POLYTECH UNIVERSITY SCHOOL OF MONTPELLIER

Industrial Project of the End of Studies
Electronic, Robotic and Industrial Informatic
Departement

YEAR 2018-2019

PROJECT

Paul LELOUP

THE IOT LORA PROJECT
The Smart Sensor

ECOLE POLYTECHNIQUE UNIVERSITAIRE DE MONTPELLIER
UNIVERSITE MONTPELLIER II SCIENCES ET TECHNIQUES DU LANGUEDOC
Place Eugène Bataillon 34095 MONTPELLIER CEDEX 5
Tél. : 04 67 14 31 60 – Fax : 04 67 14 45 14
E-mail : scola@polytech.univ-montp2.fr

POLYTECH MONTPELLIER

1. Summary

2. Acknowledgements

I thank the Laboratory of Computer Science, Robotics and Microelectronics of Montpellier (LIRMM) for having welcomed me during this Industrial End of Study Project (PIFE).

I would especially like to thank Pascal Benoit, an university lecturer at Polytech and LIRMM, who gave me his confidence to carry out this project and who supervised me every week, as well as Guillaume Patrigeon, PhD at the LIRMM, which helped me greatly in the installation of the workstation and throughout these 200 hours of work.

To do my PIFE at the LIRMM was a pleasure, I could learn a lot thanks to you and to validate my acquired of my curriculum in engineering school.

3. Introduction

The number of connected devices (Internet of Things: IoT) in circulation in the world in 2018 is 23.14 billion [1] and in the next few years the number of devices could reach between 50 and 80 billion. It is estimated that connected devices worldwide in 2013 consumed 616 Twh [2], which is greater than the consumption of France in one year (eg 544 Twh in 2015 [3]). The saving of electric current represents billions of euros if the consumption of these objects was decrease.

The project is part of an industrial end-of-course project which aims to put the student in the position of a project manager responsible for finding solutions to a concrete problem that is posed to him. The title of PIFE is "Study & Design of a Smart Sensor on FPGA".

The IoT project purpose is very interesting to me, and designing a smart object as well as programming and participating in its improvement is relevant because I want to work in this field.

The Internet of Things (IoT, Figure 1) is the extension of the internet to things and places in the physical world. The connected Internet of Things represents the exchange of information and data from real-world devices with the Internet.

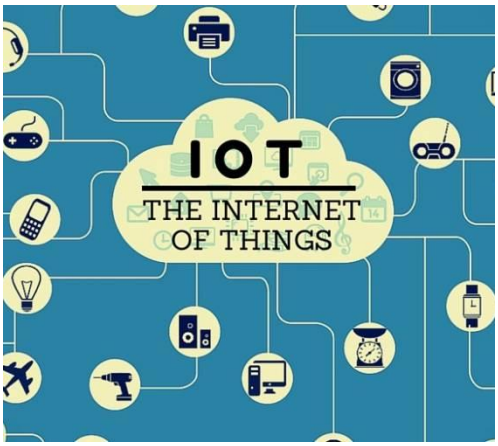


Figure 1 Illustration IoT

The place of study is the Laboratory of Informatics, Robotics and Microelectronics of Montpellier (LIRMM, Figure 2) which is a joint research unit jointly dependent on the University Montpellier (UM) and the National Center for Scientific Research (CNRS).

Figure 2 LIRMM photo

4. Specifications

4.1. Suject / Goal

Présentation of the project:

The project was proposed by Pascal Benoit, because as part of research work carried out at the LIRMM on the theme of the Internet of Things, researchers develop an experimental platform consisting of connected objects, network gateways, as well as servers. for the processing and visualization of collected data.

For this, they work on a prototype Smart Sensor: it is based on an FPGA card, to which any type of sensor and radio can be connected. This architecture allows rapid prototyping of a connected sensor architecture: for example, one may want to test and compare two types of radio transmission, or configure in the FPGA a Cortex M0 processor or a Cortex M3, or play on the size of embedded memory, all this in order to evaluate the various solutions, and to be able to compare them in terms of performance and energy.

This Smart Sensor prototype is based on a DIGILENT CMODA7 card for which a complete microcontroller architecture is describe in VHDL is available: it is composed of a Cortex M0 processor, an internal memory, and various peripherals. In order to quickly prototype various types of technologies we want to create a port expansion card that will connect different types of radio modules and sensors.

The objective of the project is to create a test bench to calculate the consumption and performance of a smart object to optimize them. This is to help Guillaume Patrigeon's thesis on "Ultra-low power adaptive integrated systems for the Internet of Things".

Firstly, it will be necessary to make a bibliographic study of the different types of sensors and radio modules possible, and to specify several cases of use.

Then I have create an electronic board extension ports that allows you to connect different sensors and radio modules in order to customize the smart object. This card will be the Smart Sensor, it will also be called the sensor node. It will connect the CMOD A7, radio modules and sensors.

Once the card is created, there will be a software part that will capture the information from the sensors, which will process them and then send them by radio.

In the end, when the test bench will be functional, we will be in a real application of intelligent object. There will be an architectural exploration later to lower the consumption of microcontrollers implemented in connected objects.

Here are the specifications for the work to be done during this Industrial End of Study Project.

Specifications :

- Make a bibliographic study of the different types of sensors and radio modules possible, and specify several use cases
- Operate the GPS module and send the data received by LoRa with the Cmod A7
- Make a Smart Sensor electronic card that will have these features:
    - Modular thanks to Pmod connectors
    - Possibility to connect at least 5 modules (sensor, radio modules)
    - Reconfigurable and flexible thanks to the Cmod A7 FPGA
    - Reprogrammable by micro USB easily accessible
    - Battery powered for outdoor testing and less power supply noise than USB
    - Easy to measure the consumption and power of each module
    - Voltage regulator and current limiter system to secure the components of the board.
- Make the datasheet of the Smart Sensor card
- Make a Pmod LoRa electronic card that will have these features:
    - Modular thanks to a Pmod connection
    - Possibility to connect a LoRa module
    - Possibility to connect an antenna
    - Easy to measure the power consumption and power of the module
- Make the datasheet of the Pmod LoRa card
- Option: Add temperature, humidity and brightness sensors and program them with the electronic board created to send the data by LoRa
- Option: Add a Bluetooth radio module and send the created frame with it
- Option: Make a consumption measurement of the Smart Sensor

Planification des taches

See the Gantt created at the beginning of the project in Appendix (Gantt).

4.2. System

The system has the same architecture as a connected object, the Smart Sensor architecture is like the diagram below (Figure 3).

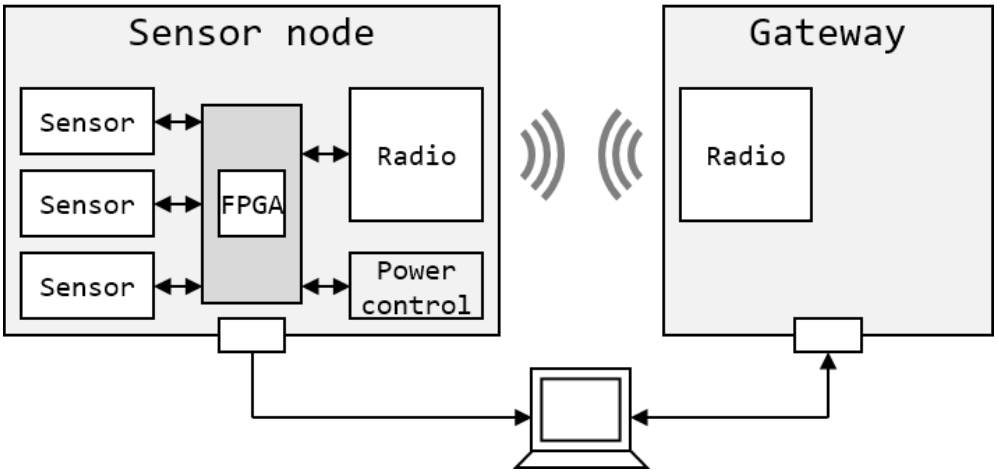


Figure 3 Architecture of Smart Sensor and his Gateway

The card consists of a microcontroller (implemented in an FPGA, the Cmod A7) that will communicate with modules that may be sensors or radio communication systems. Part for battery power is also present.

The particularity of the Smart Sensor that makes it a smart object like no other is that it is modular and therefore reconfigurable. So all possible sensors and radios can be connected to the board if they have Pmod connectivity. Here there are a picture of sensors and radio modules with the Pmod connection (Figure 4)

Figure 4 Pmod modules

We chose to use this connection because there are a very large number of sensors with it. It is also common for prototyping quickly.

These modules also have the advantage of never using analog pins, they communicate with the microcontroller using communication interfaces such as UART, SPI, I2C and GPIO.

Here is the diagram of the Smart Sensor (Figure 5). Each module has Pmod connectivity and they are all interchangeable.

It will just fit the code and indicate the change of the pins.

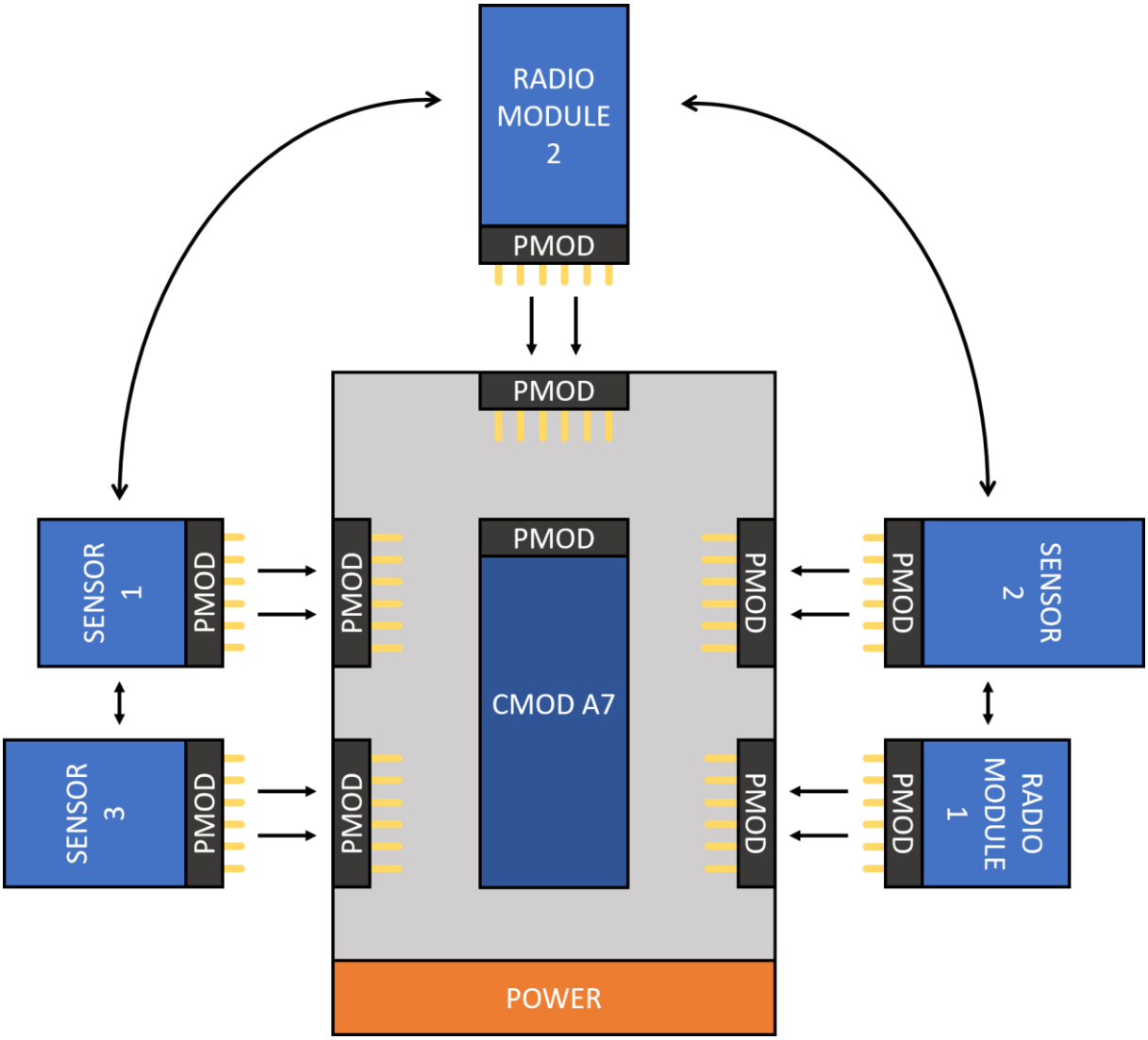


Figure 5 Smart Sensor modulaire

In terms of power, the Smart Sensor is powered by either USB-microUSB cable or battery. The two power systems each have their advantages:

- The USB-microUSB power cable allows you to upload the C code to modify the program. It also allows software development because you can view a UART on the terminal of the computer. This makes it easy to debug and view the data sent and received from the Gateway.
- Battery power has very little noise (high frequencies) and allows for precise and noise-free consumption measurements unlike USB generated by a computer. There is also the advantage of mobility, which allows for scope testing with the Gateway.

4.3. Hardware

The main controller of the node is the Digilent CmodA7. It is a compact 48-pin DIP card that features an Artix 7 FPGA from Xilinx. This is the version of the CmodA7 35T that will be used here. As the CmodA7 already has pushbuttons and light-emitting diodes, it will not be necessary to overload the motherboard of the sensor node with these elements.

In our application, the first need is that the hardware is reconfigurable. Thanks to the flexibility of the FPGA, it is possible to explore different architectures and therefore different processor solutions (for example: ARM Cortex-M0 and Cortex-M3, LIRMM's SecretBlaze, etc ...). It is also possible to replace this card with another, with a microcontroller trade to compare the proposed solutions with existing products. This study of microcontroller architectures is part of other projects within the LIRMM.

Here are the pros and cons of using an FPGA chip:

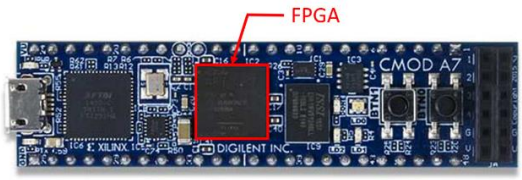


Figure 6 CMOD A7 card of DIGILENT

Advantages	Disadvantages
• Possibility of prototyping • Flexibility • Time-to-market weak • Adaptability to future evolutions thanks to the reconfiguration	• Limited integration by routing resources • Performance • High unit price for large productions

What interests us the most is the possibility of prototyping and flexibility. Although the Artix 7 FPGA present is limited in terms of performance and resources, the targeted architectures are those of very low-power microcontrollers operating at relatively low frequencies (order of ten MHz) and with a reduced number of peripherals. The limited integration problem due to routing resources has been eliminated because the system is modular with Pmod connectivity and does not require analog pins.

We will integrate in the FPGA chip a microcontroller with ARM cortex-m0 processor. To integrate a microcontroller into an FPGA, it must be described as a VHDL file (.VHD) and follow all the steps summarized in this diagram (Figure 7).

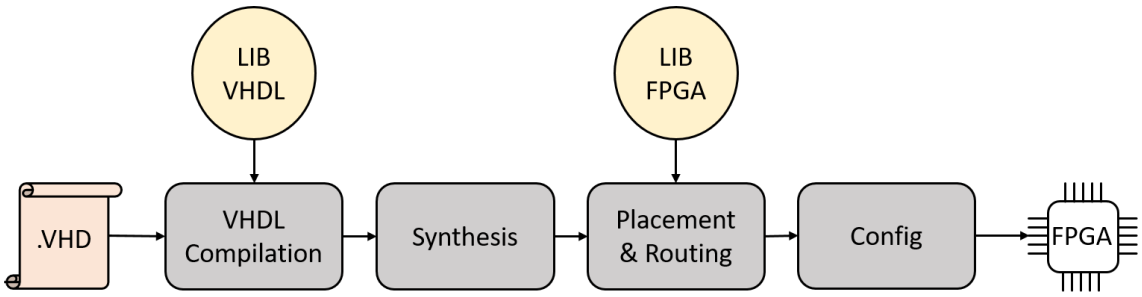


Figure 7 Integrate a microcontroleur in a FPGA

4.4. Software

The software is reprogrammable as in most cases. It allows to configure all the GPIO input and output, configure up to 12 communication interfaces (4 UART, 4 SPI and 4 I2C) each pin of these interfaces can be chosen by software.

At the level of the global structure of the code, here is the cyclic functioning of the code (Figure 8).

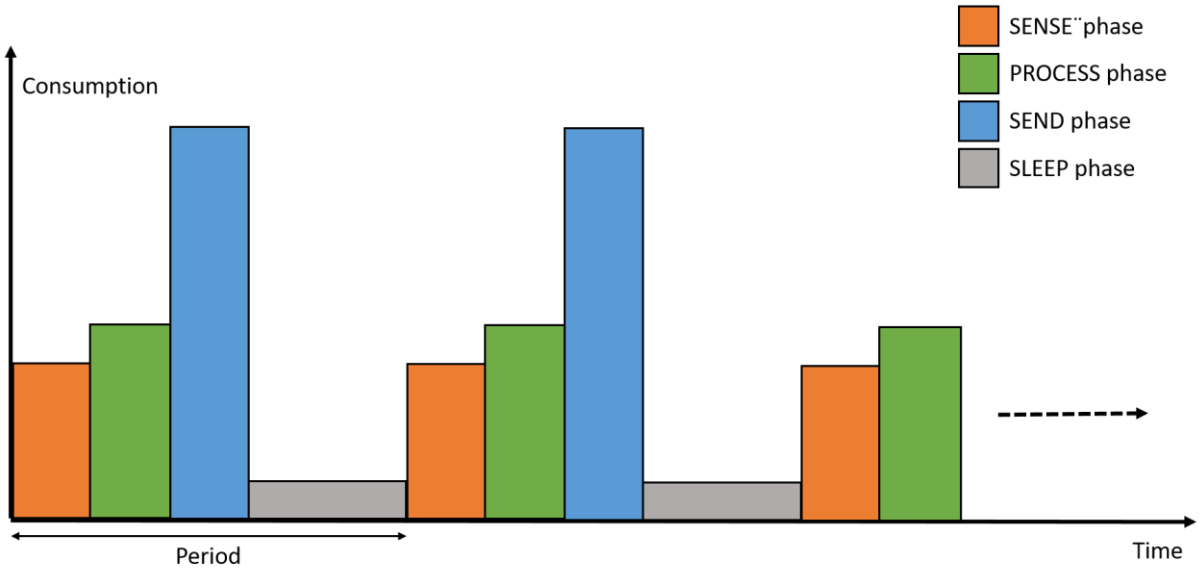


Figure 8 Differents steps of the software

It is composed by 4 parts that are:

- The SENSE phase: It allows to collect the information of the sensors and stores them in a variable.
- The PROCESS phase: It processes the bytes recorded by the previous phase in order to extract the desired information. This information is formatted and grouped in a concise message (a frame to send).
- The SEND phase: This phase sends the previously created frame with one or more radio modules (eg LoRa, BLE).
- SLEEP phase: It puts the microcontroller in sleep mode to save energy because sending the temperature every 5 ms would not be relevant in this application.

5. System

5.1. Design of the card

Figure 9 Smart Sensor Card

Electronic card

For more information on the Smart Sensor map, please go to the appendices to view the Smart Card Datasheet. The schematic of the board as well as the board is also in the datasheet.

The electronic board (Figure 9) consists of 5 female Pmod connectors where each pin of each connector is connected to CMOD A7 which is the component in the center of it.

It was chosen a power system to regulate the battery voltage. The voltage of the battery is 7.4V, a voltage higher than 5V to be able to supply all the components of the circuit. There is a first linear voltage regulator to lower the voltage from 7.4 to 5V. 5V is the voltage required by the CMOD A7 to operate. Since there is no 3.3V power pin on the CMOD A7 board, another linear regulator has been placed at 5V to generate a 3.3V supply voltage for all Pmod connectors. Since linear regulators dissipate the extra voltage into heat, it was more appropriate to put them in series to avoid overheating. There are also different components ensuring the robustness of the power supply and the low noise:

- A Schottky diode in series to prevent reverse currents and not to degrade the battery.
- A Zener diode in parallel to have the most constant supply voltage possible.
- Large capacitors in parallel to fully respond to current calls from the Smart Sensor.
- LEDs to visualize the 5V and 3.3V power supply.
- Small capacitors in parallel to limit the high frequencies on the power supply as close as possible to the output pins of the power supply.

I have also positioned pins that allow measurement of current and voltage simultaneously (power). This makes it possible to measure the consumption of each Pmod separately, of all the Pmods together or of the CMOD A7.

Problèmes rencontrés

- Size of the holes for the components to check well (All the holes of the pins of connectors must be in 1mm, the regulators of tension, the connections Pmod also).
- The gap between the two CMOD A7 pin rails was not good.
- The battery connector is in CMS, so on the back, specify it in the software otherwise symmetry problem between VCC and GND.

As solutions to these problems it was made a second version of the card of which no problem has been detected until today.

5.2. LoRa

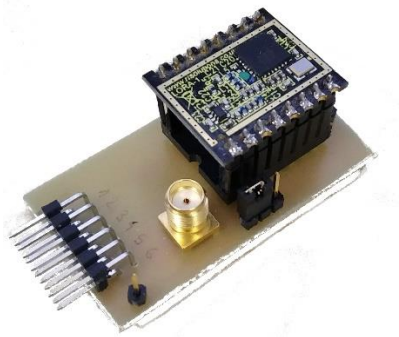


Figure 10 LoRa Pmod

For more information on the Pmod LoRa card, please go to the appendices to view the map datasheet (Datasheet Pmod LoRa). The schematic of the board as well as the board is also in the datasheet.

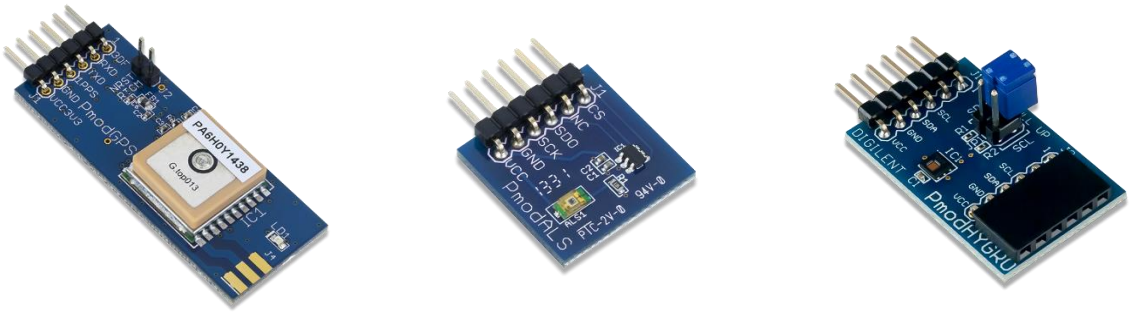
The electronic board (Figure 10) consists of a male Pmod connector where each pin is connected to LoRa RF-LoRa-868-SO module. The 5 extra pins are wired to a single connector.

An 868 MHz SMA antenna can be easily connected using the added connectors.

There is also the same system of measurement of consumption that on the map of the Smart Sensor.

This LoRa module was first used on a breadboard and without an antenna and thanks to this card, the use of it is much easier. It communicates by SPI with the microcontroller.

5.3. Pmod Sensors



(a) GPS Pmod (b) ALS Pmod (c) HYGRO Pmod

Figure 11 Differents Pmod used

This GPS Pmod (Figure 11 (a)) was the first sensor used with the CMOD A7, it is easy to use and uses the UART to communicate. It has also been used on breadboard.

The ALS Pmod (Figure 11 (b)) captures the brightness, it uses the SPI interface as well.

The latest Pmod sensor is the HYGRO Pmod (Figure 11 (c)), which captures air humidity and temperature. The one is different and uses the I2C interface.

All these sensors will make interesting measurements of consumption by the actual application of the Smart Sensor.

6. Hardware (FPGA)

6.1. System

The system has a global structure as below (Figure 12), it is the typical structure of a microcontroller.

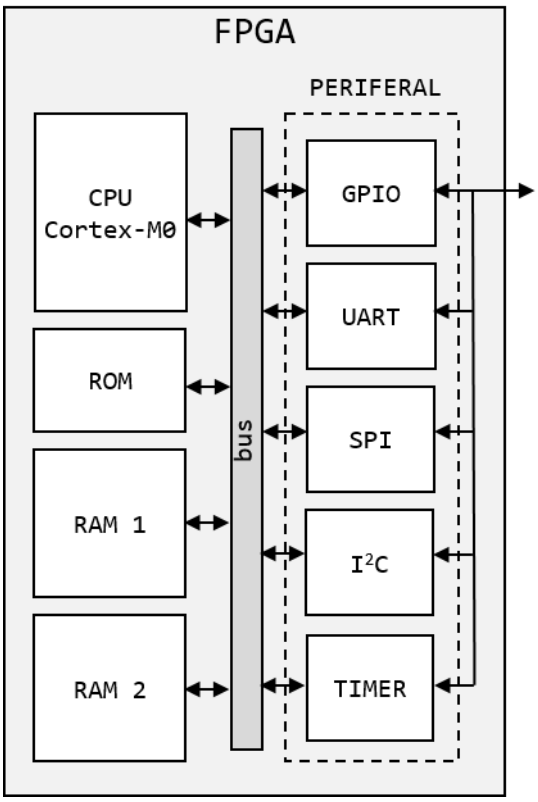


Figure 12 global Structure (FPGA)

So we have inside the FPGA chip a microcontroller composed of a processor, the Cortex-M0 whose specificities will be detailed in the next part, a ROM, two RAM and various peripherals. The TIMER will mainly serve to set a cycle of a constant period with interruption to release the microcontroller from its standby state.

6.2. Processor

The processor used in this study is the Cortex-M0 r1p0, from the well-known ARM Cortex-M series. Products based on this processor can be found to validate and compare the operation of the system implemented on the FPGA. The Cortex-M0 is a 3-stage 32-bit RISC processor that implements the ARMv6-M instruction set. It is given for a maximum frequency of 50 MHz, has 32 interrupt lines, a non-maskable interrupt line and a 32-bit single cycle multiplier in its r1p0 version. It has only one hardware interface, type AHB-Lite.

Here is the diagram of the functional blocks of the processor (Figure 13).

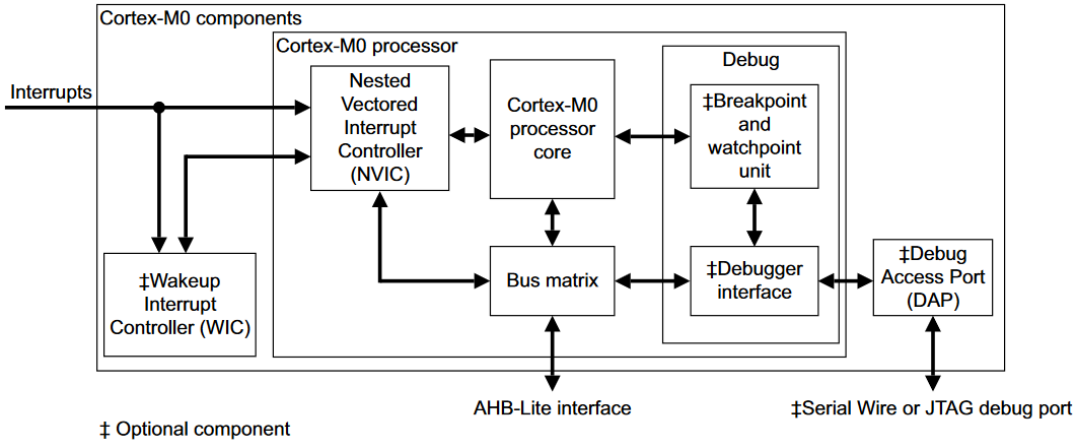


Figure 13 Functional block diagrams of the Cortex-M0

6.1. UART, SPI, I²C Interfaces

To communicate between the modules and the microcontroller, we will use communication interfaces such as UART, I2C and SPI.

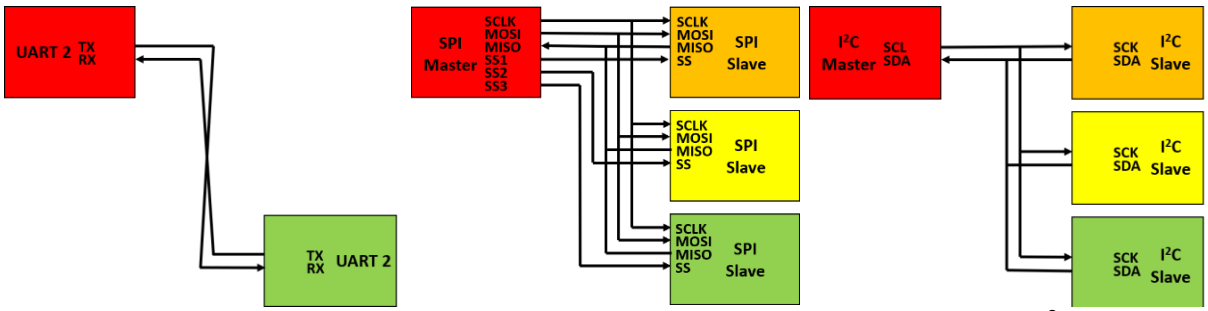


Figure 14 Communications interfaces

A UART (Figure 14 (a)), for Universal Asynchronous Receiver Transmitter, is a universal asynchronous transceiver.

An SPI link (Figure 14 (b)) (for Serial Peripheral Interface) is a synchronous serial data bus. The circuits communicate according to a master-slave scheme, where the master controls the communication. Several slaves can coexist on the same bus, in this case, the selection of the recipient is done by a dedicated line between the master and the slave called Slave Select.

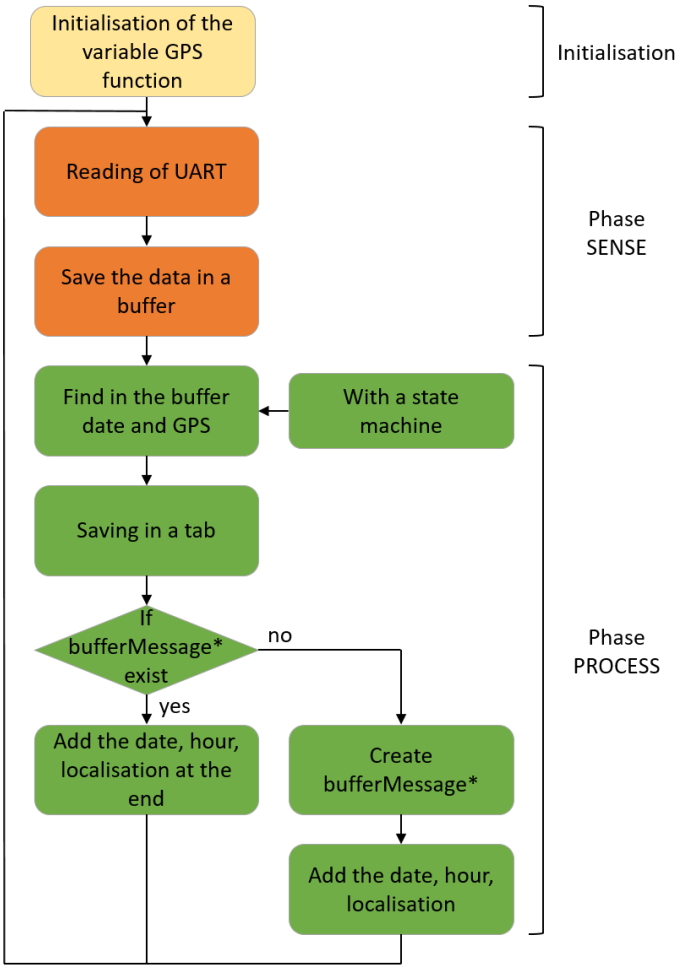
I2C (Figure 14 (c)) is a half-duplex bidirectional synchronous serial bus, where multiple devices, masters or slaves, can be connected to the bus.

UARTs, SPIs, and I2Cs are usually integrated into components like microcontrollers. In this case they are a peripheral function of the microcontroller.

7. Software (Flow chart)

7.1. GPS

The flowchart below (Figure 15) shows 3 functions, the "initGPS" function that initializes the Pmod, the "readGPS" function (Phase SENSE) and the "processMessageGPS" function (Phase PROCESS) that creates the frame (Figure 16).



*bufferMessage : It's the tab where we save data for send by radio.

Figure 15 Flow chart of the GPS function

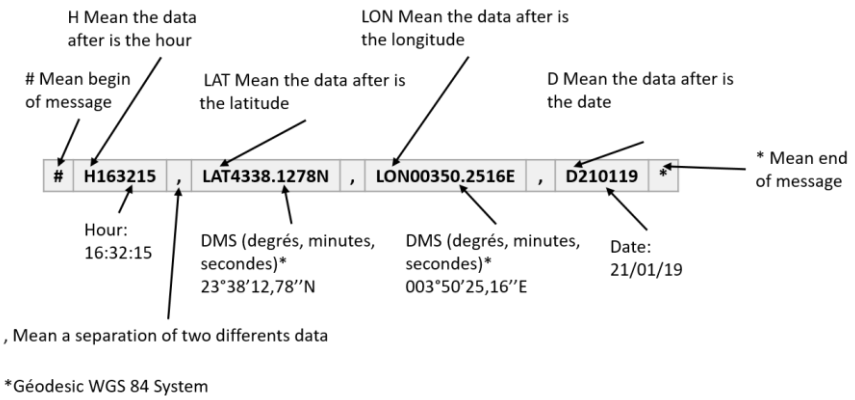
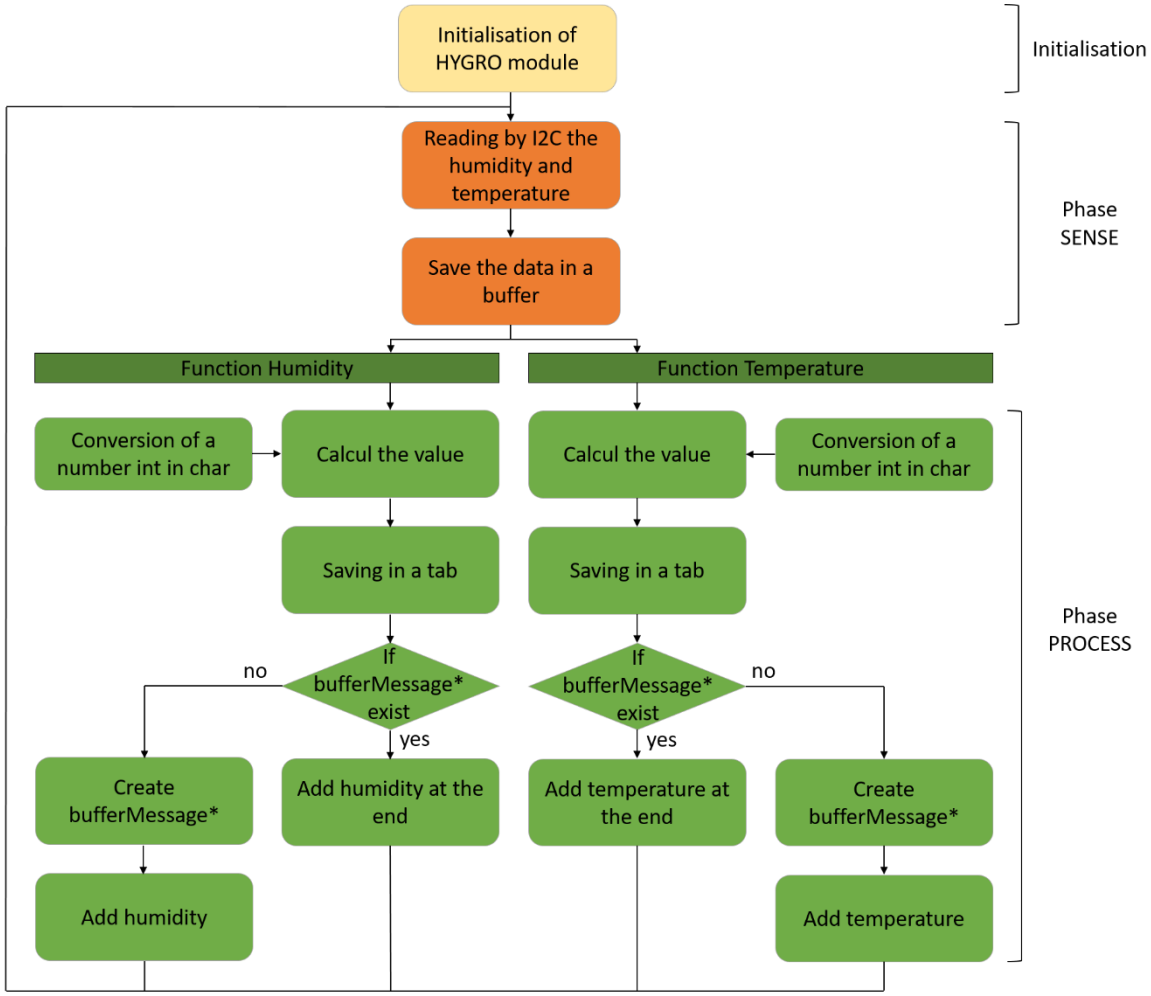


Figure 16 Function GPS frame

7.2. HYGRO

As the flow chart shows (Figure 17), there are going to be 4 functions here. The "initHYGRO" function to initialize the Pmod, the "readHYGRO" function that captures the humidity and temperature, then the last two functions "processMessageHumidityHYGRO" and "processMessageTemperatureHYGRO" which records and processes the raw data of the humidity and the temperature to create the frames (Figure 18, Figure 19).



*bufferMessage : It's the tab where we save data for send by radio.

Figure 17 Flow chart of the HYGRO function

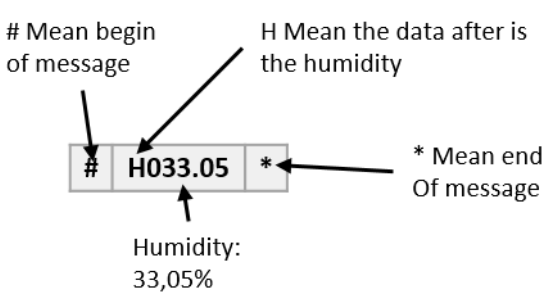


Figure 18 Function Humidity HYGRO frame

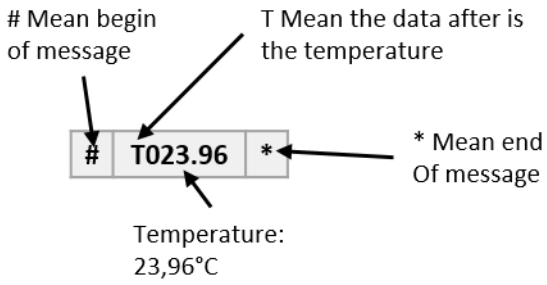
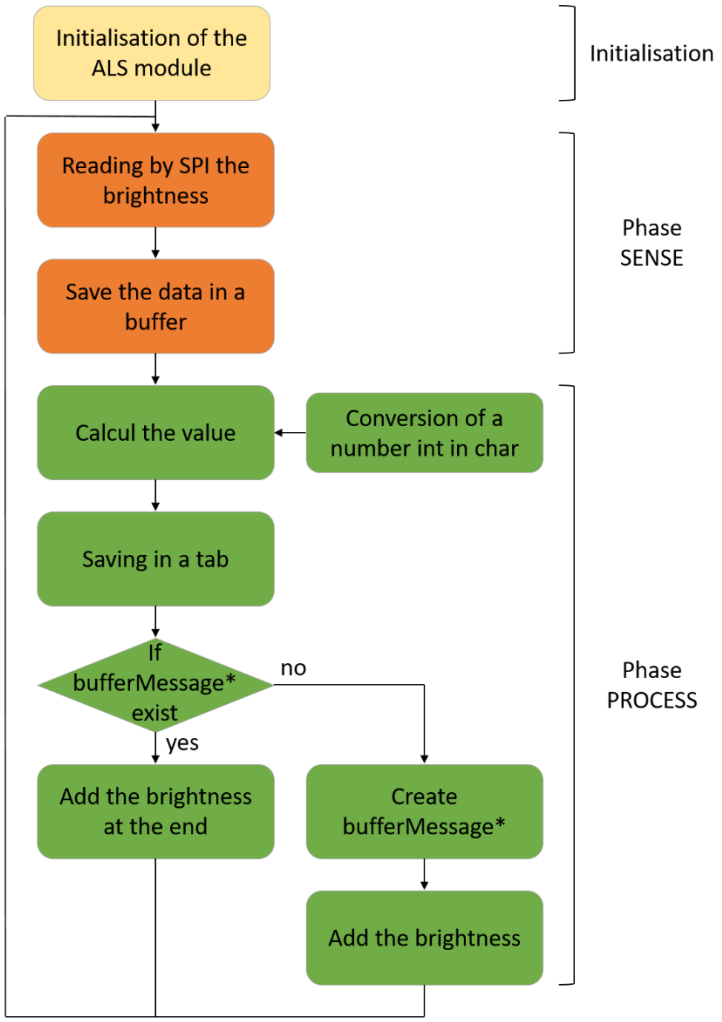


Figure 19 Function Temperature HYGRO frame

7.3. ALS

Here is the flow chart of the Pmod ALS, there is a function for the initialization, the phase SENSE of capturing raw data of the sensor and the function of the phase PROCESS which transforms the data.

In all functions of type "processMessage", there is the particularity that if the bufferMessage array is empty, they fill it with their data. Otherwise they add their data at the end of the table. This means that if we interchange the functions between them we will be able to create the frame in the desired order.



*bufferMessage : It's the tab where we save data for send by radio.

Figure 20 Flow chart of the ALS function

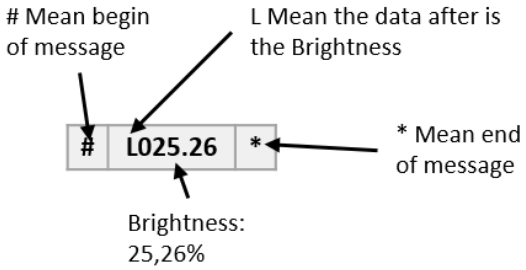


Figure 21 Function ALS frame

7.4. LoRa

There are two LoRa functions to run the Pmod:

- The "initLoRa" function initializes the module
- The "sendTabLora1" function that sends the array passed as a parameter.

These two functions are detailed in the flowchart below (Figure 22).

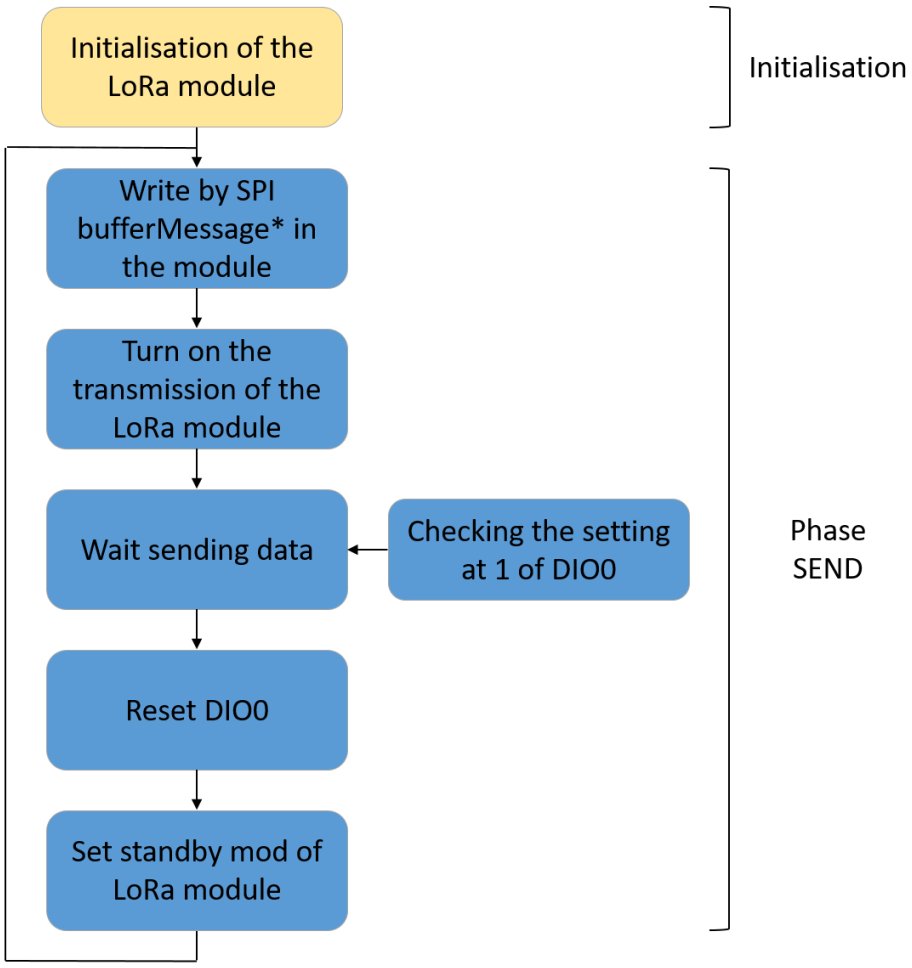


Figure 22 Flow chart of the LoRa function

The message to send in our application is bufferMessage which contains all processed and ordered sensor information.

7.5. Smart Sensor

The general flowchart of the code (Figure 23) consists of an initialization phase and 4 periodic phases.

The initialization phase in yellow makes it possible to initialize all the Pmods but also to define the GPIOs, the pins of the communications interfaces, the timers etc ...

The 4 periodic phases are described in the flowchart.

Below is an example of a frame (Figure 24) that the Smart Sensor sends.

The explanations of previous frames allow to decode this frame.

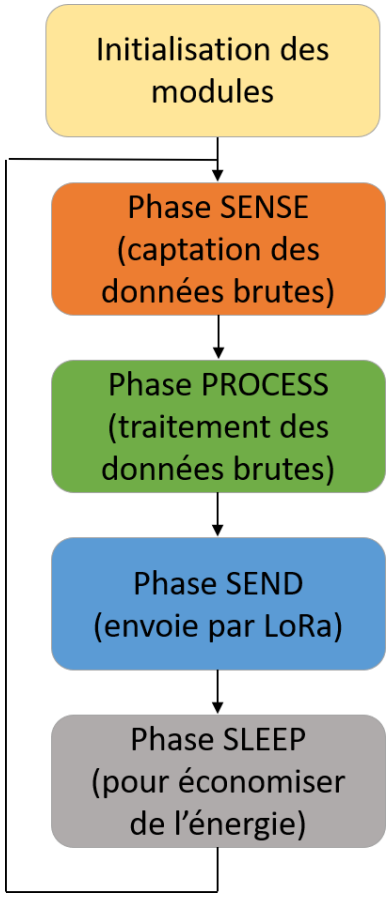


Figure 23 Flow chart of the Smart Sensor

#	H026,10	,	L014,50	,	T023,44	,	H163215	,	LAT4338.1278N	,	LON00350.2516E	,	D210119	*

Figure 24 Frame send by the Smart Sensor

8. Result

8.1. Flex Node

Here is the final result of the project (Figure 25): we see the CMOD A7, the sensors and the radio module.

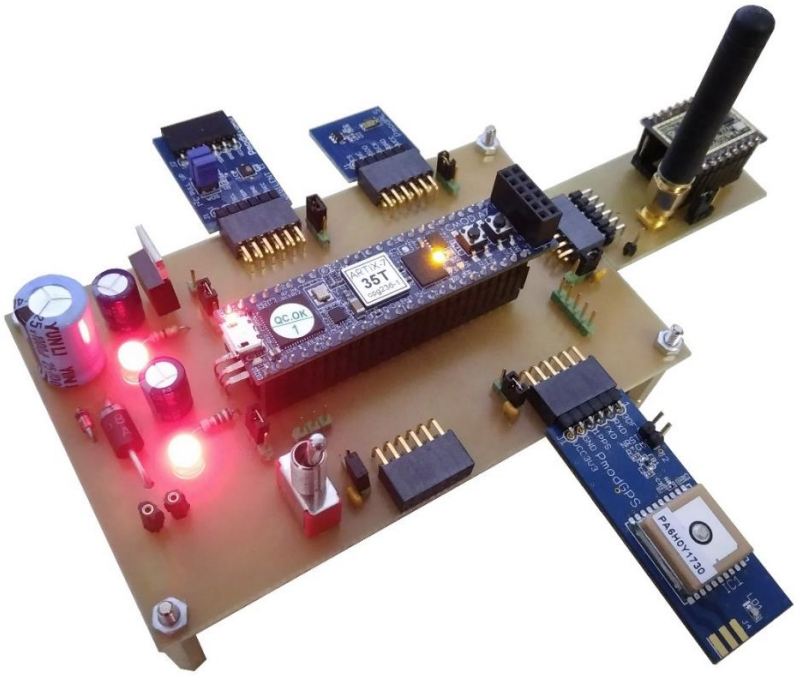


Figure 25 Picture of the Smart Sensor

8.2. Terminal

Here is the data displayed by the terminal, it validates the proper operation of the sensors and each step. Just after "PHASE SEND" we can see the frame sent then the sensor data displayed more legibly (Figure 26).

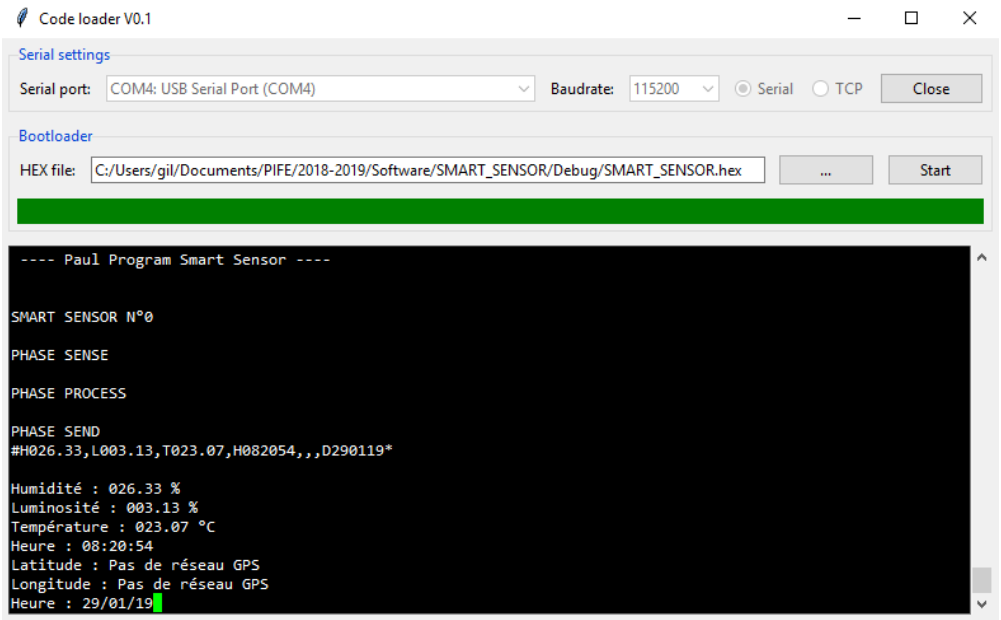


Figure 26 Terminal witch print data sensors

8.3. Test of consumption

Here is the general consumption of the Smart Sensor (Figure 27). We observe that in these diagrams there are 4 periods of 5 seconds. During the beginning of the period (Phase SENSE, Phase PROCESS and Phase SEND), there is an increase of the power up to 650 mW whereas during the SLEEP Phase the average power is 570mW. The last two curves show that the consumption of the sensors is low (≈130mW) compared to the microcontroller implanted in the CMOD A7 board (≈470mW).

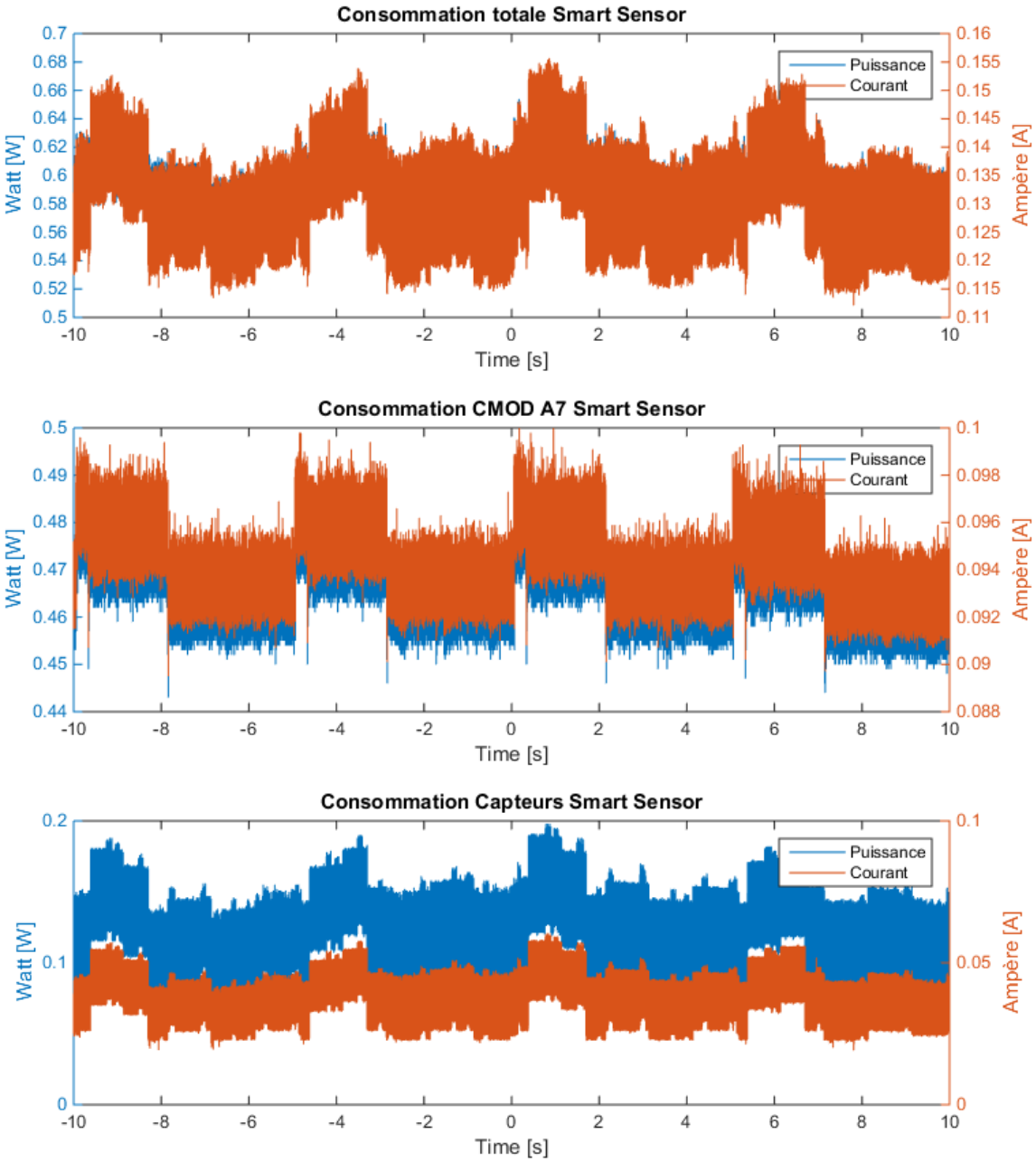


Figure 27 Total consumption, microcontroleur consumption and sensors (« capteur » in french) consumption

To visualize which sensor consumes the most and when, a measurement of each sensor was made. Thanks to the following 4 curves (Figure 28) we clearly see the SENSE Phase which captures the sensor data on the Pmod HYGRO and Pmod ALS. However the Pmod GPS sends data regularly, every 2 seconds, because it emits data without any requests.

The LoRa of the SEND Phase emits about 1.5 seconds at a power of 28mW.

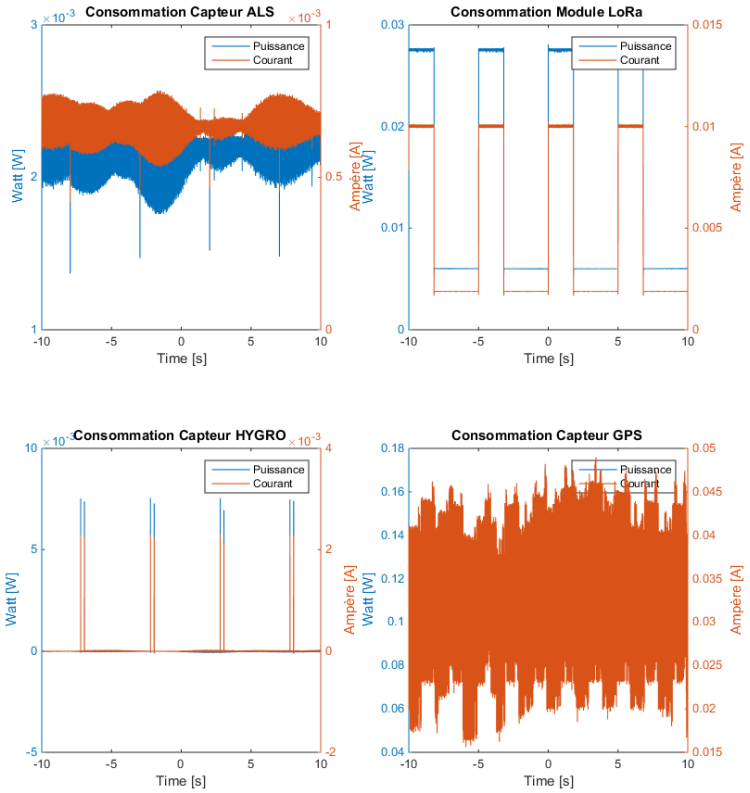


Figure 28 Consumption of each sensors/radio modules

With this chart (Figure 29) with the same scale on each curve, we see that the most important consumption is the CMOD A7. The 4 modules consume very little. In the modules it is the GPS that consumes the most (measurements made without the GPS attached to satellites). Then there is the consumption of LoRa during the sending phase which consume a lot also. The other sensors, HYGRO and ALS, consume very little.

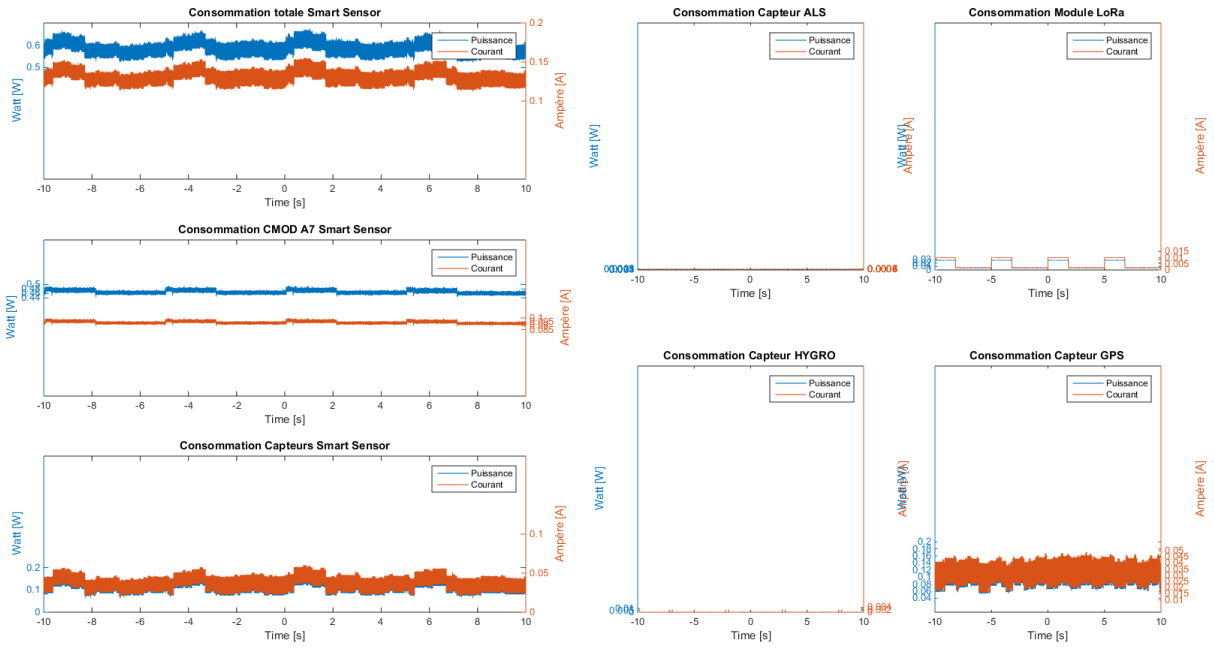


Figure 29 Global view of the consumption

9. Conclusion

The Smart Sensor is now operational, it meets the specifications and was done on time. Consumption tests show a consistent result. The cost of the Smart Sensor is around 150 € by counting the card and the components.

This industrial project of end of study brought me a lot, either personally or professionally. I was able to discover the laboratory research community. I was able to develop myself personally by working on a project composed of hardware and software.

This experience supported and anchored my knowledge learned during my training. In addition to validating my skills, I was able to develop others such as knowing how to set up an IDE or use communication interfaces other than the UART.

10. Reference

[1] https://fr.statista.com/statistiques/584481/internet-des-objets-nombre-d-appareils-connectes-dans-le-monde--2020/
[2] https://www.iea.org/publications/freepublications/publication/MoreData_LessEnergy.pdf
[3] https://www.objetconnecte.net/objets-connectes-energie-101116/
[4] https://store.digilentinc.com/pmod-modules-connectors/
[5] https://fr.farnell.com/
[6] http://www.lirmm.fr/

11. Appendix

State of the art

Etude & Conception d'un Smart Sensor sur FPGA

Projet Industriel de fin d'étude

AUTEUR : PAUL LELOUP

TUTEUR : PASCAL BENOIT

ENCADRANTS : PASCAL BENOIT & GUILLAUME PATRIGEON

2018

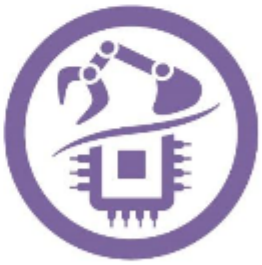


Table des matières

1. Introduction

Le lieu d'étude est le Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM) qui est une unité mixte de recherche dépendant conjointement de l'Université Montpellier (UM) et du Centre National de la Recherche Scientifique (CNRS) [1].

Le projet a lieu dans le cadre d'un projet industriel de fin d'étude qui a pour objectif de mettre l'élève dans la situation d'un chef de projet chargé de trouver des solutions à un problème concret qui lui est posé [2].

L'internet des objets (IOT) est l'extension d'Internet à des choses et à des lieux du monde physique. L'internet des objets connectés représente les échanges d'informations et de données provenant de dispositifs du monde réel avec le réseau Internet.

L'architecture de l'appareil est présentée sur le schéma ci-dessous (Figure 1). Le projet a pour but de créer la carte électronique qui reliera les capteurs, la carte FPGA et les modules de communication ainsi que de les programmer pour les faire fonctionner ensemble. Cela a pour objectif d'aider une thèse sur l'exploration d'architecture de microcontrôleur afin de baisser la consommation d'énergie.

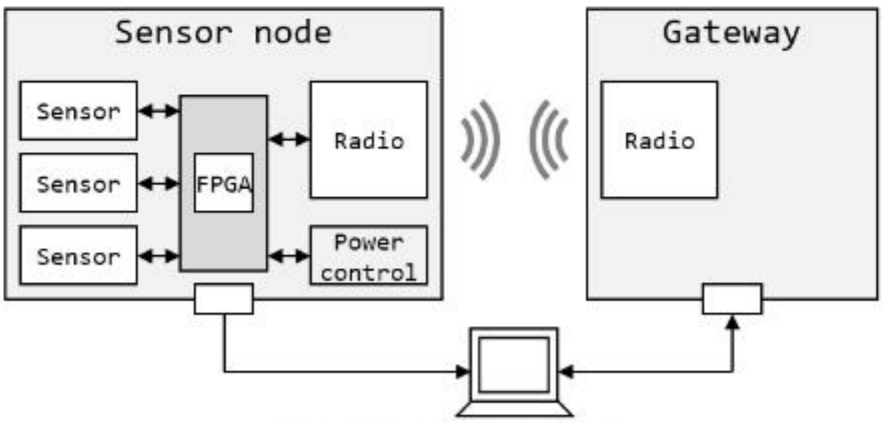
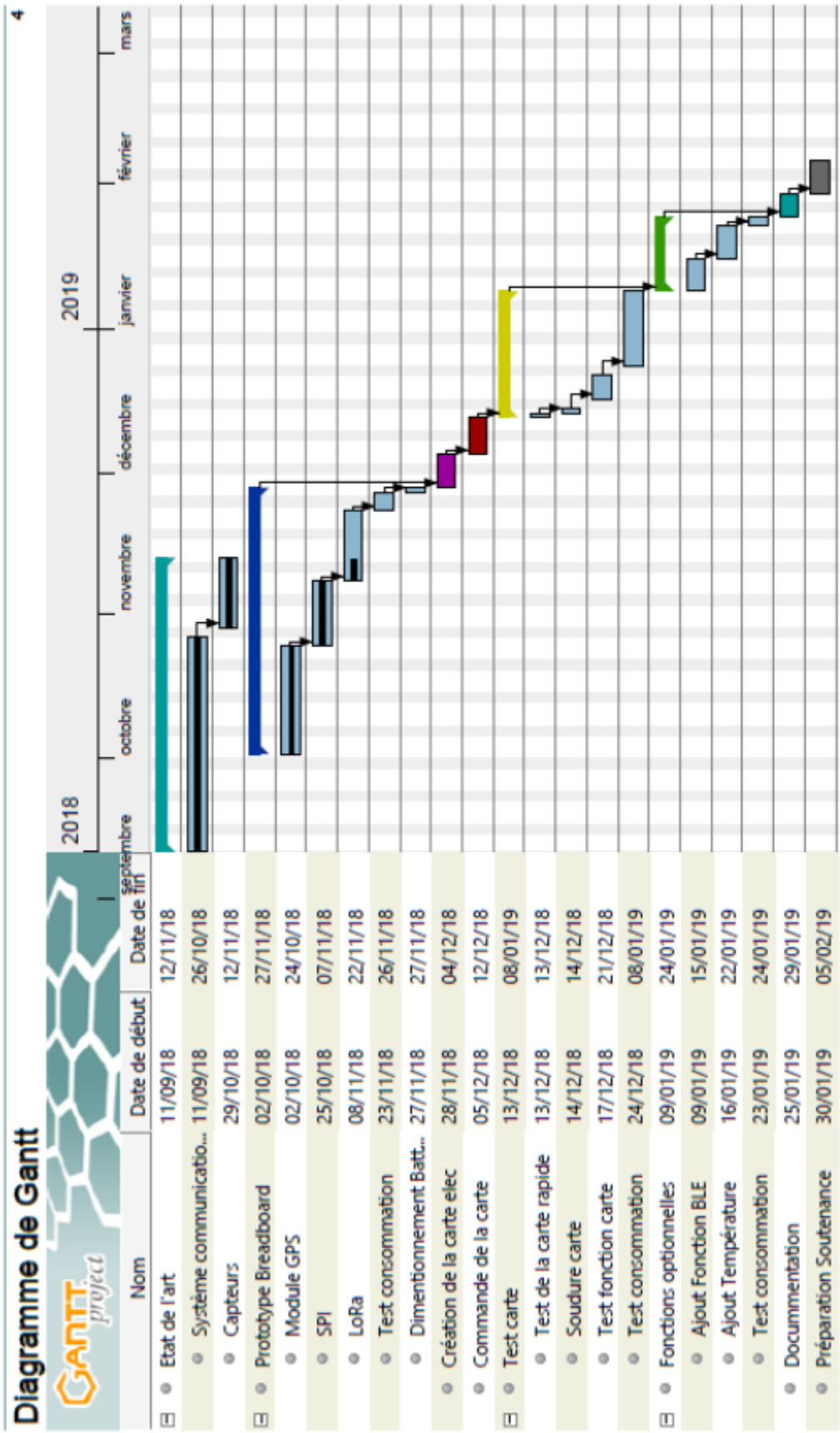


Figure 1 Architecture du Smart Sensor

Gantt

2. Gantt



3. Capteurs

3.1. Définition, types de capteurs

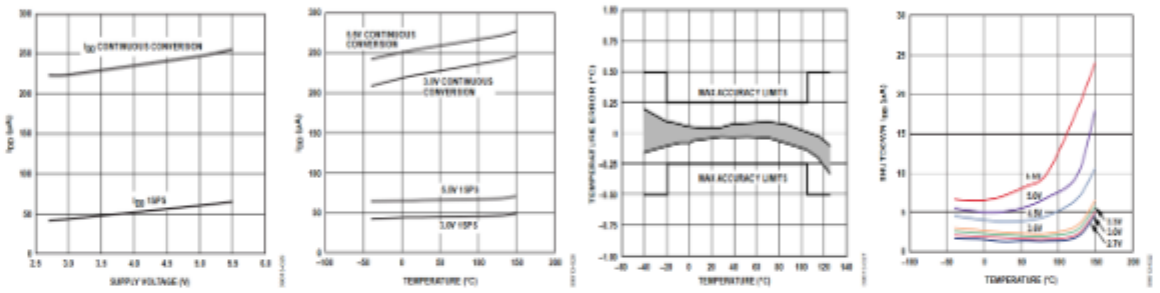
Un capteur est un dispositif qui fournit à partir d'une grandeur physique prélevée, une autre grandeur physique de nature différente (souvent électrique) [3].

Il y a deux types de capteurs différents. Les capteurs actifs sous l'action du mesurande, génère un signal électrique (courant ou tension), on peut l'appeler générateur. Les capteurs passifs sous l'action du mesurande font apparaitre une variation d'impédance. Le capteur passif ne produit pas directement un signal électrique, il sera indispensable d'utiliser un circuit électrique extérieur pour déterminer la valeur du mesurande.

Quelques exemples de capteurs passifs : température, flux optique, déformation, humidité…

Quelques exemples de capteurs actifs : flux optique, force, pression, vitesse…

Le capteur de température utilisé est le PmodTMP2 de Digilent [4] (Figure 3) qui fonctionnera dans les conditions suivantes : Vdd = 3.3V, lecture périodique. Les avantages sont les suivants :



(a) Courant moyen de fonctionnement en fonction de la tension d'alimentation

(b) Courant de fonctionnement en fonction de la température

(c) Erreur en fonction de la température

(d) Courant en mode Sleep en fonction de la température pour différentes tensions d'alimentations

Figure 2 Caractéristique du capteur de température PmodTMP2 de Digilent

Le courant moyen consommé est de 225µA en fonctionnement continu (Figure 2 (a)), il est cependant légèrement variable en fonction de la température (Figure 2 (b)). Le capteur est très précis car il a une erreur maximale de 0.5°C entre -20°C et 105°C (Figure 2 (c)). Le mode Sleep consomme seulement 5µA maximum (Figure 2 (d)) et permet d'économiser de l'énergie ce qui est très important dans notre application [5].

3.2. Analogique / Numérique

Il faut transformer ces données analogiques en données numérique. On va créer des circuits électriques pour recevoir des valeurs numériques ou utiliser, comme dans la plupart des cas, un microcontrôleur. Il existe beaucoup de capteurs sous forme de module qui délivre la donnée numérique. Capteurs utilisés :

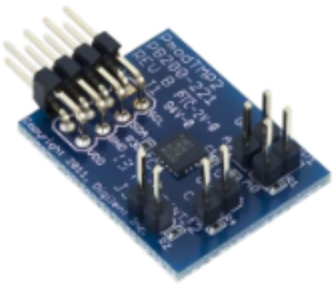
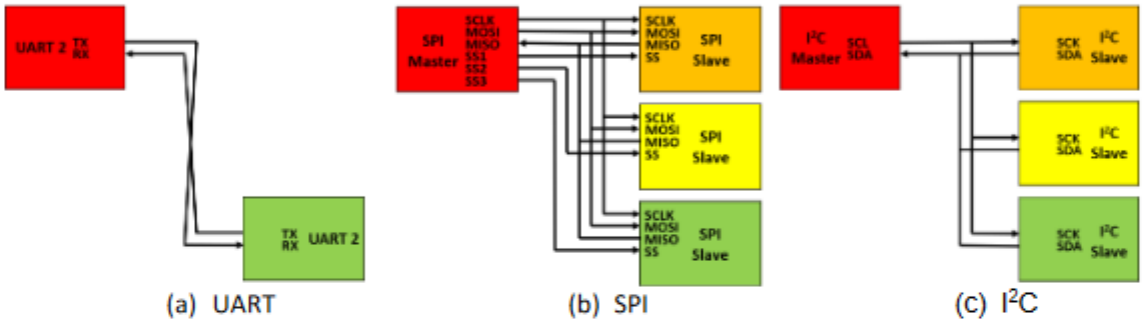


Figure 3 Capteur de température PomdTMP2 de Digilent

Figure 4 Capteur GPS PmodGPS de Digilent

Pour faire l'interface entre le module et le microcontrôleur central, il faut communiquer par des câbles. On va utiliser des interfaces de communications tels que l'UART, l'I²C, le SPI et d'autres encore.



(a) UART (b) SPI (c) I²C

Figure 5 Interfaces de communications

Un UART (Figure 5 (a)), pour Universal Asynchronous Receiver Transmitter, est un émetteur-récepteur asynchrone universel.

Une liaison SPI (Figure 5 (b)) (pour Serial Peripheral Interface) est un bus de données série synchrone. Les circuits communiquent selon un schéma maître-esclaves, où le maître contrôle la communication. Plusieurs esclaves peuvent coexister sur un même bus, dans ce cas, la sélection du destinataire se fait par une ligne dédiée entre le maître et l'esclave appelée Slave Select.

I²C (Figure 5 (c)) est un bus série synchrone bidirectionnel half-duplex, où plusieurs équipements, maîtres ou esclaves, peuvent être connectés au bus.

Les UART, SPI et I²C sont généralement intégrés dans des composants comme des microcontrôleurs. Ils ne sont dans ce cas plus un composant à proprement parler, mais une fonction périphérique du composant central.

4. Communication

4.1. Réseaux sans fil pour l'IOT

Pour l'IOT il est important que le plus d'appareils soient reliées pour avoir plus d'informations et agir de la meilleure manière possible. L'intérêt du réseau sans fil est qu'il est pratique (pas de fil) et que l'installation d'un nouvel appareil sera rapide et simple. Voici quelques réseaux sans fil adapté à l'IOT qui vont être comparés au niveau de la consommation énergétique :

Le **SIGFOX** est une technologie UNB (Ultra Narrow Band) qui opère à 868 MHz avec un débit de 1000 b/s pour une bande passante de 1 KHz et une sensibilité de -140 dBm avec une portée annoncée de 40 km. Les appareils utilisant cette technologie peuvent envoyer 140 messages par jour à une station. C'est une technologie asynchrone, donc les nœuds ne se réveillent pas pour se synchroniser. Il est donc plus simple de calculer la consommation [6].

LoRa est faite pour les applications IOT à longue portée. LoRa Alliance propose une station qui reçoit des paquets depuis un appareil et retransmet les données à un serveur par connexion TCP. Le LoRa utilise principalement de la modulation basée sur le CSS (Chirp Spread Spectrum) qui permet un débit de 0,25 à 11 kb/s avec une bande passante de 7 à 250 KHz. Dans cette comparaison seule LoRa classe A sera pris en compte [6].

Le **Bluetooth Low Energy** est une version du Bluetooth standard. Il est le plus souvent utilisé pour des applications tel que des moniteurs de fréquence cardiaque ou des contrôles à distance de température d'une pièce. L'avantage de cette technologie est qu'elle est très populaire (dans la majorité des smartphones) et que les dernières versions (4.2, 5.0) permettent d'avoir plusieurs nœuds. BLE a une fréquence d'utilisation de 2,4 GHz, le débit maximal est de 2 Mb/s. Sur la dernière version 5.0, on peut utiliser 4 schémas différents qui permettent d'avoir plusieurs débit (125 kb/s, 500 kb/s, 1 Mb/s et 2 Mb/s) pour moins consommer en fonction des applications [6].

La technologie **NB-IoT** (Narrowband Internet of Things) est une technologie de communication sans fil qui fonctionne à 180KHz avec un débit de 250 kb/s. Appliqué à ce jour principalement au secteur de l'industrie, le NB-IoT, porté par la Chine, souhaite conquérir maintenant celui de la Smart Home, du Smart Building et de la Smart City. Les atouts de cette technologie sont nombreux : faible consommation d'énergie, couverture étendue et faible coût de production et d'utilisation [7].

4.2. Energie

L'IOT est intéressant par le fait de son autonomie et de sa source d'énergie sans fil. Les résultats présentés se basent sur un article scientifique : *Comparison of the Device Lifetime in Wireless Networks for the Internet of Things* [6].

Les calculs de la durée de vie des objets connectés sont faits à partir des valeurs des variables de ce schéma (Figure 6).

t_a signifie la période d'application

t_{on} signifie la période d'émission

t_{cl} signifie la période de la synchronisation

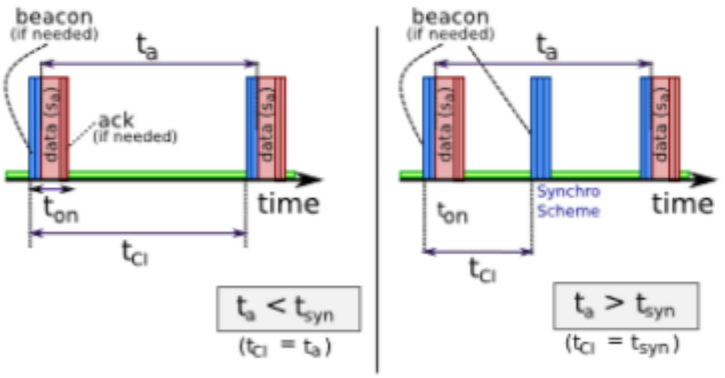


Figure 6 Calcul de consommation

Les auteurs de l'article ont créé l'algorithme qui va traiter ces valeurs et nous donner la consommation.

Voici la consommation de chaque mode (Figure 7) (il ne faut pas prendre en compte uniquement ce tableau car chaque protocole utilisera pendant des temps plus ou moins long ces modes) :

Power	P_Tx	P_Rx	P_Idle	P_Sleep
SIGFOX	147mW	39mW	Ø	4.32uW
LoRa	419.6mW	44mW	Ø	4.32uW
BLE	24.11mW	9.26mW	4.67mW	3.24uW
NB-IOT	Ø	Ø	Ø	Ø

Figure 7 Récapitulatif des consommations

NB-IOT n'a pas de valeur dans le tableau car ce système de communication n'était pas dans l'article étudié et prendre des valeurs d'autres études ne serait pas objectif étant donné que les conditions de test ne seraient pas les mêmes.

Il y a un autre algorithme qui permet de calculer la consommation en fonction de la probabilité de paquet perdu que l'on doit renvoyer.

5. Conclusion

L'enjeu majeur de l'IOT est l'autonomie, d'où la préférence d'utiliser des capteurs actifs (qui génèrent une tension ou un courant). L'intérêt est de faire de l'exploration d'architecture sur FPGA de microcontrôleur pour baisser la consommation.

Voici les résultats de mes recherches sur la consommation des protocoles radio (rappel : sauf NB-IOT) :

- Lorsque les paquets ne sont pas perdus :

Pour une émission tous les jours, toute les 100s, toute les secondes et millisecondes le BLE est le plus efficace en termes d'économie d'énergie.

- Lorsque 20 % des paquets sont perdus :

Le BLE reste plus économe en énergie mais la durée de vie diminue d 'environ 10 %.

- La dérive d'horloge :

Elle ne fait que très peu baisser la durée de vie du LoRa et du SIGFOX à une faible intensité de trafic, contrairement au BLE.

Pour notre application on va donc préférer le LoRa car il consomme finalement assez peu mais a une portée bien plus grande que le BLE. On va préférer utiliser des capteurs Pmod pour faciliter les tests de différents capteurs lorsque la carte sera terminée. Des capteurs tels que la température, la localisation vont être utilisée afin d'avoir des données pertinentes à transmettre.

L'utilisation du LoRa est intéressante pour le LIRMM car elle permettra d'obtenir des données sur cette technologie récente.

LoRa Pmod V2.0

Pmod designed by Paul LELOUP

Additional features

- Integrated RF-LoRa-868 of RF Solutions
- A Pmod Connector for plug and play (using FPGA card for prototype)
- An SMA connector for the antenna
- Small card dimensions: 27.9mm x 50.9 mm
- 4 connectors for measure current and tension of the modules
- 5 connectors for use optional PINS of the RF-LoRa-868

RF-LoRa-868 Features

- Up to 16KM Range
- Integrated LoRa Modem Semtech SX1272
- Highly Efficient Integral Impedance Matching Network
- Provides Full Functionality of the RFIC:
- 157 dB maximum link budget
- +20 dBm at 100mW constant RF output vs. V supply
- -14 dBm high efficiency PA
- Built in RF switch
- High sensitivity: down to -130 dBm
- Bullet-proof front end: IIP3 = -12.5 dBm
- 89 dB blocking immunity
- Programmable bit rate up to 300kbps
- Low RX current of 10 mA, 100nA register retention
- FSK, GFSK, MSK, GMSK, LoRaTM and OOK modulation
- Built-in bit synchronizer for clock recovery
- Preamble detection
- 127 dB Dynamic Range RSSI
- Automatic RF Sense and CAD with ultra-fast AFC
- Packet engine up to 256 bytes with CRC
- Built-in temperature sensor and low battery indicator

Application

- Prototype with FPGA
- Smart Sensor
- Home Automation
- RF Alarms
- Sensor networks
- Long Range Telemetry
- Meter Reading
- Irrigation Systems
- Wireless Applications

Part Numbers

Pmod connector

PIN	Definition	Direction	Function
1	RESET	In	Reset Trigger Input
2	DIO0	In/Out	Digital I/O software configured
3	TX_SWITCH	In	Enable TX RF Path Active High
4	RX_SWITCH	In	Enable RX RF Path Active High
5	GND	-	Ground connection
6	VCC	In	Power connection
7	SCLK	In	SPI Serial Clock Input
8	NSEL	In	Device Select Active Low
9	SDI	In	SPI Serial Data Input
10	SDO	Out	SPI Serial Data Output
11	GND	-	Ground connection
12	VCC	In	Power connection

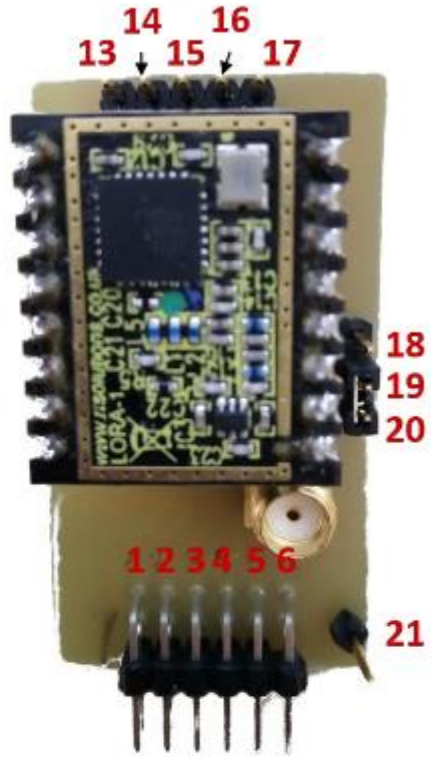
5 Pins connector

PIN	Definition	Direction	Function
13	DIO3	In/Out	Digital I/O software configured
14	DIO4	In/Out	Digital I/O software configured
15	DIO5	In/Out	Digital I/O software configured
16	DIO1	In/Out	Digital I/O software configured
17	DIO2	In/Out	Digital I/O software configured

4 Pins connector

PIN	Definition	Direction	Function
18	VCC Module	-	Pin for tension measure
19	VCC Module	-	Pin for current measure
20	VCC Pmod	-	Pin for current measure
21	GND	-	Pin for tension measure

Pin Description



Using the Pmod

First you can plug the RF-LoRa-868 like this in the Pmod card:

Then for turn ON the alimentation of the Pmod, you must put a bridge between the Pin 19 and 20. Now there is 3.3V to the module.

It's highly recommended to put an SMA antenna (868 MHz).

Go on your keyboards, you can use the Pmod!

Schematic

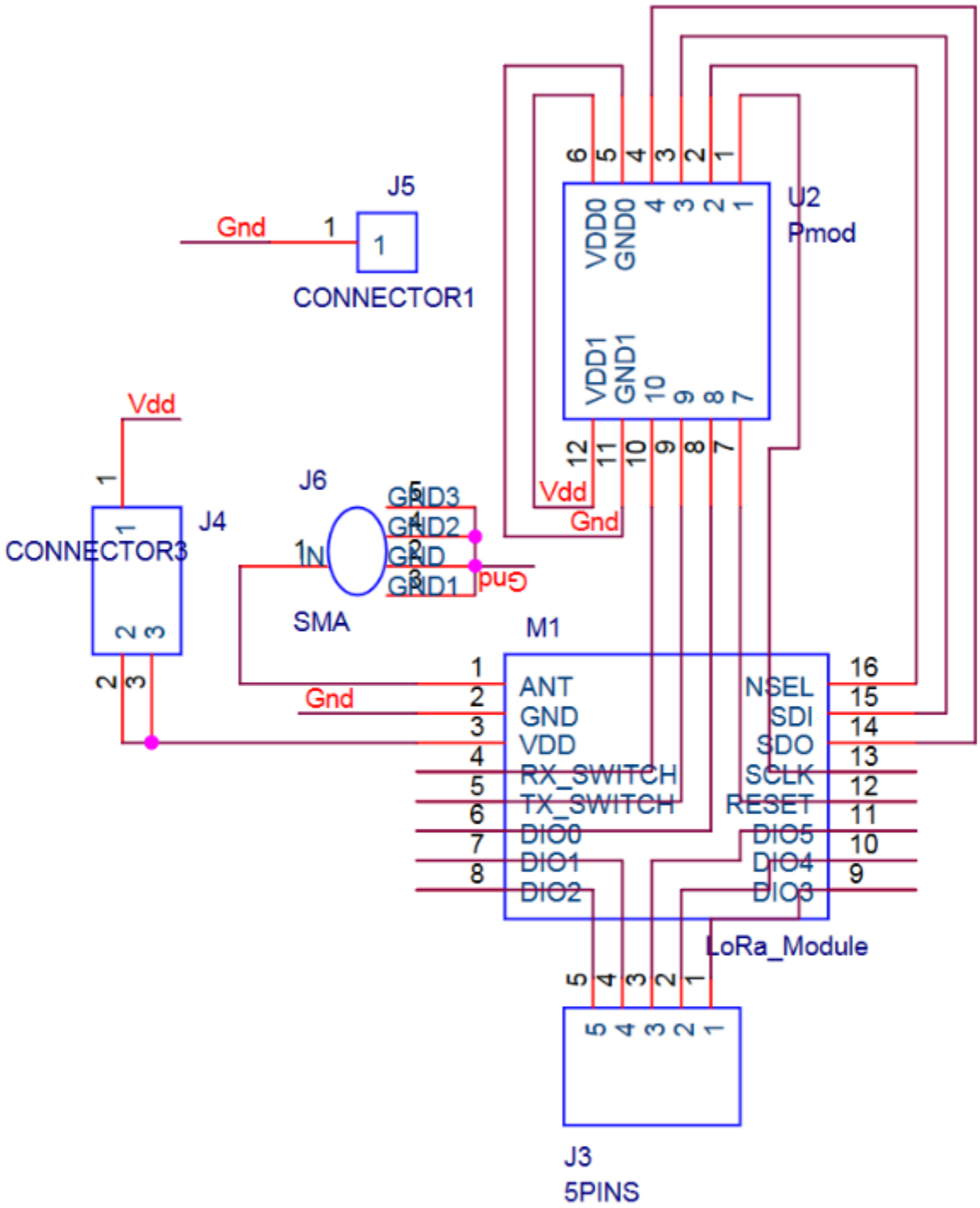


Figure 1 Schematic of Pmod v2.0

Board

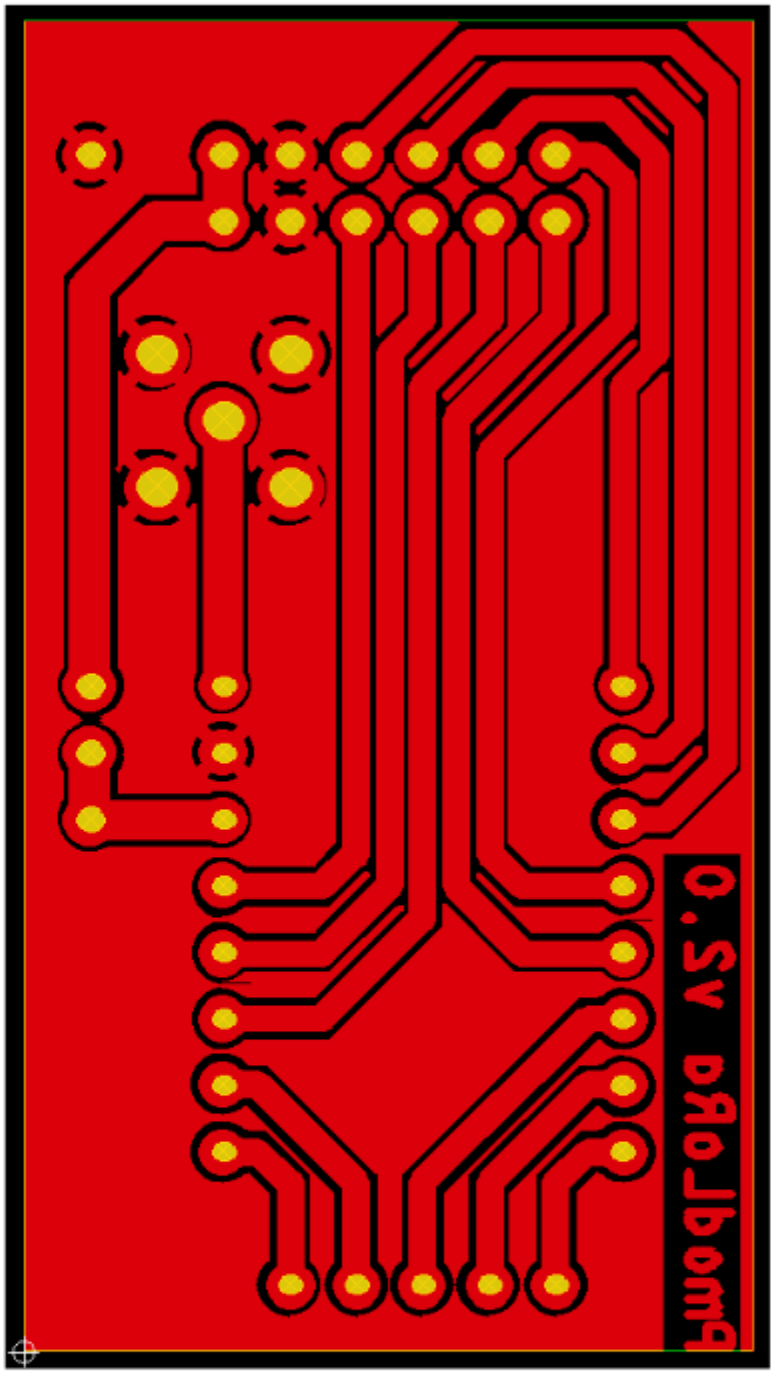


Figure 2 Bottom view Board of Pmod v2.0

References

- Datasheet RF-LORA-868
  http://www.farnell.com/datasheets/2162976.pdf?_ga=2.254649036.578254260.1548165701-413366784.1537950013
- Datasheet SX1272/73
  https://www.semtech.com/uploads/documents/sx1272.pdf

Tutoriel de prise en main du CmodA7 FPGA

Etude & Conception d'un Smart Sensor sur FPGA

AUTEUR : PAUL LELOUP

TUTEUR : PASCAL BENOIT

THESE DE : GUILLAUME PATRIGEON

2019

Table des matières

1. Prérequis

1.1. Matériel

Pour le matériel vous aurez besoin de :

- Un ordinateur
- Un FPGA CmodA7 de DIGILENT
- Un module Pmod GPS de DIGILENT
- Un câble USB-microUSB

1.2. Logiciel

Au niveau du logiciel vous aurez besoin de :

- Un IDE, Eclipse CDT :
  - Lien vers le site : https://www.eclipse.org/cdt/downloads.php
  - Téléchargement d'Eclipse version 2018.2 : https://www.eclipse.org/downloads/download.php?file=/oomph/epp/2018-12/R/eclipse-inst-win64.exe

- Le compilateur gcc-arm-none-eabi :
  - Lien vers le site : https://developer.arm.com/open-source/gnu-toolchain/gnu-rm/downloads
    Télécharger la version « 7-2017-q4-major » et cliquer sur « window zip »
    L'extraire et le déplacer dans un dossier à ne pas supprimer.

- Python 3 :
  - Lien vers le site : https://www.python.org/downloads/
  - Téléchargement de Python 3.7.2 : https://www.python.org/ftp/python/3.7.2/python-3.7.2.exe

- Java JRE :
  - Lien vers le téléchargement https://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html
    Accepter les droits
    Aller dans « Java SE Runtime Environment 8u201 » et cliquer sur la version « x64 » en « .exe »

- La librairie du cortex M0 est présente dans l'archive du tutoriel
- Les fichiers sources sont présents dans l'archive du tutoriel
- Les fichiers *linker.ld* et *startup.asm* sont présent dans l'archive du tutoriel
- L'outil Codeloader V0.1 en python est présent dans l'archive du tutoriel
- Le datasheet du CmodA7 pour avoir les PINS est présent dans l'archive du tutoriel

2. Tutoriel

2.1. Mettre en place l'IDE, le Bootload

Installer JAVA

Double cliquer sur le .exe

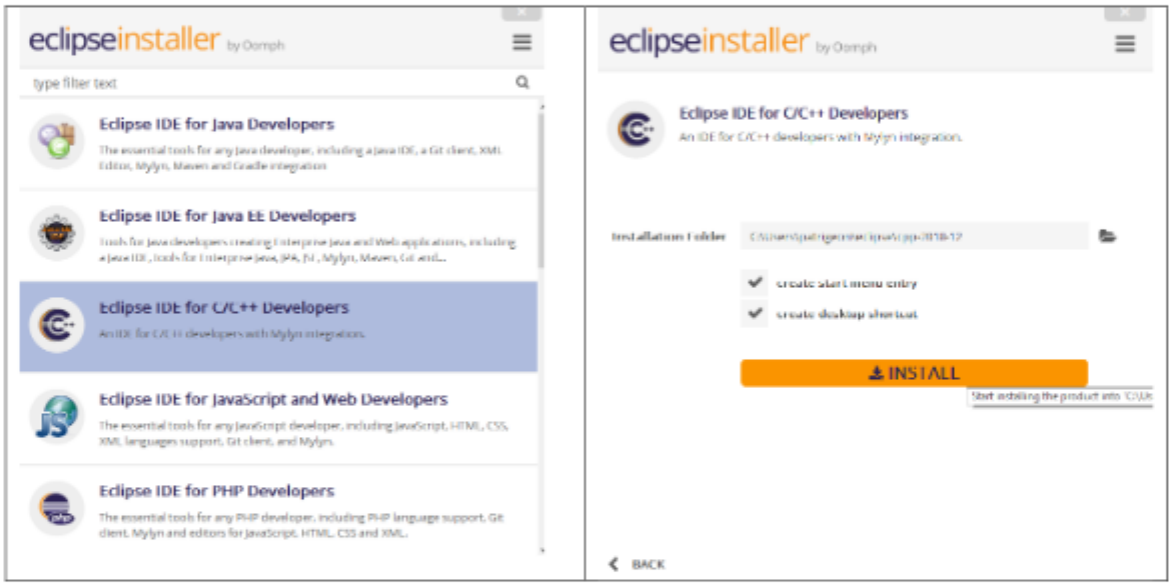
Cliquer sur installer

Cliquer sur fermer

Java est maintenant installé

Installation IDE Eclipse

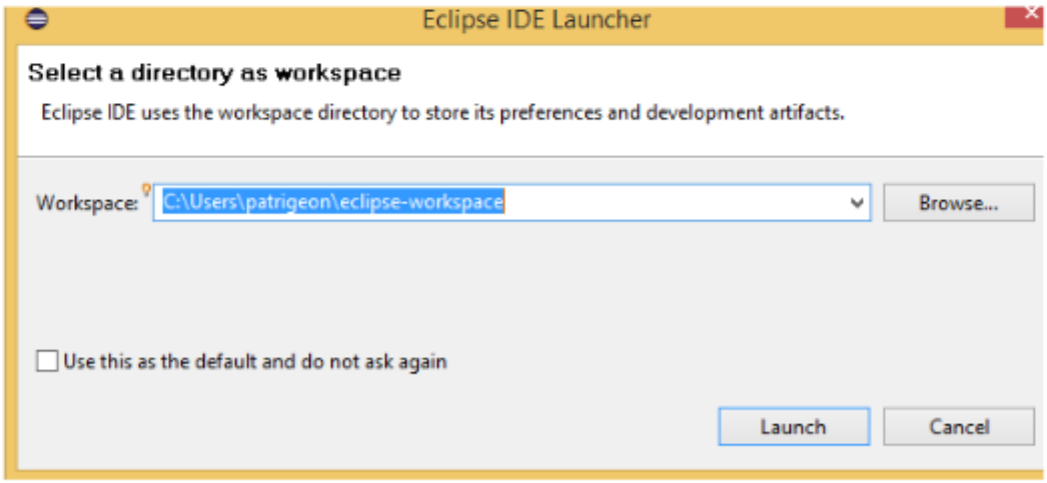
Double cliquer sur le .exe



Accepter toutes les conditions

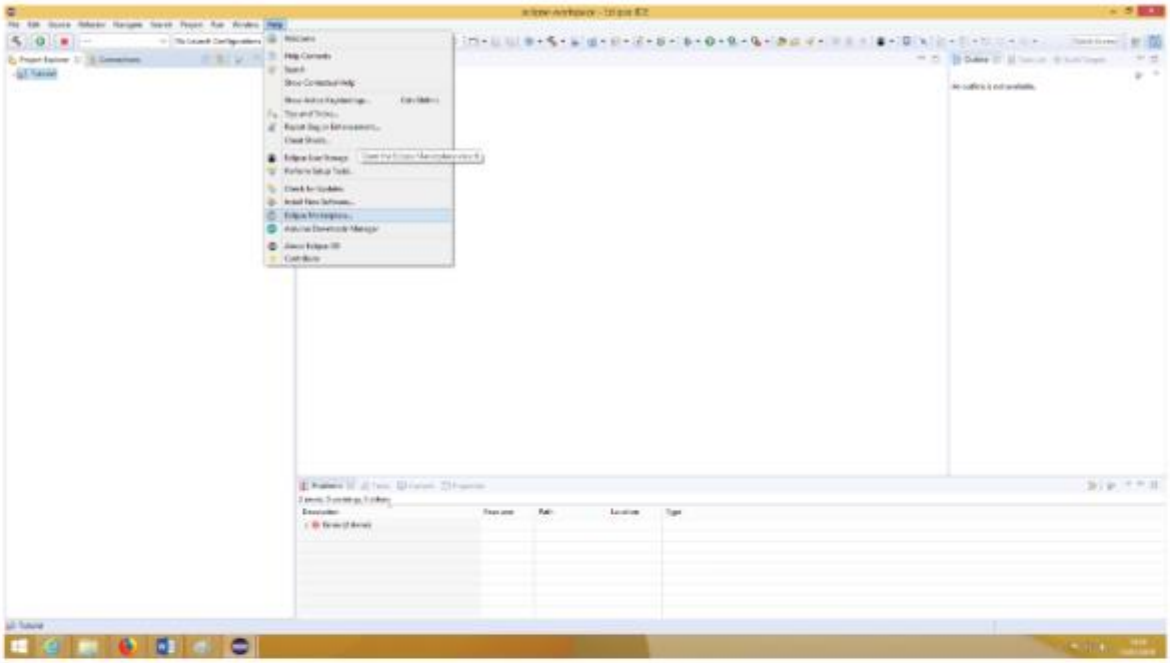
Définir le dossier où tous les projets seront enregistrés :

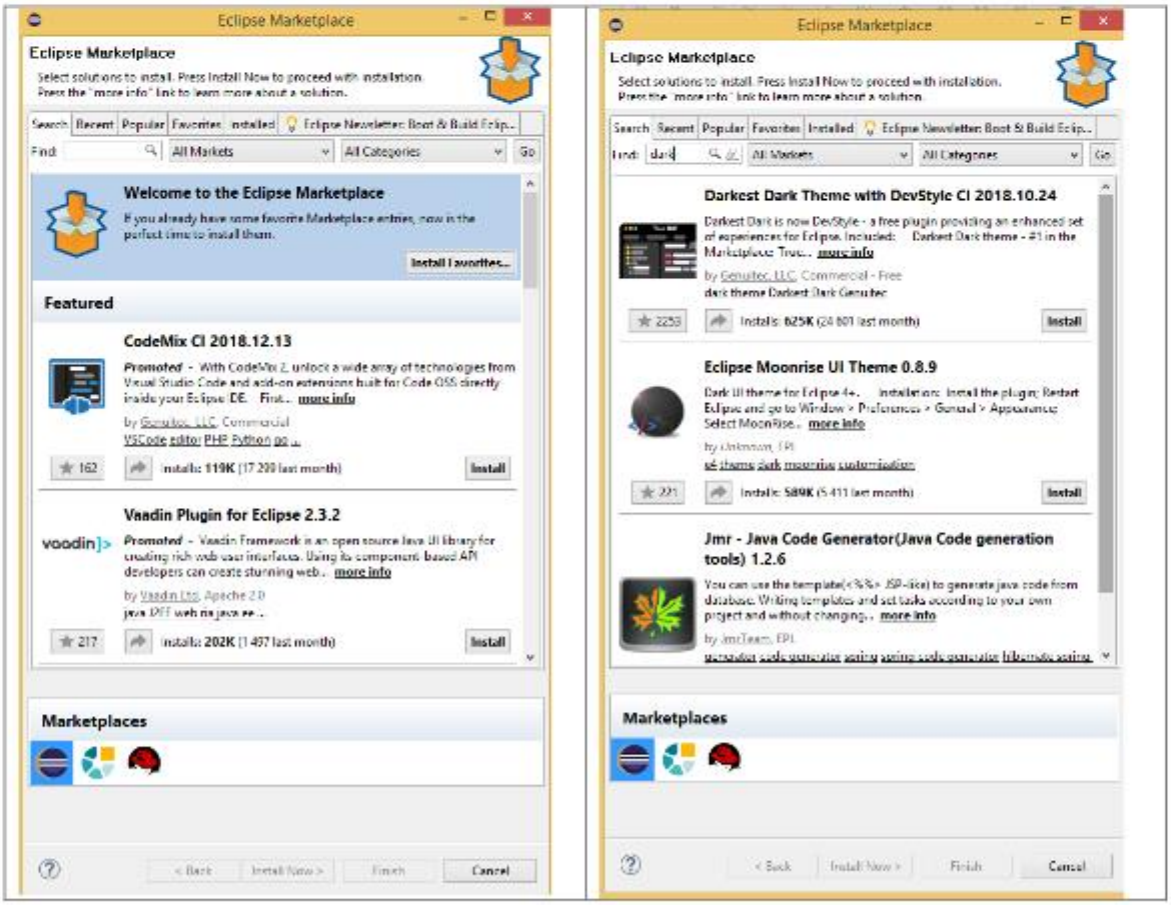
(Bien conserver le chemin qui mène aux projets)



Pour le Dark theme ;

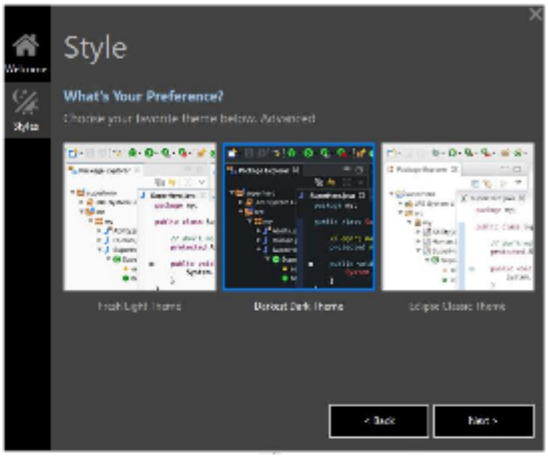
Help/marketplace





Confirmer les conditions

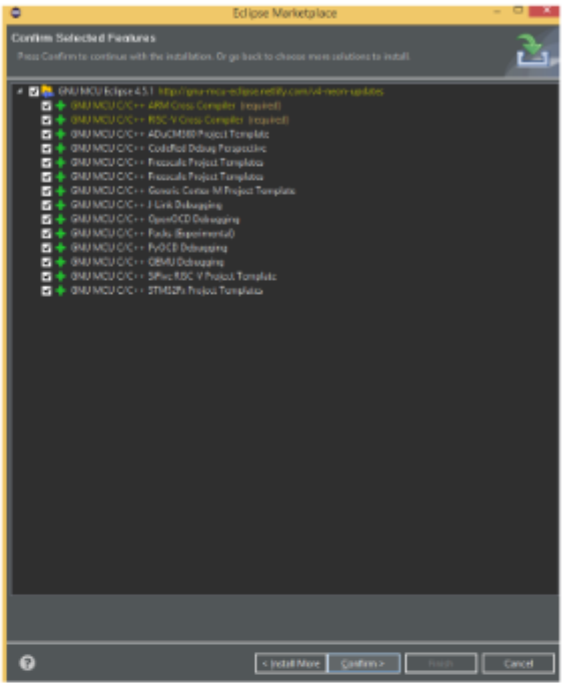
Restart



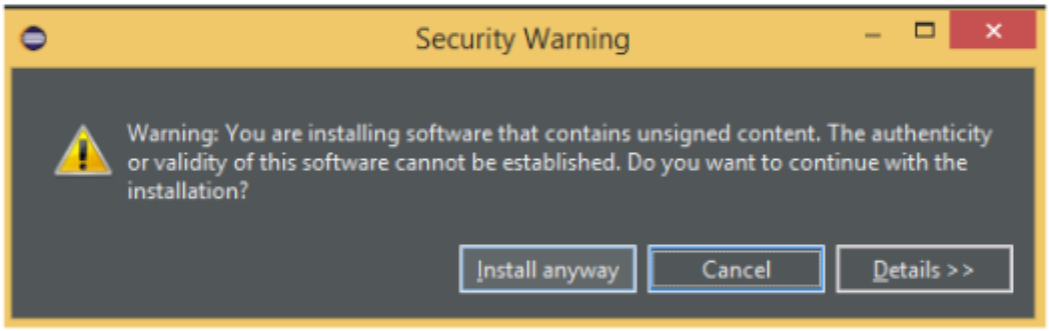
Installer GNU ARM ;

Help/marketplace

Confirmer les conditions



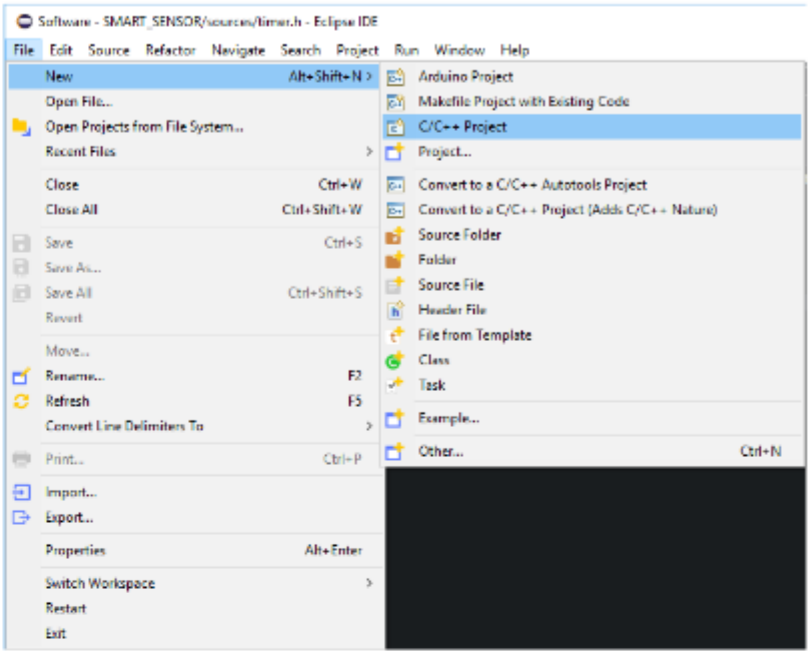
Install anyway



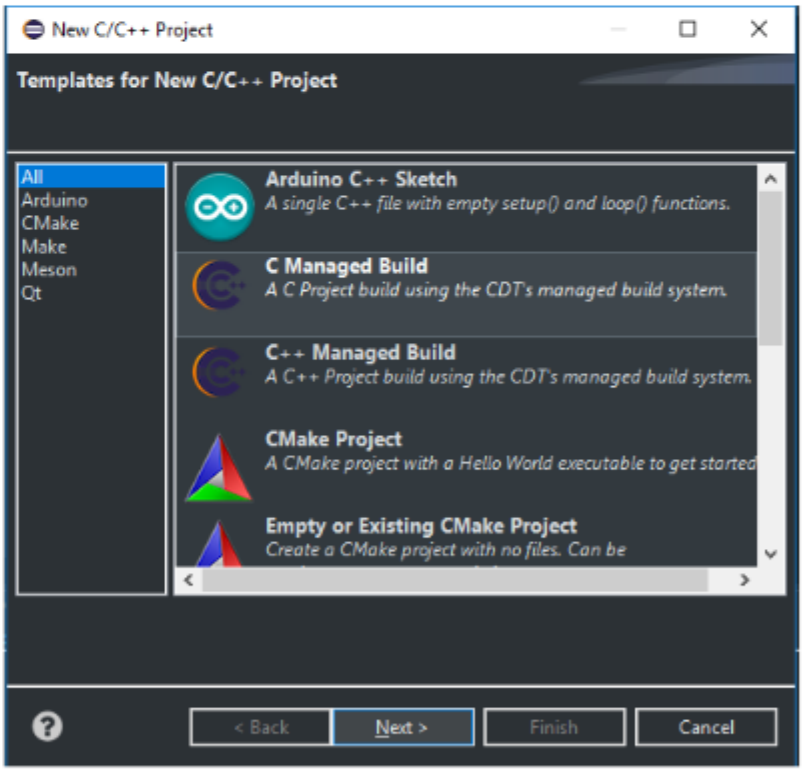
Restart

Paramètres du projet (IDE Eclipse)

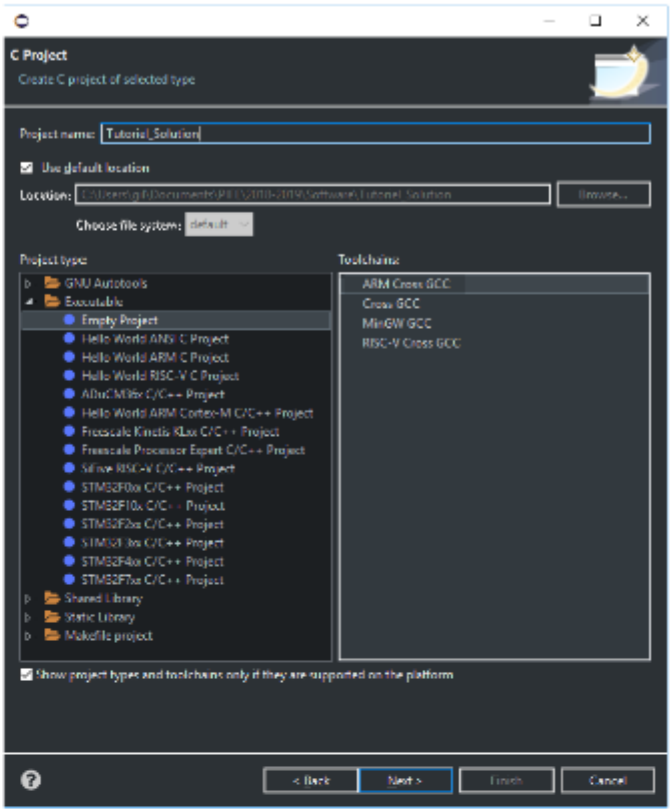
Faire un nouveau projet C/C++



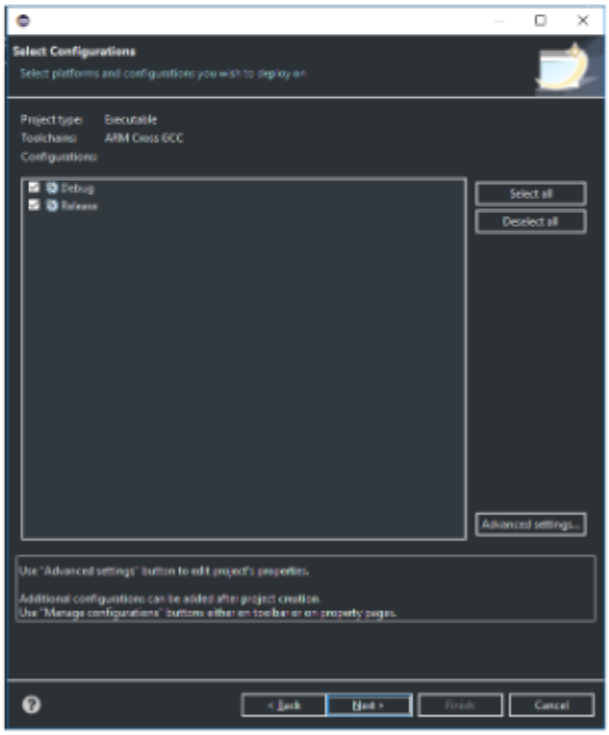
Sélectionner « C Managed Build »



Choisir le nom du projet, selectionner ARM Cross GCC et cliquer sur « Next »

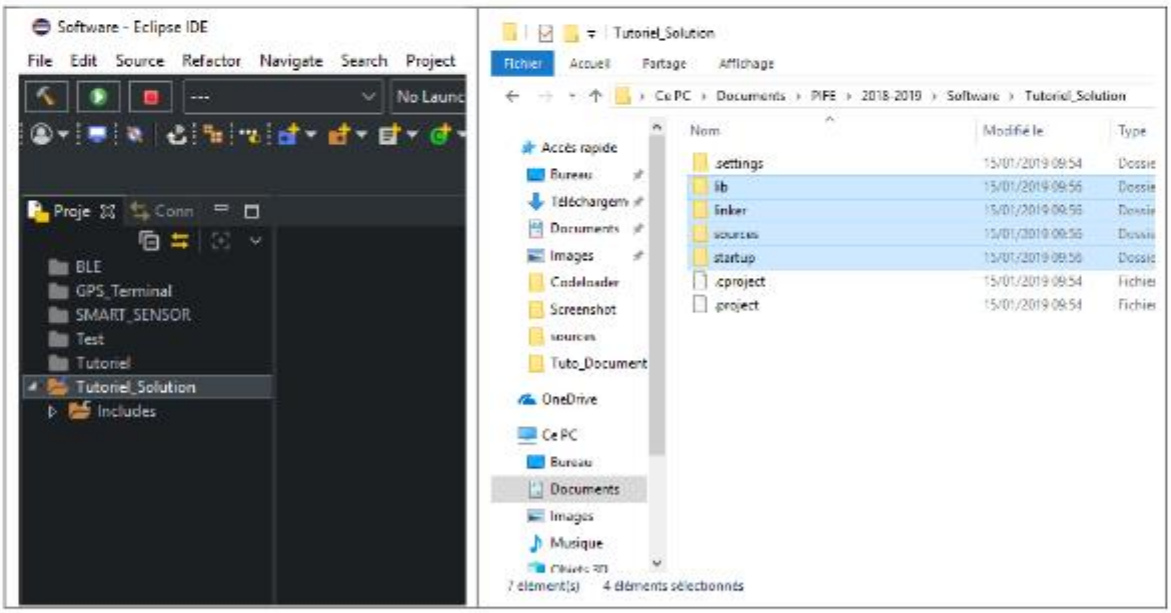


Puis cliquer encore sur « Next »

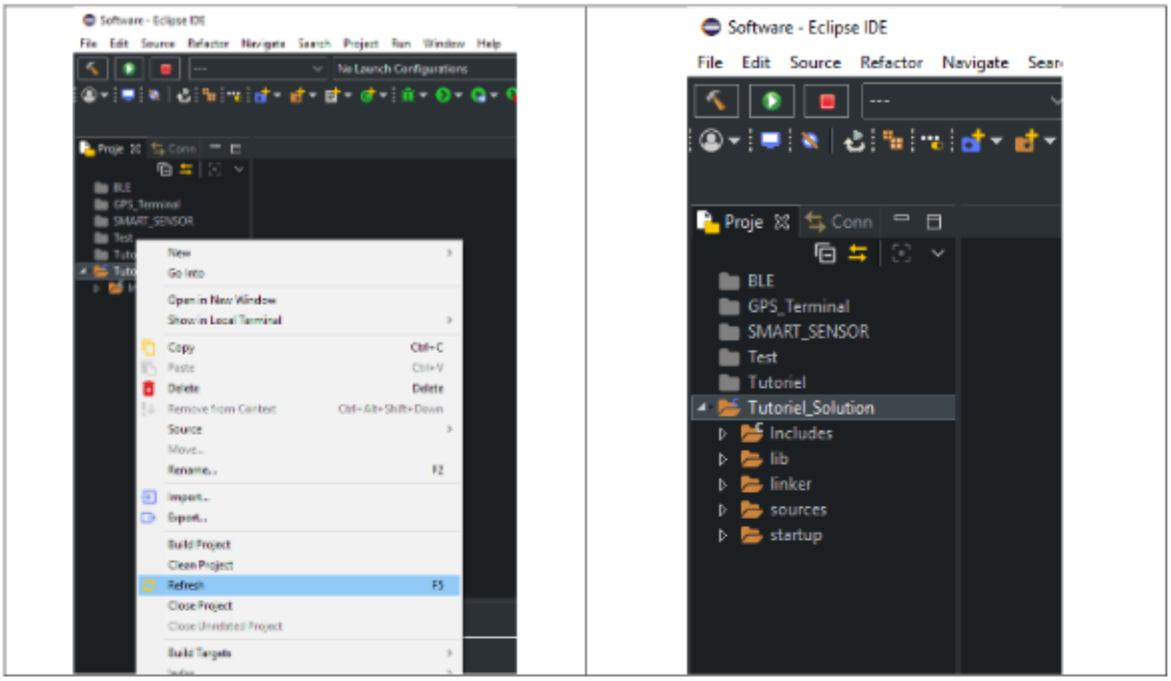


Pour le GNU ARM Cross Toolchain, sélectionner le path de gcc-arm-none-eabi-7-2017-q4-major-win32 préalablement téléchargé.

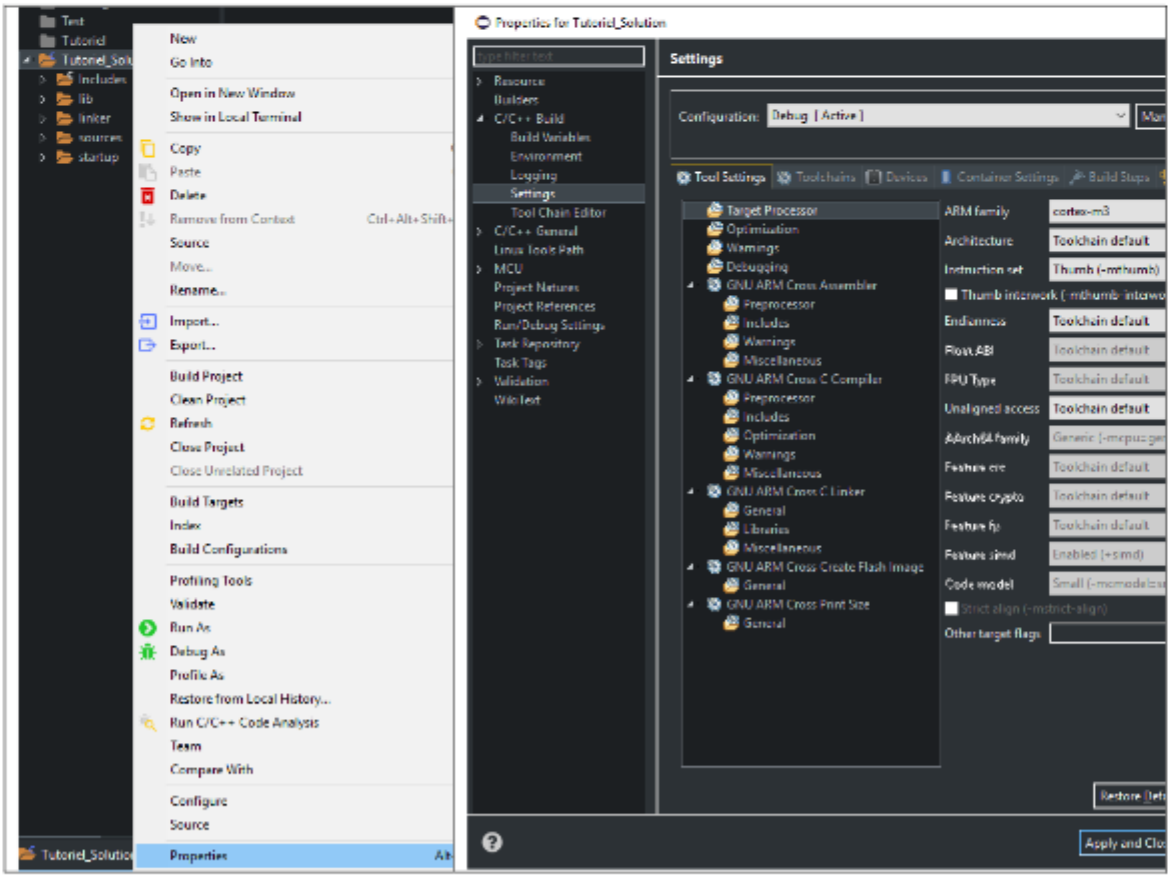
Le dossier compressé « Tutoriel_vX.X » qui vous a été envoyé par mail contient le dossier « Software ». A l'intérieur de celui-ci , vous trouverez 4 dossiers : les fichiers sources, la librairie, le linker et le startup. Ensuite copier/coller ces dossiers dans le dossier de votre projet



Clic droit sur le projet dans Eclipse et cliquer sur Refresh.



Dans l'IDE, selectionner le projet et aller dans les Properties :



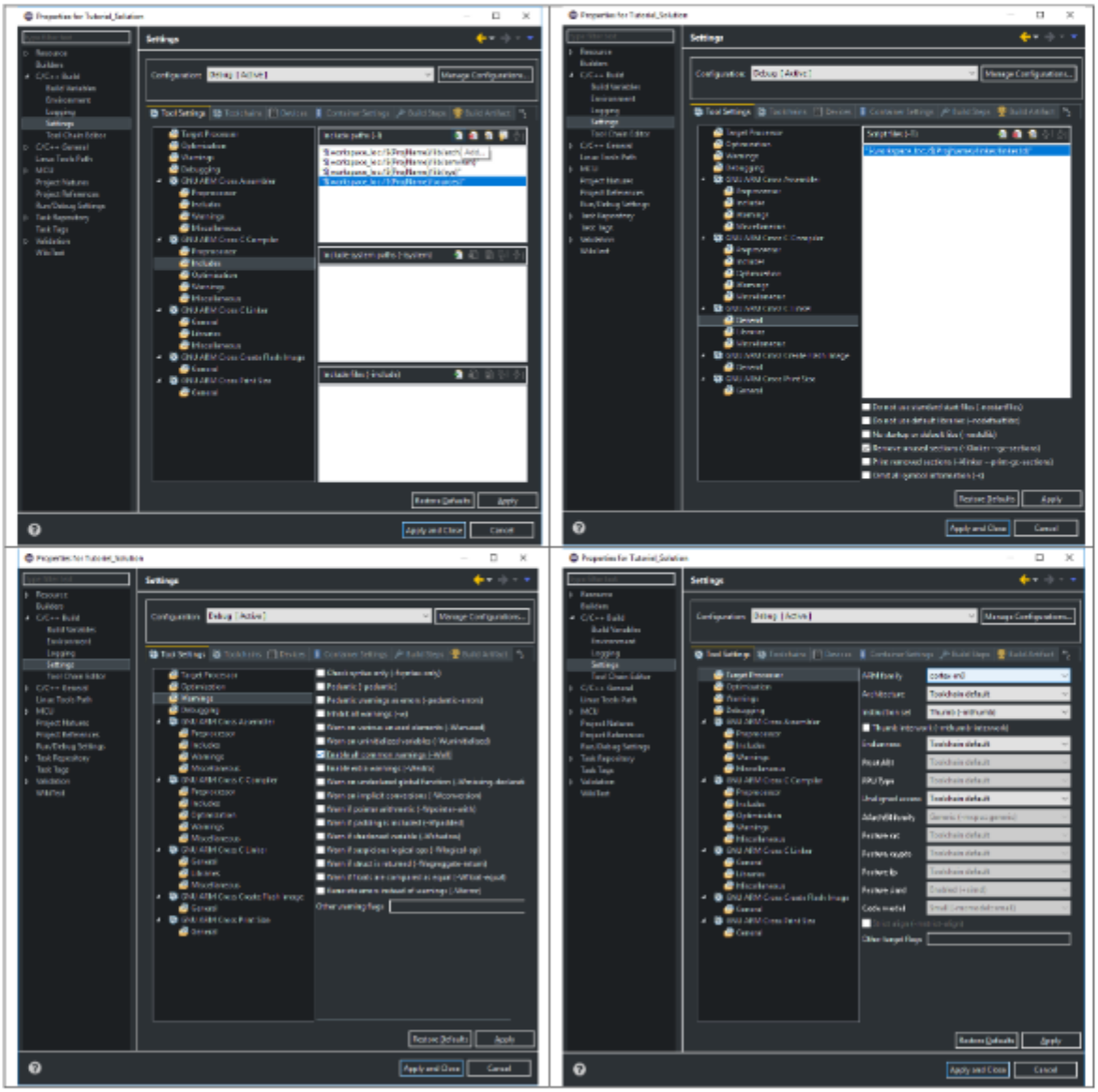
- Properties/C/C++ Build/Settings/Tool Settings/GNU ARM Cross C Compiler/Includes/Include paths
- Ajouter le dossier source et les 3 dossier librairies.

- Properties/C/C++ Build/Settings/Tool Settings/GNU ARM Cross C Linker/General
- Ajouter le fichier linker

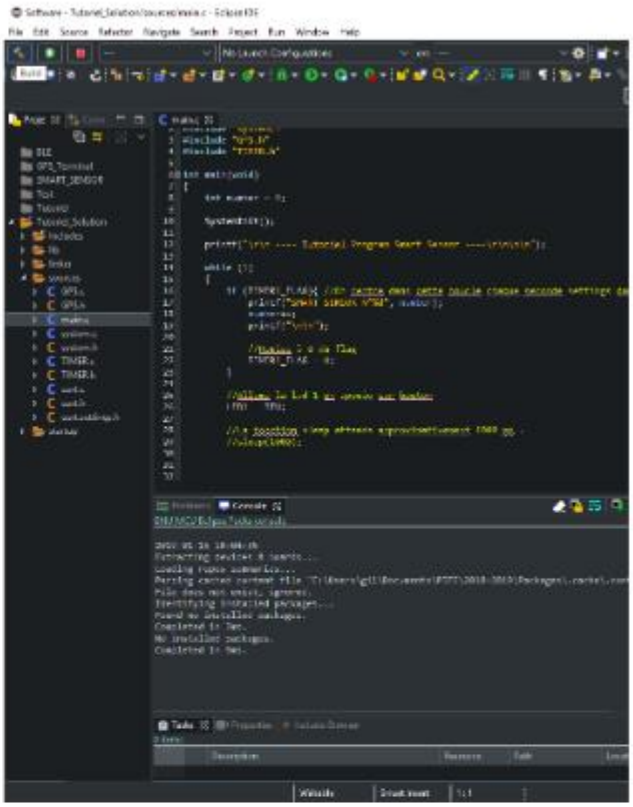
- Properties/C/C++ Build/Settings/Tool Settings/Warnings
- Cocher la case « Enable all common warnings (-Wall)

- Properties/C/C++ Build/Settings/Tool Settings/Target Processor
- Selectionner le cortex-m0 dans ARM family

Voici le résultat attendu :



Compiler le projet en cliquant sur le marteau en haut à gauche :



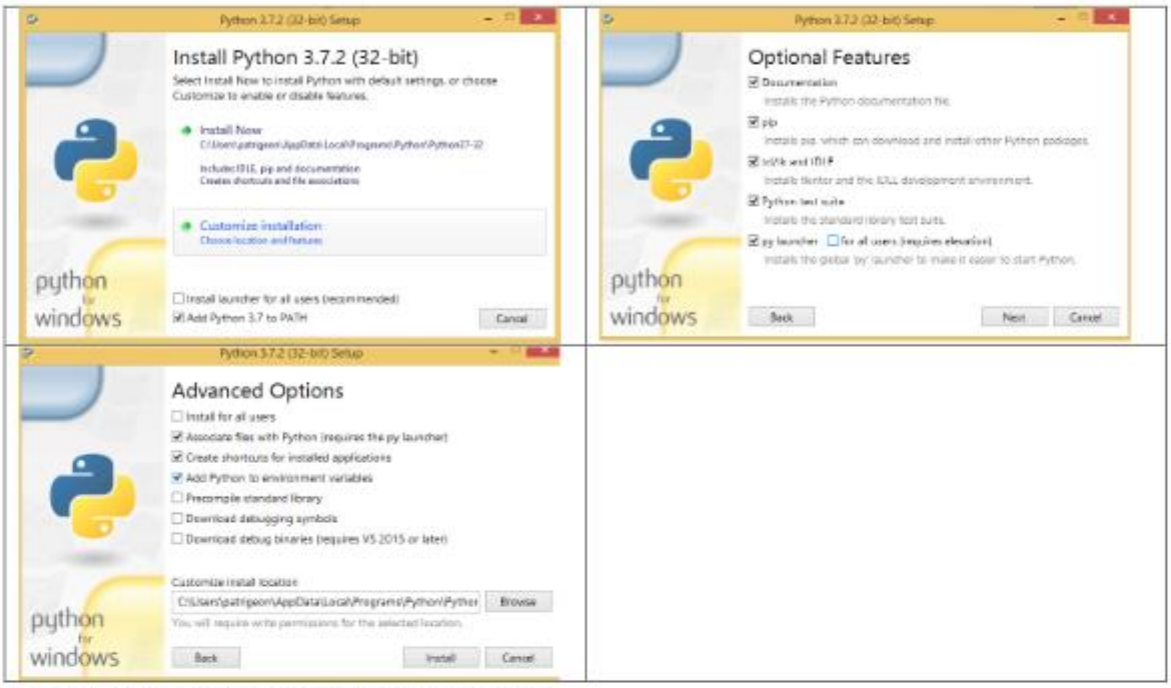
Si tout a été fait sans erreur, l'IDE doit compiler le projet et créer un dossier Debug qui contiendra le programme à télécharger dans le FPGA qui sera le fichier .hex.


```
'Invoking: GNU ARM Cross Create Flash Image'
arm-none-eabi-objcopy -O ihex "Tutoriel_Solution.elf"  "Tutoriel_Solution.hex"
'Finished building: Tutoriel_Solution.hex'
' '
'Invoking: GNU ARM Cross Print Size'
arm-none-eabi-size --format=berkeley "Tutoriel_Solution.elf"
   text    data     bss     dec     hex filename
   3540    1080    1148    5768    1688 Tutoriel_Solution.elf
'Finished building: Tutoriel_Solution.siz'
' '

10:14:23 Build Finished. 0 errors, 0 warnings. (took 5s.779ms)
```

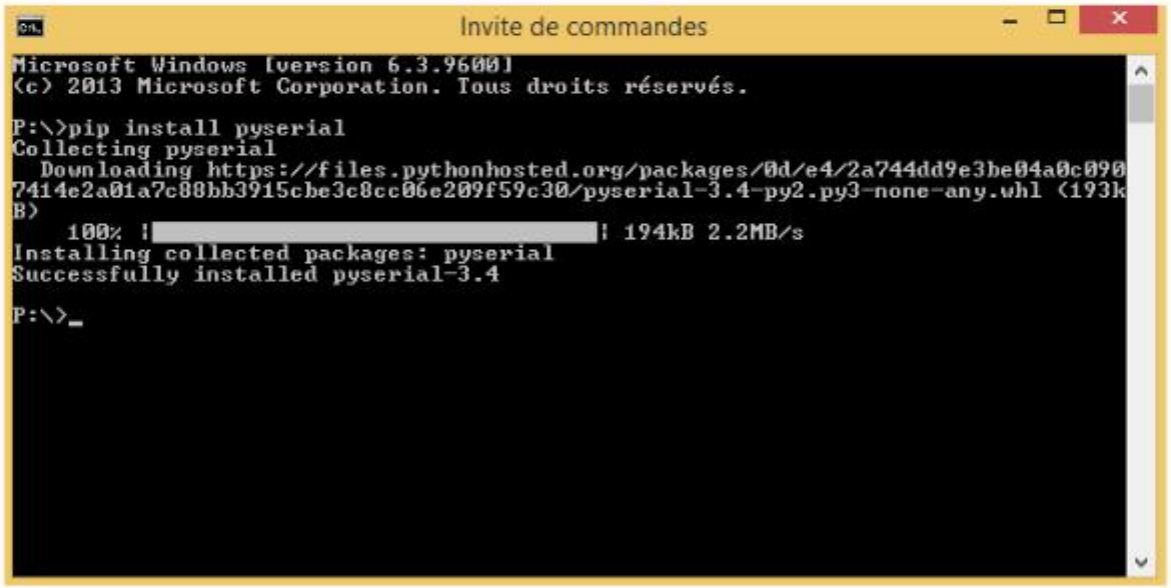
Installation du Codeloader

Lancer Python .exe



Taper cmd dans la barre de recherche windows

Taper : pip install pyserial et taper enter

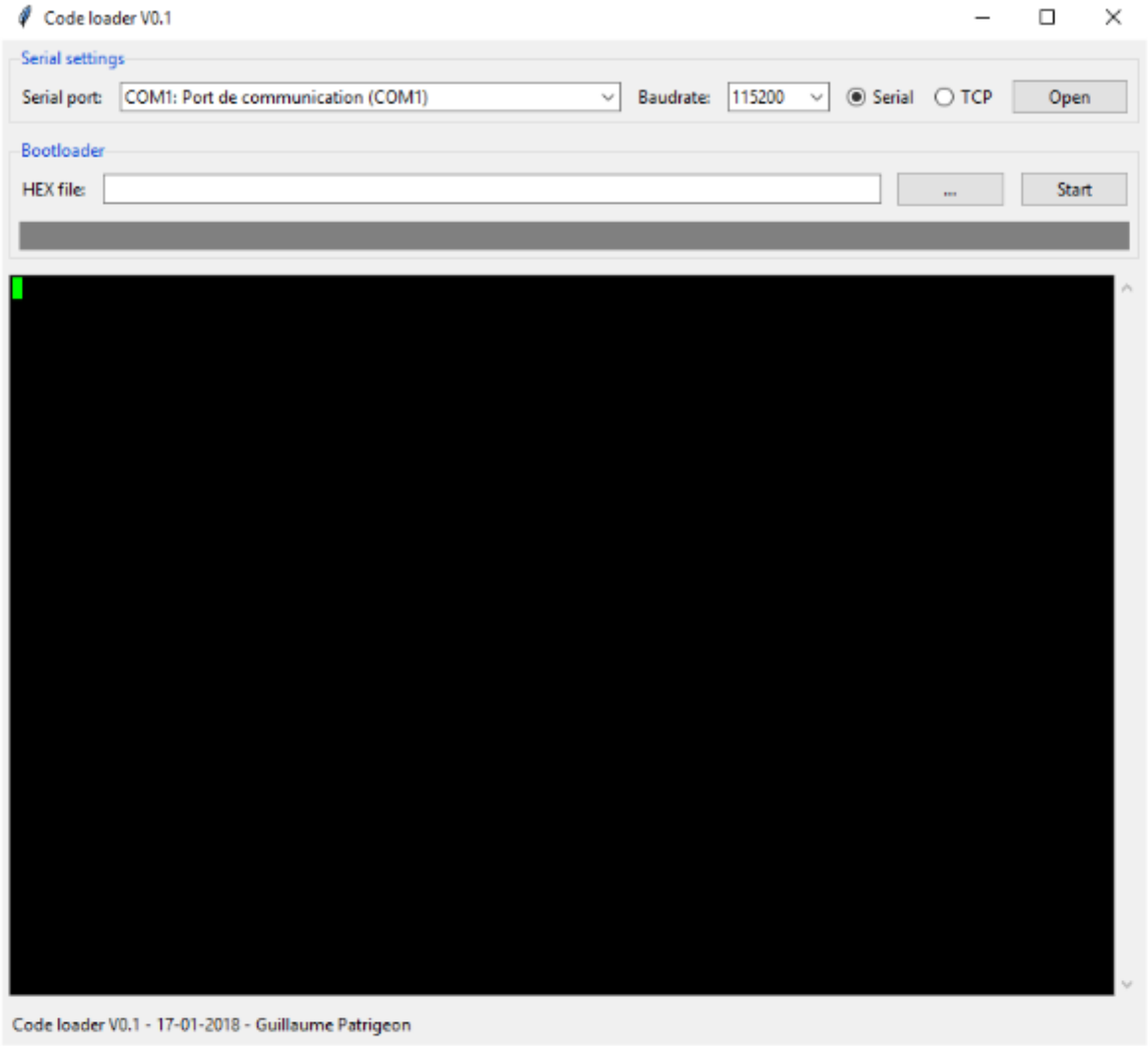


Lancer main.pyw pour verifier la bonne installation !!!!

Il se trouve dans le dossier compressé du tutoriel dans le dossier Codeloader. Décompresser le dossier préalablement.

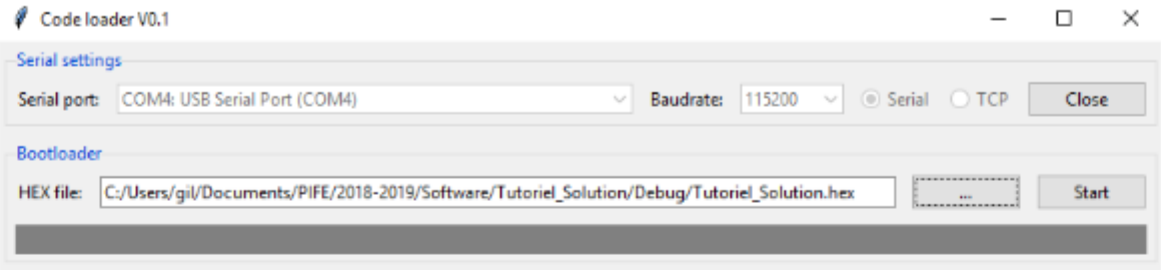
Utilisation du Codeloader

Double cliquez sur le main.pyw pour ouvrir le codeloader, cette fenêtre devra apparaitre :

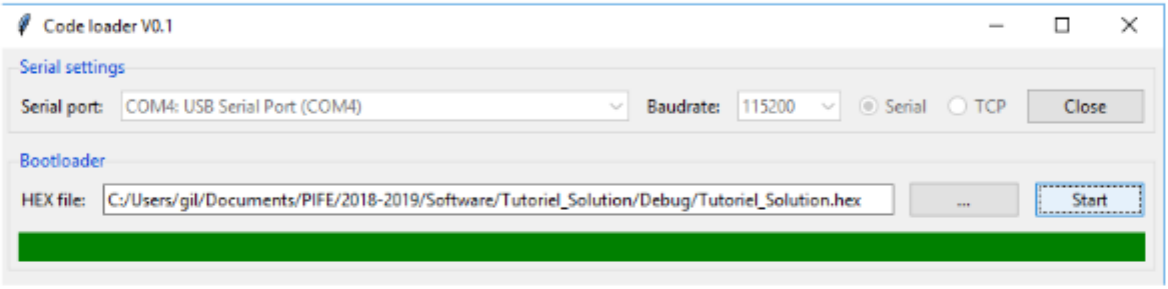


Une fois le Codeloader lancé, sélectionner le port COM où est branché le CmodA7, mettre un Baudrate de 115200 en liaison Serial puis ouvrir la liaison série.

Sélectionner ensuite le fichier hex file et télécharger le code.

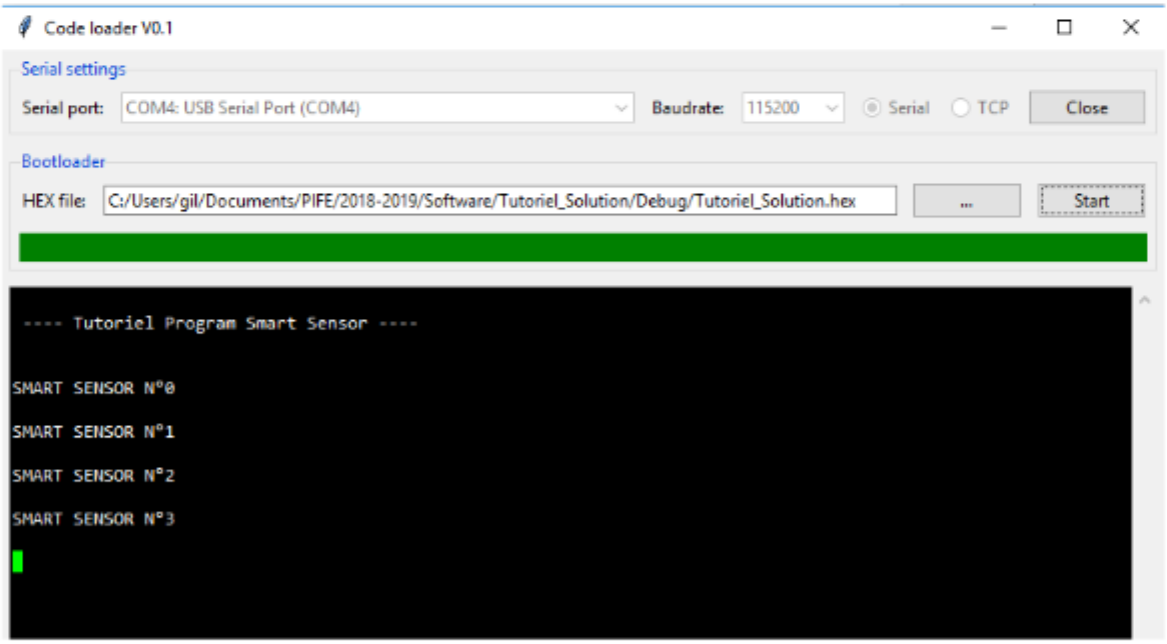


Pour télécharger le code, il faut appuyer sur start et rapidement appuyer sur le reset du CmodA7 qui est le bouton le plus proche de la connectique Pmod.



Si le téléchargement est réussi, la barre de chargement est verte, si le reset n'as pas été appuyé à temps, la barre deviendra rouge.

Si tout a été fait correctement, des informations s'affichent à l'écran et lorsque l'on appuie le bouton BTN1, la LED1 s'allume.



Faire clignoter la LED 2

Adapter le code pour faire clignoter la LED2 avec une période de 2 secondes

2.2. Utiliser l'interface UART avec le module GPS

Initialiser l'UART2

Il faut aller dans le system.c pour initialiser les PIN en entrée/sortie de l'UART2 puis définir l'utilisation de l'UART2 dans uart.setting.h

Les afficher

Renvoyer les données de l'UART2 sur l'UART1 pour l'afficher à l'écran.

Les stocker

Renvoyer les données de l'UART2 dans un tableau, puis afficher le tableau à l'écran.

2.3. Mettre sous un bon format

Machine d'état

Faire une fonction dans GPS.c pour n'afficher qu'une seule trame. Celle qui est intéressantes pour le projet (date, heure, latitude, longitude). Voir la signification des trames sur le datasheet du module.

Mettre cette trame dans un tableau et afficher ce tableau.

Ensuite afficher l'heure, la date de manière lisible pour quelqu'un qui ne connait pas la structure de la trame.

Créer une trame

Créer une trame avec les informations intéressantes dans le but de l'envoyer plus tard par un module LoRa.

3. Pour approfondir

3.1. Utiliser les autres interfaces du CmodA7, SPI et I2C

3.2. Envoyer la trame par LoRa