COOPERATION AND DEVELOPMENT PROJECT

COLLABORATION ULB - ISIG

Design and implementation of a telecommunication system for rapid and flexible deployment of mobile teams in the field in emergency or cooperation situations (NGO)

Teams members:
Cécile **CASTIAUX**
Nicolas **WALLEMACQ**
Margaux **MANNAERTS**
Nathan **GARTNER**

Coordinators:
Orianne **BASTIN**
Maxime **PÉTRÉ**
Supervisor:
Antoine **NONCLERCQ**
Signature :

Academic year 2020-2021

Abstract

"Design and implementation of a telecommunication system for rapid and flexible deployment of mobile teams in the field in emergency or cooperation situations (NGO)" by Cécile Castiaux, Nicolas Wallemacq, Margaux Mannaerts, Nathan Gartner, Université Libre de Bruxelles, 2020-2021.

In order to respond to the request of Médecins Sans Frontières (MSF) to design and implement a battery monitoring system, a cooperation and development project was undertaken. This project brought together Venn Telecom, MSF, Belgian engineering students from the Ecole Polytechnique de Bruxelles, who take care of the battery state of charge evaluation system, and the computer science students from ISIG (Institut Supérieur d'Informatique et de Gestion de Goma), who oversee designing software that will be able to establish bi-directional communication with the Belgian students' prototype. This software will make it possible to process the information of this state of charge as well as to send instructions and provide the possibility to control the module remotely according to the user's preferences. Indeed, MSF uses communication cases when they must intervene in the field in hostile environments such as those found around Goma in the Democratic Republic of Congo. These cases use Wi-Fi routers with antennas to connect to the network, and in most cases these elements are powered by Zendure Supertank power banks. Since it was not possible to remotely access the state of charge and the remaining autonomy of the power bank and therefore of the global telecommunications system, the idea came up to make it a collaborative project which will provide an experience of international collaboration between students from different countries. This report focuses on the Belgian student prototype, which is almost fully functional. For the moment, a measurement of the incoming and outgoing currents of the Zendure is made via a small algorithm which finally gives the percentage of remaining capacity of the battery. The exactitude of this algorithm has not been tested yet. A communication module is used to send the GPS position and the state of charge of the monitored battery to the website implemented by ISIG students.

Key words : Collaboration Project, Battery, Monitoring, State of charge, Communication.

Word count :287.

Acknowledgements

We would to give a special thank you to Michel Osée who was always available and very helpful in the development of our project.
Thank you to Geoffrey Vanbienne for his help in designing the PCB and his always warm welcome in the lab, to Jean Landercy and Rudy Ercek for their time in helping us with the communication of our prototype.
Thanks to Pierre-Gilles Dehaye from Venn Telecom, he gave us crucial technical advice from an external view. His input always enabled us to see the project in a global way and to make us think about several facets that we had not thought of.
Thanks to Pierre Mathys for his experienced look at the project and for his help in choosing the components.

Contents

Introduction

New technologies play a crucial role in many fields, including the fields linked to medical emergencies and the development cooperation. A rapid deployment of a communication system is critical for teams working in remote regions and needing to be in contact with a central unit. In recent years, such communication technology became mainstream in developed countries but it is not yet the case in developing countries.

Present in developing countries, Médecins Sans Frontière (MSF), a humanitarian NGO, deeply needs a reliable communication system for telemedecine. One of its doctors working on the field may for instance need an expert's medical advice and thus needs an access to internet to be able to send its physical examination of a patient and receive the advice.

For that matter, MSF disposes of telecommunication cases containing a battery powered router, such as the one illustrated on Figure 1. The router itself relies on the 2G/3G/4G network and it provides an internet access via Wi-Fi to MSF's teams, allowing them to send and receive data via the internet.

Figure 1: MSF's communication case

Unfortunately, nothing is yet implemented to track those cases when they are powered off and to monitor their batteries: state of charge, time remaining, battery ageing. This implementation would greatly help MSF for the maintenance of the batteries and to locate the telecommunication cases when they are lost.

In this context, MSF asked the *Cellule de coopération au développement de l'École polytechnique de Bruxelles (Codepo)* if a group of engineering students could take up this challenge and create an autonomous device able to monitor and track its cases and whose information can be remotely accessed via a web portal. Helped by Venn Telecom, acting as the technical advisor, the CODEPO's students will take care of the hardware part of the project and will work together with students from the Institut Supérieur d'Informatique et de gestion de Goma (ISIG) who will take care of the web interface. The aim is that ULB's student go in July 2021 to Goma, in Congo, to combine the prototype to ISIG's student's software and confront it to the reality on the ground. This project is a unique occasion for the students to simulate the professionals - clients relation (with MSF acting as the clients and the students acting as the professionals) and to work together and coordinate the work with another team of professionals.

This report details the progress of the elaboration of this device by this year's CODEPO's team of students.

1 Requirements

Based on the instructions that were given and based on discussions with MSF, the following requirements for the device were noted. The device should:

1. Measure the battery's state of charge (SOC), remaining time of use and information about its ageing
2. Measure its GPS position (or equivalent)
3. Transmit the relevant information by USSD to a centralised platform on the web
4. Be able to modify its behaviour or its program via remote instructions
5. Consume as less energy as possible. A 3 months autonomy is targeted even if the monitored batteries are powered down
6. Be as small as possible but at least fit in the communication case
7. Be allowed in a plane and withstand a travel by plane

The following additional - but non-essential - features can be added if there is enough time and that it doesn't interfere with the previous requirements. The device could:

1. Measure the temperature and humidity of the surroundings
2. Emit sounds to communicate alerts to the users
3. Display information on a screen to the users

2 Case

This section will briefly explain the components of MSF's communication case. It has been modelled in 3D with SolidWorks software.

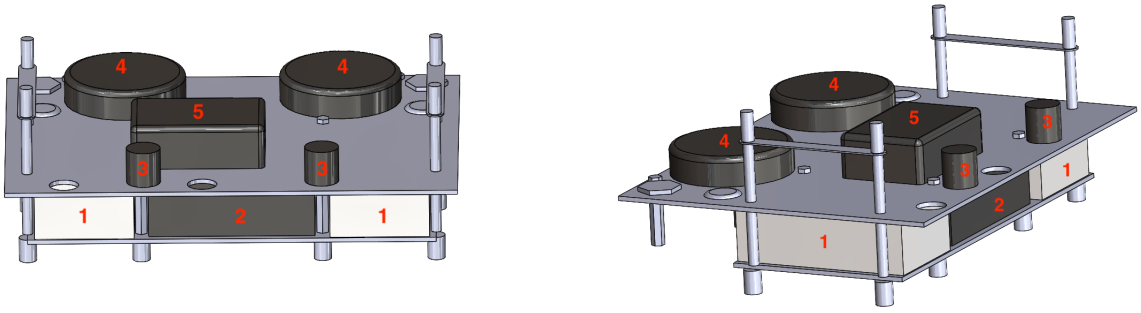

Figure 2: 3D model views of the components of the case

2.1 Battery

The two batteries included in the case are "Zendure Supertanks" (Zendure being the company that produces this battery). It is these batteries that will have to be monitored. They are marked with a number 1 on Figure 2. It's specifications are as follows:

- **Capacity:** 27 000mAh/99.9Wh (which is the limit allowed for cary on bagages in a plane)
- **Dimensions:** 119mm x 73mm x 42mm

- **Weight:** 500g
- **Input:** USB-C: 5-20V, up to 5A, 100W
- **Output:**
  - USB-C(1): 5-20V, up to 5A,100W
  - USB-C(2): 5-20V, up to 3A, 100W
  - USB-1: 5V/3A,9V/2A, 12V/1.2A, 18W
  - USB-2: 5V/3A, 15W
- **Product Life:** Over 500 full charge-discharge cycles
- **Operation Temperature:**
  - 0°C to 45°C (Charging)
  - -20°C to 65°C (Discharging)

2.2 Pepwave Max Transit Duo

This component is the case's router. It is marked with the number 2 in Figure 2. This is where the users of the case insert SIM cards of the country they are in. The router is connected to 2 wifi antennas and 2 GPS/Cellular antennas.

2.3 Wifi Antennas

These are marked with the number 3 on the Figure 2. It is thanks to these antennas that the users get wifi in remote areas. They are located on the upper side of the case.

2.4 GPS/Cellular Antennas

The router transforms 3G/4G connection in WiFi signal. The Cellular antennas (marked with the number 4) are the ones "capturing" 3G or 4G signal.

2.5 Transformer

The transformer is located at the upper face of the case. It has an outlet which can be plugged into the mains supply. It is also connected to the 2 Zendures in order for them to get charged whenever the outlet is plugged. (Number 5 on Figure 2).

3 Previous Work

This project is the follow-up of the 2019-2020's CODEPO's team's work. Due to the coronavirus crisis, they did not get the chance to exchange face to face with the ISIG students, but this didn't stop them from making a working prototype and talk to the other students online. This section explains the methods and results of last year's team's prototypes.

3.1 Measurement of the battery's state of charge

The tension of the Zendure battery stays constant during it's charge/discharge making it hard to deduct it's percentage from this value.

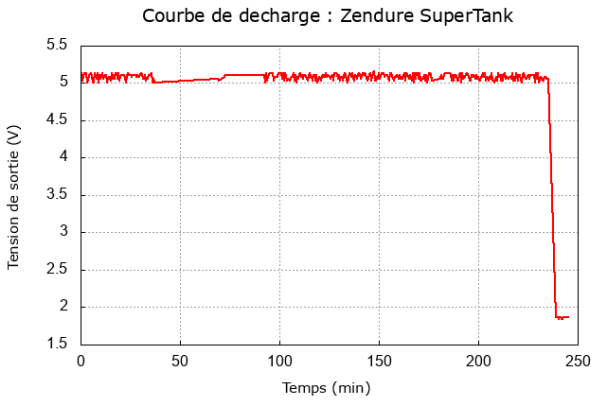

Figure 3: Discharge of the Zendure

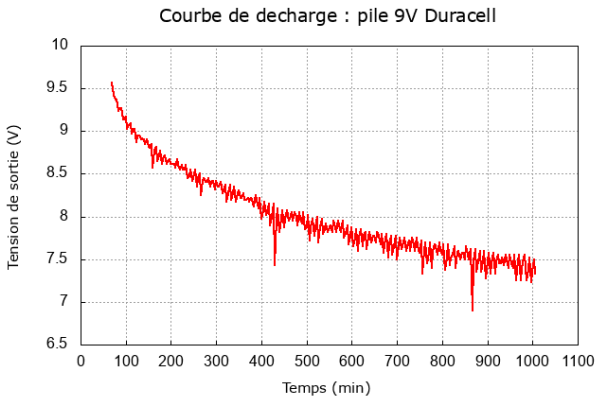

Figure 4: Discharge of a 9V battery

Their final approach was to do a binary balance on the battery. This consisted in placing volt and amperes captors at different locations (see Figure 5) in order to know if the battery is in use or not and in charge or not. With this information and the knowledge of the maximum capacity, it is possible to approximate the battery's percentage.

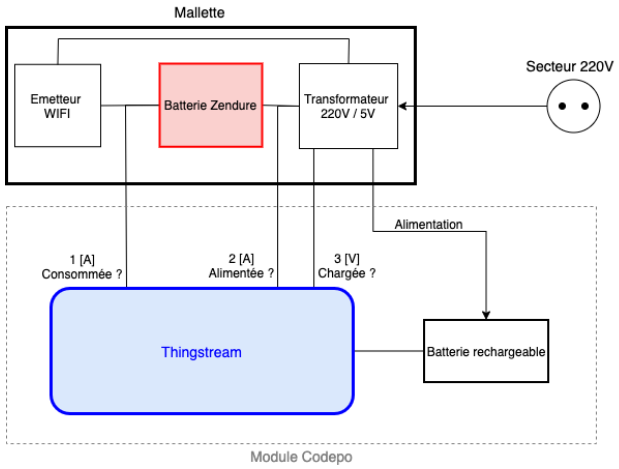

Figure 5: Functional Diagram of 19-20 CODEPO's prototype

3.2 Location and communication

Last year's team decided to choose USSD technology in order to establish communication between their prototype and a platform. It enables a simple bidirectional communication and is functional even in areas where there is less signal coverage. They decided to work with Thingstream, which is a company that allows the use of USSD technology thanks to micro-controllers and their server. CODEPO 2019-2020 opted for the "Starter kit" micro-controller which has 2 antennas (GPS and GSM).

3.3 Implemented Features

These are the different features that were implemented in last year's prototype.

- Battery's SOC with a 5% error
- Alert messages when the battery is 20% , 5% , = 100%
- A Very Low Power mode (VLP)
- Bidirectionnal communication
- Sending of information on the battery's state (charging, decharging, etc)
- Sending of the prototype's location
- Determination of the prototype's mode
- Battery's SOC calibration (with a LED that turns green when the battery reaches 100%)

This following table shows the consumption for different cycles of the prototype when it is powered with 12V. It is possible to get the position with GPS or GSM.

Mode	$P_{moy}[W]$	$W_{moy}[J]$	$Temps[s]$
Cycle without position	0,358	11,462	32
GPS cycle	0,394	15,354	39
Only GPS	0,298	1,491	5
GPS search	0,19232	/	/
GSM cycle	0,358	23,264	65
GSM only	0,374	7,478	20
Cycle GPS and GSM	0,406	27,222	67
Veille RUN	0,040	/	/
Veille VLPR	0,010	/	/

Table 1: Energy consumption for different cycles

Figure 6: Last year's prototype

3.4 Weak points

The main weak points of last year's prototype were:

- Its dependence on the Thingstream's starter kit which is voluminous, more expensive and has proven to be defective (destructive short-circuits due to a bad conception of the motherboard)
- Bad wiring and no PCB to miniaturise the wiring
- Its autonomy: 5 days versus the 3 months required
- Its power source: a non rechargeable 9V battery (as seen in the picture of the prototype)
- Once the monitoring device is powered-down, all the information about the monitored battery's SOC is lost and can't be recovered

4 Reflections based on last year's weak points

Based on last year's prototype's weak points explained in Section 3.4, we decided to take these different decisions:

- We won't use the Thingstream Starter Kit as our communication module: we'll stick to the USSD communication and to the Thingstream platform but we'll use a more compact and simpler model. Another reason to change the communication module is that the Thingstream Starter Kit was discontinued.
- We will change the device's power supply: instead of using a non-rechargeable 9V battery, we will use a 5V rechargeable power-bank with an improved capacity compared to the previous one.
- We will implement a coulomb counting method to calculate the battery percentage.
- We will implement a way to stock the data in a permanent memory in order to have it accessible even though the device might be powered off.
- We will design our device to be as much energy-efficient as possible. This will be done using energy-efficient components and switching on certain devices/pins only when they are necessary.

4.1 Implementation of this reflection

4.1.1 First semester's approach

The Figure 7 illustrates the approach undertaken during the first semester. The green area represents our monitoring device and includes thus the elements we will add inside the communication case.

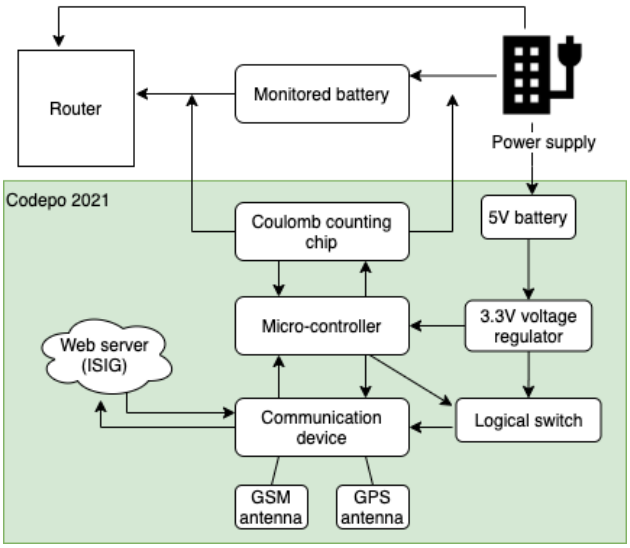

Figure 7: First semester approach: block diagram

In this approach, 2 devices run constantly: an industrial-quality coulomb-counting chip and the micro-controller. We can afford to let the micro-controller run constantly thanks to the low current consumption attainable in nowadays micro-controllers (see Section 7.1.4). At any time, the State of Charge (SOC) value can be accessed by the micro-controller via I^2C. The former acts as the conductor of the whole schema. At each time interval t (which can be remotely modified), it reads the SOC value on the coulomb counting chip and switches on the communication device. Once totally switched on, the former signals if it is ready to transmit data and/or if a message was received from the server [1].

The micro-controller asks the communication device for its GPS coordinates. Finally, the micro-controller condensates the SOC & GPS coordinates to a string variable that is transmitted to the communication device in order to be sent via USSD to a remote server. If the message is transmitted with success, the micro-controller switches off the communication device until it is needed later.

Our device is powered by a rechargeable power-bank. It provides a 5V input voltage (which needs to be reduced to 3.3V and regulated) needed for our communication device and micro-controller. If the communication case is powered by an external power supply, it automatically recharges the monitored battery and our monitoring device's battery.

Problem with this approach :

We followed this initial approach using the BQ34110 battery gauge chip from Texas Instrument for the first semester and the beginning of the second semester. This choice was motivated by the fact that the engineers at Texas Instrument responded favourably to our emails explaining the envisaged use of this component within the framework of our project, as well as by Mr. Pierre Mathys who, after examining the datasheets, advised us to use this chip while alerting us to the possible bad behaviour of the SOC estimation

[1] In this case, it transmits the message content to the micro-controller. A message from the server could be an instruction to change the functioning mode of the monitoring device but more on that in later Sections.

algorithm as well as to the unexpected reactions of the Zendure. In theory, this component would have made it possible to monitor a battery by reading a multitude of parameters, including voltage, current, SOC, SOH, remaining capacity, average time to empty, etc.

However, this solution has never given correct and relevant results. Indeed, after several days of chip calibration tests and battery charge/discharge cycles, the data collected on the BQSTUDIO computer interface was different each time and completely out of sync with the real behaviour of the battery, there was no correlation between the tests. We realised that we could not control the chip and that it was very difficult to understand. It was like trusting a black box that was supposed to calculate too many unnecessary variables by itself and send back the requested information without having any real control over it.

Following a discussion with Mr. Pierre-Gilles Dehaye from Venn Telecom, it was concluded that the chip was in fact not suitable for our application. It turns out that it is generally used for UPS (uninterruptible power supply) batteries that serve as backup power when a regular power source fails or voltage drops to an unacceptable level and which are therefore not required to charge/discharge frequently as it is the case with our prototype. The chip also had only one input for calculating the battery parameters, which meant that a second device with a second chip was needed to control the input and output of the Zendure, ensuring that the information from each chip was correctly gathered to deduce the actual state of charge. The Zendure being an intelligent battery, it can also be too reactive to a battery gauge, its electronics "fighting" against the IT device.

An alternative method was then chosen to calculate the SOC and is explained below.

4.1.2 Final approach

Our final approach is essentially the same than the first one, except the way we compute the SOC. In order to calculate what is coming in and going out of the battery, the idea is now to simply intercept the input and output of the battery by inserting a known resistor in series and calculating the voltage at the terminals of these resistors thanks to a micro-controller, the incoming/outgoing current is thus deduced. Then the SOC is calculated using a gauge system implemented in the micro-controller code representing the instantaneous capacity of the battery. Assuming the current is constant for a certain time interval, it is possible to increment/decrement the gauge as it goes along. Further details on how the measure of the SOC will be explained in Section 6.

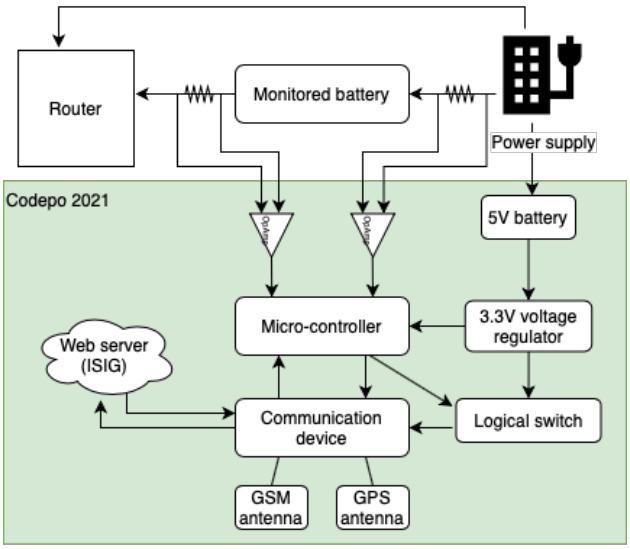

Figure 8: Final approach: block diagram

5 Communication

5.1 Communication between the device and the server

Our device needs to communicate with a remote server designed by Goma's students for MSF. The communication must be :

- bi-directional: the device must be able to send data to the web server but the web server must also be able to send instructions to the device (to change its functioning mode for example);
- reliable: our device won't always be turned on or have access to a reliable connectivity, so we need a way to ensure that the messages will be transmitted at an appropriated time;
- robust: the messages need to be well received;
- simple: the limited hardware and the autonomy required constrain us to find a simple and low-consumption solution for the transmission of the messages.

Such bi-directional communication can be achieved with Message Queuing Telemetry Transport (MQTT) which is a handy messaging protocol for Internet of Things (IoT) devices. This is the communication protocol we chose to implement in our device. This protocol is briefly explained in the next subsection.

5.1.1 MQTT

MQTT is a publish-and-subscribe protocol. Instead of communicating directly to a server, client devices publish and subscribe to topics handled by a broker[2]. Here is small practical example to illustrate how it works. As can be seen on Figure 9, a MQTT client (the temperature sensor) publishes a message (24°) to a topic. This message is not sent directly to the other clients subscribed to this same topic (the mobile device and the backend system) but it is instead stored in an MQTT broker. Only once a client establishes a connection and subscribes to the topic will he begin to receive the messages from that point onwards.

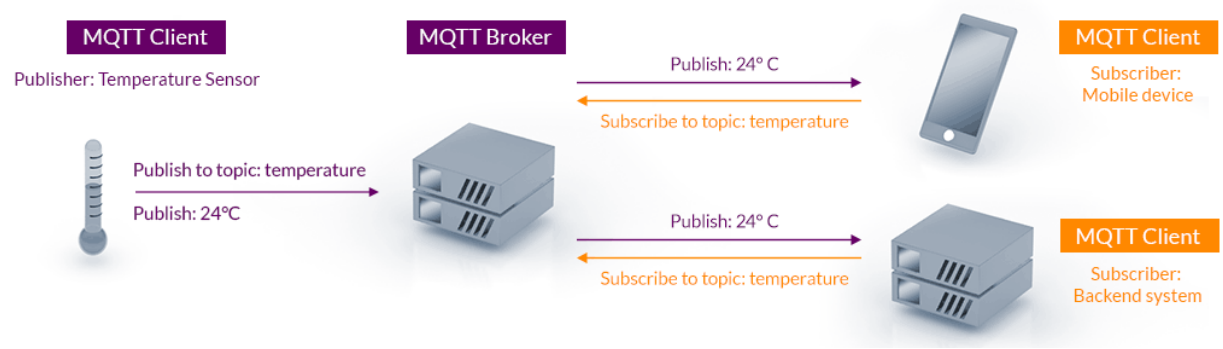

Figure 9: MQTT Publish / Subscribe Architecture[2]

The MQTT messaging protocol comes with great advantages that totally suit our project. Some of those advantages[2][7] are:

- it allows bi-directional communications;
- it is lightweight and efficient: it requires minimal resources and can be used on small micro-controllers;
- it is easily scalable: broadcasting a message to one device or to hundreds is the same as long as all the concerned devices are subscribed to the same topic. Similarly, hundreds of devices can post messages on the same topic;

- it provides reliable message delivery;
- it works with unreliable networks: MQTT can store messages at a broker until our device is ready to receive it;
- it is secure: encryption and authentication are easily implementable with MQTT.

To simplify our prototyping and deployment of our device, we will use the services of u-blox Thingstream.

5.1.2 Thingstream platform

It is cloud-based platform and administration interface. It will enable our device to get an easy access to IoT connectivity thanks to their *MQTT Anywhere* IoT communication service and get MQTT anywhere in the world (access to 600+ cellular networks across 190 countries). With this service, our device gains access to an enterprise-grade MQTT broker and a user-friendly representation of the messages exchanged between the platform and the devices[8]. It supports 2G, 3G, LTE, and LTE-M and can be used within a IoT device to send and receive MQTT-SN[2] messages using either USSD or UDP protocols as a transport mechanism. The transport mechanism used between our device and the MQTT broker is USSD. The topology of the network is presented on Figure 10.

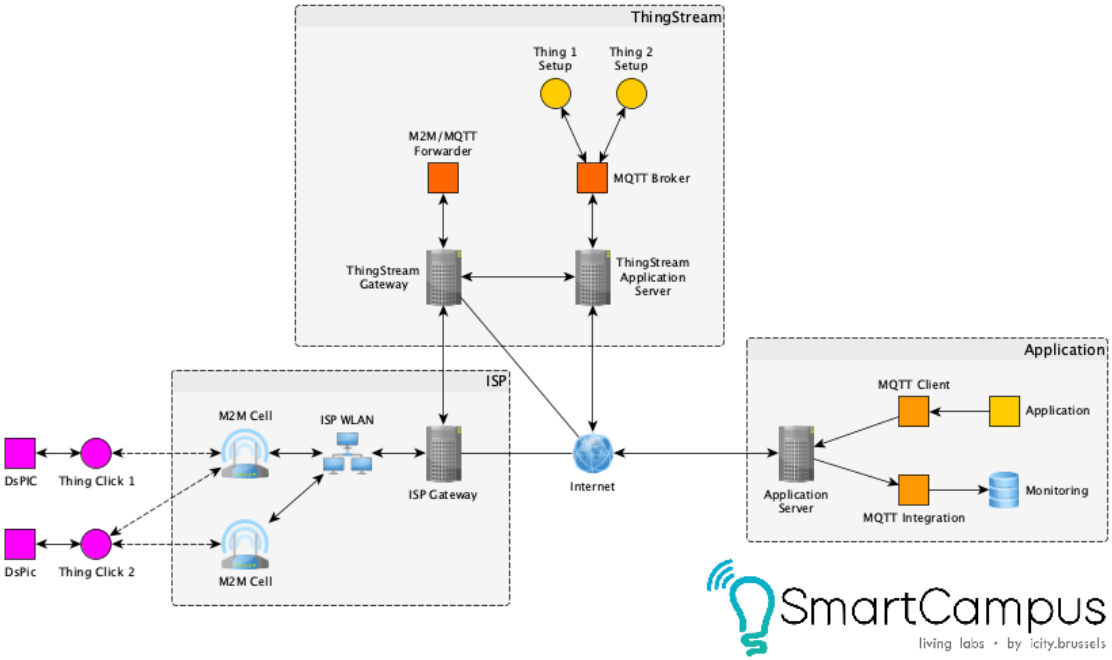

Figure 10: Topology of the network. Courtesy of LANDERCY Jean (SmartCAMPUS)

5.1.3 USSD protocol

USSD stands for Unstructured Supplementary Service Data. It is a real-time communication or instant messaging protocol, used to send messages across a GSM network between a mobile client and an application server. This technology is similar to SMS but it presents an interactive nature and is session-based. Its

[2]MQTT for Sensor Networks, an optimized version of MQTT designed specifically for efficient operation in large low-power IoT sensor networks

turnaround response time for interactive applications is shorter than for SMS and it requires less network coverage[1], which is something we can expect in the remote areas the device will be located.

5.1.4 Downlink

The downlink communication (server to device) is represented on Figure 11.

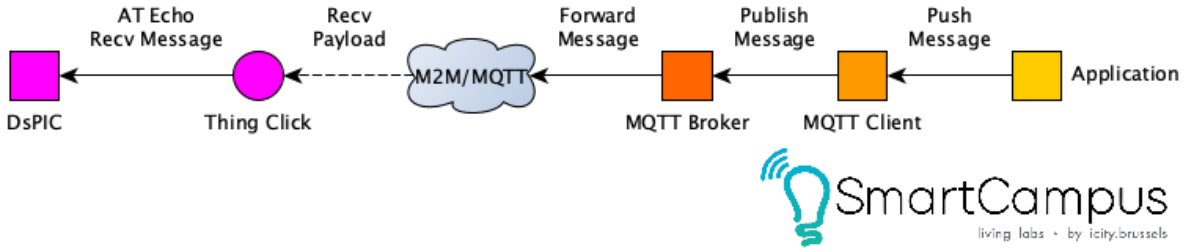

Figure 11: Downlink. Courtesy of LANDERCY Jean (SmartCAMPUS)

As mentionned in 5.1, a communication web-to-device is needed to instruct the device to change its (predefined) mode of operation or to change some parameters (for more details, see Chapter 8). An example of a downlink message could be:

```
"m=1,t=24,i=1,c=27000"
```

where:

- m = functioning mode
- t = hours between transmissions
- i = minutes of sleep between the SOC updates
- c = max capacity of the battery (if it needs to be reset to a certain value).

The message is posted by the application (MSF's website) on the same topic that the device will subscribe to. Such message is published with a Quality of Service of 2 (QoS=2) and instructed to be retained by the broker to ensure that the message is well received by the device once it is connects to the service and subscribes to the topic.

5.1.5 Uplink

The uplink communication (device to server) is represented on Figure 12.

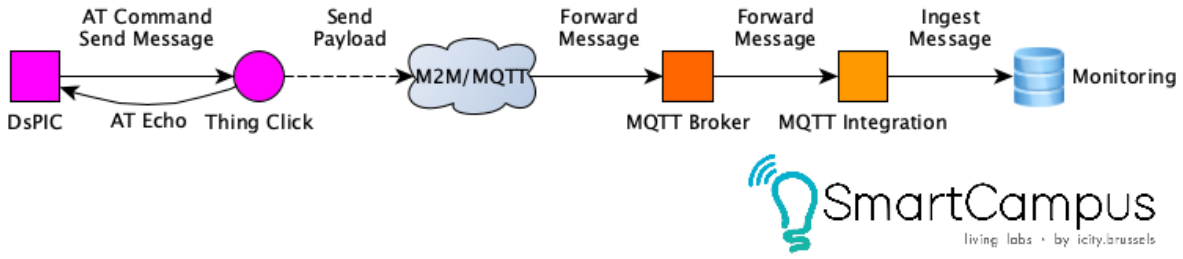

Figure 12: Uplink. Courtesy of LANDERCY Jean (SmartCAMPUS)

The device posts messages on the topic that the monitoring's application (MSF's website) is subscribed to. Such message contains, for example, the id of the telecommunication case, the state of charge, the GPS coordinates and the functioning mode the device is on. Such messages are published with a QoS=1. An example of a downlink message could be:

```
{"caseid": "12345678", "SOC": 78, "Latitude": -1.663402, "Longitude": 29.228020, "mode": 1}
```

where:

- caseid = unique identifier of MSF's case.
- SOC = state of charge of the monitored battery, expressed in %
- mode = functioning mode the device is on

This message follows the JSON format. This was specifically asked by GOMA's students in order to easily treat and extract the data for MSF's website.

5.2 UART Communication between the micro-controller and the Thingstream Click

UART interface is used by the micro-controller to communicate different commands to the communication device (cf. Section 7.1.1).

The universal asynchronous receiver/transmitter (UART) is an interface that can be found in embedded communication systems or computers. In the case of a computer, it represents the connection between the computer and RS-232, RS-422 and RS-485, serial bus interfaces which are norms standardising a serial data transmission protocol.This mode of transmission is characterised by a sequence of data elements on one channel as opposed to parallel transmission, which sends data simultaneously on different channels. The last mentioned mode is used by the computer, so it is necessary to transform the data to transmit it through a serial bus interface.

As we can see in the figure 13, UART transmit and receive data frame format is composed of the start bit, which is the beginning of the data transmission, followed by 5 to 8 data bits with or without parity, the stop bit and the idle state. When the data transmission starts, the logical signal "0" is sent. At the end of a data, the logical signal "1" is sent by the stop bit. The idle state is a logical "1" and is used for character synchronisation[9].

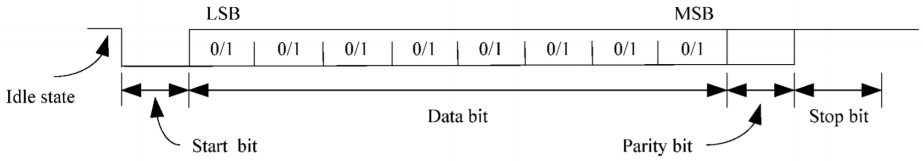

Figure 13: UART data frame format

6 Measurement of the battery's State of Charge (SOC)

Several methods exist to calculate the state of charge of a battery such as Direct Measurement, Open Circuit Voltage Method, Terminal Voltage Method, etc...

The problem with most of these methods is that they are based on the voltage loss at the output of the battery when it is active. In our case, the Zendure maintains an output voltage of 5V. The simplest method to calculate the state of charge of the power bank without taking into account the voltage supplied, is therefore to make a balance of the incoming and outgoing charges while knowing the maximum capacity of the battery. This is the coulomb counting method.

6.1 Coulomb counting method

The SOC is one of the most important parameters for batteries. In general, the SOC of a battery is defined as the ratio of its current capacity $Q(t)$ to the nominal capacity Q_n. The nominal capacity is given by the manufacturer and represents the maximum amount of charge that can be stored in the battery. This value can degrade over time. The SOC can be defined as follows [4]:

$$SOC(t) = \frac{Q(t)}{Q_n} \tag{1}$$

The Coulomb counting method measures the charging/discharging current of a battery and integrates it over time in order to estimate the SOC. This method also takes into account the previously estimated SOC values, $SOC(t-1)$, as it can be seen in the following expression [5] :

$$SOC(t) = SOC(t-1) + \int \frac{I(t)}{Q_n} dt \tag{2}$$

6.2 Final design solution

After wasting a considerable amount of time trying to get TI's battery gauge chip to work properly, the group had to turn to another solution as shown in the figure 14.

By inserting resistors R1 and R2 of known values, and using the voltmeter functionality of the dsPIC33FJ128MC802 micro-controller (for the choice of the components, refer to Section 7.1), the idea is to convert the measured voltage into a current that is assumed to be constant over a certain time interval that can be modified according to the use of the prototype.

The intermediate wiring consists of differential operational amplifiers (MCP6281 or MCP6271) supplied with [0V-3.3V]. This configuration allows to avoid possible short-circuit (contrary to an inverter assembly for example) as well as to have a great freedom of choice of resistors in order to amplify/attenuate the signal if necessary.

The output voltages of the op-amps are then measured at pins AN0, AN1 of the micro-controller. A great advantage to using the dsPIC is that there are 2 more pins (AN4 and AN5) that are available to mesure voltages. This makes the prototype easily adaptable to monitor a second battery. A PICkit 3 programmer is used to load the code into the micro-controller. It will thus not be present in the final prototype.

The code implemented on the dsPIC via the MPLAB X IDE platform is used to create an energy unit gauge representing the instantaneous capacity of the battery. This gauge can be incremented/decremented according to the battery's incoming/outgoing current balance.

This is the equation implemented in the code:

$$gauge(t) = gauge(t-1) + \frac{voltage\_in}{resistor\_in} time\_in - \frac{voltage\_out}{resistor\_out} time\_out \tag{3}$$

where the $voltage\_in$ and $voltage\_out$ are the measured voltage drops on the 2 resistors and $time\_in$ and $time\_out$ are the time intervals during which the current is assumed constant.

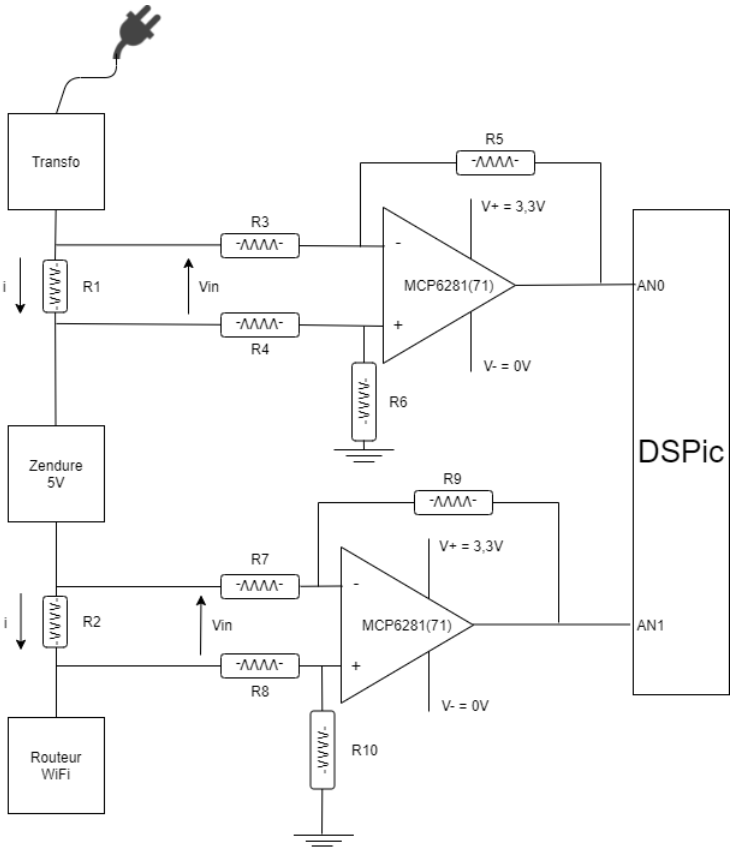

Figure 14: Final design solution - Measurement of the SOC

6.3 Assumptions and scenarios for the use of the Zendure battery

The different scenarios for the use of the Zendure battery are shown in the diagram in the Figure 15.

However, two important assumptions have been made:

1. The battery used is a Zendure with a small display showing its remaining percentage.
2. The codepo prototype is always connected to the battery.

In both cases, if the conditions are not met, a gauge update or a calibration (full charge and discharge) should be performed:

1. If the battery used is not a Zendure or if it has a different capacity, the maximum value of the gauge is therefore no longer relevant and it will be necessary to activate the calibration mode which requires a complete charge/discharge of the battery to know the extreme values of the gauge. The battery is considered to be 100% charged when the incoming current is zero, so the gauge will record this max value as a new reference.
2. If the codepo prototype is disconnected[3] from the Zendure and it is used for other tasks or has been recharged. The max value of the gauge is still relevant but not the instantaneous value. There are 2 ways to update the gauge:

[3]Accidental short-time disconnections and reconnections are considered as micro cuts and do not require any calibration.

- Change its value via the ISIG platform based on what the Zendure screen displays (*the least accurate method*)
- Calibration mode to only charge the battery to 100% and the gauge will then update when it sees no more current coming in (*most accurate method*).

Several ideas for improving the accuracy and control of these updates/calibrations in particular cases were given. These improvements can be implemented and tested on site in Goma:

- Possibility to put LEDs on the prototype that would light up more or less to have an idea of the instantaneous level of the gauge and be able to compare it to the percentage indicated on the Zendure.
- Possibility of making the prototype "intelligent" so that the max value of the gauge can update itself, without having to go through the ISIG interface, according to the current entering the battery during a certain period. This would be usefull to consider the ageing of the battery.
- If the power supply fails or is interrupted during a calibration, the prototype must not think that the gauge is at maximum. To do this, it is possible to implement a safety feature in the code that compares the maximum value calibrated at that moment with those calculated previously or, knowing the value of the incoming current and the maximum capacity of the battery, it is possible to evaluate the theoretical charging time and compare it with the time between the beginning of the charge and the interruption of the network to know if the measurement is realistic or not.

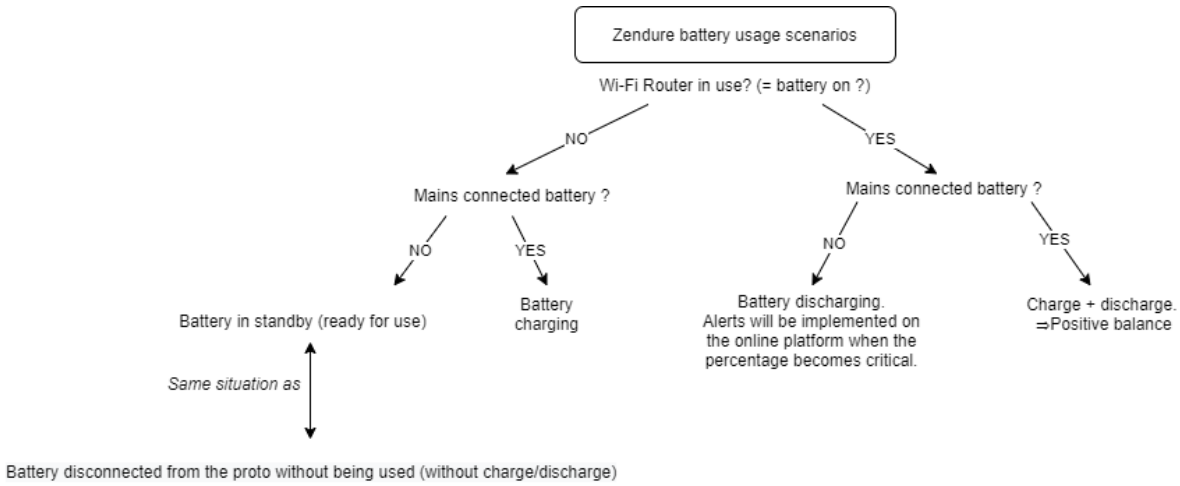

Figure 15: Zendure battery usage scenarios

7 Prototype

7.1 Choice of the components

7.1.1 Communication module

As explained in Section 5.1.2, the device will use the u-blox Thingstream services for the communication with the remote server. Thingstream itself proposes two communication modules, each consisting in a micro-controller printed on a PCB with a SIM card embedded on it, designed to communicate with the Thingstream server with USSD as a transport mechanism.

The first communication device is the *Thingstream IoT Module*. Its micro-controller is programmable: different instructions, not necessarily concerning communication, can be added. It cost 48.64€. This module can be connected with the *Thingstream IoT Starter Kit* which provides an easy access development base-board. This is the formula that was used as a communication module last year. However this product was discontinued this year.

The second communication device is the *Thingstream Click*. It presents an UART interface and can be configured and managed with a set of documented AT commands. The *Thingstream Click* does not have a micro-controller that could control the entire electronic device around the communication module. So the choice of this micro-controller is free and it is an advantage because the component will be adapted to the project (see the section 7.1.4): a very low consumption device, equipped with a read-only memory. Moreover, the pins are easily accessible which makes it easy to firstly prototize on a breadboard and secondly to solder to a PCB. The connectors for the GPS and GSM antennas are more robust than in the *Thingstream IoT Module*. To finish, it costs slightly less than the Module: 37.13€.

It is for these different advantages that the Click module was chosen for this project. A quick user guide is available on MikroElektronica website.[4].

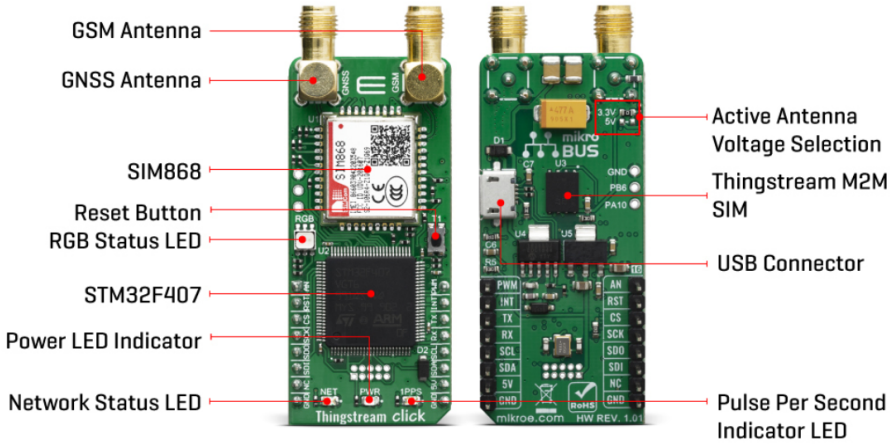

Figure 16: Thingstream Click

7.1.2 Power supply

The input voltage of the Thingstream Click is 5V, so the output voltage of the rechargeable power supply must be 5V. Three Ansmann external powerbank battery seem to meet expectations:

[4] https://download.mikroe.com/documents/add-on-boards/click/%20thingstream_click/thingstream-click-user-manual-02-10-2019.pdf

Rechargeable power supply							
Model	Size	Input Voltage	Output Voltage	Maximum Input Current	Maximum Output Current	Capacity	Price
Powerbank 5.4	67×120× 12 mm	5V	5V	2.2A	2.4A	5.4Ah	13.44€
Powerbank 10.8	71×123× 19 mm	5V	5V	2.2A	2.4A	10.8Ah	20.13€
Powerbank 20.8	79×160× 21 mm	5V	5V	2.2A	2.5A	20Ah	33.57€

Table 2: Comparation table: Rechargeable power supply

The Powerbank 20.8 has been chosen for its big capacity, its dimensions in accordance with the specifications. Moreover, this battery has two USB ports to power two components in the same time ans its price is reasonable.

Figure 17: Ansmann external Powerbank 20.8 battery

7.1.3 Operational Amplifiers

The differential operational amplifiers are used to ensure a good impedance matching of the measured tension drop on the resistors. As mentioned in section 6, the output of the Op Amp will then be connected to an analog pin on the dsPIC which will act as a voltmeter.

Figure 18: Operational amplifier MCP6281-E/P

7.1.4 Micro-controller

As mentioned in 7.1.1, the *Thingstream Click* does not provide an easily programmable micro-controller which could control the entire electronic device of the prototype.

After a discussion with Michel Osée, the latter proposed a model of micro-controller: DSPIC33FJ128MC802. He had already had the opportunity to use it in a project where *Thingstream Click* was also the communication module and where measurements on other electronic components had to be retrieved at regular time intervals. Moreover, it presents power-saving modes for a low-consumption, a 128 kbytes of Program Flash Memory and 5 16-bit Timers, which corresponds to the expectations mentioned in the beginning of the section 4. It is important to specify that it must be powered between 3 and 3.6V to operate. This micro-controller is used as starting point in the project. The following data-sheet contains the description of this micro-controller and others of the same family[5].

Figure 19: Micro-controller DSPIC33FJ128MC802

7.1.5 Voltage Regulator

As mentioned in the previous section, the micro-controller must be supplied with a voltage of 3 to 3.6V. It is then necessary to regulate the input voltage coming from the power supply. Under the advice of Michel Osée, the MCP1702-3302E/TO voltage regulator has been used in the following circuit:

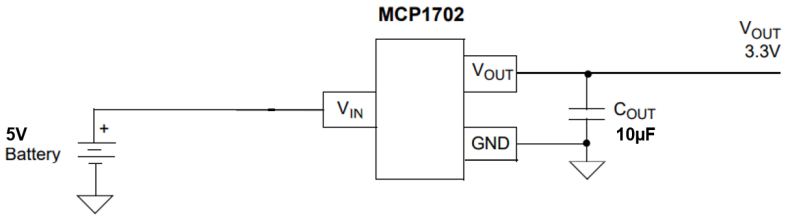

Figure 20: Circuit of voltage regulator

This voltage regulator is suitable for the prototype's electronic circuitry: it has an input operating voltage range of 2.7V to 13.2V and a standard output voltage range option of 3.3V. It can deliver up to 250mA while consuming only 2µA of quiescent current [6].

Figure 21: Voltage regulator : MCP1702-3302E/TO

7.1.6 Logical Switch

In order to be as energy-efficient as possible, we need a way to power off the communication device when it is not used. This can be done thanks to a logical switch controlled by the micro-controller.

[5] https://ww1.microchip.com/downloads/en/DeviceDoc/70291G.pdf
[6] https://docs.rs-online.com/8231/0900766b8137eb12.pdf

During the same discussion with Michel Osée mentioned above, the P-Channel MOSFET IRFD9014PBF was mentioned and is used in the following circuit:

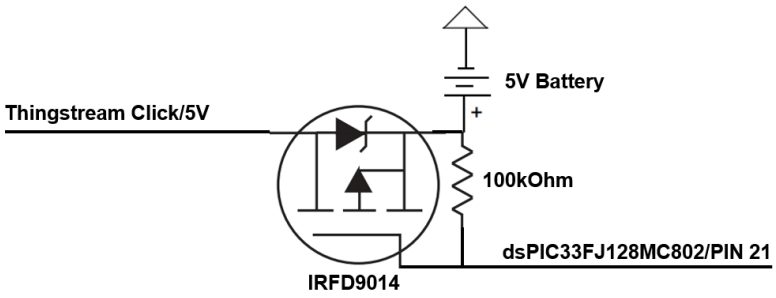

Figure 22: Circuit of logical switch

This PMOS transistor features fast switching, a ruggedized device design and a low drain-source on-sate resistance (0.5 Ohm)[7].

7.1.7 GPS antenna

To give an accurate localisation, GNSS (Global Navigation Satellite Systems) antenna can be attached to the Click module like indicated in figure 16. Different antennas have been selected:

GPS antennas							
Model	Size	Cable Length	Min. Volt	Max. Volt	Current consumption	Bandwidth	Price
Active GPS Antenna	48.5 × 39 × 15 mm	2m	/	/	/	5MHz	6.62€
Siretta Mike 3A	47 × 35 × 17 mm	0.5m	3V	5V	10mA at 3V	10MHz	7.66€

Table 3: Comparison table: GPS antenna

The first proposed antenna in the table 3 comes from the same supplier as the Thingstream Click: Mikro Electronika, but it does not provide information on important specifications, e.g. the different voltages, current consumption. In the data-sheet of the second antenna, the description states that the device is ideal for applications where saving power is important thanks to a low current like mentionned in the table 3. The length, width and cable length are smaller for the second antenna. Siretta Mike 3A seems to be more adequate for this project. Indeed, the economy of power used and the space optimisation are important aspects for the device. Its data-sheet is available on the following link [8].

[7] https://docs.rs-online.com/e320/0900766b807911b0.pdf
[8] https://docs.rs-online.com/4ca1/0900766b815ccd4a.pdf

Figure 23: Siretta GPS Antenna MIKE 3A

7.1.8 GSM antenna

A GSM antenna can also be attached to the Click module like indicated in figure 16. This allows the communication module to connect to the Thingstream global network over GSM and to send informations by USSD. Different antennas have been selected:

GSM antennas				
Model	Size	Cable lenght	Bandwidth	Price
Rubber Antenna GSM right angle	50mm	no cable	70/180Mhz	5.69€
Siretta Alpha 3A	$72 \times 25 \times 7.7$ mm	1m	100MHz	9.72€

Table 4: Comparation table: GSM antenna

The Rubber Antenna seemed much less cumbersome with a higher bandwidth and cheaper but we tested it in practise during this second semester and decided to opt for the Siretta Alpha 3A. Its datasheet can be found in the following link [9]

Figure 24: Siretta Alpha 3A

7.2 Printed Circuit Board

In order to optimise the size of the prototype, two Printed Circuit Boards (PCBs) were designed. The first design mainly contains Surface Mounted Devices (SMDs), but as this version is more difficult to weld, an "Through Hole Technology" (THT) version, as a second PCB, has been designed.

[9] `http://www.farnell.com/datasheets/2644037.pdf`

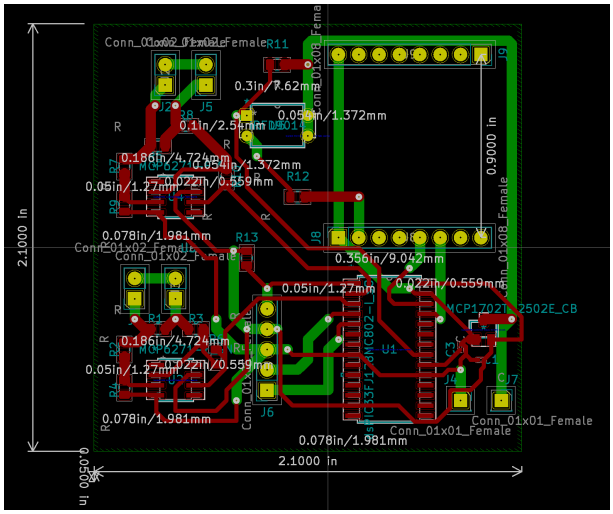

Figure 25: PCB with Surface Mounted Devices

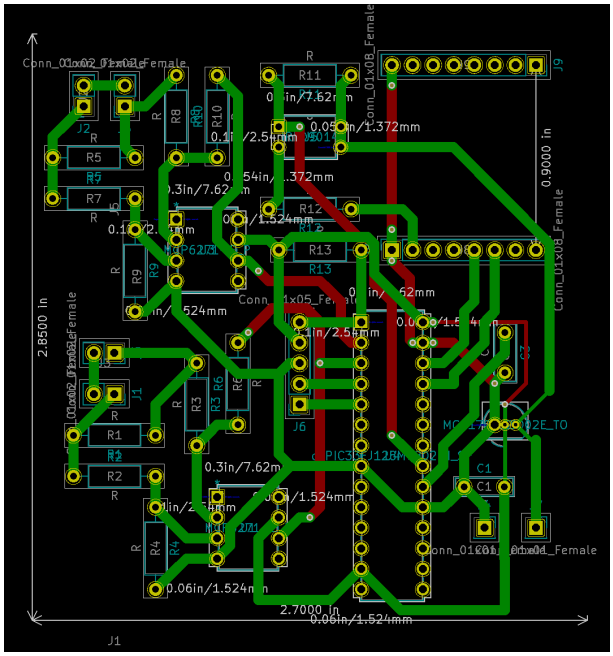

Figure 26: PCB with Through Hole Technology components only

Here are the different components to be placed on the PCB using the SMD electronic card manufacturing technique:

Figure 27: Micro-controller DSPIC33FJ128MC802

Figure 28: Voltage regulator : MCP1702S

Figure 29: Differential operational amplifiers MCP6281

7.3 PCB protective box

The PCB protection box has been designed on SolidWorks taking into account the cables entering and leaving the Zendure, the size of the PCB as well as the volume of the components that will be soldered on it. Its dimensions are as follows : 14.3cm x 7.8cm x 6.2cm (691,548 cm^3).

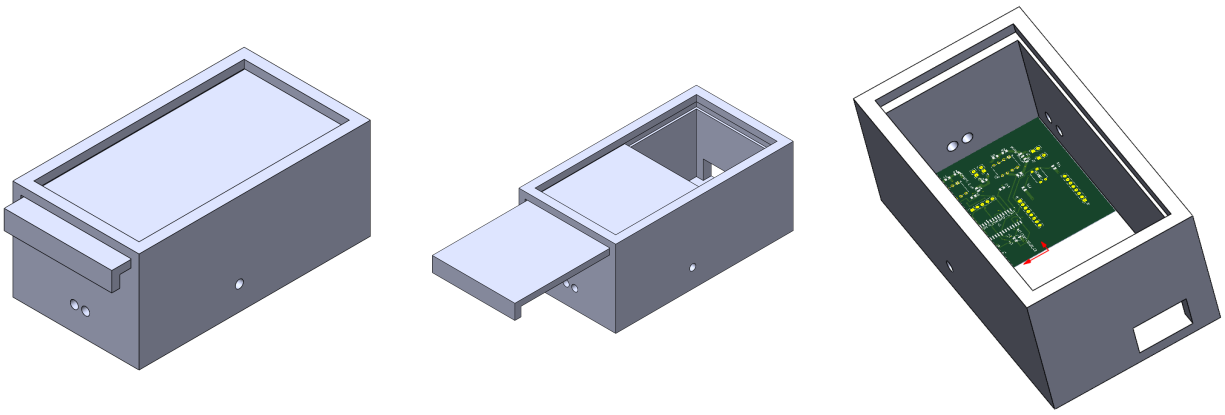

Figure 30: Different views of the closed/semi-open/open pcb protective box.

A first version has been 3D printed and will be tested in the MSF case.

7.3.1 Cost of the prototype

Here is a table with the cost of the prototype. These costs don't include the shipping fees, the cables, the conception of the PCB and the 3D impression of the case.

Cost of a prototype	
Component	**Price**
Thingstream Click	37.13€
PowerBank 20.8	33.57€
2 Op Amp MCP6281	0.55€/unit
Micro-controller DSPIC33FJI28MC802-I/SP	8.99€
Voltage Regulator	0.42€
Logical Switch	0.76€
GPS Antenna: Siretta Mike 3A	7.66€
GSM Antenna: Rubber Antenna GSM right angle	5.69€
Total cost per prototype	**95.32€**

Table 5: Cost of a prototype

8 Software

This Chapter aims to describe the C code that we implemented. In Section 8.1 we'll discuss about the different "modes" that our device will run and Section 8.2 presents the design pattern we implemented in order for our device to complete its tasks properly.

8.1 Functioning modes

From a software perspective, our prototype will run under various "modes" that can be remotely changed with a message being sent from the server to the prototype. The incoming message will be scanned by the code and actualise the MODE variable with the value provided in the message.

The different modes are the following:

- **MODE=1**: The SOC and the GPS position are transmitted to the server.
- **MODE=2**: Only the SOC value is transmitted to the server.
- **MODE=3**: Low-power mode (economy).

Similar to the MODE change, different key values will be parametered in order to be possibly changed remotely by an incoming message from the server.

Such parameters will be:

- The interval of time at which the device retrieves the SOC from the gauge
- The interval of time at which it turns the communication module on to send a message to the server. When not sending, the microcontroller powers down the communication module and does its battery gauge routine algorithm.

The reasons behind those changes can either be :

1. to ensure the security of MSF's teams on the ground in a dangerous region by turning off the GPS localisation
2. or to save-up energy by reducing the interval of transmissions and/or by stopping the acquisition of the GPS position

A simplified flowchart of the code that represents *most* (not all) of the interactions and commands that are performed is presented on Figure 33. We insist on the fact that this is a representation of the functioning of our code,not the real flow but it is handy to get a good intuition of the interactions and tasks that are performed. In reality, our code implements a design pattern that is discussed in Section 8.2.

As can be seen on Figure 33, as long as it is not time to send a message, the usual routine is to measure the current and tension in order to update the SOC. This routine is performed while the communication module (Thingstream Click) is powered off, and consists in:

- staying in deep sleep for a fixed interval of time
- waking up after this fixed interval of time to measure the incoming/outgoing current and voltage of the Zendure battery
- going back to deep sleep

When it is time to send a message, the communication module is powered on and different AT instructions are sent in order to establish a connection to the server.

The first task is to subscribe to a MQTT topic used to communicate instructions from the web to the device. The micro controller listens to the last message posted on the topic, extracts the data and update the functioning parameters of the code (which mode to enter, interval between transmissions, sleep time, etc.).

Once the incoming message is well processed, we get the relevant information depending on the mode we're on. The SOC is any case already computed so the question is whether we need to get the GPS coordinates via the TS Click's GNSS antenna.

- It is the case in **MODE==1**, so the GNSS antenna is powered on and several attempts are made to get a GNSS fix. Once we get a GNSS fix or too many unsuccessful attempts have been made, the GNSS antenna is powered off. Then, we try to send the message with the relevant data to the server, we disconnect and we destroy the connection with Thingstream.
- Otherwise, in **MODE==2**, no GPS coordinates are needed and we go straight to sending the message to the server.Then, we disconnect and we destroy the connection with Thingstream. In addition to be faster to complete, this mode also consumes less energy than the first mode.

- Otherwise, in **MODE==3**, the instruction is to save as much energy as possible so we directly disconnect, we destroy the connection and we augment the interval of time between 2 transmissions since the latter is the most energy consuming task.

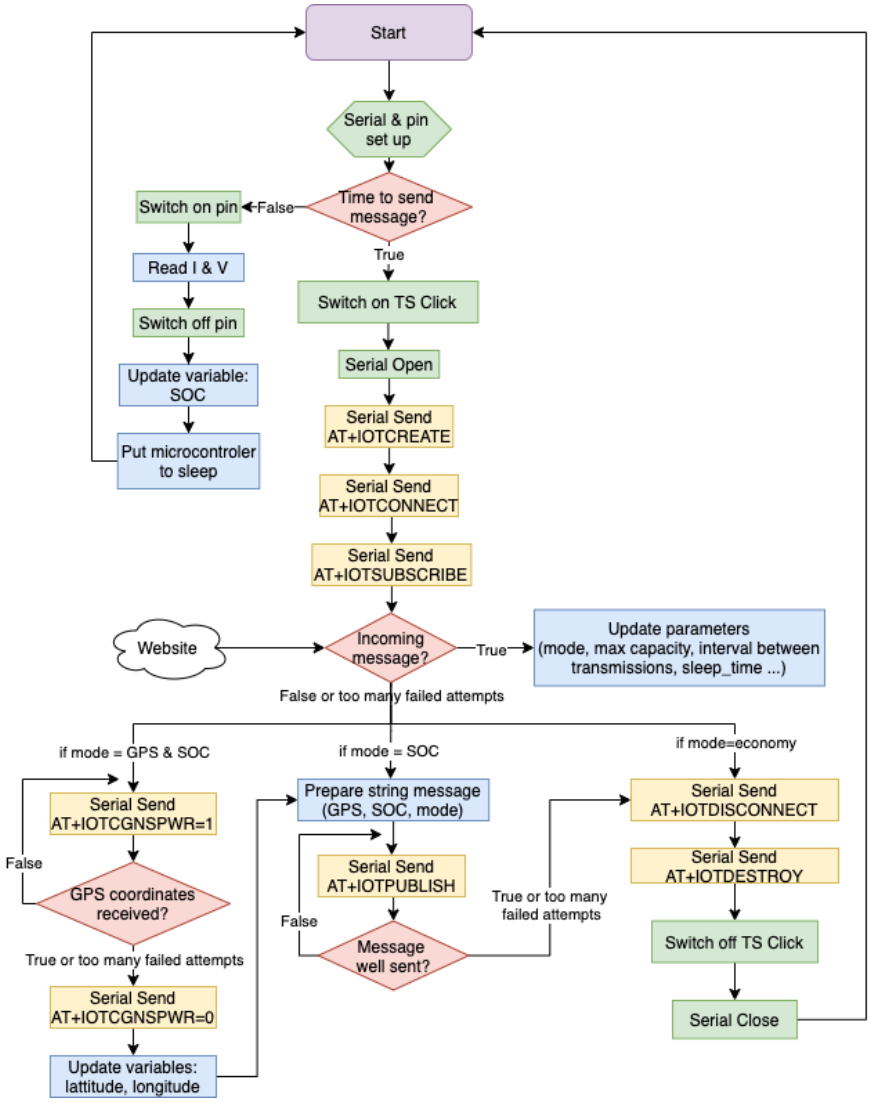

Figure 31: Flowchart of the code

8.2 Design Pattern

We decided to implement a Finite State Machine (FSM), as it is well suited for embedded systems. This design is handy to break complex engineering problems into smaller - easier - parts. FSM allows to beak down the design into a series of steps, called **states**. Each state performs a narrowly defined task. **Events** are the stimuli of the FSM and cause the FSM to transition between states.[3] [6]. This allows us to handle more easily the flow of the program and to handle errors.

The finite state machine that we proposed is presented on Figure 32. It follows the same philosophy than the flowchart presented on Figure 33, but represents the different states of our FSM.

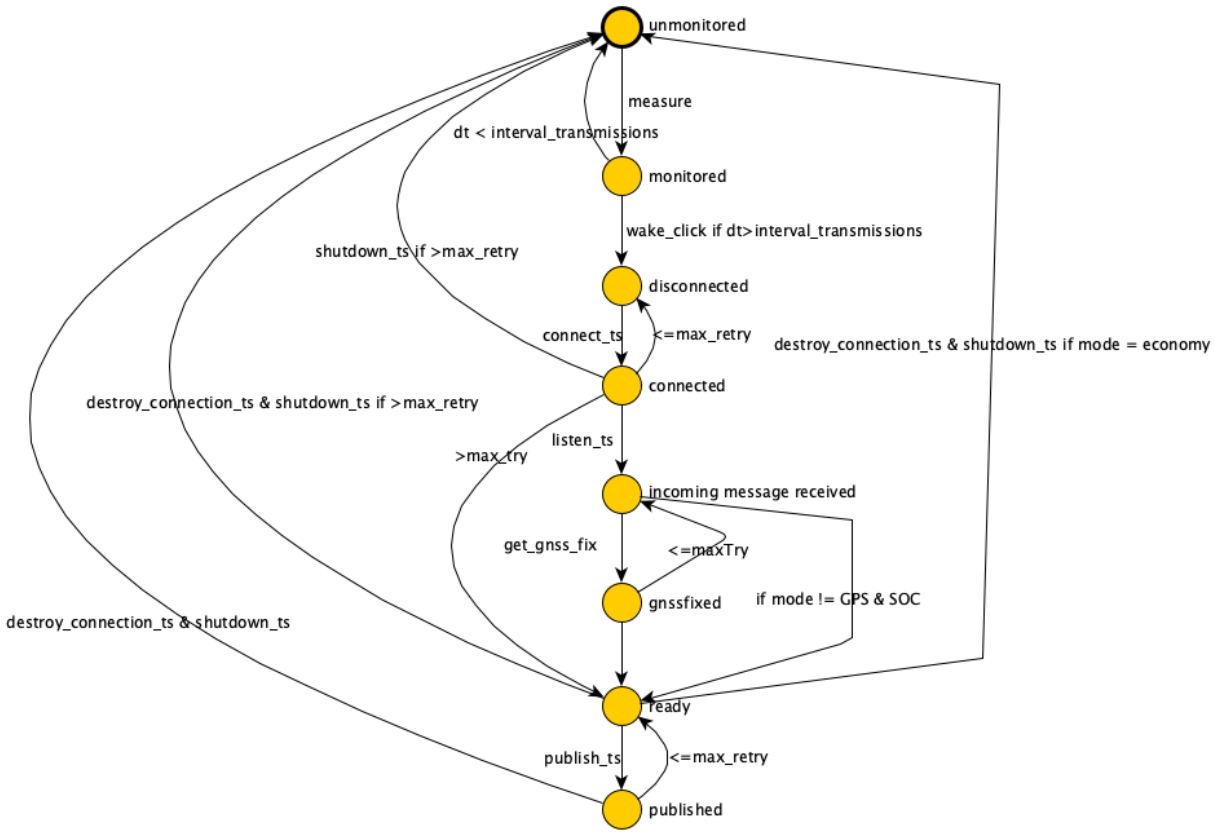

Figure 32: Finite State Machine

As can be seen on Figure 32, the flow of our code is described by states, and events allow transitions between states. Such events are either the successful completion of a task or a condition that is met.

The GitHub link for the code can be found at this address: `https://github.com/NicolasWa/CODEPO_MSF_Monitoring`.

9 Consumption of the prototype

In this section, the current values, for different functioning modes, are given in order to get an idea of the consumption of the prototype and its autonomy. These current measurements were taken on electronic circuit mounted on the breadboard since the PCB wasn't printed yet.

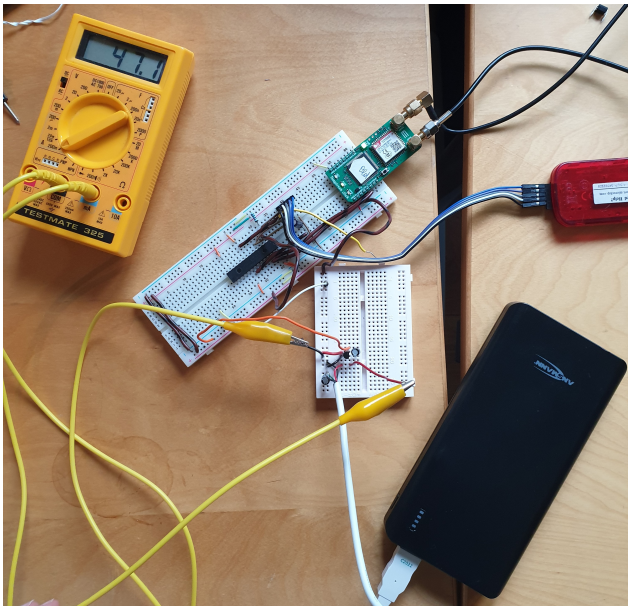

Figure 33: Electronic circuit with ammeters

To calculate the current consumed by the prototype, an ampmeter is placed between the external battery and the assembly.

Tasks	Calculated current (mA)	Power (W)
1) Low-power mode	4.2	0.01386
2) Computation of the SOC	17	0.0561
3) Click searching for GPS coordinate	47.1	0.2355
4) Sending messages	141.7	0.7085

Table 6: Consumption of the prototype

Tasks	Calculated peak current (mA)
1) Micro-controller in deep sleep mode	4.2
2) Computation of the SOC	17
3) Click searching for GPS coordinate	47.1
4) Sending messages	141.7

Table 7: Consumption of the prototype

The different contexts of the measures are listed above:

1) Low-power mode: mode mentioned in section 8.1, i.e. the lowest energy operating mode. This is the main mode of the prototype.

2) Computation of the SOC : the computation of the SOC when the communication module is turned off. This calculation is performed in a few milliseconds.

3) Click searching for GPS coordinate : the search for GPS coordinates through the Click. This search may take more or less time depending on the location of the case and therefore the available GPS network. On the Solbosh campus, the coordinates were found after more or less 4 minutes.
4) Sending messages : sending messages in a topic which takes more or less 10 seconds to execute.

As a reminder, the capacity of the Powerbank is 20Ah and the supply voltage is 5V.

Depending on the frequency at which the information is requested, the autonomy of the prototype may vary. In a practical case of use of the case illustrated during a meeting with Christophe CASTAIGNE (Field ICT Program Manager for Médecins Sans Frontières), the router was used once a day for a month. If the SOC and GPS coordinates are sent two times a day, assuming that the available GPS network is not as accessible as in Brussels and that the GPS coordinates would be picked up after 10 minutes, the consumption for one day would be:

- **Sending GSP coordinates and SOC two times a day** :

$$(23.67h * 4.2mA) + (0.328h * 47.1mA) + (0.0056h * 141.7mA) = 0.115Ah \tag{4}$$

- **Sending GSP coordinates and SOC four times a day**:

$$(23h * 4.2mA) + (0.989h * 47.1mA) + (0.011h * 141.7mA) = 0.145Ah \tag{5}$$

The prototype would then have an autonomy of 173 days for the first case and 138 days for the second. For this result, if the calculation of the SOC is done once every minute, the current consumed is neglected because it will pass to 17mA a few milliseconds compared to 60 seconds at a value of 4.2mA. These theoretical results are very encouraging, it will be necessary to test the autonomy in practice.

The prototype battery discharge time has been calculated making the same assumptions, and under the same calculation conditions, we calculated how long the device would be able to **loop** in MODE=1 (GPS & SOC) which is the worst-case scenario in terms of energy consumption. We insist on the the fact that in practice the code doesn't loop on this mode without powering down the TS Click and remaining in deep sleep most of the time. However, it gives an indication of the benefits of staying as long as possible in deep sleep and stresses the fact that one should be conscious of the impact of sending more often data to the server.

- **MODE = 1 (without any deep sleep)** :

$$20Ah/((0.0167 * 141.7mA) + (0.9833 * 47.1mA)) = 410.874h \tag{6}$$

- **Deep sleep**:

$$20Ah/4.2mA = 4761.9h \tag{7}$$

For MODE 1, in this experience, it is assumed that the GNSS antenna finds the GPS coordinates within 10 minutes and sends the SOC and GPS data within 10 seconds, i.e. in one hour the module sends the data for more or less 1 minute and receives the GPS network for more or less 59 minutes.

From this experience, we observe that if the prototype were to loop in mode 1 non stop (without going to deep sleep before the next moment to send data), it would discharge after 17 days. This is much less than the 3 months requested in the specifications. This reason drove us to make extensive use of the deep sleep functionality of the micro controller in order to save up as much energy as possible and meet the requirements.

10 Organization of the project and the various problems encountered

At the end of the first semester, the prototype was not functional. At the level of communication, we encountered compilation problems for the programming of the micro-controller DSPIC33FJ128MC802, which prevented the sending of AT commands to the communication module. For the part of the SOC calculation, we used the BQ34110 battery gauge chip from Texas Instrument and had difficulties to calibrate the Zendure on the battery management platform, which falsified the SOC value proposed by the platform. A Gantt chart for the second semester was proposed:

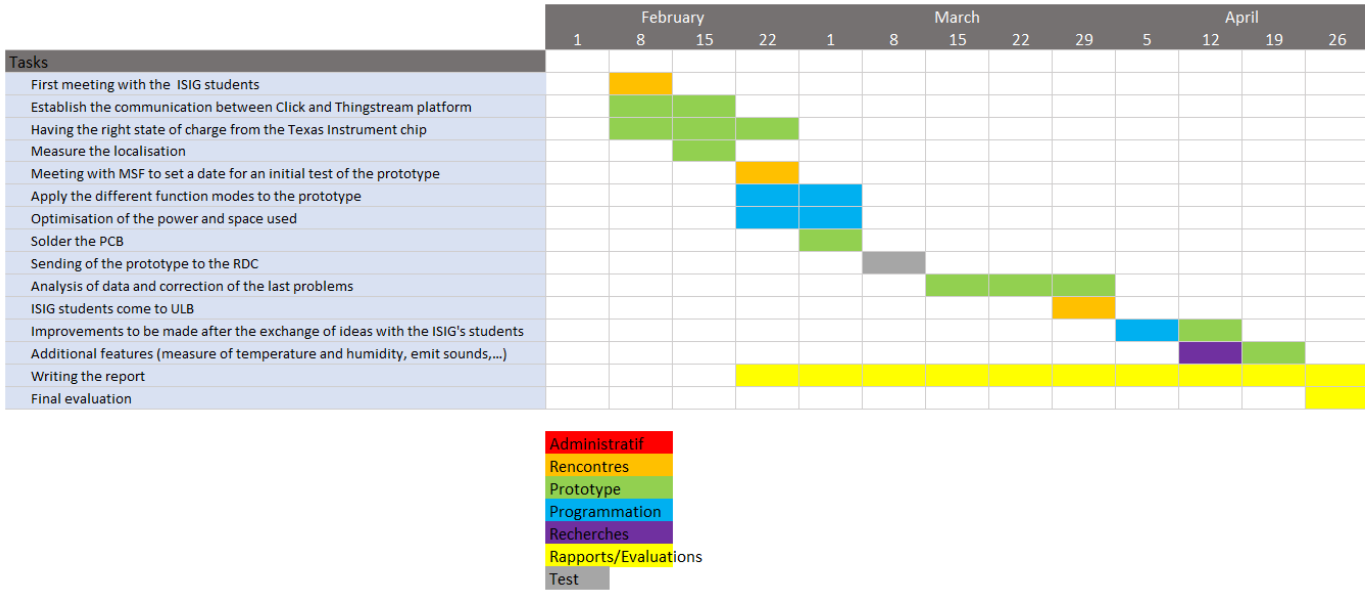

Figure 34: Gantt diagram proposed for the second semester

The goal was to make the prototype operational by the end of February in order to send it to the Democratic Republic of Congo at the beginning of March and to test it in the ground. The ISIG students could not go to Belgium because of the health crisis we are currently facing. We will have the pleasure to welcome them from August 23 to 29, 2021 to continue our collaboration.

During the second semester, in order to be able to work together on the project, we fixed two meetings per week at the beginning of February, which we maintained more or less regularly until the end of the project, in the laboratory at the level UA1 at the university. The second semester did not go exactly as planned due to various problems encountered.

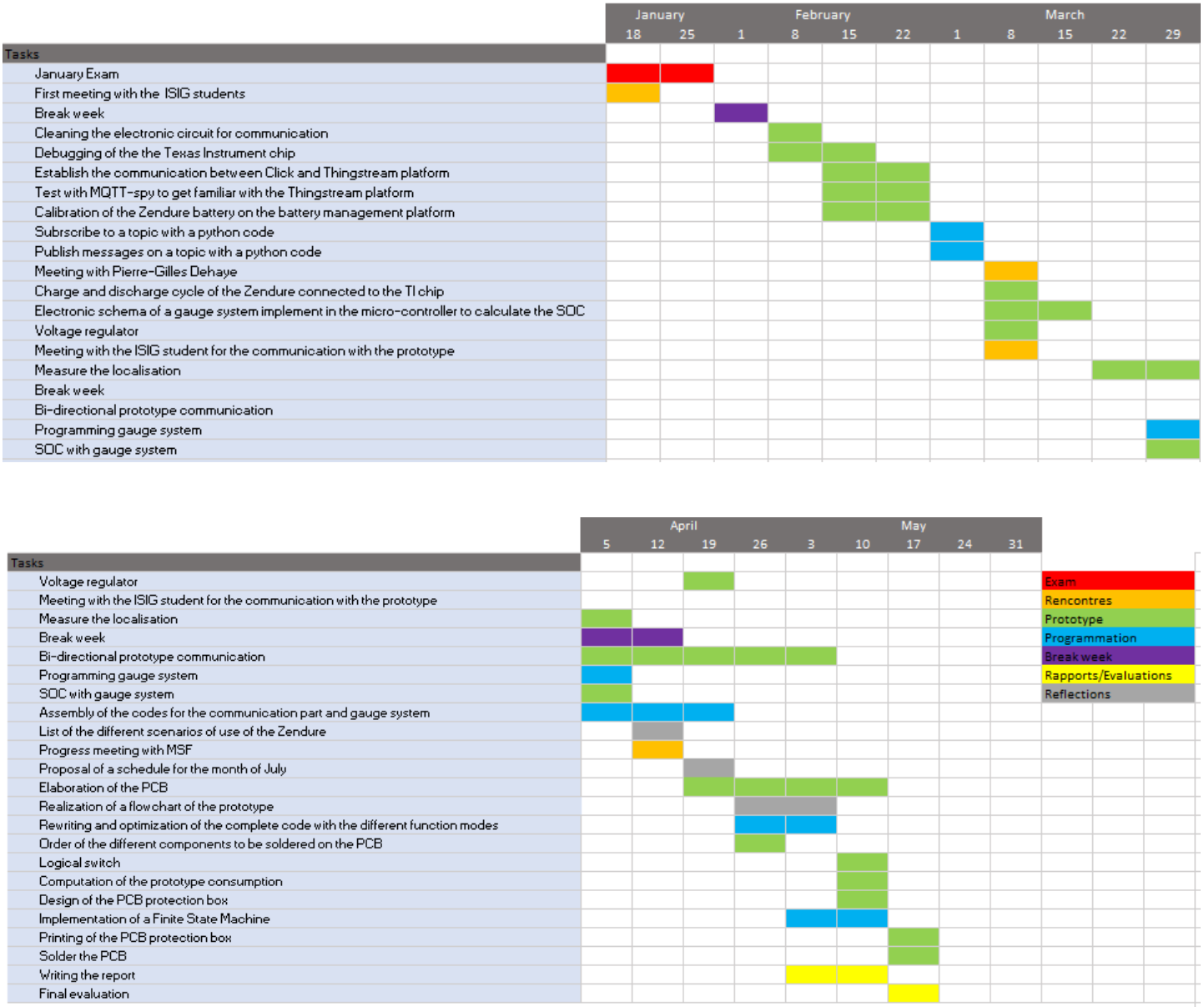

Figure 35: Gantt diagram for the second semester

In the Figure 35 are the different steps of the project during the second semester. We tried to solve the different problems with the TI chip for the SOC calculation until mid-March before opting for the gauge system.

Another difficulty, in the communication part, was to become familiar with the Thingstream platform which provided little information on its use and that of the Click module. But, thanks to the Python codes that subscribe to a topic and publish in a topic, the prototype is independent from the Thingstream platform. The last problems in the bidirectional communication could be solved after rewriting the complete code. With the new architecture and flowchart, the code was clearer and gave us more information about the reception of AT commands by the Click.

For now, the Printed Circuit Board (PCB) has been designed but hasn't been printed yet.

11 Schedule for the month of July

A schedule for the month of July has been proposed:

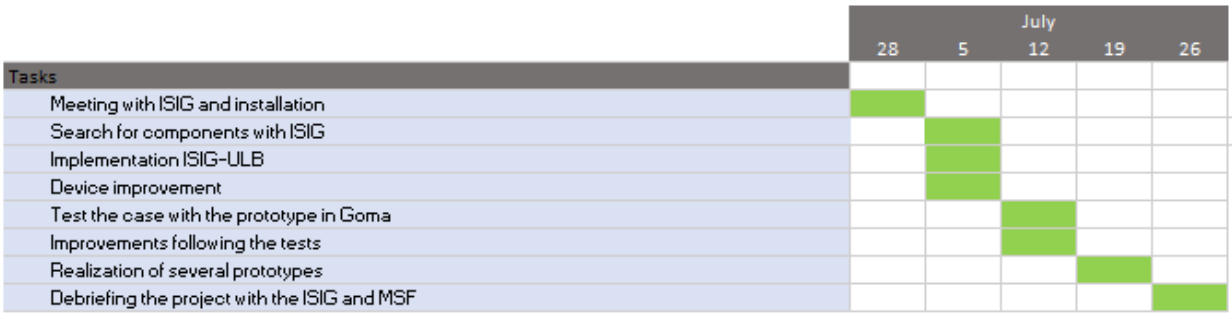

Figure 36: July schedule proposed

One of the main tasks of the stay is to reproduce our prototype in DRC. Some components not easily accessible on site will have to be ordered before the departure, like the Thingstream Click communication module. The micro-controller, the voltage regulator, the logic switch, the operational amplifiers have already been ordered in several copies. During the first week on site, we will look for the different components available on site, and we will assemble the software part worked by the ISIG with our prototype. We could then work on the various improvements to be brought to our prototype such as:

- measure the accuracy of the gauge algorithm and try to improve it,
- handle more gracefully the timeouts
- implementation of the calibration mode, with a button on the prototype,
- the calculation of the remaining time of the Zendure,
- adapt the prototype for a case containing two Zendure battery packs,
- insert a temperature sensor,
- install alerts on the web site when the battery is low or when the temperature is too high.

We will then be able to test the case and implement the prototype in Goma with the ISIG.

12 Conclusion

This project is a unique occasion for the students to experience the professionals-client relation with a real & useful project that has a great impact. The elaboration of the monitoring device asked by MSF is challenging and multi-disciplinary: the requirements have great impact on the choice of the components, on the design of the device itself and on the software.

Last year, CODEPO's team began this work and proposed a prototype that achieved some results but that had weak points: the autonomy of the device was only of a few days versus the 3 months required, the power source of the device was non rechargeable, once the device was powered down all the information about the monitored battery's SOC was lost and couldn't be recovered and the prototype wasn't small enough. The sanitary crisis provoked by the Covid-19 pandemic slowed this project and even led to the cancelling of the trip to Goma, in RDC, where the students had to implement the device on the ground.

This year's CODEPO team had to adapt to the sanitary crisis and continue the work of its predecessors. However, major changes were introduced to overcome the problems faced by last year's prototype. This year's kept the Thingstream platform and services for the communication between the device and the cloud but brought serious changes to the hardware in order to: reduce the size of the prototype, to allow the use of a rechargeable battery as the device's power supply, to be more energy efficient and to have a more accurate estimation of the SOC thanks to the coulomb counting method.

The reflection about how we would improve this prototype and the search of new adequate components took a great amount of time in the first semester. Again, during the second semester, a false lead on how to professionally measure the SOC with a dedicated chip consumed a lot of our time and budget and has proved itself unsuccessful. Late in the second semester, the group returned to its original idea of implementing a coulomb counting method. The measures of current and voltage were correctly implemented and an algorithm was written. Unfortunately, the accuracy of the battery gauge algorithm hasn't been assessed yet due to the delay introduced by the problems of implementing the professional chip and will be done as quickly as possible after the exams.
However, the device is able to receive messages from the web and send messages to the web thanks to its communication module TS Click. It can retrieve GPS coordinates, provided that the GNSS reception is good enough at the location of the device.

The main missions that the students would like to achieve for this summer are: the optimisation of their SOC algorithm, the implementation of a calibration mode and the calculation of the remaining time to empty (of the Zendure).

To conclude, the group has all the cards in their hands to finalise their prototype. They have clear objectives on their missions during July and are really looking forward finalising this project and to work with the students from Goma.

Bibliography

[1] Loserian S. Laizer Baraka W. Nyamtiga Anael Sam. "Security perspectives for USSD versus SMS in conducting mobile transactions: a case study of Tanzania". In: *INTERNATIONAL JOURNAL OF TECHNOLOGY ENHANCEMENTS AND EMERGING ENGINEERING RESEARCH (2013)* (), VOL 1, ISSUE 3.

[2] https://mqtt.org. *MQTT home page*. URL: https://mqtt.org.

[3] David Lafreniere. *State Machine Design in C*. URL: https://www.codeproject.com/Articles/1275479/State-Machine-Design-in-C.

[4] B. Blunier N. Watrin and A. Miraoui. "Review of adaptive systems for lithium batteries state-of-charge and state-of-health estimation". In: *Proceedings of IEEE Transportation Electrification Conference anddExpo* (), pp. 1–6.

[5] Kong Soon Ng et al. "Enhanced coulomb counting method for estimating state-of-charge and state-of-health of lithium-ion batteries". In: *Applied energy* 86.9 (2009), pp. 1506–1511.

[6] Amlendra On. *State Machine using C*. URL: https://aticleworld.com/state-machine-using-c/.

[7] U-Blox. *MQTT beginner's guide*. URL: https://www.u-blox.com/en/blogs/insights/mqtt-beginners-guide#mqtt-01.

[8] U-Blox. *Thingstream*. URL: https://www.u-blox.com/en/product/thingstream.

[9] Y. Wang and K. Song. "A new approach to realize UART". In: *Proceedings of 2011 International Conference on Electronic Mechanical Engineering and Information Technology*. Vol. 5. 2011, pp. 2749–2752. DOI: 10.1109/EMEIT.2011.6023602.

A Printed Circuit Board