A Performance Study of the Behavior of the Wake-Up Radio in Real-World Noisy Environments

Sebastian Lucas Sampayo
ICube Laboratory
University of Strasbourg
sampayo@unistra.fr

Julien Montavont
ICube Laboratory
University of Strasbourg
montavont@unistra.fr

Thomas Noel
ICube Laboratory
University of Strasbourg
noel@unistra.fr

Abstract

Wake-Up Radio (WuR) is a cutting-edge technology for the Internet of Things that is going to change the way end-devices communicate. Asynchronous wireless communications can benefit from WuR to reduce both energy consumption and latency comparatively to well-known duty-cycled solutions. In this article, we present an experimental platform using an existing WuR prototype and analyze its behavior when it is subject to radio interference. We implemented for the first time clear channel assessment capabilities and show that it can improve the packet delivery ratio by up to 10% on average. In particular, it can improve it from 25% up to 85% for internal interference. Besides, we experimentally extract some key physical values of this technology to provide inputs for WuR-based simulations and analytical models. Finally, we analyze the overall current consumption of a simple application to gain new insights into the WuR behavior. In low traffic scenarios, our results show that optimizing further the communication protocol stack will not significantly increase the lifetime of end-devices.

1 Introduction

In wireless sensor networks the end-devices are generally low-cost and resource-constrained. In consequence, their lifetimes are limited reducing the benefits in the application [3]. Radio communications are generally the main source of energy consumption, so a tremendous research effort have been triggered, both on hardware and software sides, to optimize the communication protocol stack to improve the lifetime of end-devices [4]. Traditionally, this is achieved by putting the radio to sleep as much as possible with some form of duty-cycled MAC protocol [9]. However, there is still a lot of waste of energy in these solutions because of the *idle-listening* - when a node listens to the channel but there is no ongoing transmission, and *overhearing* - when a node receives packets destined to another node. Also, the duty-cycle increases the latency of the application because of the limited communication window in each cycle.

In the recent years, a new technology for radio communications called Wake-Up Radio (WuR) has advanced with the promise of ending that latency-energy trade-off by reducing them both at the minimum [11], [14]. WuR consists of a secondary ultra-low power receiver that stays always-on listening to the channel while the main radio for data communication stays sleeping and wakes up on demand by the WuR. This way the energy consumption in radio communication is reduced to the minimum because the main radio is only turned on when necessary.

In this article, we perform a study on the behavior of a WuR prototype when it is subject to a real-world noisy environment. We analyze how interference can be wrongly considered as valid packets and how to deal with them. We show the importance of the utilization of Clear Channel Assessment (CCA) capabilities to reduce transmission errors, resulting in a higher Packet Delivery Ratio (PDR). Besides, we extract some key physical values of the prototype that can serve in modeling WuR communications. Finally, we provide a method to estimate the overall current consumption of an application deployed over a WuR-based network. The results show that radio communications account for a negligible part of the energy depletion in low traffic scenarios, meaning that further optimizations in the communication protocol stack will not improve the lifetime of end-devices in such cases.

The rest of the article is organized as follows. A brief introduction of the WuR technology is provided in Section 2. In Section 3 we review the existing solutions that make use of CCA in WuR communications together with the relevant performance evaluations of the WuR technology. Our CCA algorithm and current consumption model are respectively detailed in Sections 4 and 5. Section 6 describes the experimental platform setup. Finally, the results are analyzed in Section 7 and the article concludes in Section 8.

2 Wake-Up Radio

WuR is a novel technology for radio communication that has been researched over the past few years. It consists of an ultra-low-power receiver that demodulates On-Off Keying (OOK) signals into digital data and feeds it into an ultra-low-power microcontroller (MCU).

International Conference on Embedded Wireless Systems and Networks (EWSN) 2020
17–19 February, Lyon, France
ISBN: 978-0-9949886-4-5

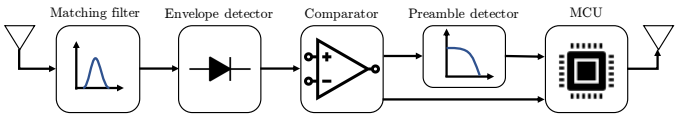

Figure 1. Wake-Up Radio building blocks

In the following, we describe the building blocks of the WuR prototype [10], [12] used in our work, illustrated in Fig. 1. The first stage consists of a *matching filter* to optimize the power transfer of the antenna to the receiver circuit at 868 MHz. Next, an *envelope detector* rectifies the RF signal generating an OOK demodulated output with low voltage. That signal is compared to an *adaptive threshold* to generate the digital data that is read by the MCU. Then, a low-pass filter after the *comparator* acts as a *preamble detector* to generate an interrupt in the MCU only when the OOK signal presents a positive pulse of at least around 1 ms, which corresponds to a data rate of 1 kbps. Finally, the PIC16LF1824T39A [2] is used as the MCU to decode the packet data and interpret the preamble detector interrupt. This microcontroller has an RF transmitter embedded with OOK modulation, a maximum of +10 dBm output power and a data rate of up to 10 kbps.

3 Related work

In [5], the authors integrated a WuR prototype [10] into a wireless mote and performed transmission range measurements between two nodes. They showed that the maximum transmission distance is around 21-24 m while the packet delivery ratio remains above 95%. They also categorized packets as *Positives* - packet that should be considered as valid by the receiver, and *Negatives* - packets that should be discarded by the receiver. *Positives* could be splitted in *True Positives* - packets that were effectively sent by a valid source, and *False Positives* - packets that were not transmitted by a valid source. Generally, noise or external interference are the source of *False Positives*. *Negatives* includes *False Negatives* - packets that were transmitted by a valid source but were corrupted somehow, and *True Negatives* - packets that were not transmitted by a valid source. They showed that *False Negatives* remain low (below 1%) while *False Positives* are negligible (stand for 0.02%). However, the *True Negatives* in [5] are generated artificially by manually corrupting the original packet. This leads to a high percentage of *True Negatives* in the results, which only means that the receiver decodes correctly those artificially-generated packets. In our study, we analyze the whole set of *Negatives* that are completely generated by noise or external interference, approaching more realistic scenarios. We qualified and quantified those packets to show that the performance of this technology is also dependent on the levels of interference in a noisy environment.

The need for CCA capabilities has been already stated in previous works [8], [13]. However, no implementation has been done so far to take advantage of the already developed hardware and provide a CCA module in any WuR prototype. To best of our knowledge, the present article is the first analysis of interference patterns of a real WuR prototype with CCA capabilities in a noisy environment.

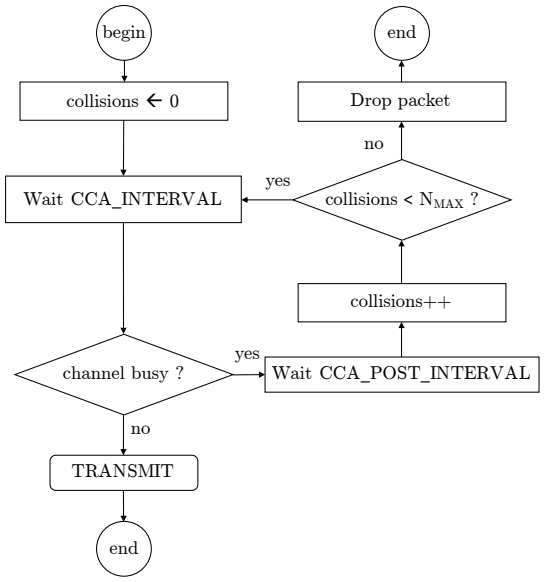

Figure 2. Transmission attempt flowchart with Clear Channel Assessment

4 Clear Channel Assessment

The CCA function takes advantage of the preamble detector module in the WuR circuit to sense activity in the channel. If a signal strong enough is received, this module generates an interrupt and turns on a *busy-channel* flag. The algorithm to transmit a packet is illustrated in Fig. 2. Whenever the transmitter wants to send a packet, it first waits for 1 ms (*CCA_INTERVAL*) and checks during that period if this flag turns on. In the negative case, it just sends the packet. Otherwise the channel is considered as busy, so it waits for a fixed amount of time called *CCA_POST_INTERVAL*, turns off the flag and restarts the procedure.

If the channel is busy due to another node transmitting a packet on the WuR, *CCA_POST_INTERVAL* should be set to the time required to transmit a packet. However, if the channel is busy due to noise or external interference, then this duration should be shorter to reduce the transmission delay. As the source of a busy channel can not be known in advance, we set the *CCA_POST_INTERVAL* to the time required to transmit half the length of a packet as a trade-off between delay and the number of performed CCA (numerous CCA attempts result in dropping packets). Notice that we can not reduce *CCA_POST_INTERVAL* to zero because the channel might be considered as free while a node is transmitting a sequence of zeros.

In the rest of the article we are going to use the words *noise*, *interference*, and *collisions* as the same concept. Since the noise comes from a real environment, we do not know exactly the source of it. It can be electrical noise, thermal noise, collisions with other packets, interference with other RF signals, and so on. Special instruments, like a spectrum analyzer, may be used to reveal the source of the noise. In this work, we remained agnostic of the medium and did not use any of those instruments.

5 Lifetime model

The main purpose of WuR is to reduce the energy consumption of radio communication in wireless sensor applications. We propose here a set of equations that are necessary to estimate the battery lifetime of a generic node that transmits λ_{tx} and receives λ_{pkts} packets per second in mean values on the WuR channel. For simplicity, we do not consider transmission over the main radio. Moreover, we do not consider any specific communication protocol, apart from the use of the CCA algorithm.

The lifetime of an end-device depends on the effective battery capacity $C_{battery}$ and the average current consumption $\bar{I}_{total}$ drained by the application, and can be estimated with Eq. 1 [6].

$$Lifetime \approx \frac{\bar{I}_{total}}{C_{battery}} \tag{1}$$

The overall current consumption of the WuR module is composed of the part spent in each state of the application, which can be reduced to transmission attempt, reception, and idle.

$$\bar{I}_{total} = \bar{I}_{st\_rx} + \bar{I}_{st\_tx} + \bar{I}_{st\_idle} \tag{2}$$

In particular, the energy spent in the reception state can be written as

$$\bar{I}_{st\_rx} = I_{rx}\alpha_{rx} + I_{cpu}\alpha_{cpu} \tag{3}$$

where I_x is the instantaneous current consumption when the device is in power mode x, and α_x is the fraction of a second spent in power mode x (it is non-dimensional). However, in the WuR circuit the variation in the current consumption when it is receiving an active signal is insignificant, so I_{rx} and I_{cpu} are approximately the same, $I_{rx} \approx I_{cpu}$.

$$\bar{I}_{st\_rx} = I_{rx}(\alpha_{rx} + \alpha_{cpu}) \tag{4}$$

Consequently, the sum $\alpha_{rx} + \alpha_{cpu}$ is essentially the fraction of a second spent on the receiving state α_{st_rx}.

$$\bar{I}_{st\_rx} = I_{rx}\alpha_{st\_rx} \tag{5}$$

which can be estimated by the next equation:

$$\alpha_{st\_rx} = \Delta t_{read}\lambda_{pkts} + \Delta t_{invalid}\lambda_{invalids} \tag{6}$$

where Δt_{read} is the time spent reading a packet, λ_{pkts} is the mean rate of received packets per second, $\Delta t_{invalid}$ is the time spent processing a signal with an invalid preamble and finally $\lambda_{invalids}$ is the mean rate of received signals with an invalid preamble per second. More specifically, the time spent on reading a packet is the sum of the length of the preamble, the length of the packet and the additional time the MCU requires to process the data.

$$\Delta t_{read} = \Delta t_{preamble} + \Delta t_{pkt} + \Delta t_{cpu\_rx} \tag{7}$$

The WuR receives and reads both *positive* packets (the ones that were transmitted by another node of the network) and *negative* packets (those that were produced from noise, interference or another network)

$$\lambda_{pkts} = \lambda_{positives} + \lambda_{negatives} \tag{8}$$

And in particular, for the positives, it only receives a fraction of the ones transmitted by the sender, depending on the reception success ratio (PDR) ρ:

$$\lambda_{positives} = \lambda_{tx}\rho \tag{9}$$

On the other hand, the average current consumption in the transmission state can be calculated with

$$\bar{I}_{st\_tx} = I_{tx}\alpha_{tx} + I_{cpu}\alpha_{cpu\_tx} \tag{10}$$

$$\alpha_{tx} = (\Delta t_{preamble} + \Delta t_{pkt})\lambda_{tx} \tag{11}$$

$$\alpha_{cpu\_tx} = (\bar{N}_{cca}\Delta t_{cca} + \Delta t_{cpu\_tx})\lambda_{tx} \tag{12}$$

where $\bar{N}_{cca}$ is the mean number of transmission attempts due to the CCA algorithm detecting a busy channel, and Δt_{cca} is the time spent in each CCA cycle, which corresponds to $\Delta t_{cca} = CCA\_INTERVAL + CCA\_POST\_INTERVAL$.

Finally, we assume that the device sleeps in low power mode during the idle state, so the average current consumption spent in that case is the sleep mode current scaled by the fraction of a second spent on this state:

$$\bar{I}_{st\_idle} = I_{sleep}(1 - \alpha_{st\_rx} - \alpha_{tx} - \alpha_{cpu\_tx}) \tag{13}$$

In order to analyze the contributions of the useful radio communication and the noise, we can decompose the current consumption of the reception state into the contribution of the actual protocol and the one of the noise, based on equations 5, 6 and 8.

$$\bar{I}_{st\_rx} = \bar{I}_{rx\_protocol} + \bar{I}_{noise} \tag{14}$$

$$\bar{I}_{rx\_protocol} = I_{rx}\Delta t_{read}\lambda_{positives} \tag{15}$$

$$\bar{I}_{noise} = I_{rx}(\Delta t_{invalid}\lambda_{invalids} + \Delta t_{read}\lambda_{negatives}) \tag{16}$$

Then, the total current consumption can be re-written as:

$$\bar{I}_{total} = \bar{I}_{rx\_protocol} + \bar{I}_{st\_tx} + \bar{I}_{noise} + \bar{I}_{st\_idle} \tag{17}$$

and finally, aggregating the components related to the communication protocol:

$$\bar{I}_{protocol} = \bar{I}_{rx\_protocol} + \bar{I}_{st\_tx} \tag{18}$$

$$\bar{I}_{total} = \bar{I}_{protocol} + \bar{I}_{noise} + \bar{I}_{st\_idle} \tag{19}$$

6 Experimental platform

The network is composed of three nodes: one sink and two transmitters referred to as transmitter 1 and transmitter 2. The sink is only receiving and logging every received signal to a computer. A 2-bytes length signal is considered as packet when the preamble is correctly received. Signals with invalid preamble are categorized as *Invalids* and are discarded by the sink. Transmitters are sending a 2-bytes length packet every second at 1 kbps. The content of each packet is respectively set to 0xAA1A and 0xA1AA for transmitter 1 and transmitter 2. Packets whose content matches the expected content (0xAA1A or 0xA1AA) are referred to as *Positives*. On the contrary, *Negatives* are packets whose content

Figure 3. Experimental platform

differs from the expected content. In addition, transmitter 2 uses our CCA module while transmitter 1 does not.

All nodes are using a 3.6 dBi gain antenna (ANT-868-CW-RCS [1]). For availability purposes, the transmitters are fed with a 3 V supply power (2 AAA batteries), while the sink is fed with 3.3 V, powered by an Arduino UNO that acts as a bridge between the WuR prototype [12] (through I2C communication) and the computer (through UART communication). Our preliminary experimentation campaign, performed in a controlled environment with no interference, showed that we can achieve a 100% packet delivery ratio with no *Negatives* (being *True Negatives* or *False Negatives*) or *False Positives* when the transmitter and the receiver are separated by a distance below 1 m. For the present campaign, we placed the transmitters and the sink at a distance of 70 cm to compare the results with that ideal case. Such a short distance reflects scenarios involving a very dense network and enables transmitters to experience the same interference. The experiment was performed indoors, next to a window in a regular office at the ICube Laboratory of the University of Strasbourg as depicted in the Fig. 3.

To measure the instantaneous current consumption of the device in each mode (I_{tx}, I_{rx}, I_{sleep} and I_{cpu}) we load specific pieces of firmware to keep the device under test (DUT) in a single mode continuously. The current measurements were taken with the multimeter FLUKE 177 True RMS in DC mode. For the sleep mode, the multimeter was used in the DC voltage mode to measure the voltage drop in a 1 kΩ resistor added in series with the power supply. In all cases, the measured value has an error between 1% and 2%, given the specified precision of reading and number of least significant digits of the multimeter, as well as the error propagation for the sleep current case. Notice that in the TX mode, the DUT was transmitting continuously a sequence of bits set to '1' at +10 dBm, so it is the worst case of maximum current consumption in that mode. In the RX mode, the DUT is continuously waiting for a reception interrupt and a transmitter (transmitting continuously a sequence of bits set to '1') was placed close to it. The CPU mode used an infinite loop incrementing a dummy variable with all the required peripherals turned on (mainly the timers and the I2C module) and no compiler optimizations. Finally, the sleep mode simply sets the DUT in low power mode. In each case the current consumption was measured for the whole board which includes the WuR receiver circuit as well as the ultra-low-power MCU circuit.

Finally, we measured the real value of all Δt variables defined in Section 5. Measurements were taken signaling a pin

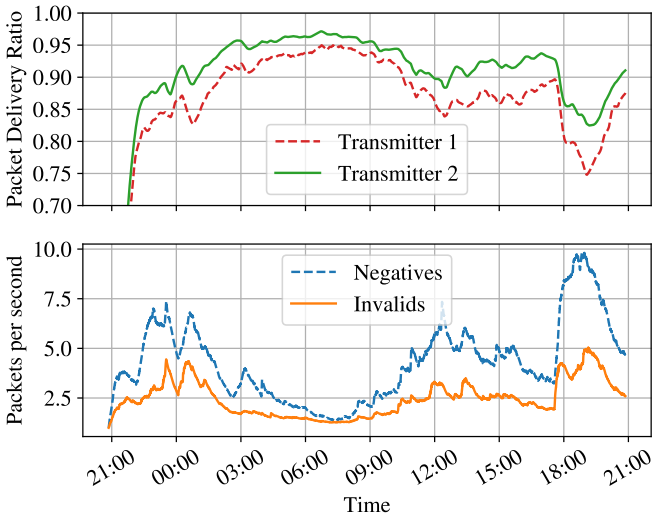

Figure 4. Packet delivery ratio and interference

on the MCU and measuring timing differences with the cursors of a digital oscilloscope Tektronix TBS1104.

7 Results

We performed numerous experimentations and all collected results were very similar. For clarity reason, results presented in this section are extracted from one experimentation, corresponding to a duration of 24 hours, which is why standard deviation or confidence intervals can not be calculated.

7.1 Packet delivery ratio

Fig. 4 shows the PDR achieved by each transmitter together with the number of *Negatives* and *Invalids* received per second. *Negatives* are packets that were received but their contents differ from the original content (i.e. differ from 0xAA1A and 0xA1AA) while *Invalids* are signals with invalid preambles. We can see that there is a high correlation between the dynamic of the noise and the behavior of the PDR for both transmitters. During quiet moments, i.e. when the noise is low, we see that both transmitters achieve their maximum PDR (e.g. 95% for transmitter 1 and 97.5% for transmitter 2 around 7h00). Transmitter 2, with CCA, has better performance, though. This can be explained if we notice that both transmitters synchronize their transmission phase every approximately 15 mins because of the clock drift, as illustrated in Fig. 5. During each synchronization period, the PDR of transmitter 1 experiences severe drops reaching down to 25% while transmitter 2 maintains a PDR higher than 85% thanks to our CCA algorithm.

Moreover, transmitter 1 is more sensitive to noise peaks. When the noise level is higher, for example around 18h00, we see that the PDR of transmitter 1 decreases to 75%, while transmitter 2 maintains a PDR higher than 82%. This means an improvement of 10% when using CCA in an environment of $\lambda_{negatives} = 10$ and $\lambda_{invalids} = 5$.

7.2 Packet contents

Fig. 6 shows the count of the 20 most significant *Negatives* received by the sink, which represent 55% of the to-

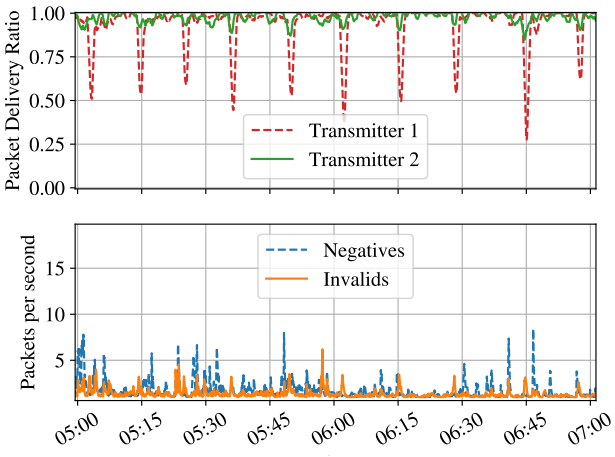

Figure 5. Transmission phase synchronization because of the clock drift

Table 1. Current consumption measurements for V = 3 V

Mode	Current [mA]
I_{tx}	14.3
I_{rx}	0.40
I_{sleep}	0.00761
I_{cpu}	0.40

tal number of *Negatives*. We can see that most of *Negatives* are long sequences of bits set to '1', namely 0xffff, 0x7fff, 0x3fff, and so on. This can be explained considering the modulation used in other wireless communication technologies where the carrier frequency is always-on during the packet transmission. For example, in frequency shift keying or phase shift keying, the signal is always-on during the packet transmission and the variations in frequency or phase respectively represent the different symbols. Since the WuR circuit is an envelope detector for OOK it can not distinguish those variations. Instead, it simply translates an on signal to a bit set to '1' and an off signal to a bit set to '0'. We are therefore convinced that the main source of *Negatives* are external wireless communication technologies.

Comparatively, if we look at the 20 closest *Negatives* to the original content (0xAA1A and 0xA1AA), they respectively represent 0.26% and 0.1% of the total number of *Negatives* for transmitter 1 and transmitter 2. Those packets were mostly generated by collisions, inverting some bits or just shifting the whole content. Thanks to CCA, we can see that transmitter 2 is less prone to collisions.

7.3 Current consumption

The instantaneous currents measured with the multimeter for each mode are presented in Table 1. On the other hand, the timing measurements done with the oscilloscope are shown in Table 2. Finally, the equations of the current consumption model presented in Section 5 were computed with the values of Table 3.

Fig. 7 shows the current consumption contribution of each term present in Equation 19 over a range of Inter Packet In-

Table 2. Timing measurements

Variable	Value [ms]
$\Delta t_{preamble}$	3
Δt_{pkt}	16
Δt_{cpu_rx}	3
Δt_{cpu_tx}	4.4
Δt_{cca}	10
$\Delta t_{invalid}$	4

Table 3. Parameters for modeling the current consumption

Variable	Value
$\lambda_{invalids}$	0-2 packets per second
$\lambda_{negatives}$	0-2 packets per second
λ_{tx}	1/100 - 1/10 packets per second
ρ	0.9
N_{cca}	10

tervals (IPI). For low traffic scenarios ($IPI > 45$ s), we can see that the idle state contributes more to the current consumption than the effective radio communication ($\bar{I}_{protocol}$). However, when the environnement is noisy, the receiver experiences many false wake ups and $\bar{I}_{noise}$ becomes the main source of current consumption. This means that, for very low traffic applications, improving further the communication protocol stack can only marginally increase the lifetime of end-devices. The IPI threshold where $\bar{I}_{protocol}$ remains the main contributor to the current consumption depends on the amount of noise ($\lambda_{negatives}$ and $\lambda_{invalids}$). As the level of noise goes up, the threshold goes down. Said differently, $\bar{I}_{protocol}$ can only be greater than $\bar{I}_{noise}$ when $\lambda_{positives} >> \lambda_{negatives}$.

8 Conclusions and future work

In this article, we presented a performance study of WuR in a real-world environment. An experimental platform was deployed to log data from a WuR-based receiver into a computer for further analysis and post-processing. We implemented a software module to perform CCA without further modifications of the hardware. In Section 7 we showed that this module improves the performance of the communication in noisy environments and reduces the number of collisions, contributing to increase the overall packet delivery ratio. In particular, it can improve the PDR from 25% up to 85% for internal interference.

We also analyzed the behavior of external interference and how they affect the performance of the WuR technology. We categorized errors as *Negatives* - packets received with a wrong content and *Invalids* - signals with an invalid preamble. The results showed that external interference mainly generate *Negatives* composed of a sequence of bits set to '1'. Using line coding scheme like 4B5B would force transitions in legacy signals and therefore would help to discard such erroneous packets.

Finally, a current consumption model was presented and evaluated based on the measurements we performed on a real prototype and the outcomes of the interference experiments. Our analysis throws new insights on the WuR behavior: in low traffic scenarios, optimizing the communication proto-

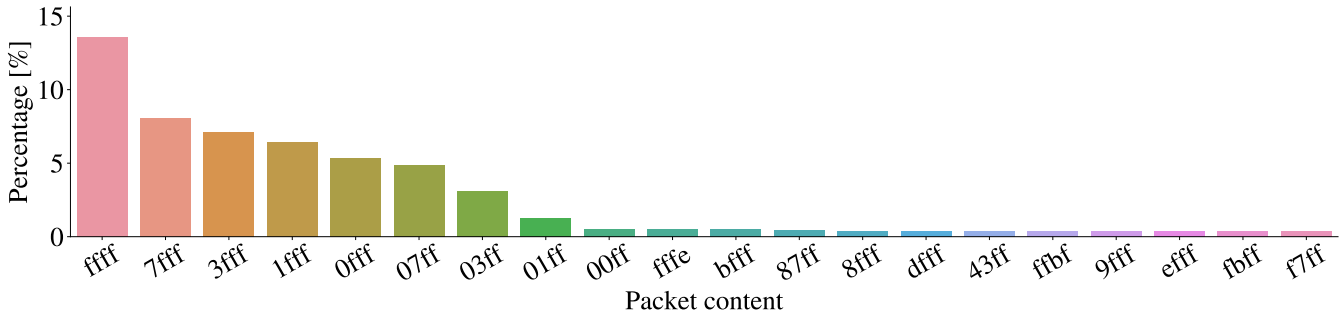

Figure 6. Negatives packets histogram

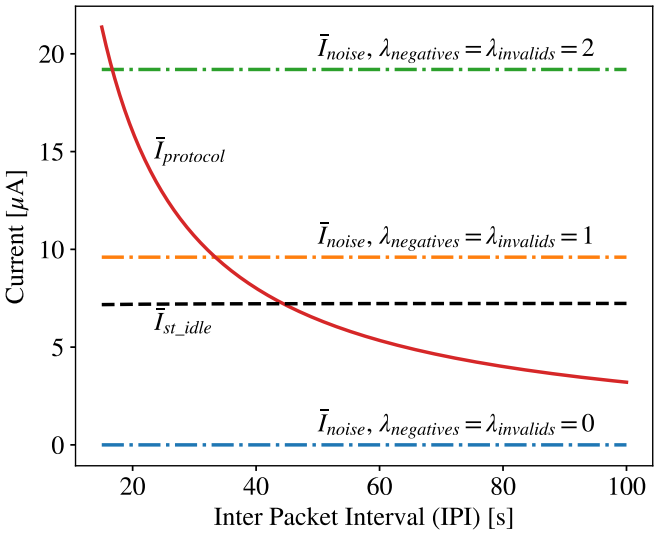

Figure 7. Current consumption contributions

col stack will only marginally increase the lifetime of the end-devices. The energy spent on false wake ups due to ambient noise or the minimum energy spent on the idle state can be more significant than the average energy consumed by the communication protocol. However, the number of false wake ups can be reduced by using early sleeping techniques [7].

As a future improvement, the experiment could be carried out through longer periods (weeks instead of days) and at a larger scale (in terms of distance and number of end-devices). Moreover, adding a proper electromagnetic shield would make the prototypes less prone to external interference. Also, a more complex test could be carried out by connecting the transmitters to the computer to log their data. This way, we would have more flexibility to control the data transmitted and differentiate exactly *Negatives* due to noise and the ones due to collisions. A random packet generation process could also be used to reduce the synchronization effect because of the clock drift between the transmitters. Finally, with a longer preamble, the firmware should be ready to communicate at 5 kbps, and with some code optimizations it might scale up to 10 kbps. Such throughput could improve the overall WuR performance by reducing the transmission delay and the contention in the medium.

9 References

[1] Linx technologies, ANT-868-CW-RCS.

[2] Microchip, PIC16LF1824T39A datasheet.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, Mar. 2002.

[4] L. Atzori, A. Iera, and G. Morabito. The internet of things: A survey. *Computer Networks*, 54(15):2787 – 2805, 2010.

[5] S. Basagni, F. Ceccarelli, C. Petrioli, N. Raman, A. V. Sheshashayee, and E. Dept. Wake-up Radio Ranges: A Performance Study. page 6.

[6] S. Farahani. *ZigBee Wireless Networks and Transceivers*. Elsevier, 2008.

[7] D. Ghose, A. Frytlog, and F. Y. Li. Enabling Early Sleeping and Early Data Transmission in Wake-up Radio-enabled IoT Networks. *Computer Networks*, 153, Mar. 2019.

[8] D. Ghose and F. Y. Li. Enabling Backoff for SCM Wake-Up Radio: Protocol and Modeling. *IEEE Communications Letters*, 21(5):1031–1034, May 2017.

[9] P. Huang, L. Xiao, S. Soltani, M. W. Mutka, and N. Xi. The Evolution of MAC Protocols in Wireless Sensor Networks: A Survey. *IEEE Communications Surveys Tutorials*, 15(1):101–120, 2013.

[10] M. Magno, V. Jelicic, B. Srbinovski, V. Bilas, E. Popovici, and L. Benini. Design, implementation, and performance evaluation of a flexible low-latency nanowatt wake-up radio receiver. *IEEE Trans. Ind. Informat.*, 12(2):633–644, April 2016.

[11] R. Piyare, A. L. Murphy, C. Kiraly, P. Tosato, and D. Brunelli. Ultra low power wake-up radios: A hardware and networking survey. *IEEE Communications Surveys Tutorials*, 19(4):2117–2157, Fourthquarter 2017.

[12] T. Polonelli, M. Magno, and L. Benini. Poster Abstract: An Ultra-Low Power Wake up Radio with Addressing and Retransmission Capabilities for Advanced Energy Efficient MAC Protocols. In *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 1–2, Apr. 2016.

[13] S. L. Sampayo, J. Montavont, and T. Noel. LoBaPS: load balancing parent selection for RPL using Wake-Up radios. In *2019 IEEE Symposium on Computers and Communications (ISCC) (IEEE ISCC 2019)*, Barcelona, Spain, June 2019.

[14] S. L. Sampayo, J. Montavont, F. Prégaldiny, and T. Noël. Is Wake-Up Radio the Ultimate Solution to the Latency-Energy Tradeoff in Multi-hop Wireless Sensor Networks? In *2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 1–8, Oct. 2018.