

Tube Pitot déporté

Détection de l'angle d'incidence au décrochage d'un avion par mesure de la vitesse et des accélérations

Meven Ricchieri
2022 - 2023

Numéro de projet : 2230

Responsable : M. Juan José Moreno
Mandataire : M. Vincent Seguin (pour AMPA)



Table des matières

1 Cahier des charges	3
2 Pré-étude	3
2.1 Contexte	3
2.2 Schéma de principe	3
2.3 Schéma bloc détaillé	4
2.4 Descriptions des blocs détaillées	4
2.4.1 Bloc Générateur (turbine)	4
2.4.2 Bloc Capteur de pression	5
2.4.3 Bloc Tube Pitot	5
2.4.4 Bloc IMU	6
2.4.5 Bloc Connecteur USB	6
2.4.6 Bloc Module BLE	7
2.4.7 Bloc LED bleue	7
2.4.8 Bloc MCU	7
2.4.9 Bloc Alimentation	7
2.4.10 Bloc Appareil Android	8
2.5 Design mécanique	8
2.6 Estimation du prix	9
2.7 Faisabilité du projet	9
3 Design électronique	10
3.1 Introduction	10
3.2 Générateur	10
3.3 LED de signalisation	12
3.4 Piles rechargeables	13
3.5 Capteur de pressions différentielles	14
3.6 IMU	15
3.7 Module Bluetooth	16
3.8 MCU	17
3.9 Alimentations	19
3.10 Pull-up I2C	20
4 Design mécanique	21
4.1 Introduction	21
4.2 Design	21
4.3 Modifications	25
4.3.1 Circuit imprimé	25
4.3.2 Boîtier	25
5 Mise en service	26
5.1 Introduction	26
5.2 Test des alimentations	26
5.3 Consommation	26
5.4 Périphériques	26
5.4.1 Signaux USART	27
5.4.2 Signaux I2C	28
5.5 Améliorations possibles	29

6 Software MCU	30
6.1 Introduction	30
6.2 Paramétrage Harmony	30
6.2.1 Fréquence d'horloge	30
6.2.2 Timers	31
6.2.3 Entrées sorties	32
6.2.4 Communications série	33
6.3 Flowcharts	34
6.3.1 Machine d'état principale	34
6.3.1.1 Etat d'initialisation	35
6.3.1.2 Etat de service	36
6.4 Codes C	37
7 Software Android	39
7.1 Introduction	39
7.2 Aperçu des interfaces	39
8 Résultat final	40
8.1 Introduction	40
8.2 État d'avancement	41
8.3 Tâches restantes	41
9 Conclusion	42
10 Annexes	43

1 Cahier des charges

Le but du projet consiste à développer un système permettant de détecter l'angle d'incidence et la vitesse au décrochage d'un avion. La fixation de ce système doit être flexible afin de pouvoir l'installer sur différents types d'avions. L'emplacement de fixation ne doit pas se trouver dans le flux d'air provenant de l'hélice afin d'éviter que la mesure de vitesse ne soit faussée. Il doit également être miniaturisé au maximum afin de produire le minimum de traînée possible et de ne pas dépasser un poids de 500g. Les données acquises par les capteurs doivent être transmises de la partie déportée à un appareil Android se trouvant dans le cockpit de l'avion au travers une communication sans fil. L'appareil Android doit traiter et afficher les données reçues, si possible graphiquement (optionnel).

Le cahier des charges complet se trouve en annexes.

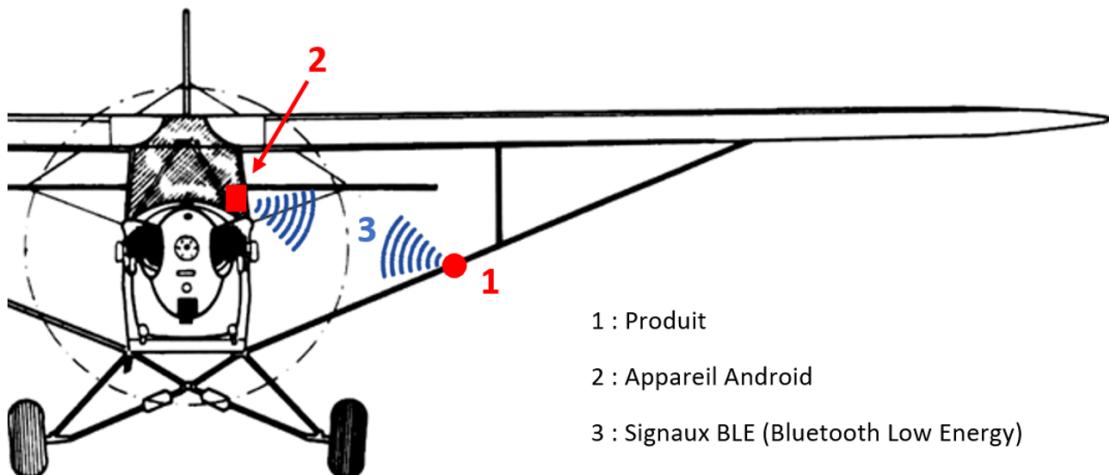
2 Pré-étude

2.1 Contexte

Cette pré-étude consiste à mener des recherches qui conduiront à une décision de lancer ou non le projet. Cette décision va reposer sur les difficultés de réalisation électrique, mécanique et software. Elle va aussi permettre d'imaginer et de comparer les différentes manières de réaliser les tâches et ainsi choisir la meilleure.

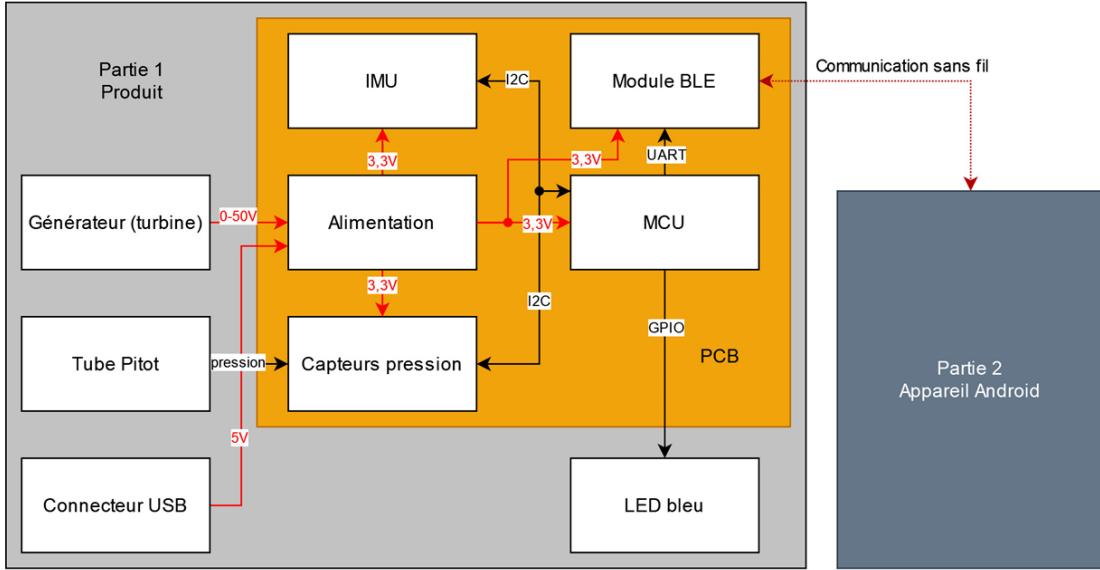
2.2 Schéma de principe

FIGURE 1 – Schéma de principe



2.3 Schéma bloc détaillé

FIGURE 2 – Schéma bloc détaillé



2.4 Descriptions des blocs détaillées

2.4.1 Bloc Générateur (turbine)

L'intérêt d'utiliser un générateur comme source d'énergie permet de simplifier l'utilisation du système, il n'y aura aucune maintenance à effectuer ou de batterie à charger. L'énergie cinétique de l'air en mouvement, par rapport à l'avion, sera captée par les aubes de la turbine axée à un générateur, ce qui va produire une tension. Le système va se mettre en marche une fois que l'axe du générateur-turbine aura atteint une certaine vitesse de rotation. Plus la vitesse sera grande, plus la tension générée sera élevée. Un simple moteur DC, va permettre de transformer cette énergie mécanique en énergie électrique. Il est possible d'utiliser d'autres technologies comme les moteurs brushless, qui ont un rendement bien meilleur, mais cela complexifie le circuit et augmente le prix. Dans ce projet, le rendement n'est pas primordial car le système ne consommera que très peu d'énergie.

$$U_t(V) = \frac{n}{k_n} - R_{mot} * I_L [2] \quad (1)$$

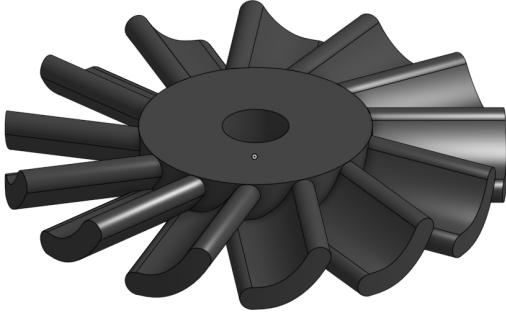
où :

U_t	= Tension aux bornes du moteur DC (V)	
k_n	= Constante de vitesse du moteur DC (rpm/V)	
R_{mot}	= Résistance aux bornes du moteur DC (Ω)	Puisque la valeur de la tension générée
n	= Vitesse moteur DC (rpm)	
I_L	= Courant de charge (I)	

réelle est dépendante de la vitesse air de l'avion, celle-ci ne sera jamais stable. Il faudra donc la transformer et la réguler avant de la transmettre aux composants électroniques, ce à quoi le bloc "Alimentation" sera destiné.

N'ayant pas de grandes connaissances dans le dimensionnement et le design de générateur-turbines, des recherches plus poussées seront nécessaires afin de rendre le système performant. Pour me faire une première idée, j'ai dessiné et imprimé une première turbine composée de 13 aubes puis j'ai effectué quelques tests. A l'aide d'un moteur DC, j'ai réussi à générer une tension d'environ 3V à une vitesse de 80km/h.

FIGURE 3 – Turbine de test N°1



2.4.2 Bloc Capteur de pression

Le système sera équipé d'un capteur de pression différentielle qui permettra de mesurer la différence entre la pression statique et la pression totale. *"La pression statique, dans un fluide en mouvement, est la pression que mesure un capteur qui se déplace à la même vitesse que le fluide"* [3]. *"La pression totale dans un fluide (eau, air, etc.) est la somme de la pression statique, de la pression dynamique, et de la densité volumique d'énergie potentielle de gravité"* [4]. Grâce aux valeurs ainsi mesurées, il sera possible de calculer la vitesse du fluide, ce qui correspondra à la vitesse air de l'avion. Pour des vitesses inférieures à Mach 0.3 (200 noeuds), l'effet de compressibilité du fluide peut être négligé [5].

$$v(m/s) = \sqrt{\frac{2 * (p_t - p_s)}{\rho}} \quad [5] \quad (2)$$

où :

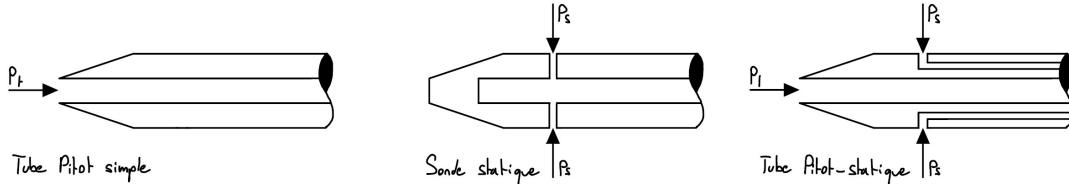
- v = Vélocité du fluide (m/s)
- p_t = Pression totale (Pa)
- p_s = Pression statique (Pa)
- ρ = Masse volumique du fluide (kg/m^3)

2.4.3 Bloc Tube Pitot

Comme expliqué au point précédent, la vitesse se calcule grâce à la pression statique et totale, pour que le capteur différentielle ait accès à ces deux pressions, il faut ajouter deux entrées d'air. Ces entrées doivent être placées correctement car dans le cas contraire, la vitesse calculée ne sera pas correcte. L'entrée de la pression statique doit être perpendiculaire à l'écoulement local du fluide (non perturbé) alors que celle de la pression totale doit être parallèle à ce flux. Il existe 3 tubes différents, le premier, appelé tube Pitot simple, est simplement un tube perforé en son centre faisant office d'entrée d'air pour la pression totale. Le second, appelé sonde statique, est perforé latéralement permettant

l'entrée d'air pour la pression statique. Le troisième est une combinaison des deux premiers, il dispose d'une entrée pour la pression totale et d'une autre pour la pression statique. L'idéal serait d'utiliser ce dernier car il simplifierait l'implémentation mécanique du système. Les deux sorties du tube seraient reliées au capteur de pression différentielle par des tuyaux. Dans le cas où un tube Pitot simple serait utilisé, il serait nécessaire de perforez le boîtier du produit afin d'avoir accès à la pression statique. La décision du type de tube sera effectuée lors du design mécanique car il n'influence aucunement la conception électronique.

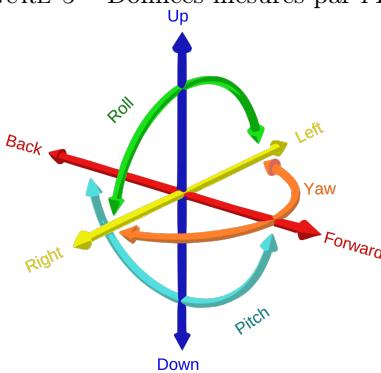
FIGURE 4 – Types de tubes Pitot



2.4.4 Bloc IMU

La centrale inertie comportera 6 capteurs, 3 gyromètres et 3 accéléromètres. *"Les gyromètres mesurent les trois composantes du vecteur vitesse angulaire (vitesses de variation des angles de roulis, de tangage et de lacet). Les accéléromètres mesurent les trois composantes du vecteur force spécifique. La force spécifique est la somme des forces extérieures autres que gravitationnelles divisée par la masse"* [1]. C'est grâce à ses capteurs qu'il sera possible de détecter le moment où l'avion commence à décrocher. La plupart de ces IMUs ont également un capteur de température intégré, la mesure de cette grandeur peut être intéressante puisqu'elle a une influence sur la valeur de la pression.

FIGURE 5 – Données mesurés par l'IMU



2.4.5 Bloc Connecteur USB

Étant donné que le système n'aura pas de source d'énergie interne, un connecteur USB-C sera implémenté afin de pouvoir tester la communication ainsi que les capteurs, en amont du vol. Il permettra également d'alimenter le système lors de son installation sur l'avion, car il devra être positionné correctement à l'horizontale pour que les mesures soient fiables.

2.4.6 Bloc Module BLE

Le transfert de données entre le produit et l'appareil Android, se fera au travers d'une communication sans fil Bluetooth Low Energy (BLE). Cette variante de Bluetooth permet de réduire significativement sa consommation par rapport au mode normal. Il sera tout de même possible qu'un module permettant de choisir le mode de fonctionnement soit implémenté. Les données transmises du produit à l'appareil Android seront principalement les valeurs lues par les capteurs alors que celle provenant de l'appareil seront plutôt des commandes ou réglages.

2.4.7 Bloc LED bleue

Comme énoncé dans le CDC, la LED bleue va permettre d'indiquer l'état du système. La LED éteinte va signifier que le système n'est pas alimenté. Un clignotement toutes les 2 secondes correspondra au système alimenté mais non appairé à un périphérique Bluetooth et finalement, 2 clignotements toutes les 2 secondes correspondra à l'état fonctionnel, appairé et transmettant les données. La LED devra être visible de jour donc son emplacement et son intensité lumineuse devront être correctement défini.

2.4.8 Bloc MCU

Le microcontrôleur sera obligatoirement un modèle 32Bit du fabricant Microchip. Il va faire le lien entre tous les périphériques implantés, c'est à dire les composants d'entrées, IMU et capteur de pression et ceux de sortie, module BLE et LED. Selon les recherches effectués précédemment, le microcontrôleur devra au minimum contenir :

- Un module UART pour la communication avec le module Bluetooth
- Un module I2C pour la communication avec l'IMU et le capteur de pression
- Une PIN GPIO pour le contrôle de la LED

2.4.9 Bloc Alimentation

Puisque la grande majorité des composants nécessaires, tel que le PIC ou les capteurs, fonctionnent avec une tension de 3.3V, il faudra donc que la sortie de ce bloc soit proche de cette valeur. Le générateur va produire une tension bien supérieur à 3.3V, donc la meilleure option est d'utiliser un convertisseur Buck afin de l'abaisser. Il permettra d'avoir un bien meilleur rendement qu'un régulateur linéaire et donc limiter la consommation.

Composant	min	max
Module BLE (consommation non continue)	5mA	10mA
Microcontrôleur	5mA	10mA
Capteur pression différentielle	3mA	4mA
LED (consommation non continue)	10mA	20mA
Centrale inertie	1mA	3mA
Total	24mA	47mA

La consommation totale devrait se situer entre 24mA et 47mA

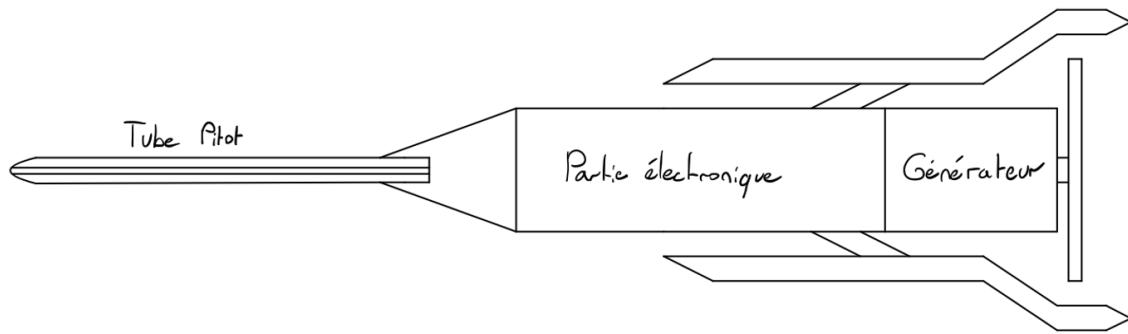
2.4.10 Bloc Appareil Android

Toutes les données obtenues au travers de la communication sans fil seront traitées et affichées sur l'écran de l'appareil. Ces données seront affichées sous forme numérique et dans l'idéal, sous forme graphiques, mais cela reste optionnel. L'application permettra d'envoyer certaines commandes au produit en appuyant sur les différents boutons qui seront affichés. Il sera aussi possible d'ajouter une valeur d'offset aux mesures pour compenser une éventuelle erreur.

2.5 Design mécanique

Comme expliqué dans le cahier des charges, l'idée est d'adapter tout le système pour qu'il puisse se trouver dans un cylindre à l'arrière du tube Pitot. Le boîtier et les autres parties mécaniques seront imprimé en une matière plastique selon la disponibilité de matériaux de l'école. L'idéal serait d'utiliser de l'acrylonitrile butadiène styrène (ABS) car ses caractéristiques mécaniques sont bien supérieures à l'acide polylactique (PLA). La grosse difficulté va être de miniaturiser le design du PCB afin de rendre la section du tube la plus mince possible. Le système de fixation sera réalisé grâce à une boule style RAM-Mounts vissé au boîtier.

FIGURE 6 – Premier croquis du design mécanique



2.6 Estimation du prix

L'estimation du prix est actuellement difficile car certains choix doivent encore être pris et la disponibilité des composants peut avoir une influence, j'ai donc mis un plage de prix.

Composant	min	max
Générateur (moteur DC)	5.-	50.-
Turbine (ABS)	0.5.-	1.-
Capteur pression différentielle	30.-	70.-
Tube Pitot	0.-	30.-
Centrale inertielle	5.-	12.-
Connecteur USB-C	1.-	3.-
Module Bluetooth	5.-	15.-
LED	0.5.-	3.-
MCU	1.-	3.-
Convertisseur Buck	3.-	8.-
Boîtier (ABS)	1.-	2.-
Boule RAM-Mounts	15.-	25.-
PCB	50.-	75.-
Total	126.-	396.-

Le prix devrait se situer entre 126.- et 396.-

2.7 Faisabilité du projet

D'après les informations trouvées, le projet est totalement réalisable. Il y aura sans doute quelques difficultés au niveau du dimensionnement de la partie turbine-générateur.

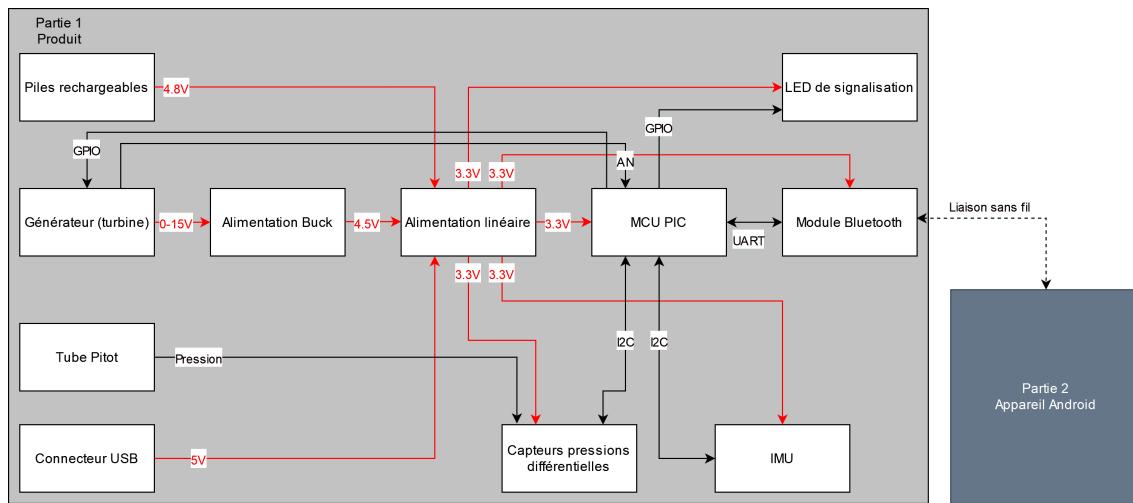
3 Design électronique

3.1 Introduction

La partie Design électronique consiste à réaliser le schéma électrique, choisir les différents composants et réaliser les dimensionnements nécessaires. Certaines modifications ont suivi la présentation de la pré-étude, ceux-ci concernent :

- La source d'alimentation
- La régulation de tension

FIGURE 7 – Schéma bloc détaillé



3.2 Générateur

Selon l'estimation de consommation totale effectuée lors de la pré-étude, il est maintenant possible de calculer approximativement la puissance minimum que le générateur devra délivrer au système.

$$P(W) = U * I \quad (3)$$

où :

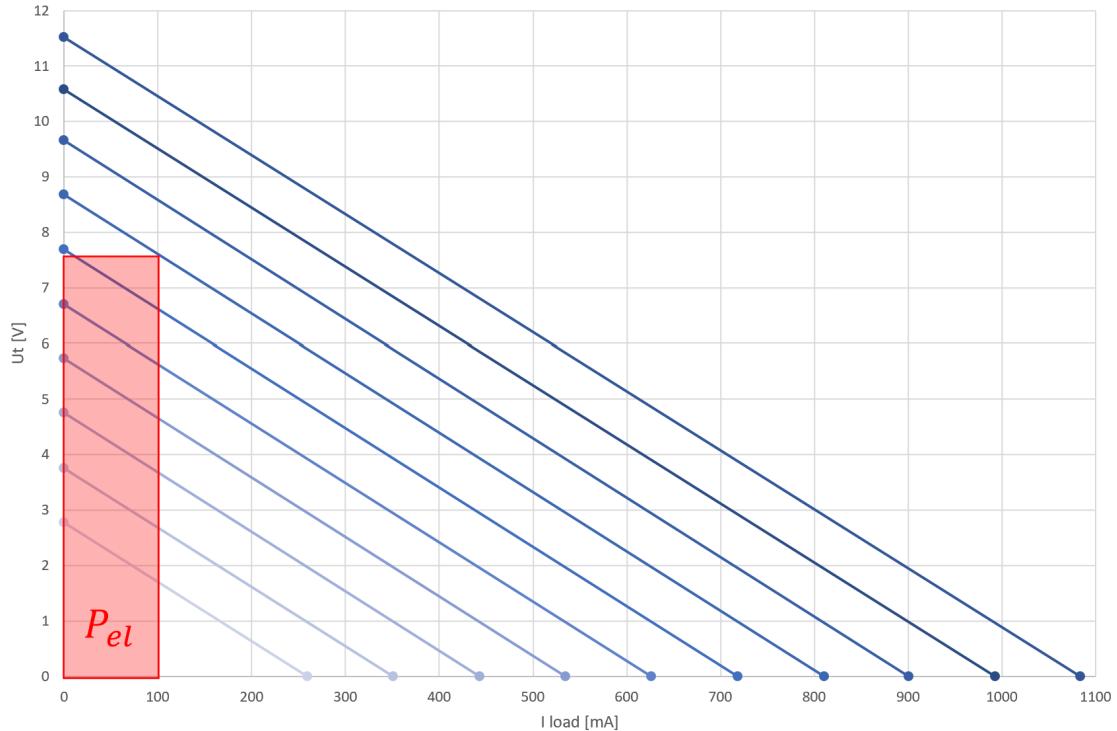
- | | | |
|-----|---|--------------------------|
| P | = | Puissance nécessaire (W) |
| U | = | Tension du circuit (V) |
| I | = | Courant consommé (A) |

Avec la formule (3) :

$$P(W) = 3.3 * 0.047 = 0.1551W = 155mW$$

Les 155mW correspondent à la puissance maximale approximative consommée après transformation, ce qui signifie qu'il y aura quelques pertes au niveau des abaisseurs de tension. Afin de ne pas avoir de mauvaises surprises, j'ai décidé de prendre une certaine marge, au lieu de 155mW, la puissance sera de 250mW. Cette puissance étant tout de même très faible, le système sera capable de fonctionner même à faibles vitesses tant que la tension minimale est atteinte. Ayant effectué des tests lors de ma pré-étude avec un moteur 3VDC et une première turbine, mon professeur M.Moreno m'en a fourni un différent, RF-370A-15370, fonctionnant jusqu'à une tension nominale de 12VDC. Pour vérifier si ce moteur pourra être utilisé, j'ai réalisé quelques mesures et calculs afin de visualiser les lignes de tension-courant du générateur.

FIGURE 8 – Lignes tension-courant du générateur



Le rectangle rouge sur la figure 8 permet de visualiser la puissance électrique. Avec celle-ci nous pouvons savoir quelle tension sera disponible au borne du moteur en fonction de la vitesse de rotation (tension à vide) et du courant consommé. Le rectangle actuel montre que la tension est d'environ 7.8V lorsque la tension à vide est de 8V (environ 3600rpm) avec un courant consommé de 100mA.

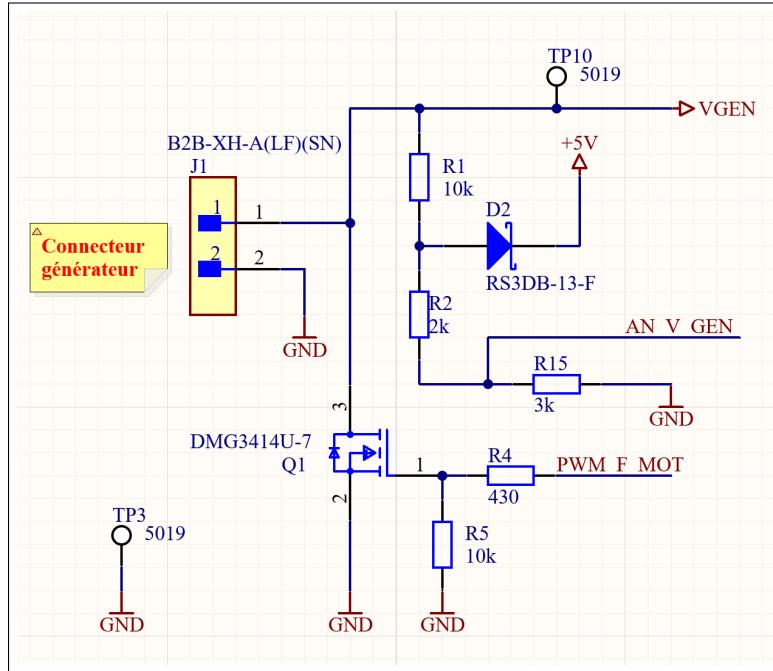
Avec la formule (3) :

$$P(W) = 7.8 * 0.100 = 0.78W = 780mW$$

Avec ce calcul, il est évident que le moteur correspond parfaitement puisque celui-ci devra fournir bien moins de puissance au système. Ce moteur là est plus adapté car il génère une tension supérieurs à la tension après régulation, 3.3VDC.

Le problème de ce moteur est qu'au moment où sa vitesse est trop élevée, cela peut le dégrader. J'ai donc décidé d'ajouter au schéma un système de freinage. La tension générée par le moteur passera par un pont diviseur avant d'entrer dans une entrée analogique du MCU. Une diode de protection permettra de court-circuiter la surtension lorsqu'elle est trop importante pour protéger le MCU. Une sortie PWM sera connectée à la gate d'un N-MOSFET afin de court-circuiter le moteur afin de le freiner lorsque la lecture de l'entrée analogique indiquera que la vitesse est trop élevée. La tension entrant dans le MCU ne pourra pas dépasser la tension max de 3.3V.

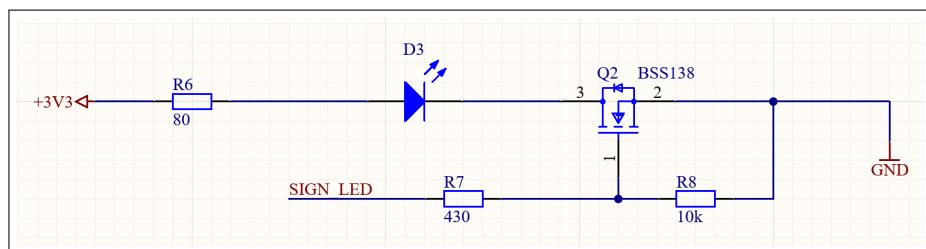
FIGURE 9 – Générateur, circuit de mesure et de freinage



3.3 LED de signalisation

La LED de signalisation sera commandée par une GPIO du MCU et permettra d'indiquer l'état du système à l'utilisateur.

FIGURE 10 – LED de signalisation



3.4 Piles rechargeables

Suite à la présentation de ma pré-étude, il a été décidé que le produit possédera deux sources d'énergie différentes, l'intérêt est de rendre le système plus flexible. La première source sera celle de base, alimentation par un générateur-turbine, et la seconde sera une alimentation à piles rechargeables. Il sera physiquement pas possible de monter les deux sources en même temps, l'idée est d'avoir un produit modulable.

Le système possédera 4 piles rechargeables NiMH AA de 1.2V chacune, connectées en série afin d'obtenir une tension de 4.8V en sortie. Celles-ci seront placées dans un support à piles cylindrique afin de respecter au maximum l'idée de design mécanique présenté dans la pré-étude (Figure 6). Aucun système de charge ne sera intégré au produit, il sera donc nécessaire d'utiliser un chargeur externe.

La durée d'utilisation totale est obtenue en effectuant la formule ci-dessous :

$$t(h) = \frac{C_A}{I} \quad (4)$$

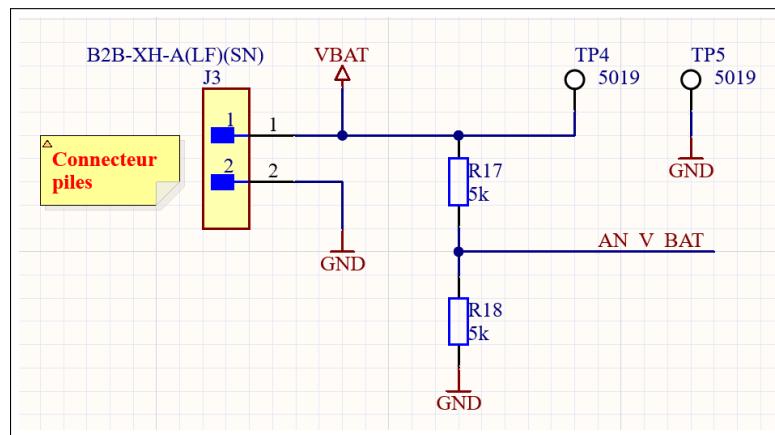
où :

- t = Durée de vie (h)
- I = Courant tiré (mA)
- C_A = Capacité batterie (mAh)

$$t(h) = \frac{2700}{50} = 54h$$

La tension d'une cellule NiMH est d'environ 1.25V chargée et de 1.1V lorsqu'elle est déchargée. De ce fait, la tension va varier entre 5V et 4.4V aux bornes du connecteur. La tension sera lue par une entrée analogique du MCU afin de connaître son état et donc avertir l'utilisateur lorsque les batteries doivent être chargées. Un diviseur de tension est nécessaire car les entrées du microcontrôleur ne supportent pas une tension de 5V.

FIGURE 11 – Connecteur piles



3.5 Capteur de pressions différentielles

Le capteur de pressions différentielles choisi est le HSCMRRN001PD2A3 du fabricant Honeywell. Il s'agit d'un capteur ayant 2 entrées d'air, une pour la pression statique et la seconde pour la pression totale. La communication se fera au moyen d'une liaison série I2C. Le composant ne nécessite pas de dimensionnement particulier au niveau hardware. Son adresse I2C est fixe et est 0b00101000 (0x28). La précision de ce capteur est de $\pm 1\%$.

Product Series	HSC	High Accuracy, Compensated/Amplified
Package	M	SMT (Surface Mount Technology)
Pressure Port	RR	Dual radial barbed ports, same side
Options	N	Dry gases only, no diagnostics
Pressure Range	001PD	± 1 psi (differential)
Output Type	2	I2C, Address 0x28
Transfer Function	A	10% to 90% of Vsupply (analog), 214 counts (digital)
Supply Voltage	3	3.3Vdc

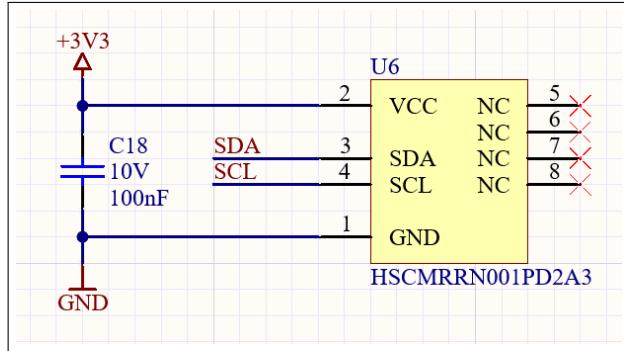
La plage de pression différentielle mesurable de ce capteur est de ± 1 psi, ce qui correspond à une pression de 6'894,76Pa.

Avec la formule (2) :

$$v(m/s) = \sqrt{\frac{2 * 6'894,76}{1.2}} = 107.197m/s = 385.91km/h$$

En utilisant la formule de la vitesse cité lors de la pré-étude, la vitesse maximale mesurable avec ce capteur est de 385.91km/h.

FIGURE 12 – HSCMRRN001PD2A3

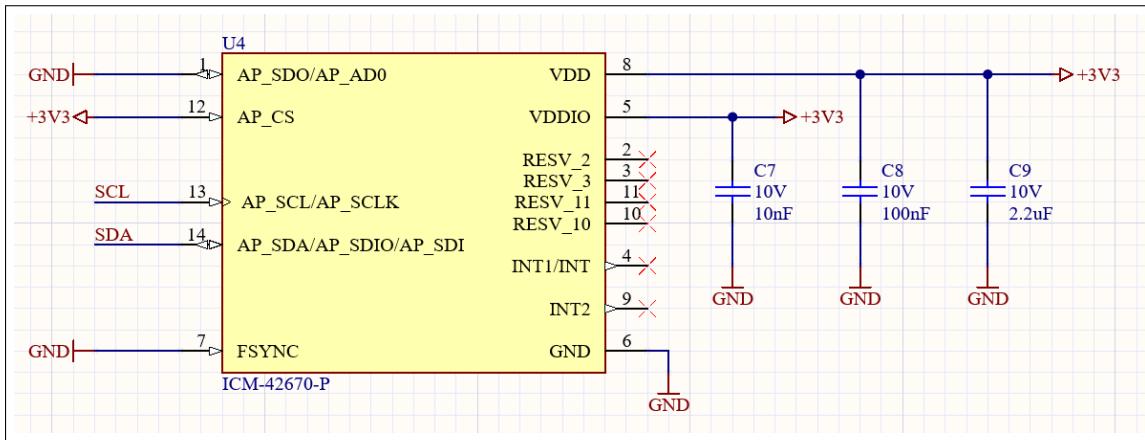


3.6 IMU

La centrale inertielles choisie est la ICM-42670-P du fabricant TDK. Il s'agit d'un composant intégrant un gyroscope 3 axes et un accéléromètre 3 axes et ayant une très faible consommation. La communication se fera au moyen d'une liaison série I2C, son adresse est b01101000 (0x68) puisque sa PIN AP_AD0 est connecté volontairement à la masse.

Au niveau du gyroscope, les axes X, Y et Z ont une plage pleine échelle programmable de ± 250 , ± 500 , ± 1000 et ± 2000 degrés/sec. La plage des accélération en X, Y et Z est également programmable de $\pm 2g$, $\pm 4g$, $\pm 8g$ et $\pm 16g$.

FIGURE 13 – ICM-42670-P



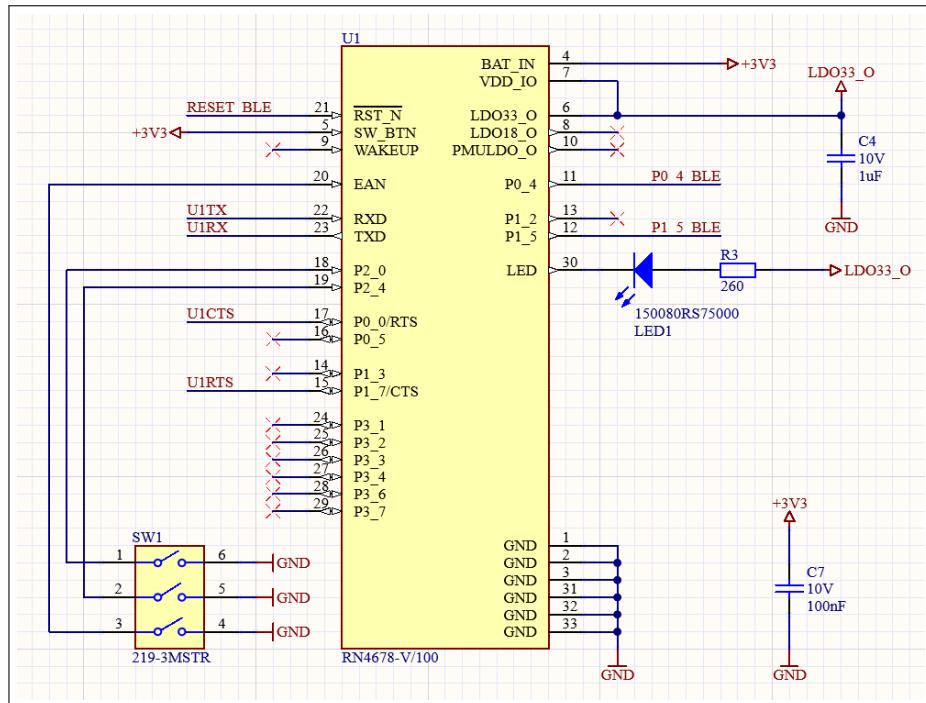
3.7 Module Bluetooth

Le module choisi, RN4678-V/100 de Microchip, possède deux technologie, le Bluetooth classique et le Bluetooth low-energy (BLE). Il est possible d'utiliser ce module en mode BLE, classique ou dual. Je l'ai sélectionné car il a été utilisé par plusieurs anciens élèves de l'ETML-ES et donc possède un driver connu et fonctionnel. Le DIP-switch SW1 à 3 entrées permet de modifier le mode opérationnel du module, c'est à dire qu'il est possible le mettre dans un mode fonctionnel, un mode écriture de l'EEPROM et test ou un mode écriture de la flash.

Le Baudrate par défaut du module est de 115200 Bauds d'où la fréquence du microcontrôleur particulière, expliquée au point MCU. Le débit du module peut aller jusqu'à 7kB/s en mode BLE et jusqu'à 32kB/s en mode classique. La LED va permettre d'afficher l'état du module lors du développement logiciel :

- Standby
- Link Back
- Low Battery
- Inquiry
- Link

FIGURE 14 – RN4678-V/100



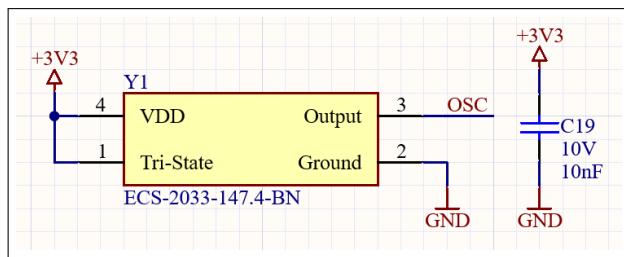
3.8 MCU

Le microcontrôleur choisi est le PIC32MX130F064B-I/SS, il possède au total 28 Pins. J'ai choisi celui-la en fonction du nombre d'entrées sorties disponibles mais également car il est en stock à l'ETML-ES. Ses principales fonctionnalités sont affichées dans le tableau ci-dessous.

PIC32MX130F064B	Valeur
Pins	28
Program Memory (kB)	64+3
Data Memory (kB)	16
Remappable Pins	20
Timers/Capture/Compare	5/5/5
UART	2
SPI/I2S	2
External Interrupts	5
Analog Comparators	3
USB On-The-Go (OTG)	No
I2C	2
PMP	Yes
10-bit 1 Msps ADC (Channels)	10
RTCC	Yes
I/O Pins	21
JTAG	Yes

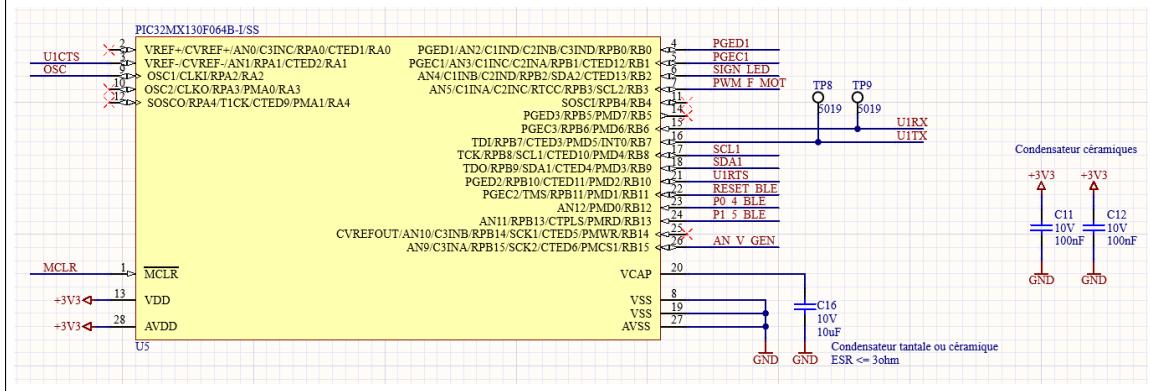
La fréquence du microcontrôleur proviendra d'un oscillateur externe de 14.7456MHz afin d'obtenir un multiple du Baudrate du module Bluetooth qui est de 115200Bd de base. 14.7456MHz correspond à $128 \times 115200\text{Bd}$.

FIGURE 15 – Oscillateur de 14.7456MHz



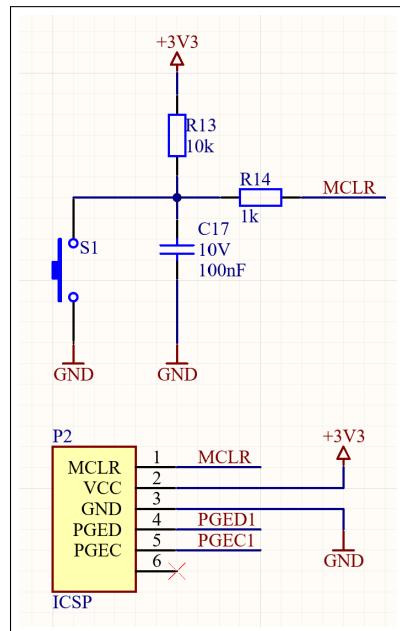
Les entrées et sorties du microcontrôleur ont été choisi à l'aide de l'utilitaire Harmony de MPLABX. Grâce à celui-ci il est plus simple de remapper les périphériques correctement.

FIGURE 16 – PIC32MX130F064B



Le circuit ci-dessous permet d'une part de programmer le microcontrôleur à l'aide d'un ICD3 ou ICD4 mais également de reset manuellement le MCU dans le cas où il y a un problème lors de la programmation.

FIGURE 17 – Circuit de reset et connecteur programmeur

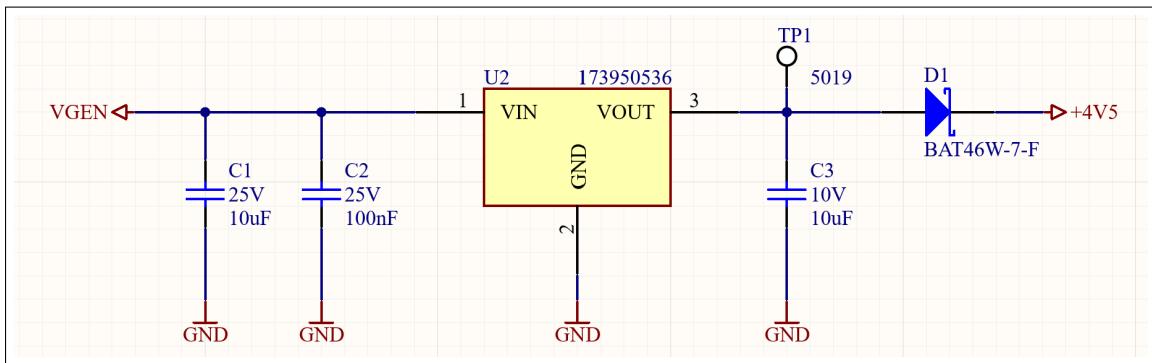


3.9 Alimentations

Le circuit d'alimentation sera composé de deux régulateurs de tension. La tension générée par la turbine entrera directement dans le premier régulateur de type Buck. La tension pouvant atteindre plus de 12V, j'ai choisi d'utiliser un régulateur Buck afin de ne pas dissiper toute la tension en chaleur. Sa sortie sera directement connectée à l'entrée du second régulateur. Celui-ci sera de type linéaire car la tension d'alimentation de la centrale inertielle ne doit pas être bruitée afin d'obtenir les mesures les plus précises possible. La diode Schottky permettra d'éviter qu'une tension n'aille dans la sortie du régulateur lorsque le module est alimenté par pile ou par USB et non pas par la turbine.

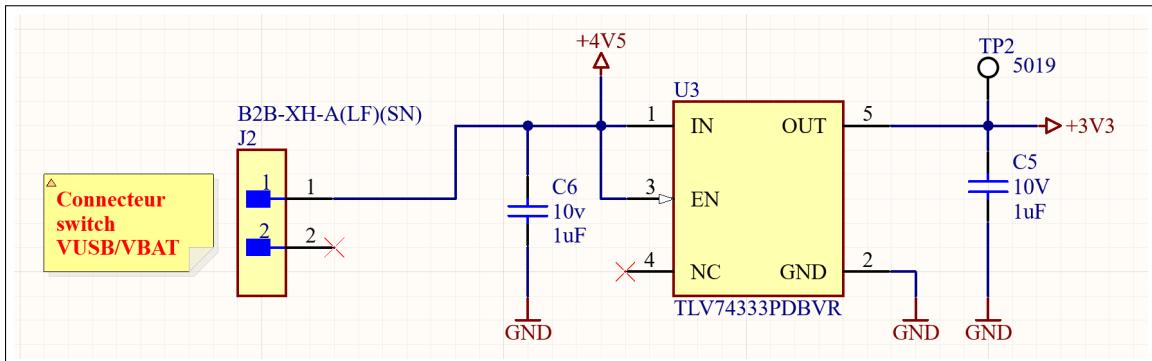
L'alimentation à découpage Wurth 173950536 est un composant intégrant l'alimentation et ses composants externes dans un même boîtier permettant ainsi de gagner de l'espace sur le PCB. La tension directe de la diode Schottky BAT45W-7-F est de 0.5V avec un courant d'environ 50mA ce qui baisse la tension entrant dans l'alimentation linéaire à 4.5V.

FIGURE 18 – 173950536



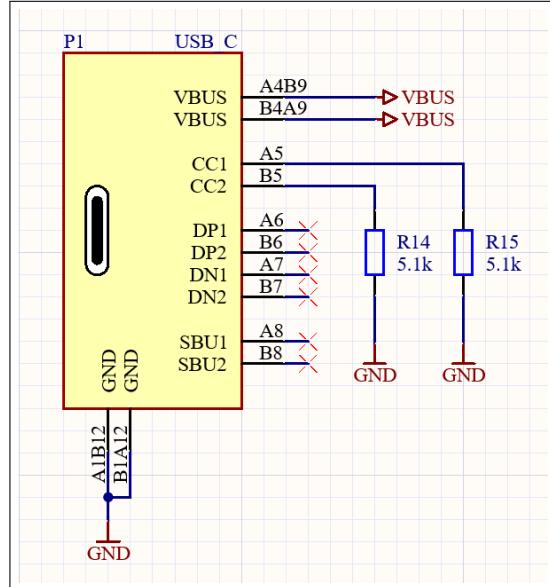
L'alimentation linéaire choisie est la TLV74333PDBVR du fabricant Texas Instrument. Son entrée sera connectée à un switch permettant de choisir soit la tension USB soit la tension des piles et comme expliqué précédemment à la sortie de l'alimentation Buck. Dans le cas où le circuit est alimenté au sol par un câble USB, le switch fera office de bouton ON/OFF.

FIGURE 19 – TLV74333PDBVR



Le connecteur USB sera de type C car c'est de nos jours le plus commun pour les nouveaux appareils. Les résistances de 5.1k branchées à la masse servent à indiquer qu'il s'agit d'un USB 2.0, bien qu'il n'y ait pas de communication.

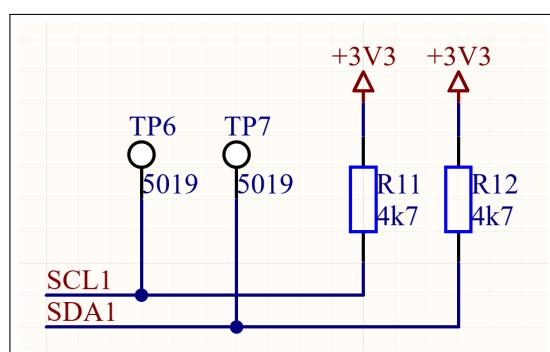
FIGURE 20 – Connecteur USB-C 2.0



3.10 Pull-up I2C

Les deux capteurs se trouvant sur le PCB utiliseront le bus I2C pour communiquer avec le MCU. Les deux lignes, SDA et SCL nécessitent l'ajout de pull-up afin de fonctionner correctement. Ces résistances peuvent avoir différentes valeurs en fonction de la vitesse de transmission. Initialement, j'ai envisagé d'utiliser des résistances de 10k ohms. Cependant, en raison de la configuration de communication en mode rapide (Fast mode), j'ai décidé de les remplacer par des résistances de 4.7k ohms.

FIGURE 21 – Pull-up I2C



4 Design mécanique

4.1 Introduction

J'ai pris la décision de fusionner le chapitre du design du circuit imprimé (PCB) avec celui du design mécanique, car ces deux aspects sont étroitement liés et cela contribue à une meilleure compréhension. Ce chapitre aborde les décisions prises lors de la conception du PCB sur Altium, notamment le placement et le routage des composants, ainsi que les choix de design du boîtier. J'ai adapté mon planning en développant simultanément la partie mécanique et celle du PCB. Cette approche m'a permis d'optimiser le volume et la résistance à l'air au maximum, tout en respectant les contraintes électriques. Le design du boîtier mécanique a été réalisé sur Onshape afin de générer des fichiers STL pour l'impression 3D.

4.2 Design

Le circuit a une forme rectangulaire mesurant 30 mm sur 120 mm, ce qui lui permet d'être inséré facilement dans une glissière prévue à cet effet dans le boîtier. Il est perforé par 2 trou de 11mm de diamètre. Ces trous permettent de faire passer deux entretoises métalliques et vis afin de fixer une boule RAM-Mounts au boîtier. Ce mode de fixation est connu et a déjà été testé par Monsieur Moreno, qui m'a donc conseillé de l'utiliser dans ce projet. Ce système de fixation offre une grande flexibilité en permettant de monter le système dans pratiquement toutes les positions et angles possibles tout en garantissant une grande fiabilité.

Sur ce circuit, les composants ont été disposés de manière à prendre en compte leur sensibilité aux perturbations. Ainsi, les capteurs sont placés à l'extrême avant afin d'être alimentés par une source de tension propre, ce qui garantit des mesures fiables. D'autre part, la partie bruyante comprenant la MCU et l'alimentation à découpage est positionnée de l'autre côté. Le placement du capteur de pression différentielle à l'avant, permet également de réduire la longueur des tuyaux allant jusqu'au tube Pitot qui se trouve à l'extrême avant du boîtier.

FIGURE 22 – Circuit vue de dessus

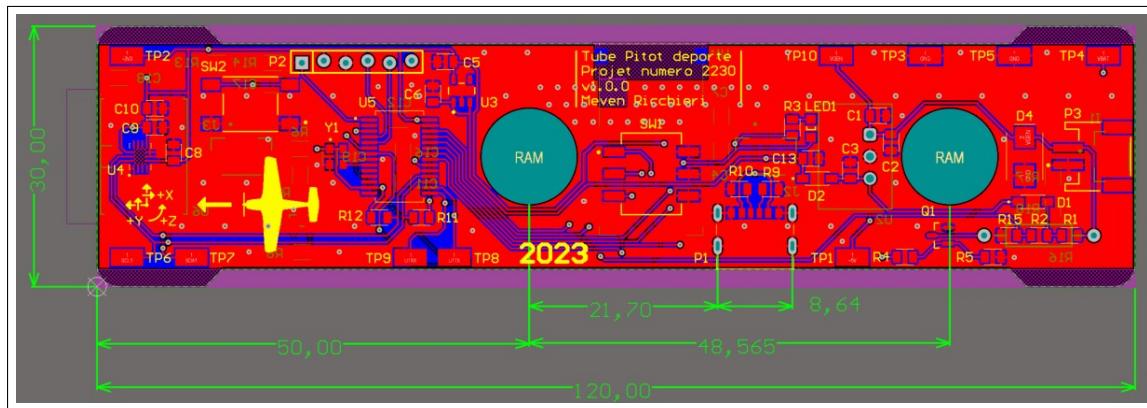
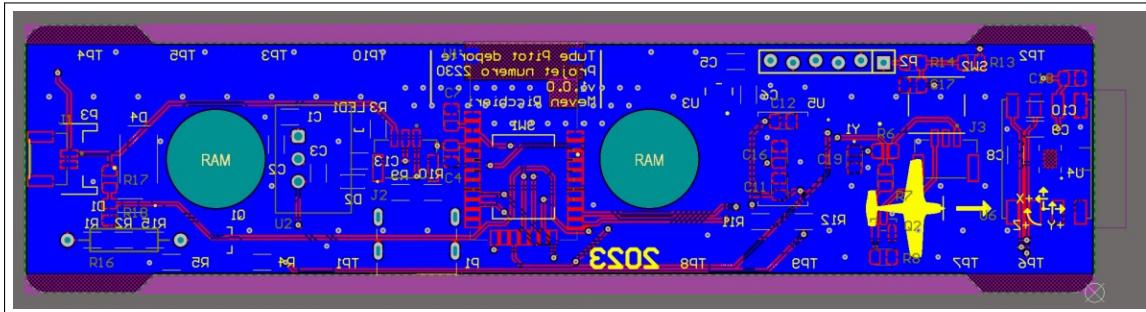
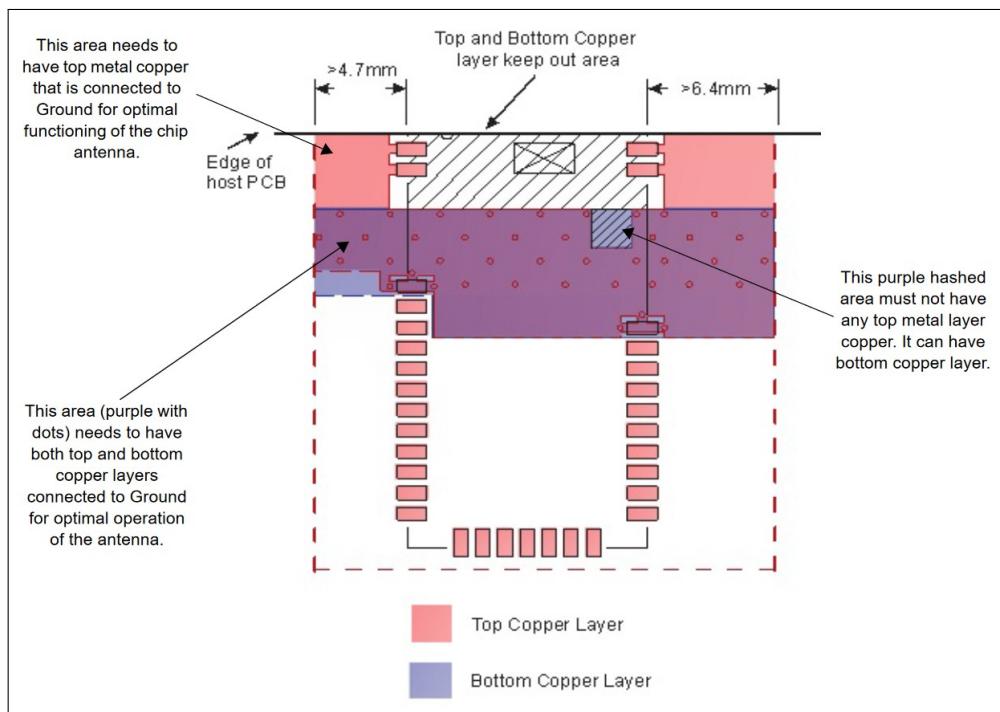


FIGURE 23 – Circuit vue de dessous



Concernant le module Bluetooth, celui-ci à du être placé de sorte à ce que son antenne soit en bordure de PCB afin d'optimiser sa portée. Le design suggéré par Microchip à également été utilisé afin de rendre le système plus robuste. Sur la Figure 24, se trouve le design suggéré par le fabricant, se trouvant dans la documentation technique du module Bluetooth RN4678.

FIGURE 24 – PCB suggéré par Microchip



Au niveau des largeurs de piste, les alimentations (VGEN, VBUS, VBAT, GND) ont une largeur de 0.3mm. Toutes les autres pistes ont une largeur de 0.254mm. Le circuit étant spécifiquement conçu pour une consommation de courant minimale, ne dépassant pas 50-60mA, les pistes sont largement surdimensionnées.

Les connecteurs sont placés de sorte à ce que le câblage soit le plus simple plus possible. Les 2 connecteurs responsables de l'alimentation se situent à l'arrière du circuit, de ce fait, ils sont au plus proche de leur source d'énergie et donc facilite le branchement. Le connecteur de la LED de signalisation et celui du switch ON/OFF sont situés sur la partie basse du PCB puisqu'ils se situent physiquement sur la partie basse du boîtier. Le connecteur USB type C est quant à lui sur le coté, ce qui permet d'y avoir accès lorsque le PCB est intégré dans le boîtier.

FIGURE 25 – Circuit vue de dessus

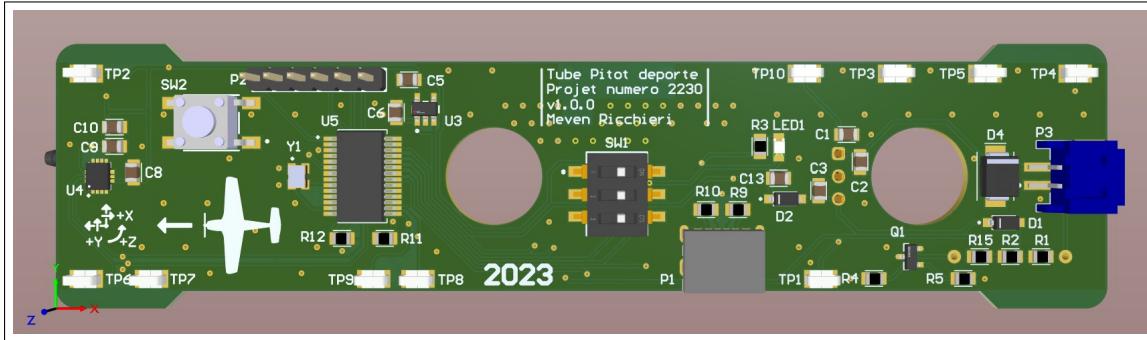
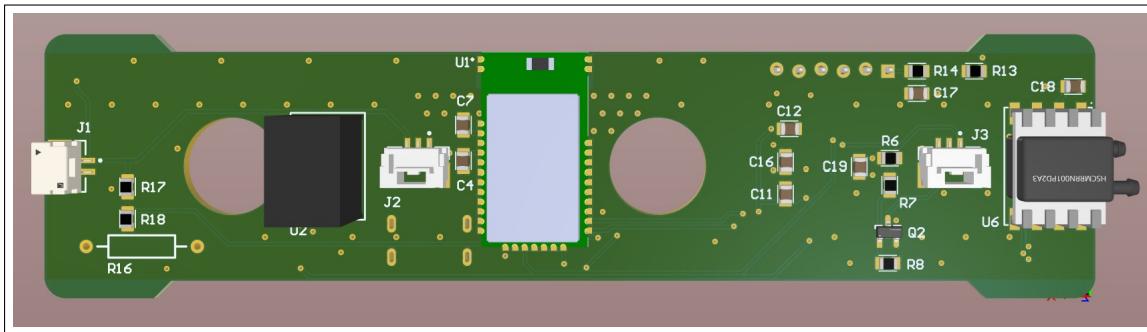


FIGURE 26 – Circuit vue de dessous



Sur la Figure 27 et sur la Figure 28, il est possible de voir comment le système complet s'intègre dans le boîtier avec les deux types d'alimentation, turbine et batteries. Les deux assemblages sont affichés en transparence, il est donc facile d'apercevoir toutes les parties du système. Les éléments importants sont énumérés ci-dessous :

1. Tube Pitot
2. Vis M2.5 permettant de fixer la partie A et B
3. Vis M5 avec entretoise de 11mm
4. Boule RAM-Mounts
5. Connecteur VGEN (tension venant du générateur)
6. Vis M2.5 permettant de fixer la partie B et C
7. Moteur DC (générateur)
8. Axe du moteur permettant de fixer la partie C et D
9. 2 Vis M3 permettant de fixer le moteur DC 7 au boîtier C
10. Switch ON/OFF
11. Connecteur VBAT (tension venant des batteries)
12. Connecteur switch ON/OFF
13. Connecteur USB type C
14. Circuit imprimé
15. Connecteur LED signalisation
16. LED de signalisation

FIGURE 27 – Assemblage complet avec turbine

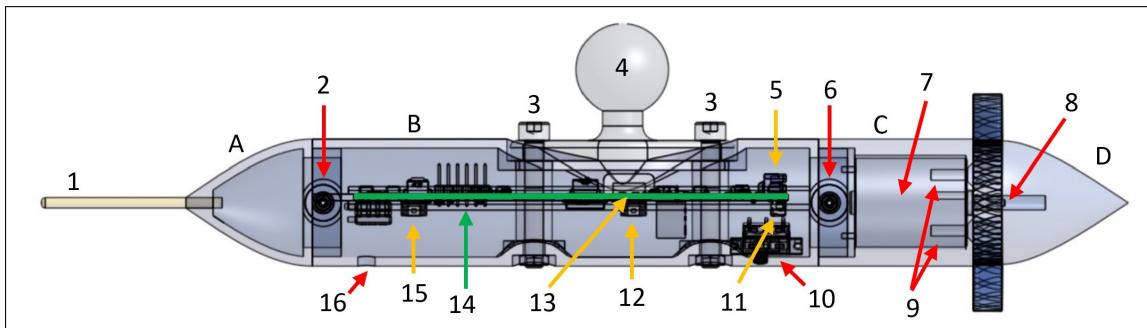
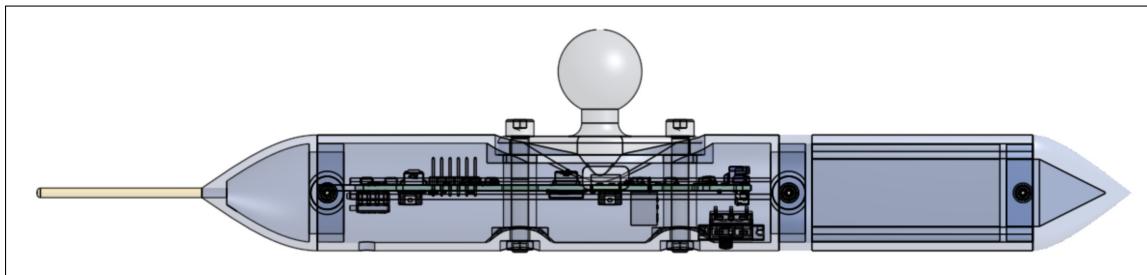
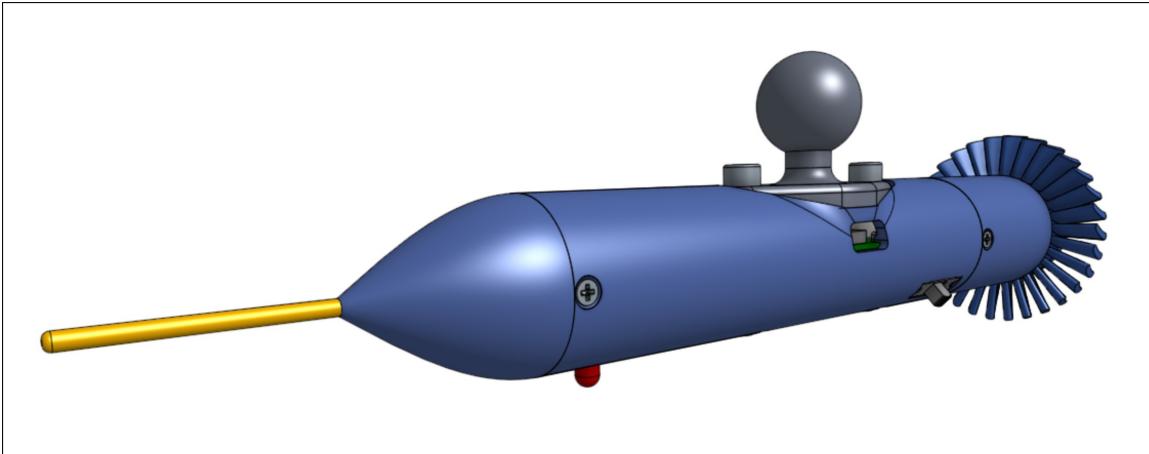


FIGURE 28 – Assemblage complet avec batteries



La Figure 29 présente le rendu final du système, intégré dans son boîtier. L'intégration est particulièrement simple et efficiente.

FIGURE 29 – Assemblage final



4.3 Modifications

Le design comporte quelques erreurs mineures qui doivent être corrigée dans le cas d'une nouvelle itération de ce projet en utilisant les mêmes documents Altium. Ces détails sont décrits ci-dessous.

4.3.1 Circuit imprimé

Au niveau du circuit imprimé, le footprint du bouton reset dénommé SW2 n'est pas le bon. En conséquent, un switch différent a été brasé sur le PCB. Sans modification, le circuit fonctionnera tout de même. Le second point à corriger concerne les lignes de la communication USART. L'erreur va jusqu'à la modification du schéma, les PINs 21 (RESET_BLE) et 22 (U1RTS) du MCU doivent être intervertis afin de pouvoir profiter du contrôle de flux hardware. Sur la version actuelle, il est fonctionnel mais nécessite un contrôle software manuel en activant et désactivant la PIN U1RTS, raison pour laquelle il n'est pas utilisé.

4.3.2 Boîtier

Au niveau du boîtier, la partie turbine pose un problème au niveau de sa fixation à l'axe du moteur. Par manque de temps, cette partie n'a pas encore été résolu et donc nécessite de la recherche. Une amélioration serait également de passer de 4 batteries 1.5V à une seule batterie différente afin de rendre le système plus léger et plus équilibré entre l'avant et l'arrière.

5 Mise en service

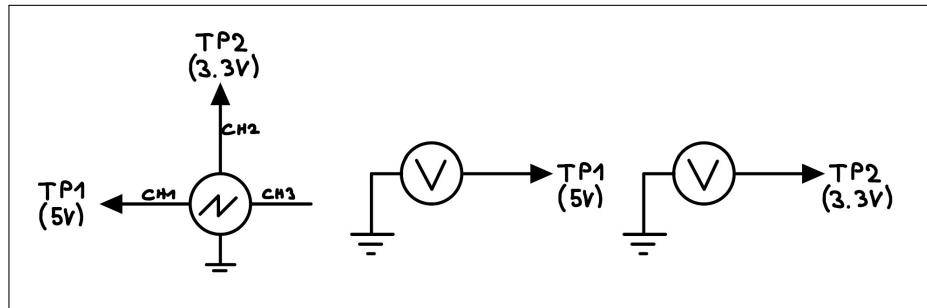
5.1 Introduction

Dans cette partie, j'ai réalisé une série de tests dans le but de valider le circuit imprimé (PCB). Notamment les sorties des alimentations, le courant consommé et le test de tous les périphériques telles que le module Bluetooth et les différents capteurs.

5.2 Test des alimentations

Les deux alimentations présentes sur le circuit fonctionnent correctement, le régulateur Buck 5V a une tension de sortie de 4.97V et le régulateur linéaire 3.3V a une tension de sortie de 3.29V. J'ai également contrôlé la qualité des sorties en mesurant la sortie de ces alimentations à l'oscilloscope.

FIGURE 30 – Schéma de mesure



5.3 Consommation

La consommation du circuit lorsque le MCU ne tourne pas est de 22.05mA avec la tension d'alimentation provenant de VBAT (6V). Lorsque le circuit est alimenté par VGEN (12V), la consommation est de 12.13mA.

5.4 Périphériques

Une fois les diverses configurations Harmony, décrites dans la section suivante de ce document, effectuées, j'ai procédé au test du capteur de pression différentielle, de la centrale inertuelle et du module Bluetooth. Tous ces composants communiquent correctement et renvoient des valeurs. Les mesures qui suivent ont été effectuées à l'aide d'un analyseur logique Saleae branché sur les points TP8 (PIC TX) et TP9 (PIC RX) pour l'USART et TP6 (SCL) et TP7 (SDA) pour l'I2C.

5.4.1 Signaux USART

Sur les deux mesures ci-dessous, on peut observer que l'intégrité des signaux USART est très bonne. Sur la Figure 31, il s'agit des signaux provenant du microcontrôleur et allant au module Bluetooth. En utilisant la formule 5 il est possible de calculer la vitesse de la communication. 4 Bytes sont reçus en l'espace de 339.16us ce qui correspond à une vitesse de 114989 bit/s. L'erreur de vitesse est calculable à l'aide de la formule 6. L'erreur de Baudrate du PIC est d'environ 0.18% par rapport à la vitesse désirée ce qui est très clairement bon.

Sur la Figure 32, il s'agit des signaux allant du module Bluetooth au PIC. 5 Bytes sont reçus en l'espace de 416.48us ce qui correspond à une vitesse de 117653 bit/s. L'erreur de vitesse est d'environ 2.13% par rapport à la vitesse voulue. Ce pourcentage d'erreur n'est pas exceptionnel et ne peut malheureusement pas être corrigé. Cependant, malgré cette légère erreur, la communication fonctionne correctement et ne pose pas de problème significatif.

$$\text{Baudrate}[bit/s] = \frac{N_{bit}}{T_{tot}} \quad (5)$$

$$\text{erreurBaudrate}[\%] = \frac{Bd_{voulu} - Bd_{mesur}}{Bd_{mesur}} * 100 \quad (6)$$

FIGURE 31 – Mesure trame USART PIC → BT

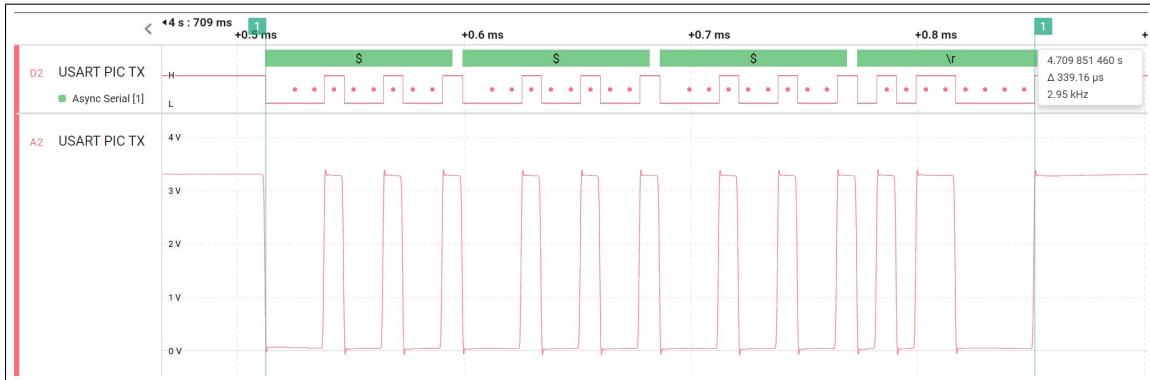
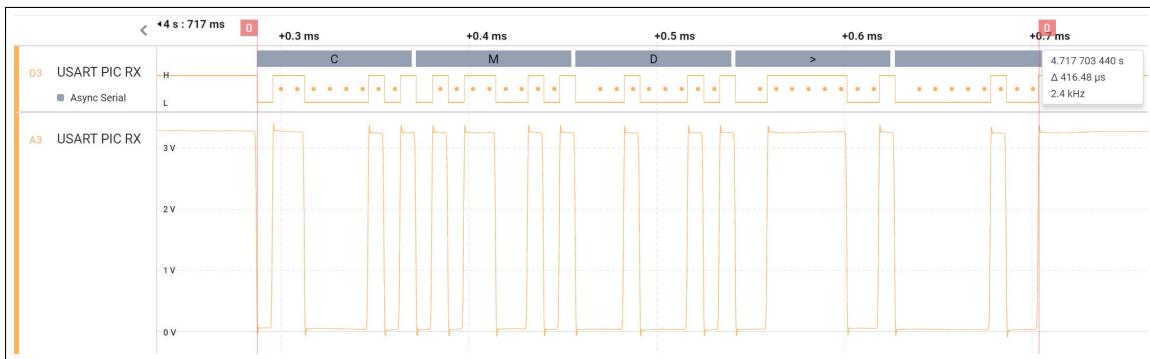


FIGURE 32 – Mesure trame USART BT → PIC



5.4.2 Signaux I2C

Sur les mesures qui suivent, nous pouvons observer que l'intégrité des signaux I2C n'est pas exceptionnelle. La partie haute des signaux n'atteint pas tout à fait la tension VCC et la partie basse n'atteint pas non plus le GND. La solution à ce problème est soit de diminuer vitesse en passant de 400kHz à 100kHz ou en réduisant la valeur des pull-up. Toutefois la communication fonctionne parfaitement car les seuils de déclenchement sont quand même franchis. Bien que cela fonctionne, j'ai décidé de réduire la vitesse de communication. L'effet de ce changement est visible sur la Figure 35.

FIGURE 33 – Mesure trame I2C à 400kHz

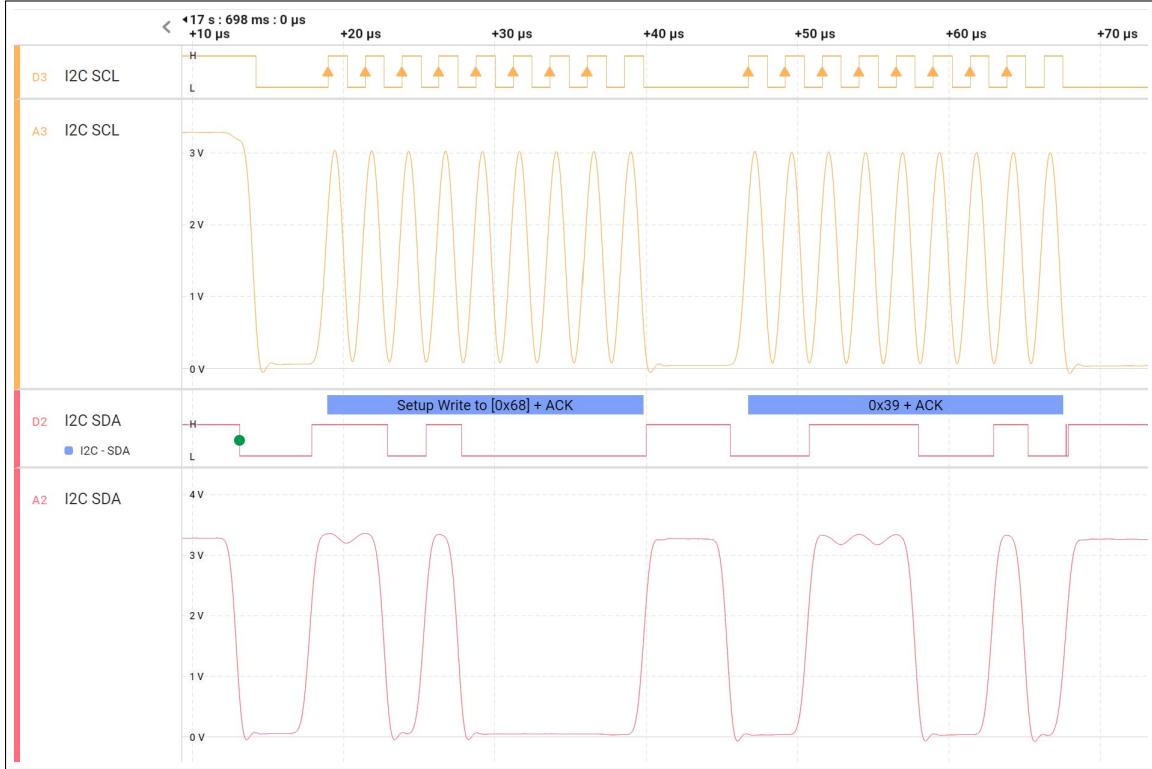


FIGURE 34 – Mesure d'une période SCL à 400kHz

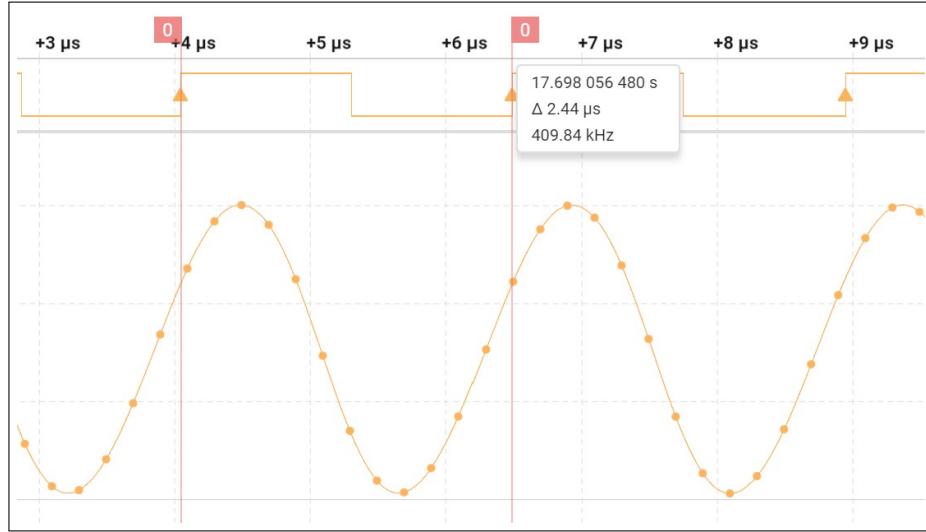
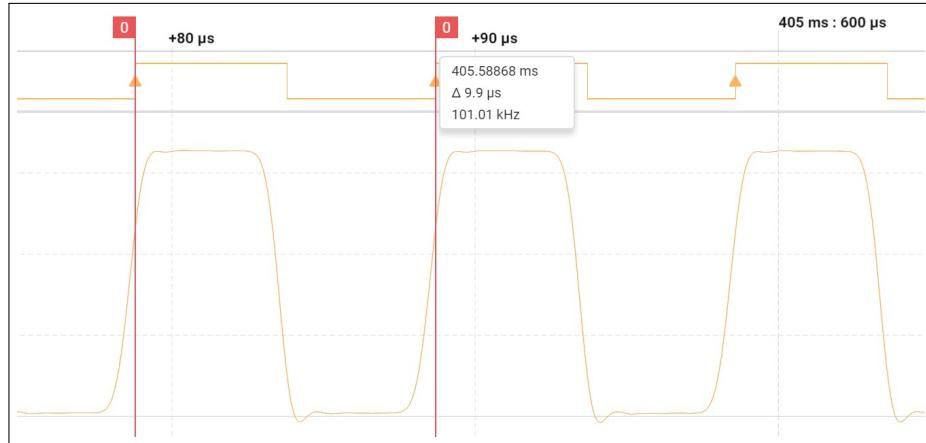


FIGURE 35 – Mesure trame I2C à 100kHz



5.5 Améliorations possibles

Après avoir effectué plusieurs tests d'alimentation du circuit à l'aide de la turbine, un problème potentiel peut se présenter. Lorsque la turbine commence à générer une tension suffisamment élevée pour activer les alimentations, le courant disponible peut ne pas être suffisant pour alimenter tous les composants. Par conséquent, cela entraîne une chute de tension et peut provoquer le blocage du MCU. L'idée serait d'implémenter un amorceur de démarrage à l'aide d'un comparateur déclenchant un MOSFET.

6 Software MCU

6.1 Introduction

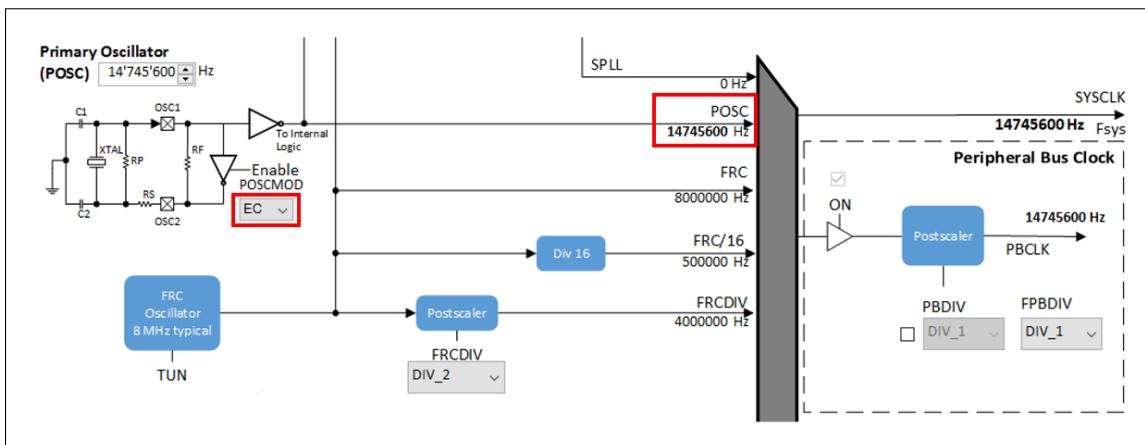
Le développement du logiciel du MCU a pris considérablement plus de temps que prévu, principalement en raison de la complexité rencontrée lors de l'intégration du driver de la centrale inertuelle, celle-ci composée de nombreux fichiers et étant pas très compréhensible. L'emplacement du projet se situe sous : C:/microchip/harmony/v2_06/apps/PROJ

6.2 Paramétrage Harmony

6.2.1 Fréquence d'horloge

Comme décrit dans la pré-étude, le microcontrôleur est cadencé par un oscillateur externe à une fréquence de 14.7456MHz. Cela permet d'optimiser la communication USART puisque cette fréquence est un multiple de 115200Hz ($115200 * 128 = 14745600\text{Hz}$), fréquence à laquelle le MCU communique avec le module Bluetooth. Le diagramme d'horloge présenté dans la Figure 36 illustre la configuration du microcontrôleur. La fréquence du système (SYSCLK) provient de l'oscillateur primaire (POSC) en tout en spécifiant qu'il s'agit d'un oscillateur externe (EC).

FIGURE 36 – Diagramme d'horloge



6.2.2 Timers

Le calcul utilisé afin d'obtenir le nombre de ticks (Timer period) des trois timers utilisés est le suivant :

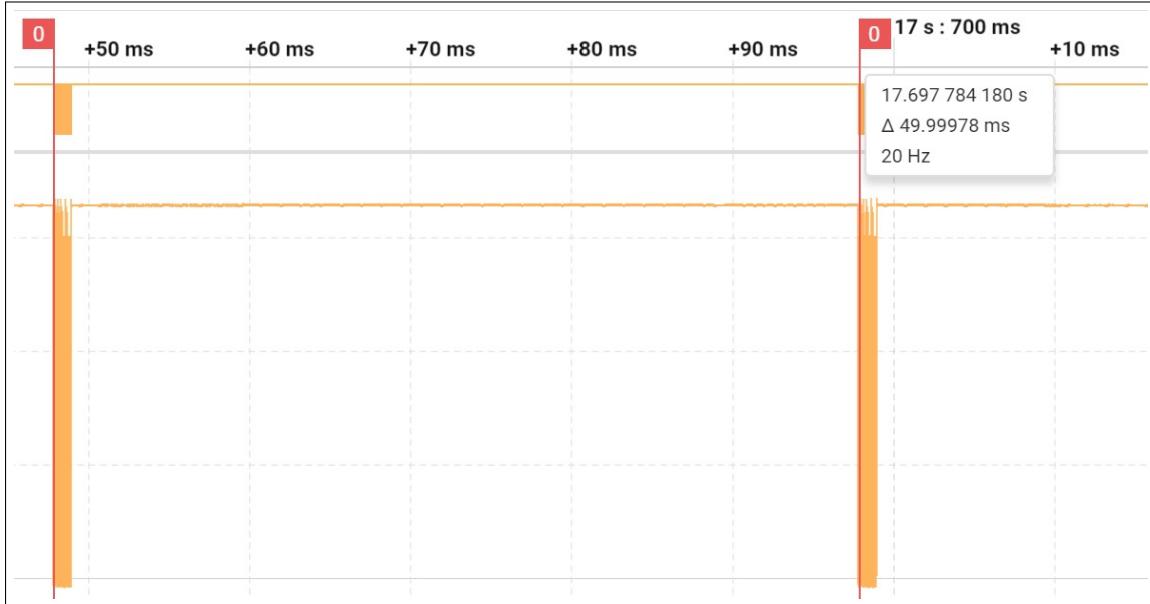
$$N_{Ticks}[-] = \frac{f_{SYS}}{f_{Timer} * PRESCL} - 1 \quad (7)$$

	Timer1	Timer2	Timer3
Instance	1	2	3
ID	2	4 et 5	3
Mode	16 bit	32 bit	16 bit
Fréquence	20Hz	3.43mHz	225Hz
Prescaler	256	1	1
Timer period	2879	4294967295	65535
Interruption priorité	Level 2	Level 0	Level 0
Enabled	YES	YES	NO

Le Timer1 a pour but de cadencer le programme principal, c'est à dire que la machine d'état principale est exécutée 20 fois par secondes. Le Timer2 est utilisé dans les fonctions de gestion de temps, nécessaire au bon fonctionnement de l'IMU et au module Bluetooth. Le Timer3 est quant à lui désactivé, il a été utilisé afin de réaliser différents tests.

Sur la Figure 37, nous pouvons voir que les communications s'effectuent bien toutes les 50ms (20Hz).

FIGURE 37 – Mesure cadence Timer1



6.2.3 Entrées sorties

Les PINs ont été configurées de sorte à respecter le schéma du circuit. Ci-dessous (Figure 38), se trouve la configuration de toutes les entrées et sorties du microcontrôleur. Dans la colonne "Function" nous pouvons voir à quel périphérique sont connectés les PINs. Les fonctions nommées "Available" sont des PINs son utilisées.

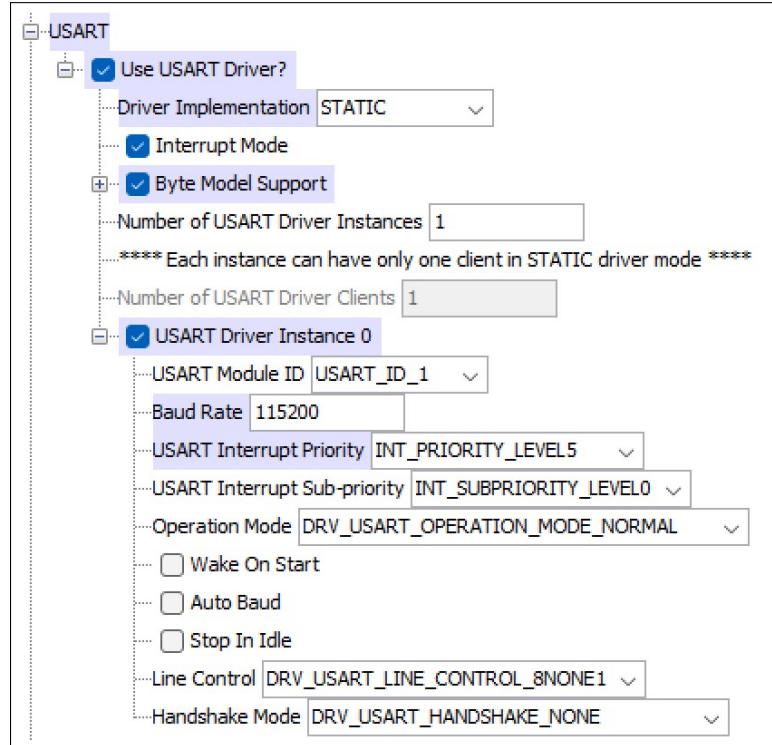
FIGURE 38 – Paramétrage des PINs

Pin Number	Pin ID	Voltage Tolerance	Name	Function	Direction (TRIS)	Mode (ANSEL)
1	MCLR	5V			In	Dig...
2	RA0			Available	In	An...
3	RA1		U1CTS_MANUAL_PIC_IN	GPIO_IN	In	Dig...
4	RB0			Available	In	An...
5	RB1			Available	Out	Dig...
6	RB2		SIGN_LED	GPIO_OUT	Out	Dig...
7	RB3			Available	In	An...
8	VSS				In	Dig...
9	RA2			Available	In	An...
10	RA3			Available	In	An...
11	RB4		P0_4	GPIO_IN	In	Dig...
12	RA4		P1_5	GPIO_IN	In	Dig...
13	VDD				In	Dig...
14	RB5	5V	TEST_OUT	GPIO_OUT	Out	Dig...
15	RB6	5V	U1RX	U1RX	n/a	Dig...
16	RB7	5V	U1TX	U1TX	n/a	Dig...
17	RB8	5V	SCL1	SCL1	n/a	Dig...
18	RB9	5V	SDA1	SDA1	n/a	Dig...
19	VSS				In	Dig...
20	VCAP				In	Dig...
21	RB10	5V	RESET_BLE	GPIO_OUT	Out	Dig...
22	RB11	5V	U1RTS_MANUAL_PIC_OUT	GPIO_OUT	Out	Dig...
23	RB12			Available	In	An...
24	RB13			Available	In	Dig...
25	RB14		AN_V_BAT	AN10	n/a	An...
26	RB15		AN_V_GEN	AN9	n/a	An...
27	AVSS				In	Dig...
28	AVDD				In	Dig...

6.2.4 Communications série

La communication USART a été configurée depuis l'utilitaire Harmony. Les paramètres sont affichés dans la Figure 39.

FIGURE 39 – Paramétrage de la communication USART



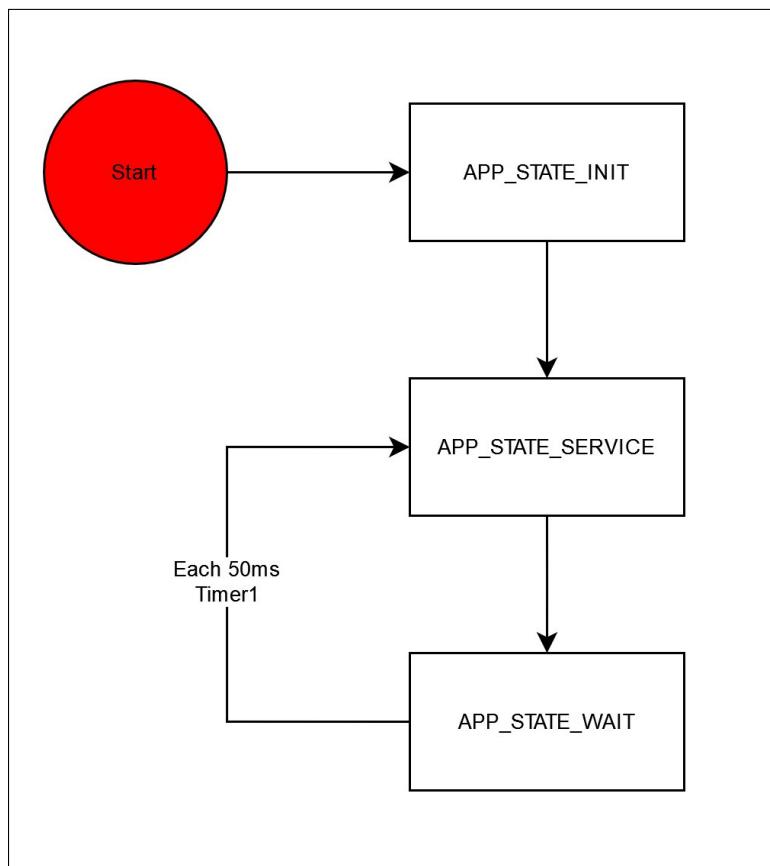
L'I2C à quant-à lui n'a pas été configuré depuis Harmony mais à l'aide du fichier C nommé "Mc32_I2cUtilCCS". Le paramètre principal, la vitesse, est réglée en SLOW mode (100kHz) puisque le FAST mode pose quelques problèmes de distorsion.

6.3 Flowcharts

6.3.1 Machine d'état principale

La Figure 40 présente le diagramme de flux de la machine d'état principale, qui est cadencée par le Timer1 à une fréquence de 20Hz. Au début, tous les périphériques du système sont initialisés avant d'entrer pour la première fois dans l'état de service. Après chaque passage dans l'état de service, le programme entre dans l'état d'attente, où le MCU ne fait rien, jusqu'à ce que l'interruption du Timer1 déclenche à nouveau l'état de service.

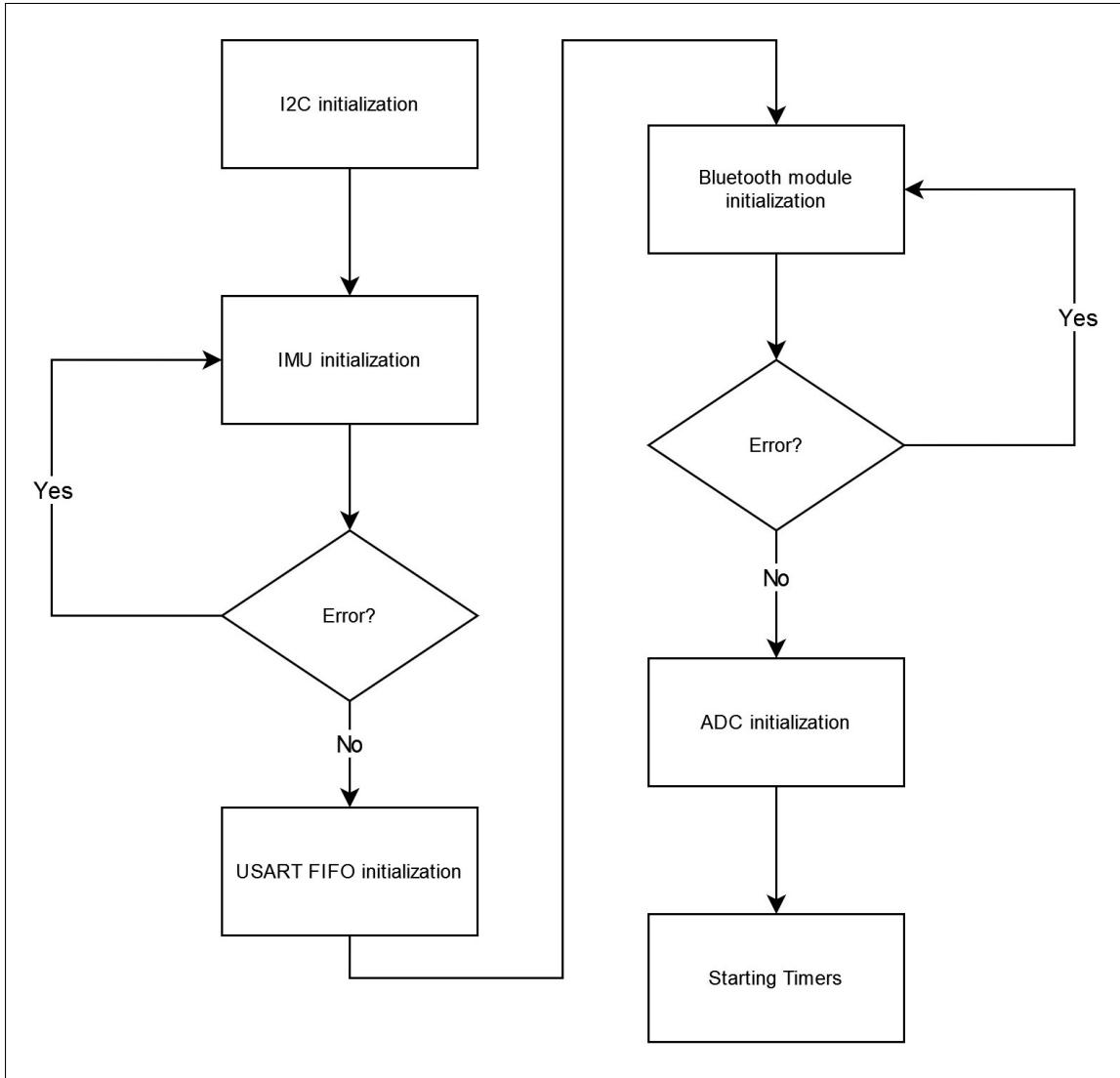
FIGURE 40 – Machine d'état principale



6.3.1.1 Etat d'initialisation

La Figure 41 montre toute la phase d'initialisation du système. Les deux composants principaux bloquent le programme tant qu'ils ne sont pas correctement initialisé. Cela permet de garantir que le système fonctionne correctement.

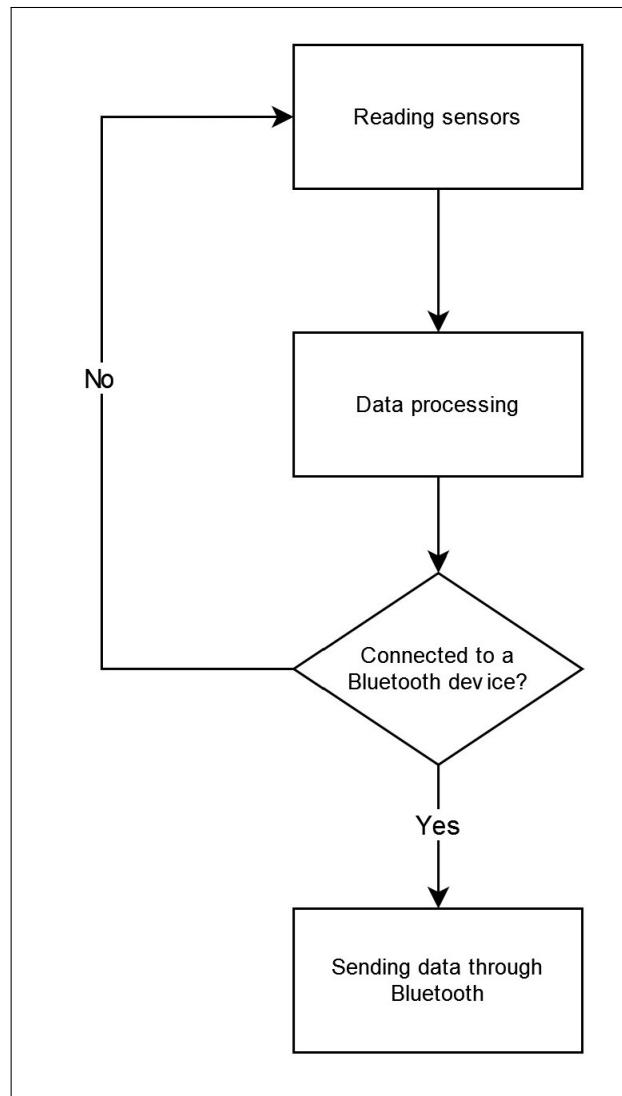
FIGURE 41 – Etat d'initialisation



6.3.1.2 Etat de service

L'état de service contient une sous-machine d'état composée de trois autres états. Le premier état a pour but de lire les capteurs, le suivant effectue les traitements nécessaires, et enfin le dernier état envoie les données au module Bluetooth qui les transmet à l'appareil Android connecté.

FIGURE 42 – Etat de service



6.4 Codes C

La mise en forme de la trame ce fait dans cette fonction ci-dessous et permet de faciliter le traitement des données aux niveau de l'application Android.

```
inline void frameFormatting(char* a_dataToSend, const SENS_DATA* sensData){

    // Saves all data into a simple frame
    // Speed in [km/h]
    // Gyros in [dps]
    // Angles in [degrees]
    // Accelerations in [g]
    // VB and VG in [V]
    sprintf(a_dataToSend, "S=%03d GX=%+.02f GY=%+.02f GZ=%+.02f GAX=%+.02f "
            "GAY=%+.02f GAZ=%+.02f AX=%+.02f AY=%+.02f AZ=%+.02f VB=%+.02f "
            "VG=%+.02f\n\r",
            sensData->velocity,
            sensData->gyroX, sensData->gyroY, sensData->gyroZ,
            sensData->GyrAngleX, sensData->GyrAngleY, sensData->GyrAngleZ,
            sensData->accelX, sensData->accelY, sensData->accelZ,
            sensData->batVoltage, sensData->genVoltage);
}
```

La fonction ci-dessous permet de lire les statuts envoyé par le module Bluetooth. Pour l'instant elle effectue uniquement un

```
void USART1_Callback_Function(void){

    char a_received[50];
    char* result;

    // Gets new data from FIFO
    getUsartData(&a_received[0]);

    // If "<RFCOMM_OPEN>" is present in the array received
    result = strstr(a_received, "<RFCOMM_OPEN>");
    if(result != NULL){

        appData.isBluetoothDiscoverable = false;
        appData.isBluetoothConnected = true;
    }

    // If "<RFCOMM_CLOSE>" is present in the array received
    result = strstr(a_received, "<RFCOMM_CLOSE>");
    if(result != NULL{

        // turnOnDiscoverBT();
        appData.isBluetoothConnected = false;
    }
}
```

La fonction de callback ci-dessous est appelée après chaque lecture de la centrale inertuelle. Elle est appelée automatiquement par le driver fourni par le fabricant (TDK InvenSense) lorsque les valeurs sont disponibles dans la structure "event". Des offsets on été ajoutés manuellement

```
void imu_callback(inv_imu_sensor_event_t *event){

    // Transforms 16bits values into degrees and saves them in the sensor data
    // structure
    // 250 dps
    sensData.gyroX = (event->gyro[0])/250 + 1; // + offset
    sensData.gyroY = (event->gyro[1])/250 + 1; // + offset
    sensData.gyroZ = (event->gyro[2])/250;

    // Reads and transforms 16bits values into
    // Transforms 16bits values into g acceleration and saves them in the sensor
    // data structure (8192 bits per g)
    sensData.accelX = (float)(event->accel[0])/8192.0;
    sensData.accelY = (float)(event->accel[1])/8192.0;
    sensData.accelZ = (float)(event->accel[2])/8192.0 + 0.075; // + offset

    // Calculate the angle with the gyro values
    // 0.05 correspond to the period between each reading 1/20Hz = 0.05s
    sensData.GyrAngleX += sensData.gyroX * 0.05;
    sensData.GyrAngleY += sensData.gyroY * 0.05;
    sensData.GyrAngleZ += sensData.gyroZ * 0.05;
}
```

Le reste du code se trouve en annexe. Dans ce sous chapitre, quelques fonctions importants sont brièvement expliquées. Le reste du code se trouve en annexe car tout est bien commenté et facile à comprendre.

7 Software Android

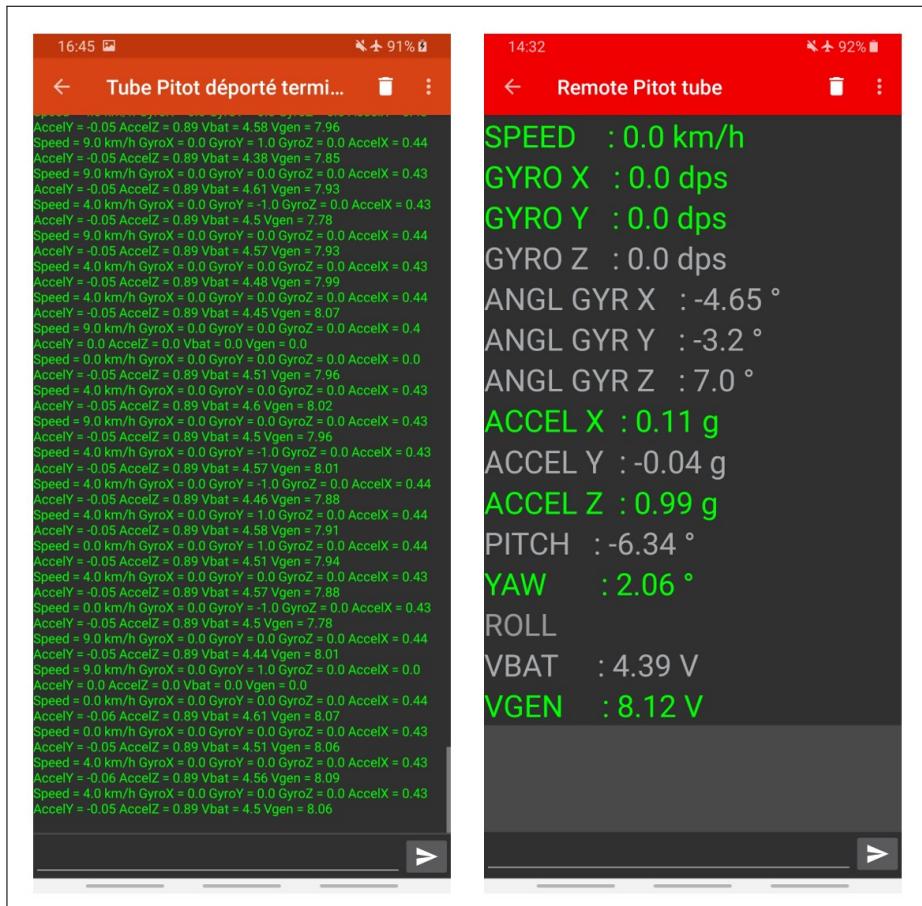
7.1 Introduction

Cette partie de rapport n'est pas documenté car l'application Android n'est pas terminé et nécessite encore un certain temps de travail. En premier lieu, J'ai penser développer une application de A à Z. Cependant la difficulté était bien supérieure à celle à laquelle je m'attendais. En conséquent, j'ai fork le repos github d'un développeur ayant réalisé un terminal Bluetooth afin de modifié son application. Le lien de son github : <https://github.com/kai-morich/SimpleBluetoothTerminal>

Un aperçu des deux applications est disponible ci-dessous. La capture d'écran de gauche correspond à l'application de base provenant de github et celle de droite est celle modifié pour le projet.

7.2 Aperçu des interfaces

FIGURE 43 – Aperçu des deux applications



8 Résultat final

8.1 Introduction

Dans ce chapitre, il est possible de voir à quel point le projet est avancé et ce qu'il reste à réaliser pour qu'il soit totalement abouti.

FIGURE 44 – Photo 1 du résultat final



FIGURE 45 – Photo 2 du résultat final



8.2 État d'avancement

L'état d'avancement actuel au jour du 15 juin 2023 est décrit ci-dessous. Le logiciel intégré dans le microcontrôleur est capable de réaliser toutes les fonctions énumérées ci-dessous :

1. Mesure de la vitesse air entre 0 et 386km/h
2. Mesure des accélérations X, Y et Z
3. Mesure des gyroscopes X, Y et Z
4. Mesure de la tension des batteries et du générateur
5. Envoi des données au travers d'une communication Bluetooth Classique

Le logiciel intégré dans l'appareil Android est capable de réaliser toutes les fonctions énumérées ci-dessous :

1. Connexions à un appareil appairé (appairage par le biais du système)
2. Réception et traitement de toutes données envoyées par l'appareil conçu
3. Affichage de toutes les valeurs brutes reçues sous forme de variables
4. Affichage de valeurs traitées par le logiciel Android

8.3 Tâches restantes

Les tâches incomplètes ou non débutées sont énumérées ci-dessous :

1. Terminer le développement de l'application Android
 - (a) Cacher les données non essentielles
 - (b) Moyenner les données
 - (c) Optimiser la facilité de lecture des données
 - (d) Ajouter la fonction de mise à zéro (offsets)
 - (e) Ajouter la fonction de logging des données
2. Effectuer un test réel en vol
3. Dessiner la nouvelle turbine avec un système de fixation plus robuste

9 Conclusion

En conclusion, ce projet de développement d'un système de détection de l'angle d'incidence et de la vitesse au décrochage d'un avion présente des défis techniques importants. L'objectif était de concevoir un système flexible, pouvant être installé sur différents types d'avions, tout en évitant l'interférence du flux d'air provenant de l'hélice pour assurer des mesures précises de vitesse. De plus, la contrainte de miniaturisation était primordiale pour minimiser la traînée et respecter une limite de poids de 500g.

La réalisation de ce projet a été une expérience inestimable qui m'a permis d'élargir mes compétences en travaillant avec des composants variés que je n'avais jamais manipulés auparavant. Cette opportunité a renforcé ma détermination et mon dévouement envers le projet, car il incarnait une occasion unique de contribuer à un domaine qui suscite en moi une profonde inspiration et fascination : l'aéronautique. Travailler sur ce projet a été une expérience incroyablement gratifiante et motivante, comblant mes aspirations en tant qu'enthousiaste passionné d'aéronautique.

Je tiens à exprimer mes sincères remerciements à M. Castoldi et à M. Moreno pour leur précieuse assistance tout au long de ce projet.

Lausanne, le 16 juin 2023

Meven Ricchieri

10 Annexes

- Références
- Cahier des charges
- Schéma électrique
- Fichiers C et H réalisés ou modifiés
- Fichier Java
- Documents CAO
- BOM
- Ficher de modifications
- Journal de travail
- Planning
- Mode d'emploi
- Affiche

Références

- [1] *Centrale à inertie.* fr. Page Version ID : 192388914. Mars 2022. URL : https://fr.wikipedia.org/w/index.php?title=Centrale_%C3%A0_inertie&oldid=192388914 (visité le 07/12/2022).
- [2] *maxon Motors as Generators.* en-US. URL : <https://support.maxongroup.com/hc/en-us/articles/360004496254-maxon-Motors-as-Generators> (visité le 04/12/2022).
- [3] *Pression statique.* fr. Page Version ID : 169954162. Avr. 2020. URL : https://fr.wikipedia.org/w/index.php?title=Pression_statique&oldid=169954162 (visité le 03/12/2022).
- [4] *Pression totale.* fr. Page Version ID : 198361997. Nov. 2022. URL : https://fr.wikipedia.org/w/index.php?title=Pression_totale&oldid=198361997 (visité le 03/12/2022).
- [5] *Tube de Pitot.* fr. Page Version ID : 198892898. Nov. 2022. URL : https://fr.wikipedia.org/w/index.php?title=Tube_de_Pitot&oldid=198892898 (visité le 04/12/2022).