# Low-cost testbed development and its applications in cognitive radio prototyping

Tomaž Šolc, Carolina Fortuna, Mihael Mohorčič

To be published in *Cognitive Radio and Networking for Heterogeneous Wireless Networks*. Copyright (c) 2014 Springer International Publishing Switzerland. Permission granted to use this manuscript for non-commercial and educational purposes.

**Abstract** Having a huge potential to improve the way radio spectrum is being used, the techniques that are used for the research in cognitive radio are maturing and therefore move from being evaluated in a simulation environment to more realistic environments such as dedicated testbeds. In this chapter we describe our experiences with the design, deployment and experimental use of the LOG-a-TEC embedded, outdoor cognitive radio testbed, based on the VESNA sensor node platform. We describe the choice of experimental low-cost reconfigurable radio frontends for LOG-a-TEC and discuss the potential capabilities of custom designs. The core part of this chapter gives practical experiences with designing the embedded testbed infrastructure, covering topology design and performance evaluation of the management network as well as our considerations in the choice of network protocols employed in the LOG-a-TEC testbed. Finally, we provide two use cases where the LOG-a-TEC testbed has been used for performing experiments with cognitive radio, one relevant to the investigation of coexistence of primary and secondary users in TV white spaces and the other addressing power allocation and interference control in the case of shared spectrum.

Tomaž Šolc,
Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia, e-mail: tomaz.solc@ijs.si

Carolina Fortuna,
Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia, e-mail: carolina.fortuna@ijs.si

Mihael Mohorčič,
Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia, e-mail: miha.mohorcic@ijs.si

## 1 Introduction

As a research area in wireless communications that is fundamentally important for more efficient use of radio spectrum, cognitive radio already surpassed the early phase of theoretical only investigations in controlled environments and computer simulations, and has already entered the phase of experimental research and pilot trials in increasingly realistic operating environments. This has been recognised by a number of institutions and projects worldwide, resulting in numerous testbeds, on one hand trying to be sufficiently representative for actual targeting scenarios, and on the other hand striving to be easy for use and to be able to accommodate as diverse algorithms and approaches as possible. As a result, they are inclined towards the use of high-performance general purpose processors and software defined radio (SDR) platforms. Conventional testbeds that are built around this architecture offer a fast way for researchers to implement and test various protocols. However cognitive radio technologies are also interesting for low-powered and low-bandwidth applications, like wireless sensor nodes and body area networks. Such applications often do not have the possibility of including a software defined radio platform in the final product due to power and cost constraints. While most of the basic research can still be done on conventional testbeds, aspects like power, storage and processing limitations on such systems can only be reliably tested on embedded hardware that more closely resembles realistic platforms such applications would run on. Embedded testbeds can also be more easily deployed in configurations where a conventional approach would be difficult. One example of that are outdoor environments, where operation of delicate laboratory equipment can be problematic. Another example are locations that lack existing infrastructure such as Ethernet wiring or constant power supply. The relatively low cost of embedded devices can also be an important benefit in testbeds consisting of a large number of nodes.

A node in a conventional testbed is typically built around a computer running a GNU/Linux-based operating system with a software defined radio front-end and a TCP/IP based management network. A typical setup consists of nodes with a gigabyte of RAM, x86 CPU in the gigahertz clock range and connection to a gigabit Ethernet network. On the other hand, testbeds built up by embedded devices such as the Tmote Sky [1] or VESNA [2] wireless sensor nodes, are powered by microcontrollers with CPU cores like AVR or ARM with clock frequencies in the 10 to 100 MHz range and RAM sizes from 1 to 100 kilobytes. Typically communication between the nodes involves low-power narrow-band radios with bulk transfer rates in the range from 100 bytes per second to 100 kilobytes per second. Nodes in such a setup either run specialized, small-footprint operating systems like Contiki [3] and TinyOS [4] or even just bare bone firmware without a proper operating system. Network protocols encountered in such testbeds are for example IEEE 802.15.4, ZigBee and Bluetooth. Such systems offer very little built-in management functionality that is usually expected from conventional, networked multi-user computers. Therefore the design and use of embedded testbeds pose several additional challenges over those found in conventional testbeds.

In this chapter we describe our experiences with the design, deployment and experimental use of the LOG-a-TEC embedded, outdoor cognitive radio testbed, based on the VESNA [2] sensor node platform, which is integrated with the CREW federation [5] of cognitive radio testbeds. Where possible, we also mention other approaches to design challenges that were taken in other similar testbeds. We describe the choice of experimental radio frontends for LOG-a-TEC and discuss the possibility of custom designs. We also give practical instructions for designing the testbed infrastructure, like the management network and our considerations in the choice of network protocols employed in the LOG-a-TEC testbed. Finally, we provide two use cases where the LOG-a-TEC testbed has been used for performing experiments with cognitive radio.

The chapter is organized as follows. Section 2 discusses radio front-end for embedded testbeds and gives details for the ones used in LOG-a-TEC. Section 3 discusses the infrastructure of the testbed including deployment considerations, network design constraints, planning of the management network and the mechanisms that enable access, control and reconfiguration. Section 4 presents a signal generation use case in which the SNE-ISMTV-TI868 transceiver is used to emulate wireless microphone profiles. Section 5 presents an interference mitigation use case using a simple power allocation experiment on LOG-a-TEC. Finally, Section 6 summarizes the chapter.

## 2 Radio front-ends for embedded testbeds

Software defined radio has become an indispensable tool in communications research. It has allowed for very short development cycles for various protocols from the physical layer up. It has also significantly lowered the entry barrier to prototyping. With a software defined radio architecture a researcher only needs a personal computer with a relatively inexpensive radio frontend, while previously a fully equipped electronics laboratory was needed. The developer also does not need to know details of radio frequency electronic circuit design. In fact, physical radio front-ends can often be interchanged in a modern SDR design without much work thanks to layers of software that provide high-level abstractions over the hardware details. With the advent of frameworks like GNU Radio [6] and IRIS [7] one can develop complete RF systems in high level languages such as Python and XML or using user-friendly graphical tools like the GNU Radio Companion.

However this flexibility of SDR architectures and the convenience it offers to the developer comes with a cost in terms of processing performance, RAM and power requirements. While the throughput of embedded general-purpose processors is continuously improving, low-powered embedded devices are still a long way from supporting full-fledged SDR frameworks. One of the reasons is that battery capacity has not been scaling together with processor performance. It is likely therefore that battery powered and low cost devices will not be able to support the SDR development model for the foreseeable future.

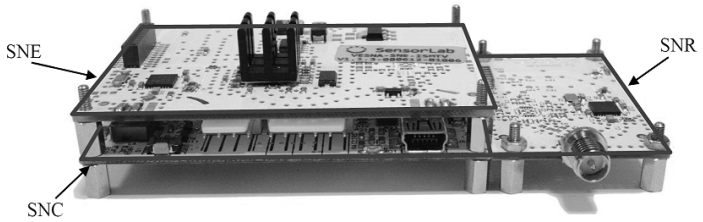


**Fig. 1** Modular design of the VESNA wireless sensor network platform consisting of three types of modules, core (SNC), radio (SNR) and expansion (SNE).

On the other hand, the basic requirements for a cognitive radio can also be satisfied with architectures that are not fully software defined. There are commercially available low-power reconfigurable integrated transceivers that target developers of proprietary radio protocols for embedded devices. These are capable of covering a large part of radio requirements for simple cognitive terminal and dynamic spectrum access implementations: they are reasonably accurate at sensing the spectrum usage using energy detection, they have frequency agile local oscillators that permit fast frequency hopping and interference avoidance and allow software to reconfigure parameters of their transmissions like modulation scheme and channel bandwidth.

This chapter is focusing on cognitive radio testbeds based on low-power embedded devices and as a concrete representative of such devices we mostly refer to the VESNA sensor node platform [2].

## *2.1 VESNA with the SNE-ISMTV expansion*

VESNA [2] is a modular wireless sensor node platform generally consisting of three printed circuit boards: Sensor Node Core (SNC), Sensor Node Radio (SNR) and Sensor Node Expansion (SNE) as depicted in Figure 1.

The SNC board contains a 32-bit ARM Cortex-M3 microcontroller running at up to 72 MHz CPU clock. It has 96 kB of RAM and 1 MB of flash program memory. Additional non-volatile memory is provided by a micro SD card. The microcontroller integrates 3 A/D converters that can be used to sample analog signals with up to 1 Msample/s. Two of these converters can also be combined to sample a single signal with 2 Msamples/s. Power is provided by a versatile subsystem that is capable of powering the sensor node from diverse sources, including photovoltaic cells and batteries.

VESNA supports a number of different wireless interfaces on the SNR board. Some SNR boards contain bare integrated transceivers while other carry radio modules that include both a transceiver and an additional microcontroller for network stack offloading. In the most common wireless sensor network deployment use case, VESNA nodes use a proprietary Atmel ZigBit low-powered radio module to connect to a wireless mesh network. In such use case, the SNE board is used for application

specific hardware, usually to connect sensors to the node that cannot be accommodated by the interfaces on the core board itself.

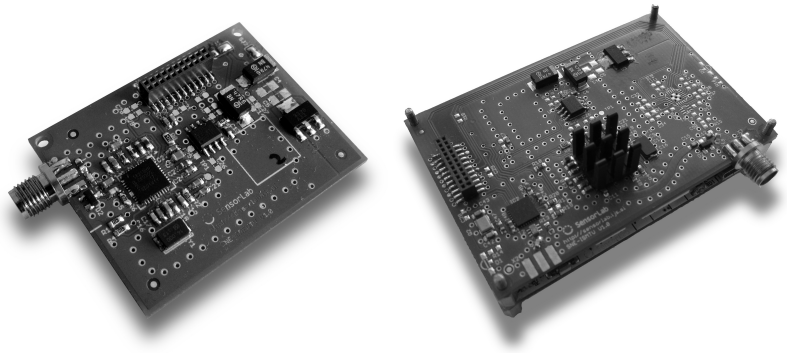

**Fig. 2** UHF receiver expansion boards used for cognitive radio experimentation: Early prototype (left) and final VESNA SNE-ISMTV-UHF circuit board (right).

For experiments with cognitive radio and deployment in cognitive radio embedded testbeds such as LOG-a-TEC, this setup has been left basically unmodified. However, instead of using the radio hardware on the SNR boards for experimental use, we have developed a series of SNE expansion boards that contain a separate set of transceivers. This had an important benefit of leaving the original mesh network interface on VESNA unmodified. That interface can then be used for a reliable wireless testbed management network, independent of the experimental work.

The series of SNE boards for cognitive radio experimentation have been named SNE-ISMTV [8], mnemonically identifying the supported frequency bands. Different versions of the SNE-ISMTV share a single printed circuit board layout. They only differ in electronic component placement. The SNE-ISMTV-TI2400 contains a Texas Instruments CC2500 transceiver for the 2.4 GHz band, SNE-ISMTV-TI868 a Texas Instruments CC1101 sub 1-GHz transceiver and SNE-ISMTV-UHF a custom designed energy detection receiver for the UHF band as shown in Figure 2.

## *2.2 Reconfigurable integrated transceivers*

A list of transceivers for sub-1 GHz frequencies that are commonly found on small embedded devices, such as wireless sensor nodes, is given in Table 1. Same vendors often offer similar devices that work in the 2.4 GHz international ISM band. While they are mostly controlled via a digital bus like SPI (Serial Peripheral Interface) or $I^2C$ (Inter-Integrated Circuit) from a separate microcontroller on the embedded device, it is also common to find such devices integrated with a general-purpose processing core into a single chip.

| Transceiver | Modems | Bitrate [kb/s] | $f_c$ [MHz] | $BW$ [kHz] | Hop time [$\mu$s] |
|---|---|---|---|---|---|
| TI CC1101 | FSK, GFSK, OOK ASK, MSK | 0.6 - 500 | 779 - 928 | 58 - 812 | 75 |
| Silicon Labs Si4313 | FSK, GFSK, OOK | 0.2 - 128 | 240 - 960 | 2.6 - 620 | 200 |
| Nordic Semi nRF905 | GFSK | 50 | 430 - 928 | 100 | 650 |

**Table 1** List of common reconfigurable sub-1 GHz transceivers.

A typical block diagram of these reconfigurable transceivers is shown in Figure 3. They all have a single chip design that requires very little in terms of external components. Rarely, they are coupled to an external power amplifier for range extension and/or an external low noise amplifier for improved noise figure.

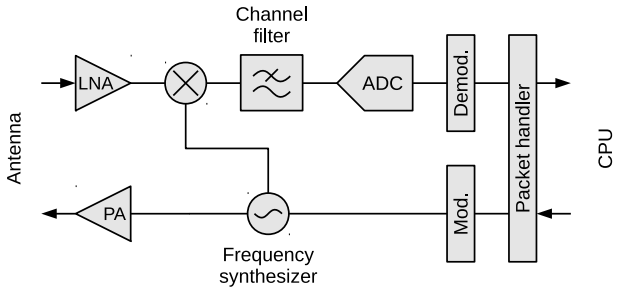


**Fig. 3** Block diagram of a typical reconfigurable transceiver.

A PLL frequency synthesizer is used both as a local oscillator in the reception mode and as a frequency agile sine-wave generator in transmission mode. A programmable digital or an analog filter bank provides channel selectivity that is reconfigurable from software. Several modulation and demodulation blocks can also be software selected. Additionally, packet handling logic can be enabled, which can be used to off-load start symbol detection, whitening, CRC calculation and similar tasks from the main CPU. Since the frequency synthesizer is shared between receive and transmit chains they typically allow only half-duplex operation.

Disabling packet handling hardware allows for continuous reception or transmission by streaming digital data to or from the transceiver. This allows for interpretation of such a stream as a sampled analog signal which opens interesting possibilities, like emulation of analog transmissions. It should be noted though that these transceivers do not typically allow access to the raw signal samples from the ADC. The only data stream available to the CPU is one provided by integrated demodulator blocks. This prevents development of spectrum sensing methods beyond energy detection and signal processing using such hardware.

### 2.2.1 Energy detection

Most transceivers include an integrated logarithmic detector behind the channel filter that can be used as a *received signal strength indicator (RSSI)*. While this capability is meant to be used for the implementation of clear-channel assessment, listen-before-talk and CS-MAC protocols, it can also be used as a simple energy detector for signals within the tuned RF channel. This capability is well known in the research community since it is often used for spectrum sensing experiments or as a low-cost spectrum analyzer. In this case, the transceiver is set to receive mode and swept through a list of channels as presented in the flowchart in Figure 4. On each channel, a RSSI measurement is taken. Such measurements are limited in the resolution bandwidth by the available settings for the channel filter and in the sweep time by the settling of various automatic gain control (AGC) stages in the receiver.

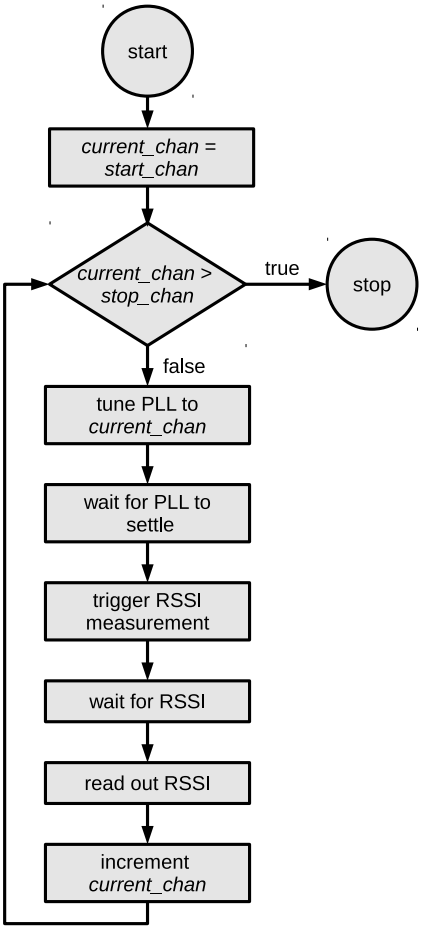


**Fig. 4** Typical flowchart for a procedure performing energy detection spectrum sensing on a part of spectrum between two RF channels, *start_chan* and *stop_chan*, with a reconfigurable transceiver.

Typical measurements take between 5 to 50 ms per channel and have a resolution of 0.5 dB.

When using the RSSI indicator in such a way, care must be taken with respect to the interpretation of the obtained results. Chips often have offsets that require calibration, if absolute values of the incident power are required. Detectors commonly also have a strong temperature dependency that might lead to wrong conclusions (as noted for example in [9]).

It is also important to note that since the logarithmic detector measures total power after the channel filter, its readings are only approximately equal to the signal power when signal power is significantly above the receiver noise floor. In cases where the signal to noise ratio is close or below 1, only reading the RSSI is not an accurate indicator of the signal level and more advanced methods are required, such as using link quality indicator (LQI) as indicator of signal to noise ratio [10].

### 2.2.2 Programming interface

The connection between a low-powered transceiver and the central CPU on the embedded device usually takes the form of an embedded serial control bus, like SPI or $I^2C$. The transceiver exposes a small address space (typically 8-bit) to the controlling CPU over this bus. Most transceivers also provide additional interrupt lines. These can be used to simplify software on the CPU and also provide a more accurate timing of packet reception or transmission than what can be achieved using a serial bus.

The radio configuration typically requires setting around 50 hardware registers accessible from the control bus. For example, the channel filter bandwidth may be set by writing appropriate values to bits 4 through 7 into an 8-bit register at address 0x10 as presented in Figure 5 line 9. Usually the exact register settings are derived from the calibration values that are only known to the transceiver manufacturer. Because of that, the specified performance of the radio can often only be achieved by using values calculated by specialized software for each transceiver configuration such as the one depicted in Figure 6. In such case, a cognitive terminal must store a predefined set of radio configurations that have been precalculated using the manufacturer's software.

Data reception and transmission is similarly performed through FIFO buffers accessible over the control bus, although sometimes a completely separate digital bus is used for streaming data as is the case for example in CC1101 in continuous transmission/reception mode.

There is currently no unified interface that would allow for fast prototyping of solutions using such hardware. Compared to high level scripting provided by the GNU Radio platform, here typically the developer has to have intimate knowledge of the radio hardware. In the future, standardised transceiver APIs might improve this situation. Similarly, there has been recently some development in open source SDR frameworks that allow for delegation of certain signal processing tasks to hardware blocks like the ones found on the reconfigurable transceivers discussed here (for ex-

```
1.  uint8_t initSeq[] = {
2.      CC1100_IOCFG2,   0x0B, // GDO2Output Pin Configuration
3.      CC1100_IOCFG0,   0x0C, // GDO0Output Pin Configuration
4.      CC1100_PKTCTRL0, 0x22, // Packet Automation Control
5.      CC1100_FSCTRL1,  0x08, // Frequency Synthesizer Control
6.      CC1100_FREQ2,    0x58, // Frequency Control Word
7.      CC1100_FREQ1,    0xE3, // Frequency Control Word
8.      CC1100_FREQ0,    0x8E, // Frequency Control Word
9.      CC1100_MDMCFG4,  0x86, // Modem Configuration
10.     CC1100_MDMCFG3,  0x75, // Modem Configuration
11.     CC1100_MDMCFG2,  0x00, // Modem Configuration
12.     CC1100_MDMCFG0,  0xE5, // Modem Configuration
13.     CC1100_DEVIATN,  0x67, // Modem Deviation Setting
14.     CC1100_MCSM0,    0x18, // Main Radio Control State
15.                            // Machine Configuration
16.     CC1100_FOCCFG,   0x16, // Frequency Offset Compensation
17.     CC1100_FSCAL1,   0x00, // Frequency Synthesizer Calibration
18.     CC1100_FSCAL0,   0x11, // Frequency Synthesizer Calibration
19.     CC1100_PATABLE,  0xFE, // TX power 0 dBm
20.     0xFF,     0xFF
21. };
22.
23. static void radioSetup(void)
24. {
25.     vsnCC_disableRadioInterrupt();
26.     vsnCC_init();
27.     vsnCC_radioReset();
28.
29.     vsnCC_strobe(CC1100_SIDLE);
30.     radioWaitState(CC_MARCSTATE_IDLE);
31.
32.     int n;
33.     for(n = 0; initSeq[n] != 0xff; n += 2) {
34.         uint8_t reg = initSeq[n];
35.         uint8_t value = initSeq[n+1];
36.         vsnCC_write(reg, value);
37.     }
38. }
```

**Fig. 5** Example code required to setup a CC-series transceiver for continuous transmission on VESNA platform.

ample specialized engines in IRIS). However, these frameworks still remain much too large for a typical microcontroller system today.

### 2.2.3 SNE-ISMTV-TI868 and SNE-ISMTV-TI2400

As an example of using using low-power embedded devices for energy detection based spectrum sensing we refer in the following to two versions of purposely developed SNE-ISMTV expansion board for VESNA using Texas Instruments CCxxxx transceivers with the main specifications summarised in Table 2 for SNE-ISMTV-TI868 and in Table 3 for SNE-ISMTV-TI2400. SNE-ISMTV-TI868 uses the CC1101 chip and operates in the European 868 MHz short range devices (SRD) unlicensed band and upper channels of the UHF broadcast band. SNE-

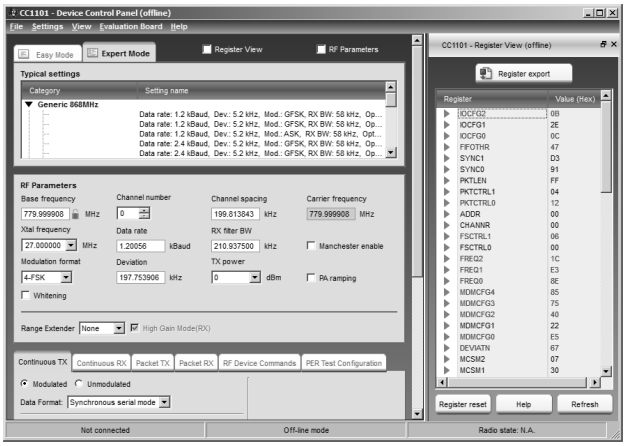


**Fig. 6** Screenshot of Texas Instruments SmartRF studio software.

ISMTV-TI2400, on the other hand, uses the CC2500 chip and operates in the 2.4 GHz unlicensed international industrial, scientific, medical (ISM) band. Except for the difference in the radio frequency, which requires different antenna matching circuit, the two boards are mostly identical in hardware design. Similarly, the software interface of CC1101 and CC2500 is largely identical, which greatly simplified driver software.

| Parameter | Value |
|---|---|
| Central frequency range | 780 to 871 MHz |
| Central frequency resolution | 50 kHz |
| Channel filter bandwidth | 60, 100, 200, 400 or 800 kHz |
| Power detector range | -123 to 4 dBm |
| Power detector resolution | 0.5 dBm |
| Average noise level (1 Hz BW, 868 MHz) | -150 dBm |
| Channel sampling time | 5 ms |
| Transmission power range | -30 to 12 dBm |
| Transmission power resolution | 2 dBm |

**Table 2** SNE-ISMTV-TI868 specifications.

The analogue receive path on CCxxxx series transceivers consists of a low-noise amplifier, quadrature down-converter and a channel filter. Additional filtering, gain control and demodulation are performed in the digital domain, after the signal has been digitized by an analogue-to-digital converter (ADC) at an intermediate frequency. A fractional-N PLL frequency synthesizer is used as a local oscillator. For transmission, the same frequency synthesizer is used to produce a modulated quadrature signal which is amplified by a power amplifier stage. 50 Ω antenna matching is performed by an external LC balun network.

| Parameter | Value |
|---|---|
| Central frequency range | 2400 to 2480 MHz |
| Central frequency resolution | 50 kHz |
| Channel filter bandwidth | 60, 100, 200, 400 or 800 kHz |
| Power detector range | -123 to 4 dBm |
| Power detector resolution | 0.5 dBm |
| Average noise level (1 Hz BW, 2400 MHz) | -159 dBm |
| Channel sampling time | 5 ms |
| Transmission power range | -55 to 1 dBm |
| Transmission power resolution | 2 dBm |

**Table 3** SNE-ISMTV-TI2400 specifications.

The transceiver is connected to the sensor node core CPU with an SPI bus, which exposes radio configuration registers and FIFO buffers. This low-level interface to the transceiver has a form of a finite state machine. Digital radio control logic allows reconfiguration of the frequency synthesizer settings (base frequency, channel spacing) and baseband channel filter bandwidth. A logarithmic-response detector can be used to measure the RSSI. Baseband modulator and demodulator blocks are capable of 2-FSK, 4-FSK, GFSK, MSK and ASK/OOK modulations. Continuous transmission and reception using these modulations is possible with data streamed via a dedicated digital bus on the transceiver's auxiliary pins. Also available is a pseudo-random sequence generator which can be used in continuous transmission mode.

For packet-based transmissions, integrated packet handling hardware implements a proprietary packet encapsulation scheme. This makes it mostly incompatible with standard MAC layers like the one defined by IEEE 802.15.4. The hardware is capable of preamble and synchronization word detection, data integrity check using CRC, address filtering and data whitening.

In cognitive radio experiments so far, SNE-ISMTV-TI868 and SNE-ISMTV-TI2400 have been used as narrow-band energy detection spectrum sensing receivers using the RSSI functionality in the 868 MHz and 2.4 GHz bands to build radio environment maps or to supply channel occupancy information to cognitive terminals.

VESNA nodes with SNE-ISMTV expansion have also been used as controlled interferers and targets for spectrum sensing using other devices. In this case, the continuous transmission mode of the transceiver has been used. The data sent was either a pseudo-random sequence generated by the transceiver or a stream generated on the sensor node's CPU. With the latter option we have been able to approximate analogue transmissions generated by wireless microphones.

To help with the problem of complicated low-level interface exposed by this type of hardware an abstraction layer for software running on the sensor node has been developed. This simplifies programming for two common tasks: energy detection and signal generation. For each version of the SNE-ISMTV board, several hardware profiles have been defined that specify PLL settings (channel base and spacing) and modem configuration.

## 2.3 Getting closer to SDR with custom hardware

Microcontrollers that integrate multiple analog-to-digital and digital-to-analog converters in the Msample/s range make for an inviting target for an embedded version of a software-defined architecture. With a suitable front-end and anti-aliasing filters these can be used to generate or sample signals directly.

However, the processor throughput of such devices is typically far behind the capability of processing a continuous stream from its ADC in real-time. For example, a STM32F1 family microcontroller running at 64 MHz, which is near the top of the range for an ARM Cortex M3 core without DSP extensions, will only process signals on the order of 100 ksample/s even when doing the simplest signal processing tasks.

Processors with large RAM sizes have the possibility of capturing an interval of signal samples and then doing off-line processing. This type of processing, however, has limited usability in cognitive radio applications beyond spectrum sensing.

### 2.3.1 TV Band Spectrum sensing with SNE-ISMTV-UHF

The VESNA expansion board for spectrum sensing in the UHF and VHF bands has been developed specifically for research into heterogeneous spectrum sharing and cognitive radio applications in the TV-band whitespaces. It is based on a cheap and compact multi-standard TV silicon tuner design which allows it to be deployed in large sensor networks for distributed sensing applications.

SNE-ISMTV-UHF contains a VHF and UHF TV band receiver based on the NXP TDA18219HN silicon tuner and was designed for spectrum sensing experimentation in TV white spaces (TVWS). Figure 2 presented an early version of the receiver and the final printed circuit board used in LOG-a-TEC. The TDA18219HN silicon tuner integrates a low-noise amplifier, RF tracking filters, single down-conversion low intermediate-frequency image-rejection mixer, frequency synthesizer and selectable channel filters. It also includes multiple stages of analogue automatic gain control (AGC).

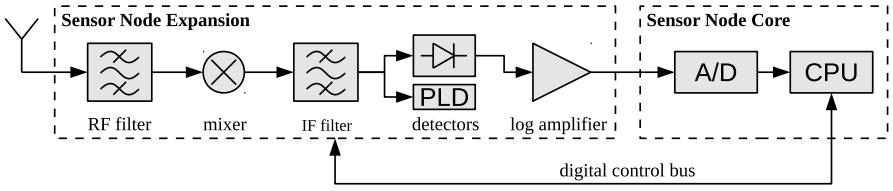


**Fig. 7** Block diagram for signal reception using VESNA with SNE-ISMTV-UHF.

Individual receiver stages can be reconfigured through a state machine with an $I^2C$ bus interface. Digital control logic also controls built-in test tone generator and RF filter calibration.

For energy detection, SNE-ISMTV-UHF offers two detectors with a logarithmic response. A high-range detector is built into TDA18219HN and has a range from -80 to 0 dBm with 1 dBm resolution. The measurement is controlled via a state machine internal to TDA18219HN which is armed and triggered through the digital $I^2C$ bus. The measurement process includes signal averaging.

For measuring signals below -80 dBm, a demodulating logarithmic amplifier Analog Devices AD8307 is connected between the TDA18219HN intermediate frequency output and the 1 Msample/s analogue-to-digital converter on the VESNA SNC (Sensor Node Core) as depicted in Figure 7. This puts sampling of the signal power level in control of the firmware, allowing various averaging and sampling methods. Additionally, gain in the intermediate frequency stage can be adjusted via a variable-gain amplifier controlled by a digital-to-analogue converter (DAC) in the SNC.

To lower the power consumption, both the logarithmic amplifier (via ENB pin) and the tuner (via $I^2C$ sleep-mode control registers) can be powered down.

Two hardware profiles are provided that differ in energy detector resolution bandwidth. A wide-band setting sets the channel filter to 8 MHz, allowing energy detection at the bandwidth of a complete DVB-T channel. A narrow-band setting sets the channel filter to 1.7 MHz and is suitable for detection of wireless microphones and secondary users in the TV white spaces.

While TDA18219HN and the SNE-ISMTV designs allow operation down to 42 MHz, there are currently no hardware profiles available for VHF frequency band.

Since it is based on a DVB-T tuner, this extension board is not capable of signal transmission in contrast to the extension boards SNE-ISMTV-TI868 and SNE-ISMTV-TI2400, as described in the previous section.

## *2.4 Summary*

In summary, reconfigurable low-power transceivers are capable of performing many radio frontend duties in a cognitive radio setup. However the development of solutions is far from being as accessible as with software defined radio architectures. Such devices deployed in a testbed can be used for instance in a supporting role for other, more full featured cognitive terminals. They can perform energy detection spectrum sensing in a distributed fashion. Reconfigurable transceivers are also flexible enough to take an active role in certain dynamic spectrum access schemes.

With SNE-ISMTV-UHF, we are looking at ways of getting closer to software defined radio using a custom designed low-cost receiver for the VHF and UHF bands based on a commercial integrated silicon tuner. In contrast to reconfigurable transceivers, however, devices with tuners can only be used in cognitive radio applications for spectrum sensing, with capabilities depending on the accessibility of the signal at different stages within the receiver.

## 3 Testbed infrastructure

In order to perform real-life experiments with communication protocols, representative testbeds are needed. Such testbeds can typically consist of a few devices up to hundreds of devices. On the devices experimental implementations of protocols are being run in various scenarios and the results are recorded and later analyzed. This way the performance of the protocols can be evaluated and possibly improved. To facilitate the research activity in dynamic spectrum allocation we created a testbed called LOG-a-TEC primarily for experimentation with spectrum sensing, cognitive radio networking and testing of arbitrary protocols on low-cost low-complexity hardware.

In order to use the embedded devices such as VESNA based sensor nodes with expansion boards described in Section 2 in a cognitive radio testbed, the overall testbed infrastructure needs to be carefully planned and set up accordingly, including (i) the management network, wireless or wired, for configuration of nodes and collection of sensing data, (ii) locations for mounting nodes and antennas, (iii) provision of power for nodes in case of external powering, (iv) adequate data storage and processing capabilities, and (v) a suitable system administration user interface.

Because a cognitive radio testbed is used to experiment with the communication between the nodes, the radio interface used by experiments is unreliable and nodes need to be equipped with additional radio interface for the management network to provide remote access to the nodes.

A testbed infrastructure may also provide power to the nodes in the testbed. In comparison to conventional testbeds, where external power is a must, the physical requirements for the infrastructure are much more relaxed in embedded testbeds. Nodes can be battery powered for the duration of the experiment. Also the management network need not to consist of gigabit Ethernet cabling but can be wireless.

While an embedded testbed is focused on cognitive radio applications that might one day be used in mobile, battery powered devices, most such testbeds are not themselves battery powered, especially those that are permanently deployed. A testbed might use low-powered processors and radio to conduct exclusively tests of wireless protocols and does not focus on the power consumption aspect of its operation. Thus it is not uncommon that in an embedded testbed each low-powered node is accompanied by a conventional, networked computer that provides management functionality. This significantly simplifies management of the testbed. Embedded operating systems running on nodes often do not support run-time reprogramming or include system administration functionality that is common on multi-user systems like GNU/Linux.

### *3.1 LOG-a-TEC testbed deployment*

The LOG-a-TEC testbed was deployed in the town of Logatec, Slovenia [11][12]. It consists of two clusters, each cluster being composed of 25 devices. One cluster

is located in the industrial zone and the other in the town center. The location of the two clusters inside Logatec is presented in Figure 8. The pentagonal markers represent the locations the nodes. The cluster of markers situated in the upper side of Figure 8 corresponds to the industrial zone, while the cluster of markers on the lower side of the same figure corresponds to the town centre. The majority of devices in both clusters are installed on public light poles and connected to external power supply.

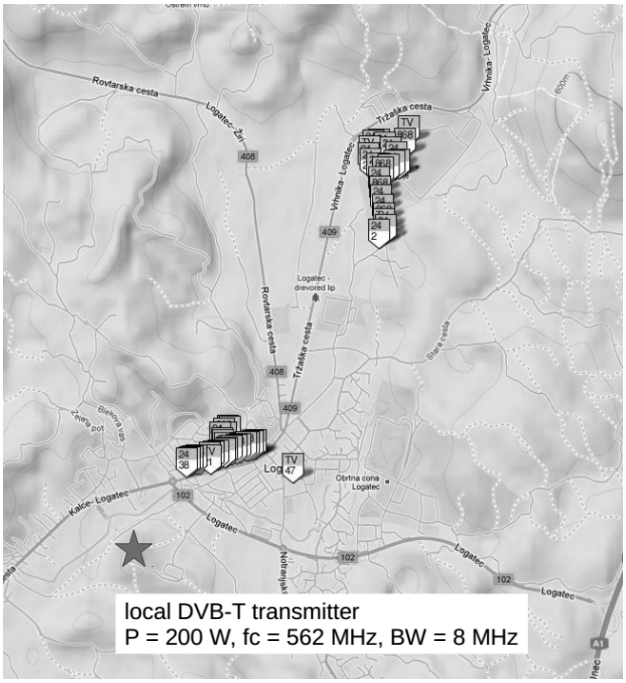


**Fig. 8** Locations of the device clusters inside Logatec.

The deployed devices are based on the VESNA hardware platform [2] described in Section 2. Each VESNA is equipped with multiple radio modules:

- Atmel's ZigBit radio module ATZB-900-B0 operating in the license-free 868 MHz European SRD band is used for the management network.
- One of the three radio modules for spectrum sensing and cognitive radio experimentation that can be hosted on the SNE-ISMTV extension board described in Section 2.1, i.e. CC1101 and CC2500 from Texas Instruments or the custom designed spectrum sensing receiver operating in the UHF TV band.
- Atmel's IEEE 802.15.4 compliant radio AT86RF212 operating in the license-free 868 MHz European SRD band for communication experiments.

Each VESNA is thus equipped with one of the following three combinations of the radios:

- ATZB-900-B0 + CC1101 for 868 MHz sensing + AT86RF212
- ATZB-900-B0 + CC2500 for 2.4 GHz sensing + AT86RF212
- ATZB-900-B0 + TV receiver for UHF sensing + AT86RF212

The logical block diagram of one cluster of the testbed is depicted in Figure 9. The cluster has two parts that are situated at different physical locations. The first part, where the experiments will be performed, is located in Logatec and is illustrated on the left side of Figure 9 and consists of a set of VESNA nodes and the cluster coordinator. This part is connected to the Internet through its Gateway that corresponds to the VESNA coordinator and internally uses a ZigBee network for communication. The second part of the testbed, presented on the right hand side of Figure 9, is the infrastructure serving the experimental part. This part consists of a PC running an HTTP server that is connected to the Internet and communicating with the Gateway. This server provides an HTTP API which supports programmatic access to VESNA's REST interface and also hosts the LOG-a-TEC web portal depicted in Figure 10. This portal renders a map with the location of VESNAs and their status, allows manually issuing GET and POST requests and supports running simulations with the GRASS-RaPlaT [13] tool.

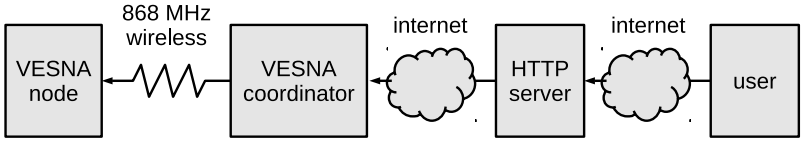


**Fig. 9** The logical block diagram of one cluster in LOG-a-TEC.

The currently supported VESNA software stack is depicted in Figure 11 and can use libopencm3 or STM32FWLib as an abstraction layer towards hardware registers and MCU peripherals. While the first library is released under an open source license (LGPL), the second is proprietary. VESNA drivers then provide a high level abstraction to support storage, various sensors and functionalities of the platform which may run the event driven Contiki operating system if needed. In many applications, including LOG-a-TEC, we use application specific software without an operating system. On VESNA, an application can run (1) on top of the Contiki OS, (2) run without an OS using the drivers or (3) run without an OS using libopencm3. The decision on how to build an application depends on licensing constraints and on the complexity of the application. For the case of LOG-a-TEC, the second option using STM32FWLib, VESNA drivers and a custom application was selected. The application provides (1) a REST interface that supports remote experiments, (2) native code for local experiments and (3) testbed management functionalities such as over-the-air reprogramming and monitoring.

The control of the SNE-ISMTV extension for the purpose of experimentation can be achieved in several ways as illustrated in Figure 11. Currently there are four ways of controlling an experiment that differ in terms of complexity, response time and location. The higher the complexity required to control the hardware to perform

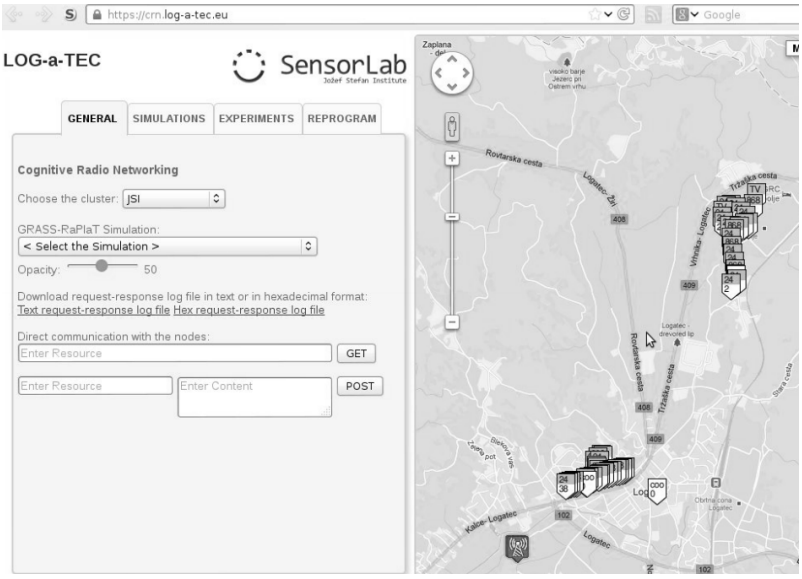


**Fig. 10** The LOG-a-TEC web portal.

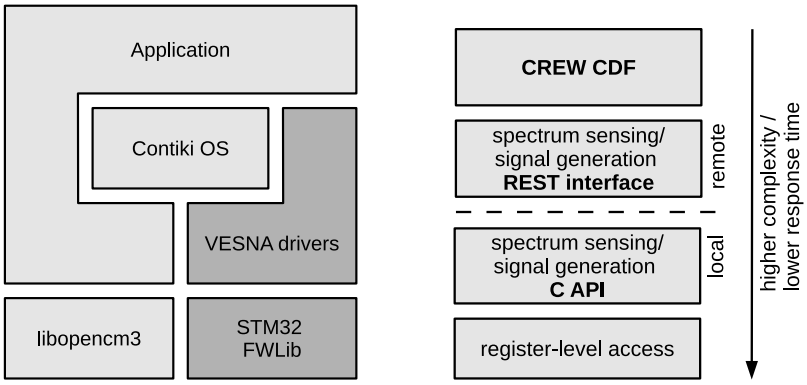


**Fig. 11** VESNA software stack (left) and layers of abstraction over SNE-ISMTV expansion boards (right).

the experiment, the lower the response time. Additionally, there are two options that allow remote control of the experiment and two options that allow local only control requiring physical connection to the hardware.

A low-complexity approach that supports remote experiment control using a device and testbed independent language using the XML syntax is to define an experiment using the CREW common data format (CDF) and then request the LOG-a-TEC infrastructure to run the defined experiment. The resulting measurement will be provided in the form of CSV or Matlab files containing time, frequency and power values resulting from the sensing experiment. A slightly higher complex-

ity approach that also supports remote experiment control is to use the available REST interface. This assumes the experimenter downloads the available Python or Javascript tools [1] and writes the logic of the experiment using these tools. Using these tools, the experimenter is able to specify actions such as (i) sense band B at time T using device D and method M, and (ii) transmit at frequency F and power P. This second approach is the most frequently used.

A higher complexity approach to controlling the experiment is via direct access to the C API from native code. This option removes network latency caused by network round trips, on-board processing and interference from the wireless management network operating on 868 MHz. Finally, the most complex and time consuming way of controlling the experiment is to directly program the hardware (i.e. the CC1101, CC2500 and TDA18219 chips) and exploit the full capabilities of the hardware. This approach is time consuming as it requires testing but results in lower response time in terms of spectrum sensing and transmission.

## *3.2 Network design constraints*

The design of ZigBee based wireless management network, which provides remote operation of the deployed testbed, needs to take into consideration the constraints of the devices and possible deployment locations [14]. One of the constraints of the design of the network is the availability of frequency bands where the ZigBee network can operate. There are two choices for this network, the 2.4 GHz ISM band and the 868 MHz SRD band. The 868 MHz SRD band offers better propagation characteristics and it does not interfere with the crowded 2.4 GHz ISM band, where majority of spectrum sharing experiments are expected to be performed as opposed to TVWS related experiments in UHF band. Thus, the 868 MHz SRD band was chosen as the operating band for the management network in the LOG-a-TEC testbed.

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 (or alternative infrastructure): devices are mounted on light poles belonging to the public lightning system. Possible exceptions are the Gateway devices.
- Power connections to the light poles: the devices in LOG-a-TEC have always available power source. Light poles are powered in groups, so economically the most suitable solution is to equip all light poles of the selected light pole groups.
- Connectivity: because all the devices will be accessed through the Gateway, for maximum performance it is important to have minimum number of hops from any device to the Gateway.

[1] The tools are available for download at `https://github.com/sensorlab`

- 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.

The first two constraints 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 regarding network connectivity it is important to evaluate also the network performance required for the operation of the testbed. The usual management activity for the network generates small amount of traffic while the following activities require the transfer of large amounts of data:

- Transmission of firmware images for devices, in order to run custom applications during experiments: For testing 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 be several hundreds of kilobytes for each different firmware type. Since the firmware distribution mechanism makes use of the broadcast nature of radio communications, multiple nodes can receive a firmware image instantly by a single transmission.
- Collection of spectrum sensing data: During spectrum sensing experiments significant amount of data is collected in the testbed, which 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 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 based management network and on the performance of the sensing radios.

For the management network operating in the 868 MHz SRD band, the Atmel's ATZB-900-B0 ZigBit™modules [15] 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 both clusters of the testbed, the VESNA node with the Internet access is configured as the coordinator of the ZigBit network. All the remaining nodes in the testbed are configured as routers, since this configuration appears to support larger throughput compared to end devices, most likely because of radio duty-cycle limitation of the latter configuration.

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 [16]. 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 planned, enabling security in the ZigBit network is necessary for preventing malicious 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 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.

## *3.3 Performance evaluation of the management network*

Before the actual deployment of the testbed the behavior of the management network was determined experimentally by measuring radio link quality between candidate locations of VESNA nodes. For the experiments, two VESNA nodes were used, both equipped with the ATZB-900-B0 radio modules. One VESNA was acting as an echoing device denoted with (E), while the other VESNA was transmitting requests to the first one and receiving the responses, hence it is denoted with (T/R). This setup is schematically depicted in Figure 12. The VESNA node (E) was responding to the requests sent from the VESNA node (T/R). The VESNA node (T/R) was connected with an RS232 connection to a PC which was saving the measurement logs. The PC and the VESNA node (T/R) connected to it were moving during the measurement campaign between candidate locations, while the VESNA node (E) was fixed to selected location. Both nodes involved in measurements are depicted in Figure 13.

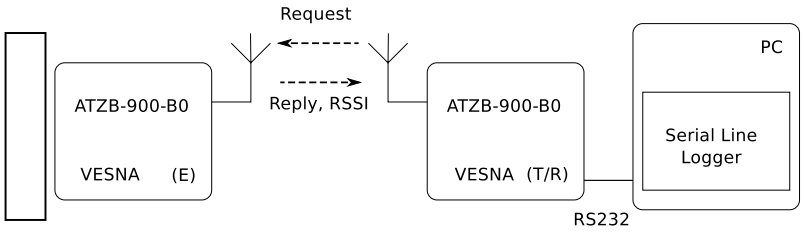


**Fig. 12** Measurement setup.

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

- 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.

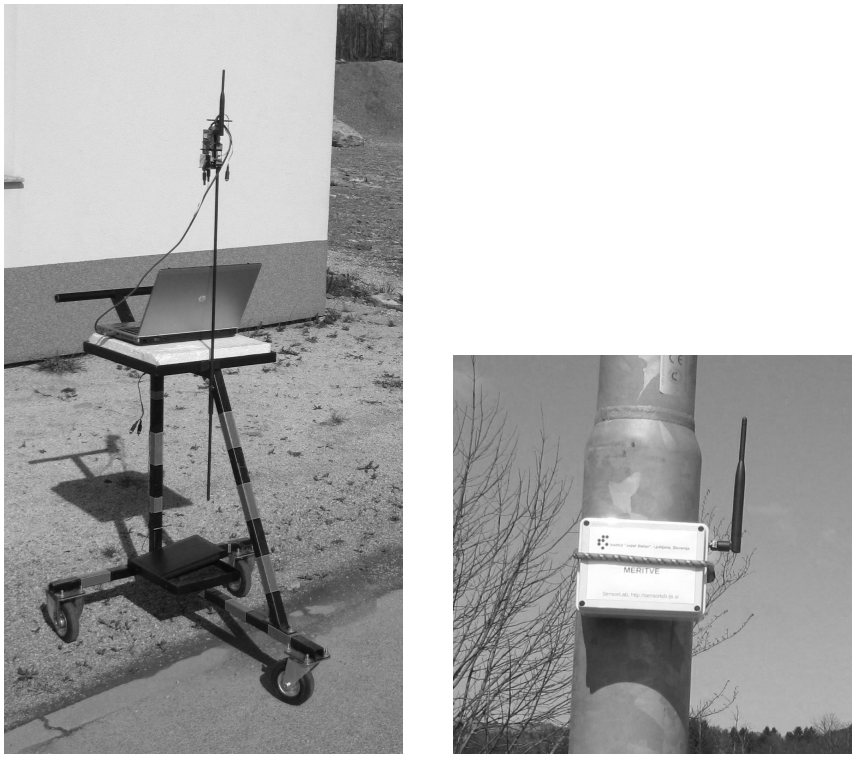


**Fig. 13** VESNA node (T/R) with a PC on a mobile measurement platform (left) and VESNA node (E) at a fixed location (right).

- 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 [15]. 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 [17].

For every 30 requests, the VESNA node (T/R) printed the statistics about packet loss, time needed for measurements, average round trip time and RSSI. A representative part of the output from the VESNA node (T/R) is presented in Figure 14.

This data was complemented by a measurement location record in terms of GPS coordinates obtained by a dedicated GPS receiver, a GPSMAP 60CS device produced by Garmin.

```
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
```

**Fig. 14** Statistics about packet loss provided for each 30 requests.

## *3.4 Measurements in Logatec*

Based on the methodology for performance evaluation of management network, described in Section 3.3, field measurements have been carried out at candidate locations for deployment of VESNA based spectrum sensing nodes in two clusters in Logatec. The obtained measurement results are presented in the following subsections.

### 3.4.1 Measurement results in the industrial zone cluster

The locations of the light poles in the industrial zone are depicted in Figure 15 on a satellite and on a map view of the zone. Figure 15 also shows the numbers assigned to the light pole locations. Location (C) represents the only location with available fixed Internet access, making it the most suitable candidate location for the coordinator.

Two sets of measurements have been carried out in this zone: in one set the link quality between the position of the coordinator and one subset of the light pole positions has been measured, and in the second set the link quality between location number 5 and another subset of the light pole locations has been measured. The two subsets of measurement locations partially overlap, and together they cover all possible device locations. The results from both first sets of measurements are presented in Figure 16.

From the results it can be seen that for locations farther away from the coordinator than location 5, over 10% of the transmitted packets is lost. Hence it has been considered that the nodes that are closer than location 5 are reachable 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 the measurements it has been concluded that the ZigBit modules operating in the 868 MHz SRD band can reach nodes at least at

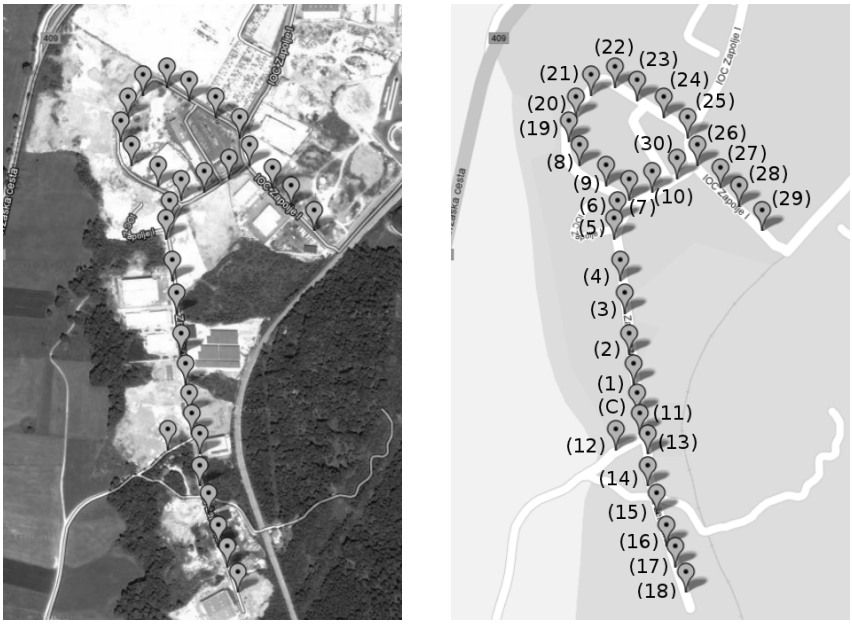


**Fig. 15** Satellite (left) and map view (right) of measurement locations in the industrial zone of Logatec. Location designations are given in parentheses.

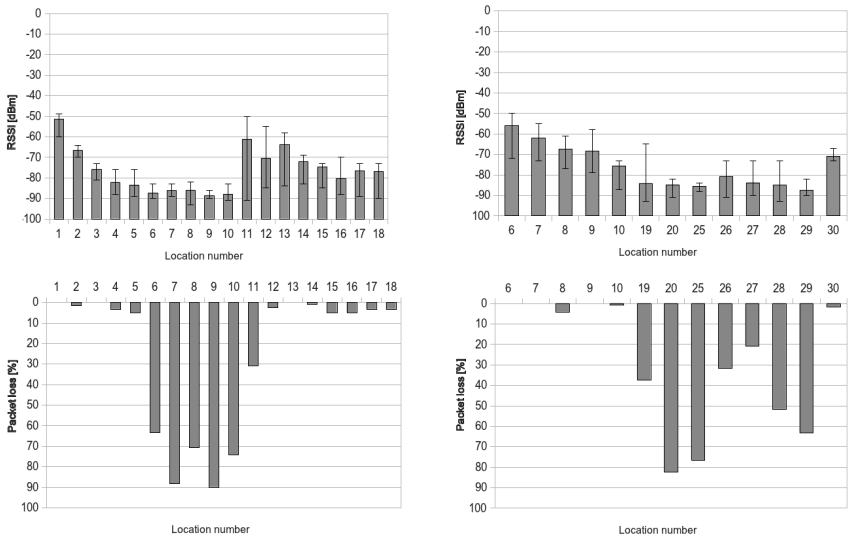


**Fig. 16** RSSI and packet loss in the industrial zone measured between light poles and the coordinator's location (top and bottom left) and measured between light poles and location number 5 (top and bottom right).

5 light poles away. These estimates are pessimistic because the measurement have been carried out at approximately 1.2 metres above the ground level, whereas nodes were to be installed on the public light poles at approximately 4-5 metres above the ground level, where propagation conditions are expected to be more favourable.

### 3.4.2 Measurement results in the City center cluster

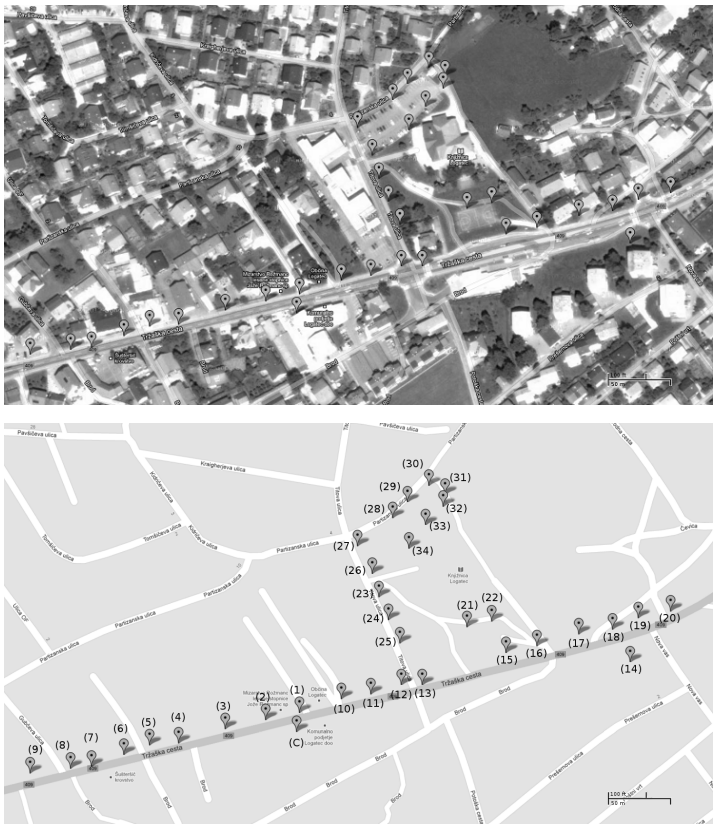


**Fig. 17** Satellite (top) and map view (bottom) of measurement locations in the center of Logatec. Location designations are given in parentheses.

The locations of the light poles in the center of Logatec are presented in Figure 17. The numbers on the map represent the numbers given to the public light poles in the center of Logatec. The location marked with (C) shows the preferred candidate location for the coordinator, providing fixed Internet access. The public light poles in the center are installed more densely than in the industrial zone.

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

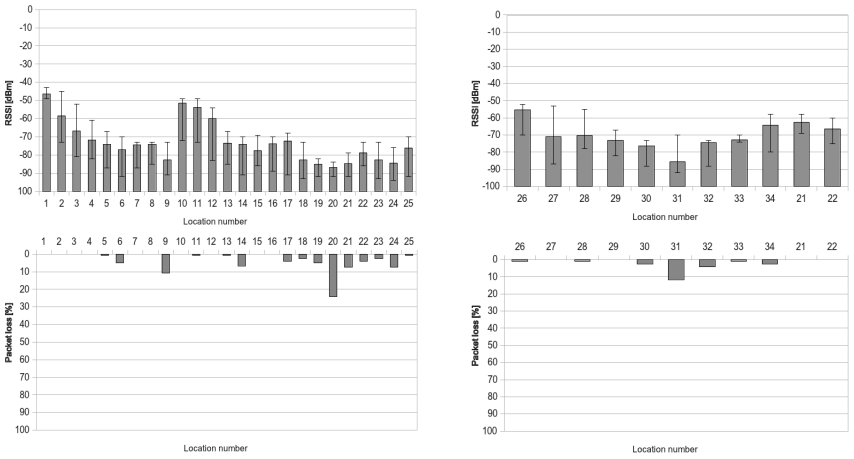


**Fig. 18** RSSI and packet loss in the center of Logatec measured between light poles and the coordinator's location (top and bottom left) and measured between light poles and location number 25 (top and bottom right).

The results of the two measurement sets are shown in Figure 18. From the first set of measurements it can be seen that the coordinator has 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 second set of results shown in Figure 18 confirms 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.

### 3.4.3 Selecting device locations for testbed deployment

ZigBit modules used in the LOG-a-TEC testbed for the management network form a network with a tree topology starting from the coordinator as the root of the tree. Based on the measurement results given in Sections 3.4.1 and 3.4.2 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 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 for the industrial zone cluster the communication range in the 868 MHz SRD frequency band is at least 5 light poles, and even larger for the city center with more dense public light poles. This means that VESNA nodes with SNE-ISMTV-TI868 operating in the 868 MHz SRD band can be installed on every third light pole, without affecting the sensing performance significantly. VESNA nodes with SNE-ISMTV-TI2400 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 is enough to install only a few UHF band TV receivers in the two clusters of the testbed.


**Fig. 19** Device types deployed in Logatec industrial zone (left) and city center (right).

Based on the observations from above, the light pole allocations depicted in Figure 19 have been created for the industrial zone and for the city center. On both figures the striped, white and shaded markers with a dot indicate the locations of the 868 MHz SRD band, 2.4 GHz ISM band and the UHF band sensing nodes, respectively, and the white markers without a dot show locations reserved for future expansion of the testbed. The locations of the two coordinators are not shown explicitly, although their locations can be easily found from Figure 15 and Figure 17 and they coincide with the locations of the UHF band sensing nodes.

As it can be seen from Figure 19, three VESNA nodes with SNE-ISMTV-UHF receivers are planned for the industrial zone, and four for the city 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 SRD 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 were installed in the industrial zone and 15 in the city center, whereas for 868 MHz SRD band sensing 9 and 7 sensors were installed in the industrial zone and in the city center, respectively. The reserved locations are marked on the maps because the public light poles are powered in groups, so the reserved locations will 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.

## *3.5 LOG-a-TEC testbed access, control and reconfiguration*

Remote access to and configuration of nodes in a testbed is supported through the wired or wireless management network. In case of the VESNA-based testbed the management network among nodes is based on ZigBee, as implemented in the ZigBit port to Atmel modules. It forms a wireless multihop network, which means the gateway is able to communicate also with the nodes out of its direct radio range.

For the purpose of communication between sensor network and the remote server (infrastructure side) we developed a new protocol (see Figure 20), which was inspired by the HTTP protocol and is simple enough for fast implementation on VESNA nodes. The protocol defines two types of requests, GET and POST, which are understood by every VESNA node. The GET is used for safe requests which do not change the state of the system and POST for unsafe requests which change the state of the system. The response from a node following a GET request is considered to be in binary format and handled accordingly, although general responses are in text format and only the spectrum sensing data is in binary format. The ending of each response is indicated with the sequence `OK\r\n`.

The protocol includes simple but efficient error handling mechanism. There are two types of errors defined. The first is JUNK-INPUT, which is the more common situation when the resource name is mistyped and the parser on the node does not recognize it. The second type of error is CORRUPTED-DATA, used when cyclic redundancy check (CRC) check did not succeed thus we can conclude that the error happened somewhere on the line between the infrastructure and the gateway. The last situation will occur with very low probability.

The protocol is designed as a client-server protocol. In our case the server side is on sensor nodes and the client side is on the remote server. Before we can access the resources the gateway has to establish a connection with the infrastructure side. This is done by establishing a secure SSL encrypted socket with the remote server. The gateway has an Ethernet module embedded on the expansion board which is used to connect the gateway to the internet. The Ethernet module is configured to get the IP address from DHCP server and then automatically tries to set up an encrypted SSL socket with one of the SSL servers listening on a specific port located on the remote server. Once the connection has been established one could access any resource (sensor, radio module, etc.) or procedure on any of the nodes. Procedures

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

     resource:   abstract resource identifier
     examples:   - firmare/version
                 - sensors/temperature
     arg1:       parameter 1 name
     val1:       value of parameter 1
     ...
     argN:       parameter N name
     valN:       value of parameter N

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

     resource:   abstract resource, e.g. firmware
     arguments:  arguments given to the handler of POST
     example:    2.34/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=;
                 value represented as an unsigned decimal
                 number

Responses from the coordinator have the general form:

   <response to a specific request>\r\n
   OK\r\n
```

**Fig. 20** Resource access protocol.

pre-prepared on the nodes include remote reprogramming, start spectrum sensing, collect spectrum data, configure nodes as transmitters, configure frequency band, etc.

## *3.6 Summary*

This section presents the testbed and management network planning performed for the LOG-a-TEC testbed, taking into account high-level goals of the testbed and the constraints and performance requirements of the management network. The methodology for performance evaluation of the network used in the planning of the testbed is presented, and results from actual field measurements are provided and analysed, justifying the selection of device locations and their distribution in the LOG-a-TEC testbed as per sensing functionality. Section concludes with the description of the protocol used for remote access, control and configuration of the

LOG-a-TEC testbed making use of the underlying management network based on Atmel's ZigBit network.

## 4 Signal generation use case

The LOG-a-TEC testbed presented in previous sections can be used for different experiments in communications, facilitating the transition of technology from fully controlled computer simulation environments to actual prototyping. As explained in Section 2, the main role of VESNA nodes equipped with the SNE-ISMTV expansion boards is spectrum sensing, and as deployed in the LOG-a-TEC testbed they can perform distributed spectrum sensing. However, investigation of primary-secondary scenarios requires also controlled emulation of primary users, especially wireless microphones. To remove the need to manually introduce wireless microphones into the testbed during the experimentation, VESNA nodes can also support remotely controlled emulation of wireless microphone transmissions. In particular, this has been enabled on VESNA nodes equipped with the SNE-ISMTV-TI868 transceiver, that is capable of transmitting a narrow-band signal at frequencies between 780 MHz and 862 MHz. This frequency span includes the upper UHF channels that are used by licensed wireless microphone users. In LOG-a-TEC, one or more of these nodes can be remotely triggered to act as wireless microphones during the experiment or demonstration.

To create as realistic environment as possible in the testbed it was decided to use the IEEE wireless microphone emulation profiles [18]. This method approximates a radio signal transmitted by a legacy analogue studio wireless microphone (PMSE) using a frequency modulated continuous sine wave $s(t)$.

$$s(t) = A\cos\left(2\pi f_c t + \frac{f_{dev}}{f_m}\cos\left(2\pi f_m t\right)\right) \tag{1}$$

In Equation 1, $A$ is the carrier amplitude, $f_c$ is the carrier frequency (between 780 and 862 MHz), $f_{dev}$ is the frequency deviation and $f_m$ is the frequency of the continuous sine wave base band signal. Three signal profiles are defined in literature as summarised in Table 4, i.e. silent, soft speaker and loud speaker. Each profile corresponds to one typical operating condition of the wireless microphone and defines the values for $f_{dev}$ and $f_m$.

| Operating mode | $f_m$ [kHz] | $f_{dev}$ [kHz] |
|---|---|---|
| Silent | 32.0 | 5.0 |
| Soft | 3.9 | 15.0 |
| Loud | 13.4 | 32.6 |

**Table 4** FM parameters for wireless microphone simulation profiles.

The SNE-ISMTV-868 transceiver is not directly capable of transmitting a modulated analogue signal as required by this method. However it contains a digital frequency-shift keying modulator block with 4 symbols (4FSK) that can be used to transmit a continuous signal. In FSK each symbol is represented by a sine with a frequency that is shifted from the carrier frequency by a discrete amount, determined by the symbol value. When fed with an appropriate stream of symbols at a high enough rate, such a modulator can be used to approximate an analogue waveform, as depicted in Figure 21.

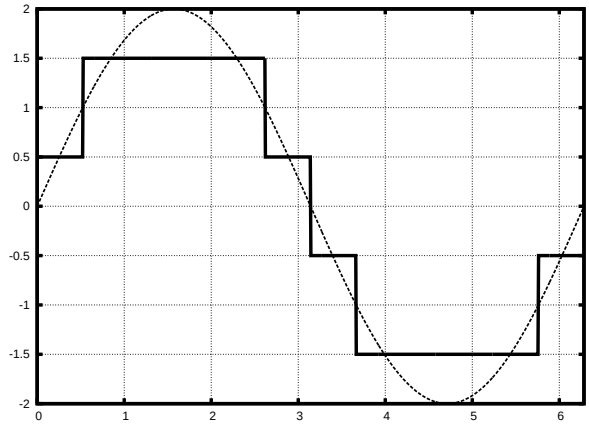


**Fig. 21** Approximating a sine wave with frequency-shift keying transmission. Dotted line depicts an ideal analogue signal while the bold line shows an approximation achievable with a 4FSK modulator.

Using the 4FSK modulator an approximation was hence implemented of the wireless microphone emulation profiles. The block diagram of the implementation is depicted in Figure 22. The transceiver was configured for 200 ksymbols/s, which was the maximum symbol rate that was achievable on the VESNA core CPU running at 64 MHz. This is was equivalent to a transmission of a sampled analogue signal with 2-bit quantization and $f_s = 200$ kHz sampling rate. The continuous symbol stream was provided by the VESNA core CPU to the transceiver through a synchronous serial bus.

To generate the appropriate symbol stream for the modulator hardware block a direct digital synthesis (DDS) algorithm was implemented in software on the CPU. A DDS consists of a phase accumulator, a tuning word $TW$ register and a table of length $L$ that maps between phase value in the desired signal amplitude. For each signal amplitude value emitted by the DDS algorithm, the value of the tuning word is added to the phase accumulator. Hence the frequency of the baseband signal $f_m$ can be adjusted by software simply by setting an appropriate tuning word value, according to Equation 2.

$$TW = L\frac{f_m}{f_s} \tag{2}$$

In our case an output coder was also required to map the signal amplitude to the appropriate 4FSK symbol. Best mappings between signal amplitude and 4FSK symbol were determined experimentally, as they were not provided by the manufacturer of the transceiver.

To set the frequency deviation $f_{dev}$ according to Table 4, the hardware FSK modulator deviation setting was adjusted by writing to the appropriate transceiver hardware register. By adjusting these two settings (FSK deviation and tuning word) we were able to closely match the FM parameters required for all three wireless microphone simulation profiles. An example of the spectrum for the VESNA-emulated wireless microphone signal corresponding to a loud speaker compared to the same signal produced by a USRP device is depicted in Figure 23[2].

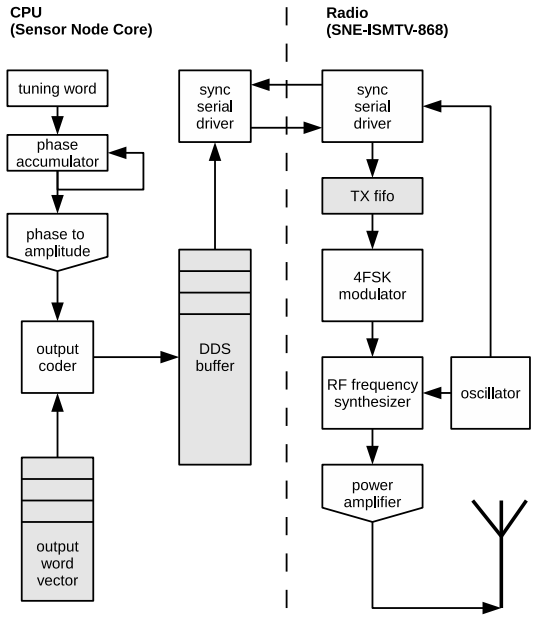


**Fig. 22** Block diagram of Direct Signal Synthesis on VESNA with SNE-ISMTV expansion.

The three supported wireless microphone emulation profiles are available in all VESNA nodes with the SNE-ISMTV-TI868 transceiver in the LOG-a-TEC testbed and were seamlessly integrated with the signal generation API, which is exposed by sensor nodes. To cover the entire UHF frequency band two configurations were required for each emulation profile due to the limitation on the number of channels per

[2] The program source code for the wireless microphone emulation is available at `https://github.com/avian2/vesna-audio-synthesis`

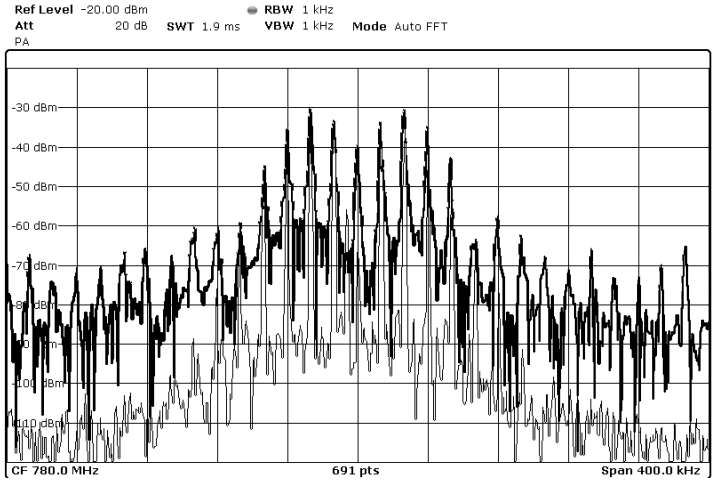


**Fig. 23** Spectrum of a loud speaker wireless microphone simulation signal produced by SNE-ISMTV (upper, thick trace) compared to the same signal produced by a USRP device (lower, thin trace).

hardware transceiver configuration (256). Hence when queried for signal generation profiles (see Figure 24), these nodes report 6 wireless microphone signal generation configurations.

## *4.1 Summary*

SNE-ISMTV-UHF is a low cost custom designed receiver only and therefore unable to transmit in VHF and UHF bands. In order to support the emulation of wireless microphone for LOG-a-TEC, the SNE-ISMTV-TI868 extension board was used to enable generating wireless microphone profiles that are compliant with the IEEE requirements. This functionality can be used in a coexistence study of incumbent and cognitive radio systems in the TV white spaces, as for example in the CREW-TV project concerned with the investigation of the benefits of combining a TVWS database with a distributed sensing network.

```
GET nodes?8/generator/deviceConfigList
dev #0, CC1100, 9 configs:
cfg #0: CC1100, FM noise, 200 kHz deviation:
    base: 779999908 Hz, spacing: 199814 Hz, bw: 197754 Hz, channels: 256,
    min power: -30 dBm, max power: 12 dBm, time: 5 ms
cfg #1: CC1100, FM noise, 200 kHz deviation:
    base: 829999924 Hz, spacing: 199814 Hz, bw: 197754 Hz, channels: 160,
    min power: -30 dBm, max power: 12 dBm, time: 5 ms
cfg #2: CC1100, wireless mic, silent:
    base: 779999908 Hz, spacing: 199814 Hz, bw: 197754 Hz, channels: 256,
    min power: -30 dBm, max power: 12 dBm, time: 5 ms
cfg #3: CC1100, wireless mic, silent:
    base: 829999924 Hz, spacing: 199814 Hz, bw: 197754 Hz, channels: 160,
    min power: -30 dBm, max power: 12 dBm, time: 5 ms
cfg #4: CC1100, wireless mic, soft speaker:
    base: 779999908 Hz, spacing: 199814 Hz, bw: 197754 Hz, channels: 256,
    min power: -30 dBm, max power: 12 dBm, time: 5 ms
cfg #5: CC1100, wireless mic, soft speaker:
    base: 829999924 Hz, spacing: 199814 Hz, bw: 197754 Hz, channels: 160,
    min power: -30 dBm, max power: 12 dBm, time: 5 ms
cfg #6: CC1100, wireless mic, loud speaker:
    base: 779999908 Hz, spacing: 199814 Hz, bw: 197754 Hz, channels: 256,
    min power: -30 dBm, max power: 12 dBm, time: 5 ms
cfg #7: CC1100, wireless mic, loud speaker:
    base: 829999924 Hz, spacing: 199814 Hz, bw: 197754 Hz, channels: 160,
    min power: -30 dBm, max power: 12 dBm, time: 5 ms
cfg #8: CC1100, 868 MHz SRD band, FM noise, 50 kHz deviation:
    base: 863999695 Hz, spacing: 49953 Hz, bw: 49438 Hz, channels: 140,
    min power: -30 dBm, max power: 12 dBm, time: 5 ms
```

**Fig. 24** The profiles currently supported by the VESNA SNE-ISMTV-TI868 in LOG-a-TEC.

## 5 An interference mitigation use case

The use case presented in Section 4 demonstrates how the LOG-a-TEC testbed, or even individual VESNA nodes, can be used to support the investigation of coexistence of primary and secondary users in TVWS, while the use case in this section demonstrates the use of the testbed for investigation of approaches to coexistence of secondary devices in shared radio spectrum. In such use case the experimenter is faced with real-world problems and challenges with the implementation of the investigated approach or algorithm that have to be appropriately addressed. In the following we show an example of a game theoretic approach to cognitive radio with power allocation and interference mitigation game with more detailed insights available in [19].

Game theoretic approaches to the problem of resource management in wireless communications have been extensively studied over the last decade as they provide a more flexible and dynamic alternative to co-existence and resource sharing. One of the simplest forms of co-existence is coordinating the interference among several transmitters by limiting the transmit power. For instance, in [20], the authors propose a theoretical framework for a game-theoretic power control scheme for wireless and ad-hoc networks. In [21], the authors reformulate the problem by also considering energy efficiency alongside with interference and propose the ProActive Power Up-

date (PAPU) algorithm. Simulation results showed that PAPU can utilize energy more efficiently by sacrificing a small amount of network utility compared to the Asynchronous Distributed Pricing (ADP) protocol proposed in [20]. An example use case enabled by the LOG-a-TEC testbed is to evaluate the PAPU algorithm in a realistic outdoor scenario.

## *5.1 Summary of the PAPU algorithm*

The power allocation game proposed in [21] is formulated as follows. Given a wireless network of $N$ transmit-receive pairs ($Tx_i$-$Rx_i$), where a "pair" is referred to as a "player", the objective is to find stable points of power allocation for each player such that the players' global utility is maximum while the cumulated power levels are kept to a minimum.

More formally, given a set of $N$ players, $N = \{1,2,...,N\}$, and their corresponding power allocation profile $P = \{p_1, p_2, ..., p_N\}$, the utility function of each player is given by:

$$u_i = log(1 + \frac{h_{ii}p_i}{n_0 + \frac{1}{B}\sum_{j \neq i} h_{ji}p_j}) \tag{3}$$

where $p_i$, $p_j$ are the transmit powers of players $i$ and $j$, $h_{ii}$ is the direct gain, $h_{ji}$ is the channel gain between transmitter $j$ and receiver $i$, $B$ is the total channel bandwidth and $n_0$ is the noise power. For simplicity and in accordance to [21], we consider B=1. Now, the objective is to maximize the global utility function (4a), while minimizing the globally allocated power (4b) where $p_i \in [0, P_i^{max}]$.

$$max \sum_i u_i \tag{4a}$$

$$min \sum_i p_i \tag{4b}$$

In the evaluations performed in a simulation environment [21], the values for the direct gain, the cross gains, the transmitted power and noise were chosen conveniently to provide the proof of concept. Communication and reconfiguration delay were not taken into account.

The validation of theoretical models in real-world set-ups poses several constraints that are most often testbed specific. We employed the following methodology for investigating the feasibility of experimenting with interference mitigation based on the power allocation game in LOG-a-TEC:

- Identification of the experimental set-up and the constraints.
- Adaptation of the theoretical framework for the use in a testbed rather than in a simulation scenario.
- Empirical determination of the values of parameters such as channel gain.
- Implementation and experimental evaluation.

## *5.2 Experimental set-up*

After identifying and understanding the theoretical framework we desire to validate, the next step is to understand the constraints imposed by the testbed we plan to use. These constraints will require an adaptation of the framework so that it can be validated in a realistic set-up. For the case of LOG-a-TEC, the following constraints can be identified:

- **Topological** - The topology of the testbed is determined by the alignment of the public light poles on which the sensor nodes are mounted. The theoretical framework behind PAPU assumes that the cross gains are significantly smaller than the direct gains. If this constraint is not satisfied, the game might not converge. This constraint and the topology of the testbed narrows down the choice for the location of the players.
- **Transmission capability** - The testbed nodes are able to transmit on one channel at a time, therefore limiting the type of games that can be supported to single channel ones.
- **Power levels** - The nodes' CC2500 transceivers support discrete power levels. Unlike in the theoretical framework, where continuous power levels are considered, the empirical game has to be adjusted to one of the associated power levels specified by the radio chip. This clearly affects players' strategies as well as the resulting stable points of power allocation for players.
- **Sensing** - The nodes in the testbed use energy detection for spectrum sensing. This simple method cannot distinguish between different types of detected signals. As a result, the accuracy of the measurements is lower, therefore transmit power level adaptations of the players may be misguided by errors.
- **Delay** - The delay for setting up a transmission or a sensing vary depending on the nodes, since the management network through which this setup is conducted is wireless. Typical values for this setting are from 1 to 3 seconds. The delay affects the speed of the game, thus the time required to converge.
- **Synchronization** - The nodes in the testbed do not use a clock synchronization protocol, therefore the lack of synchronization has to be taken into account when designing and implementing the game.
- **Calibration** - The low cost nodes are not calibrated, therefore a setting of 0 dBm transmission power might result in a slight shift of the level of the actual transmitted power. This affects the final strategies of the players.

Considering the theoretical framework behind the PAPU algorithm and the constraints imposed by the LOG-a-TEC testbed we defined the interference-aware power control game between two players operating at 2.4 GHz, in the industrial zone cluster. The discrete transmit power levels for the VESNA nodes in LOG-a-TEC can be set to [0, -2, -4, -6, -8, -10, -12, -14, -16, -18, -20, -22, -24, -26, -28, -30, -55] dBm. The two players are transmit-receive pairs coexisting in the same area, as depicted in Figure 25. Player 1 is formed by the Tx-Rx pair Node25-Node2 whereas player 2 is formed by the Tx-Rx pair Node16-Node17.

The distances between the nodes are: $d(TX_1, RX_1) \approx 50$m, $d(TX_2, RX_2) \approx 65$m, $d(TX_2, RX_1) \approx 230$m, $d(TX_1, RX_2) \approx 150$m.

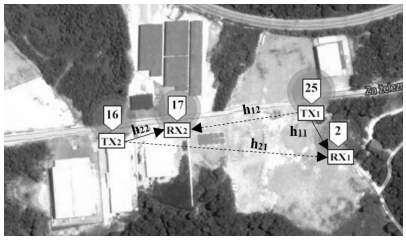


**Fig. 25** Inter-network interference between wireless systems where $h_{ij}$ is the channel gain between transmitter i and receiver j. Pentagonal markers point to sensor node locations. Numbers in markers are location identifiers.

## *5.3 The adaptation of the theoretical framework*

The game adopted in this use case is based on players (users) adapting their transmit power level when detecting a change in other players' power level. The new transmit power level, to which a node adapts, is called the best response. The best response of any of the players involved in the game is given by [21]:

$$b_i(p_{-i}) = \frac{1}{c_i} - \frac{\sum_{j \neq i} h_{ji} p_j + n_0}{h_{ii}} = \frac{1}{c_i} - \frac{I + n_0}{h_{ii}} \tag{5}$$

where $b_i(p_{-i})$ represents the best response of player $i$ given the current state of the game (the power profile for all other players is denoted by $p_{-i}$), $c_i$ represents player $i$'s energy cost, $h_{ij}$ are the channel gains, $p_j$ is the transmitted power for all the other players and $n_0$ is the noise. In typical controlled evaluation setups, such as simulators, the values for $h_{ji}$ and $p_j$ are chosen conveniently to guarantee convergence of the game. In a real-world testbed, however, these parameters have to be acquired from the operating environment and may influence the outcome of the game. As a result, the following adapted expressions were used in the experimental evaluation:

$$b_i(p_{-i}) = \frac{1}{c_i} - \frac{\sum_{j \neq i} h_{ji} p_j + n_0}{h'_{ii}} = \frac{1}{c_i} - \frac{Pr_{measured} i - P_{useful}}{h'_{ii}} \tag{6}$$

$$b_i(p_{-i}) = \frac{1}{c_i} - \frac{\sum_{j \neq i} h_{ji} p_j + n_0}{h'_{ii}} = \frac{1}{c_i} - \frac{Pr_{measured|p_i=0}}{h'_{ii}} \tag{7}$$

where $h'_{ii}$ stands for the measured or the measurement-based estimation of the direct gain, $Pr_{measured} i$ stands for the received power measured by player $i$ when player $i$

is also transmitting ($p_i \neq 0$), $P_{useful}$ stands for the estimated useful power received by $Rx_i$ when $Tx_i$ is transmitting, and $Pr_{measured|p_i=0}$ stands for the received power measured by player $i$ when player $i$'s transmitter is silent ($p_i = 0$).

For the entire system to be stable, the PAPU algorithm must converge to a Nash equilibrium. The convergence issue studied in [21] gives the following condition for the convergence and stability of PAPU:

$$|\frac{h_{ji}}{h_{ii}}| < \frac{1}{N}, i = 1, ..., N \tag{8}$$

Eq. (8) is a decisive factor when choosing the topology of nodes in the testbed on which the power allocation game can be implemented. If Eq. (8) is not fulfilled for all players, there will be no strategy profile that will satisfy the players.

In the theoretical case, considering Eq. (5), an equilibrium is reached in the game when $p_i(t-1) = p_i(t)$ for all players at once. However, in practice, where $p_i$ can take only discrete values, a more robust stopping criterion is needed in order to determine the equilibrium. In this study, the criterion is given by the following condition: $|b_{i-k}(p_{-i}) - b_i(p_{-i})| < P_{th}$, where $k = \overline{0,4}$ and $P_{th}$ stands for the threshold power used to compensate for the environment dynamics.

Based on the PAPU algorithm, we defined the following power control protocol:

- Step 1: Each player $i$ initializes its power $p_i^0, p_{-i}^0$.
- Step 2: At time $t$ if player $i$ updates its power, player $i$ will alert the neighbors that a power change has been made.
- Step 3: If player $i$ detects a change in other players' power, it updates its power according to Eq. (7) and alerts the neighbors of the change.
- Step 4: Player $i$ checks if the Nash equilibrium condition is satisfied
- Step 5: If the Nash equilibrium condition is satisfied, the game is stopped.

The code implementing this protocol is too complex to include in this chapter, however it is publicly available in out repository on GitHub [3].

## *5.4 Empirical parameter determination*

The channel estimation for the topology presented in Figure 25 was performed starting from empirical data. The gain between pairs of nodes has been measured and three strategies were investigated for deriving their values for the purpose of the PAPU algorithm, i.e. average gain, instantaneous gain and estimated gain using Kalman filter.

The channel gain was measured as follows. First, the receiver $Rx_j$ measured the received power $P_{noise}$ knowing that the transmitter $Tx_i$ was silent using the code presented in Figure 26 lines 1-9 and 15-23. Then, the receiver measured the received

[3] https://github.com/sensorlab/logatec-games/blob/master/Power_Allocation_Game/powerAllocation.py

power $P_{Rx_j}$ knowing that the transmitter was transmitting with a given power $P_{Tx_i}$ using the code in Figure 26 lines 1-9, 11-19 and 25-27. The gain (lines 29-31 in Figure 26) was then computed according to:

$$h_{ij} = \frac{P_{Rx_j} - P_{noise}}{P_{Tx_i}} \tag{9}$$

```
1.  # select the pair of nodes
2.  tx_node = Node(coordinator_id, tx_node_id)
3.  rx_node = Node(coordinator_id, rx_node_id)
4.
5.  # settings
6.  p_tx_dBm = 0
7.  sensing_duration = 2
8.  time_after = 1.5
9.  attempts = 0
10.
11. # these lines are needed just for signal measurement
12. tx_node.setGeneratorConfiguration(measuring_freq, p_tx_dBm)
13. tx_node.generatorStart(time.time(), transmitting_duration)
14.
15. #configure rx_node
16. rx_node.setSenseConfiguration(measuring_freq,
17.     measuring_freq, 400e3)
18. #start sensing
19. rx_node.senseStart(time.time()+time_after, sensing_duration, 5)
20.
21. #these lines are needed just for noise computation
22. noise_power = GainCalculations.getAverageDataMeasurementsFromFile(
23.     coordinator_id ,rx_node.node_id)[1][0]
24.
25. #these lines are needed just for noise computation
26. received_power = GainCalculations.getAverageDataMeasurementsFromFile(
27.     coordinator_id,rx_node.node_id)[1][0]
28.
29. #these lines are needed just for gain computation
30. gain = (received_power - noise_power) /
31.     (math.pow(10.00, p_tx_dBm/10.00) * 0.001)
```

**Fig. 26** Example code required to measure the noise between a pair of nodes. (Note: This is example code, the actual working code is available at `https://github.com/sensorlab/logatec-games/blob/master/gain-computation/gainCalculations.py`).

Long term channel gain measurements were performed for 19 consecutive days, between August 5th and August 23rd 2013, two times a day, in the afternoon (12:30 - 13:30) and late evening (21:00 - 22:00). For channel gain measurements, a signal at 2420 MHz with a bandwidth of 400 kHz was used at the transmitter side. The measured direct channel gains for player 1 are represented in a chronological order in the plot depicted in Figure 27. The vertical white and grey shadings represent the afternoon period with white and the late evening period with gray, respectively.

As shown in Figure 27, the average gain for the pair of nodes representing Player 1 is -74.8 dB with a standard deviation of 6.952 dB and a dynamic range of 18.9

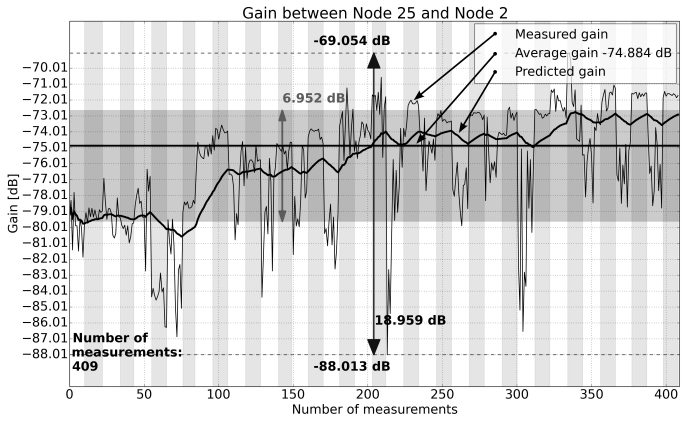


**Fig. 27** Long term measurements of direct channel gains for player 1 (from August 5th to August 23rd 2013 between 12:30 - 13:30 - white vertical shading and 21:00 - 22:00 gray vertical shading).

dB for the 19-day period considered. Even in short term measurement periods, such as an afternoon or an evening session, gain variations of over 5 dB can be found. Similar observations can be made for other pairs of nodes. The gain is highly dynamic and thus using the average as a means of approximation does not reflect the actual state of the channel. Therefore the best-response strategy of the power allocation game will yield unrealistic values. This method would be suitable if the gain variation were not considerable (e.g. $< 3$ dB), which happens only occasionally.

Alternative to average gain as input to the best-response strategy is the the instantaneous channel gain. This approach, however, can prove expensive in terms of delay (for the particular case of the LOG-a-TEC testbed one such measurement takes 5-6 s) and power consumption, and it is prone to measurement errors.

The third approach considered in this use case for deriving the channel gains is to use a simple channel estimation technique such as the Kalman filter. This method consists of two steps: the first step is estimating the next measured gain and the second step represents the correction of the estimated gain by performing a new gain measurement. The resulting predicted gain is then computed. The values of the Kalman predictor for long term measurements are depicted in Figure 27. For a short history, it approximates well the channel gain (see first 3-5 measurements) but it gets worse as the history gets longer.

By analyzing the long-term measurements, three important conclusions reflecting the implementation of the game can be drawn. First, the channel gain Kalman predictor approach is more suitable for the power allocation game because it compensates for the measurement errors to which the instantaneous gain is prone, while also relying on measurement history. Second, the Kalman filter based estimations of the gain are more accurate than the average. This is supported by the smaller values of the mean square error between the Kalman estimations and the instantaneous value (K−inst) than the values of the mean square error between average and instantaneous values (avg−inst) as listed in Table 5. Third, using a shorter history for the

Kalman predictor yields better results while also having some practical advantages as far as implementation is concerned. At the beginning of each experiment, the Kalman predictor is used with a history of 9 most recent measurements.

| MSE | Average gain (avg−inst) | Predicted gain (K−inst) |
|---|---|---|
| $h_{11}$ | $4.65 \times 10^{-16}$ | $2.14 \times 10^{-16}$ |
| $h_{12}$ | $7.9 \times 10^{-17}$ | $1.59 \times 10^{-17}$ |
| $h_{21}$ | $2.8 \times 10^{-19}$ | $2.55 \times 10^{-19}$ |
| $h_{22}$ | $1.26 \times 10^{-15}$ | $8.11 \times 10^{-16}$ |

**Table 5** Mean squared error for the average and predicted gain with respect to instantaneous gain.

## *5.5 Experimental results*

The setup presented in previous subsections was used for experimental performance evaluation of the PAPU algorithm for power allocation and interference mitigation. In particular, experimental results were obtained to show (i) the effect of energy cost on the best response of the players, (ii) the effect of the channel gains on the best response of the players, and (iii) the influence of real-world operating conditions on reaching a Nash equilibrium.

The best response formula used for the empirical evaluation of the game is given in Eq. 5. The first term of the formula is the cost $c_i$ whose role is to penalize the players for transmitting with high power. The feasible values of the cost also have to take into account the power levels supported by the testbed. In the case of LOG-a-TEC, the best response has to respect the lover boundary of -55 dBm and the upper boundary of 0 dBm. This constraint excludes a large set of low values of the cost. Furthermore, the experiments have shown that a very high cost value prevents the game to converge to an equilibrium making the communication system unstable. This observation has led to putting a higher boundary on the cost value. In the worst case scenario observed during the experiments, when the $I + n_0$ is -84 dBm for the first player and -72 dBm for the second player, the feasible values for $c_i$ lie in the $[1000; 4000]$ interval.

The gain is another of the parameters of the best response formula from Eq. 5 and the logarithmic representation of the dependency of $b_i(p_{-i})$ to $h_{ii}$ for Player 1 is depicted in Figure 28. For a fixed cost of $c_i = 1000$ for both players and the worst case scenario where the values of $I + n_0$ are $-88$ dBm for player 1, it can be seen that the best response has high variations for small gains and negligible variations for high ones.

The existence of a Nash equilibrium for the PAPU algorithm has been proven in [21]. The simulated value of the Nash equilibrium for PAPU considering $c_i = 1000$ and average value of the gain is (-0.24, -1.25) as can be seen in Figure 29. By

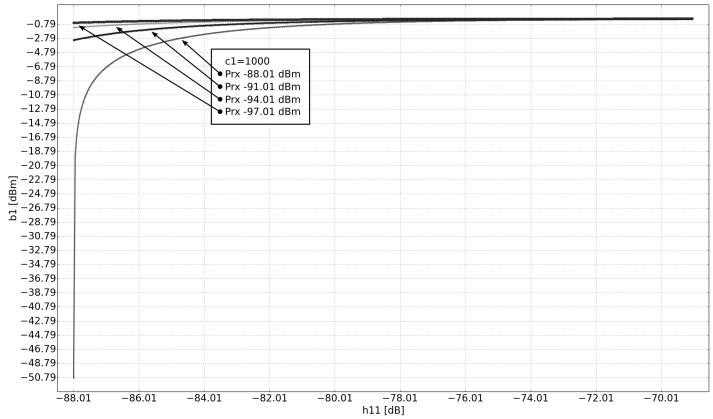


**Fig. 28** The influence of the channel gains on the best response for Player 1 $b_1(p_{-1})$ versus $h_{11}$ and $I + n_0$.

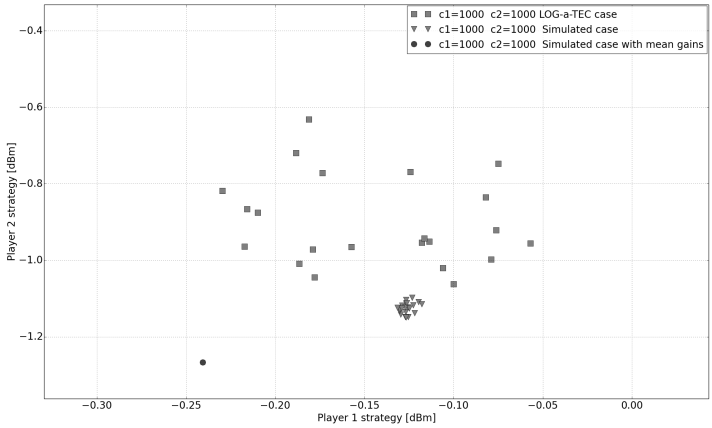


**Fig. 29** Experimental Nash equilibrium compared to simulated one for two players.

introducing more realistic gains based on the Kalman estimator, the simulated Nash equilibrium slightly varies around (-0.14, -1.1), as depicted with triangles in Figure 29. In this case the influence of the predicted gains to the player's final strategies is evident.

By running the game on the real−world testbed, the Nash equilibria are more spread as shown with squares in Figure 29, mostly due to interference and noise. Still, the variation of the best response is relatively small with 0.3 dBm for Player 1 and 0.5 dBm for Player 2. In the first two cases where simulation was used, the Nash equilibrium results in a (0, -2) value. For the case of LOG-a-TEC, in the experimental setup, the multitude of Nash equilibria obtained in different runs, after being rounded to the discrete values of the transmission power supported in LOG-a-TEC, are (0,0) and (0,-2).

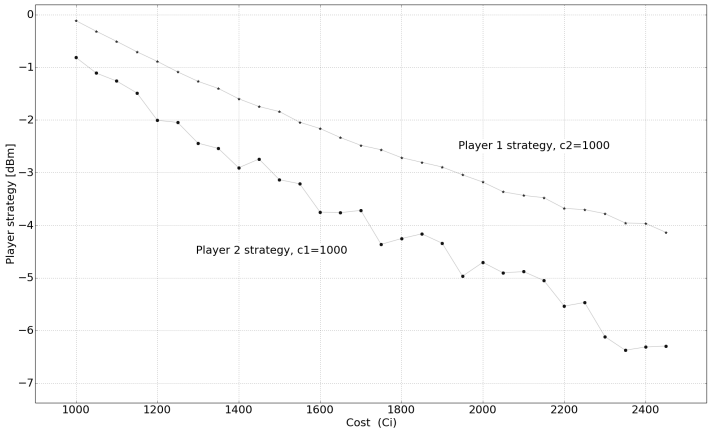


**Fig. 30** Players' best response as function of players' costs.

Figure 30 depicts player strategies for different cost schemes. It can be seen that with the increase of the cost $c_i$, there is a decrease of the allocated powers. All the experiments were performed as following: the channel gains $h_{ii}$ were computed at the beginning of each experiment by predicting the gain, interference and noise power levels. $I + n_0$, were measured and updated at each iteration, $k = \overline{0,4}$ and $P_{th} = 0.8$ dBm. For Figure 30, all the experiments were repeated 20 times and the Nash equilibrium are given as averaged values.

## *5.6 Summary*

This section demonstrated the role of a testbed such as LOG-a-TEC in validating techniques proposed for cognitive radios with more detailed insights available in [19]. We selected the ProActive Power Update algorithm based on game theoretic framework, and showed how it can be adapted for validation on LOG-a-TEC. Besides validating the proposed framework and algorithms, the effort enabled studying the effects of the empirical parameter estimation on the best response and players' strategies which represent the Nash Equilibria. The results showed that for a certain cost range, the system can reach the Nash equilibria. The equilibria and the convergence time are strongly influenced by the cost but also by the channel gains, requiring special care in selecting the topology of nodes within the testbed.

These results may prove useful in developing new protocols in decentralized CR networks and the current implementation is openly available in our repository on GitHub [4].

[4] `https://github.com/sensorlab/logatec-games`

## 6 Summary of the chapter

In this chapter we described in our view and based on our experience the most important of the aspects related to the design and deployment of the LOG-a-TEC embedded, outdoor cognitive radio testbed. We started the chapter by discussing radio front-end for embedded testbeds and described the SNE-ISMTV expansion board custom designed for the VESNA sensor node for supporting cognitive radio experimentation with low cost low-power devices. The expansion board can be equipped with off-the-shelf reconfigurable transceivers for experimentation in license free SRD and ISM bands at 868 MHz and 2400 MHz, respectively, or with more sophisticated custom designed hardware operating in VHF and UHF TV bands. The chapter continues with practical experiences for designing the testbed infrastructure, like the management network and our considerations in the choice of network protocols employed in the LOG-a-TEC testbed.

After describing all these aspects of LOG-a-TEC, we provide two use cases that can be supported by the testbed. The first one consists of generating a wireless microphone profile for realizing experiments in UHF bands where the corresponding SNE-ISMTV-UHF is only capable of receiving. The wireless microphone profiles are generated using the SNE-ISMTV-TI868 extension and can be used for instance to perform coexistence experiments where the spectrum sensing data is fed into a hybrid geolocation database. The second use case consists of the validation of an interference mitigation strategy using a game theoretic framework. The discussion includes the adaptations required by the theoretical framework as well as discussion with respect to the constraints posed by the testbed operating in real-world environment. An evaluation of the approach and discussion related to the trade-offs that were observed is also provided.

In summary, the chapter gives a quick insight into the potential for experimentally driven research held by embedded testbeds consisting of low cost low-power devices. At the same time it points out some important choices one needs to make in the process of the design, which have significant impact on subsequent usability and functionality of the testbed. These especially include the issues related to topology design, power supply, network management, remote access and reconfigurability of devices.

## 7 Acknowledgements

The authors would like to thank all our colleagues from SensorLab and the external collaborators who contributed directly or indirectly to this book chapter. This work was partially supported by the FP7 project CREW (FP7-ICT-258301) and by the Slovenian Research Agency through the research grants P2-0016 and J2-4197, and it was contributed also to the COST Action IC0902.

## References


1. "TmoteSky ultra low power IEEE 802.15.4 compliant wireless sensor module." http://www.eecs.harvard.edu/~konrad/projects/shimmer/references/tmote-sky-datasheet.pdf, Jan. 2014.
2. "VESNA sensor node." http://sensorlab.ijs.si/hardware.html, Jan. 2014.
3. A. Dunkels, B. Gronvall, and T. Voigt, "Contiki-a lightweight and flexible operating system for tiny networked sensors," in *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, pp. 455–462, IEEE, 2004.
4. P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, *et al.*, "TinyOS: An operating system for sensor networks," in *Ambient intelligence*, pp. 115–148, Springer, 2005.
5. "CREW - Cognitive Radio Experimentation World." http://www.crew-project.eu/, Jan. 2014.
6. E. Blossom, "GNU radio: tools for exploring the radio frequency spectrum," *Linux journal*, vol. 2004, no. 122, p. 4, 2004.
7. P. D. Sutton, J. Lotze, H. Lahlou, S. A. Fahmy, K. E. Nolan, B. Ozgul, T. W. Rondeau, J. Noguera, and L. E. Doyle, "Iris: an architecture for cognitive radio networking testbeds," *Communications Magazine, IEEE*, vol. 48, no. 9, pp. 114–122, 2010.
8. T. Šolc, "SNE-ISMTV: VESNA wireless sensor node expansion for cognitive radio experiments," in *Wireless Communication Systems (ISWCS 2013), Proceedings of the Tenth International Symposium on*, pp. 1–2, VDE, 2013.
9. C. A. Boano, J. Brown, Z. He, U. Roedig, and T. Voigt, "Low-power radio communication in industrial outdoor deployments: The impact of weather conditions and atex-compliance," in *Sensor Applications, Experimentation, and Logistics*, pp. 159–176, Springer, 2010.
10. S. Lanzisera, A. M. Mehta, and K. S. J. Pister, "Reducing average power in wireless sensor networks through data rate adaptation," in *Communications, 2009. ICC'09. IEEE International Conference on*, pp. 1–6, IEEE, 2009.
11. M. Mohorcic, M. Smolnikar, and T. Javornik, "Wireless sensor network based infrastructure for experimentally driven research," in *Wireless Communication Systems (ISWCS 2013), Proceedings of the Tenth International Symposium on*, pp. 1–5, VDE, 2013.
12. "LOG-a-TEC Cognitive radio." http://log-a-tec.eu/cr.html, Jan. 2014.
13. A. Hrovat, I. Ozimek, A. Vilhar, T. Celcer, I. Saje, and T. Javornik, "Radio coverage calculations of terrestrial wireless networks using an open-source grass system," *Transactions on Communications, WSEAS*, vol. 9, no. 10, pp. 646–657, 2010.
14. T. Šolc and Z. Padrah, "Network design for the log-a-tec outdoor testbed," *2nd International Workshop on Measurement-based Experimental Research, Methodology and Tools*, 2013.
15. "ZigBit 700/800/900 MHz Wireless Modules datasheet." http://www.atmel.com/Images/doc8227.pdf, Jan. 2014.
16. "Atmel AVR2050: Atmel BitCloud Developer Guide." http://www.atmel.com/images/doc8199.pdf, Jan. 2014.
17. "AVR Low Power 700/800/900 MHz Transceiver for IEEE 802.15.4-2006, IEEE 802.15.4c-2009, ZigBee, 6LoWPAN, and ISM Applications Data Sheet." http://www.atmel.com/images/doc8168.pdf, Jan. 2014.
18. C. Clanton, M. Kenkel, and Y. Tang, "Wireless microphone signal simulation method," *IEEE 802.22-07/0124r0*, 2007.
19. C. Anton, A. Toma, L. Cremene, M. Mohorcic, and C. Fortuna, "Power Allocation Game for Interference Mitigation in a Real-world Experimental Testbed," in *2014 IEEE International Conference on Communications*, IEEE, 2014.
20. J. Huang, R. A. Berry, and M. L. Honig, "Distributed interference compensation for wireless networks," *Selected Areas in Communications, IEEE Journal on*, vol. 24, no. 5, pp. 1074–1084, 2006.
21. G. Fang, E. Dutkiewicz, K. Yu, R. Vesilo, and Y. Yu, "Distributed inter-network interference coordination for wireless body area networks," *Global Telecommunications Conference (GLOBECOM 2010)*, pp. 1–5, 2010.