2 WrenLoRa

"The wren is small and rather inconspicuous. But it lives life at a fast, relentless pace and it sings this way too - it trembles as it puts everything into its song, which lasts about 5 seconds and usually ends in a trill. They are either the first or one of the first birds to start singing at dawn and once they start their song is so loud that it drowns out everything else."

Figure 2.1 - WrenLoRa packet generator during preliminary tests in Campiglia M.ma (LI)

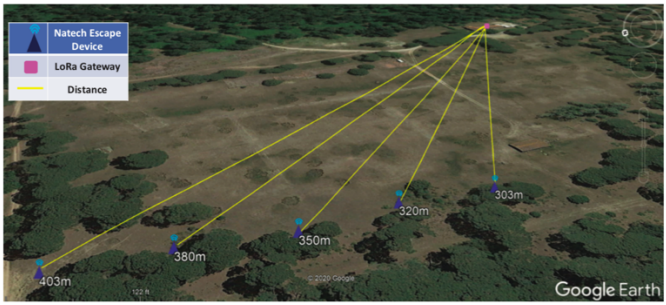
This chapter, to better explain design criteria and requirements of the packet generator, firstly introduces the original testbed and the experimental setup used to collect the data. In the subsequent section the hardware and software implementation is described. Finally, a kind of user manual for the packet generator concludes the chapter.

2.1 the experimental setup

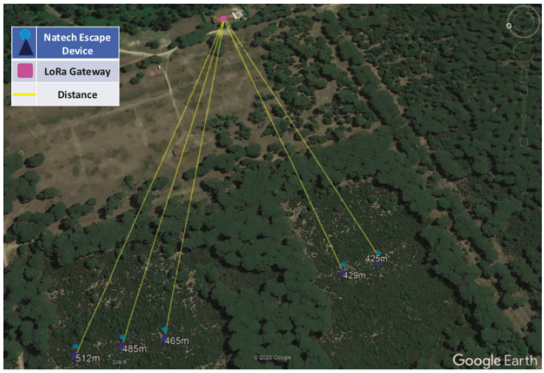
2.1.1 the original testbed

In [11] the testbed for the LoRaWAN infrastructure hosted in San Rossore natural park[2] (Pisa – Tuscany) is presented in detail.
A commercial gateway working in the EU868 MHz band is placed in an elevated position (antenna ~20 m above ground), using an existing building, and is equipped with satellite backhauling.
The Natech Escape devices [12] (i.e. the LoRaWAN end-nodes) are deployed in three different sites within the park, each one characterized by a different path to the gateway in terms of distance and obstructions.
Fig. 2.2 shows the areas of vegetation crossed in the three cases.

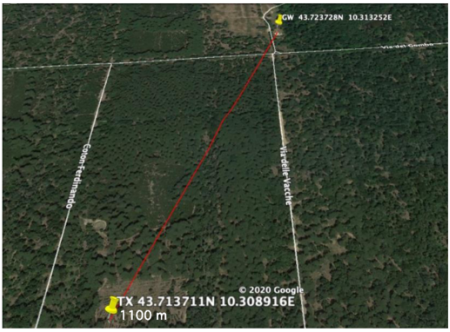
[2] https://www.parcosanrossore.org/



(a)



(b)



(c)

Figure 2.2 - the three sites where end-nodes are deployed: (a) site 1 LoS path ~350 m; (b) site 2 Light-dense vegetation path ~450 m; (c) site 3 Very-dense vegetation path ~1100 m

Statistics on the Packet Delivery Rate (PDR) obtained from each site, presented in [11], are summarized in *Table 2.1*.
The ratio between the number of successfully delivered and transmitted packets, i.e. in percent the PDR, offers a better understanding of the reliability of the connection than simple PHY level indications such as RSSI [*ibid.*].
The data set is obtained using the PHY layer settings proposed by LoRaWAN in the EU868 frequency band: CR 4/5, BW 125 kHz, SF adaptively chosen in the 7-12 range.
The power output of the end-nodes was 16,15 dBm EIRP with an antenna height[3] of 3 m.

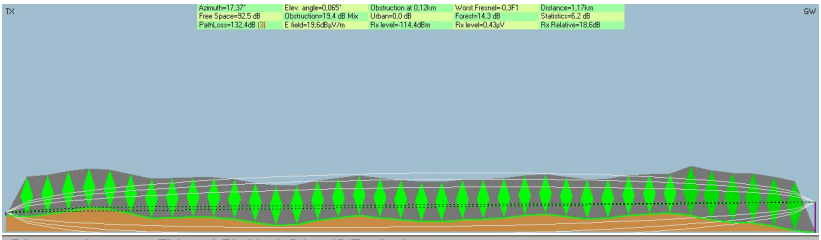
	Site 1	Site 2	Site 3
PDR	96%	73%	4%
min distance from GW	303 m	425 m	1170 m
vegetation	LoS	light-dense	very-dense

Table 2.1 - PDRs for the three sites. Data obtained transmitting 15200 packets from each site.

Despite the relatively short distance, it is clear that it is impossible to achieve reliable communication at Site 3, even using the highest SFs.
A simulation of this link of 1170 m, obtained with the Longley-Rice propagation model[4], is presented in Fig. 2.3.

[3] [11] incorrectly reports the heights of nodes and gateway

[4] ITM coverage prediction model (Irregular Terrain Model) by the US Institute for Telecommunications Science, aka NTIA-ITS Longley-Rice model, in conjunction with the 3-arc-second (~90 m resolution) SRTM (Shuttle Radar Topography Mission) geographical data in Radio Mobile software



Distance between TX and GW is 1,2 km (0,7 miles)
True North Azimuth = 17,37°, Magnetic North Azimuth = 14,59°, Elevation angle = 0,0647°
Terrain elevation variation is 12,4 m
Propagation mode is diffraction, single obstruction, 0,3F1 at 0,1km
Average frequency is 868,063 MHz
Free Space = 92,5 dB, Obstruction = 19,4 dB Mix, Urban = 0,0 dB, Forest = 14,3 dB, Statistics = 6,2 dB
Total propagation loss is 132,4 dB
System gain from TX to GW is 151,0 dB
System gain from GW to TX is 151,0 dB
Worst reception is 18,6 dB over the required signal to meet
50,000% of time, 70,000% of situations
Warning 3

Figure 2.3 – simulation details and profile of the link between a node in site 3 and the gateway: note terrain and vegetation

The simulation shows how vegetation and terrain heavily impact Fresnel's zones, but estimates a path loss of 132,4 dB.
Such a path loss would lead to an acceptable link budget and a much higher reliability than observed, but on the other hand, unless the model parameters are properly tuned through a sample set of survey measurements, deviations from the estimated value of up to 15 dB are possible [38]. Assuming the worst case scenario, this takes the system to the lower sensitivity threshold (*cf. Table 1.1*) and confirms the experienced behaviour where most packets are lost.

Fig. 2.4 represents the typical vegetation that obstructs the link in site 3.

Reported PDRs, simulations and site surveys make it clear that this environment exploits the limits of LoRa modulation, even within a kilometre radius.

Figure 2.4 - WrenLoRa in the vegetation of San Rossore park: notice the taller pine trees and the lower but denser holm oaks

2.1.2 the experiment

The analysis of the original testbed leads to the conclusion that the type of vegetation encountered represents a major obstacle in reliable communication.

Minding the original IoT application for smart agriculture, the current proposal aims to understand if it is possible to push the limits of LoRa technology and obtain, through the fine tuning of PHY layer settings, at least a kilometre coverage even in such a

scenario: it is common in fact, especially in central Italy, to have cultivated fields interspersed with wooded areas.
To do this, the experimental setup investigates the modulation behaviour in a point-to-point link involving exclusively the PHY layer. This is possible through the use of a pair of ad-hoc devices that simplify measurements by not implementing the upper layers of LoRaWAN and thus eliminating restrictive conditions such as TTN's fair use policies, the reduced sub-set of available parameters and being limited to the use of the 868 MHz band only.

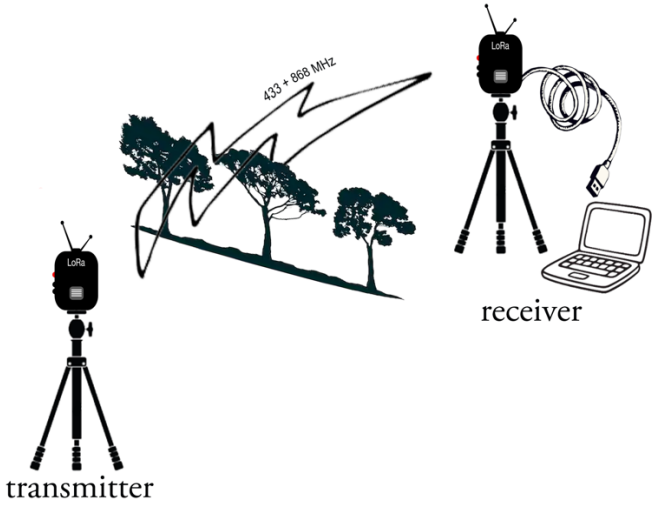


Figure 2.5 – a sketch of the experimental setup

Fig. 2.5 depicts the idea: a suitable path for a p2p link in the vegetation is chosen and the two devices are positioned at its ends. One assumes the role of packet generator, the other acts as receiver and passes the decoded packets to a USB connected computer that gathers them for statistical analysis.

In San Rossore park two types of path were used: the position of the receiver was static and the same in both cases; the transmitter was nomadic and the distance from receiver was increased reaching pre-established positions, ranging from 400 m to 2000 m. One set of transmitter positions was inside the forest, the other at the edge, but nonetheless the visibility of the RX was obstructed by the vegetation, in a situation similar to site 3 of the original testbed, which is about 1 km further west.
The map in Fig. 2.6 shows the position of the receiver and the two sets of positions (distinguished by color) where transmissions were made.

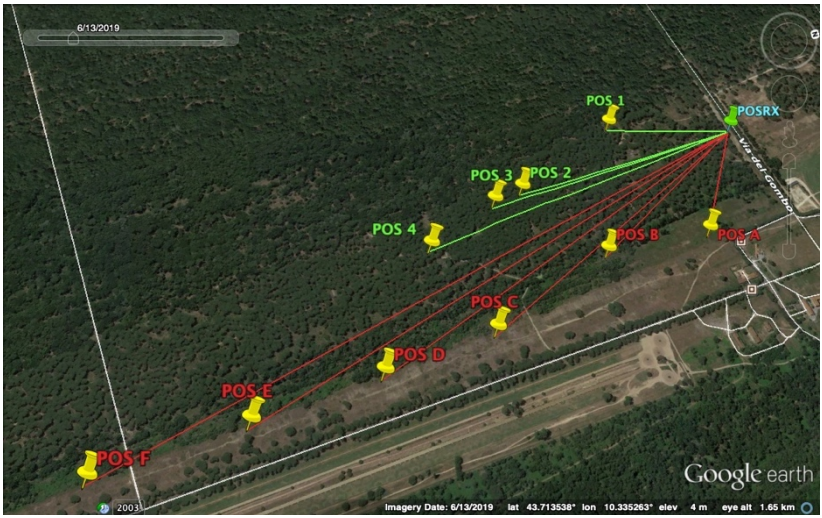


Figure 2.6 – in green and red the two sets of experimented paths

Fig. 2.4 presents an example of position in the green subset, where the view of the sky is almost obscured by the pine tree canopies, Fig. 2.7 shows instead a position of the red set where vegetation

ends at about 20 m from the transmitter, reproducing the installation in protection of a cultivated field.

Figure 2.7 - WrenLoRa in a position of the red subset

It should be noted that the paths of the red set cross a strip of vegetation about 100 m lower in height than the pine trees but perhaps denser (*cf.* also *Fig. 2.8, 2.9*).

Figure 2.8 - the barrier of vegetation in POS E (1700 m from RX)

Figure 2.9 - POS F (2050 m from RX)

2.2 design and implementation

2.2.1 key requirements

The first requirement emerging from the description of the experimental setup is a transportable packet generator capable of stand-alone operation for easy field deployment. Portability is achieved by providing WrenLoRa with a lithium battery and the necessary circuitry for power supply. A display and a set of buttons implement a simple user interface to control settings and operations without a laptop connected, except when the device acts as a receiver.

The UI shall provide access to the selection of layer 1 key parameters (SF, BW, CR, frequency…) and the software has to implement routines to generate bursts of LoRa frames, with the ability to choose, in compliance with Duty Cycle legal limits, how many packets to send and with which interval.
Another desirable feature is the possibility to connect an existing LoRaWAN infrastructure to test the coverage of a gateway.
This is made possible by implementing in the packet generator the minimum set of higher-level functions required for network authentication and packet sending to applications.
Finally WrenLoRa must also be able to work in the EU433 MHz band and simply switch between the available bands for easy comparison.

2.2.2 the hardware realization

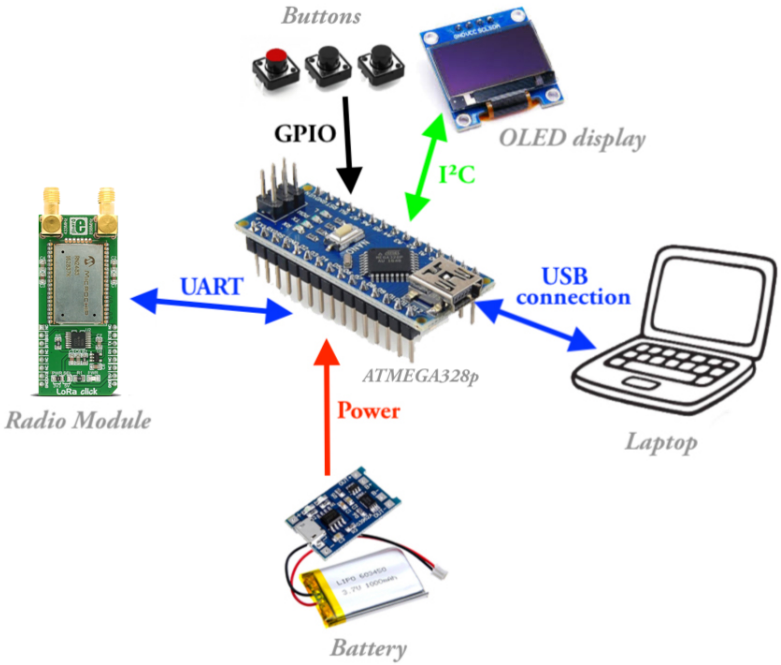


Figure 2.10 - a representation of WrenLoRa components and their interconnections

As shown in Fig. 2.8, the packet generator is built around an ATMEGA328p: this 8-bit AVR RISC-based microcontroller by Microchip combines 32KB flash memory, 1024B EEPROM, 2KB SRAM, 23 general purpose I/O lines and serial interfaces (USART, I²C, SPI bus).
To easily develop the prototypes an Arduino Nano board was chosen, which exposes several interfaces and ports of the MCU and provides a USB connection to program it and communicate with

the USART. For one of the prototypes an Arduino Uno has been used, which differs from the Nano exclusively for the form factor of the board.
Initially two radio modules were used, one for each band, connected to the SPI bus: the RFM95W by HopeRF [39] for the EU868 band and the Ai-Thinker Ra-02 for the 433 MHz [40]. The first uses a Semtech sx1276 chip (capable of covering the range 137-1020 MHz), the latter an sx1278 (137-525 MHz).
With these devices the MCU has to directly manage the transceiver, using low-level libraries to configure the chip registers. The first goal to test a p2p link with these modules is easily achievable, but to implement on top of these libraries also the functions necessary to connect a LoRaWAN network takes a lot of memory in terms of RAM and program space, going beyond the possibilities of the ATMEGA328p
Therefore, in order to make WrenLoRa also able to test a real-world LoRa infrastructure, it was necessary to use a full-stack modem such as the Microchip RN2483 [6].
The Microchip solution embeds an sx1276 transceiver but also a MCU that makes available through simple AT commands the setting of PHY layer parameters and the ABP/OTAA [22] registration methods necessary to join LoRaWAN infrastructures. In this way WrenLoRa's ATMEGA328p is only responsible for generating packets and handling the user interface via graphic display or USB connection to a terminal.
The RN2483 is connected to the main microprocessor via a secondary UART on GPIO pins. The primary WrenLoRa serial port is in fact dedicated to the USB connection to the outer world. Fig. 2.9 presents the block diagram of the radio module.

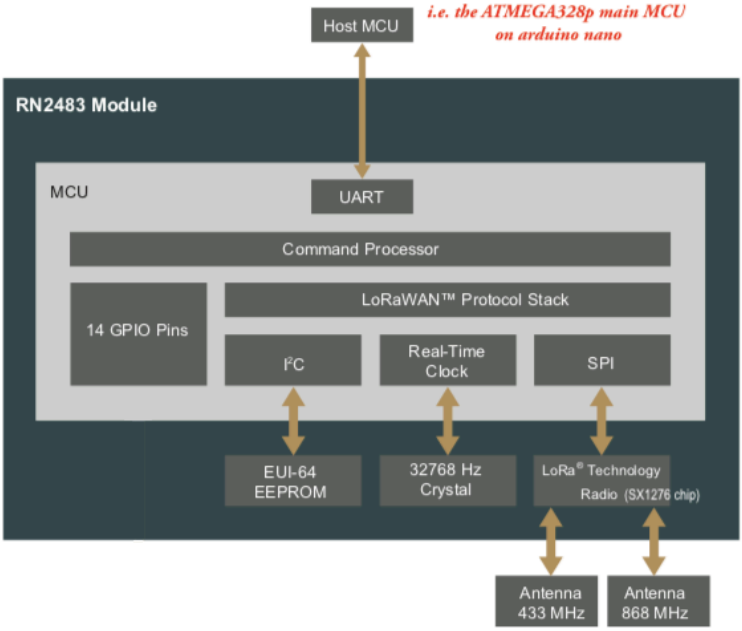


Figure 2.11 - RN2483 block diagram

A minor downside of this module is the use of a 3,3 V logic, thus a level shifter is required in the 5V communication with the host MCU: the used break-out board (Fig. 2.10) integrates this voltage-level translator [41].
Arduino boards working directly at 3,3 V are on the market, but they lack USB connectivity, so the prototypes, given also the boards that were at hand, use the 5 V version.

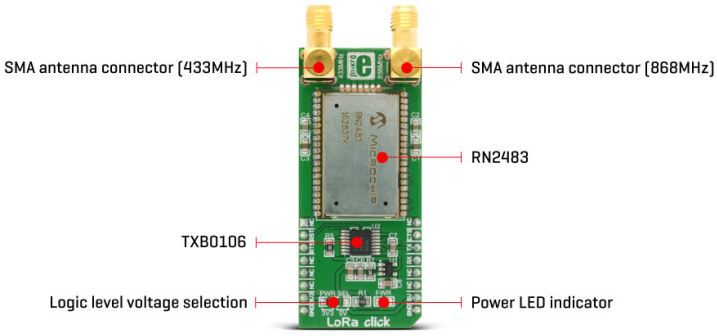


Figure 2.12 – the used RN2483 breakout board. Note the on-board bidirectional voltage-level translator TXB0106 necessary to convert the 3,3 V logic of the radio module to the 5 V logic of the ATMEGA328p

Prototyping is nowadays made easier by the wide variety of ready-to-use modules developed for the Arduino and DIY community in general. WrenLoRa uses these commercial off-the-shelf modules for the OLED display graphic interface, the step-up voltage converter and the lithium-ion cell charger, reducing the assembly only to a bunch of wires between PCBs and speeding up the development process.

Figures 2.11 and 2.12 show two of the prototypes that were made.

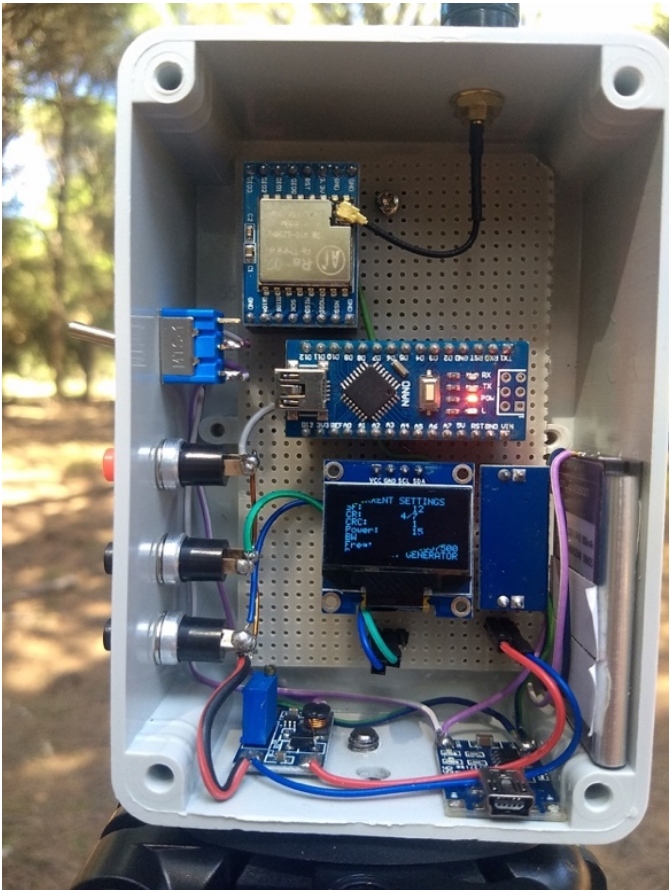


Figure 2.13 - an early implementation of WrenLoRa with SPI radio modules. This device fulfills the role of receiver or transmitter in a p2p link but can't authenticate to a LoRaWAN infrastructure. Note the circuits for Li-ion cell management and user interface. Black stripes on display are due to the refresh rate.

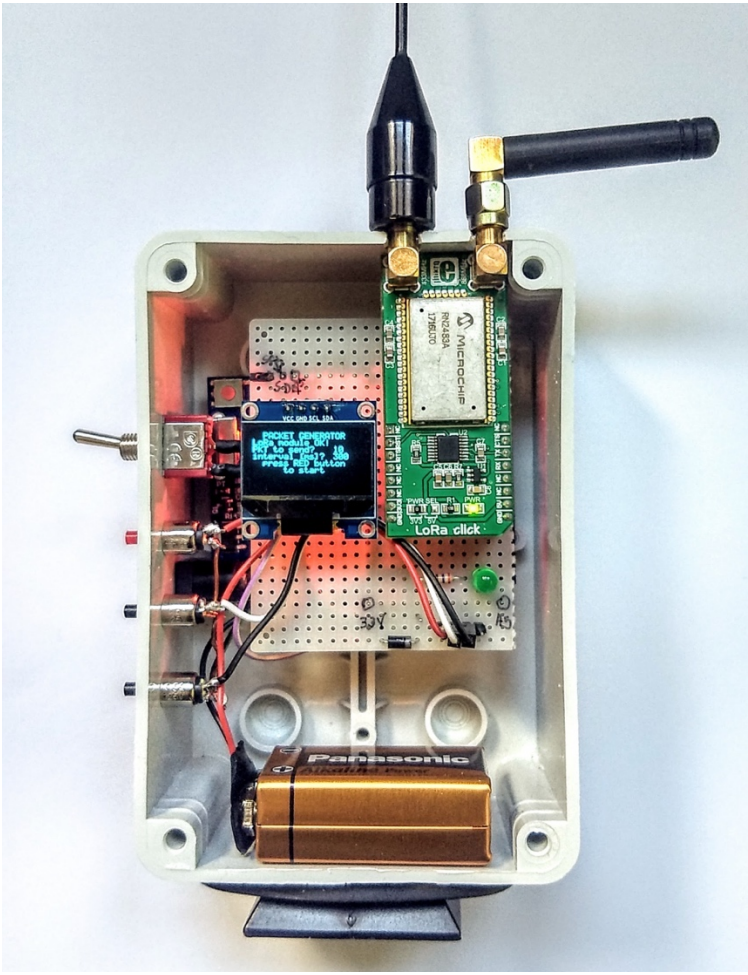


Figure 2.14 - The last prototype capable of ABP authentication to LoRaWAN thanks to the Microchip RN2483 module. In this implementation a 9 volt alkaline battery was used for simplicity as power supply. Moreover an Arduino Uno is used instead of a Nano: same characteristics but a different form factor.

2.2.3 the software

The simple Arduino IDE (Integrated Development Environment) was used to write the 812 lines of WrenLoRa code.

The software is made public through a repository[5] on GitHub under an MIT license. Schematics are also available for practical realization.

Due to the strict memory constraints imposed by the MCU used, the code was written trying to achieve maximum efficiency, making extensive use of arrays and pointers to streamline user interface routines.

Opensource libraries made possible the communication with the I²C connected OLED display [42] and the radio modules [43] [44].

A version of WrenLoRa software used in the first prototype with the two distinct SPI radio modules (cheaper) is also proposed in the repository: stripped from the GUI and with only a serial user interface, through the use of IBM LMIC (LoRaWAN-MAC-in-C) library [45] it is possible also to implement ABP for LoRaWAN authentication, requiring only 24920 bytes (77%) of program storage space and 1203 bytes (58%) of dynamic memory.

The final implementation using the RN2483, complete of GUI, serial user interface for configuration and receiver mode, ABP beaconing and packet generator routines, takes up instead 26048 bytes (80%) of program storage space, 1361 bytes (66%) of dynamic memory.

[5] www.github.com/iw5ejm/wrenlora

2.3 User Manual

The user has three buttons and a 0.91" dot matrix OLED display to interact with WrenLoRa. Alternatively, the packet generator accepts commands and communicates via the serial port and a terminal program, like PuTTY, Screen, the Arduino Serial monitor, etc.

WrenLoRa can perform three operating modes, or roles: *packet generator*, *receiver* or *ABP beacon.*

The first two are used in the p2p link to test the PHY layer (*cf. sec. 2.1.2*), the third one emulates a LoRaWAN node, able to authenticate itself to the network infrastructure and automatically send messages at a preset time interval.

Settings (PHY layer parameters, role, etc.) are stored in the ATMEGA328p non-volatile flash memory and reloaded at power-up.

At power on the user can enter the configuration menu within a timeout deadline, otherwise WrenLoRa takes the last used role and settings.

A summary of the operating logic of the device is presented with the simplified state machine diagram of Fig. 2.13.

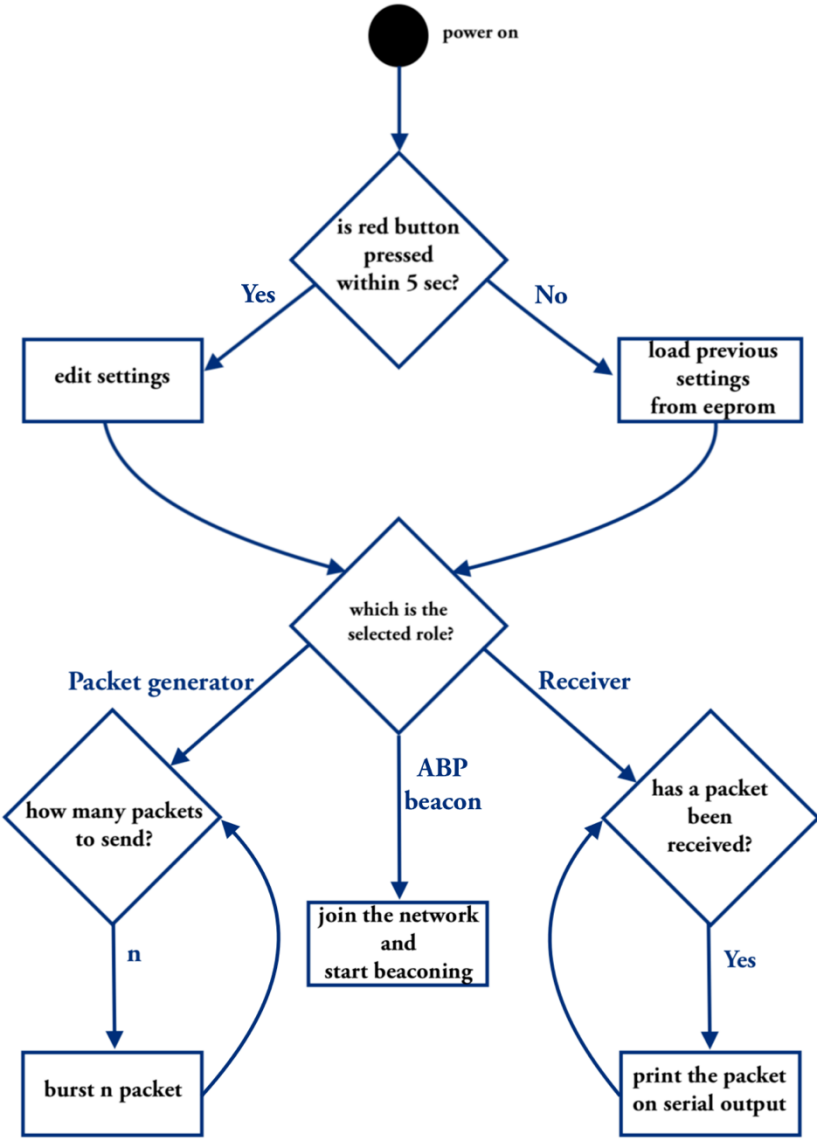
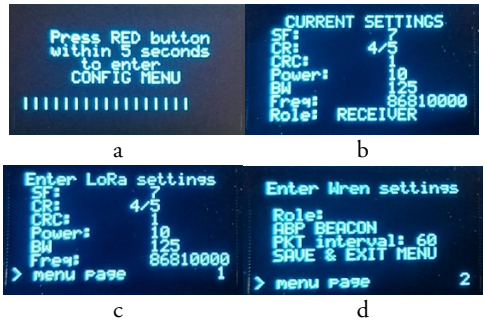


Figure 2.15 – state machine diagram of WrenLoRa

2.3.1 the graphical user interface

Figure 2.16 - WrenLoRa

Fig. 2.14 highlights the controls available to the user.
The function of buttons varies with the current menu, but usually, as intuition suggests, the + and – buttons respectively increase or decrease a value; the *red* button chooses the value to modify and confirms action requests. The switch controls the power supply.
When switched on, the message in Fig. 2.15a is displayed for five seconds with a progress bar.
Once the timeout has expired, after displaying the previously saved settings (Fig. 2.15b), WrenLoRa assumes the last used role.



a b

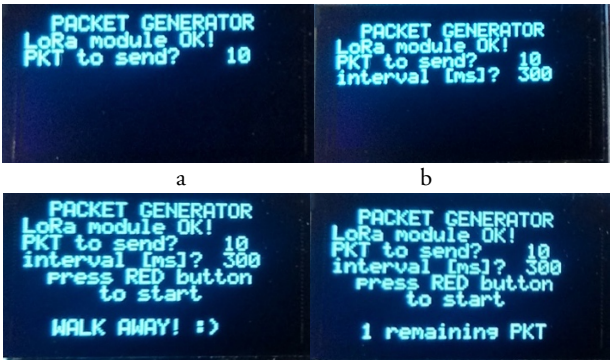
c d

Figure 2.17 – display screenshots

Pressing the *red* button within the 5 seconds accesses the configuration menu. It consists of two pages where it is possible to adjust all PHY layer settings and the role of WrenLoRa (Fig. 2.15c and d).

In this menu when the red button is pressed the cursor ">" cycles between parameters. Pressing the + or - buttons edits the parameter selected by the cursor.
To change the menu page, simply move the cursor to the last line and press the + or - button indifferently, like any other parameter.
To exit the menu and save the settings in the non-volatile flash

memory, select the "SAVE & EXIT MENU" line with the cursor (Fig. 2.15d) and press the + or - button.
If the **"packet generator"** role is selected, WrenLoRa enters the packet bursting routine.



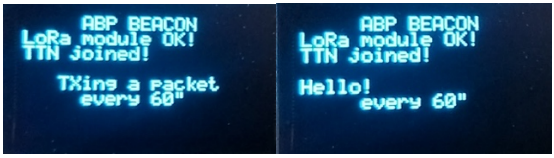
a b c d

Figure 2.18 – Packet generator mode

If the radio module has been successfully initialized the message "Lora module OK!" is displayed and the user is prompted to enter the number of packets to be generated in the burst (Fig. 2.16a).
Pressing + or – buttons modify the number of packets, pushing the *red* button leads to screen in Fig. 2.16b where WrenLoRa asks for the time interval between packets in milliseconds. Again pressing the + or - button changes the value accordingly. Confirmed the interval with the *red* button WrenLoRa is ready to start the burst.
By pressing the red button again, the packet generator gives the user 5 seconds to step away from the device and then the burst starts. In the last line there will be a countdown of the remaining packets to be transmitted (Fig. 2.16d).
During the transmission of each packet the green led lights up.

Just before starting the burst, a special packet with a summary of the selected PHY layer settings is transmitted. The others have a fixed payload length of 9 bytes. The sent message is "hello xxx", where xxx is a counter, needed to calculate the PDR, that identifies the packet.

If the **"ABP beacon"** role is selected in the configuration menu the device enters the beaconing routine to test the coverage of a LoRaWAN gateway. When the LoRaWAN TheThingsNetwork infrastructure is reachable and successfully authenticates the device, the message "TTN joined!" is displayed (Fig. 2.17a). WrenLoRa then starts sending messages at the chosen interval in the configuration menu (Fig. 2.15d). To enable stand-alone operation the procedure does not require any action by the user after power-on if this role is selected. The keys required for authentication are stored in the flash of the radio module via the serial menu.



a b

Figure 2.19 – ABP beacon mode

This mode is compatible with the TTN mapper to draw heatmaps for the coverage of LoRaWAN gateways.

When the role "**Receiver"** is selected in the configuration menu, WrenLoRa displays the message in Fig. 2.18 and it is necessary to connect the device to a computer running a terminal application.

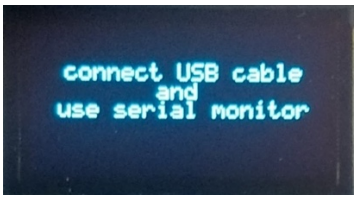


Figure 2.20 - Receiver mode

2.3.2 the serial user interface

All settings of WrenLoRa can also be changed via the serial port. These are the parameters of the serial connection: *9600 bps, 8 data bits, no parity, one stop bit*. There will be no echo of the user input. This connection is required in receiver mode to gather the decoded packets.
When the device is switched on and connected to a serial terminal, it prompts the user, similarly to the GUI, to enter the configuration menu within a timeout (Fig. 2.19).
Once the timeout has expired, the current settings are displayed and the last role used is assumed.

```
WrenLoRa field tester 0.9b
send "e" within 5 seconds to edit settings
.....................................................
= Current settings: =
SF: 7
CR: 4/5
CRC: 1
Power: 14
BW [kHz]: 125000
Freq [kHz]: 868100000
Role: RECEIVER

WrenLoRa Receiver

868 MHz RFM95w selected
LoRa module OK!

entering receiving loop
```

Figure 2.21 – messages printed through serial connection during power-up

At any time it is possible to enter commands to change the settings of layer 1, the role and other parameters of WrenLoRa. Fig. 2.20 shows a list of available commands, obtained by sending the "h" command, as help.

```
= Aviable commands: =
h print this menù
H sHow current settings
S save settings and quit menu
s change SF
c change CR
C change CRC
p change Power
b change Bandwidth
f change Freq
t change delay between ABP beacons
n change nr of PKTs to send
i change delay between PKTs
r change role
```

Figure 2.22 - available serial commands

Fig. 2.21 provides an example of changing the SF and the operating mode. To save the parameters and initialize the radio module with the new settings, send the "S" command.

```
SF: 8
SF: 9
Role: ABP BEACON
Role: PKT GENERATOR

= Current settings: =
SF: 9
CR: 4/5
CRC: 1
Power: 14
BW [kHz]: 125000
Freq [kHz]: 868100000
delay between ABP beacons: 300
PKTs to send: : 10
delay between PKTs: 1500
Role: PKT GENERATOR
after changes REMEMBER to send 'S' to init radio module with new parameters and save them to eeprom
```

Figure 2.23 - example of parameters modification

When the **"receiver"** role is selected the received packets are automatically printed on the serial connection.
The passed packets have successfully completed the CRC check and have not been corrupted.
In the experimental campaign the Microcom software on Linux was used as a terminal application, able to log the serial output to a file. The received packet dataset was built by analysing this file and importing the packets into a CSV spreadsheet to determine the PDR, given the number of packets transmitted.

Via the serial port it is also possible with the "o" command to store the session security keys needed to authenticate LoRaWAN networks in the radio module.
The device identifier is printed before the request of the keys to allow their generation through the backend console of TheThingsNetwork.