

|  |
| --- |
| Camera Open MV  -  PCB |

|  |  |  |
| --- | --- | --- |
| Ulster University |  |  |
|  |  | *Fannie Bichemin – Intern*  *Student at ENSEA (Ecole Nationale Supérieure de l’Electronique et de ses Applications)* |



Table of contents

[Introduction 3](#_Toc170129051)

[Open MV Camera - Schematic update 4](#_Toc170129052)

[Removed components 6](#_Toc170129053)

[Added Components 7](#_Toc170129054)

[Explanation of each component 8](#_Toc170129055)

[Converter 8](#_Toc170129056)

[Regulator 8](#_Toc170129057)

[Programming Pins 9](#_Toc170129058)

[Bluetooth module & Antenna connector 9](#_Toc170129059)

[1. Description 9](#_Toc170129060)

[2. Bluetooth module problems 10](#_Toc170129061)

[3. Antenna connector 12](#_Toc170129062)

[Microcontroller 13](#_Toc170129063)

[Image sensor 13](#_Toc170129064)

[LEDs 14](#_Toc170129065)

[USB-C connector and Lipo charger 14](#_Toc170129066)

[Routing 15](#_Toc170129067)

[The different versions of the board 15](#_Toc170129068)

[Appendices 17](#_Toc170129069)

[Converter documentation 17](#_Toc170129070)

[Regulator documentation 18](#_Toc170129071)

[Flasher Programmer documentation 18](#_Toc170129072)

[Bluetooth module documentation 19](#_Toc170129073)

[STM32 documentation 19](#_Toc170129074)

[Image sensor documentation 20](#_Toc170129075)

[Battery charge documentation 20](#_Toc170129076)

[Bill Of Materials 21](#_Toc170129077)

# Introduction

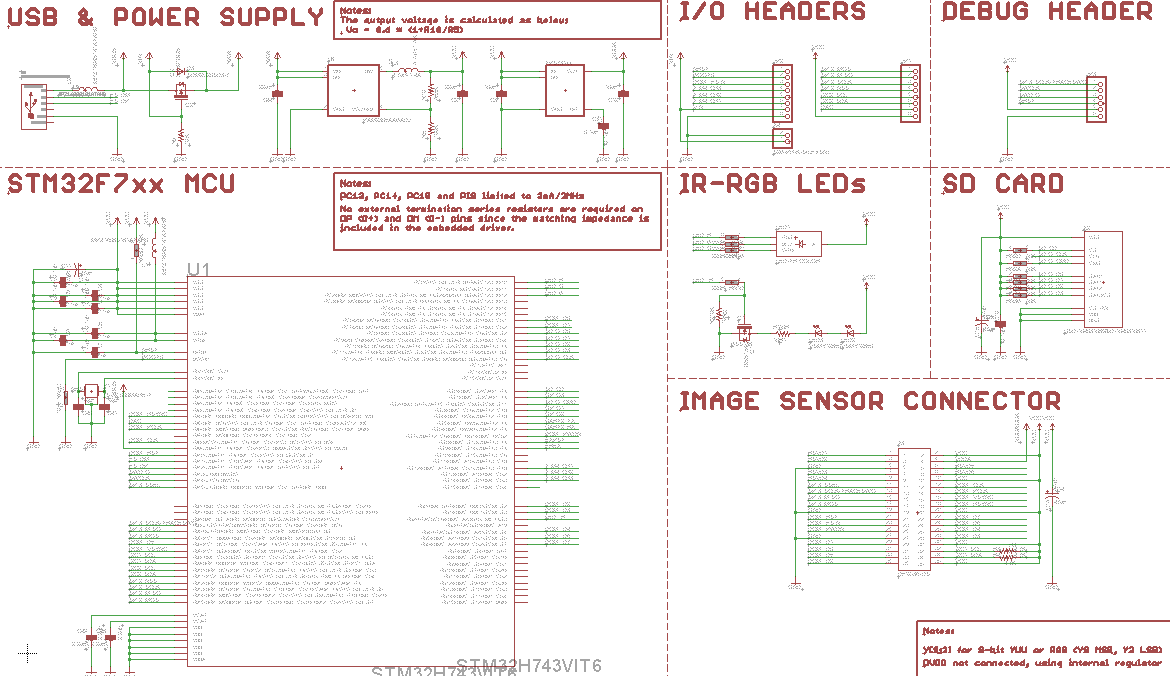
OpenMv camera (H7 R2 ) is a camera which can be used thanks to machine vision. This device is an open-source and anyone can contribute to the OpenMv camera GitHub : <https://github.com/openmv/openmv.git>

The aim is to create a PCB to control this camera and to broadcast data thanks to a Bluetooth module. It will be used for a medical project : analyze blood or saliva. The aim is to notify patients if they need to go to the hospital through a mobile app.

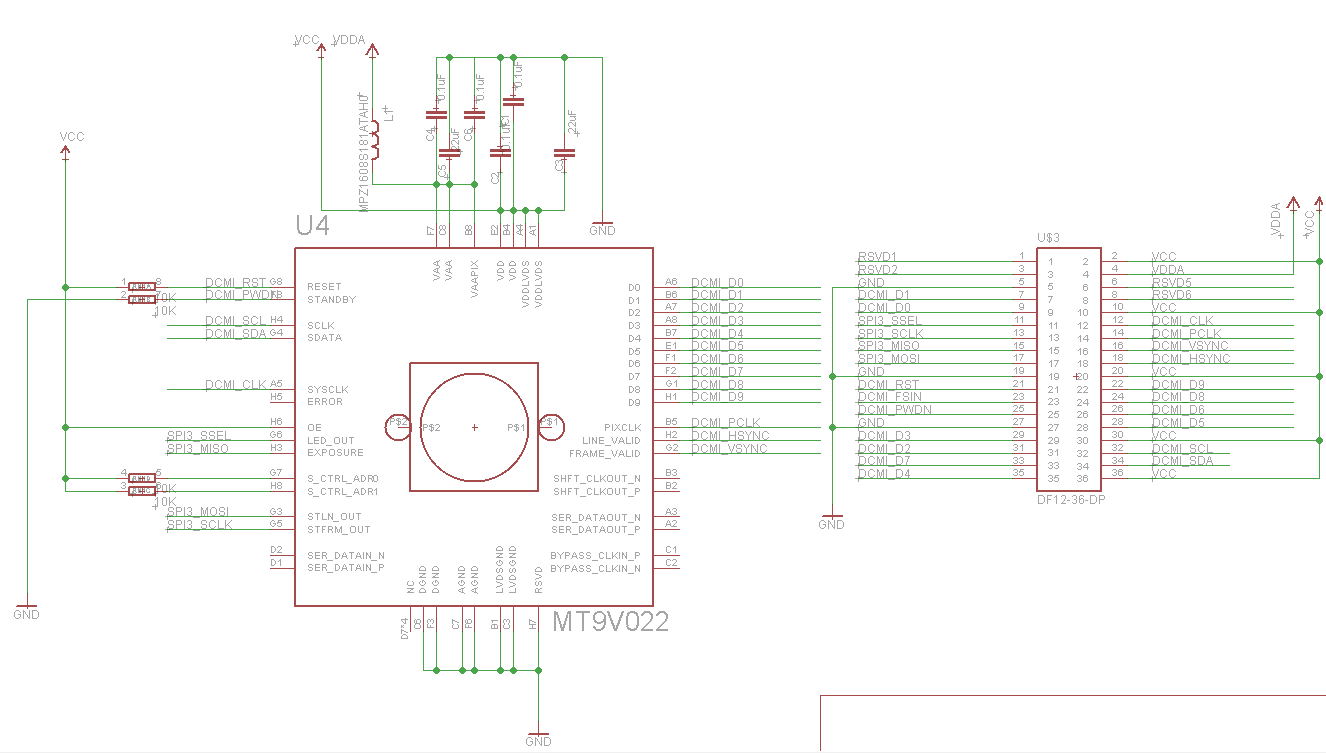
The camera works with an image sensor. Many bus are included : I2C, CAN, I/O or PWM.

# Open MV Camera - Schematic update

The schematic of the electronic board is available on the internet :

****

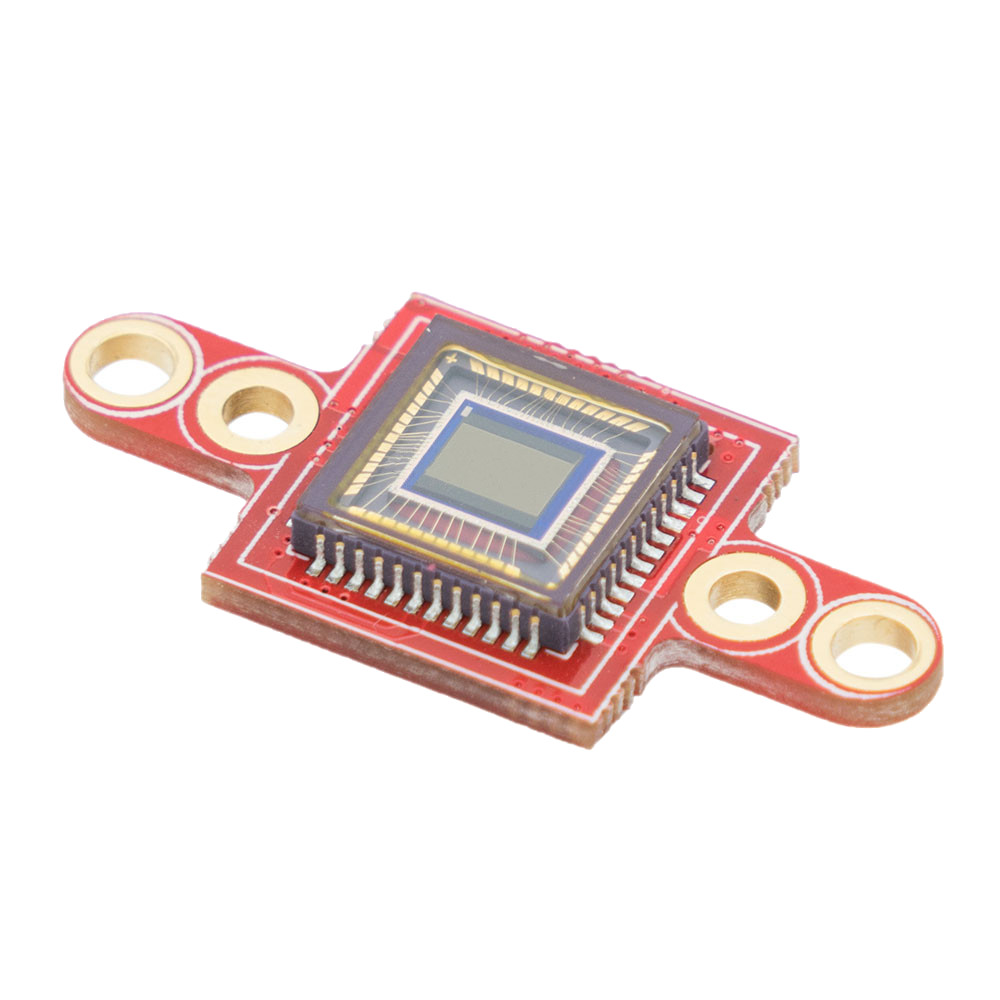
***Fig 1. :*** *Schematic of the electronic board used to control Open MV camera, from the official website of its designer*

****

***Fig 2.*** *:**Schematic of the electronic board for camera sensors, from the official website of its designer*

The PCB provided by OpenMV company is the following : 

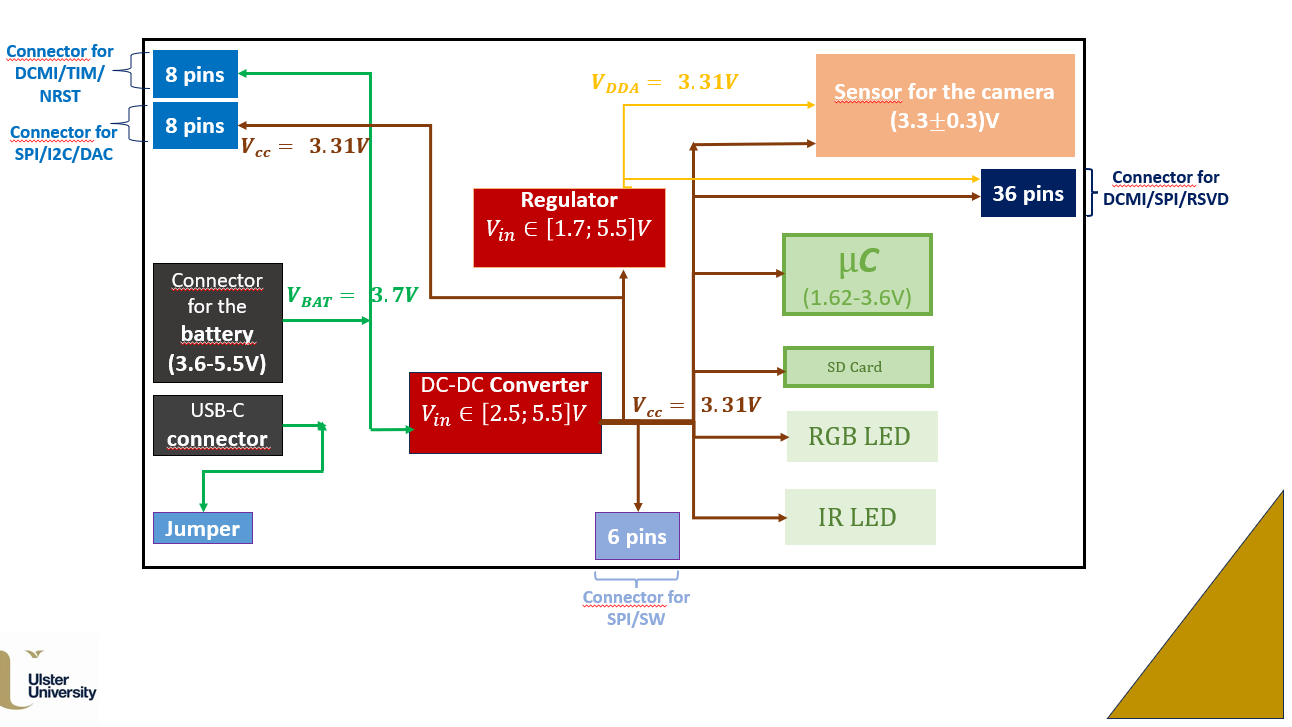
**Camera sensors PCB**



**Main board**

***Fig 3.*** *:**Electronic board – OpenMV Camera*

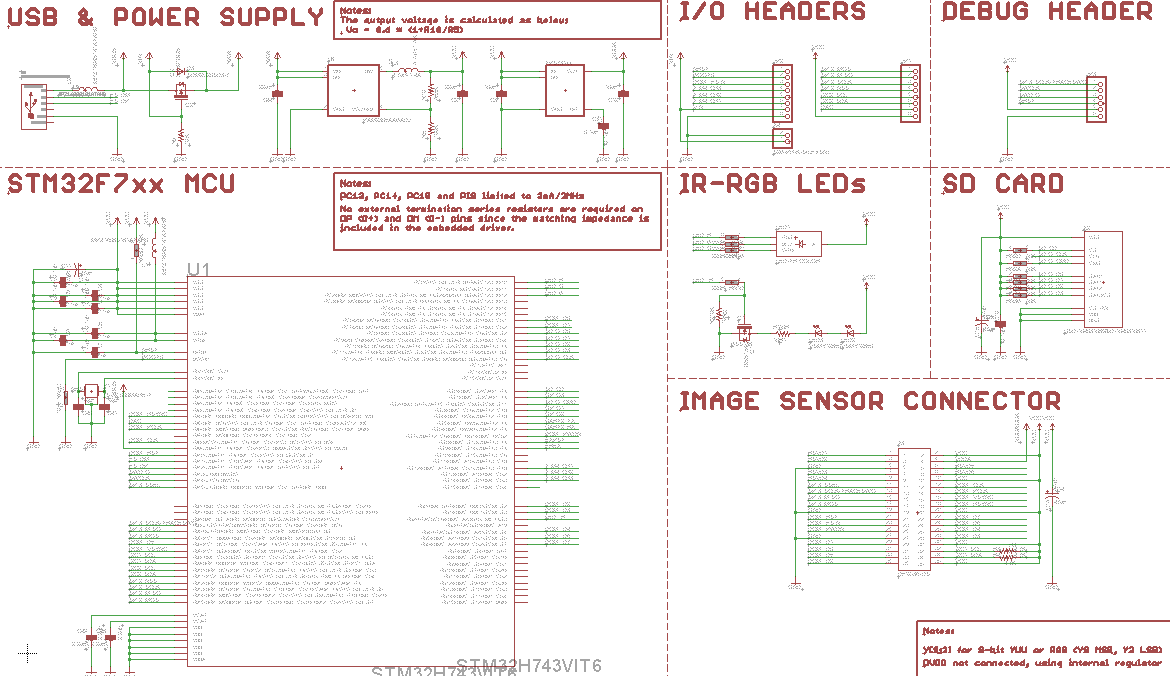
The following diagram allows us to understand how each component is connected and what is the purpose of this PCB :

****

***Fig 4.*** *:**Diagram explaining the electronic board*

The next part of the report focuses on the changes we implemented to adapt this electronic board to our project.

## Removed components

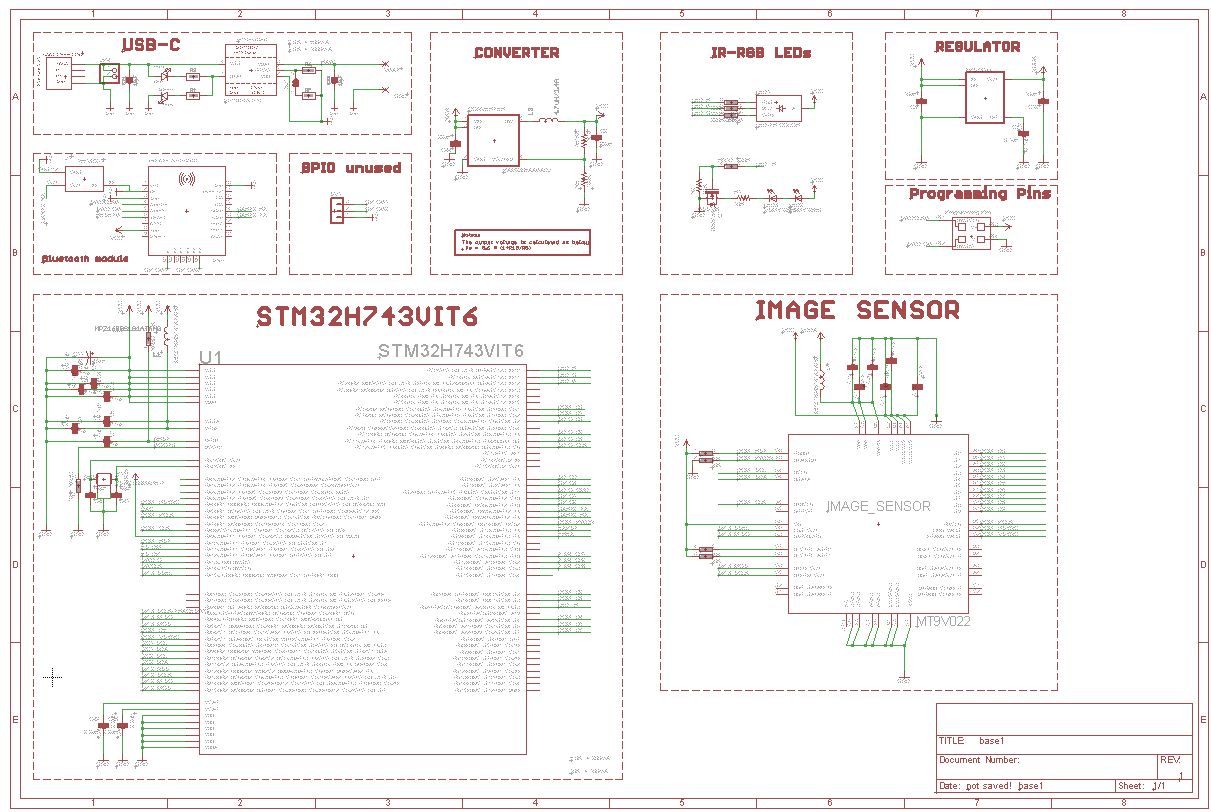
****

***Fig 5.*** *:**Components removed from the schematic*

As we will use the camera for a specific project, we needed to remove unnecessary components and to add some others.

* **SD Card :** Firstly, we removed SD Card components because we will not have to store data on the PCB.
* **I/O Headers & Debug Headers :** As we knew exactly which sensors we needed to use, we didn’t require to keep all the headers and decided to remove them.
* **Image sensors connector :** The designer of this board decided not to put the camera directly on the main PCB but to screw a small PCB on the main one. The sensors used for the camera were connected to the microcontroller thanks to a special connector. We decided to delete the image sensor connector because we wanted to connect directly the camera and its sensors on the main board.

## Added Components

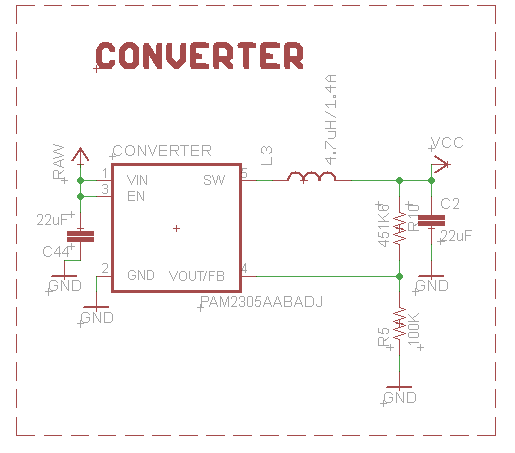
****

***Fig 6.*** *:**Components added to the schematic*

* **USB-C :** The first modification was to change USB connector. In fact, it was better to add a USB-C connector instead of a micro-USB because it is the new norm.
* **Image sensor :** Then, we added the sensors used for the camera openMV on the main board. We wanted to create an electronic board with all components on it, so we will glue plastic camera on our board.
* **Bluetooth Module :** As we wanted to be able to broadcast and receive data thanks to Bluetooth, we added a Bluetooth module on the schematic.
* **GPIO unused :** Many GPIO ports were still available on the microcontroller and on the Bluetooth module. We decided to add two pads so we can connect sensors with wires if we need to add new sensors later on.
* **Programming pins :** We added a special component used to program electronic components. This one will be used for the Bluetooth Module programming. In fact, the microcontroller (STM32) will be programmed thanks to USB.

# Explanation of each component

## Converter

****



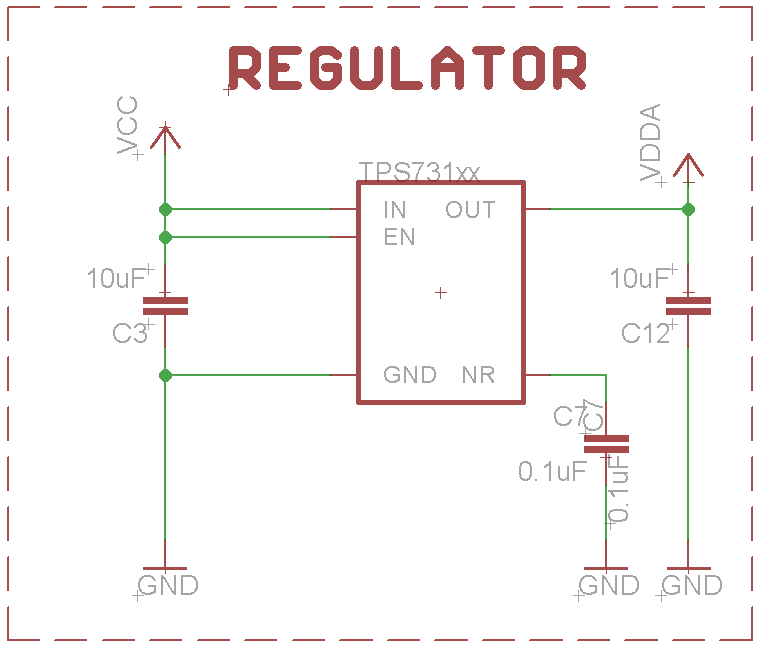
***Fig 7.*** *: Schematic and representation of the converter*

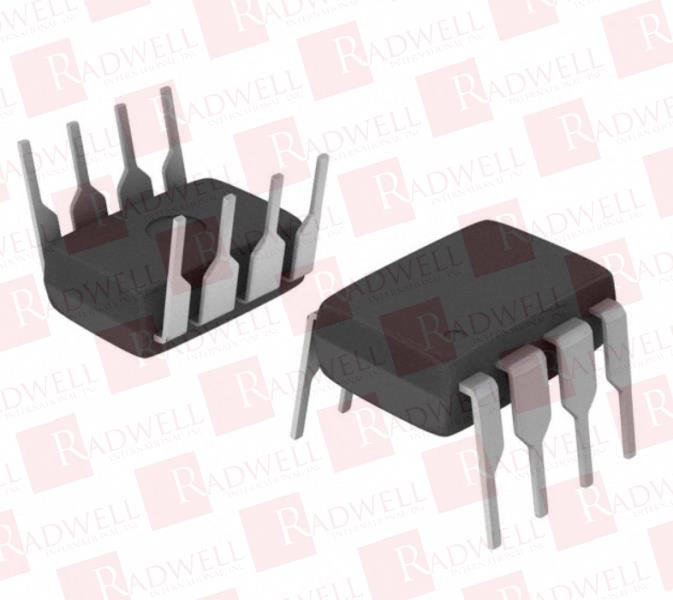
The converter is a DC-DC converter which supports input voltage from 2.5V to 5.5V.

This component is used to supply a voltage of 3.31V to several components on the electronic board (microcontroller, RGB LED, IR LED). In fact, the battery used delivers 3.7V, so the converter is useful to provide a lower voltage.

* ***More information on the converter in the appendix :*** [***click here***](#_Converter_documentation)

## Regulator

******

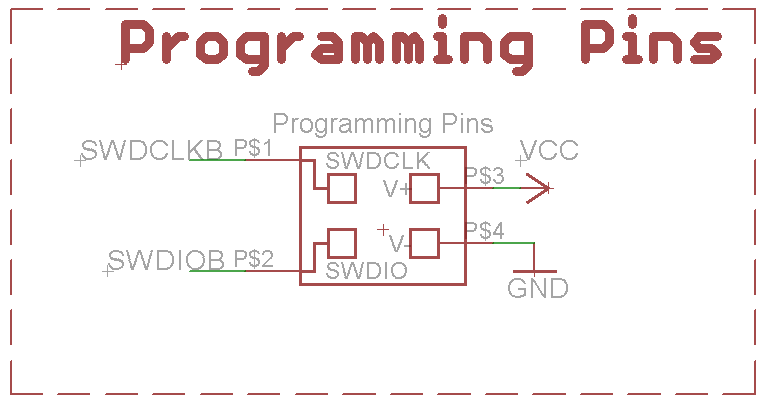


***Fig 8.*** *: Schematic and representation of the regulator*

This component is a ultra-low-dropout voltage regulator. The main characteristic is that it has a low current consumption. It is used to supply power to image sensor.

* ***More information on the regulator in the appendix :*** [*click here*](#_Regulator_documentation)

## Programming Pins



***Fig 9.*** *: Schematic of programming pins & Flasher programmer*

We needed to add this component to be able to program the Bluetooth Module. Furthermore, the microcontroller of the board will be programmed thanks to the USB connector.

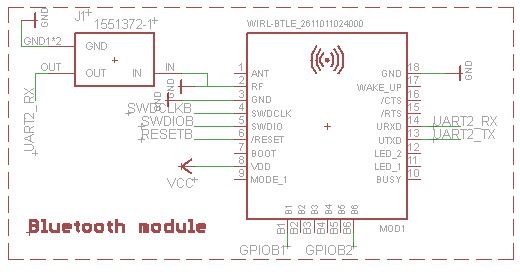
We have to follow the following steps to use programming pins : set up our development environment, select the board we are using, choose the programmer, upload our code.

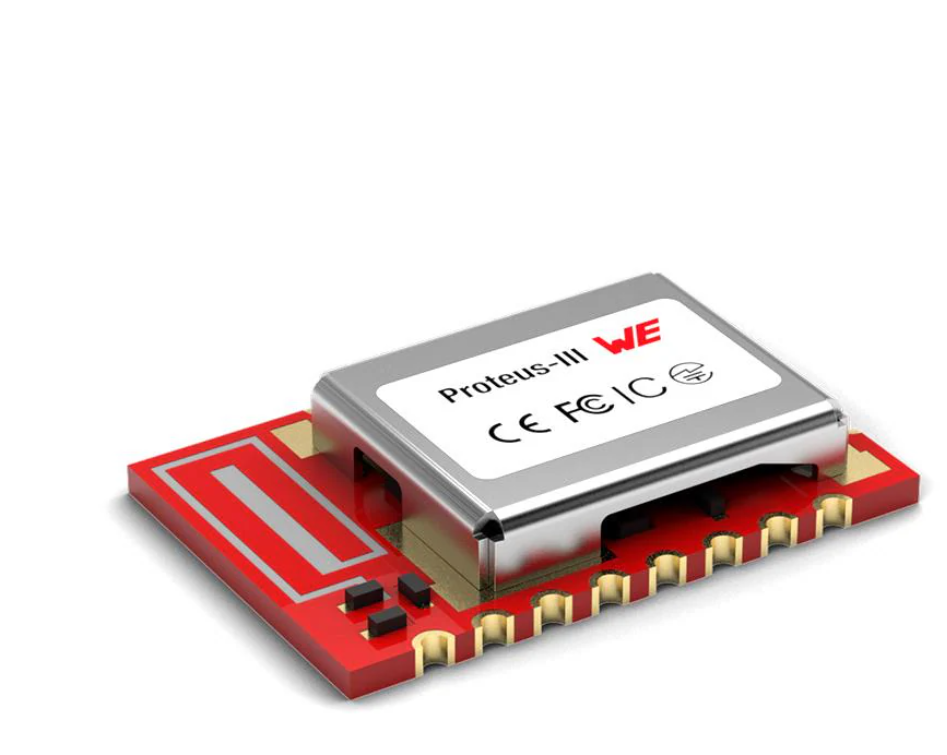
We chose to add a four pads component because we can use a special cable to connect our board to a flasher programmer.

* ***More information on the Flash Programmer in the appendix :*** [*click here*](#_Plug-of-nails_documentation)

## Bluetooth module & Antenna connector

### Description





***Fig 10.*** *: Schematic and representation of the Bluetooth module and of the antenna connector*

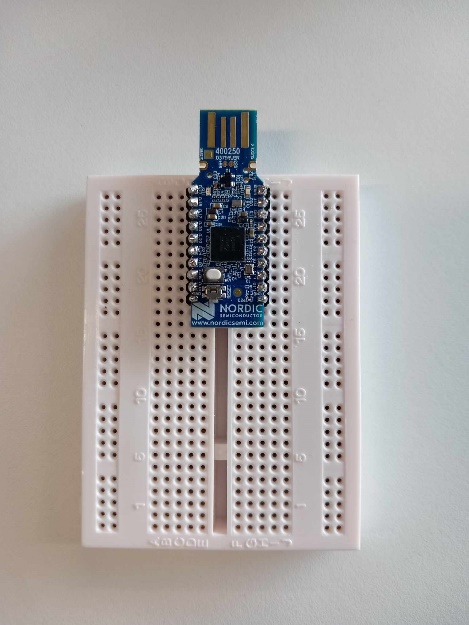
Bluetooth module will be used to collect and process data. Our first choice was this module : [Getting Started with XIAO nRF52840 | Seeed Studio Wiki](https://wiki.seeedstudio.com/XIAO_BLE/) . Unfortunately, it was not possible to add its footprint on the schematic. As we could not fix the problem, it was probably an Eagle parameter error, we decided to use another Bluetooth module. The main characteristic was that we needed to find a module with a Nordic semiconductor. We chose a module which requires a voltage between 1.8V and 3.8V. Its communication protocol is BLE (Bluetooth Low Energy).

**N.B**: Labels were added on the schematic (SDWDCLK***B***, SWDIO***B***, RESET***B***, GPIO***B***1, GPIO***B***2). The letter “***B*** ” was wrote to specify that these labels are linked to ***B***luetooth module ports. We needed to be careful because SDWDCLK, SWDIO, RESET, GPIO1 and GPIO2 are labels used for STM32 ports.

* ***More information on the Bluetooth module in the appendix :*** [*click here*](#_Bluetooth_module_documentation)

### Bluetooth module problems

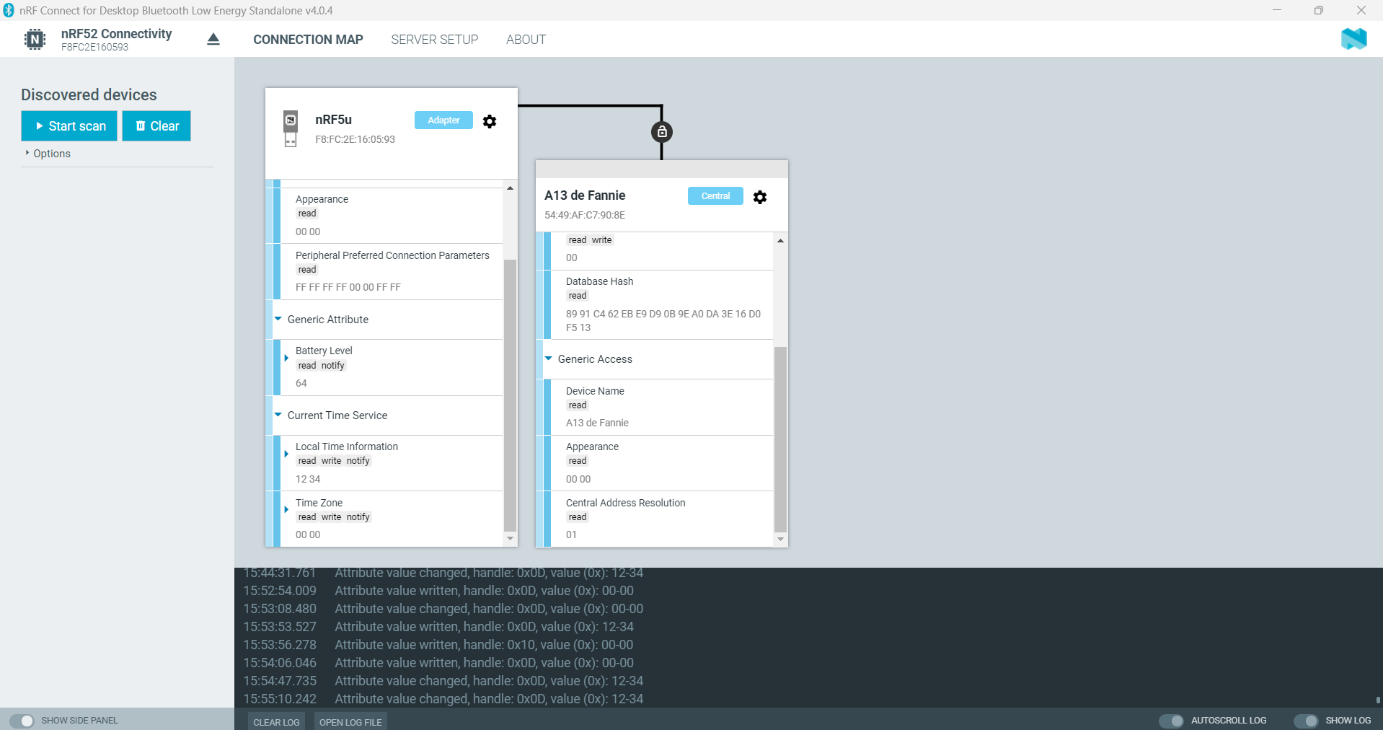
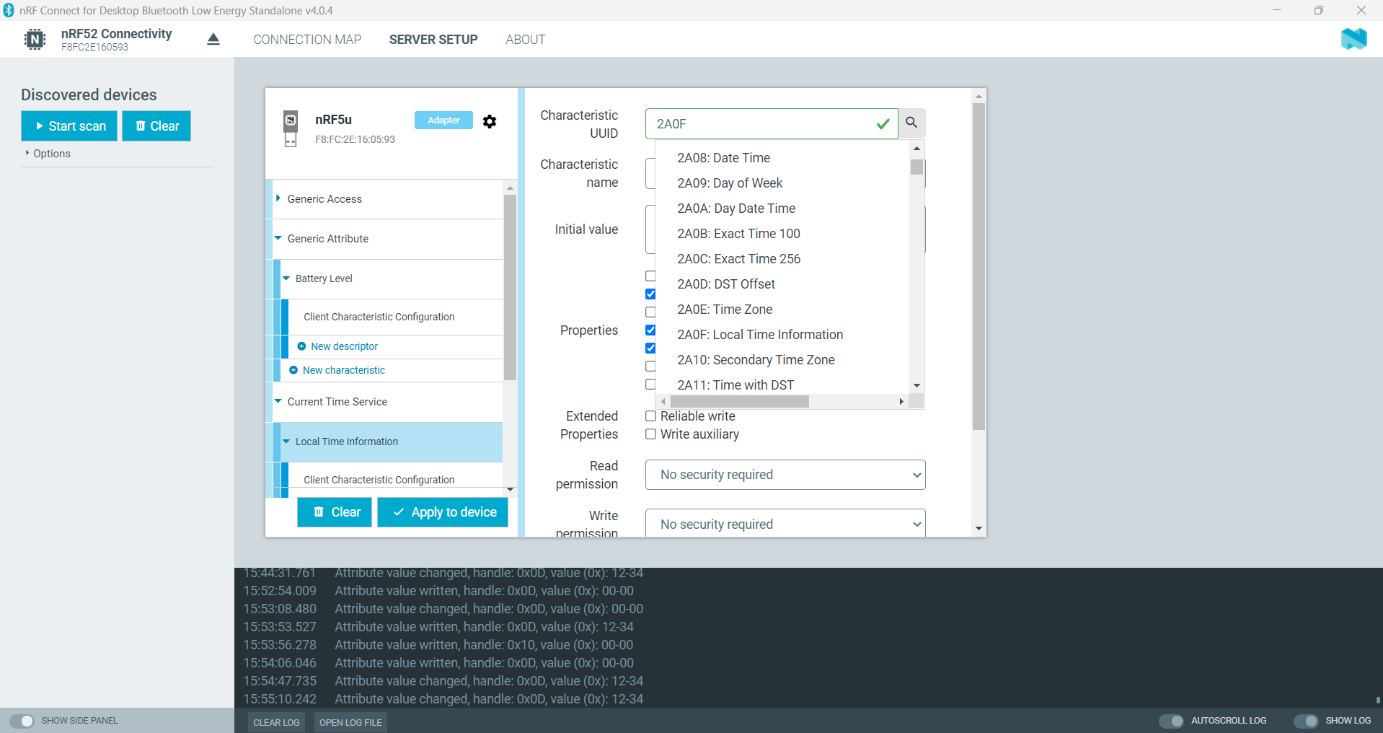
The first attempt was to use a Nordic module Bluetooth (nRF52840 {Dongle}) . The first aim was to try to getting started with the software nRF Connect. It is used to connect Nordic components to computers and to develop computer codes.



Thanks to nRF Connect for Desktop Bluetooth Low Energy (BLE), mobile phones can be connected to the nRF52840 :

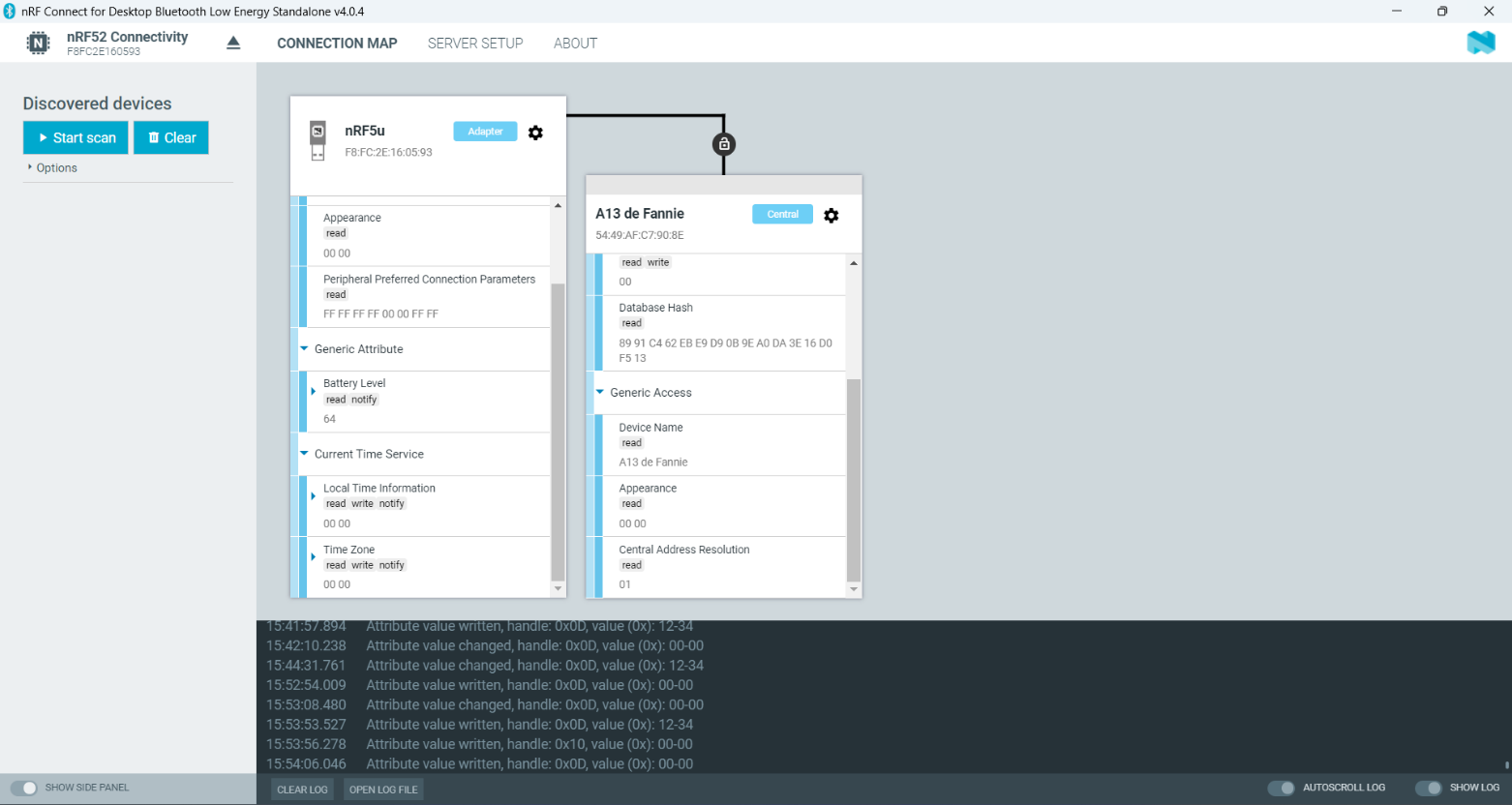
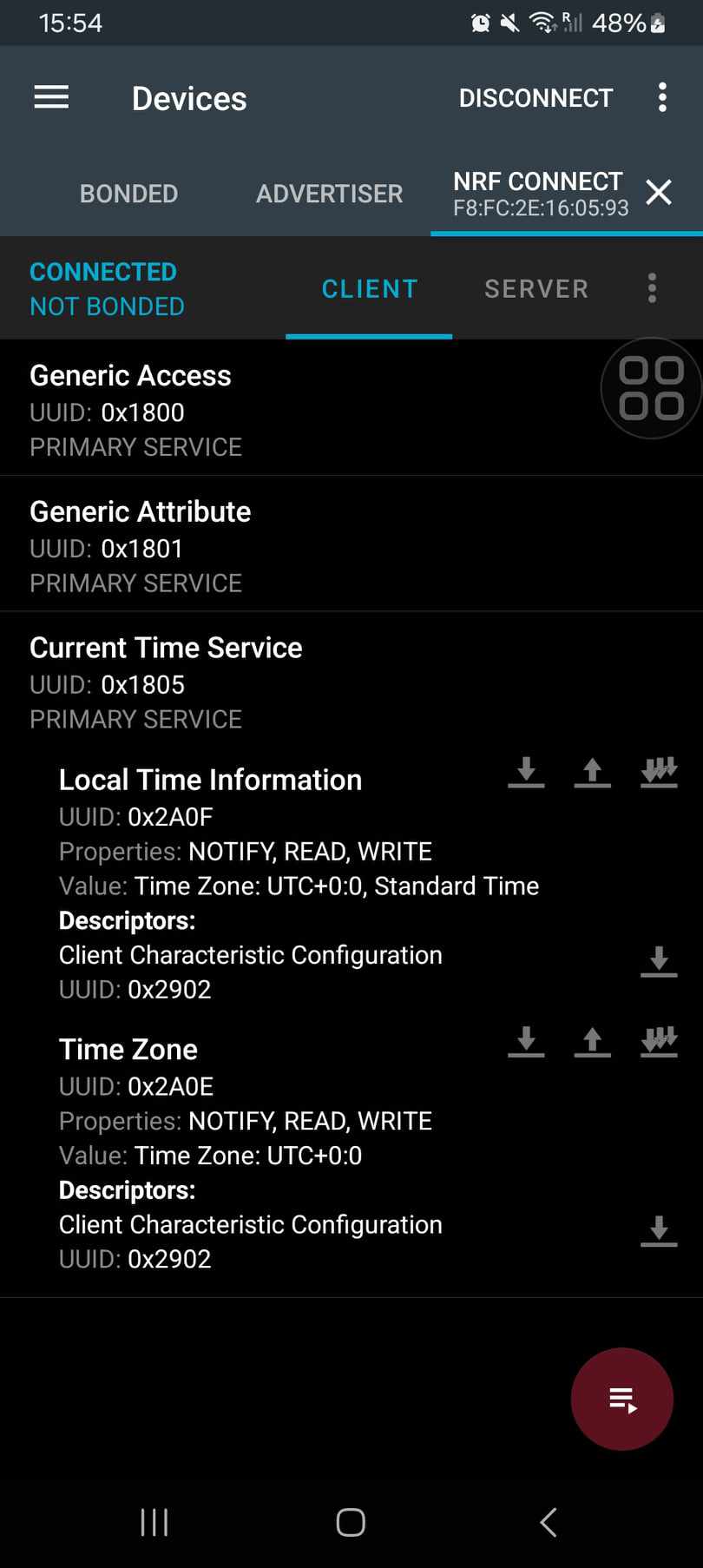
* We need to change characteristics to choose the data we want to broadcast.
* It is possible to change many parameters on the nRF Connect for Desktop. For example, I chose to modify time information and Time zone for the first test :

***Fig 11.a****: nRF 52840 Dongle*



***Fig 11.b****: Parameters of the nRF 52840 Dongle*

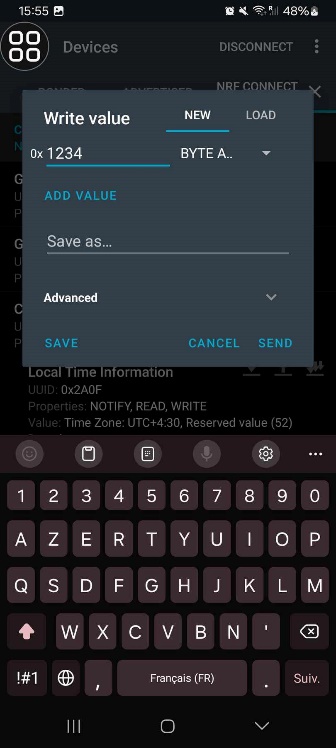
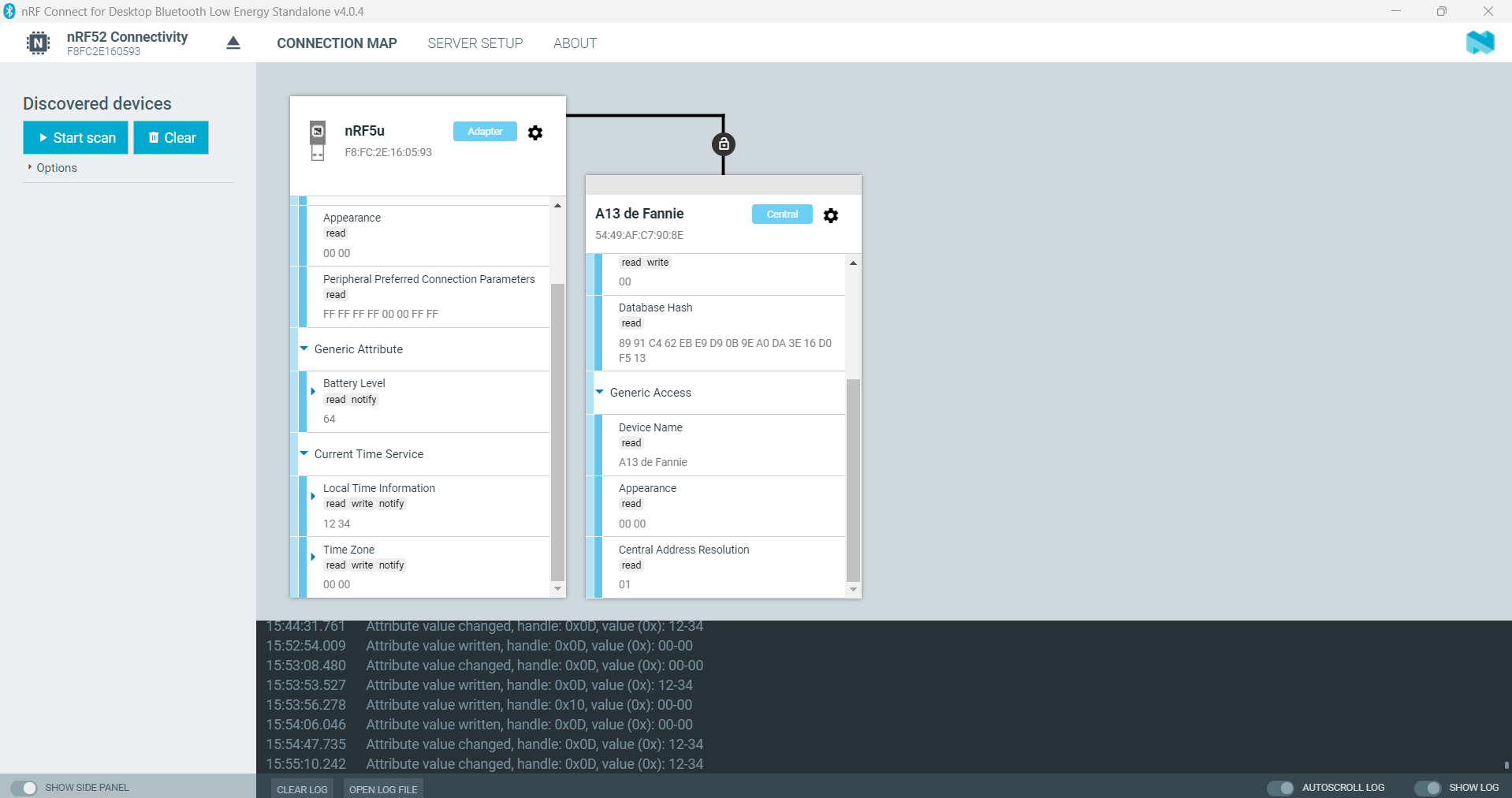
Values of parameters can be read on the Computer app and on the mobile phone app too :



***Fig 13****: Parameters of the nRF 52840 that we can see on the nRF mobile phone app*

***Fig 12****: Default Parameters of nRF 52840*

The values can be modified from mobile app and can be sent thanks to Bluetooth to the nRF 52 840 board.



***Fig 14****: Values changed on the mobile phone app* ***Fig 15****: Values read on the Computer app*

It is also possible to change data values on the computer app and to receive them on the mobile phone app.

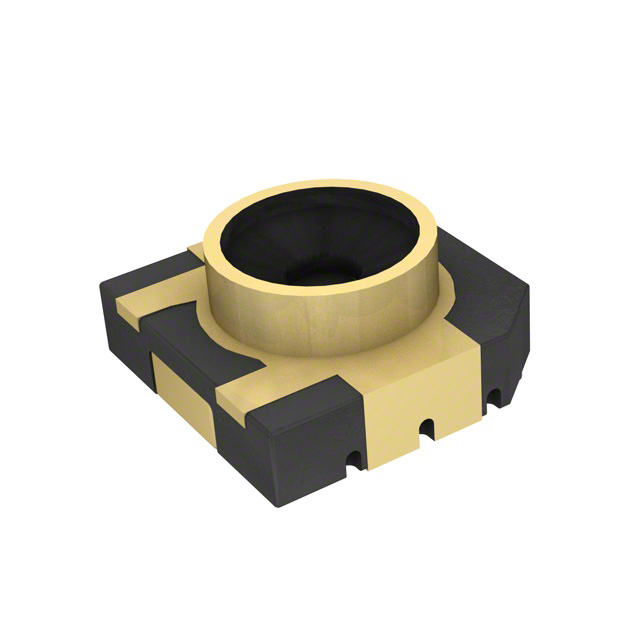


The second attempt was to use the xiaonRF52840 sense module to add Bluetooth communication on the PCB. This module can be programmed with Arduino IDE or with nRF Connect for computer and for mobile phone. I tried to test some examples given by Arduino IDE and then I used nRF Connect app in order to transfer data between the board and my mobile phone.

***Fig 16****: Seed xiao nRF52840*

### Antenna connector

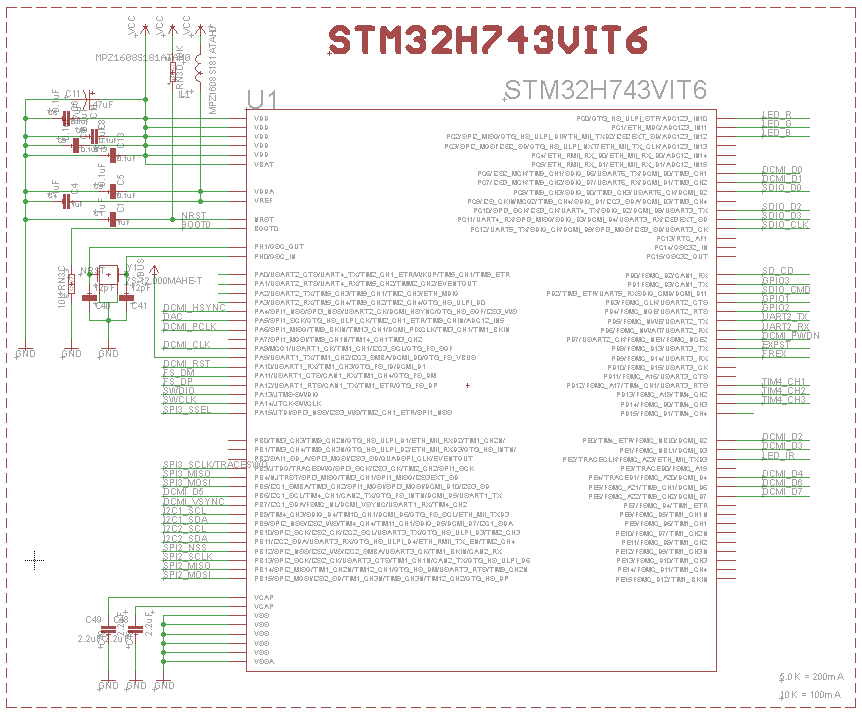
As explained in the user manual for the Bluetooth module, it is possible to use an internal or an external antenna. We decided to add an antenna connector (a socket connector) :

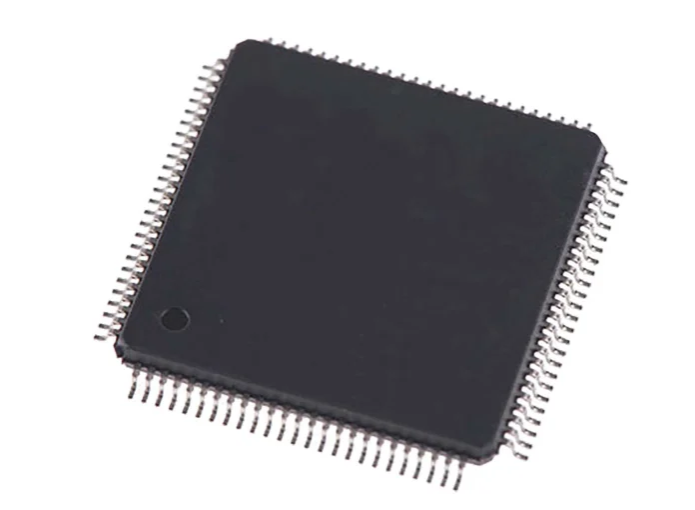


***Fig 17****: 3D model of the antenna connector*

* ***More information on the antenna connector here :*** [1551372\_Dwg.pdf (digikey.com)](https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/399/1551372_Dwg.pdf)

## Microcontroller



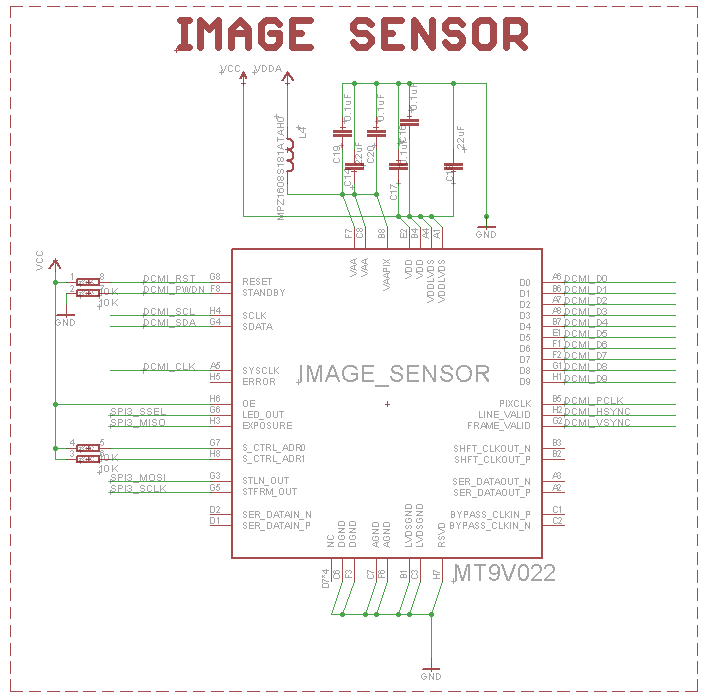


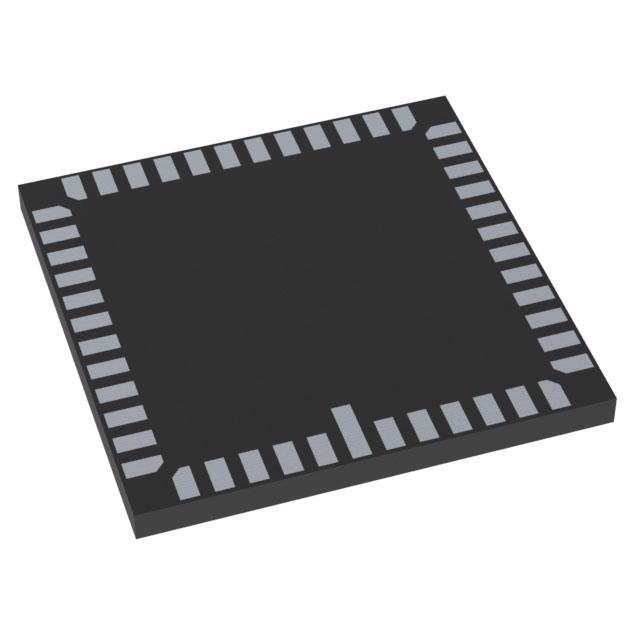
***Fig 18. :*** *Schematic and representation of the STM32*

This microcontroller requires a voltage between 1.62V and 3.6V . Up to 168 I/O ports can be used. It controls LEDs and SPI, I2C, DCIM or GPIO ports are available. For example, some of these ports are used to manage the image sensor.

* ***More information on the microcontroller in the appendix :*** [*click here*](#_STM32_documentation)

## Image sensor



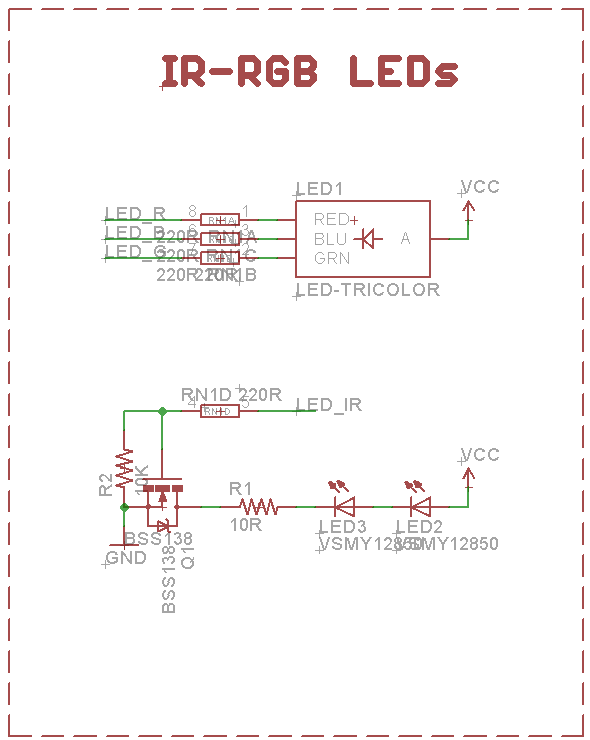


***Fig 19.*** *: Schematic and representation of the image sensor*

This sensor is a CMOS active-pixel digital image sensor used for OpenMV Camera. We decided to add this component on our board instead of a connector to only have one PCB with the camera glue on it.

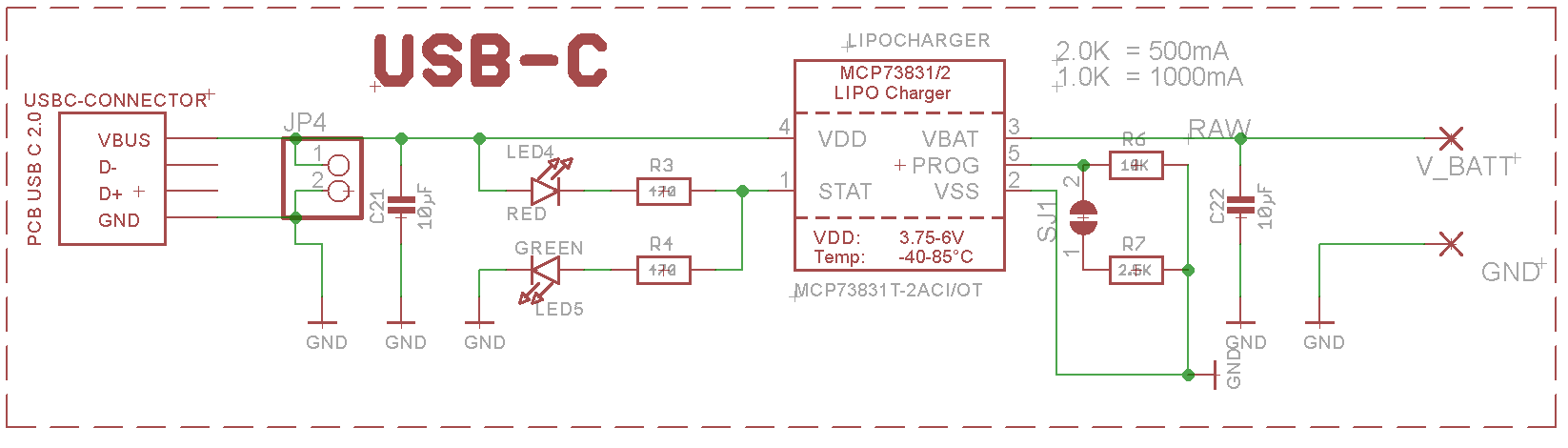
* ***More information on the converter in the appendix :*** [*click here*](#_Image_sensor_documentation)

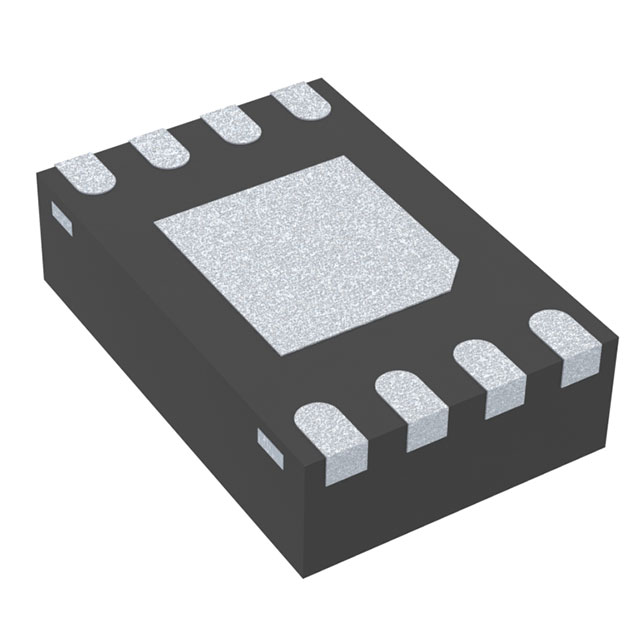
## LEDs



***Fig 20.*** *: Schematic and representation of the LEDs*

## USB-C connector and Lipo charger





***Fig 21.*** *: Schematic and representation of the Lipo charger*

We added an USB-C connector instead of a micro-USB because USB-C is the new norm.

* ***More information on the converter in the appendix :*** [*click here*](#_Battery_charge_documentation)

# Routing

## The different versions of the board

A computer screen shot of a circuit board

Description automatically generated

A blueprint of a computer chip

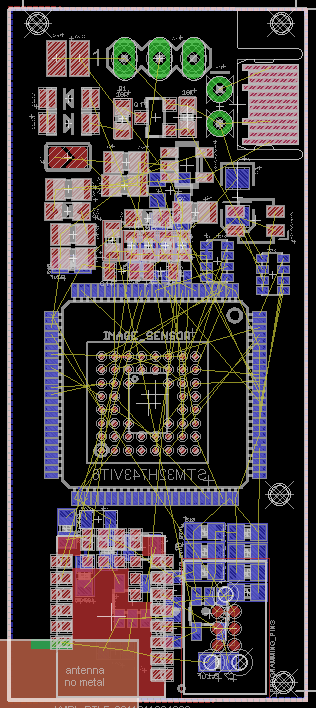
Description automatically generatedA computer chip with a black background

Description automatically generated

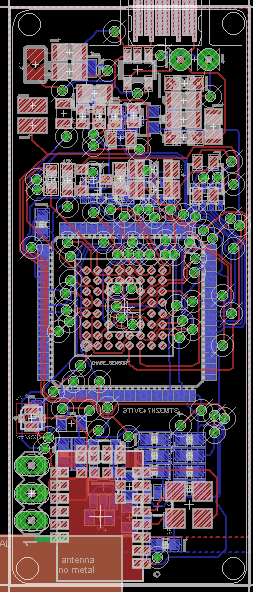
***Fig 22.*** *: Board – version 1* ***Fig 23.*** *: Board - version 2* ***Fig 24.*** *: Board – version 3*

* The first version was designed to try to place all the components on the top of the board. I needed to change it because the board was not optimized .
* The second version was better for components placement but I had to narrow the length of the board.
* Then, with the third version I tried to place all the components on both side (Top & Bottom) but I had to reduce the width of the board.

For the last version, I tried to optimize the board before routing :



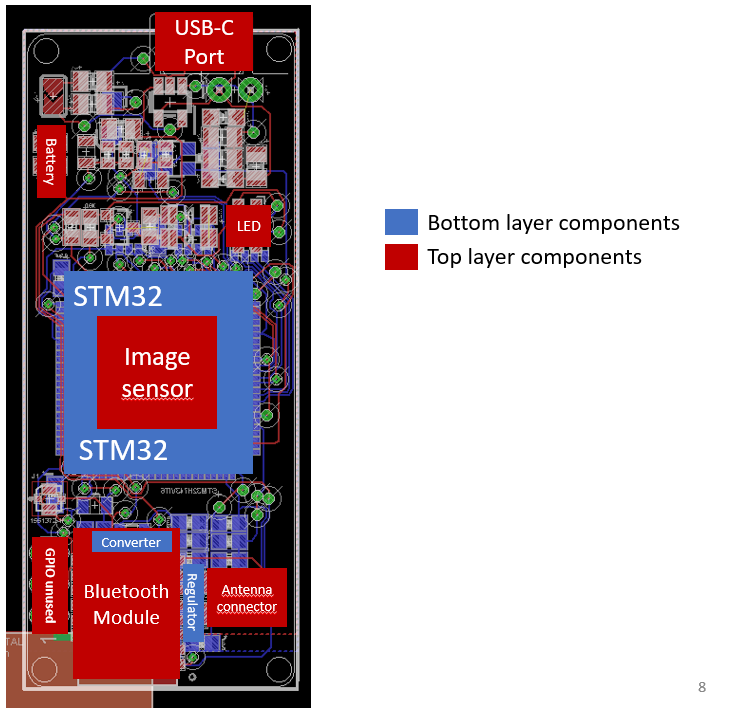
***Fig 25.*** *: Board - version 4*



The last step was to route the board. We chose the following parameters :

* Route – width : 0.006
* Circle Via – drill : 0.01

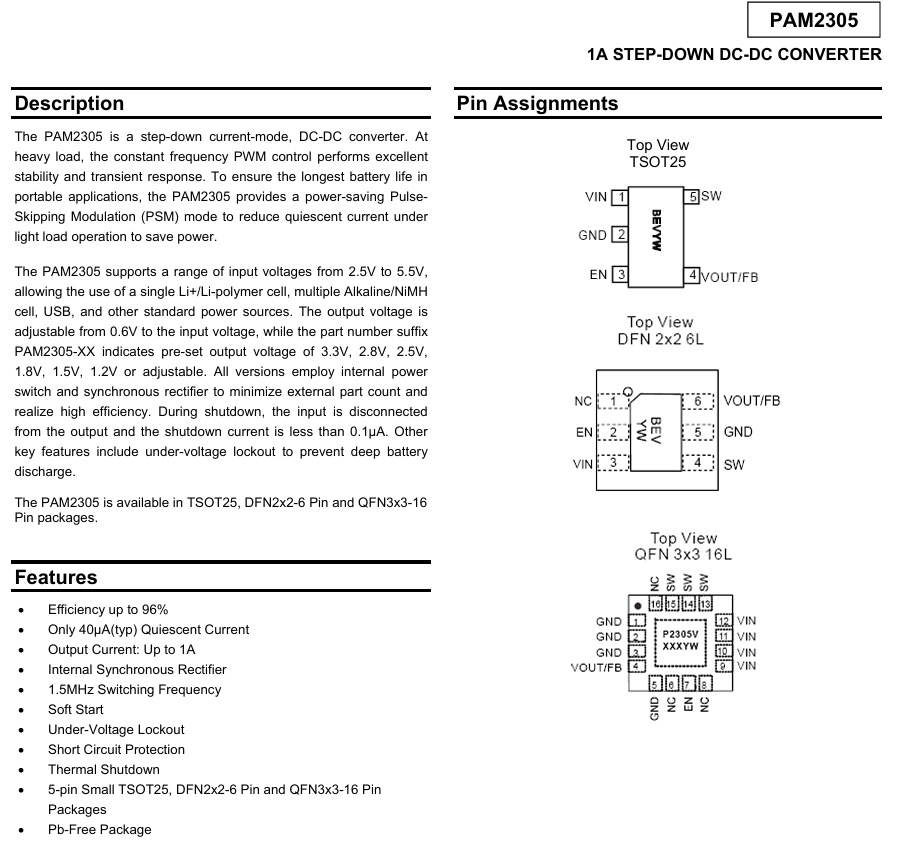
***Fig 26.*** *: Routed board*



***Fig 27.*** *: Legend of the final version of the PCB*

# Appendices

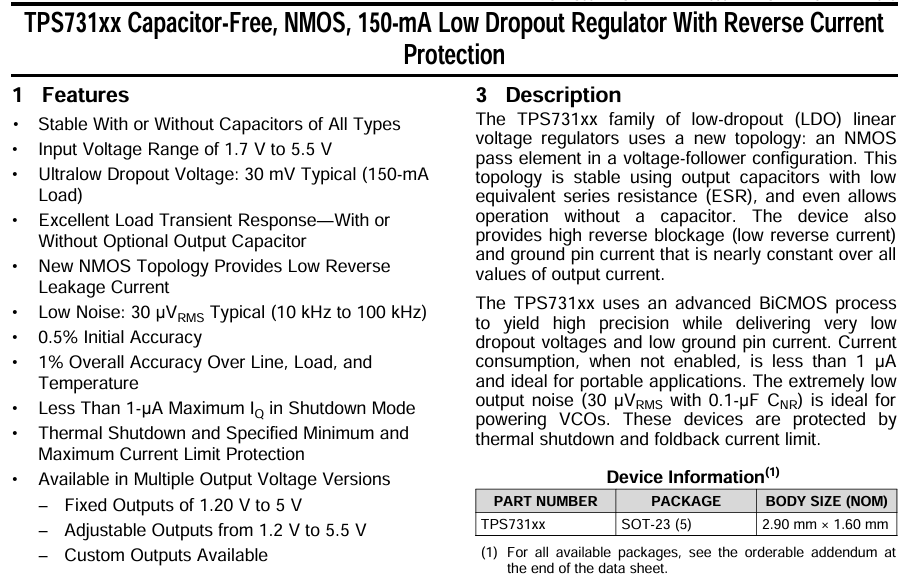
## Converter documentation



***Fig 28. :*** *Extract from converter datasheet*

Converter datasheet link : [PAM2305-247289.pdf (mouser.com)](https://eu.mouser.com/datasheet/2/115/PAM2305-247289.pdf)

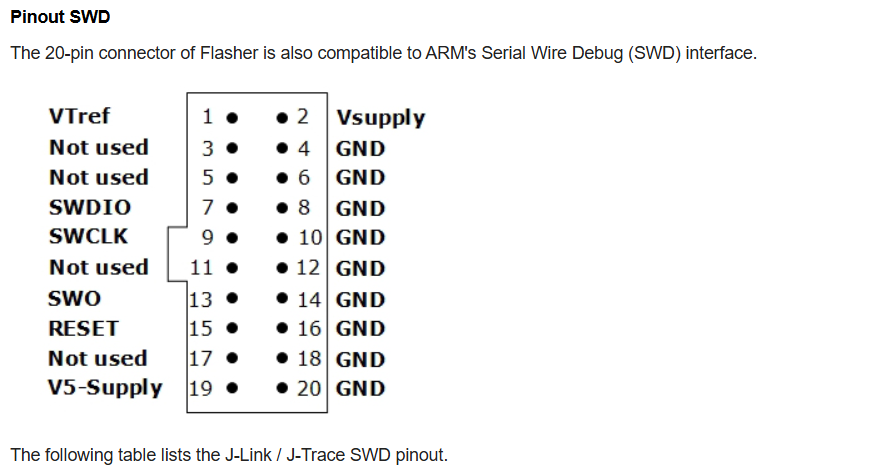
## Regulator documentation



***Fig 29. :*** *Extract from regulator datasheet*

Regulator datasheet link :[tps731.pdf (ti.com)](https://www.ti.com/lit/ds/symlink/tps731.pdf)

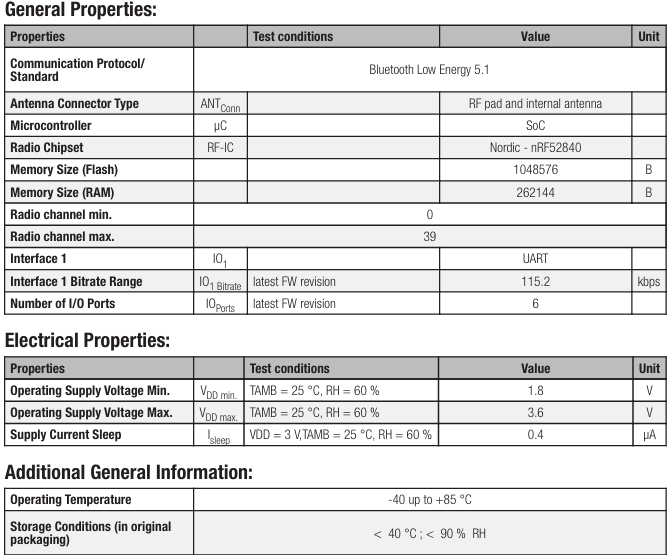
## Flasher Programmer documentation



***Fig 30. :*** *Extract from Flasher programmer datasheet*

Flasher Programmer datasheet link : [UM08022 Flasher - SEGGER Wiki](https://wiki.segger.com/UM08022_Flasher)

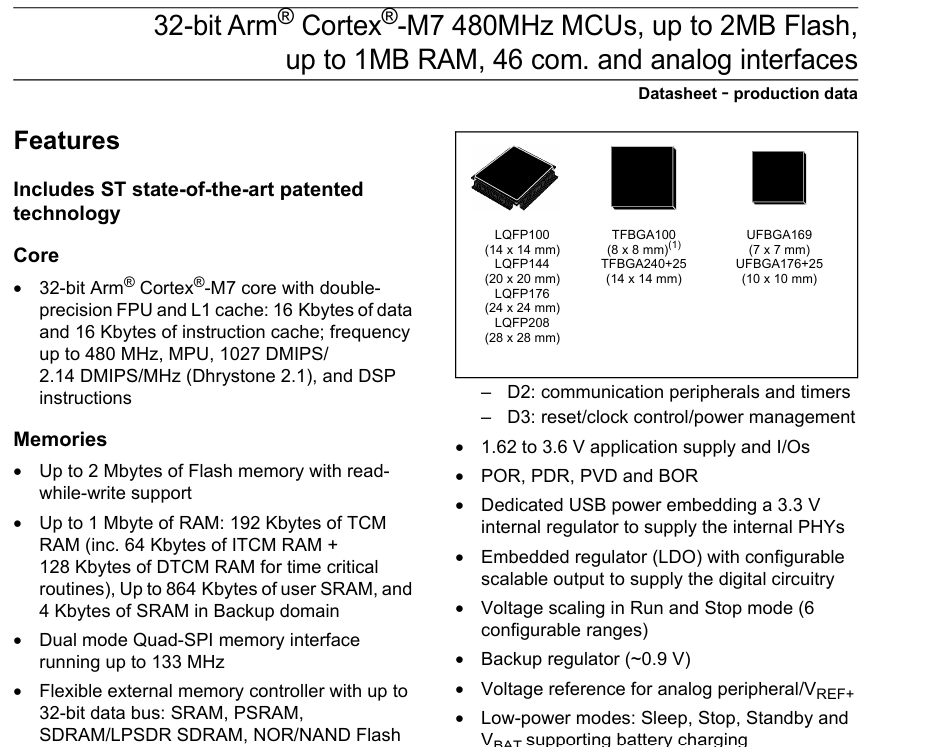
## Bluetooth module documentation



***Fig 31. :*** *Extract from Bluetooth module datasheet*

Bluetooth Module datasheet link : [A700000007100829.pdf (rs-online.com)](https://docs.rs-online.com/735a/A700000007100829.pdf)

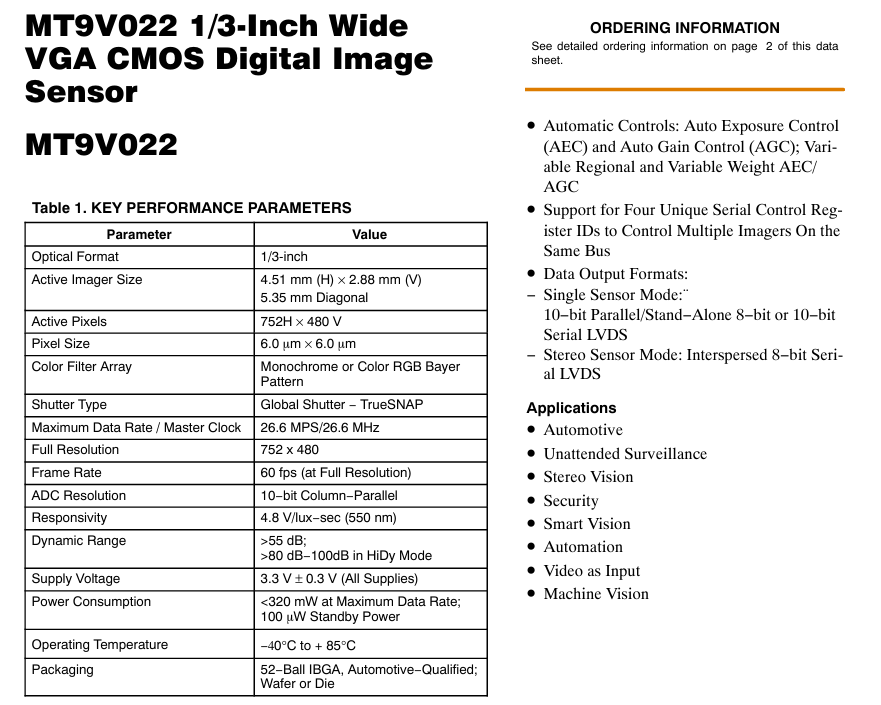
## STM32 documentation



***Fig 32. :*** *Extract from stm32 sdatasheet*

Microcontroller datasheet link : [stm32h745zg-1760969.pdf (mouser.com)](https://eu.mouser.com/datasheet/2/389/stm32h745zg-1760969.pdf)

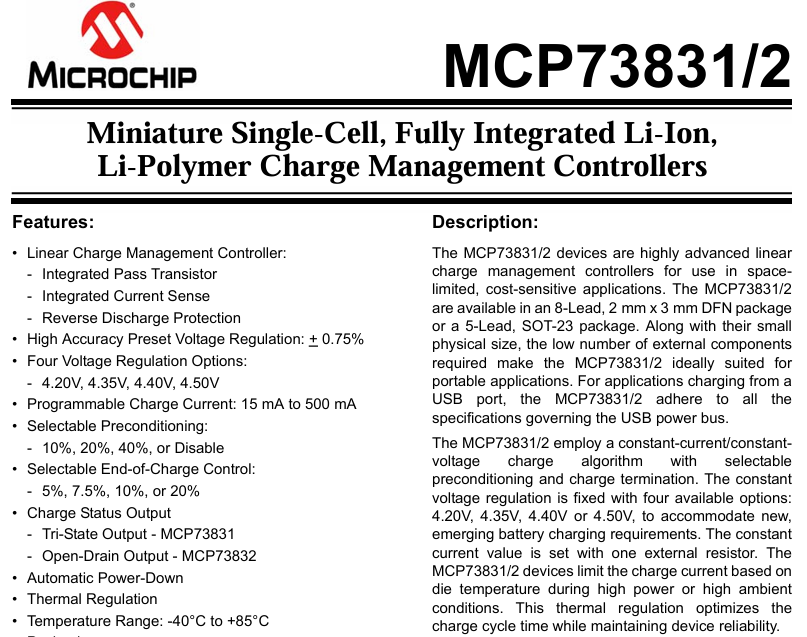
## Image sensor documentation

.

***Fig 33. :*** *Extract from Image sensor datasheet*

Image sensor datasheet link : [MT9V022\_D-2316502.pdf (mouser.com)](https://eu.mouser.com/datasheet/2/308/1/MT9V022_D-2316502.pdf)

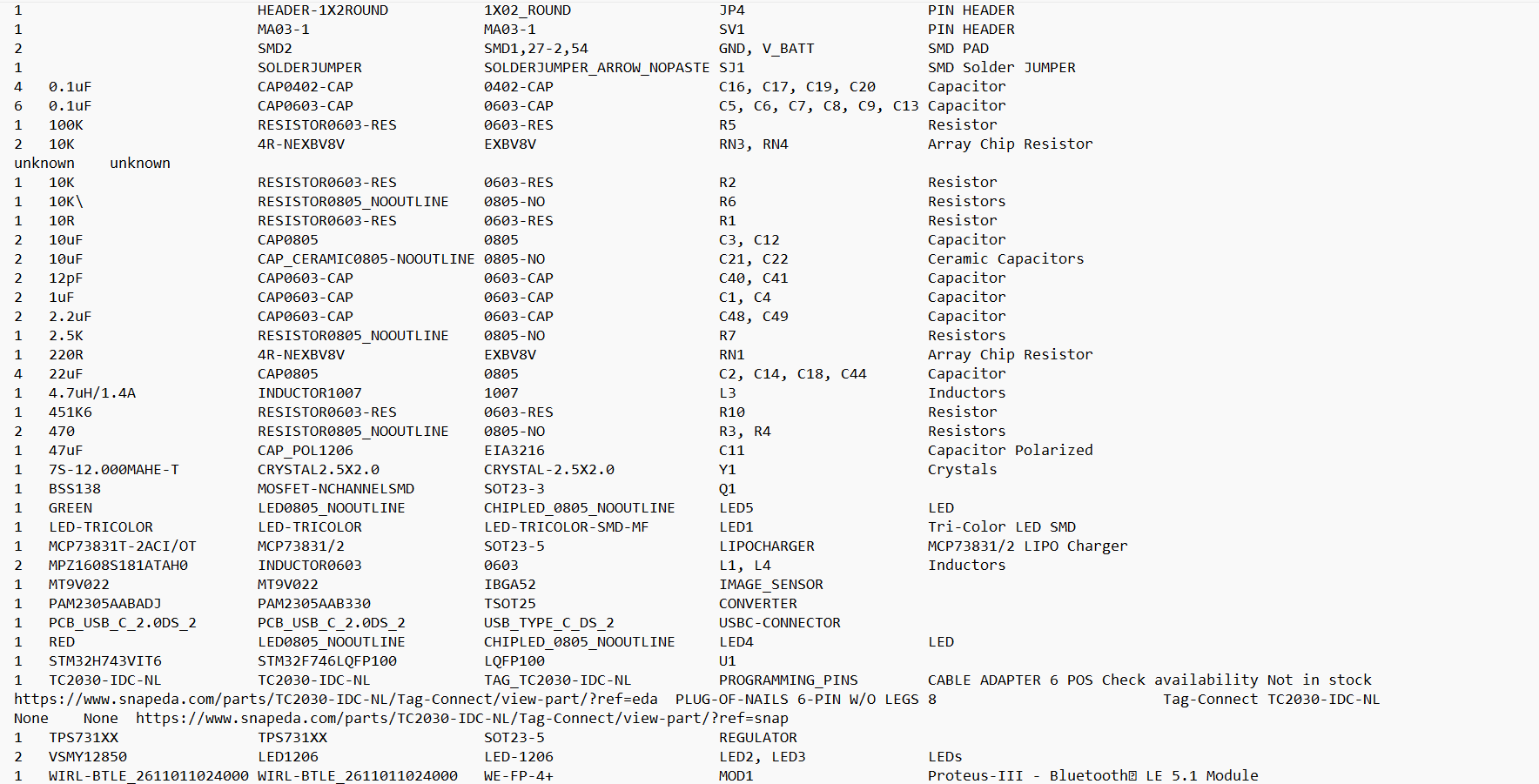
## Battery charge documentation



***Fig 34. :*** *Extract from Battery charge datasheet*

Battery Charge datasheet link : [0900766b814f3cb1.pdf (rs-online.com)](https://docs.rs-online.com/6a41/0900766b814f3cb1.pdf)

## Bill Of Materials



***Fig 35. :*** *BOM (Bill Of Materials) of the new PCB*