

Mael LONGEFAY
Jules POPESCO
Fabien PIOT

Rapport de projet : **SUB-Explorer**

Étude et Réalisation d'un Prototype



Professeur référent : Den PALESSONGA

Année 2022-2023

Remerciements

Tout d'abord, nous tenons à témoigner toute notre reconnaissance à Monsieur Den PALESSONGA pour nous avoir guidé durant notre dernière à ESME SUDRIA sur ce projet. Ses conseils et son point de vue ont été des atouts de taille pour mener à bien nos objectifs.

Ensuite, nous tenons à remercier Monsieur Philippe DEBADIER pour son aide sur le plan technique ainsi que son point de vue général sur notre travail. Son expertise nous a permis de gagner un temps considérable sur certaines parties.

Également, la concrétisation de notre projet n'aurait pas aussi bien avancé sans l'aide de Madame Capucine Thery au FABLAB ainsi que Nicolas GUILLET Lhermite à l'atelier. Sans leurs connaissances ainsi que leurs outils mis à disposition, nous aurions mis beaucoup plus de temps à parvenir à nos fins

Enfin, nous tenons à remercier chaque personne qui a pu avoir un impact de près ou de loin sur l'avancement du projet dans sa globalité.

Introduction

Ce projet s'inscrit dans le cadre de notre formation en école d'ingénieur en dernière année l'ESME Sudria au sein la majeure Systèmes embarqués.

Afin de réaliser notre projet de fins d'études, nous avons choisi de réaliser un projet ambitieux : la conception d'un drone sous-marin (UUV - Unmanned Underwater Vehicle). Ce projet vise à développer une plateforme autonome capable de réaliser des missions d'exploration en espace restreint, dans le but de soutenir la recherche scientifique et permettre une exploration plus avancée.

Dans un premier temps, nous présenterons d'une manière générale le projet ainsi que les objectifs que nous avons défini. Ensuite, nous aborderons la veille technologique du produit ainsi que le développement logiciel concernant toute la conception que nous avons réalisé.

Table des figures

- figure 1: Schéma Bloc Fonctionnel du projet]
- Figure 2 : schéma descriptif de la caméra Raspberry
- Figure 3 et 4 : schémas descriptifs des pins de connexions des Raspberry Pi 3B+ et 4B
- Figure 5 et 6: schéma descriptif du fonctionnement du moteur courant continue
- Figure 7 : Schéma de branchement driver moteur CC
- Figure 8 : fonctionnement du moteur pas à pas
- Figure 9 : Schéma du driver moteur pas à pas
- Figure 10 : Table de vérité pour les micro-step
- Figure 11 :Schéma fonctionnel du driver A4988
- Figure 12 : Format du registre de température
- Figure 14 : Table de valeur (Relation entre la température et la sortie numérique)
- Figure 15 : Branchement du capteur avec l'intelligence numérique
- Figure 16 : Commandes en I2C de notre capteur
- Figure 17 : Table de calcul de la pression et de la température
- Figure 18 : Schéma de branchement
- Figure 19 : Attribution des pins sur le récepteur
- Figure 20 : Caractéristique du Mosfet
- Figure 21 : Schéma électronique Raspberry Pi 4B
- Figure 22 : Schéma électronique moteur pas à pas et driver A4988
- Figure 23 : Schéma électronique moteur courant continu et driver DRV8838
- Figure 24 : Schéma électronique récepteur radio 2.4GHz
- Figure 25 : Schéma électronique LED contrôlée par mosfet
- Figure 26 : Schéma électronique Buck-boost
- Figure 27 : Schéma électronique Raspberry Pi 3B+
- Figure 28 : Schéma électronique capteur température
- Figure 29 : Schéma électronique capteur de pression
- Figure 30 : Visuel du PCB seul
- Figure 31 : PCB sur Raspberry 4B
- Figure 32 : Visuel du PCB seul
- Figure 33 : Visuel du PCB sur la Raspberry Pi 3B
- Figure 34 : Visuel Modèle "Phase 1"
- Figure 35 : Visuel Modèle final
- Figure 36 : Vue éclatée
- Figure 37 : logigramme général du fonctionnement du programme du drone
- Figure 38 : Rapport cyclique 25%, 50%, 75%
- Figure 39 : Code initialisation moteur
- Figure 40 : Programme de test du driver/moteur DC
- Figure 41 : Description des channels de la télécommande Flysky
- Figure 42 : Trame I2C
- Figure 43 : Exemple de code, permettant de récupérer la valeur de la pression

Table des matières

Remerciements	2
Introduction	3
Table des matières	4
1 – Présentation générale	7
1.1 Introduction du projet	7
1.2 Cahier des charges	7
1.2.1 Les Fonctionnalités Principales du Drone	7
1.2.2 Les Fonctionnalités Secondaires du Drone	8
1.2.3 Les Fonctionnalités Supplémentaires	8
1.3 Schéma bloc général	9
2 – Veille technologique	10
2.1 Etudes des composants	10
2.1.1 Raspberry Pi Camera Module V2.1	10
2.1.2 Intelligence Numérique :	11
2.1.2.1 Raspberry Pi 4B - 2GO	11
2.1.2.2 Raspberry Pi 3B+	12
2.1.3 Moteur à courant continu	13
2.1.4 Driver Moteur à courant continu	14
2.1.5 Moteur Pas à Pas	15
2.1.6 Driver Moteur Pas à Pas	16
2.1.7 Capteur de Température	18
2.1.8 Capteur de Pression	19
2.1.9 Récepteur Télécommande 2.4GHz	21
2.1.10 Bandeau LED et MOSFET	22
2.1.11 Convertisseur Buck-Boost	23
2.1.12 Batterie	24
2.1.13 Antenne Wifi	24
2.2 Schéma électronique	25
2.2.1 Schéma électronique - Raspberry Pi 4B	25
2.2.1.1 L'intelligence numérique	25
2.2.1.2 Les moteurs pas-à-pas	26
2.2.1.3 Le moteur à courant continu	26
2.2.1.4 Le récepteur 2.4GHz	27
2.2.1.5 Le bandeau LED	27
2.2.1.6 Le Buck-Boost	28
2.2.2 Schéma électronique - Raspberry Pi 3B+	29
2.2.2.1 L'intelligence numérique	29
2.2.2.2 Capteur de température	29
2.2.2.3 Capteur de pression	30
2.2.3 PCB (Printed Circuit Board)	30

2.2.3.1 PCB pour la Raspberry Pi 4B	30
2.2.3.2 PCB pour la Raspberry Pi 3B+	32
2.3 Bilan de consommation et autonomie	33
2.3.1 Bilan de consommation	33
2.3.2 Calcul de la capacité de la batterie souhaitée	33
2.4 Structure du drone	35
2.4.1 Première ébauche	35
2.4.2 Modèle final	36
3 – Développement logiciel	38
3.1 Description fonctionnelle du code	38
3.2 Trames utilisées	39
3.2.1 PWM	39
3.2.2 Protocole I2C	41
3.3 Étude des programmes	43
3.3.1 Communication Récepteur 2.4GHz – Raspberry Pi 4B	43
3.3.1.1 Définition d'un programme d'initialisation	43
3.3.1.2 Début du programme et de la boucle sans fin	44
3.3.1.3 Récupération des duty-cycles pour les utiliser pour les moteurs	45
3.3.2 Contrôle des moteurs après la réception des commandes faites sur la radiocommande	46
3.3.2.1 Initialisation des variables de la partie moteurs et LED	46
3.3.2.2	47
3.3.3 Gestion de la caméra et des capteurs sur un Web-Serveur local	48
4 – Bilan	49
4.1 Bilan financier	49
4.2 Bilan temporel	51
4.3 Conclusion	54
Annexes	55
Glossaire des Annexes	55
Bibliographie	56
Schéma Bloc Fonctionnel	57
Schéma Électronique – Raspberry Pi 4B	58
Schéma Électronique – Raspberry Pi 3B+	59
Schéma de routage – PCB Raspberry Pi 4B (Recto)	60
Schéma de routage – PCB Raspberry Pi 4B (Verso)	61
Schéma de routage – PCB Raspberry Pi 3B+ (Recto)	62
Schéma de routage – PCB Raspberry Pi 3B+ (Verso)	63
Vue 3D de l'assemblage final du drone	64
Vue éclatée regroupant les différentes impressions 3D du drone	65
Logigramme Code : Communication Récepteur 2.4GHz – Raspberry Pi 4B	66
Logigramme Code : Raspberry Pi 4B, Contrôle des moteurs et du bandeau LED	67
Logigramme Code : Raspberry Pi 3B+, Gestion Caméra & Capteurs	68
GANTT	69

1 – Présentation générale

1.1 Introduction du projet

Notre objectif principal est de mettre au point un système de pilotage efficace pour notre drone sous-marin, en veillant à sa motorisation directionnelle et à sa radiocommunication. Nous prévoyons également d'intégrer des outils scientifiques à notre plateforme, ainsi que de travailler sur l'optimisation de son design. En somme, ce projet est l'opportunité pour nous de mettre en pratique nos connaissances en matière de systèmes embarqués, tout en nous confrontant à des défis technologiques passionnants.

Dans le cadre de ce projet, nous mettons d'abord la priorité sur le pilotage de la plateforme (motorisation directionnelle, radiocommunication). Les outils scientifiques et l'optimisation du design seront vus dans un second temps.

1.2 Cahier des charges

Le cahier des charges de notre projet vise à définir les fonctions que doit remplir le drone. Nous avons identifié trois catégories de fonctionnalités que nous allons détailler :

- Les Fonctionnalités Principales du Drone
- Les Fonctionnalités Secondaires du Drone
- Les Fonctionnalités Supplémentaires

1.2.1 Les Fonctionnalités Principales du Drone

Dans cette partie, nous définissons les fonctions clés de notre drone. Ces fonctions déterminent l'utilité et les performances de notre sous-marin. Ainsi ses besoins sont les suivants :

- UUV manœuvrable sur 3 dimensions.
→ Motorisation directionnelle (lacet, tangage, profondeur, direction).
- L'UUV est radiocommandé.
→ Conception d'un flotteur permettant de maintenir l'antenne (récepteur 2.4GHz) hors de l'eau afin de simplifier la communication entre le système de pilotage et la machine.
→ Utilisation d'une télécommande et d'un récepteur 2.4GHz.
- Le drone doit être étanche afin de protéger le système de l'eau.

- Le drone doit également répondre à une limite de taille afin qu'il puisse se déplacer dans des espaces restreints. Nous avons donc pour objectif de taille que le drone fasse maximum 600 mm de longueur et 300 mm de diamètre.
- La flottaison de l'UUV doit être neutre : nous devons faire en sorte d'obtenir un équilibre entre le poids et la poussée d'Archimède du drone.
- Nous voulons que le drone puisse effectuer des missions scientifiques d'une durée de 30 minutes à 1 heure en autonomie.

1.2.2 Les Fonctionnalités Secondaires du Drone

Ces fonctionnalités ne sont pas essentielles à la performance de notre drone. Cependant, elles améliorent l'expérience de l'utilisateur ou la récupération de données utiles à la recherche :

- Retour vidéo (mise en place d'une caméra embarquée pour pouvoir voir ce qui se trouve dans l'environnement du drone même lorsque celui-ci n'est pas visible depuis la surface).
- Système d'éclairage LED afin de pouvoir voir convenablement l'environnement où se trouve le drone via le retour caméra.
- Capteurs de température et de pression (utile pour de la recherche et pour assurer le maintien structurel du drône en fonction de la pression exercée sur lui de part la pression de l'eau - liée à la profondeur à laquelle il se trouve).
- Structure résistante aux chocs et protégeant les composants fragiles pouvant être exposés à l'extérieur du drone (par exemple les capteurs ou encore les hélices des moteurs).

1.2.3 Les Fonctionnalités Supplémentaires

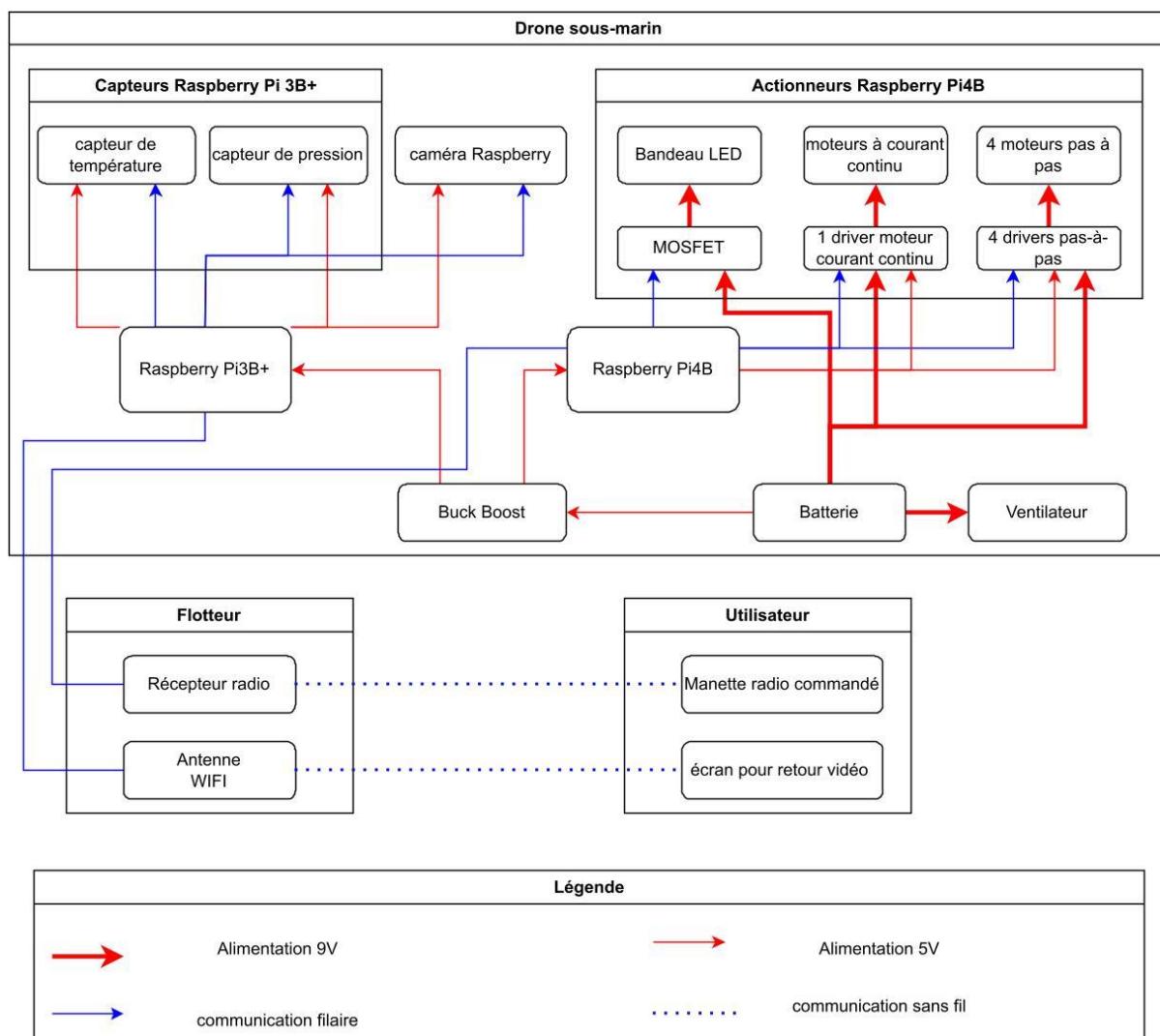
Il s'agit de fonctions qui pourraient être intégrées à notre drone en fonction du temps et des moyens dont nous disposons. Parmi ces fonctionnalités nous pouvons retrouver les éléments suivants :

- Ajouter des systèmes pour la collecte de données scientifiques : échantillonneur, sondeur, GPS
- Intégrer une capacité de stockage et de récupération rapide des données.
- Contrôle des outils scientifiques.
- Développement d'une application mobile pour le pilotage de tout le drone (contrôle de son déplacement dans l'espace et des éléments scientifiques qu'il emporte et permettant l'affichage du flux vidéo en temps réel).

1.3 Schéma bloc général

Pour comprendre le fonctionnement de notre drone sous-marin, il est utile de se référer à son schéma fonctionnel. Ce schéma représente la structure du drone et les éléments qu'il embarque, ainsi que les interactions entre ces différents éléments. Il se décompose en différents blocs, chacun correspondant à une fonction précise (flotteur / capteurs / actionneurs).

Grâce au schéma fonctionnel, nous pouvons visualiser l'organisation générale de notre drone et comprendre comment chaque élément contribue à son fonctionnement. Nous pouvons également identifier les interfaces et les points de communication entre les différents éléments de notre drone. Pour résumer, le schéma fonctionnel de notre drone sous-marin nous permet de mieux appréhender son architecture et son fonctionnement, ce qui est crucial pour sa conception et son développement.



[figure 1: Schéma Bloc Fonctionnel du projet]

2 – Veille technologique

2.1 Etudes des composants

Dans cette partie nous allons étudier les composants que nous allons utiliser, expliquer nos choix et enfin comment les utiliser.

2.1.1 Raspberry Pi Camera Module V2.1

Avant de parler de notre intelligence numérique, nous savons que nous avons besoin d'un retour vidéo de ce que fait notre drône sous-marin. Nous avons décidé de choisir une Camera Module V2.1 pour Raspberry Pi 4 : elle peut se connecter facilement à la carte Raspberry et il existe des bibliothèques de fonctions de la caméra déjà existantes ce qui apporte un gain de temps. De plus, l'utilisateur peut obtenir un retour caméra de bonne qualité avec.

Caractéristiques de la caméra choisie :

Fabricant : Raspberry Pi

Référence Fabricant : Raspberry Pi Camera Module V2.1

Certification : RoHS

Référence Caméra Embarquée : Sony IMX219 (caméra de 8 mégapixels)

Résolution Photo : 3280 x 2464 pixels

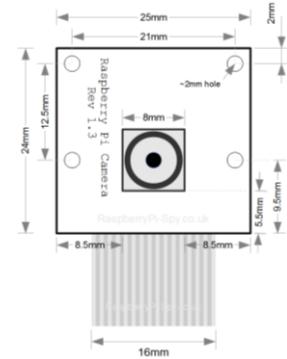
Résolution Vidéo : 1080 pixels - 30 fps / 720 pixels - 60 fps / 640x480 pixels - 90 fps

Objectif à focale fixe embarquée (autour des 50 cm de l'objectif)

Dimensions : 25mm x 24mm x 9mm

Poids : environ 3g

Consommation : 200-250mA



[Photo 2 : schéma descriptif de la caméra Raspberry]

2.1.2 Intelligence Numérique :

2.1.2.1 Raspberry Pi 4B - 2GO

Nous avons choisi la Raspberry Pi 4 pour les nombreux apports que possède cette carte. D'une part, ce choix s'est fait par la caméra choisie (Raspberry Camera Module V2.1), pleinement fonctionnelle et avantageuse avec. D'une autre part, la Raspberry possède toutes les caractéristiques essentielles au bon développement de notre projet.

Caractéristiques de la carte Raspberry Pi 4B :

Fabricant : Raspberry

Processeur : 1.5GHz quad-core 64-bit ARM Cortex-A72

RAM : 4GB de mémoire SDRAM LPDDR4

Communications possibles : Ethernet Gigabit en RJ45, Bluetooth 5.0, Deux ports USB 3.0 et deux ports USB 2.0, 2 ports micro-HDMI pour deux écrans jusqu'à 4k (3840 x 2160 pixels) à 60 FPS, micro USB,

GPU : VideoCore VI,

BUS : I2C, SPI, série

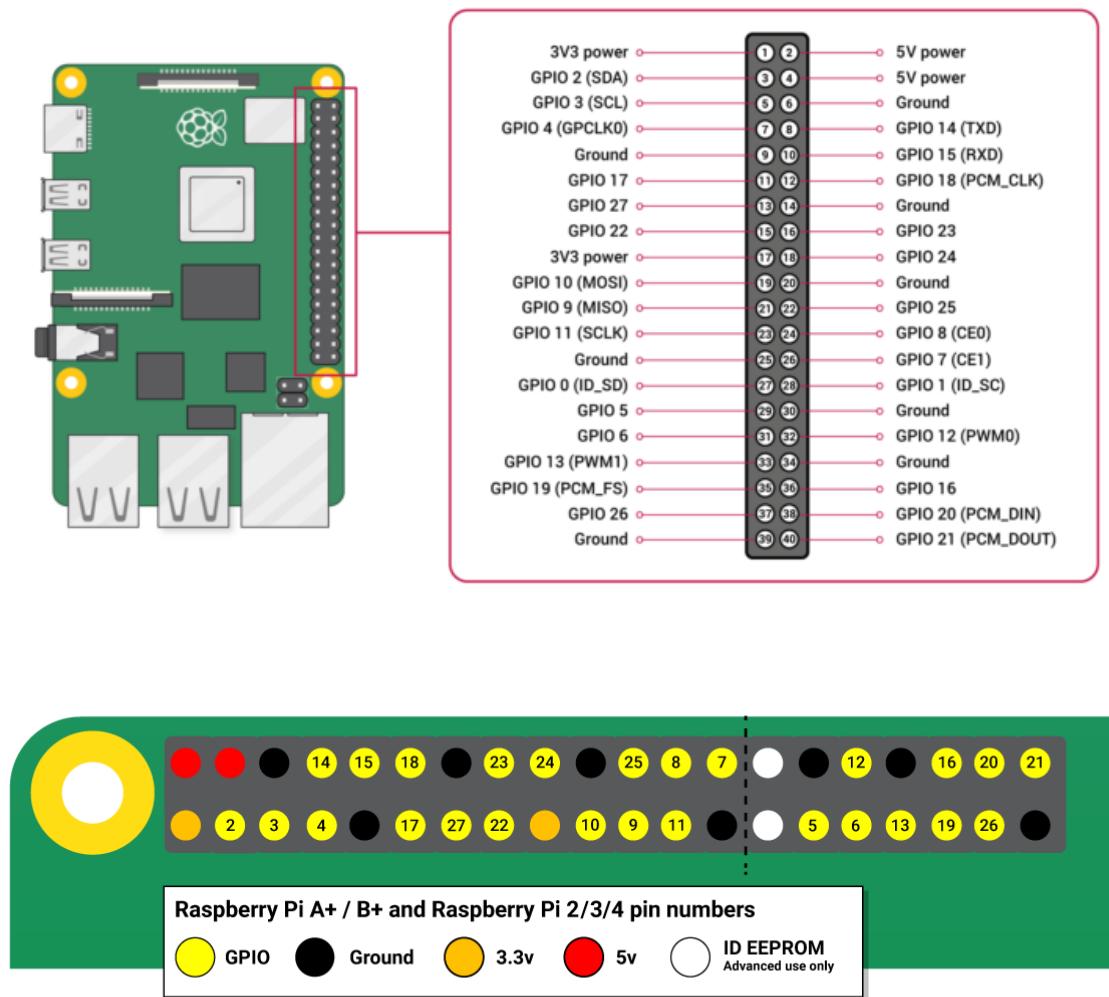
Dimensions: 88 x 58 x 17 mm

Poids: 46 g

Consommation : 5V/3A DC, mais si les composants attachés à la carte consomment moins de 500mA, la carte consomme 2.5A

28 GPIO avec des options d'interface différentes :

- 6x UART
- 6x I2C
- 5x SPI
- 1x SDIO interface
- 1x DPI (Parallel RGB Display)
- 1x PCM
- 2x PWM channels
- 3x GPCLK outputs



[Figure 3 et 4 : schémas descriptifs des pins de connexions des Raspberry Pi 3B+ et 4B]

Software : système logiciel prenant le noyau Linux en charge. Développé et maintenu activement – De nombreux pilotes en amont – Espace utilisateur stable et bien pris en charge – Disponibilité des fonctions GPU à l'aide d'API standard

2.1.2.2 Raspberry Pi 3B+

Nous utilisons une Raspberry Pi 3 en supplément de la Raspberry pi 4 afin d'avoir plus de threads à gérer pour des tâches en temps réel. Cette carte nous a été fournie suite aux problèmes techniques rencontrés lors des différents tests et de retours reçus. De cette manière, la Raspberry pi 4 est dédiée à la motorisation et à la radiocommunication entre les moteurs et la télécommande tandis que la Raspberry pi 3 gère la caméra et les capteurs. Également, la Raspberry pi 4 va moins chauffer. Si nous avions obtenu une Raspberry pi 4 de 4Go nous n'aurions pas eu à doubler le nombre de cartes.

La différence entre ces deux cartes se situe principalement entre la mémoire vive (1Go uniquement pour la Raspberry pi 3 contre 1,2 ou 4 Go sur la version 4) et des connexions plus rapides chez la Pi4 (300Mbps contre 1GBps, 4 connexions USB 2.0 contre 2 USB 2.0 et 2 USB 3.0, Bluetooth 4.1 contre Bluetooth 5.0).

2.1.3 Moteur à courant continu

Puisque nous faisons un sous-marin, nous avons choisi un moteur à courant continu pour la propulsion, ce qui permet d'obtenir un moteur suffisamment puissant et peu coûteux. Nous l'avons ajouté en remplacement d'un moteur brushless, jugé trop puissant pour notre drone.

Caractéristiques des moteurs courant continu choisis :

Fabricant : LICHIFIT

Vitesse maximale à vide : 16800 rotations par minute

Dimensions : 50mm de long et 31mm de diamètre

Longueur du bras du rotor : 26mm

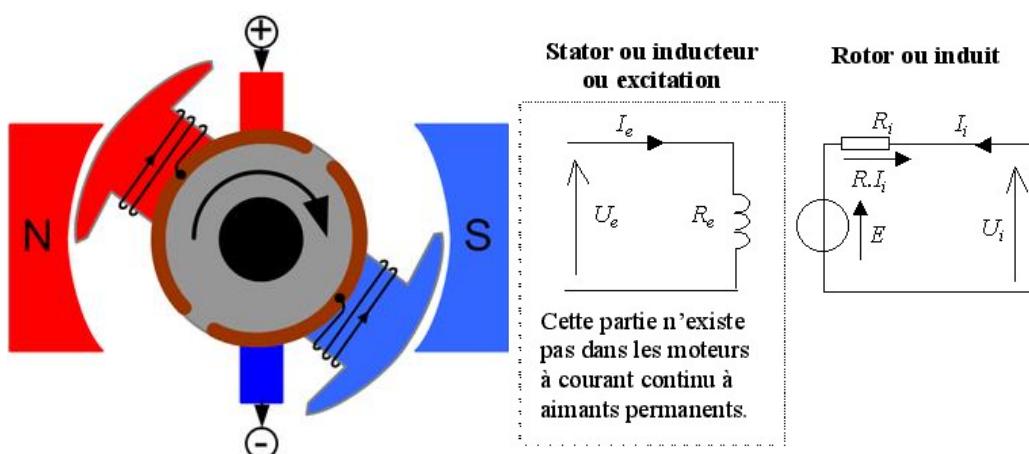
Poids : Environ 90g

Tension de fonctionnement : 3 à 12V

Tension de fonctionnement recommandée : 6 à 9V

Fonctionnement du moteur à courant continu :

Le moteur à courant continu (CC) est un type de moteur électrique qui utilise un rotor et un stator pour convertir l'énergie électrique en mouvement mécanique. Le rotor est composé d'aimants permanents ou d'un enroulement de fil conducteur relié à un collecteur, et le stator contient des bobines de fil conducteur alimentées par une source d'énergie électrique continue. Lorsque le courant électrique circule dans les bobines du stator, il crée un champ magnétique qui interagit avec le champ magnétique produit par les aimants permanents ou le rotor enroulé, ce qui crée un couple pour faire tourner le rotor. La vitesse et la direction de rotation du moteur peuvent être contrôlées en modifiant la tension et la polarité du courant électrique appliquée aux bobines du stator. Les moteurs à courant continu sont largement utilisés dans de nombreuses applications industrielles et domestiques en raison de leur faible coût, de leur simplicité de conception et de leur capacité à fournir un couple élevé à faible vitesse.



[Figure 5 et 6: schéma descriptif du fonctionnement du moteur courant continu]

Si l'information logique est reçue en PWM, les transistors seront enclenchés pendant des périodes plus courtes, et le moteur aura une vitesse de rotation réduite.

2.1.4 Driver Moteur à courant continu

Afin de correspondre au moteur à courant continu choisi, nous avons trouvé un driver moteur permettant de contrôler le moteur en question - moteur de propulsion.

Caractéristiques du driver du moteur brushless choisis :

Fabricant : RUIZHI

Référence Fabricant : ESC 320A

Dimensions : 43mm x 29mm x 27mm

Poids : 41g

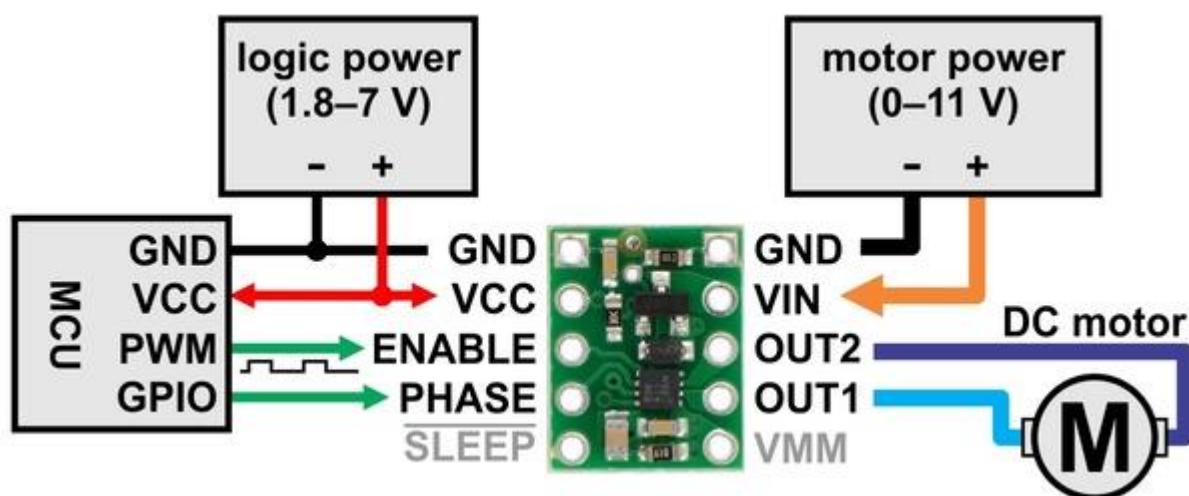
Tension de fonctionnement : 6 à 12V

Courant qu'il peut faire passer au moteur : 2A

Courant consommé par l'utilisation du driver : non indiqué mais de l'ordre du mA

Fonctionnement :

Le driver permet de contrôler le moteur à courant continu en vitesse et direction de rotation. Il est alimenté par un courant continu en provenance de la batterie et d'un courant au voltage réduit venant de la carte électronique. La consommation du driver est infime, le courant sera véhiculé dans les 2 phases du moteur tel que sur le schéma ci-dessous :



[Figure 7 : Schéma de branchement driver moteur CC]

Il fonctionne en modulant la tension et la polarité du courant électrique appliquée aux bobines du stator du moteur. Le driver moteur reçoit un signal de commande généré par un microcontrôleur puis est traité pour produire une tension de sortie qui est appliquée aux bornes du moteur. Il s'agit ici d'un drive à pont en H (fréquemment utilisé pour les moteurs à courant continu) qui utilise des transistors pour commuter la tension et la polarité du courant électrique dans les bobines du moteur. Les drivers à pont en H permettent de contrôler la vitesse et la

direction de rotation du moteur en inversant la polarité du courant électrique. Le driver moteur peut également fournir une protection contre les surcharges et les courts-circuits pour éviter d'endommager le moteur.

2.1.5 Moteur Pas à Pas

Le moteur pas à pas est plus lourd et sa vitesse de rotation maximale est plus faible comparée à celle d'un moteur brushless. Cependant il est capable de déployer plus de couple à basse vitesse et permet une plus grande précision dans les débattements.

Caractéristiques des moteurs pas à pas choisis :

Fabricant : STEPPERONLINE

Référence Fabricant : 17HS08-1044S

Nombre de pas par révolution : 200

Dimensions : 42mm x 42mm x 22mm

Longueur du bras du rotor : 20mm

Couple moteur : 16 Nm⁻²

Inertie du rotor : 0.15 g*cm²

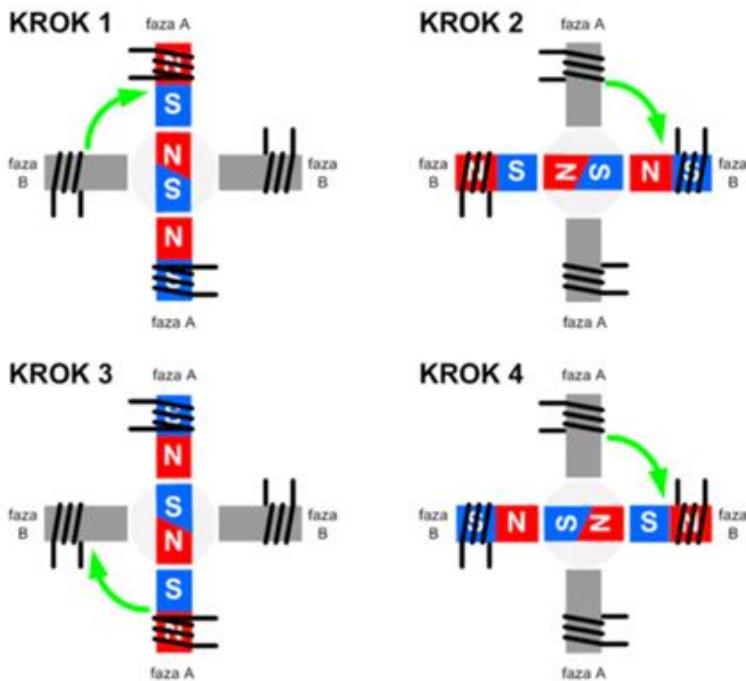
Poids : 150g

Tension de fonctionnement : 3.5V

Fonctionnement:

Le moteur pas à pas est un moteur à courant continu sans balais dans lequel la rotation est divisée en un certain nombre de pas qui résultent de la structure du moteur. Généralement, une rotation d'arbre complète de 360° est divisée en 200 pas, ce qui signifie qu'un seul pas d'arbre est effectué tous les 1,8°. On utilisera un moteur bipolaire dans le cadre de notre projet (deux phases distinctes). Dans un fonctionnement en pas complet, on alimente une seule bobine à la fois.

Fonctionnement en pas entier :



[Figure 8 : fonctionnement du moteur pas à pas]

Pour augmenter la précision du moteur, il est possible d'utiliser un fonctionnement en demi-pas. Il convient alors d'utiliser un courant alternatif alimentant les deux bobines. Les phases sont commandées grâce à un pont en H contenu dans le driver.

2.1.6 Driver Moteur Pas à Pas

Caractéristiques des drivers des moteurs pas à pas choisis :

Fabricant : Pololu

Référence Fabricant : A4988 1182

Dimensions : 20mm x 15mm

Tension de fonctionnement : 8 à 35V

Courant qu'il peut faire passer au moteur : 1A (possible de monter à 2A mais nécessite un refroidissement additionnel)

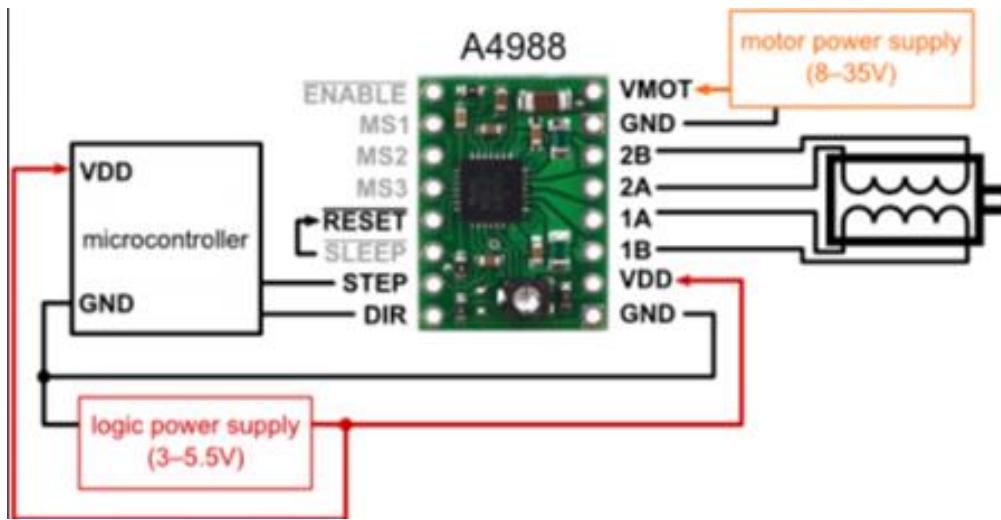
Courant consommé par l'utilisation du driver : 8mA

Fonctionnement :

Le driver alimente le moteur pas à pas et contrôle le sens de rotation et la course de celui-ci. Le driver A4988 peut être entièrement contrôlé grâce à des signaux analogiques.

Le pin DIR permet d'inverser le sens de circulation du courant dans les bobines, et donc de changer le sens de rotation du moteur. Le pin STEP permet de déclencher un cycle complet des deux ponts en H, entraînant un pas complet du moteur.

Schéma de câblage, fonctionnement en pas complet uniquement :



[Figure 9 : Schéma du driver moteur pas à pas]

Afin d'améliorer la précision, il est possible de faire fonctionner le driver en demi, quart, huitième et seizième de pas. La réception de signaux analogique dans les pins MS simultanément au pin STEP permet de « bloquer » certains transistors pendant le cycle et ainsi réduire la course du moteur durant la période d'un pas complet.

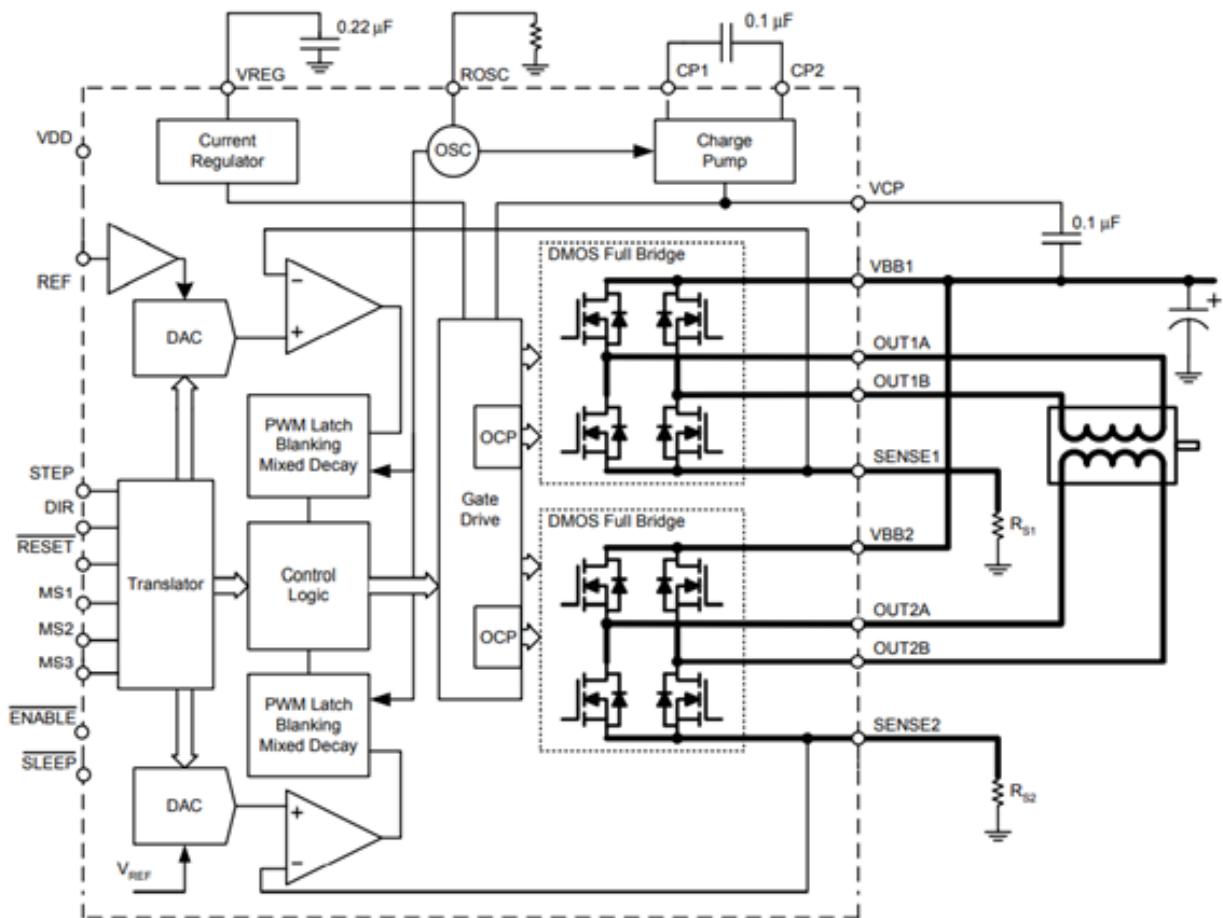
[Figure 10 : Table de vérité pour les micro-step]

MS1	MS2	MS3	Microstep Resolution	Excitation Mode
L	L	L	Full Step	2 Phase
H	L	L	Half Step	1-2 Phase
L	H	L	Quarter Step	W1-2 Phase
H	H	L	Eighth Step	2W1-2 Phase
H	H	H	Sixteenth Step	4W1-2 Phase

[Figure 11 : Schéma fonctionnel du driver A4988]

On remarque **en gras** le circuit d'alimentation du moteur, relié à la partie de contrôle par les deux ponts en H. Le driver comporte également des pin SLEEP, ENABLE et RESET, en état

« haut » par défaut, permettant d'intégrer mieux le composant dans un système plus



complexe.

2.1.7 Capteur de Température

Caractéristiques du capteur de température choisi :

Fabricant : DFROBOT

Référence Fabricant : DFR0198

Certification : RoHS

Dimensions de la sonde : 35mm de long * 6mm de diamètre

Tension de fonctionnement : 3 à 5.5V

Plage de détection du capteur : de -55°C à 125°C

Précision du capteur : $\pm 0.5^{\circ}\text{C}$ (entre -10°C et 85°C)

Sortie du capteur : Numérique

[Figure 12 : Format du registre de température]

	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
LS BYTE	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
	BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
MS BYTE	S	S	S	S	S	2^6	2^5	2^4

S = SIGN

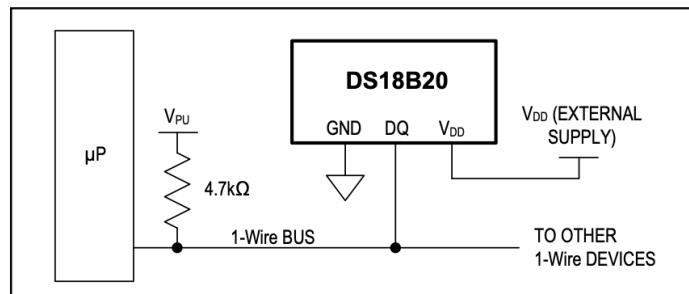
Ainsi nous pouvons constater que les 5 bits de poids forts définissent le signe de notre température.

[Figure 14 : Table de valeur (Relation entre la température et la sortie numérique)]

TEMPERATURE ($^{\circ}\text{C}$)	DIGITAL OUTPUT (BINARY)	DIGITAL OUTPUT (HEX)
+125	0000 0111 1101 0000	07D0h
+85*	0000 0101 0101 0000	0550h
+25.0625	0000 0001 1001 0001	0191h
+10.125	0000 0000 1010 0010	00A2h
+0.5	0000 0000 0000 1000	0008h
0	0000 0000 0000 0000	0000h
-0.5	1111 1111 1111 1000	FFF8h
-10.125	1111 1111 0101 1110	FF5Eh
-25.0625	1111 1110 0110 1111	FE6Fh
-55	1111 1100 1001 0000	FC90h

*The power-on reset value of the temperature register is +85°C.

[Figure 15 : Branchement du capteur avec l'intelligence numérique]



2.1.8 Capteur de Pression

Caractéristiques du capteur de pression choisi :

Fabricant : TE Connectivity

Référence Fabricant : MS5837-30BA

Spécificité : Capteur Étanche | I2C

Certification : RoHS

Dimensions du capteur : 3.3mm x 3.3mm x 2.75mm

Tension de fonctionnement : 1.5 à 3.6V (recommandé : 3V)

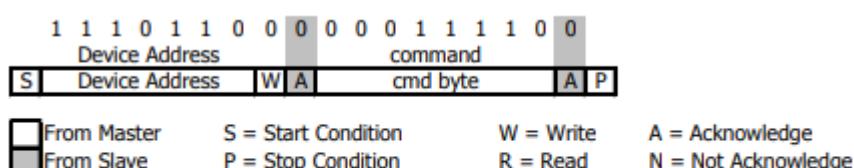
Courant consommé en fonctionnement : 0.6µA

Courant consommé au repos : 0.1µA (à 25°C)

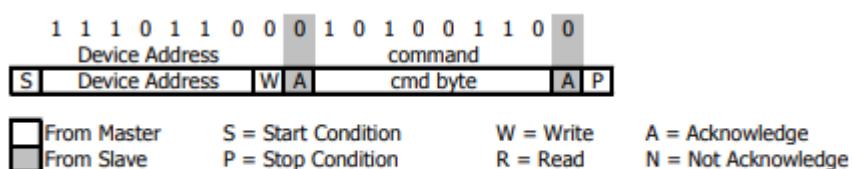
Plage de détection du capteur : 0 à 30 Bar (utilisable entre -20°C et 85°C)

Précision du capteur : ± 0.2 mBar (soit ± 2mm d'altitude)

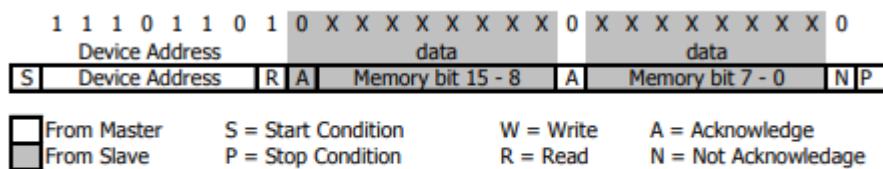
[Figure 16 : Commandes en I2C de notre capteur]



I²C Reset Command

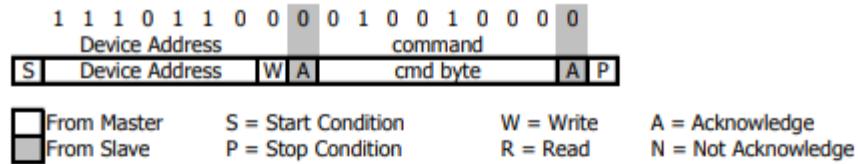


I²C Command to read memory address= 011

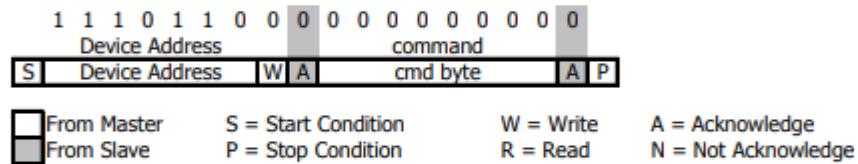


I²C answer from MS5837-30BA

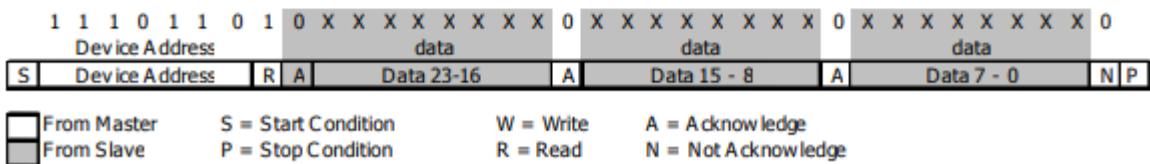
Groupe 3S4 - SUB-EXPLORER : drone sous marin d'exploration



I²C command to initiate a pressure conversion (OSR=4096, typ=D1)



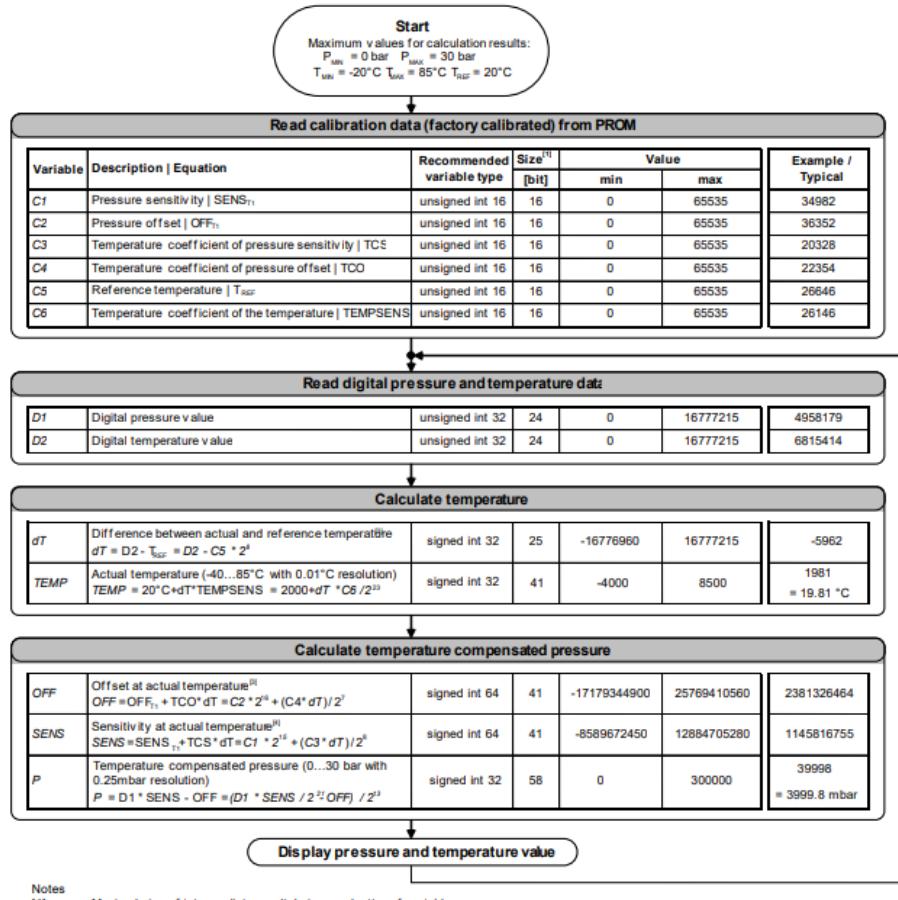
I²C ADC read sequence



I²C answer from MS5837-30BA

Groupe 3S4 - SUB-EXPLORER : drone sous marin d'exploration

[Figure 17 : Table de calcul de la pression et de la température]



Flow chart for pressure and temperature reading and software compensation.

2.1.9 Récepteur Télécommande 2.4GHz

Caractéristiques du récepteur 2.4GHz choisi :

Fabricant : DTXMX Flysky

Référence Fabricant : FS-iA6B

Spécificité : Fonctionnement 2.4GHz

Plage Radio Fréquence : 2,408 - 2,475 GHz

Dimensions boîtier : 43mm x 26mm x 15mm

Longueur des 2 antennes : 26mm

Poids : 16g

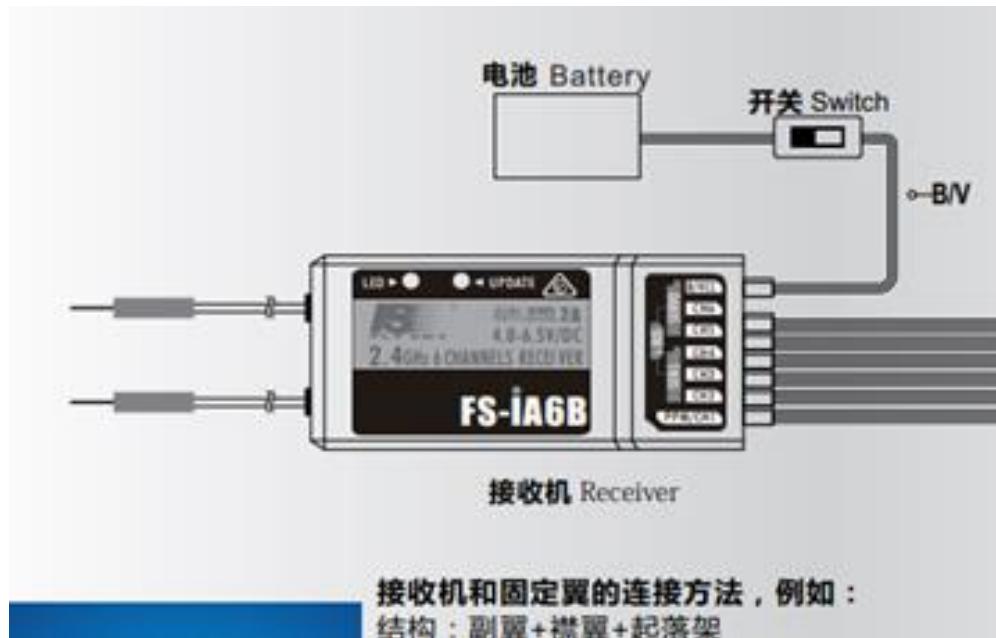
Tension de fonctionnement : 4 à 6.5V

Fonctionnement :

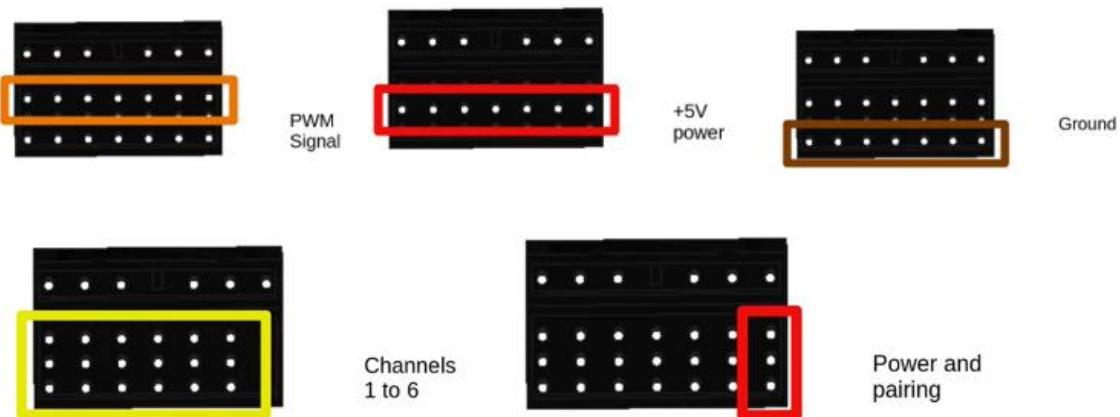
Le récepteur radio 2.4GHz permet de convertir les données envoyées par la télécommande en signaux PWR. Ces signaux sont par la suite captés par les GPIO (utilisé en PWM « logiciel ») grâce à la fonction d'**input capture**. Ce récepteur permet de contrôler 6 channels distincts. Cinq valeurs seront reçues, correspondant chacune à un débattement, et seront traduites et définissent la puissance d'utilisation des moteurs ainsi que leur sens de rotation. Le sixième canal permet de contrôler l'allumage des LED.

Les deux antennes permettant de recevoir les signaux radio doivent être placées perpendiculairement l'une par rapport à l'autre pour une réception optimale.

[Figure 18 : Schéma de branchement]



[Figure 19 : Attribution des pins sur le récepteur]



2.1.10 Bandeau LED et MOSFET

Caractéristiques du bandeau LED choisi :

Fabricant : BTF-LIGHTNING

Référence Fabricant : WS2812B Pixel LED Lights Strip

Spécificité : IP67 (Étanche)

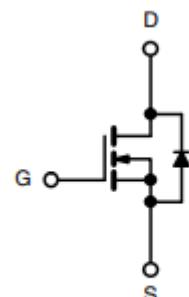
Certification : RoHS

Nombres de LED/m : 100

Taille du bandeau total : 1m

Tension de fonctionnement : 5V

Durée de vie : 50000 heures



N-Channel MOSFET

Caractéristiques techniques du MOSFET :

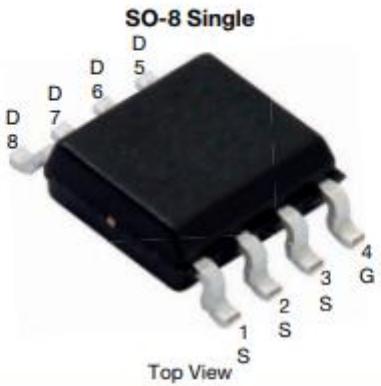
Fabricant : Vishay Siliconix

Référence : Si4116DY

Certification : RoHs, 100 % Rg and UIS tested

Tension Vds : 25 V

[Figure 20 : Caractéristique du Mosfet]



The diagram shows a black SO-8 package labeled "SO-8 Single". The pins are numbered 1 through 8 around the perimeter. Pin 1 is labeled "S" (Source), pin 2 is labeled "S" (Source), pin 3 is labeled "S" (Source), pin 4 is labeled "G" (Gate), pin 5 is labeled "D" (Drain), pin 6 is labeled "D" (Drain), pin 7 is labeled "D" (Drain), and pin 8 is labeled "D" (Drain). Below the package, the text "Top View" is written.

PRODUCT SUMMARY	
V_{DS} (V)	25
$R_{DS(on)}$ max. (Ω) at $V_{GS} = 10$ V	0.0086
$R_{DS(on)}$ max. (Ω) at $V_{GS} = 4.5$ V	0.0095
$R_{DS(on)}$ max. (Ω) at $V_{GS} = 2.5$ V	0.0115
Q_g typ. (nC)	17.5
I_D (A) ^a	18
Configuration	Single

Fonctionnement du MOSFET :

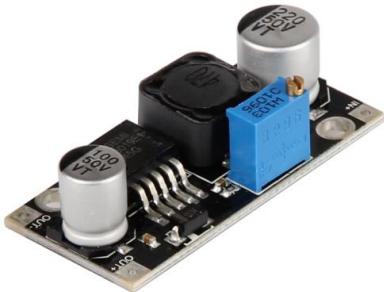
Un MOSFET à canal N est un type de transistor de la gamme des semi-conducteurs. Il peut être utilisé pour amplifier ou commuter des signaux électroniques. Lorsqu'une tension est appliquée entre la broche de la source S et la broche du drain D, une charge électrique s'accumule dans le canal N, créant ainsi un chemin pour le courant à travers le MOSFET. La quantité de courant qui peut être transmise à travers le canal dépend de la tension appliquée à la broche de la grille G. En contrôlant la tension appliquée à la broche de la grille, il est possible de réguler la quantité de courant qui traverse le canal, ce qui permet d'amplifier ou de commuter des signaux électroniques.

2.1.11 Convertisseur Buck-Boost

Nous utilisons un convertisseur Buck-Boost afin de réguler la tension à travers tout le sous marin.

Technical Specifications

Model	XL6019 (Step-Up adjustable voltage regulator)
Article No.	SBC-XL6019
Input Voltage	5 - 40V DC
Output Voltage	5 - 40V DC (adjustable via potentiometer)
Input Current	Max. 5A
Special Features	High efficiency (up to 94%) Integrated MOSFET-Amplifier



2.1.12 Batterie

Nous utilisons une pile 9V de 5500 mAh pour l'ensemble des composants du sous-marin. Cette partie est liée à la partie 2.3 au bilan de consommation. Elle a également été choisie pour sa capacité à être rechargée en USB-C.
Son autonomie est de 30 minutes. Tous les calculs dans la partie 3.2 correspondante.

2.1.13 Antenne Wifi

Le répéteur est utilisé pour bien transmettre le flux vidéo de la caméra.

Caractéristiques du répéteur wifi :

marque : Ralink

connectivité : sans fils

Interface matérielle : USB

système d'exploitation : Linux

dimensions : 10X10X10 cm

poids : 10g

2.2 Schéma électrique

Dans le but de définir le schéma électrique le plus adéquat à notre projet, nous avons travaillé sur Fusion360 afin de pouvoir créer le schéma électrique ainsi que le PCB associé.

Par conséquent, les parties ci-dessous décrivent le schéma électrique partie par partie avec des explications sur l'utilisation de chacun des composants et branchements.

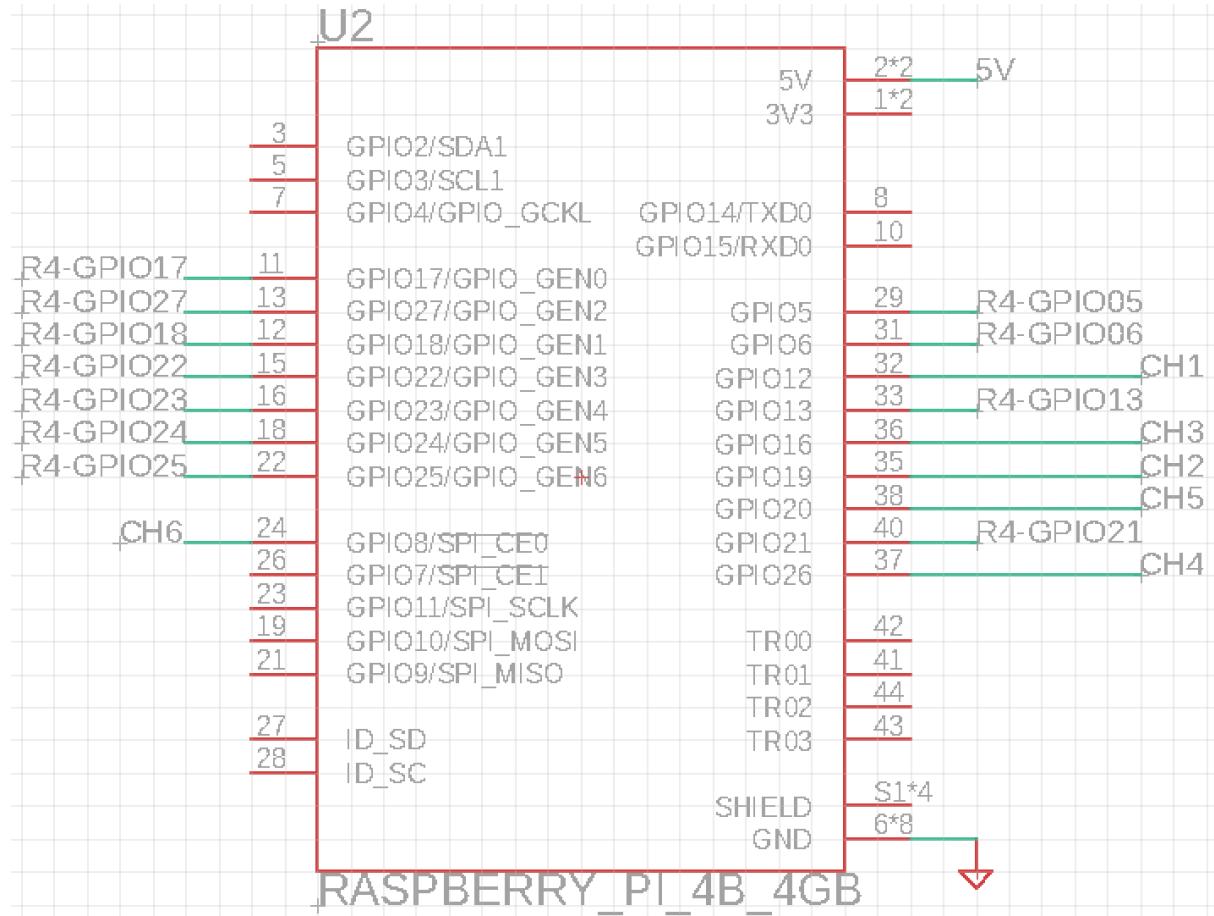
Pour plus de clarté concernant l'utilité de tous les composants, vous pouvez retrouver en annexe le schéma électrique complet.

2.2.1 Schéma électrique - Raspberry Pi 4B

2.2.1.1 L'intelligence numérique

Pour commencer, voici le schéma électrique regroupant les connexions sur l'intelligence numérique, ici la Raspberry Pi 4B.

[Figure 21 : Schéma électrique Raspberry Pi 4B]

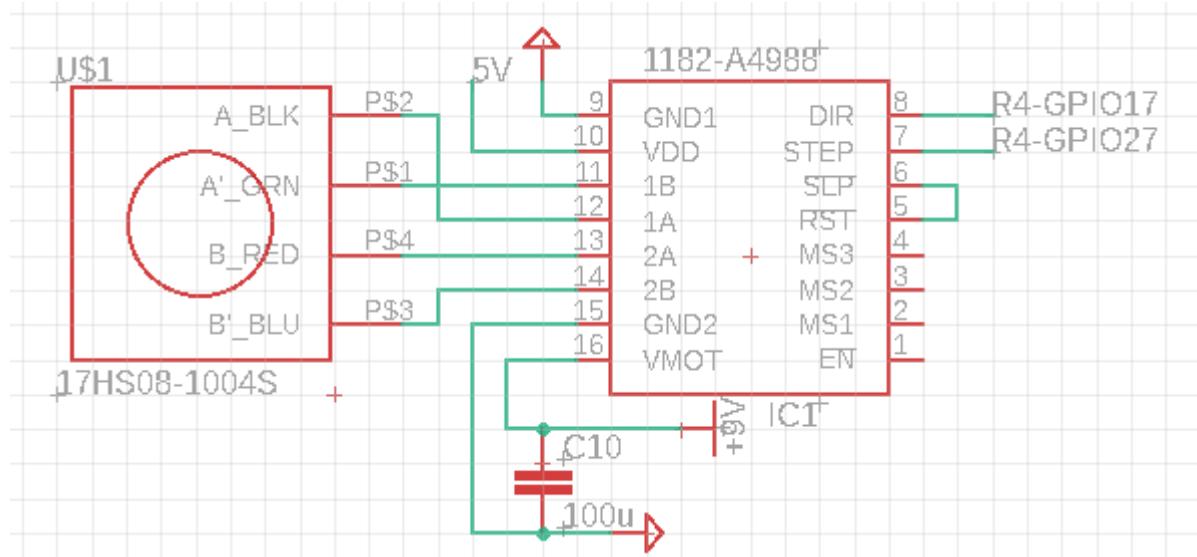


On peut voir que la Raspberry Pi 4B sera beaucoup sollicitée. Cette Raspberry Pi sera utilisée pour la gestion de tous les déplacements (translatifs et rotatifs) ainsi que le contrôle du bandeau LED, le tout contrôlé à distance à l'aide d'une télécommande 2.4GHz.

2.2.1.2 Les moteurs pas-à-pas

Ci-dessous une description du schéma électronique d'un des quatre moteurs pas-à-pas 17HS08-1004S relié à la Raspberry Pi par un driver moteur 1182-A4988. Chaque moteur pas à pas est câblé de cette manière.

[Figure 22 : Schéma électronique moteur pas à pas et driver A4988]

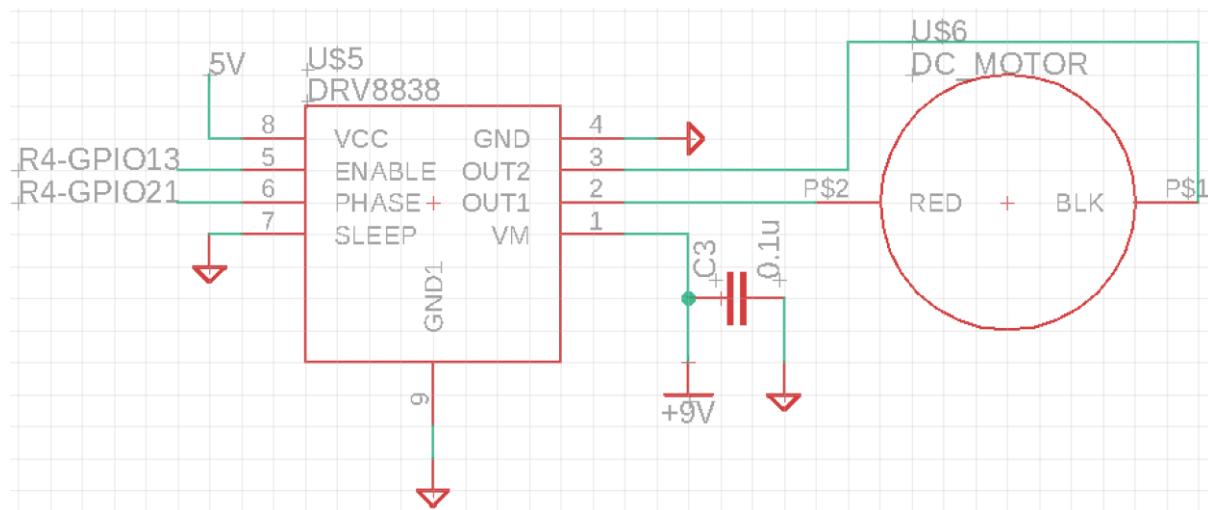


Pour l'utilisation correcte de ces moteurs, nous utilisons un driver relié par 2 pin à notre intelligence électronique. Pour ce driver-ci, le pin DIR est relié au GPIO17 de la Raspberry Pi 4B. Ce pin sert à définir le sens de rotation du moteur pas-à-pas qu'il contrôle. Le pin STEP quant à lui sert à activer ou non la rotation du moteur pas-à-pas. L'important également est le fait que la tension d'utilisation de notre moteur est de supérieure à celle utilisée pour la Raspberry. Ici notre moteur prend comme tension d'usage 9V et on relie le canal d'alimentation à la masse avec un condensateur pour abaisser son impédance dans la bande passante du signal.

2.2.1.3 Le moteur à courant continu

Ci-dessous la représentation du moteur courant continu et de son driver sur le schéma électrique connectée à la Raspberry pi 4 par le driver moteur DRV8838 :

[Figure 23 : Schéma électrique moteur courant continu et driver DRV8838]

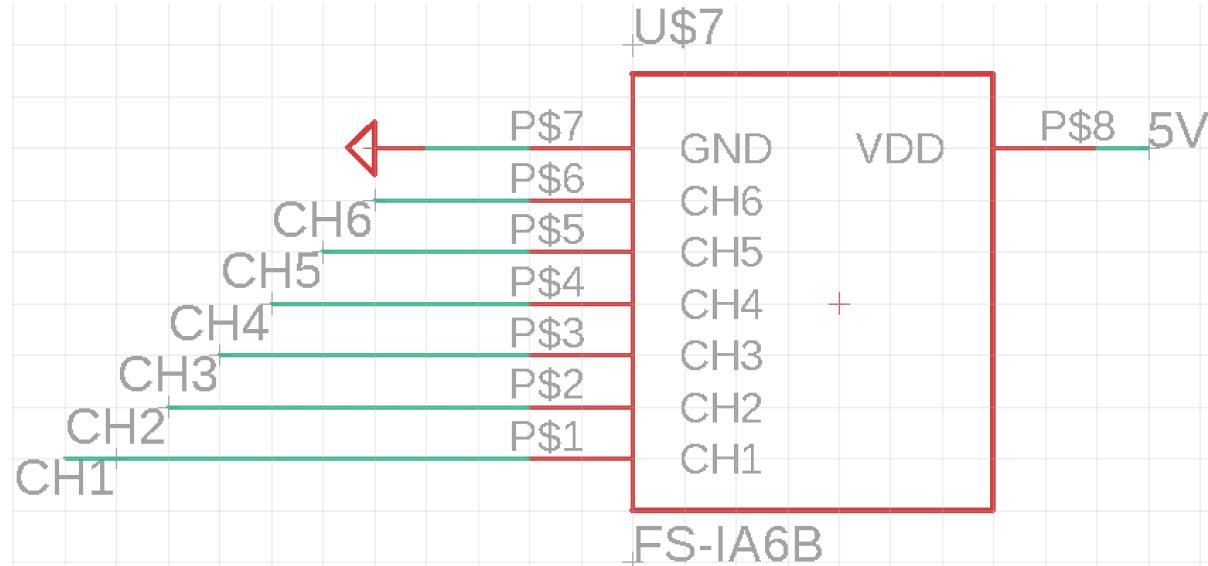


Pour l'utilisation correcte du moteur DC, nous utilisons un driver relié (comme pour les moteurs pas-à-pas) par 2 pin à notre intelligence électrique. Pour ce driver-ci, le pin ENABLE est relié au GPIO13 de la Raspberry Pi 4B. Ce pin reçoit de la part de la Raspberry un signal en PWM permettant de contrôler la vitesse du moteur en fonction du duty-cycle du signal PWM. Le pin PHASE sert quant à lui à définir le sens de rotation du moteur pas-à-pas qu'il contrôle. Les pins OUT1 et OUT2 permettent d'activer le moteur dans un sens ou dans l'autre en fonction de quel pin parmi les deux est l'alimentation et l'autre la masse (défini par le pin PHASE). Tout comme pour les moteurs pas-à-pas, la tension d'utilisation de notre moteur DC est de minimum 7.4V, il est donc important également que la tension d'utilisation de notre moteur soit supérieure à 7.4V. Ainsi nous, par simplicité, nous utilisons la tension de notre batterie, donc 9V. On relie également le canal d'alimentation à la masse avec un condensateur pour abaisser son impédance dans la bande passante du signal.

2.2.1.4 Le récepteur 2.4GHz

Dans cette partie, la représentation schéma électronique connectant le récepteur de la télécommande 2.4 GHz (FS-IA6B) branché en IN à la Raspberry Pi 4B :

[Figure 24 : Schéma électronique récepteur radio 2.4GHz]

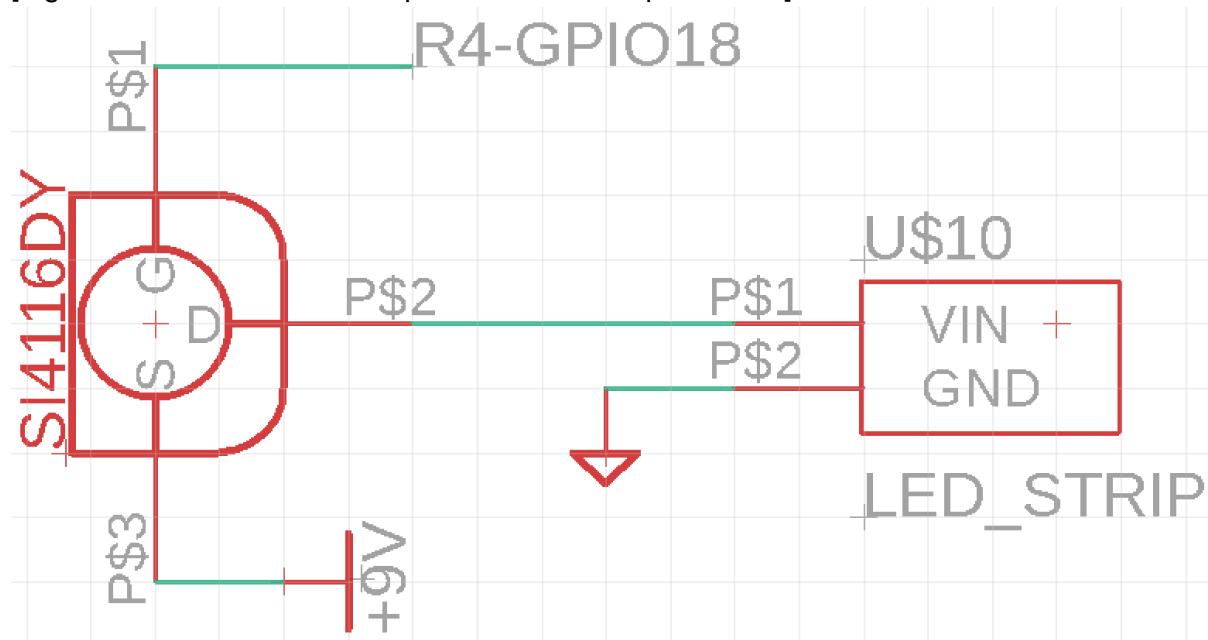


Ces 6 channels permettent de contrôler les différents moteurs ainsi que le bandeau LED simultanément, dépendant des données réceptionnées par le flotteur sur lequel il y aura le récepteur.

2.2.1.5 Le bandeau LED

Voici la représentation des branchements pour le contrôle du bandeau LED nommé WS2812B. L'utilisation d'un mosfet (SI4116DY dans notre cas) est nécessaire pour obtenir une luminosité vraiment adéquate aux besoins de notre sous-marin. L'alimentation du mosfet à l'aide du GPIO18 permet de laisser passer le courant de 9V pour alimenter le bandeau LED. Puisque un GPIO de Raspberry peut fournir jusqu'à 5V, l'utilisation d'un mosfet est nécessaire.

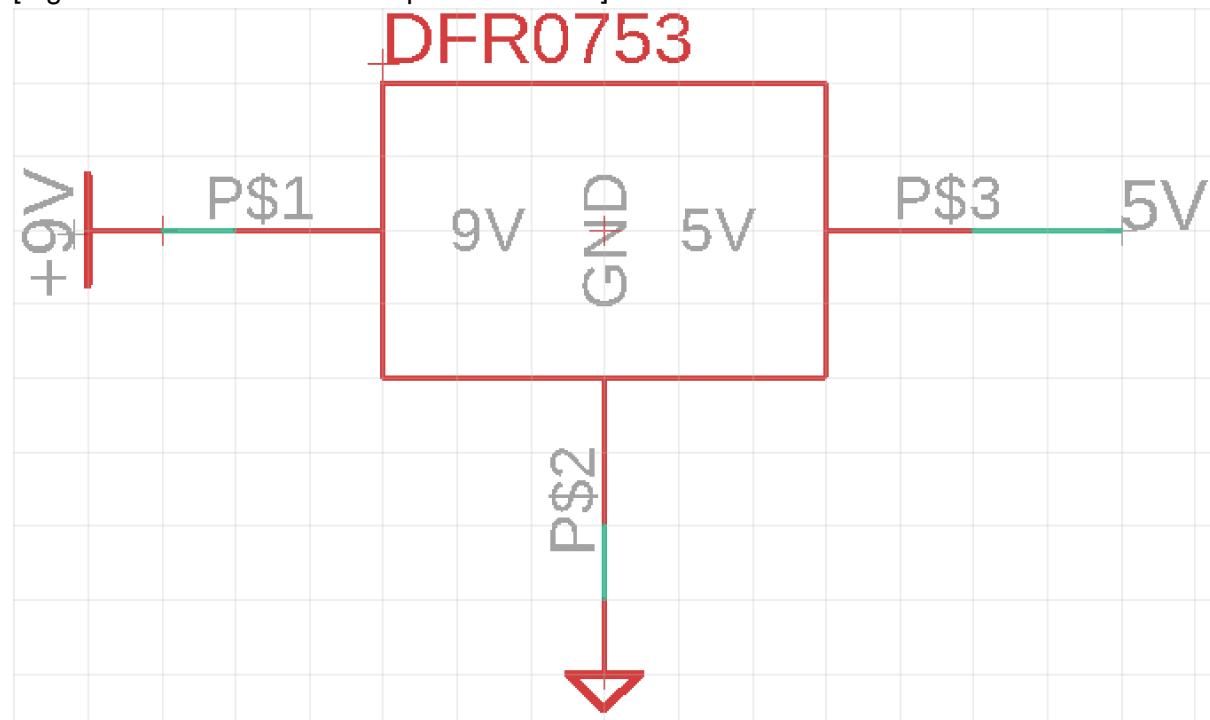
[Figure 25 : Schéma électronique LED contrôlée par mosfet]



2.2.1.6 Le Buck-Boost

Ci-dessous la représentation du régulateur de tension DFR0753 permettant de convertir la tension de l'alimentation de 9V en 5V afin de pouvoir utiliser le 5V pour les composants n'ayant pas besoin de beaucoup de tension pour fonctionner comme les deux Raspberry et capteurs par exemple. Laissant tout de même la possibilité d'utiliser le 9V pour les moteurs et le bandeau LED.

[Figure 26 : Schéma électronique Buck-boost]

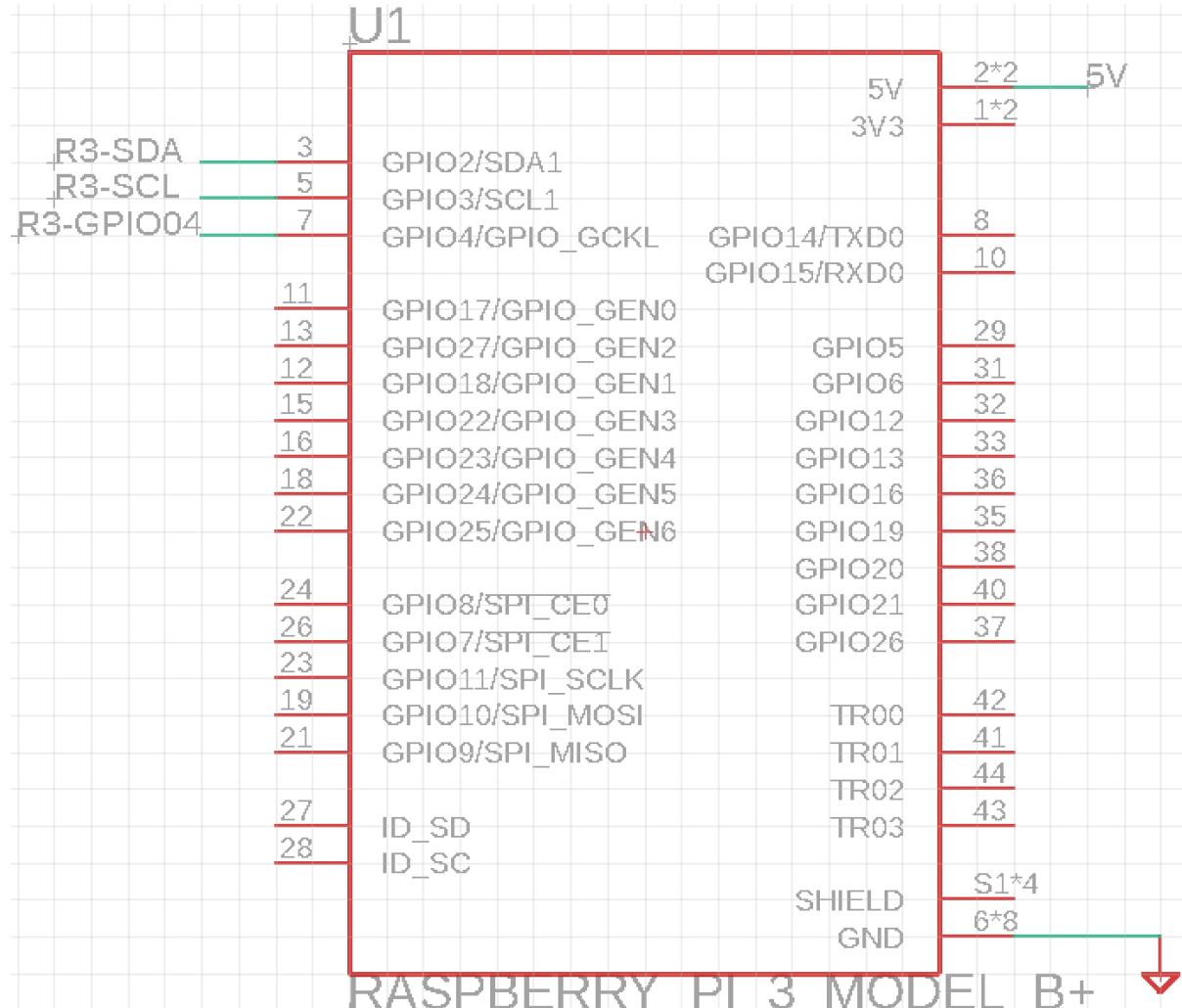


2.2.2 Schéma électrique - Raspberry Pi 3B+

2.2.2.1 L'intelligence numérique

Pour commencer, voici le schéma électrique regroupant les connexions sur l'intelligence numérique, ici la Raspberry Pi 4B.

[Figure 27 : Schéma électrique Raspberry Pi 3B+]

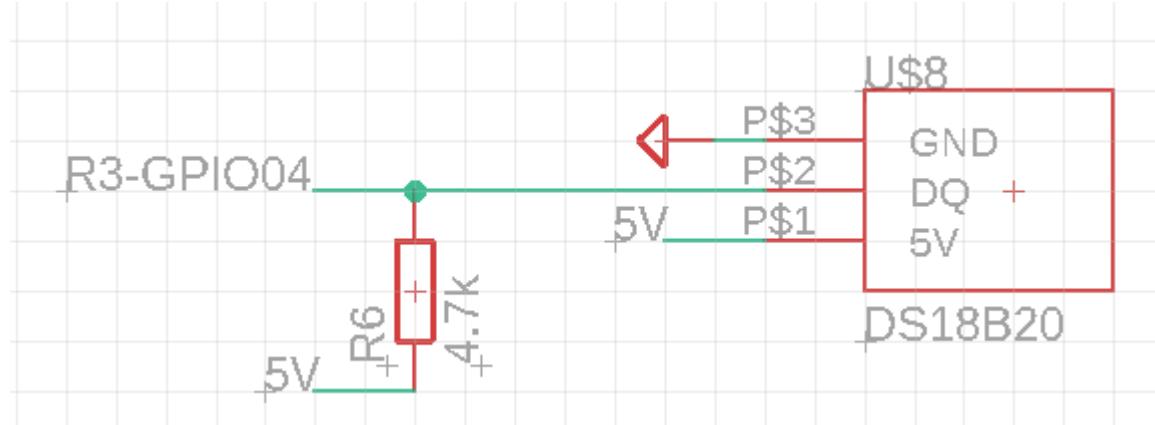


On peut voir que la Raspberry Pi 3B+ semble assez peu sollicitée. Cependant, cette Raspberry Pi sera utilisée pour la récupération de données des capteurs de température et de pression, la création d'un stream de la caméra retransmis sur un serveur local afin que l'utilisateur puisse récolter toutes les informations concernant l'environnement du sous-marin.

2.2.2.2 Capteur de température

Ci-dessous la représentation du capteur de température DS18B20 branché en IN/OUT à la Raspberry Pi 3B+ :

[Figure 28 : Schéma électronique capteur température]

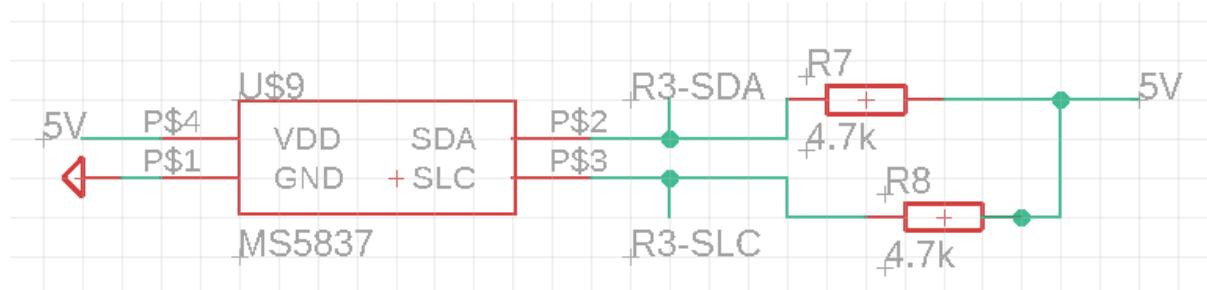


Le pin DQ est relié au GPIO4 de la Raspberry et est le pin qui envoie les données à l'intelligence numérique. Puis à l'aide d'un code, nous pouvons traiter les données pour avoir un affichage en degré Celsius ou Fahrenheit.

2.2.2.3 Capteur de pression

Ci-dessous la représentation du capteur de pression MS5837 branché en I2C à la Raspberry pi 3 :

[Figure 29 : Schéma électronique capteur de pression]



2.2.3 PCB (Printed Circuit Board)

Pour les deux parties à venir, les visuels des schémas de routage sont disponibles en annexes.

2.2.3.1 PCB pour la Raspberry Pi 4B

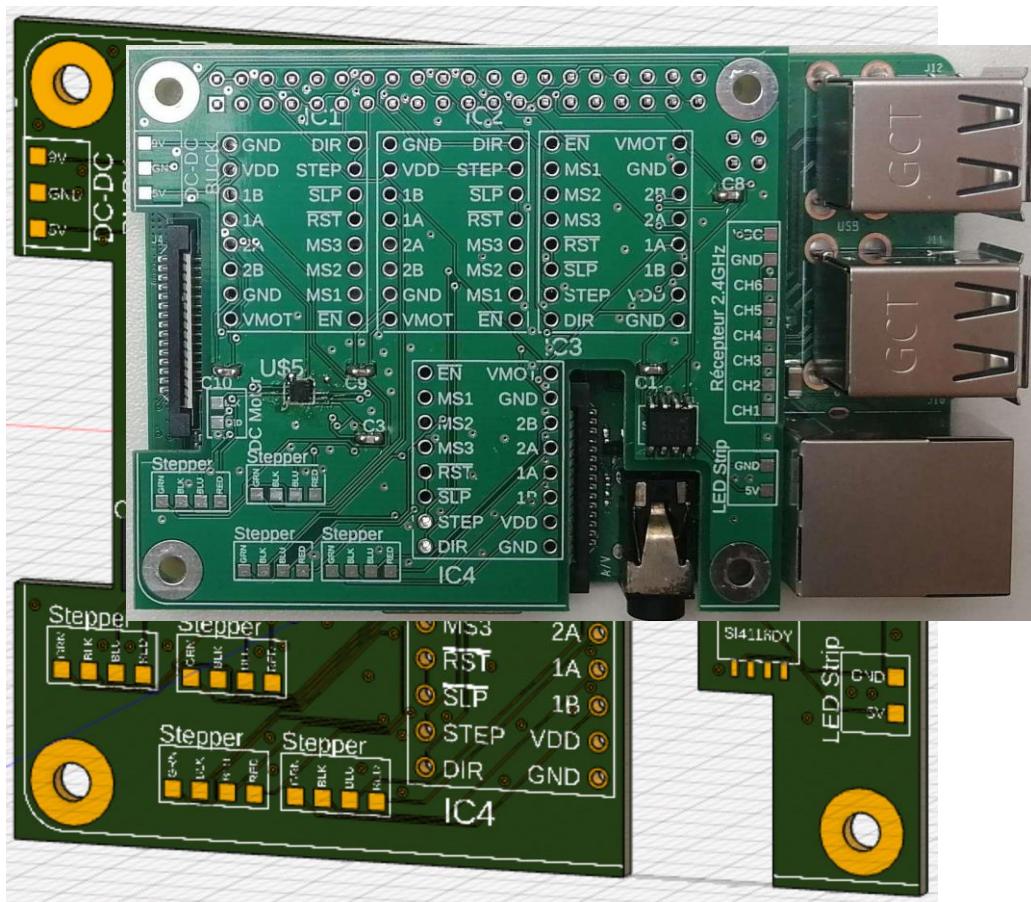
Une fois le schéma électronique complètement réalisé, nous pouvons passer à la conception des PCB. Pour se faire, nous choisissons de dimensionner le PCB afin qu'il puisse se positionner sur les pins des deux Raspberry Pi. Pour se faire, nous avons regardé quels sont les éléments gênants qui sont sur les deux Raspberry Pi afin de faire en sorte que les deux PCB puissent se positionner parfaitement. Ainsi, pour la Raspberry Pi 4B, nous en avons constaté 4 et sont les suivants :

- Les ports USB2, USB3 et Ethernet
 - Le port Jack 3.65mm
 - La connectique caméra
 - La connectique “Display”

Tous ces éléments sont gênants de par leur taille. Le problème ici étant la hauteur des composants. Ainsi, nous avons défini le dimensionnement du PCB pour la Raspberry Pi 4B en conséquence.

Une fois le dimensionnement fait, le choix des composants qui doivent figurer sur le PCB, il reste plus qu'à travailler dessus pour faire le routage de toutes les pistes. On se retrouve avec le résultat suivant :

[Figure 30 : Visuel du PCB seul]



[Figure 31 : PCB sur Raspberry 4B]

2.2.3.2 PCB pour la Raspberry Pi 3B+

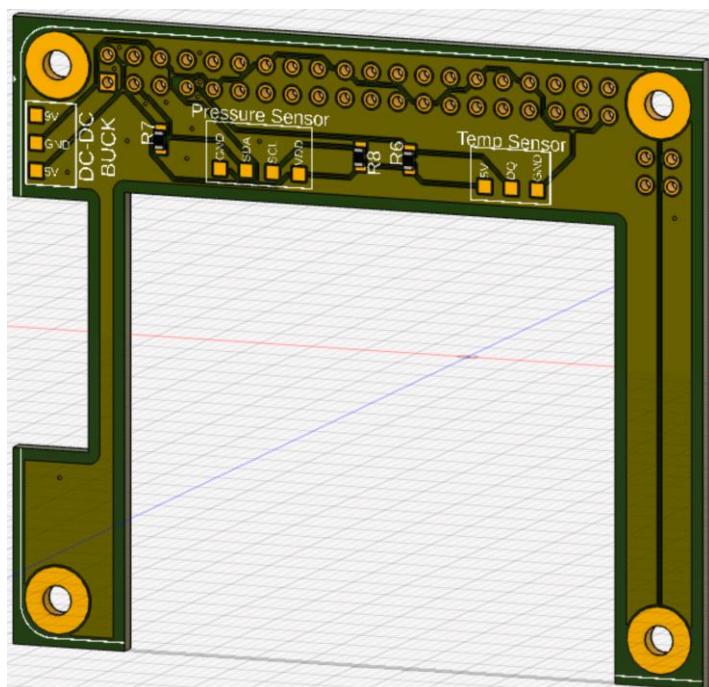
Pour la Raspberry Pi 3B+, les éléments qui vont se retrouver sur le PCB sont moins nombreux. Ainsi les éléments importants sont principalement :

- Les ports USB2, USB3 et Ethernet
- La connectique "Display"

La connectique caméra est également très importante ici car nous allons l'utiliser. Ici aussi, ces éléments sont gênants de par leur hauteur.

Tout comme le premier PCB, une fois le dimensionnement fait, le choix des composants qui doivent figurer sur le PCB, il reste plus qu'à travailler dessus pour faire le routage de toutes les pistes. On se retrouve avec le résultat suivant :

[Figure 32 : Visuel du PCB seul]



[Figure 33 : Visuel du PCB sur la Raspberry Pi 3B]



2.3 Bilan de consommation et autonomie

2.3.1 Bilan de consommation

Afin d'effectuer un bilan de consommation, nous avons listé :

- Raspberry Pi 4B : 3A *1
- Raspberry Pi 3B+ : 3A *1
- Caméra : 250mA *1
- Moteur Courant Continu : 1A *1
- Moteur Pas-à-Pas : 1A *4
- Capteur de Température : 1.5mA *1
- Capteur de Pression : 1.25mA *1
- Driver Moteur Brushless : Donnée non renseignée et négligeable (de l'ordre du mA)
- Driver Moteur Pas-à-Pas : Donnée non renseignée et négligeable (de l'ordre du mA)
- Bandeau LED : Donnée non renseignée et négligeable comparé à la consommation des deux intelligences numériques et des différents moteurs
- Emetteur/Récepteur 2.4GHz : Donnée non renseignée

Dans notre calcul du dimensionnement de la batterie, nous ne prenons pas en compte la consommation des drivers de moteur car ils ne font que fournir du courant et que leur consommation est très faible et négligeable comparé à la consommation d'autres composants. Le bilan de consommation de tous les composants est par addition de 11.25A, auquel nous allons ajouter 1A en plus soit 12.25A de part les composants dont la consommation n'est pas indiquée sur la datasheet tel que le bandeau LED et le récepteur 2.4GHz.

2.3.2 Calcul de la capacité de la batterie souhaitée

Comme indiqué dans le Cahier des Charges, notre objectif est de faire un drone dont l'autonomie est de 30 minutes ou d'une heure. Ainsi, nous pouvons mettre en place les calculs suivants afin de définir la capacité de la batterie que nous voudrions embarquer à l'intérieur du drone.

Afin de permettre une autonomie du drone de 30 minutes, nous avons besoin d'une batterie de 4625mAh. Le calcul associé est le suivant :

$$\begin{aligned} \text{Battery}_{\text{Capacity}} &= \text{Consommation}_{\text{Totale}} * \text{Autonomie} \\ \Rightarrow \quad \text{Battery}_{\text{Capacity}} &= 12250mA * 0.5h \\ \Leftrightarrow \quad \text{Battery}_{\text{Capacity}} &= 6125mAh \end{aligned}$$

Afin de permettre une autonomie du drone de 1 heure, nous avons besoin d'une batterie de 9250mAh. Le calcul associé est le suivant :

$$\begin{aligned} \text{Battery}_{\text{Capacity}} &= \text{Consommation}_{\text{Totale}} * \text{Autonomie} \\ \Rightarrow \quad \text{Battery}_{\text{Capacity}} &= 12250mA * 1h \\ \Leftrightarrow \quad \text{Battery}_{\text{Capacity}} &= 12250mAh \end{aligned}$$

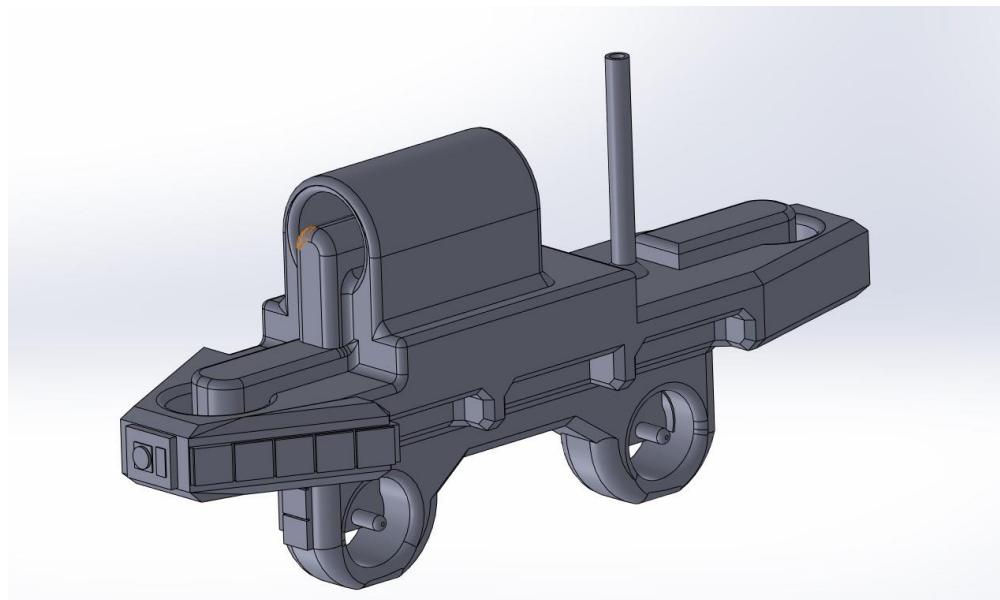
2.4 Structure du drone

2.4.1 Première ébauche

Un important défi dans la réalisation de ce projet concernait la conception des éléments structurels de notre drone. Puisque l'idée d'utiliser une coque préfabriquée pour y intégrer notre électronique a été rapidement mise à l'écart et que les modèles trouvés ne pouvant satisfaire nos exigences (notamment concernant l'aspect exploration en milieu clos), nous avons décidé de concevoir entièrement la structure.

Dans un premier temps, nous avons établi les principales conditions s'appliquant à la conception de la coque : elle doit pouvoir accueillir toute l'électronique, l'ensemble doit avoir une flottabilité nulle (poussée d'Archimède compensant intégralement le poids) et être entièrement étanche. Idéalement, nous souhaitions que les éléments les plus fragiles de notre sous-marin soient relativement protégés, pour permettre l'exploration d'espaces confinés.

[Figure 34 : Visuel Modèle “Phase 1”]



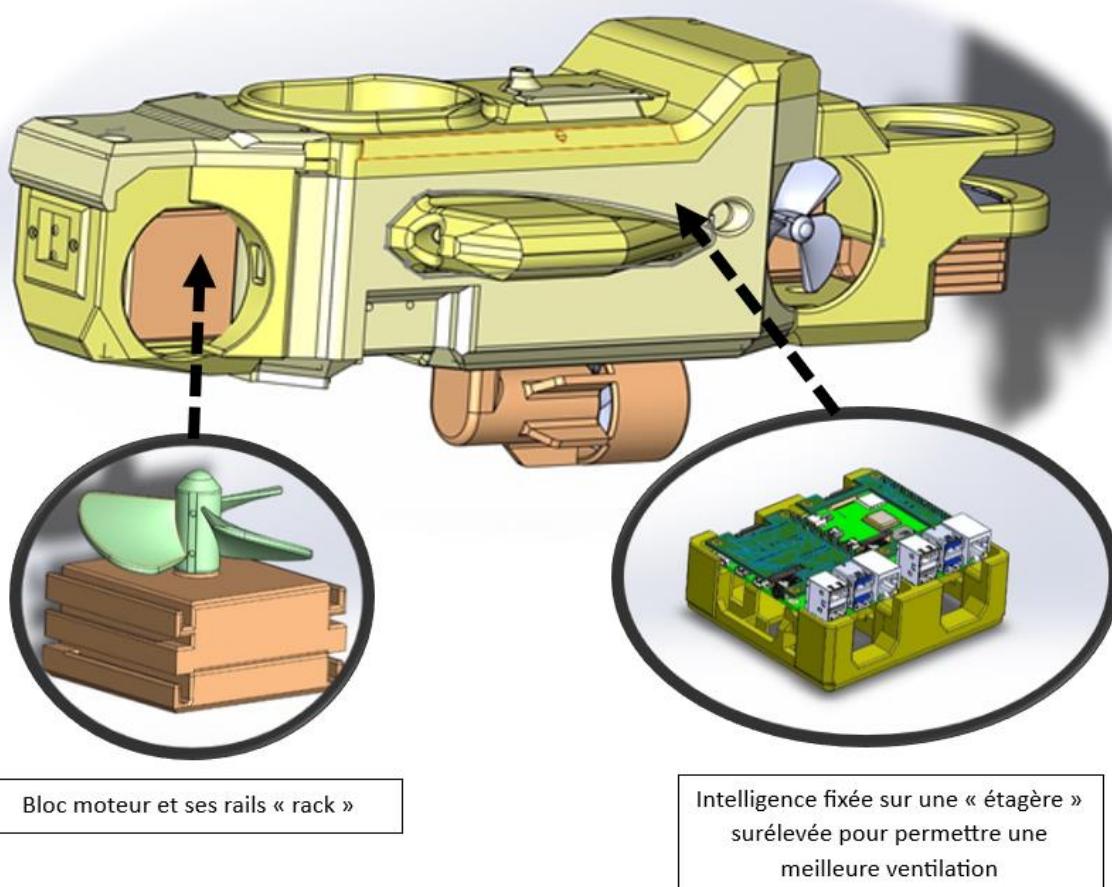
Nous avons imaginé la structure en plusieurs pièces : des boîtiers intégrant des gouttières nous permettent d'étanchéifier chaque moteur. Ces blocs moteurs peuvent ensuite s'imbriquer par des racks à différents emplacements de la coque. Par souci de répartition du poids, les moteurs pas à pas sont placés aux extrémités du drone. Le moteur CC de propulsion est, quant à lui, fixé au centre de la structure. Sur la phase 1, il est sur le dessus de la structure, devant le puit réservé au fil d'Ariane . Les moteurs de direction latéraux sont répartis sous la structure pour réduire l'encombrement.

La caméra est intégrée au-devant de l'ensemble structurel, afin de donner au pilote un champ de vision maximal. Sur les flancs, des emplacements permettent d'accueillir des flotteurs ou des poids morts afin d'équilibrer le drone et de lui donner la flottabilité nulle recherchée.

Une étude plus approfondie de la structure ainsi que la modification de certaines exigences du cahier des charges ont amené à d'importantes évolutions dans le design de la coque.

2.4.2 Modèle final

[Figure 35 : Visuel Modèle final]



Sur ce design final, nous avons opté pour un aspect plus compact, par rapport à la silhouette plus élancée de la phase 1. Nous avons voulu rendre la structure plus solide, et qu'elle puisse offrir une meilleure protection aux moteurs et à la caméra.

Les moteurs de direction sont intégrés directement dans la coque, ce qui rend les hélices moins vulnérables aux chocs. Le bloc du moteur de propulsion a été placé au-

dessous de la structure. Avec ce positionnement, on garantit un meilleur équilibre du drone. Le poids du moteur CC sert de « quille » au navire. Cela permet également d'utiliser le moteur lorsqu'il est en surface. L'ajout d'ailes permet de stabiliser le drone pendant ses déplacements.

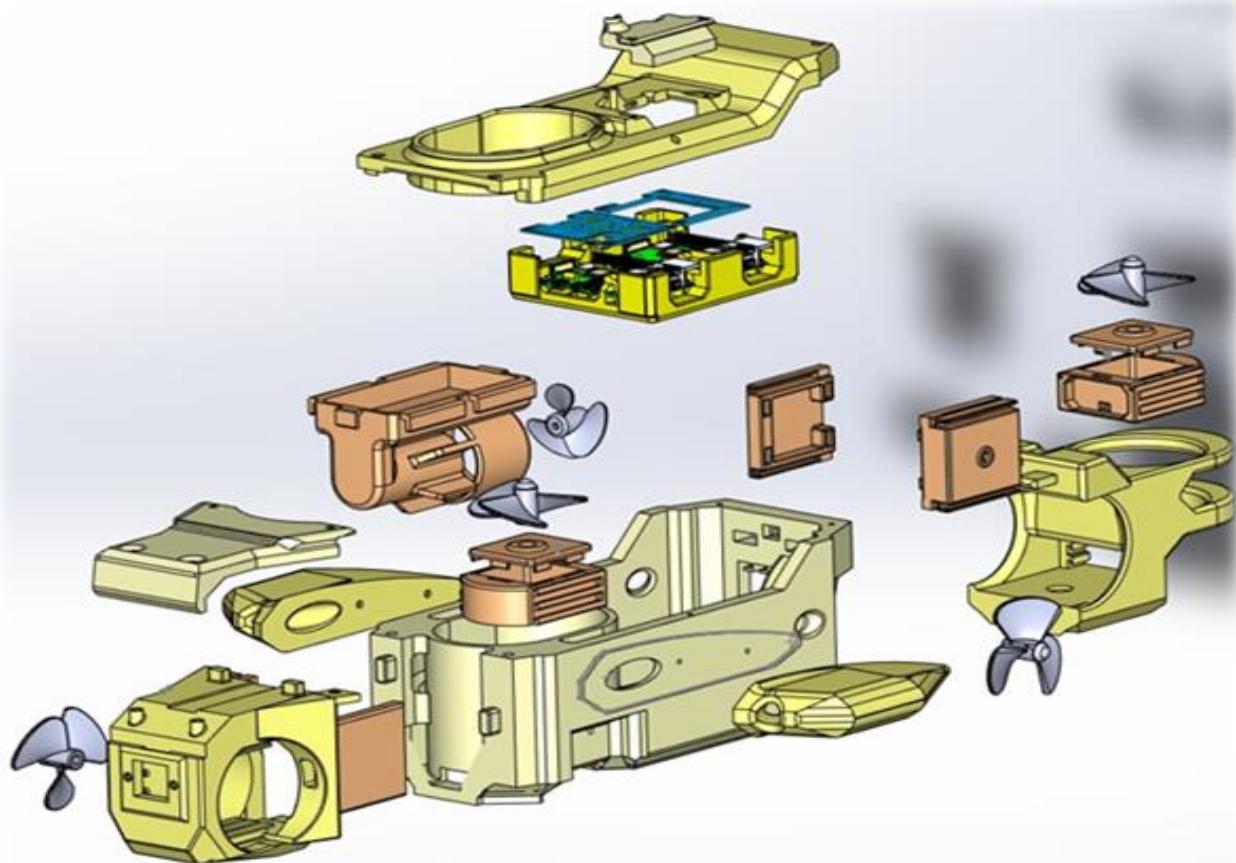
L'ajout d'une seconde carte Raspberry nous a contraint à augmenter la largeur du drone et la forte génération de chaleur à l'intérieur de l'espace étanche nous a obligé à intégrer un système de refroidissement. Nous avons opté pour un couple de tuyau souple, permettant à un flux d'eau, supposé froid, de passer au-dessus des composants. L'échange thermique s'effectue en surface de ces tuyaux. L'entrée de ces tuyaux se situe directement sur le bord d'attaque des ailes, permettant une meilleure entrée de l'eau pendant les déplacements vers l'avant ou l'arrière (l'utilisation du moteur CC générant une importante quantité de chaleur). Un ventilateur permet d'évacuer la chaleur vers cette zone de refroidissement.

La caméra est toujours placée en avant de la structure, mais est désormais protégée par une pièce en plexiglass. On retrouve sur le dessus le puits pour le fil d'Ariane, à côté de celui du capteur de température. On retrouve également des emplacements de chaque côté de la coque pour des flotteurs ou poids morts, comme sur le premier design.

Au total, on retrouve une trentaine de pièces, essentiellement fixées entre elles par des inserts de vis, ou grâce aux systèmes de « rack ». L'étanchéité est majoritairement permise grâce à des joints toriques placés le long des arêtes de contact. L'ensemble est relativement simple à démonter. Vers la fin de notre projet, nous avons tenu à ce que cette structure puisse être simplement modifiée, si des élèves venaient à reprendre notre projet plus tard. On pourrait par exemple imaginer l'intégration d'un module de capteurs (pressions, température, sonar...), voire d'un échantillonneur. Les blocs moteurs peuvent être facilement détachés et placés sur une nouvelle pièce structurelle.

Groupe 3S4 - SUB-EXPLORER : drone sous marin d'exploration

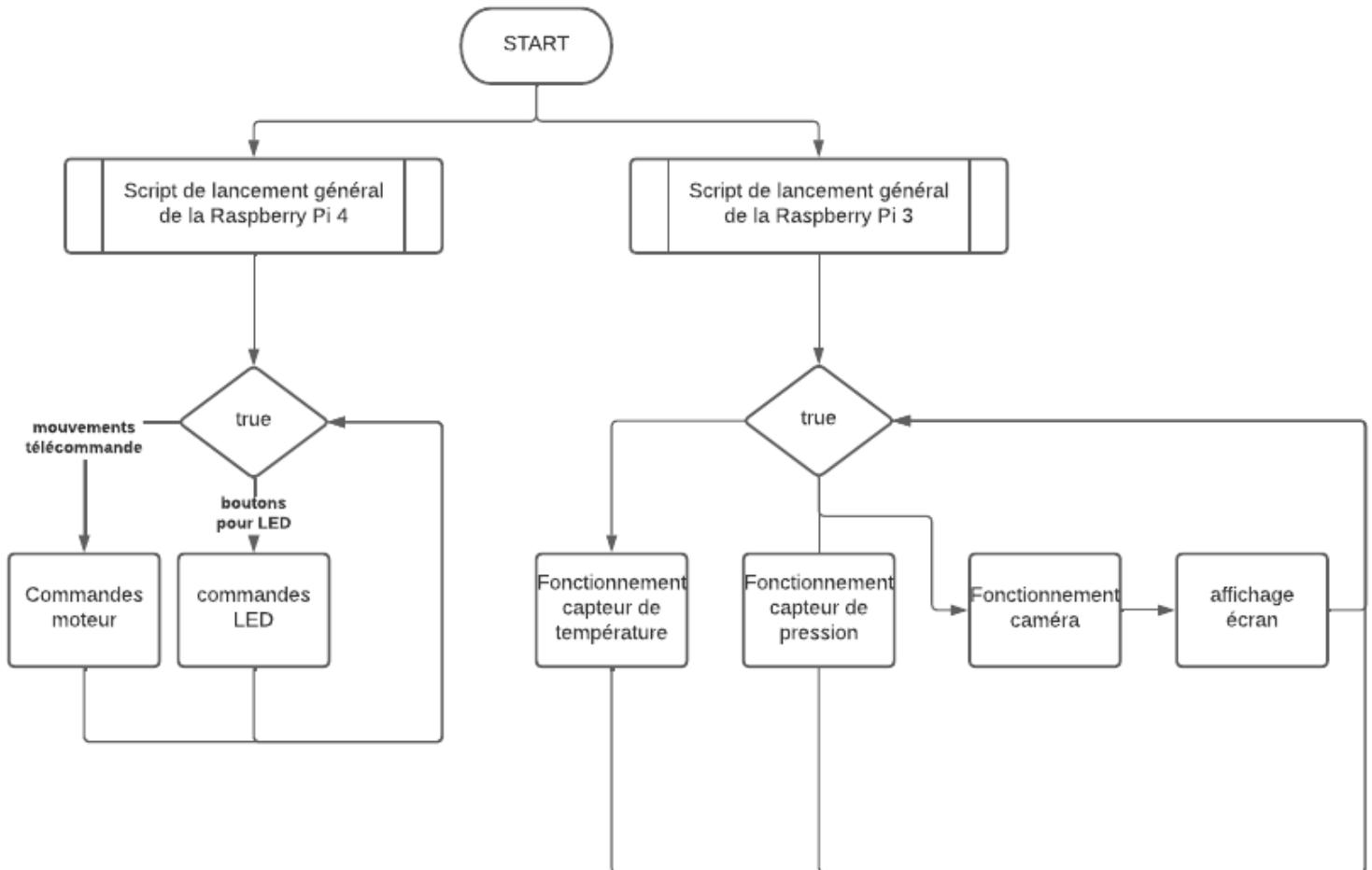
[Figure 36 : Vue éclatée]



3 – Développement logiciel

3.1 Description fonctionnelle du code

[Figure 37 : logigramme général du fonctionnement du programme du drone]



3.2 Trames utilisées

3.2.1 PWM

Description du protocole :

Le PWM (Pulse Width Modulation), ou Modulation de Largeur d'Impulsion (MLI) en français, est un protocole permettant de réduire la puissance moyenne délivrée d'une sortie digitale (0 ou 1) en modulant les impulsions du signal.

La modulation permet d'obtenir une sortie pseudo analogique, pouvant prendre une valeur entre 0 et 1024. Cette valeur correspond au rapport cyclique défini pour la sortie. Ce rapport correspond au temps pendant lequel le signal est à sa valeur haute (1) par rapport à celui pendant lequel il est en valeur basse (0). Un rapport cyclique de 25% signifie que la valeur sera maintenue, au total, pendant 25% du temps du cycle. Dans le cas d'une carte Arduino, la fréquence de modulation est fixée (490 Hz). La Raspberry Pi 4B nous permet cependant de choisir une fréquence entre 10Hz et 8KHz (la valeur par défaut étant 582Hz).

[Figure 38 : Rapport cyclique 25%, 50%, 75%]



Applications dans le projet :

Dans le cadre de notre projet, nous utiliserons le PWM pour deux utilisations. D'abord, le protocole permet le contrôle précis du moteur à courant continu (moteur de propulsion). Comme décrit précédemment, le driver du moteur brushless définit le courant délivré au moteur en fonction de la puissance moyenne qui lui sera envoyé.

[Figure 39 : Code initialisation moteur]

```

import os
import time
os.system ("sudo pigpiod")
time.sleep(1)
import pigpio

ESC=18 #Definition du pin utilisé

pi = pigpio.pi();
pi.set_servo_pulsewidth(ESC, 0) #Démarrage à 0

max_value = 1024
min_value = 300 #On définit une valeur maximum et minimum, dépendant du driver.

```

[Figure 40 : Programme de test du driver/moteur DC]

```

def control():
    time.sleep(1)
    speed = 580 # Valeur de démarrage
    print "Controls - e pour ralentir & a pour accélérer"
    while True:
        pi.set_servo_pulsewidth(ESC, speed)
        inp = raw_input()

        elif inp == "e":
            speed += 10
            print "speed = %d" % speed
        elif inp == "a":
            speed -= 10
            print "speed = %d" % speed
        else:
            print "Mauvaise commande"

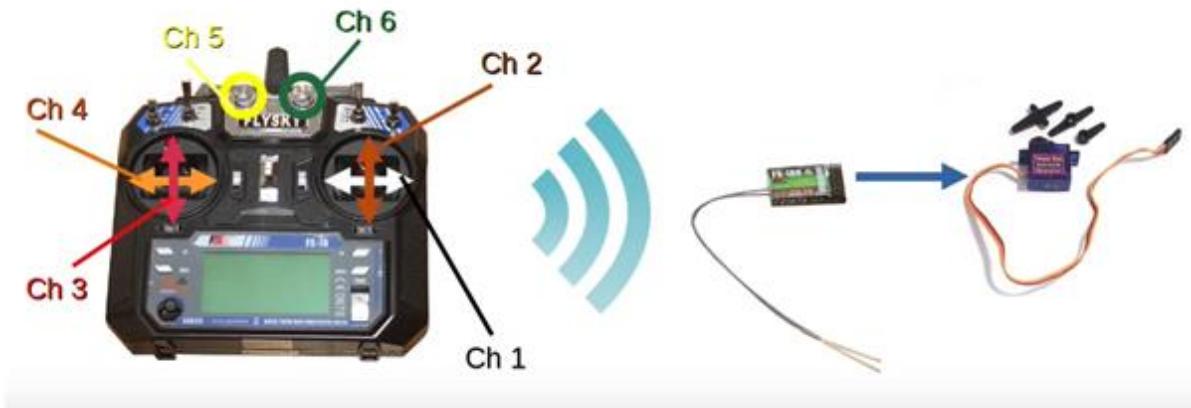
control()

```

La seconde application du protocole PWM est pour la réception des instructions reçue par l'antenne FS-IA6B 2.4GHz. Chaque Channel est relié à une sortie PWM et correspond à un débattement du sous-marin.

Le degré d'inclinaison des manettes de la télécommande indiquera une valeur du rapport cyclique, qui sera ensuite traduit puis envoyé au driver du moteur correspondant au débattement. On utilisera la fonction Input Capture, permettant de mesurer le temps entre les fronts montants et descendants consécutifs. Ces temps permettent ensuite de définir la valeur du rapport cyclique de l'instruction envoyée, et de la, régler en conséquence la puissance et le sens de rotation des moteurs.

[Figure 41 : Description des channels de la télécommande Flysky]



3.2.2 Protocole I2C

Description	du	protocole	:
--------------------	-----------	------------------	---

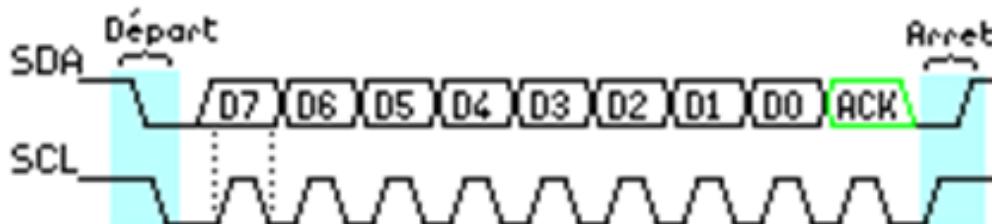
Le bus I2C permet de faire communiquer entre eux des composants électroniques grâce à trois fils : un signal de donnée (SDA), un signal d'horloge (SCL), et un signal de référence électrique (Masse). Les données sont transmises en série à 100 Kbits/s en mode standard et jusqu'à 400 Kbits/s en mode rapide. Cela ouvre la porte de cette technologie à toutes les applications où la vitesse n'est pas primordiale.

Le protocole I2C définit la succession des états logiques possibles sur SDA et SCL, et la façon dont doivent réagir les circuits en cas de conflit. Pour prendre le contrôle du bus, il faut que celui-ci soit au repos (SDA et SCL à '1'). Pour transmettre des données sur le bus, il faut donc surveiller deux conditions particulières :

- La condition de départ. (SDA passe à '0' alors que SCL reste à '1')
- La condition d'arrêt. (SDA passe à '1' alors que SCL reste à '1')

Lorsqu'un circuit, après avoir vérifié que le bus est libre, prend le contrôle de celui-ci, il en devient le maître. C'est toujours le maître qui génère le signal d'horloge.

[Figure 42 : Trame I2C]



Après avoir imposé la condition de départ, le maître applique sur SDA le bit de poids fort D7. Il valide ensuite la donnée en appliquant pendant un instant un niveau '1' sur la ligne SCL. Lorsque SCL revient à '0', il recommence l'opération jusqu'à ce que l'octet complet soit

transmis. Il envoie alors un bit ACK à '1' tout en scrutant l'état réel de SDA. L'esclave doit alors imposer un niveau '0' pour signaler au maître que la transmission s'est effectuée correctement. Les sorties de chacun étant à collecteurs ouverts, le maître voie le '0' et peut alors passer à la suite.

Applications dans le projet :

L'utilisation de l'I2C permet, dans le cadre de notre projet, de récupérer des mesures de pression précises grâce au capteur MS5837.

[Figure 43 : Exemple de code, permettant de récupérer la valeur de la pression]

```
import ms5837
import time

sensor = ms5837.MS5837_30BA() # Default I2C bus is 1 (Raspberry Pi 3)

# We must initialize the sensor before reading it
if not sensor.init():
    print("Sensor could not be initialized")
    exit(1)

# Print readings.
while True:
    if sensor.read():
        print("P: %0.1f mbar  %0.3f psi\tT: %0.2f C  %0.2f F") % (
            sensor.pressure(), # Default is mbar (no arguments)
            sensor.pressure(ms5837.UNITS_psi), # Request psi
            sensor.temperature(), # Default is degrees C (no arguments)
            sensor.temperature(ms5837.UNITS_Farenheit))) # Request Farenheit
    else:
        print("Sensor read failed!")
        exit(1)
```

Les valeurs seront enregistrées sur un fichier et pourront être récupérées pour analyse lors de la récupération de l'UUV.

3.3 Étude des programmes

Tous les flowcharts suivants sont regroupés en un seul dans la partie annexe afin de voir tous les blocs assemblés ensemble.

3.3.1 Communication Récepteur 2.4GHz – Raspberry Pi 4B

Pour commencer la programmation pour le contrôle des moteurs ainsi que de notre bandeau LED, nous avons mis en place un code permettant la réception des données envoyées par la télécommande radio 2.4GHz. Ces données sont envoyées et réceptionnées sur des pins de la Raspberry Pi 4B. Voici la structure du code mis en place pour cette étape.

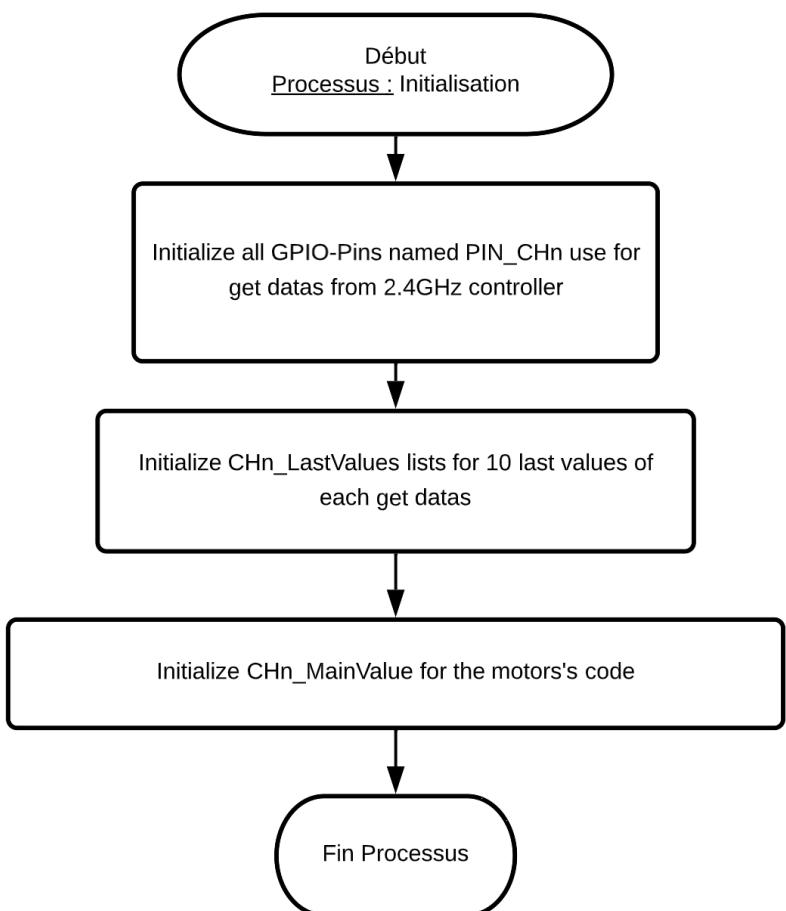
3.3.1.1 Définition d'un programme d'initialisation

Cette partie du programme sert à l'initialisation de chacune des valeurs qui nous seront utiles pour le bon fonctionnement de notre programme.

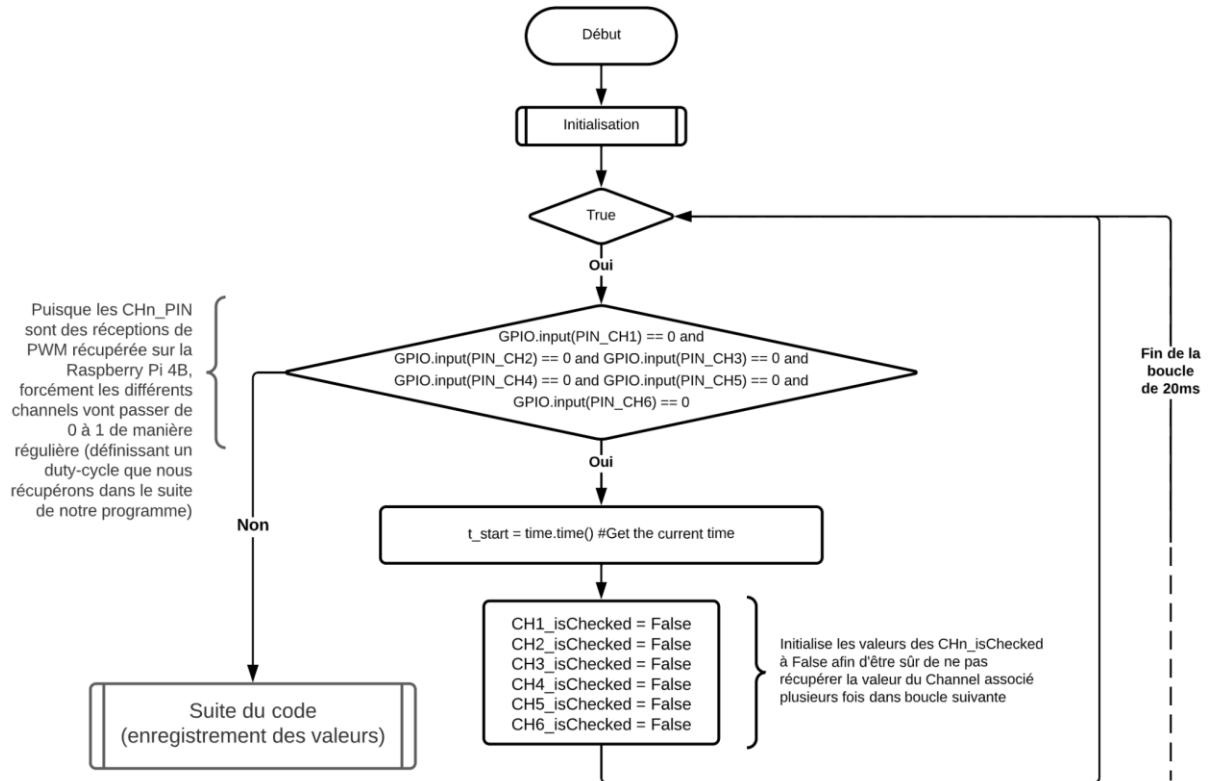
On peut retrouver la nomination de chacun des pins servant à la réception des duty-cycles des PWM réceptionné depuis le récepteur 2.4GHz.

On y retrouve également un tableau regroupant les 20 dernières valeurs de chacune des valeurs récupérées par le récepteur radio.

Ainsi que l'initialisation de 6 variables définissant la valeur la plus présente dans le tableau des 20 dernières valeurs récupérées (servant à unifier les mouvements sur une seule valeur majeure - plus efficace qu'une moyenne).



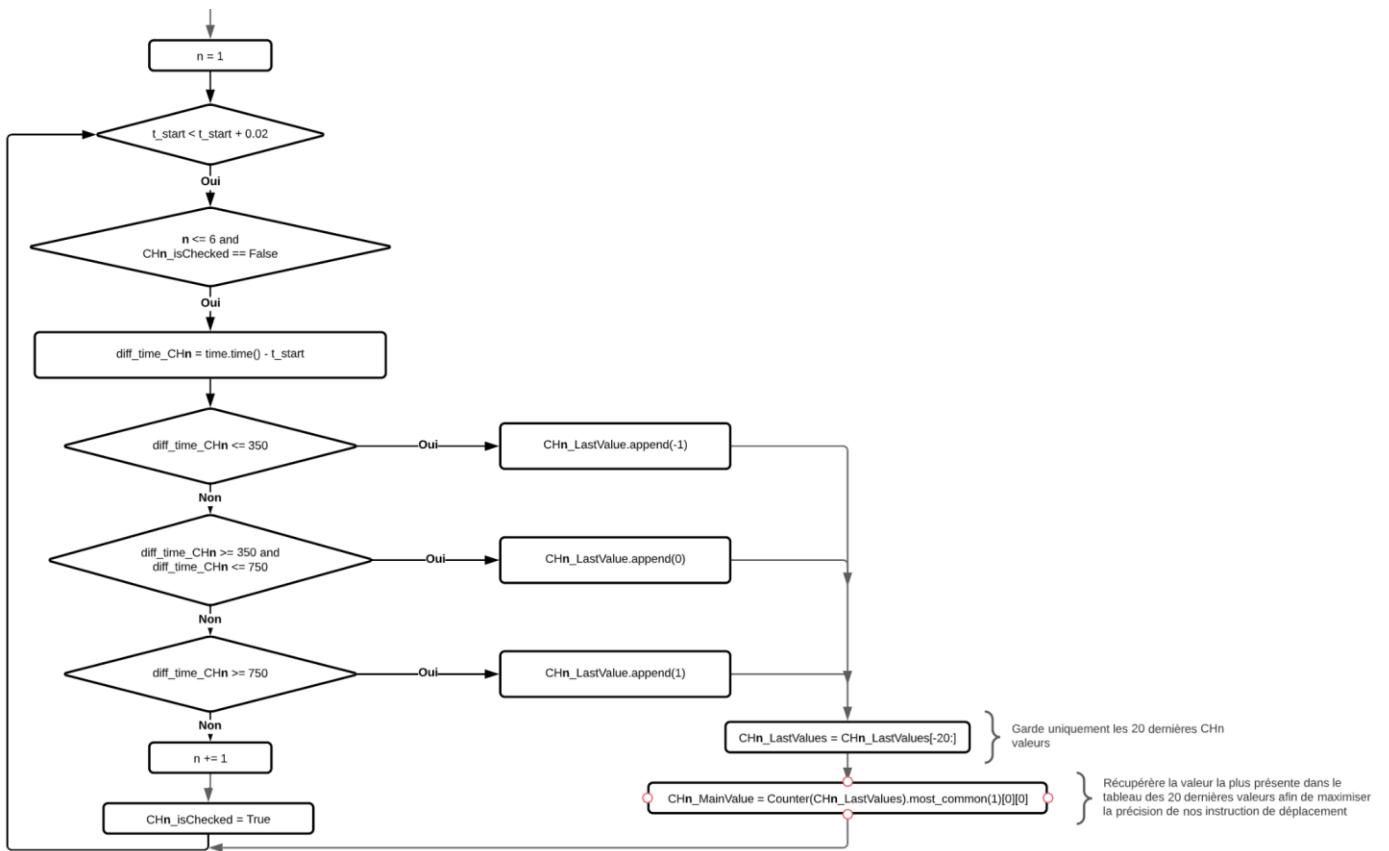
3.3.1.2 Début du programme et de la boucle sans fin



Dans cette partie du programme, on retrouve l'appel à la fonction d'initialisation des différentes valeurs utiles pour la récupération des valeurs envoyées par la télécommande radio.

Puisque les valeurs que nous voulons récupérer sont des duty-cycles définis sur une même période de 20ms, nous savons que tous les GPIO seront tous à un état bas à chaque début de nouvelle période. Ainsi, cette condition nous permet de récupérer le `current time` qui sera utile pour la boucle suivante. Nous définissons également des valeurs `isChecked` afin de récupérer uniquement une seule et unique valeur de duty-cycle par channel par période.

3.3.1.3 Récupération des duty-cycles pour les utiliser pour les moteurs

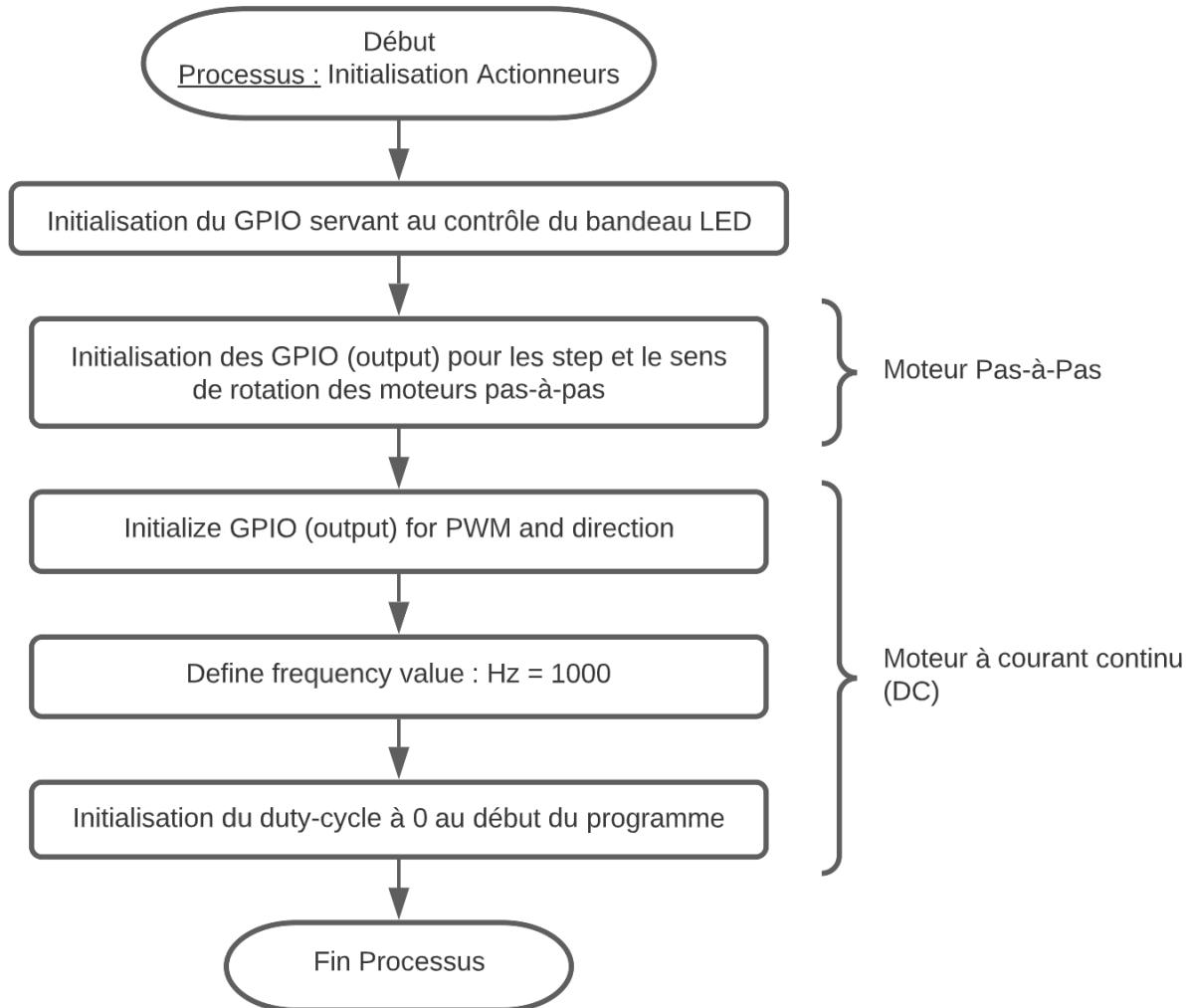


Ci-dessus, le flowchart représente la suite du code précédent et définit la réception des valeurs de chaque duty-cycle (un par channel), ainsi que le traitement de ces données afin d'obtenir des valeurs idéales à la gestion de nos différents moteurs pour le contrôle directionnel du sous-marin.

Cette récupération de valeur se fait toutes les 20ms et pour les 6 channels servant au contrôle de nos 4 moteurs pas-à-pas, de notre moteur à courant continu et de notre bandeau LED (une simple modification du flowchart est nécessaire pour la gestion du bandeau LED de manière plus efficace cependant le principe reste le même donc pas détaillé dans le flowchart ci-dessus).

3.3.2 Contrôle des moteurs après la réception des commandes faites sur la radiocommande

3.3.2.1 Initialisation des variables de la partie moteurs et LED

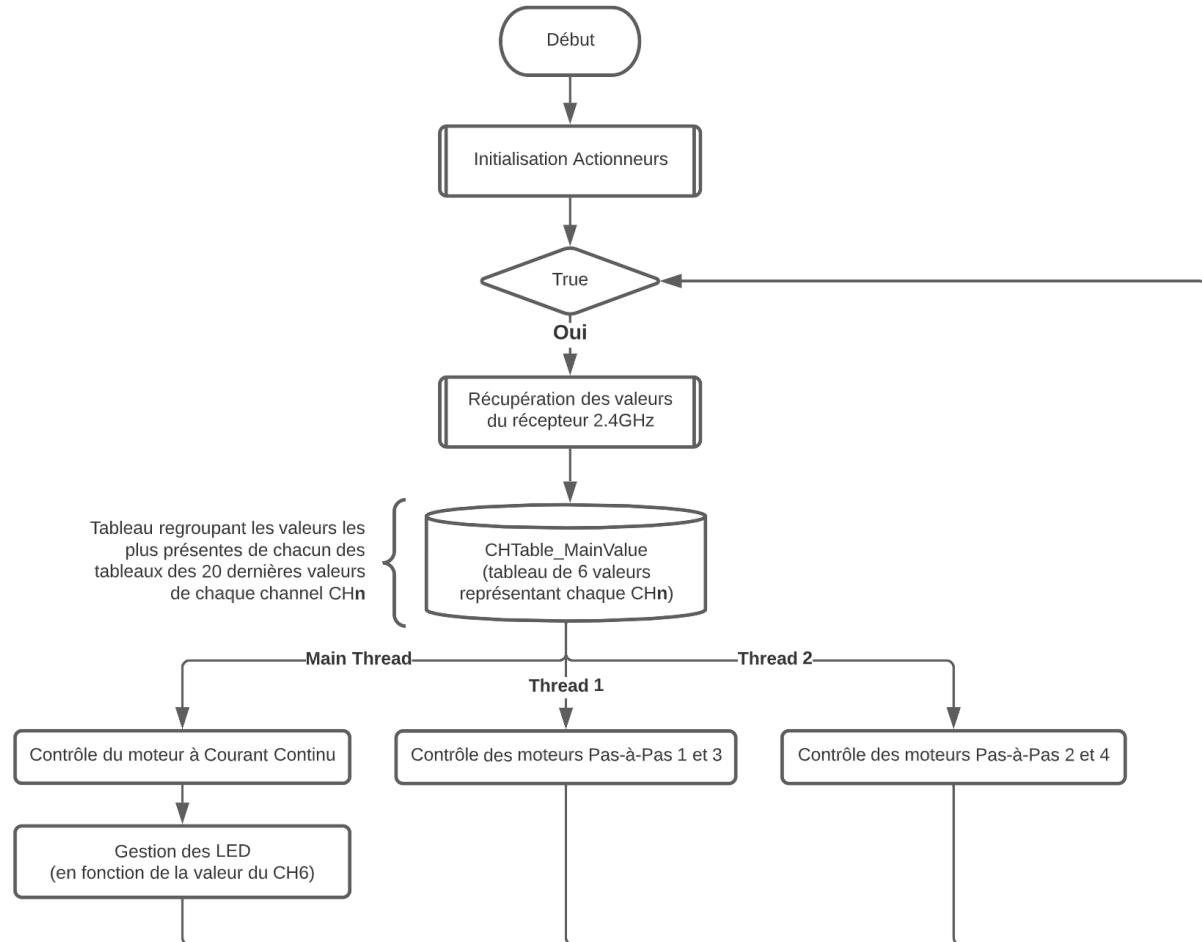


Cette partie du programme sert à l'initialisation de chacune des valeurs qui nous seront utiles pour le bon fonctionnement de notre programme (ici la gestion de nos moteurs et de notre bandeau LED).

On peut retrouver la nomination de chacun des pins servant aux contrôle des moteurs pas-à-pas, moteurs à courant continu et du bandeau LED. Nous ajoutons également le duty-cycle du moteur DC par sécurité (afin d'éviter un éventuel démarrage du moteur DC lors du lancement du drône).

3.3.2.2 Gestions des moteurs en multi-thread

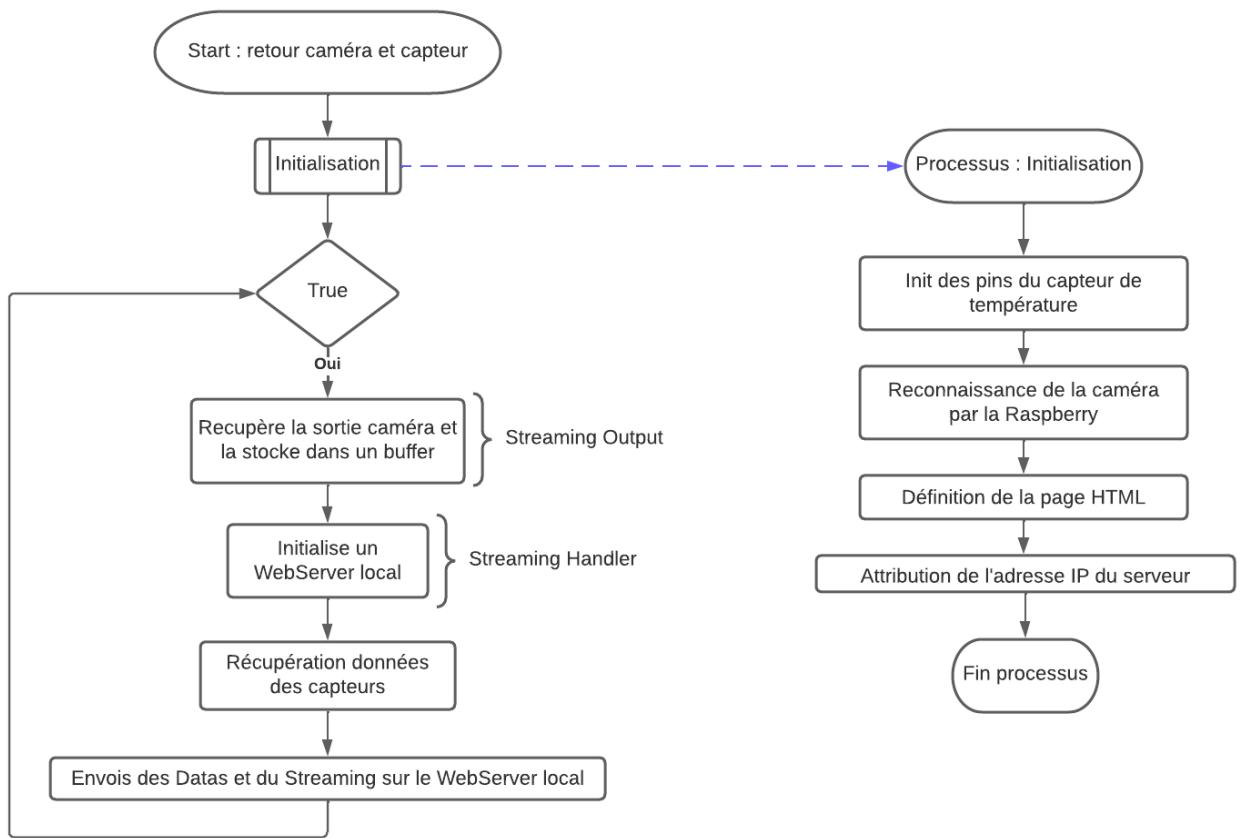
Afin de simplifier au maximum les codes, nous avons mis en place le flowchart suivant :



Dans ce flowchart, qui prend sa place dans la même boucle infini que celle de la réception des données provenant du récepteur 2.4GHz (récepteur radio), nous pouvons définir un tableau regroupant toutes les valeurs utiles au contrôle de nos moteurs et de notre bandeau LED. Ce tableau regroupe dans 6 valeurs (CH1 jusqu'à CH6).

Comme le montre ce programme, nous avons mis en place une gestion des moteurs par multithread, c'est-à-dire que nous utilisons plusieurs coeurs du microprocesseur de notre Raspberry Pi 4B. Chacun des coeurs à une fonction bien précise. Le premier cœur sert à la réception des données radios, à la gestion du moteur DC (courant continu) ainsi que du contrôle du bandeau LED. Le 2ème et 3ème cœur servent respectivement à la gestion des moteurs pas-à-pas 1-3 et 2-4. Les moteurs pas à pas étant contrôlés par paire afin de permettre des mouvements tels que ceux de translation et rotation (la translation étant le mouvement prioritaire dans tous les cas). Ainsi, de part ses multiples coeurs, notre sous-marin est capable de prendre de la profondeur tout en faisant une translation latérale (straf movement).

3.3.3 Gestion de la caméra et des capteurs sur un Web-Serveur local



Ce flowchart décrit comment nous réceptionnons le streaming vidéo de la caméra ainsi que les données des capteurs. La caméra n'utilisant pas de pins GPIO, elle doit d'abord être reconnue par la Raspberry. Le pin data du capteur de température est le seul nécessitant une initialisation (initialisé en INPUT).

La forme de la page du WebServer local est définie en format HTML directement dans le fichier Python. On référence aussi l'adresse IP qui sera utilisée pour se connecter et recevoir le retour caméra.

L'initialisation achevée, on débute la boucle infinie while True: permettant le transfert des données sur le serveur. Nous utiliserons trois classes : StreamingOutput pour récupérer le streaming caméra et le stocker dans un buffer. Il est ensuite transféré en même temps que la valeur du capteur de température vers le WebServer, défini par la classe StreamingHandler.

Une boucle Try permet de modifier la résolution de la caméra et son orientation.

4 – Bilan

4.1 Bilan financier

IC				
Produit	Quantité	prix/unité (€)	RoHS	Manufacture
Raspberry Pi 4B - 2Go	1	159.99	Oui	Raspberry Pi
Raspberry Pi 3B+	1	139.00	Oui	Raspberry Pi
Printed Circuit Board - Raspberry Pi 4B	1	8.77	N/A	PCBWay
Printed Circuit Board - Raspberry Pi 3B+	1		N/A	PCBWay

Mémoire Programme				
Produit	Quantité	prix/unité (€)	RoHS	Manufacture
Carte Micro SD Classe C	2	6,53 soit 13.06	N/A	Kodak

Connecteurs				
Produit	Quantité	prix/unité (€)	RoHS	Manufacture
Raspberry Camera Adapter 20cm	1	1.98	Oui	Adafruit Industries LLC
câble électrique multipolaire	3	6.47 soit au total 19.41	N/A	ManoMano
Câble flexible pour Raspberry Pi camera	1	1.83	Oui	Adafruit
Rallonge câble USB	1	7.49	Oui	Ralink

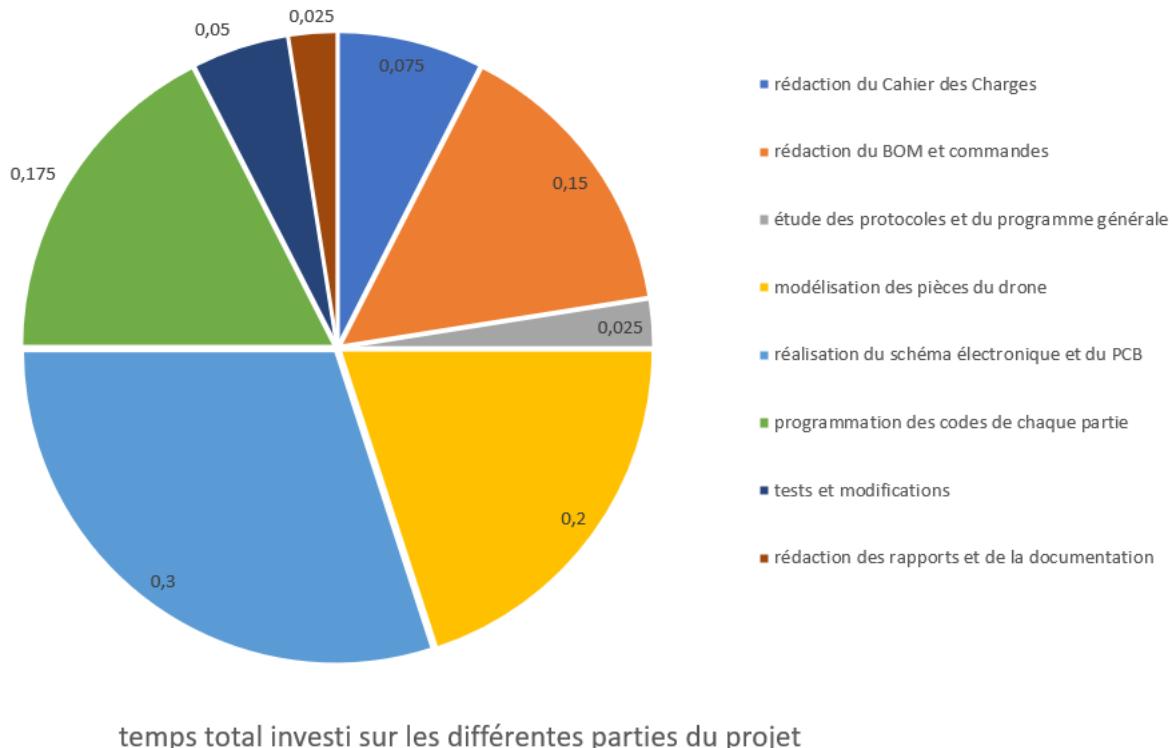
Externes				
Produit	Quantité	prix/unité (€)	RoHS	Manufacture
Raspberry Camera Module	1	24.11	Oui	Raspberry Pi / Seeed Studio
Télécommande 2.4G (+ Émetteur/Récepteur)	1	75.99	N/A	FlyskyRC
Bandéau LED (blanc) Longueur 1m - Norme IP67	1	13.2784	Oui	BTF-LIGHTING
Capteur de Température Étanche	1	6.84	Oui	DFROBOT
Capteur de Pression Étanche	1	14.13	Oui	TE CONNECTIVITY
Moteur courant continu	1	23.2 soit au total 46.4	N/A	JMT
Driver moteur courant continu	1	25.98 soit au total 51.96	N/A	BGNing
Moteur Pas-à-Pas	4	26.99	N/A	LICHIFIT
Driver-Moteur Pas-à-Pas	4	14.99	N/A	RUIZHI
Répéteur WiFi	1	9.19	Oui	Ralink
Buck Booster	1	4.00	Oui	Joy-it
Batterie	1	18	Oui	-

Composants Passifs				
Produit	Quantité	prix/unité (€)	RoHS	Manufacture
Condensateur	4	0.1		
Résistance	3	0.05		

Au total, nous arrivons à la somme de 652.28 €.

4.2 Bilan temporel

Ci-dessous se trouvent le graphique en secteur du temps investi dans les différentes tâches du projet dans sa globalité.



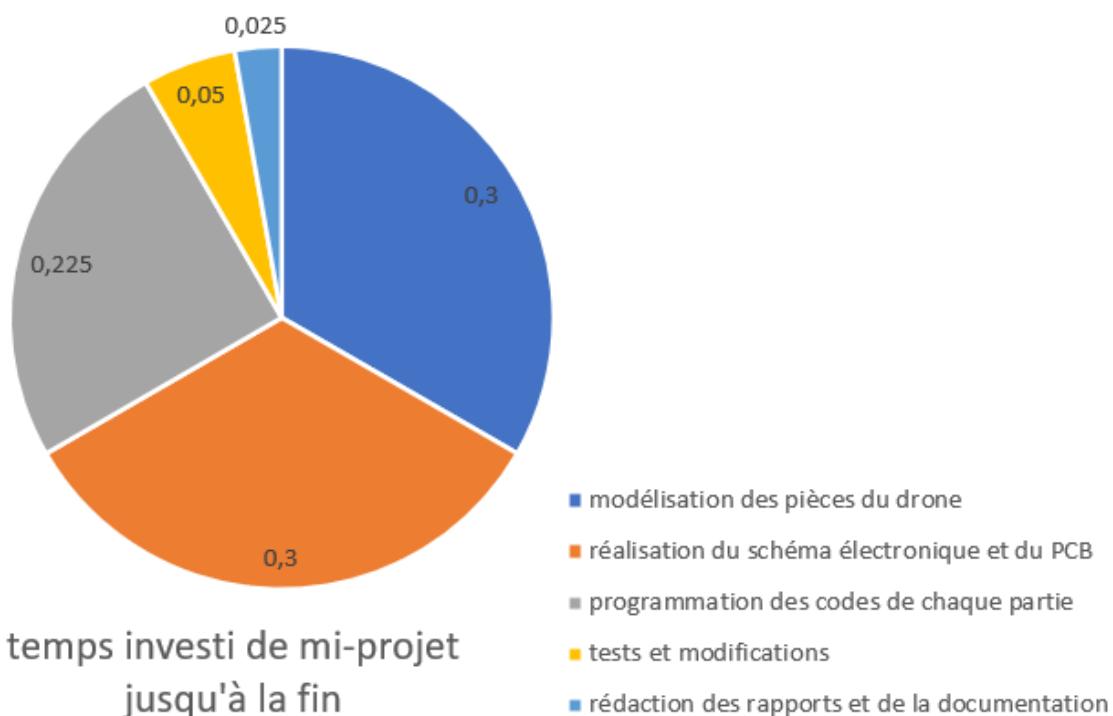
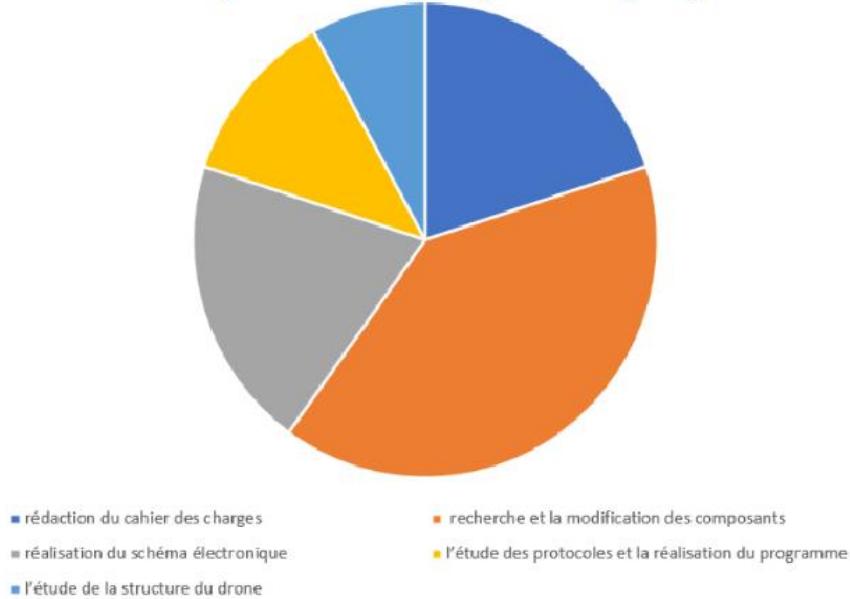
Nous pouvons constater sur graphique les tâches qui ont pris le plus de temps :

- 30% du temps a été accordée à la réalisation du schéma électronique et du PCB. A noter que ce secteur contient la réalisation des schémas électroniques dans son ensemble, incluant le premier schéma électronique, sa modification et la réalisation du schéma électronique avec la carte Raspberry supplémentaire
- 20% du temps a été alloué à la modélisation du drone et de son impression
- 17.5% du temps a été dédiée à la programmation des codes de chaque parties
- 15% du temps a été dédiée à la rédaction du BOM et à la commande des composants
- 7.5% du temps a été dédiée à la rédaction du cahier des charges avec les objectifs que nous nous sommes fixés, la forme du drone qui en résulte
- 5% du temps a été dédiée aux tests effectués et aux modifications nécessaires
- 2.5% a été dédiée à l'étude des protocoles et du programme
- 2.5% a été dédiée à la rédaction des rapports et de la documentation

Il est important de noter que nous avons travaillé certaines parties en parallèle tels que la réalisation du PCB et du schéma électronique, la modélisation du drone et des programmes. La modélisation en 3D du drone ainsi que la réalisation du PCB sont essentiels l'un pour l'autre lorsqu'il faut définir la taille du PCB et prévoir comment inclure la carte dans le sous-marin. De même, certaines parties sont plus conséquentes que d'autres car elles concentrent des travaux réalisés de façon séparée dans le temps, tels que les programmes, la commande des composants et le PCB. Les secteurs de réalisation (schéma électronique + PCB, commande des composants, rédaction du cahier des charges, programmation, et modélisation) incluent

également dans leurs parties respectives le temps consacré aux retours et modifications après les réunions de projet et l'étude de certains composants. Enfin, la partie tests contient l'ensemble des essais les plus concrets réalisés sur les composants une fois que nous les avons reçus et assemblés.

Temps investi jusqu'à mi-projet



La gestion du temps ainsi que la répartition des tâches et de leur importance a été un des points essentiels au bon avancement du projet. En annexe se trouve le Gantt théorique établie en entier.

Groupe 3S4 - SUB-EXPLORER : drone sous marin d'exploration

4.3 Conclusion

Tout d'abord, ce projet de fins d'études a été le projet le plus ambitieux que nous avons réalisé durant nos années d'études en école d'ingénieur. Nous avons investi beaucoup de temps pour concrétiser notre idée sur l'ensemble des phases de la conception du produit. En effet, nous avons imaginé le drone sous-marin et défini nos propres exigences dans le cahier des charges, puis nous avons effectué les dimensionnements nécessaires pour concevoir les schémas électroniques et les PCB. Également, nous avons modélisé et imprimé les différentes pièces selon les contraintes et objectifs, codé les programmes et enfin réalisé les tests, la validation et l'intégration de chaque partie entre elles. En travaillant tous sur chacune des différentes étapes du projet, nous possédons désormais un point de vue technique sur l'ensemble du processus de construction du drone sous-marin.

Ensuite, ce projet nous a permis d'apprendre et renforcer des connaissances multiples sur le plan humain. Nous avons travaillé en équipe autant qu'en autonomie sur différentes parties grâce à une gestion du temps et une préparation en amont sur notre emploi du temps. En se référant fréquemment à l'avancement du projet et en décomposant les tâches en sous-tâches, nous avons gardé une ligne directrice jusqu'au bout sans accumuler des retards trop conséquents. L'avancée du sous-marin s'est également effectuée en travaillant chacun en parallèle selon les atouts de chaque personne. Nous avons constaté sur ce projet plus qu'auparavant l'importance de la communication entre nous ainsi qu'avec les acteurs extérieurs avec un impact direct sur notre produit.

Également, nous avons beaucoup progressé avec ce projet sur le plan technique pour le concrétiser le plus possible. D'un point de vue Hardware, nous avons dimensionné ainsi que réalisé et imprimé nos circuits électroniques. Au niveau structural, nous avons dimensionné, modélisé, imprimé et assemblé chaque pièce entre elles. Enfin, d'un point de vue Software, nous avons programmé et testé chaque partie du drone pour les faire fonctionner ensemble. L'adaptation a été un élément essentiel à la bonne réalisation du projet face aux contraintes qui nous ont été ajoutées en plein parcours. Si nous ne sommes pas allés jusqu'au bout de certaines parties, nous avons tenu à rendre notre le produit fonctionnel dans son ensemble en priorisant les tâches principales tels que le fonctionnement entre les moteurs et la télécommande.

Finalement, notre drone sous-marin possède cette qualité d'être très adaptable et modulable. Il est possible de poursuivre le travail en implémentant des modules divers, d'optimiser le drone actuel voire de l'améliorer. A l'instar des drones standards évoluant dans les airs, ce drone sous-marin se présente comme un appareil avec de nombreuses possibilités d'avenir.

Annexes

Glossaire des Annexes

Annexes	54
Glossaire des Annexes	54
Bibliographie	55
Schéma Bloc Fonctionnel	56
Schéma Électronique – Raspberry Pi 4B	57
Schéma Électronique – Raspberry Pi 3B+	58
Schéma de routage – PCB Raspberry Pi 4B (Recto)	59
Schéma de routage – PCB Raspberry Pi 4B (Verso)	60
Schéma de routage – PCB Raspberry Pi 3B+ (Recto)	61
Schéma de routage – PCB Raspberry Pi 3B+ (Verso)	62
Vue 3D de l'assemblage final du drone	63
Vue éclatée regroupant les différentes impressions 3D du drone	64
Logigramme Code : Communication Récepteur 2.4GHz – Raspberry Pi 4B	65
Logigramme Code : Raspberry Pi 4B, Contrôle des moteurs et du bandeau LED	66
Logigramme Code : Raspberry Pi 3B+, Gestion Caméra & Capteurs	67
GANTT	68

Groupe 3S4 - SUB-EXPLORER : drone sous marin d'exploration

Schéma Bloc Fonctionnel

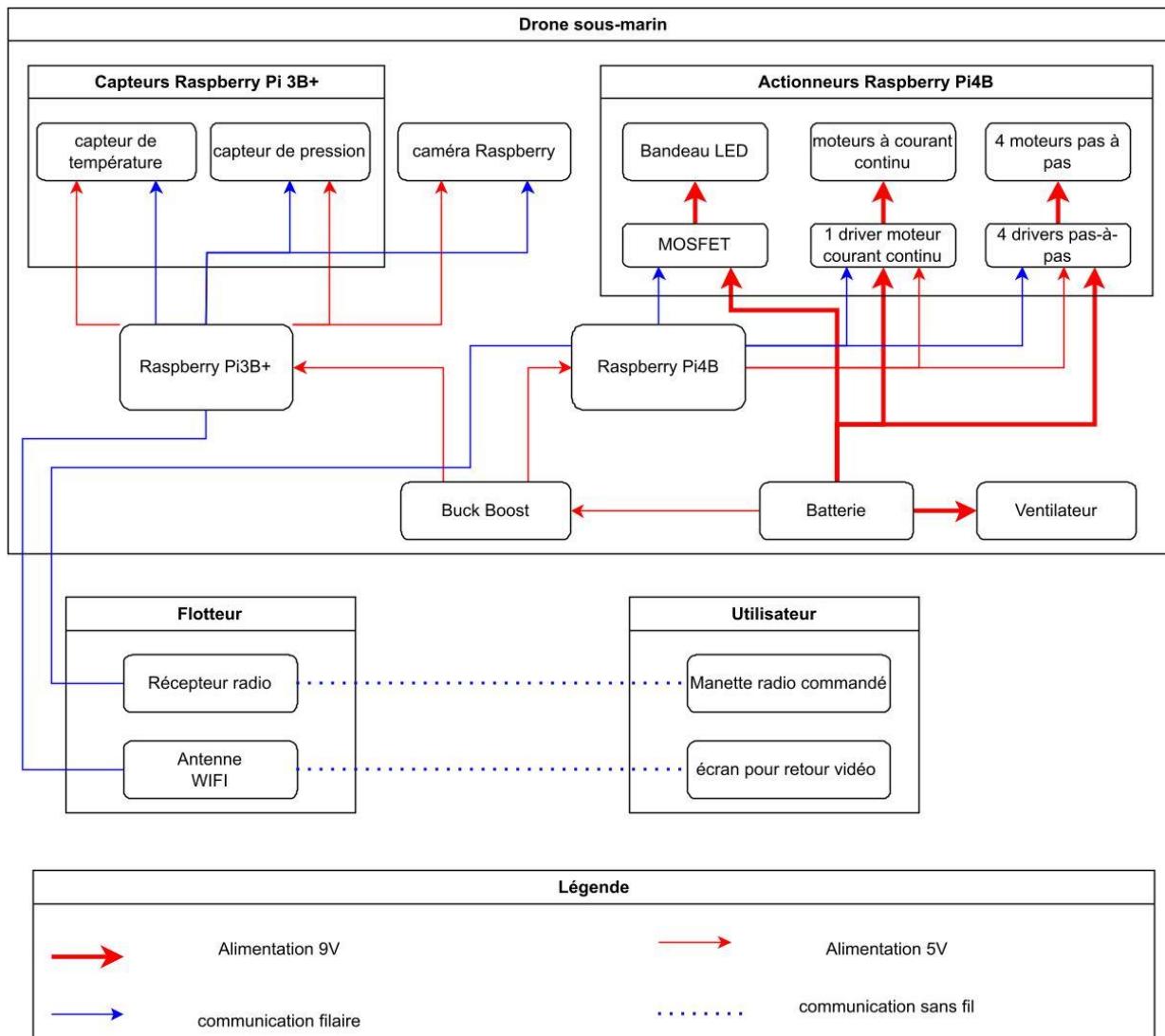


Schéma Électronique – Raspberry Pi 4B

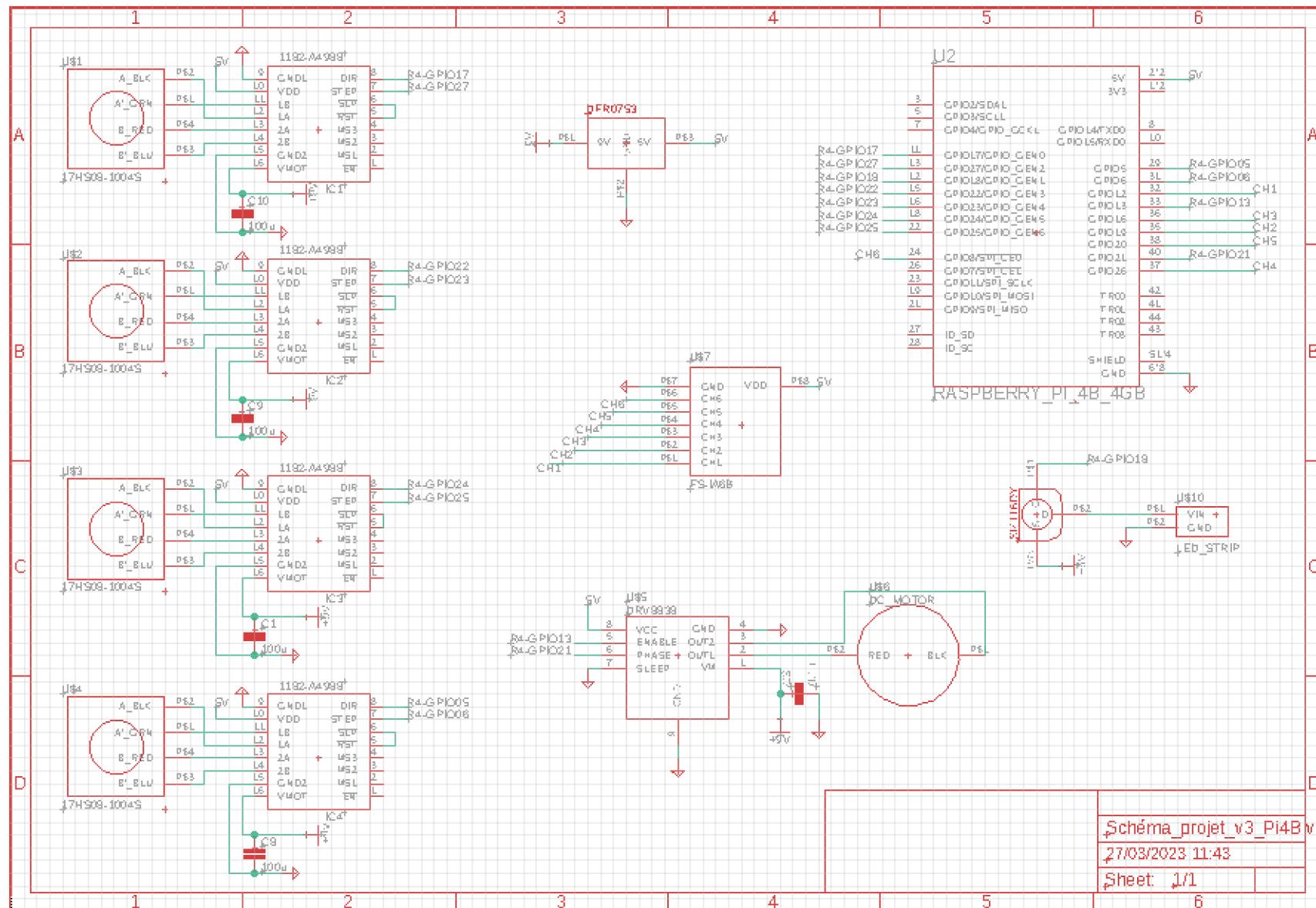


Schéma Électronique – Raspberry Pi 3B+

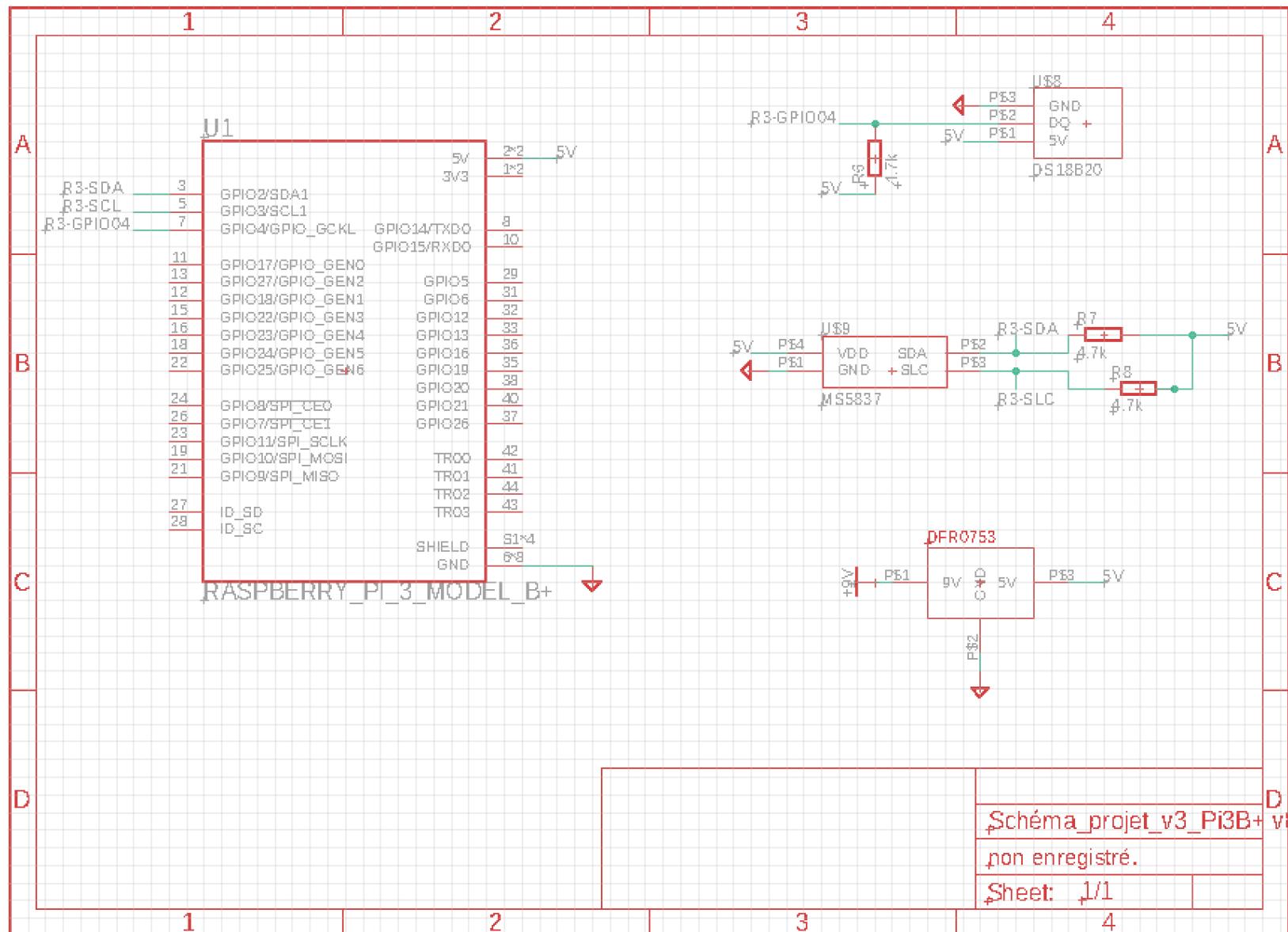


Schéma de routage – PCB Raspberry Pi 4B (Recto)

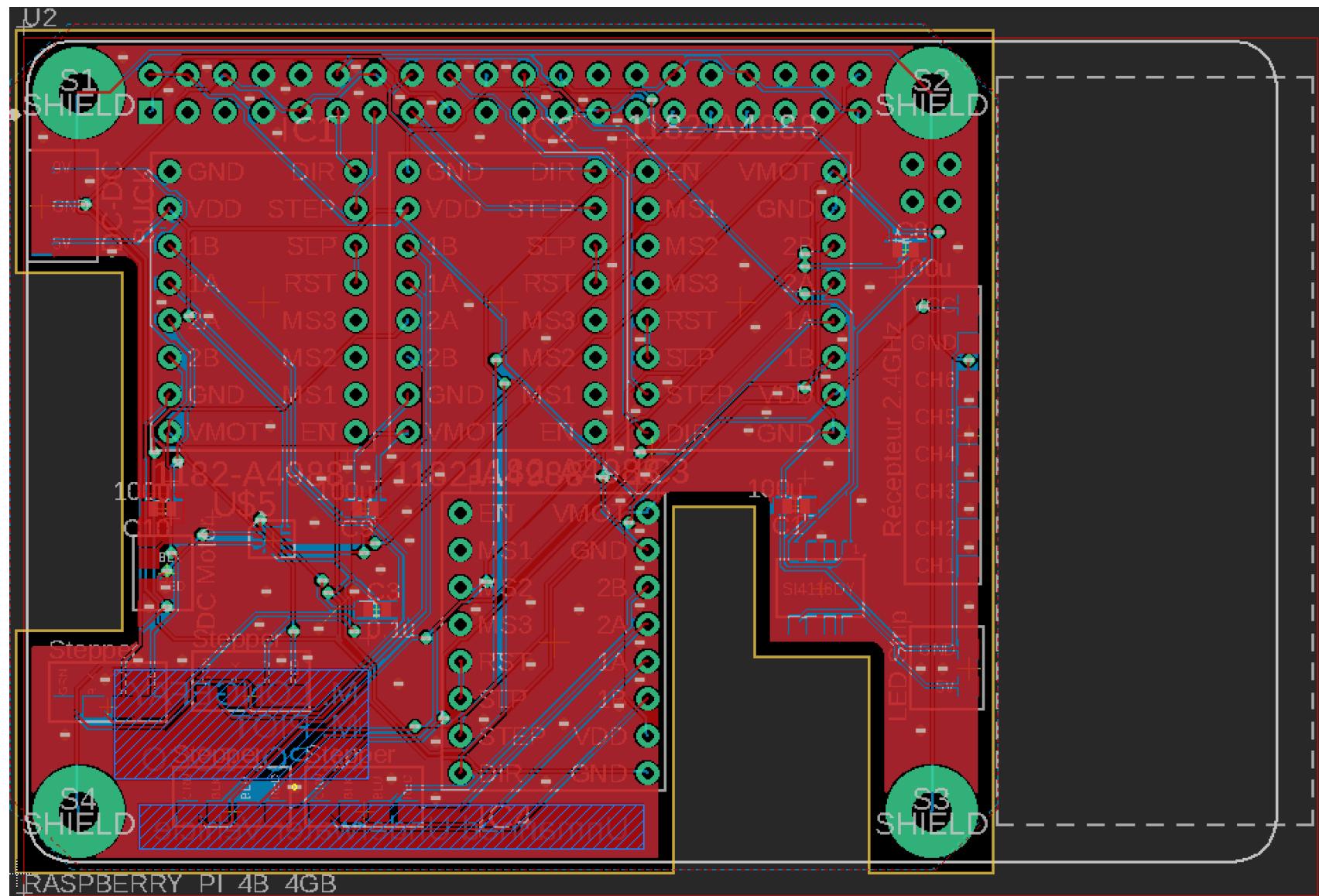
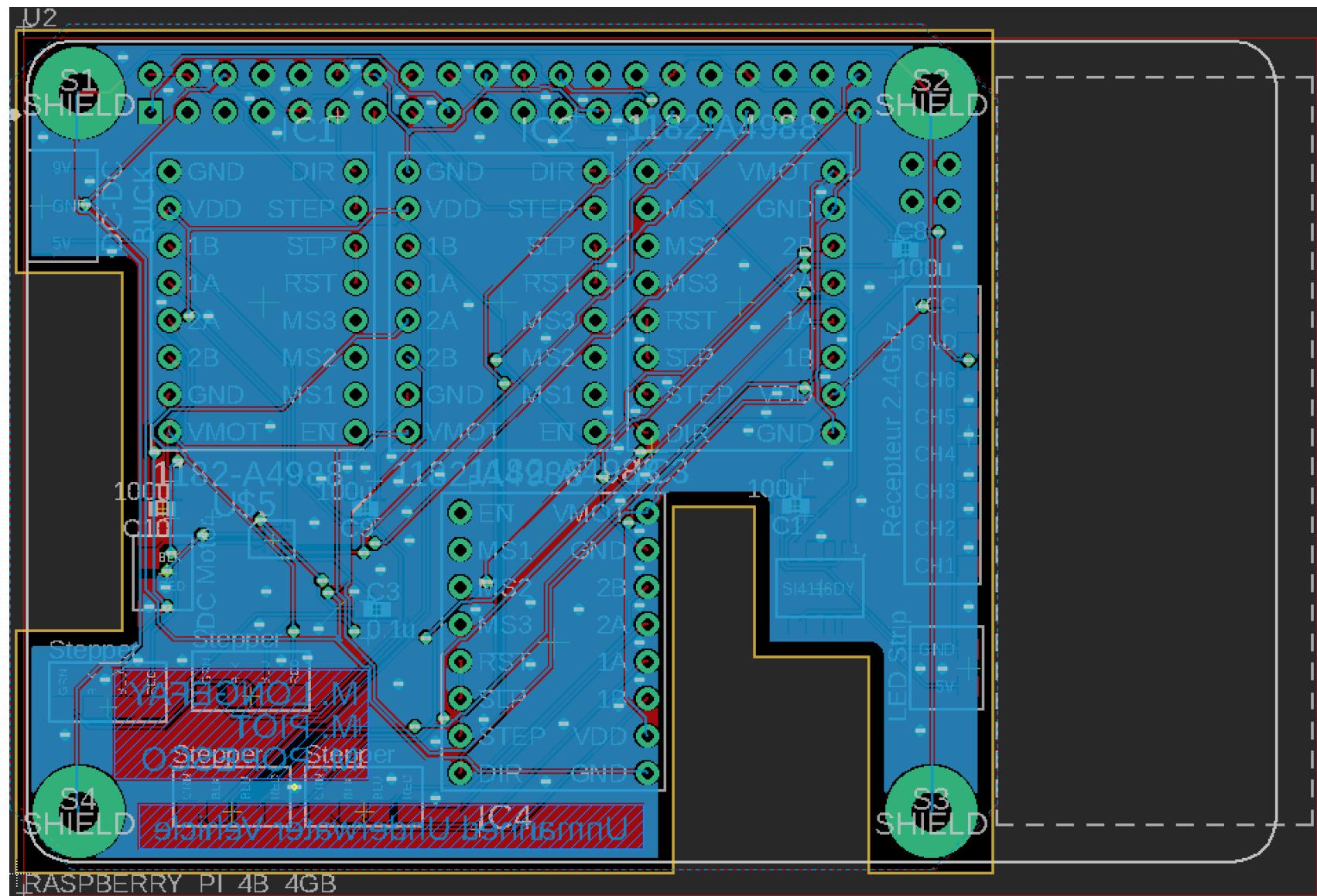
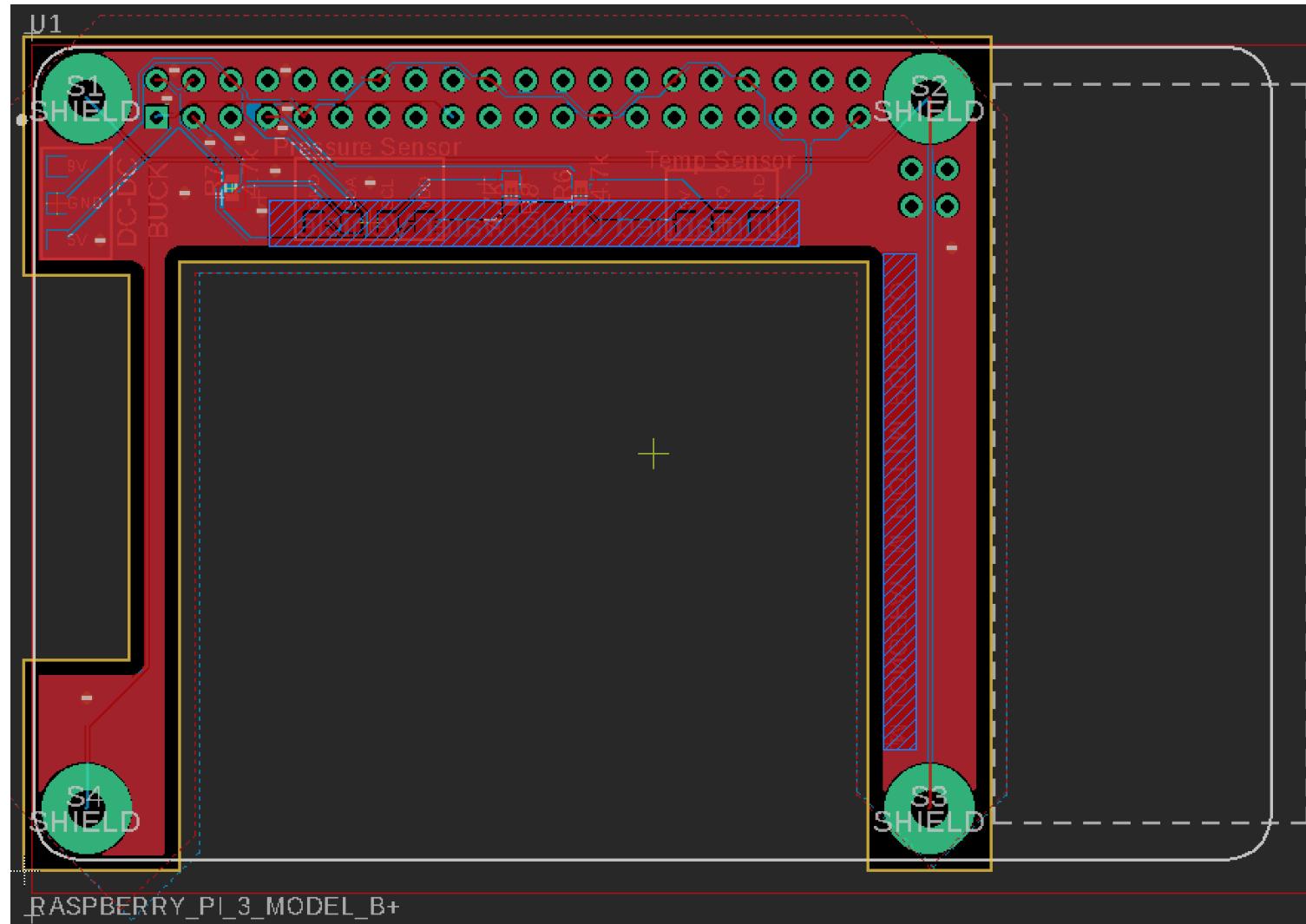


Schéma de routage – PCB Raspberry Pi 4B (Verso)



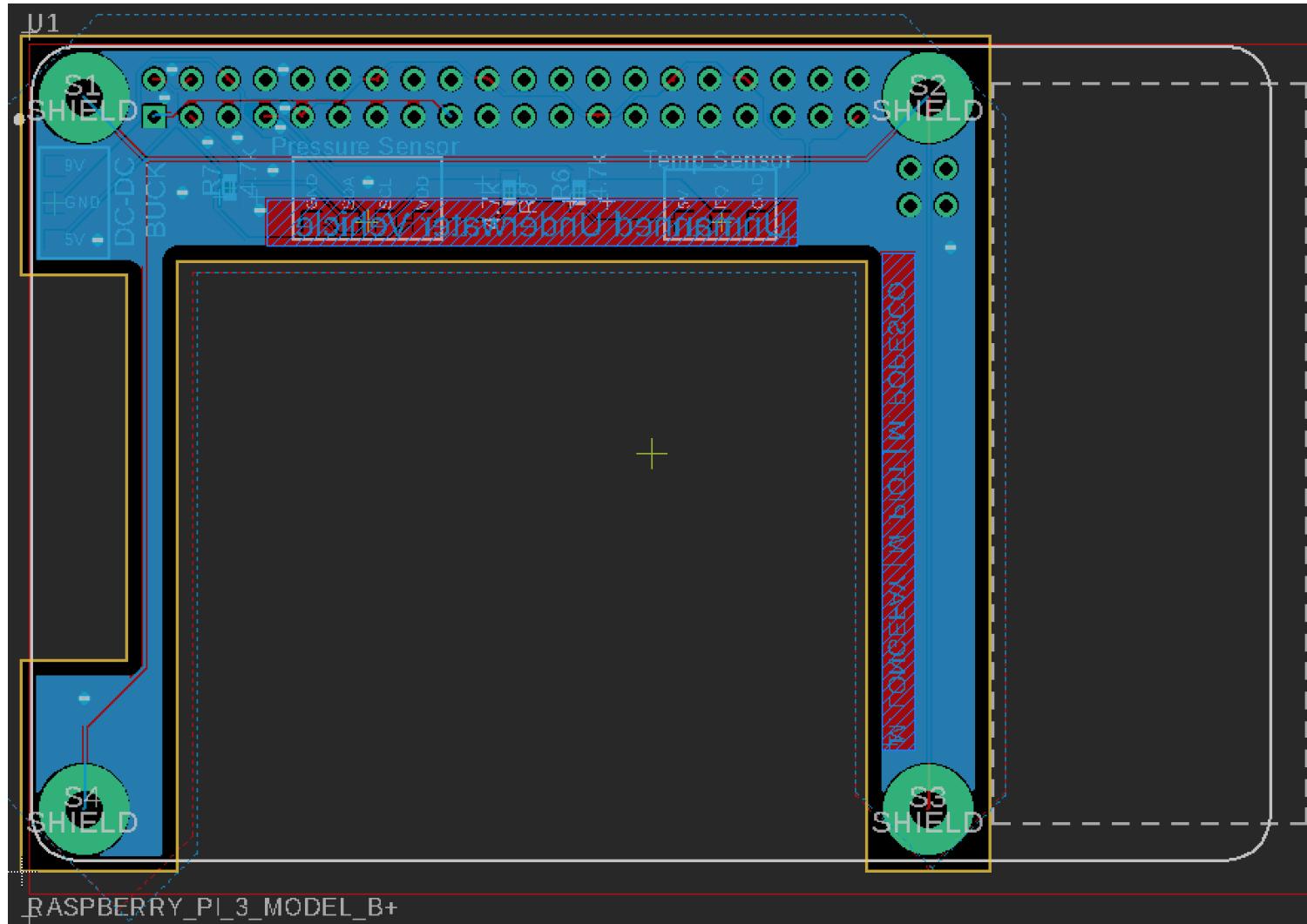
Groupe 3S4 - SUB-EXPLORER : drone sous marin d'exploration

Schéma de routage – PCB Raspberry Pi 3B+ (Recto)

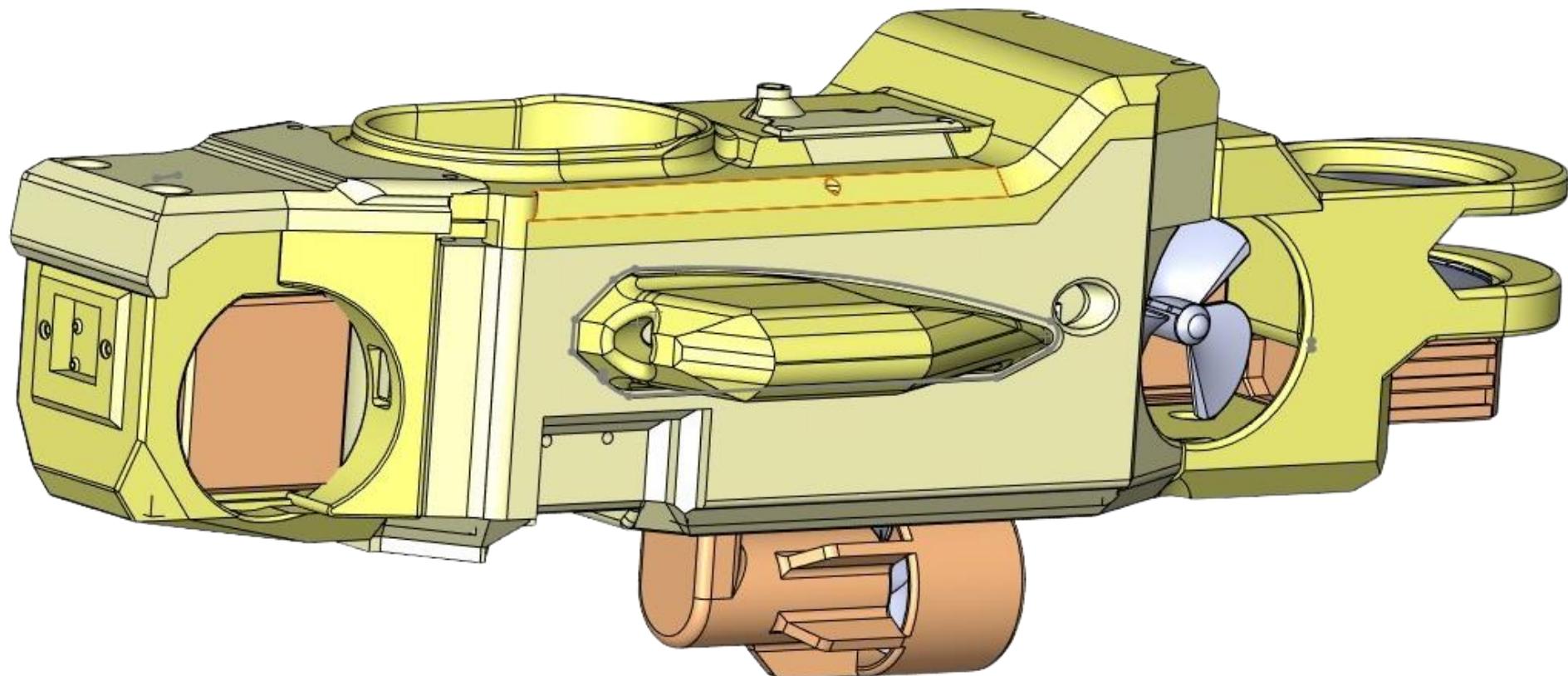


Groupe 3S4 - SUB-EXPLORER : drone sous marin d'exploration

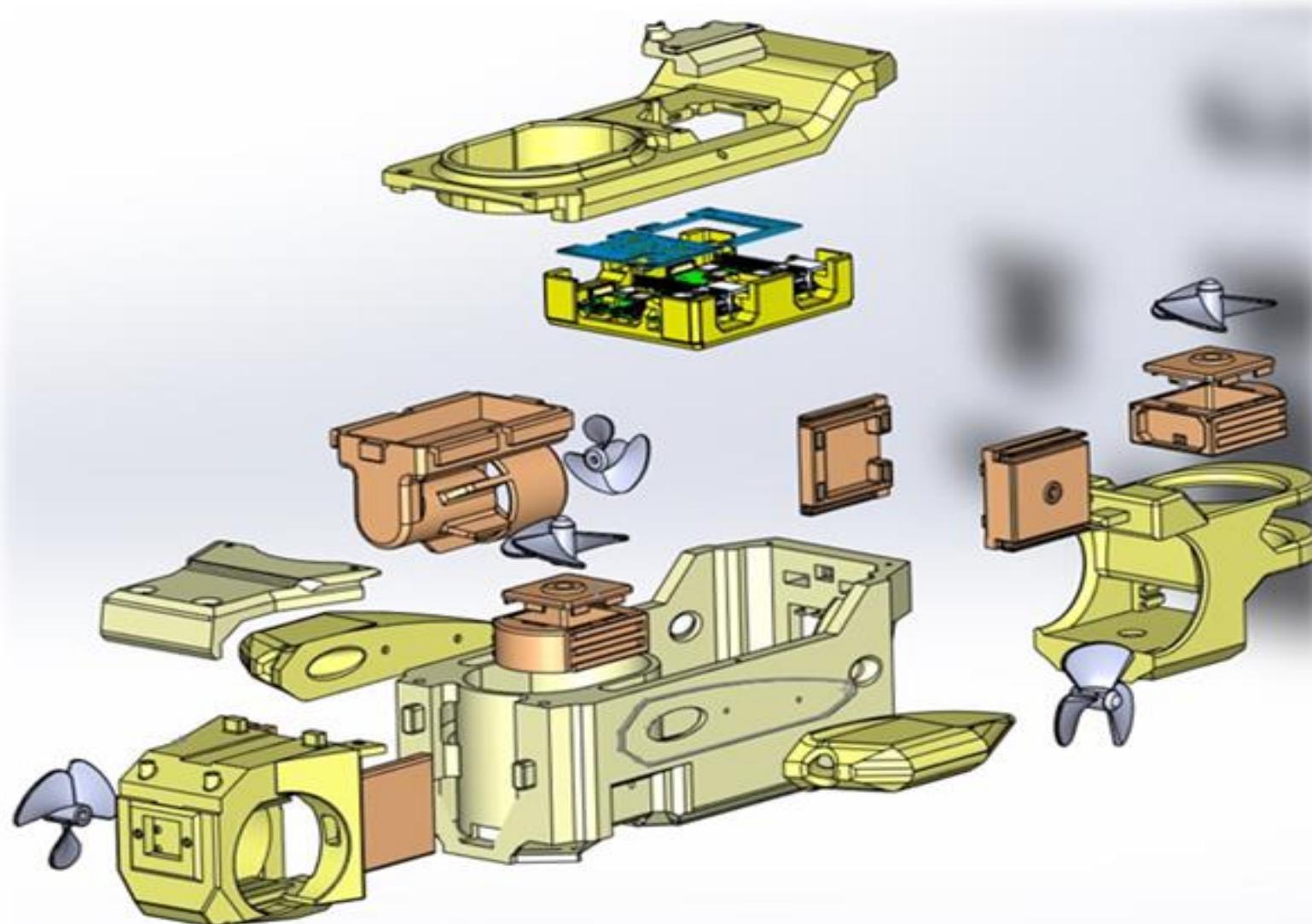
Schéma de routage – PCB Raspberry Pi 3B+ (Verso)



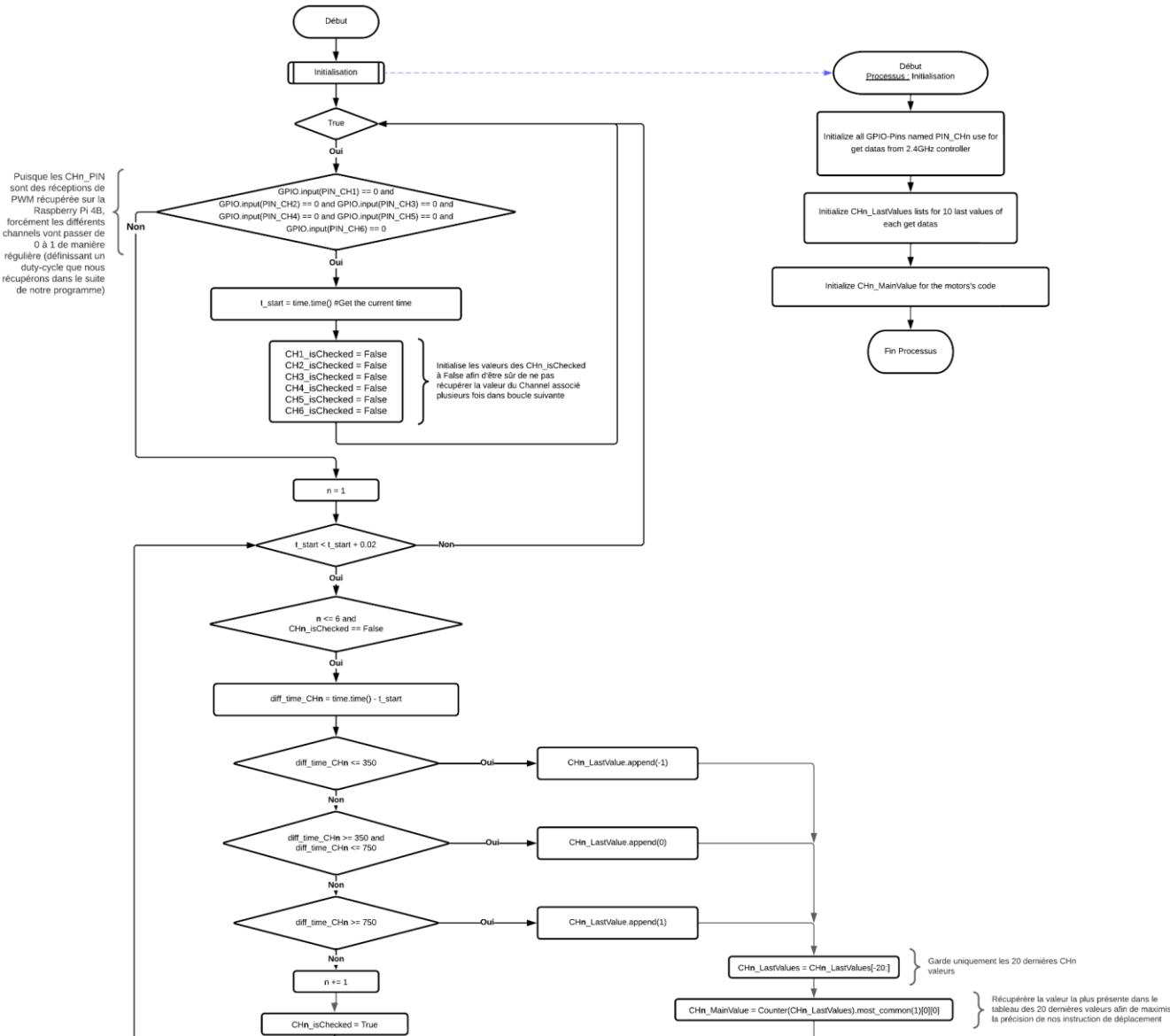
Vue 3D de l'assemblage final du drone



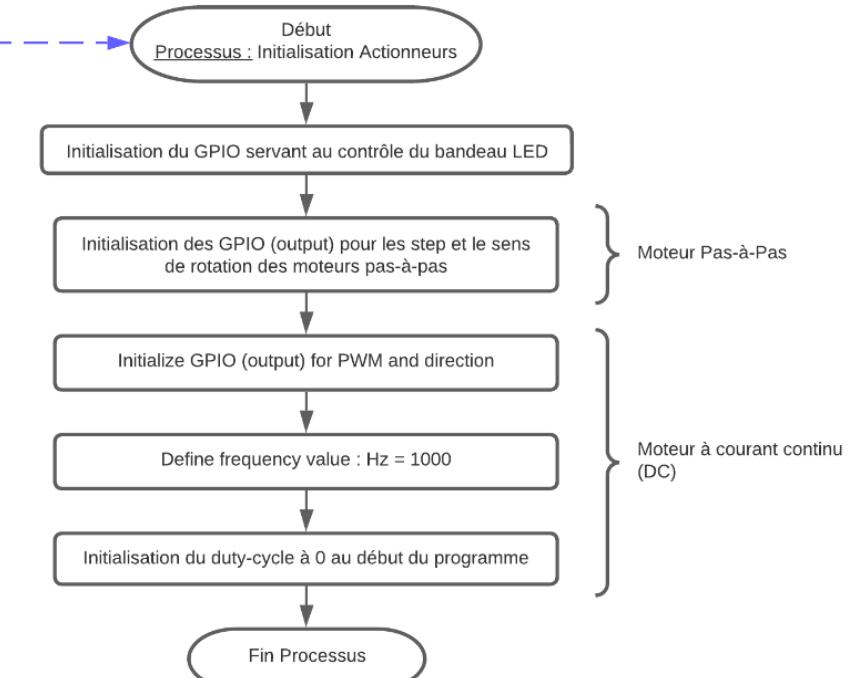
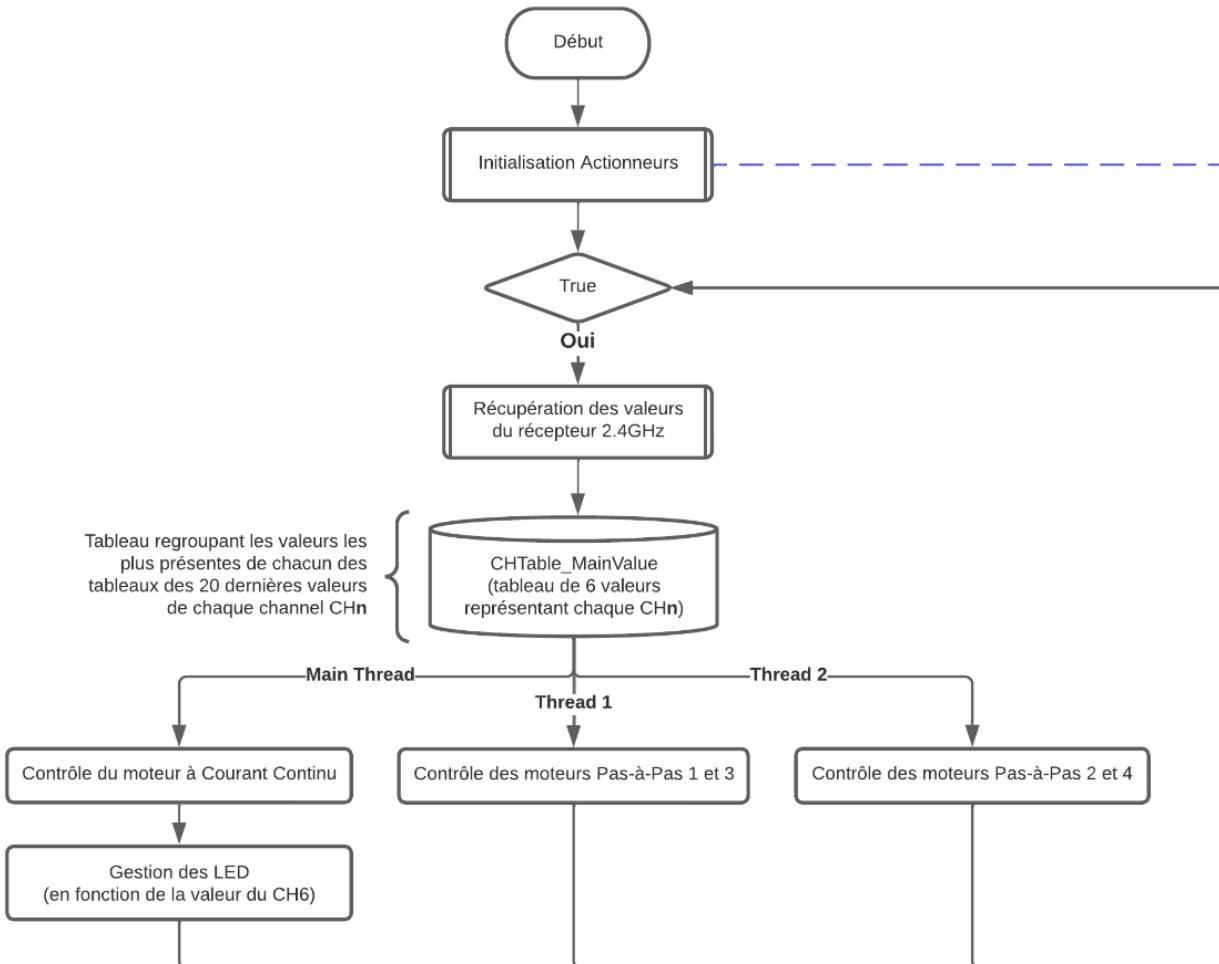
Vue éclatée regroupant les différentes impressions 3D du drone



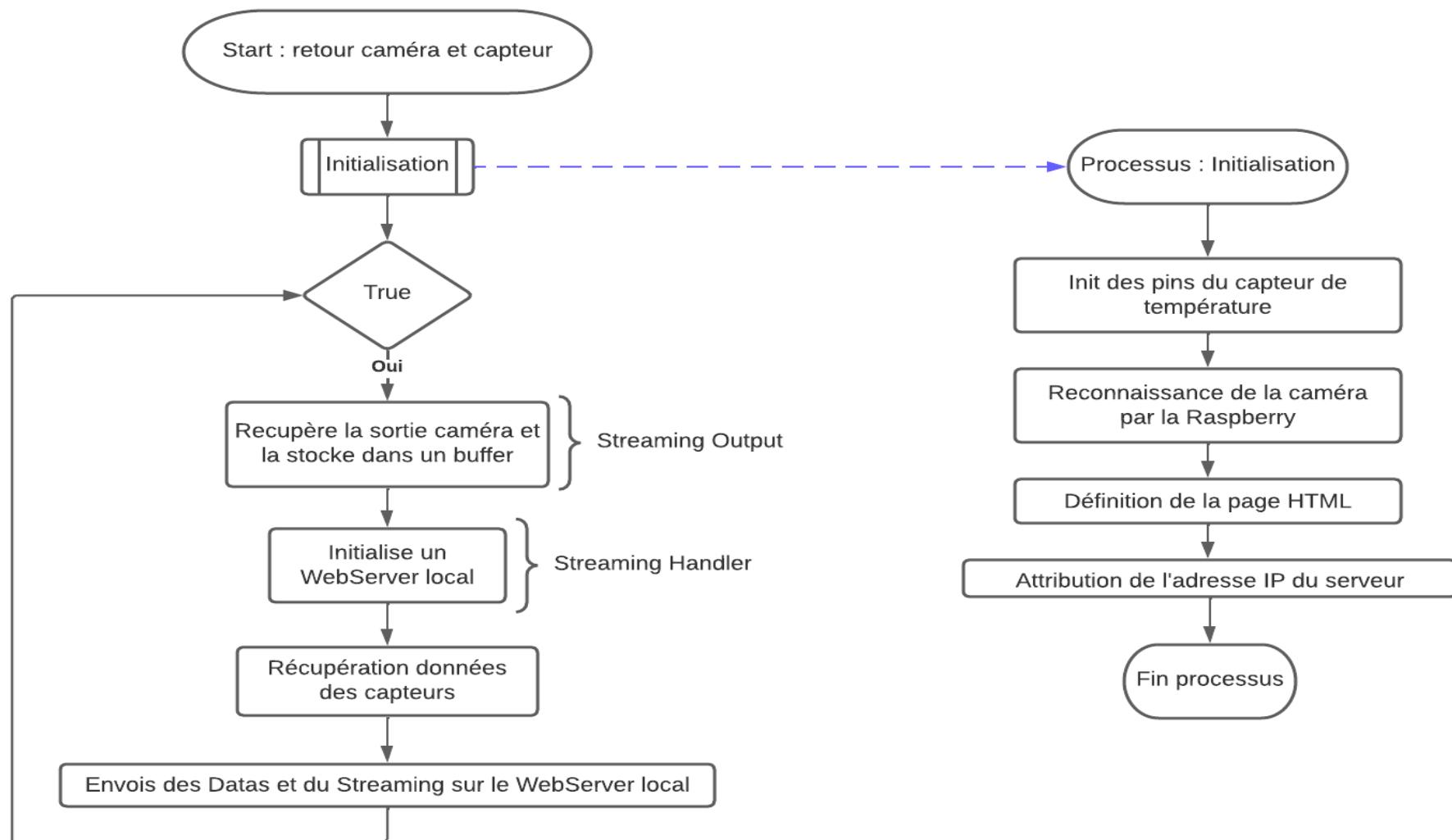
Logigramme Code : Communication Récepteur 2.4GHz – Raspberry Pi 4B



Logigramme Code : Raspberry Pi 4B, Contrôle des moteurs et du bandeau LED



Logigramme Code : Raspberry Pi 3B+, Gestion Caméra & Capteurs



Groupe 3S4 - SUB-EXPLORER : drone sous marin d'exploration

GANTT

