Computers*. 772–776 (IEEE, Pacific Grove, CA, USA, 2004).

Cognitive Radio Experimentation World (CREW) project. CREW project website. `http://www.crew-project.eu/` (accessed October 2012).

Cognitive Radio Experimentation World (CREW) Project. Common data format. `http://www.crew-project.eu/content/common-data-format` (accessed October 2012).

Digi International Inc. Digi Connect ME® 9210 - System-on-Module w/ Integrated 10/100 Ethernet. `http://www.digi.com/pdf/pb_digiconnectme9210.pdf` (accessed October 2012).

Dunkels, A.; Grönvall, B. & Voigt, T. Contiki - a Lightweight and Flexible Operating System for Tiny Networked Sensors. In: *Proceedings of the First IEEE Workshop on Embedded Networked Sensors (Emnets-I)*. 455–462 (Tampa, Florida, USA, 2004).

Ergen, S. C. ZigBee/IEEE 802.15.4 Summary. `http://pages.cs.wisc.edu/~suman/courses/838/papers/zigbee.pdf` (accessed October 2012).

Ettus Research. Ettus Research LLC product list. http://www.ettus.com/products (accessed October 2012).

Ettus Research. Universal Software Radio Peripheral 2. http://www.olifantasia.com/gnuradio/usrp/files/datasheets/ds_usrp2.pdf (accessed October 2012).

Ettus Research. USRP datasheets. http://www.ettus.com/downloads/ (accessed October 2012).

FCC. Facilitating Opportunities for Flexible, Efficient, and Reliable Spectrum Use Employing Cognitive Radio Technologies, Notice of Proposed Rule Making and Order, FCC 03-322 (2003).

Fluke Networks. Airmagnet. http://Airmagnet.flukenetworks.com/assets/datasheets/Airmagnet_SpectrumXT_Datasheet.pdf (accessed October 2012).

GNU Radio project. GNU Radio website. http://gnuradio.org (accessed October 2012).

Golmie, N.; Rebala, O. & Chevrollier, N. Bluetooth adaptive frequency hopping and scheduling. In: *Proc. of Military Communications Conference 2003 (MILCOM'03), IEEE.* **2**, 1138–1142 (IEEE, Boston, MA, USA, 2003).

IEEE. IEEE Standard for Information Technology Telecommunications and Information Exchange Between Systems Local and Metropolitan Area Networks Specific Requirements Part 11: Wireless Lan Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 1: Radio Resource Measurement of Wireless Lans. *IEEE Std 802.11k-2008 (Amendment to IEEE Std 802.11-2007)* c1–222 (2008).

IEEE 802 LAN/MAN Standards Committee. IEEE 802.22 Working Group on Wireless Regional Area Networks. http://www.ieee802.org/22/ (accessed October 2012).

Ingels, M. *et al.* A 5mm2 40nm LP CMOS 0.1-to-3GHz multistandard transceiver. In: *Solid-State Circuits Conference Digest of Technical Papers (ISSCC).* 458–459 (IEEE International, San Francisco, CA, USA, 2010).

Internet Engineering Task Force. Constrained Application Protocol (CoAP). https://datatracker.ietf.org/doc/draft-ietf-core-coap/ (accessed October 2012).

Iyer, A. *et al.* SpecNet: Spectrum Sensing Sans Frontières. In: *8th USENIX Symposium on Networked Systems Design and Implementation (NSDI).* 26–26 (USENIX Association, Boston, MA, USA, 2011).

Kim, H.; Cordeiro, C.; Challapali, K. & Shin, K. G. An Experimental Approach to Spectrum Sensing in Cognitive Radio Networks with Off-the-Shelf IEEE 802.11 Devices. In: *Proceedings of the IEEE Workshop on Cognitive Radio Networks, in conjunction with IEEE CCNC.* (IEEE, Las Vegas, Nevada, USA, 2007).

Levis, P. *et al.* T2: A second generation OS for embedded sensor networks. Tech. rep., Telecommunication Networks Group, Technische Universitaet Berlin (2005).

MEMSIC. TelosB. http://www.memsic.com (accessed October 2012).

Mercier, B. *et al.* Sensor Networks for Cognitive Radio: Theory and System Design. *Electrical Engineering* (2010).

Metageek. Metageek Wi-Spy. http://www.metageek.com (accessed October 2012).

Mishra, S. *et al.* A real time cognitive radio testbed for physical and link layer experiments. In: *Proceedings of the first IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*. 562–567 (IEEE, Baltimore, Maryland, USA, 2005).

Mitola, J. & Maguire, G. Q. Cognitive radio: making software radios more personal. *IEEE Personal Communications* **6**, 13–18 (1999).

Munin monitoring. Munin. `http://munin-monitoring.org/` (accessed October 2012).

Nolan, K. *et al.* Dynamic Spectrum Access and Coexistence Experiences Involving Two Independently Developed Cognitive Radio Testbeds. In: *Proceedings of the 2nd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*. 270–275 (IEEE, Dublin, Ireland, 2007).

Open Air Interface project. AgileRF hardware platform. `http://www.openairinterface.org/agilerf` (accessed October 2012).

Padrah, Z.; Denkovski, D.; Gavrilovska, L. & Mohorčič, M. Integrating the low-cost low complexity VESNA spectrum sensor into a heterogeneous spectrum sensing platform. In: *Proceedings of 3rd COST IC0902 Workshop*. (COST, Ohrid, Macedonia, 2012).

Peha, J. Approaches to spectrum sharing. *Communications Magazine, IEEE* **43**, 10–12 (2005).

Polastre, J.; Szewczyk, R. & Culler, D. Telos: enabling ultra-low power wireless research. In: *Fourth International Symposium on Information Processing in Sensor Networks (IPSN)*. 364–369 (ACM, Los Angeles, California, USA, 2005).

Pollin, S. *et al.* Digital and Analog Solution for Low-Power Multi-Band Sensing. In: *2010 IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*. 1–2 (IEEE, Singapore, 2010).

Rice University. Wireless Open-Access Research Platform. `http://warp.rice.edu/` (accessed October 2102).

Roke Manor Research. The UK Frequency Allocations. `http://www.onlineconversion.com/downloads/uk_frequency_allocations_chart.pdf` (accessed October 2012).

SensorLab. People. `http://sensorlab.ijs.si/people.html` (accessed October 2012).

SensorLab. VESNA. `http://sensorlab.ijs.si/hardware.html` (accessed October 2012).

SensorLab. SensorLab VESNA open source development environment setup manual for Linux based development. `http://sensorlab.github.com/vesna-manual/` (accessed October 2012).

Shellhammer, S. Spectrum sensing in IEEE 802.22. In: *Proceedings of the 1st IAPR Workshop on Cognitive Information Processing (CIP)*. 9–10 (IEEE, Santorini, Greece, 2008).

Sokolowski, C.; Petrova, M.; de Baynast, A. & Mahonen, P. Cognitive Radio Testbed: Exploiting Limited Feedback in Tomorrow's Wireless Communication Networks. In: *Proceedings of the 2008 IEEE International Conference on Communications Workshops (ICC Workshops)*. 493–498 (IEEE, Beijing, China, 2008).

Sutton, P. *et al.* Iris: an architecture for cognitive radio networking testbeds. *Communications Magazine, IEEE* **48**, 114–122 (2010).

Texas Instruments. CC1101: Low-Power Sub-1 GHz RF Transceiver. http://www.ti.com/lit/ds/symlink/cc1101.pdf (accessed October 2012).

Texas Instruments. CC2500: Low-Cost Low-Power 2.4 GHz RF Transceiver. http://www.ti.com/lit/ds/symlink/cc2500.pdf (accessed October 2012).

Texas Instruments. SmartRF Studio. http://www.ti.com/tool/smartrftm-studio (accessed October 2012).

Texas Instruments. DN505 – RSSI interpretation and timing. http://www.ti.com/general/docs/lit/getliterature.tsp?baseLiteratureNumber=swra114 (accessed October 2012).

Texas Instruments. EZ430-RF2500, MSP430 Wireless Development Tool. http://www.ti.com/tool/ez430-rf2500 (accessed October 2012).

Texas Instruments. CC2420 2.4 GHz IEEE 802.15.4/ZigBee-ready RF Transceiver. http://www.ti.com/lit/gpn/cc2420 (accessed October 2012).

Valenta, V. *et al.* Survey on spectrum utilization in Europe: Measurements, analyses and observations. In: *2010 Proceedings of the Fifth International Conference on Cognitive Radio Oriented Wireless Networks & Communications (CROWNCOM)*. 1–5 (IEEE, Cannes, France, 2010).

Van Wesemael, P. *et al.* Robust distributed sensing with heterogeneous devices. In: *Proceedings of the 2012 Future Network & Mobile Summit*. (IEEE, Lisbon, Portugal, 2012).

Yan, Z.; Ma, Z.; Cao, H.; Li, G. & Wang, W. Spectrum Sensing, Access and Coexistence Testbed for Cognitive Radio using USRP. In: *Proceedings of 4th IEEE International Conference on Circuits and Systems for Communications (ICCSC)*. 270–274 (IEEE, Shanghai, China, 2008).

Yucek, T. & Arslan, H. A survey of spectrum sensing algorithms for cognitive radio applications. *IEEE Communications Surveys Tutorials* **11**, 116–130 (2009).

Zeng, Y. & Liang, Y.-C. Eigenvalue based Spectrum Sensing Algorithms for Cognitive Radio. *IEEE Trans Communications* **abs/0804.2** (2008).

Zhao, Q. & Sadler, B. A Survey of Dynamic Spectrum Access. *Signal Processing Magazine, IEEE* **24**, 79–89 (2007).

Index of Figures

Index of Tables

Index of Algorithms

Appendix A: List of publications

List of publications related to this thesis.

A.1 Publications related to this thesis

A.1.1 Published scientific conference contribution

Van Wesemael, P. *et al.* Robust distributed sensing with heterogeneous devices. In: *Proceedings of the 2012 Future Network & Mobile Summit.* (IEEE, Lisbon, Portugal, 2012).

Heller, C. *et al.* Spectrum sensing for cognitive wireless applications inside aircraft cabins. In: *2012 IEEE/AIAA 31st Digital Avionics Systems Conference (DASC).* (IEEE, Wiliamsburg, USA, 2012).

Liu, W.; Šolc, T. & Padrah, Z. A set of methodologies for heterogeneous spectrum sensing. In: *Proceedings of 2012 Wireless Innovation Forum European Conference on Communications Technologies (SDR'12-WInnComm-Europe).* (Wireless Innovation Forum, Brussels, Belgium, 2012)

Padrah, Z.; Šolc, T. & Mohorčič, M. VESNA based platform for spectrum sensing in ISM bands. In: *Proceedings of 4th Jožef Stefan International Postgraduate School Students Conference.* (Jožef Stefan International Postgraduate School, Ljubljana, Slovenia, 2012).

A.1.2 Published professional conference contribution abstract

Smolnikar, M.; Šolc, T.; Vučnik, M., Padrah, Z. & Mohorčič, M. Eksperimentalno senzorsko omrežje LOG-a-TEC za razvoj in preizkušanje principov kognitivnega radia. In: *Strokovni seminar Radijske komunikacije.* (Faculty of Electrical Engineering and Faculty of Computer and Information Science, Ljubljana, 2012).

A.1.3 Final research report

Smolnikar, M.; Šolc, T.; Padrah, Z.; Mohorčič, M. & Javornik, T. Definition of test scenarios and benchmarks for VESNA testbed. Project deliverable, D4.3. (Jožef Stefan Institute, Ljubljana, Slovenia, 2012).

Padrah, Z.; Šolc, T. & Mohorčič, M. Experimentation results of a second set of test cases, Project deliverable, D6.2. (Jožef Stefan Institute, Ljubljana, Slovenia, 2012).

Padrah, Z.; Šolc, T. & Mohorčič, M. Experimentation results of a second set of test cases. Project deliverable, D6.2. (Jožef Stefan Institute, Ljubljana, Slovenia, 2012).

Smolnikar, M.; Padrah, Z. & Vučnik, M. Methodology for performance evaluation. Project deliverable, D4.2. (Jožef Stefan Institute, Ljubljana, Slovenia, 2012).

Mohorčič, M. *et. al.* Definition of internal usage scenarios, federation functionality and the use federation as applicable to the VESNA-based testbed. Project deliverable, D2.4. (Jožef Stefan Institute, Ljubljana, Slovenia, 2011).

A.1.4 Treatise, preliminary study, study

Padrah, Z.; Denkovski, D.; Gavrilovska, L. & Mohorčič, M. Integrating the low-coast low complexity VESNA spectrum sensor into a heterogeneous spectrum sensing platform. In: *3rd Workshop of COST action IC0902 Cognitive Radio and Networking for Cooperative Coexistence of Heterogeneous Spectrum Sensing Platform.* (COST, Ohrid, Macedonia, 2012).

Fortuna, C.; Mihelin, M.; Padrah, Z. & Holland, O.T. A common data format for spectrum sensing information. In: *2nd Workshop of COST action IC0902 Cognitive Radio and Networking for Cooperative Coexistence of Heterogeneous Wireless Networks.* (COST, Barcelona, Spain, 2011).

A.2 Other publications

A.2.1 Published scientific conference contribution

Fortuna, C.; Oniga, P.; Padrah, Z.; Mohorčič, M & Moraru, A. Metadata management for the web of things: a practical perspective. In: *Proceedings of the 10th International Conference on Pervasive Computing and 16th International Symposium on Wearable Computing.* (ACM, Newcastle, UK, 2012).

Vučnik, M.; Padrah, Z.; Fortuna, C. & Mohorčič, M. Development of discovery and identification protocol for sensor networks. In: *Proceedings of the 4th Jožef Stefan International Postgraduate School Students Conference.* (Jožef Stefan International Postgraduate School, Ljubljana, Slovenia, 2012).

Padrah, Z.; Fortuna, C. & Mohorčič, M. Connecting Contiki enabled versatile sensor nodes via CC1101 radio. In: *Proceedings of the 3rd Jožef Stefan International Postgraduate School Students Conference.* (Jožef Stefan International Postgraduate School, Ljubljana, Slovenia, 2011).

Fortuna, C.; Ivan, B.; Padrah, Z.; Bradenško, L; Fortuna, B. & Mohorčič, M. Demonstration: wireless access network selection enabled by semantic technologies. In: *Proceedings of the 8th International Semantic Web Conference.* (Springer, Chantilly, VA, USA, 2009).

A.2.2 Unpublished conference contribution

Padrah, Z.; Verbič, A.; Mihelin, M.; Fortuna, C. & Mohorčič, M. Connecting Contiki enabled versatile sensor nodes via CC1101 radio. In: *Newcom++ Energing Topic Workshop on Smart Grids.* (Newcom++, Barcelona, Spain, 2011).

Appendix B: Biography

Zoltan Padrah was born on 16 October 1987 in Sighetu Marmaţiei, Romania. In his school years he has shown interest in mathematics and informatics, and participated in numerous student competitions, up to national level. In 2006 he enrolled in the faculty of Electronics, Telecommunications and Information Technology in Cluj-Napoca, Romania. Four years later he received B.Sc. degree in Telecommunications Technologies and Systems.

In 2010 he enrolled in the Information and Communication Technologies programme at the Jožef Stefan International Postgraduate School, Ljubljana, Slovenia. The domain of his M.Sc. studies is telecommunication systems.

He works on software for wireless sensor nodes, including dedicated operating systems and applications without any operating system. His research interests include wireless sensor networks and cognitive radio. In particular, he has been contributing to the establishment of a spectrum sensing testbed for cognitive radio, which uses low-cost low-complexity hardware. His work is covered within the projects of the Department of Communication Systems and the activities of SensorLab, where he is an external collaborator, and member, respectively.