

LICENCE 3

Sciences pour l'Ingénieur (SPI)

Parcours Électronique,
Énergie Électrique et Automatique

RAPPORT DE STAGE

Contribution au développement d'un
système de pesée connecté
pour ruches intelligentes



Stage du 04/05/2026 au 19/06/2026



**Fablab Cohabit – Université
de Bordeaux**

IUT de Bordeaux
Bâtiment 10A
15 rue de Naudet
33170 Gradignan



RÉALISÉ PAR

Hiba Belmaate

Étudiante en Licence 3 EEA
Parcours Électronique,
Énergie Électrique et Automatique



ENCADRANTS

Hamida Hallil-Abbas
Pierre Grange Praderas
Nicolas Breard
Jean-Baptiste Bonnemaïson



Remerciements

Je tiens à exprimer ma profonde gratitude à toutes les personnes qui ont contribué au bon déroulement de mon stage et à la réalisation de ce projet.

Je remercie tout particulièrement **Monsieur Pierre Grange Praderas**, mon tuteur de stage, pour son accompagnement constant, sa disponibilité et sa confiance tout au long de cette expérience. Son encadrement quotidien, ses conseils techniques et son soutien m'ont permis de progresser aussi bien sur le plan technique que méthodologique. Sa présence à chaque étape du projet a été un véritable atout pour mener à bien les missions qui m'ont été confiées.

Mes sincères remerciements s'adressent également à **Monsieur Nicolas Breard**, développeur du projet Mellia, dont le travail a constitué une base essentielle pour ce projet de convergence entre les solutions Mellia et OpenBeeLab. Les échanges que nous avons eus m'ont permis de mieux comprendre les choix de conception réalisés ainsi que les enjeux techniques liés au développement d'une ruche connectée.

Je souhaite également remercier **Madame Hamida Hallil-Abbas** pour ses précieux conseils, son regard critique et son accompagnement tout au long du stage. Son expérience et sa vision académique m'ont aidée à prendre du recul sur mon travail, à structurer ma démarche et à améliorer aussi bien la conduite du projet que sa présentation.

J'adresse également mes remerciements à **Monsieur Jean-Baptiste Bonnemaison** pour son accueil, sa disponibilité et son implication au sein du FabLab Coh@bit. Ses conseils et sa contribution à la dynamique de l'équipe ont participé à rendre cette expérience particulièrement enrichissante.

Je tiens aussi à remercier chaleureusement **Jhodi** pour son aide précieuse tout au long du stage. Sa disponibilité, son accompagnement dans les recherches documentaires ainsi que ses nombreux conseils pratiques m'ont été d'une grande utilité dans l'avancement du projet et dans la compréhension de nombreux aspects techniques.

Enfin, je remercie l'ensemble de l'équipe du **FabLab Coh@bit** pour son accueil chaleureux, sa bienveillance et l'environnement de travail stimulant qu'elle m'a offert durant cette période. Ce stage a constitué une expérience particulièrement enrichissante, tant sur le plan professionnel que personnel, et m'a permis de développer de nouvelles compétences tout en découvrant le fonctionnement d'un projet collaboratif dans le domaine des systèmes embarqués et de l'Internet des objets.

Résumé

Ce stage a été réalisé au sein du **FabLab Coh@bit de l'IUT de Bordeaux**, dans le cadre d'un projet de développement d'une ruche connectée destinée à la surveillance à distance des colonies d'abeilles. L'objectif principal du projet était d'étudier et de faire converger deux solutions existantes, **OpenBeeLab** et **Mellia**, afin de concevoir une architecture combinant la robustesse mécanique de la première et les performances électroniques de la seconde.

Le travail effectué a consisté à analyser les architectures existantes, à définir une nouvelle solution matérielle adaptée aux besoins du projet, puis à concevoir les schémas électroniques associés. Une attention particulière a été portée à l'intégration d'un moteur pas à pas, d'un capteur optique et d'un système de mesure permettant la recherche automatique de l'équilibre de la balance. La conception du circuit imprimé (PCB), l'étude de l'affectation des entrées-sorties du microcontrôleur ainsi que le développement des algorithmes embarqués ont également fait partie des missions réalisées.

Des phases de tests et de validation ont été menées sur une carte STM32 Nucleo afin de vérifier le fonctionnement du pilotage moteur, de la détection de position et des algorithmes développés. Les résultats obtenus ont permis de valider les choix techniques retenus et de démontrer la faisabilité de la solution proposée.

Mots-clés : ruche connectée, systèmes embarqués, STM32, PCB, moteur pas à pas, capteur optique, Internet des objets (IoT), Mellia, OpenBeeLab.

Abstract

This internship was carried out at the **Coh@bit FabLab of the Bordeaux Institute of Technology (IUT de Bordeaux)**, a multidisciplinary innovation and prototyping platform dedicated to digital fabrication, embedded systems, electronics and connected objects. The project focused on the development of a smart beehive monitoring system designed to remotely supervise bee colonies while minimizing disturbances to the hive.

The main objective of the internship was to contribute to the convergence of two existing projects, namely **Mellia** and **OpenBeeLab**, in order to design a new solution combining the mechanical robustness of OpenBeeLab with the advanced electronic features of Mellia. The work involved hardware architecture analysis, electronic design, PCB development, embedded software implementation and experimental validation using STM32 microcontrollers.

This project is part of the broader field of the **Internet of Things (IoT)** and precision agriculture, which are rapidly growing research areas worldwide. Smart beehive monitoring has attracted significant interest from international research laboratories and institutions due to the crucial role of bees in biodiversity and food production. Several research groups in Europe, North America and Australia are working on connected beehive technologies, environmental monitoring systems and low-power wireless sensor networks.

Among the internationally recognized research organizations involved in related fields are the Centre National de la Recherche Scientifique in France, the Fraunhofer Society in Germany, and the Massachusetts Institute of Technology in the United States. These institutions contribute to advances in embedded systems, wireless communication, IoT platforms and smart environmental monitoring technologies.

This internship provided valuable experience in electronic system design, embedded programming and hardware validation while contributing to an innovative project at the intersection of IoT technologies and sustainable agriculture.

Keywords: Smart Beehive, Internet of Things, Embedded Systems, STM32, PCB Design, Wireless Sensors, Precision Agriculture.

Table des matières

Remerciements	2
Résumé	3
Abstract	4
Table des figures.....	7
Liste des Annexes.....	8
Cahier des Charges.....	9
Introduction générale.....	10
Chapitre I : Contexte du projet et étude de l'existant.....	11
1.1 Contexte et problématique :	11
1.2 Présentation de la structure d'accueil :	11
1.2.1 Organisme d'accueil :	11
1.2.2 Origine et organisation du projet :.....	11
1.2.3 Ressources techniques et documentaires :	12
1.2.4 Répartition du travail :.....	12
1.3 Étude des solutions existantes :	12
1.3.1 Présentation de la solution Mellia :	12
Architecture matérielle de Mellia.....	13
Analyse fonctionnelle de la balance Mellia	14
Avantages et limites de Mellia	15
1.3.2 Présentation de la solution OpenBeeLab :	15
Architecture matérielle de OpenBeeLab	16
g. Driver moteur ULN2003	18
Analyse fonctionnelle de la balance OpenBeeLab.....	19
Avantages et limites de la solution OpenBeeLab	19
1.4 Comparaison entre Mellia et OpenBeeLab	20
Chapitre II : Conception et développement de la solution proposée.....	21
2.1 Étude de la convergence entre Mellia et OpenBeeLab :	21
2.1.1 Principe et contexte de la fusion :	21
2.1.2 Bénéfices de l'architecture retenue :.....	21
2.2 Choix des éléments conservés, supprimés et ajoutés :	21
2.2.1 Éléments conservés :	21
2.2.2 Éléments supprimés :	21
2.2.3 Éléments ajoutés et modifications apportées :	22
2.3 Conception du schéma électronique et du PCB :	22
2.3.1 Élaboration du schéma électronique :	22
2.3.2 Conception du PCB :.....	23
Chapitre III : Tests et validation	24
3.1 Mise en place de l'environnement de test :	24
3.2 Configuration du microcontrôleur STM32 :	24
3.3 Tests de validation des fonctionnalités :	25
3.3.1 Test des LEDs de signalisation :	25
3.3.2 Test de pilotage du moteur pas à pas :	25
3.3.3 Test des capteurs de début et de fin de course :	26
3.3.4 Test de lecture du capteur optique par ADC :	26
3.3.5 Caractérisation de la résolution du système :	27

3.3.6 Détermination du temps de stabilisation :	27
3.3.7 Test de la recherche d'équilibre :	27
3.3.8 Test de la pesée :	28
3.4 Bilan du stage :	29
Conclusion et perspectives	30
Bibliographie	31
Documentation des projets	31
Microcontrôleurs et modules de communication	31
Capteurs et acquisition de données	31
Alimentation et régulation	32
Motorisation et électronique de puissance	32
Outils logiciels et conception	32
Protocoles et interfaces	32
Annexes	33

Table des figures

- Figure 1** : Structure métallique de la balance Mellia
- Figure 2** : Carte électronique de la balance Mellia
- Figure 3** : Schéma fonctionnel de la balance Mellia
- Figure 4** : Structure métallique de la balance OpenBeeLab
- Figure 5** : Carte électronique de la balance OpenBeeLab
- Figure 6** : Structure mécanique de OpenBeeLab
- Figure 7** : Bras de levier de la balance OpenBeeLab
- Figure 8** : Contrepoids mobile
- Figure 9** : Moteur pas à pas
- Figure 10** : Capteur optique
- Figure 11** : Driver moteur
- Figure 12** : Schéma fonctionnel de la balance OpenBeeLab
- Figure 13** : Analyse comparative des solutions Mellia et OpenBeeLab
- Figure 14** : Schéma électronique
- Figure 15** : Carte de test (Nucleo-F411RE)
- Figure 16** : Configuration de la carte de test
- Figure 17** : Capteur de fin de course
- Figure 18** : Course du contrepoids
- Figure 19** : Comparaison des convertisseurs ADC utilisés et envisagés
- Figure 20** : Répétabilité de la mesure de la course moteur
- Figure 21** : Acquisition des valeurs ADC lors du test du capteur optique
- Figure 22** : Résultats expérimentaux de la caractérisation de la résolution du système
- Figure 23** : Comparaison du temps de stabilisation pour différents écarts entre demi-pas
- Figure 24** : Résultats de la recherche d'équilibre

Liste des Annexes

Annexes	33
Annexe A – Algorithme du test de clignotement des LEDs.....	33
Annexe B – Algorithme du pilotage du moteur pas à pas.....	33
Annexe C – Résultats des essais de la détection de la course	34
Annexe D – Algorithme du test des fins de course	34
Annexe E – Algorithme du test de lecture d’ADC.....	35
Annexe F– Résultats de lecture d’ADC	36
Annexe G – Résultats expérimentaux de la caractérisation de la résolution du système.....	36
Annexe H– Algorithme du test de résolution autour de l’équilibre.....	37
Annexe I– Algorithme du test de stabilisation	38
Annexe J– Résultats du test de stabilisation.....	39
Annexe K– Algorithme de la recherche d’équilibre	39
Annexe L – Résultats de la recherche d’équilibre	40
Annexe M – Algorithmes de la pesée	40
Annexe N – Diagramme de Gantt	42

Cahier des Charges

Le projet de ruche connectée avait pour objectif de développer une solution de pesage capable de surveiller l'évolution du poids d'une ruche tout en limitant les interventions humaines. Dans cette optique, il a été décidé de s'appuyer sur les travaux réalisés dans le cadre des projets Mellia et Openbeelab afin de concevoir une architecture combinant leurs principaux avantages.

Le système développé devait être capable d'acquérir les mesures nécessaires au pesage, de piloter un moteur pas à pas assurant le déplacement du contrepoids et de gérer les informations provenant du capteur de fin de course utilisé comme référence de position. Il devait également intégrer un algorithme de recherche d'équilibre permettant d'ajuster automatiquement la position du contrepoids afin d'estimer la masse appliquée sur la balance.

Dans le cadre de ce stage, les travaux confiés consistaient à étudier les solutions existantes, à compléter leur documentation technique, à développer et adapter les fonctionnalités logicielles nécessaires au fonctionnement du système, puis à réaliser les essais permettant de valider les différentes fonctions mises en œuvre. L'objectif final était d'obtenir une solution de pesage fonctionnelle pouvant être intégrée à une ruche connectée pour assurer un suivi fiable de son état.

Introduction générale

Dans le cadre de ma troisième année de Licence Sciences pour l'Ingénieur, parcours Électronique, Énergie Électrique et Automatique (EEA), j'ai effectué un stage au sein du FabLab Coh@bit de l'IUT de Bordeaux. Ce stage s'inscrit dans le développement d'une balance connectée destinée au suivi des ruches, un outil permettant de surveiller l'évolution de leur poids et de collecter des données utiles à l'étude de l'activité des colonies d'abeilles.

Les travaux menés s'appuient sur deux projets existants, Mellia et OpenBeeLab, qui proposent chacun une solution de pesée connectée reposant sur des approches différentes. L'objectif du stage a consisté à participer à la conception d'une nouvelle version du système en combinant les points forts de ces deux projets. Cette démarche a conduit à la réutilisation de la structure mécanique d'OpenBeeLab tout en développant une nouvelle électronique de commande inspirée de l'architecture Mellia.

Les missions réalisées ont porté sur plusieurs aspects du projet : l'étude du fonctionnement du système, le développement logiciel sur microcontrôleur STM32, la mise en œuvre des algorithmes de recherche d'équilibre et de pesée, la conception électronique sous KiCad ainsi que la réalisation de nombreux essais expérimentaux destinés à caractériser le comportement du prototype.

Ce stage a permis de mettre en pratique des compétences acquises au cours de la formation dans les domaines de l'électronique embarquée, de l'automatique, de l'acquisition de données et de la programmation des systèmes temps réel. Il a également permis d'aborder des problématiques concrètes liées à la conception, à l'intégration et à la validation d'un système de mesure complet associant électronique, mécanique et traitement des données.

Afin de présenter l'ensemble des travaux réalisés, ce rapport est structuré en trois chapitres. Le premier chapitre présente le contexte du projet ainsi qu'une étude des solutions existantes ayant servi de base au développement de la balance connectée. Le deuxième chapitre est consacré à la conception et au développement de la solution retenue, en détaillant les choix matériels et logiciels effectués. Enfin, le troisième chapitre présente les essais expérimentaux réalisés et analyse les résultats obtenus afin de valider le fonctionnement et les performances du système développé, avant de conclure sur les perspectives d'évolution du projet.

Chapitre I : Contexte du projet et étude de l'existant

1.1 Contexte et problématique :

Les abeilles jouent un rôle essentiel dans le maintien de la biodiversité et dans la production alimentaire mondiale. Selon l'Organisation des Nations Unies pour l'alimentation et l'agriculture (FAO), près de **75 %** des cultures destinées à l'alimentation humaine dépendent, au moins en partie, de la pollinisation réalisée par les insectes, dont les abeilles constituent les principaux acteurs. Leur activité contribue ainsi à la reproduction de nombreuses espèces végétales et au rendement de nombreuses cultures agricoles.

Cependant, les colonies d'abeilles sont aujourd'hui confrontées à diverses menaces telles que les maladies, les parasites, l'utilisation de pesticides ou encore les changements climatiques. Afin de préserver la santé de leurs ruches et d'optimiser leur production, les apiculteurs doivent effectuer un suivi régulier de plusieurs paramètres, notamment l'évolution du poids des ruches. Cette surveillance nécessite souvent des déplacements fréquents sur les sites d'exploitation, qui peuvent être éloignés de plusieurs kilomètres, entraînant une perte de temps importante ainsi que des coûts supplémentaires.

Face à ces contraintes, le développement de ruches connectées constitue une solution particulièrement intéressante. Grâce à l'utilisation de capteurs et de systèmes embarqués, il devient possible de collecter et de transmettre à distance des informations sur l'état de la ruche sans perturber la colonie. Ces technologies permettent ainsi d'améliorer le suivi des ruches tout en facilitant le travail des apiculteurs.

La problématique de ce projet peut ainsi être formulée de la manière suivante :

Comment concevoir une solution de mesure fiable, précise et robuste pour surveiller une ruche à distance tout en réduisant les déplacements de l'apiculteur ?

1.2 Présentation de la structure d'accueil :

1.2.1 Organisme d'accueil :

Le stage a été réalisé au sein du **FabLab Coh@bit** de l'IUT de Bordeaux, un espace dédié à l'innovation et au prototypage dans les domaines de l'électronique, des systèmes embarqués et de la fabrication numérique. Le FabLab met à disposition des étudiants et des enseignants des ressources techniques permettant de développer et d'expérimenter des projets concrets.

Grâce à la collaboration entre étudiants de différentes spécialités, notamment en électronique et en mécanique, le FabLab favorise l'amélioration continue de projets existants et le développement de nouvelles solutions répondant à des problématiques réelles.

1.2.2 Origine et organisation du projet :

Le projet étudié s'appuie sur deux solutions de pesage de ruche existantes : **OpenBeeLab** et **Mellia**. Chacune de ces plateformes présente des avantages spécifiques, aussi bien sur le plan mécanique que sur le plan électronique.

Dans le cadre des activités du FabLab Coh@bit, plusieurs étudiants ont contribué à l'amélioration de ces systèmes au fil des années. Les étudiants en mécanique ont notamment travaillé sur l'optimisation de la structure de la balance, tandis que les étudiants en électronique ont développé et amélioré les aspects liés à l'acquisition des mesures, au traitement des données et à la communication.

L'analyse des deux plateformes a mis en évidence la complémentarité de leurs architectures. Cette observation a conduit à l'idée de développer une nouvelle solution [1] capable de combiner les principaux avantages d'**OpenBeeLab** et de **Mellia** afin d'obtenir un système de pesage plus performant et mieux adapté aux besoins du projet.

1.2.3 Ressources techniques et documentaires :

Afin de comprendre le fonctionnement des systèmes existants, plusieurs ressources techniques et documentaires ont été consultées au cours du stage. Celles-ci comprenaient notamment le wiki du projet, la documentation technique associée aux plateformes OpenBeeLab et Mellia, les schémas électroniques réalisés sous KiCad, les dépôts Git contenant les codes sources ainsi que les documentations des principaux composants électroniques utilisés.

L'analyse de ces ressources a permis d'acquérir une vision globale des deux systèmes, de comprendre leurs architectures respectives et d'identifier les évolutions nécessaires pour le développement de la solution proposée.

1.2.4 Répartition du travail :

Le projet a été réalisé en binôme avec **Romain Sineus**. Les phases d'étude, de conception et de développement ont été menées conjointement durant la majeure partie du stage.

Romain Sineus ayant terminé son stage trois semaines avant la fin du projet, les essais expérimentaux, l'analyse des résultats et la validation finale de la balance ont été réalisés individuellement.

Le planning prévisionnel du projet est présenté en annexe sous forme de diagramme de Gantt.

1.3 Étude des solutions existantes :

1.3.1 Présentation de la solution Mellia :



Figure 1 : Structure métallique de la balance Mellia

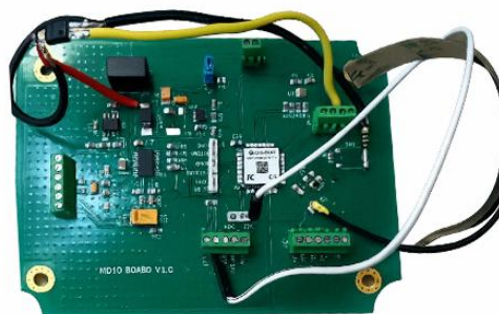


Figure 2 : Carte électronique de la balance Mellia

Mellia[2] est une plateforme de ruche connectée conçue pour assurer la surveillance à distance de colonies d'abeilles. Le système permet de collecter différentes données relatives à l'état de la

ruche, notamment son poids, la température et l'humidité, puis de les transmettre via un réseau Lo-RaWAN.

La fonction de pesage repose sur une **jauge de contrainte** associée à un convertisseur analogique-numérique (ADC) 24 bits, permettant de réaliser une mesure directe du poids avec une grande précision. Les données sont ensuite traitées par un microcontrôleur intégrant les fonctionnalités de communication nécessaires à leur transmission.

Grâce à cette architecture, Mellia offre une solution complète de suivi des ruches, capable de fournir à l'apiculteur des informations en temps réel tout en limitant les déplacements sur le rucher. Son caractère open source facilite également son évolution et son adaptation à de nouveaux besoins.

Architecture matérielle de Mellia

L'architecture matérielle de Mellia repose sur une séparation entre une **partie analogique**, dédiée à l'acquisition des mesures, et une **partie numérique**, chargée du traitement et de la transmission des données. Cette organisation permet de limiter l'influence des perturbations électriques sur les signaux de faible amplitude issus des capteurs. Les échanges entre ces deux parties sont assurés par un **optocoupleur**, garantissant une isolation électrique et contribuant ainsi à la fiabilité des mesures.

Les principaux composants constituant la plateforme Mellia sont présentés ci-dessous.

a. Jauge de contrainte

La mesure du poids est réalisée à l'aide d'une jauge de contrainte. Lorsqu'une charge est appliquée sur la balance, la déformation mécanique de la jauge provoque une variation de sa résistance électrique. Cette variation génère un signal analogique proportionnel à la masse supportée, qui constitue la grandeur d'entrée de la chaîne de mesure.

b. Convertisseur analogique-numérique ADS1232

L'ADS1232 [11] est un convertisseur analogique-numérique (ADC) de 24 bits spécialement conçu pour les applications de pesage. Il convertit les faibles signaux délivrés par la jauge de contrainte en données numériques exploitables par le microcontrôleur. Sa haute résolution permet de détecter de très faibles variations de poids et d'obtenir des mesures précises.

c. Référence de tension REF5050

La référence de tension REF5050 [12] fournit une tension stable utilisée par la chaîne d'acquisition analogique. Elle permet de réduire l'influence des variations d'alimentation sur les mesures et contribue ainsi à améliorer la précision et la stabilité du système de pesage.

d. Capteur environnemental BME680

Le capteur BME680 [13] permet de mesurer plusieurs paramètres environnementaux tels que la température, l'humidité et la pression atmosphérique. Ces données fournissent des informations complémentaires sur l'environnement de la ruche et peuvent être corrélées avec l'activité de la colonie.

e. Gestion de l'alimentation

La carte intègre plusieurs régulateurs de tension assurant l'alimentation des différents blocs électroniques. Le système est conçu pour fonctionner de manière autonome grâce à une batterie associée à un panneau solaire, permettant une utilisation prolongée sur le terrain sans intervention fréquente de l'utilisateur.

f. Module de communication LoRa-E5

La transmission des données est assurée par le module LoRa-E5[9] utilisant la technologie LoRaWAN[10]. Cette solution permet de communiquer sur de longues distances tout en conservant une faible consommation énergétique, ce qui est particulièrement adapté aux applications de surveillance à distance.

g. Circuit de protection et isolation

La carte comporte plusieurs composants destinés à protéger le système contre les perturbations électriques. Une diode empêche notamment les retours de courant vers la source d'alimentation, tandis que l'optocoupleur assure l'isolation entre les parties analogique et numérique. Ces dispositifs contribuent à la robustesse et à la fiabilité globale de la plateforme.

La combinaison de ces différents éléments permet à Mellia de réaliser des mesures précises du poids de la ruche tout en assurant l'acquisition de paramètres environnementaux et la transmission des données vers une plateforme de supervision distante.

Analyse fonctionnelle de la balance Mellia

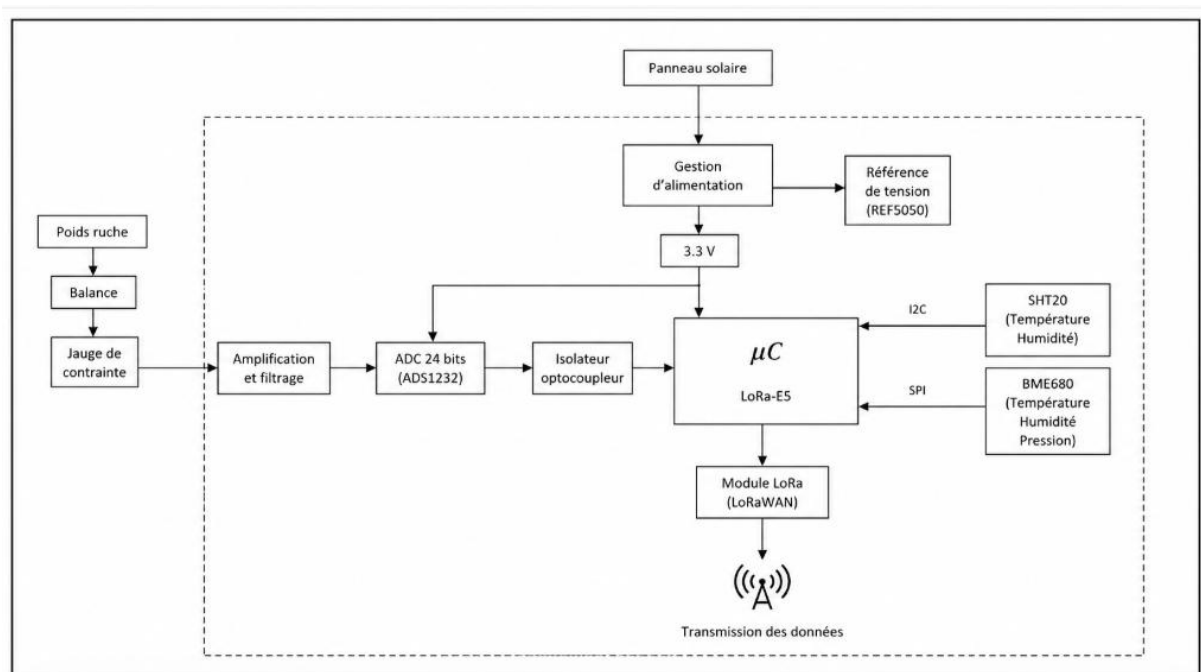


Figure 3 : Schéma fonctionnel de la balance Mellia

Le schéma fonctionnel de Mellia met en évidence les principales étapes de la chaîne de mesure. Le poids de la ruche est converti en signal électrique par la jauge de contrainte, puis numérisé par

l'ADS1232 avant d'être traité par le microcontrôleur LoRa-E5. Les données de pesage et les mesures environnementales sont ensuite transmises à distance via le réseau LoRaWAN.

Avantages et limites de Mellia

Avantages :

- Surveillance à distance de la ruche grâce à la transmission des données via le réseau LoRaWAN.
- Réduction des déplacements inutiles de l'apiculteur.
- Mesure directe du poids de la ruche à l'aide d'une jauge de contrainte.
- Acquisition de plusieurs paramètres environnementaux : température, humidité et pression atmosphérique.
- Alimentation autonome grâce à une batterie associée à un panneau solaire.
- Interface web permettant la consultation des données collectées.
- Solution open source favorisant l'amélioration et l'évolution du système.

Limites :

- Coût relativement élevé, supérieur à 400 € HT par ruche.
- Architecture électronique complexe nécessitant plusieurs composants spécialisés.
- Mise en œuvre et calibration de la jauge de contrainte relativement délicates.
- Sensibilité de la chaîne de mesure aux perturbations électriques.
- Dépendance à une infrastructure LoRaWAN pour l'exploitation des données à distance.

1.3.2 Présentation de la solution OpenBeeLab :



Figure 4 : Structure métallique de la balance OpenBeeLab

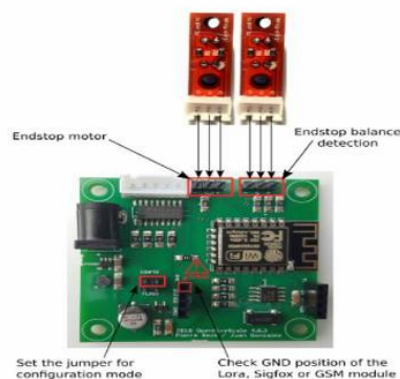


Figure 5 : Carte électronique de la balance OpenBeeLab

OpenBeeLab [3] est un projet open source de balance connectée destiné au suivi du poids des ruches. Contrairement aux solutions basées sur des jauges de contrainte, cette balance repose sur un système mécanique utilisant un contrepoids mobile pour estimer la masse de la ruche.

Le projet a été conçu pour permettre aux apiculteurs de surveiller l'évolution du poids de leurs ruches tout au long de l'année sans avoir à les manipuler. Une attention particulière a également

été portée à la simplicité de fabrication, à la réparabilité du système et à la maîtrise des coûts afin de proposer une solution accessible et durable.

Le cahier des charges prévoit notamment une capacité de mesure pouvant atteindre 150 kg, une précision de l'ordre de 1.2 g ainsi qu'une adaptation à différents types de ruches. La balance doit également résister aux conditions extérieures et être facilement maintenable en cas de panne ou d'usure.

Grâce à son architecture principalement mécanique, OpenBeeLab constitue une solution robuste, économique et réparable, répondant aux besoins essentiels du suivi de poids des ruches.

Architecture matérielle de OpenBeeLab

L'architecture matérielle d'OpenBeeLab repose sur un système de pesage mécanique utilisant le principe du levier et de l'équilibre des moments. Contrairement à Mellia, qui réalise une mesure directe à l'aide d'une jauge de contrainte, OpenBeeLab estime la masse de la ruche en déterminant la position d'un contrepoids mobile nécessaire pour équilibrer la charge appliquée.

Les principaux éléments constituant la balance sont les suivants :

a. Structure mécanique

La structure constitue le support principal de la balance. Elle assure le maintien de la ruche, la transmission des efforts mécaniques et la stabilité de l'ensemble. Sa conception doit garantir une bonne rigidité afin de limiter les déformations susceptibles d'affecter la précision des mesures.

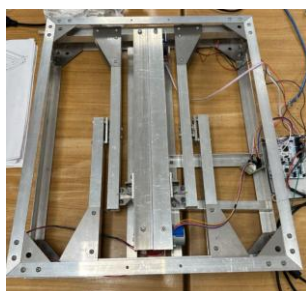


Figure 6 : Structure mécanique de OpenBeeLab

b. Bras de levier et plateau de pesage

Le plateau de pesage supporte directement la ruche. Il est relié à un système de levier permettant de transformer la charge appliquée en un déséquilibre mécanique exploitable par le mécanisme de compensation.



Figure 7 : Bras de levier de la balance OpenBeeLab

c. Contrepoids mobile

Le contrepoids mobile est l'élément central du système de mesure. Son déplacement le long du bras de levier permet de compenser progressivement le moment généré par le poids de la ruche. La position d'équilibre obtenue est directement liée à la masse appliquée sur la balance.

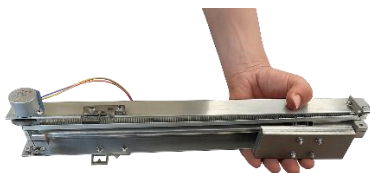


Figure 8 : Contrepoids mobile

d. Moteur pas à pas

Le déplacement du contrepoids est assuré par un moteur pas à pas. Grâce à son positionnement précis et reproductible, il permet d'ajuster finement la position du contrepoids lors de la recherche de l'équilibre. Le moteur [18] utilisé possède une résolution de **2048 pas par tour**, ce qui lui confère une grande précision de positionnement et facilite l'obtention d'une mesure stable et répétable.



Figure 9 : Moteur pas à pas

e. Système de transmission et de guidage

Le mouvement du moteur est transmis au contrepoids par un mécanisme de guidage assurant un déplacement linéaire contrôlé. Ce système doit limiter les frottements et les jeux mécaniques afin de préserver la précision du pesage.

f. Capteur optique

Le capteur optique sert de référence de position. Il permet de détecter une position connue du contrepoids lors de l'initialisation de la balance et garantit ainsi la répétabilité des mesures en recalibrant périodiquement le système.



Figure 10 : Capteur optique

g. Driver moteur ULN2003A

Le moteur pas à pas est piloté par un driver moteur [19], dont le rôle est d'adapter les signaux de commande issus du microcontrôleur à la puissance nécessaire au fonctionnement du moteur. Il permet de contrôler précisément le sens de rotation et le déplacement du contrepoids tout en protégeant l'électronique de commande contre les courants élevés requis par le moteur. Le driver constitue ainsi un élément indispensable pour assurer la recherche automatique de l'équilibre mécanique.

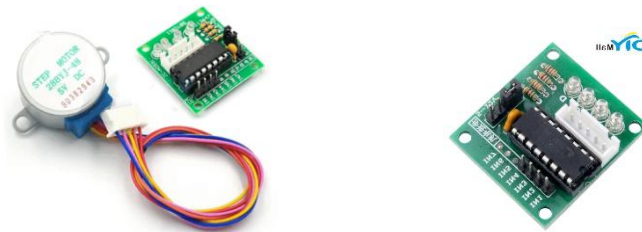


Figure 11 : Driver moteur

h. Carte électronique de commande

La carte électronique assure le pilotage du moteur pas à pas, l'acquisition des informations issues du capteur optique ainsi que l'exécution de l'algorithme de recherche de l'équilibre. Elle détermine la position finale du contrepoids, utilisée pour estimer la masse de la ruche.

Cette architecture repose principalement sur des composants mécaniques simples et facilement remplaçables, ce qui contribue à réduire les coûts de fabrication tout en facilitant la maintenance de la balance.

Analyse fonctionnelle de la balance OpenBeeLab

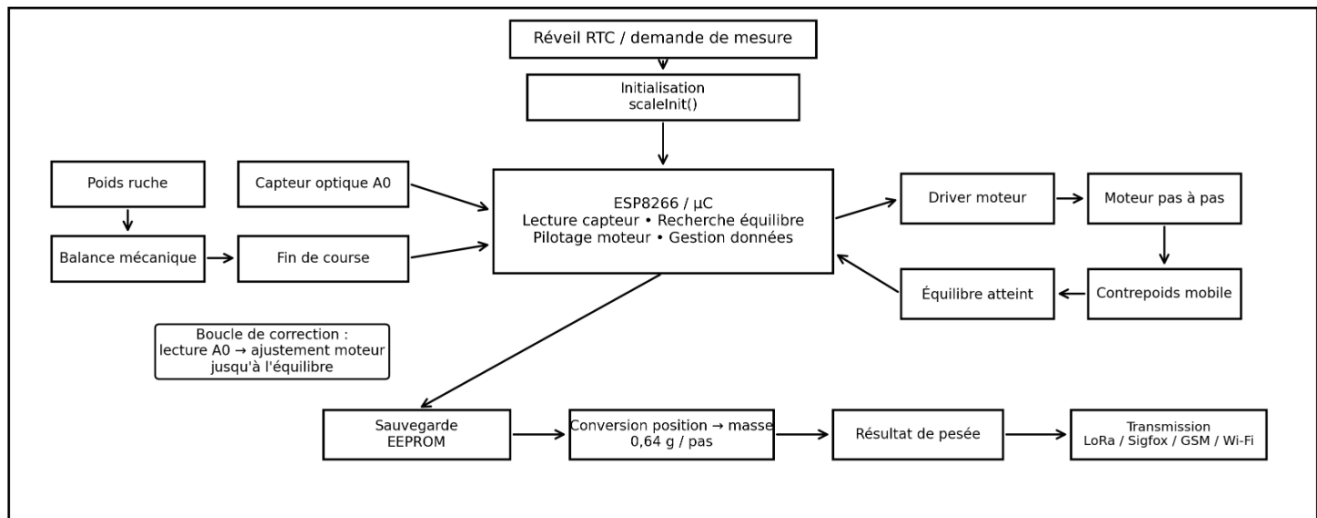


Figure 12 : Schéma fonctionnel de la balance OpenBeeLab

Le système OpenBeeLab [5] réalise la mesure du poids de la ruche en recherchant automatiquement une position d'équilibre mécanique. Ce principe de fonctionnement est mis en œuvre grâce à l'algorithme développé dans le projet **OpenHiveScale**[4], qui assure le pilotage du moteur et la recherche automatique de la position d'équilibre. À chaque cycle de mesure, déclenché par l'horloge temps réel (RTC), le moteur pas à pas déplace progressivement un contrepoids mobile le long du bras de levier. La position du mécanisme est surveillée en continu par un capteur optique qui permet de détecter l'état d'équilibre de la balance. Lorsque l'équilibre est atteint, la position finale du moteur est enregistrée et utilisée pour estimer la masse appliquée sur la balance. Le calcul du poids repose sur la correspondance entre le nombre de pas effectués par le moteur et la masse mesurée, avec un coefficient d'étalonnage de 0,64 g par pas moteur. Les données de pesage peuvent ensuite être transmises à distance via différents moyens de communication. Dans la version OpenHiveScale, la transmission est principalement réalisée à l'aide d'un module **Wi-Fi ESP8266** [8], permettant l'envoi des données vers un serveur distant.

Avantages et limites de la solution OpenBeeLab

Avantages :

- Architecture mécanique simple et robuste.
- Absence de jauge de contrainte et d'électronique de mesure complexe.
- Adaptable à différents types de ruches.
- Système réparable et facilement maintenable.
- Coût de fabrication relativement faible.
- Fonctionnement sans manutention de la ruche.
- Conception open source favorisant les évolutions du système.

Limites :

- Temps de mesure plus long en raison de la recherche d'équilibre.
- Présence de pièces mécaniques soumises à l'usure.
- Sensibilité aux frottements et aux jeux mécaniques.
- Structure plus encombrante qu'une balance à jauge de contrainte.
- Nécessité d'un recalage périodique à l'aide du capteur optique.

- Consommation énergétique liée aux déplacements du moteur pas à pas.

1.4 Comparaison entre Mellia et OpenBeeLab

Critère	Mellia	OpenBeeLab
Principe de mesure	Jauge de contrainte + ADC 24 bits	Équilibre mécanique par contrepoids mobile
Type de mesure	Directe	Indirecte
Précision	Élevée	Bonne
Électronique	Complexe	Simple
Partie mécanique	Simple	Développée
Temps de mesure	Rapide	Plus long
Maintenance	Délicate	Facile
Réparabilité	Moyenne	Élevée
Consommation	Faible	Plus importante
Coût	> 400 € HT	Plus faible
Transmission	LoRaWAN	LoRa / Sigfox / GSM / Wi-Fi
Adaptabilité	Bonne	Très bonne
Open source	Oui	Oui

Figure 13 : Analyse comparative des solutions Mellia et OpenBeeLab

L'étude des deux solutions met en évidence leur complémentarité. Mellia se distingue par la précision de sa chaîne de mesure électronique et son système de communication intégré, tandis qu'OpenBeeLab présente une architecture mécanique robuste, réparable et économique. Cette analyse a conduit à envisager une nouvelle solution combinant les principaux avantages de ces deux plateformes afin de répondre au mieux aux exigences du projet.

- Ce premier chapitre a permis de présenter le contexte du projet ainsi que les solutions existantes étudiées. L'analyse des systèmes Mellia et OpenBeeLab a mis en évidence leurs principales caractéristiques et a permis d'orienter les choix techniques retenus pour le développement de la nouvelle balance connectée.

Chapitre II : Conception et développement de la solution proposée

2.1 Étude de la convergence entre Mellia et OpenBeeLab :

2.1.1 Principe et contexte de la fusion :

Les analyses réalisées sur les solutions Mellia et OpenBeeLab ont montré que ces deux plateformes répondaient au même besoin de suivi des ruches tout en adoptant des approches différentes. Mellia se distingue principalement par son architecture électronique moderne et ses capacités de communication, tandis qu'OpenBeeLab repose sur une structure mécanique robuste et un système de pesage original basé sur la recherche d'équilibre.

Dans le cadre des travaux menés au FabLab Coh@bit, l'idée est apparue de combiner les points forts de ces deux solutions afin de concevoir une nouvelle balance connectée plus performante. Le principe retenu consiste à conserver la structure mécanique développée dans OpenBeeLab tout en remplaçant son électronique par une nouvelle carte inspirée de l'architecture Mellia.

Cette approche permet de capitaliser sur les développements existants tout en proposant une solution plus adaptée aux besoins actuels du projet.

2.1.2 Bénéfices de l'architecture retenue :

L'architecture retenue permet de tirer parti des éléments les plus pertinents de chaque solution tout en limitant leurs contraintes respectives. La conservation de la structure mécanique d'OpenBeeLab garantit un système de pesage robuste, déjà validé expérimentalement et adapté aux conditions d'utilisation d'une ruche.

L'utilisation d'une nouvelle électronique inspirée de Mellia offre quant à elle une meilleure intégration des fonctions de commande, de communication et de gestion énergétique. Cette approche contribue également à simplifier l'architecture du système en réduisant le nombre de cartes et de composants nécessaires.

Enfin, la réutilisation de développements existants permet de limiter le temps de conception et les coûts associés tout en facilitant les évolutions futures de la balance connectée.

2.2 Choix des éléments conservés, supprimés et ajoutés :

2.2.1 Éléments conservés :

Afin de capitaliser sur les travaux déjà réalisés, plusieurs éléments de la balance OpenBeeLab ont été conservés. La structure mécanique a notamment été maintenue en raison de sa robustesse et de son principe de pesage déjà validé. Le moteur pas à pas ainsi que le capteur de fin de course ont également été conservés puisqu'ils assurent respectivement le déplacement du contrepoids et le recalage du système lors des phases de mesure.

L'architecture électronique de Mellia a également été retenue comme base de développement afin de bénéficier d'une solution déjà fonctionnelle et adaptée aux besoins du projet.

2.2.2 Éléments supprimés :

Certains composants présents sur la carte Mellia n'étaient plus nécessaires dans le cadre de la nouvelle architecture. La jauge de contrainte a notamment été supprimée puisque le pesage repose désormais sur le principe mécanique d'OpenBeeLab. Cette suppression permet également de réduire le coût global de la balance, la jauge constituant l'un des composants les plus onéreux du système Mellia.

Le capteur d'humidité [14] a également été retiré afin de simplifier l'électronique et de concentrer le développement sur la fonction principale de pesage.

2.2.3 Éléments ajoutés et modifications apportées :

Le driver moteur a été intégré directement sur la carte électronique afin de réduire le nombre de modules externes, simplifier le câblage et améliorer l'intégration globale du système. Cette modification permet également de faciliter l'assemblage et la maintenance de la balance.

Un connecteur destiné à un bouton poussoir de tare a été ajouté afin de permettre à l'utilisateur d'effectuer facilement une remise à zéro de la mesure sans avoir à intervenir directement sur la carte électronique. Cette fonction améliore l'ergonomie et facilite l'utilisation du système sur le terrain.

Deux connecteurs pour LEDs de signalisation ont également été ajoutés. Ces LEDs, déportées sur le boîtier, permettent de fournir des informations visuelles sur l'état de fonctionnement de la balance, telles que l'alimentation, l'activité du système ou certaines phases de mesure.

Enfin, un connecteur I²C a été intégré afin de rendre la carte plus évolutive. Cette interface permet l'ajout futur de capteurs ou de modules externes compatibles avec ce protocole, sans nécessiter de modification majeure de l'électronique existante.

2.3 Conception du schéma électronique et du PCB :

2.3.1 Élaboration du schéma électronique :

Après avoir défini les éléments conservés, supprimés et ajoutés, la conception de la nouvelle carte électronique a été réalisée sous **KiCad [21]**. L'objectif était d'adapter l'architecture de la carte Mellia afin de l'intégrer au fonctionnement mécanique d'OpenBeeLab.

Le schéma électronique a donc été modifié en conservant les blocs principaux de Mellia, notamment le microcontrôleur, la communication LoRa, la gestion de l'alimentation et les circuits de régulation. En parallèle, les éléments liés à la mesure par jauge de contrainte ont été supprimés, car le pesage est désormais assuré par le système mécanique à contrepoids d'OpenBeeLab.

De nouveaux blocs ont ensuite été ajoutés au schéma, notamment le driver moteur intégré, les connecteurs destinés au moteur pas à pas et au capteur de fin de course, ainsi qu'un connecteur pour le bouton poussoir de tare, deux connecteurs pour les LEDs indicatrices et un connecteur I²C externe. Ces ajouts permettent de relier la carte électronique aux éléments placés dans le boîtier ou sur la structure mécanique de la balance.

Cette étape a permis d'obtenir un schéma électronique cohérent avec la nouvelle architecture proposée, en intégrant sur une même carte les fonctions de commande, de communication et d'interfaçage avec les éléments externes.

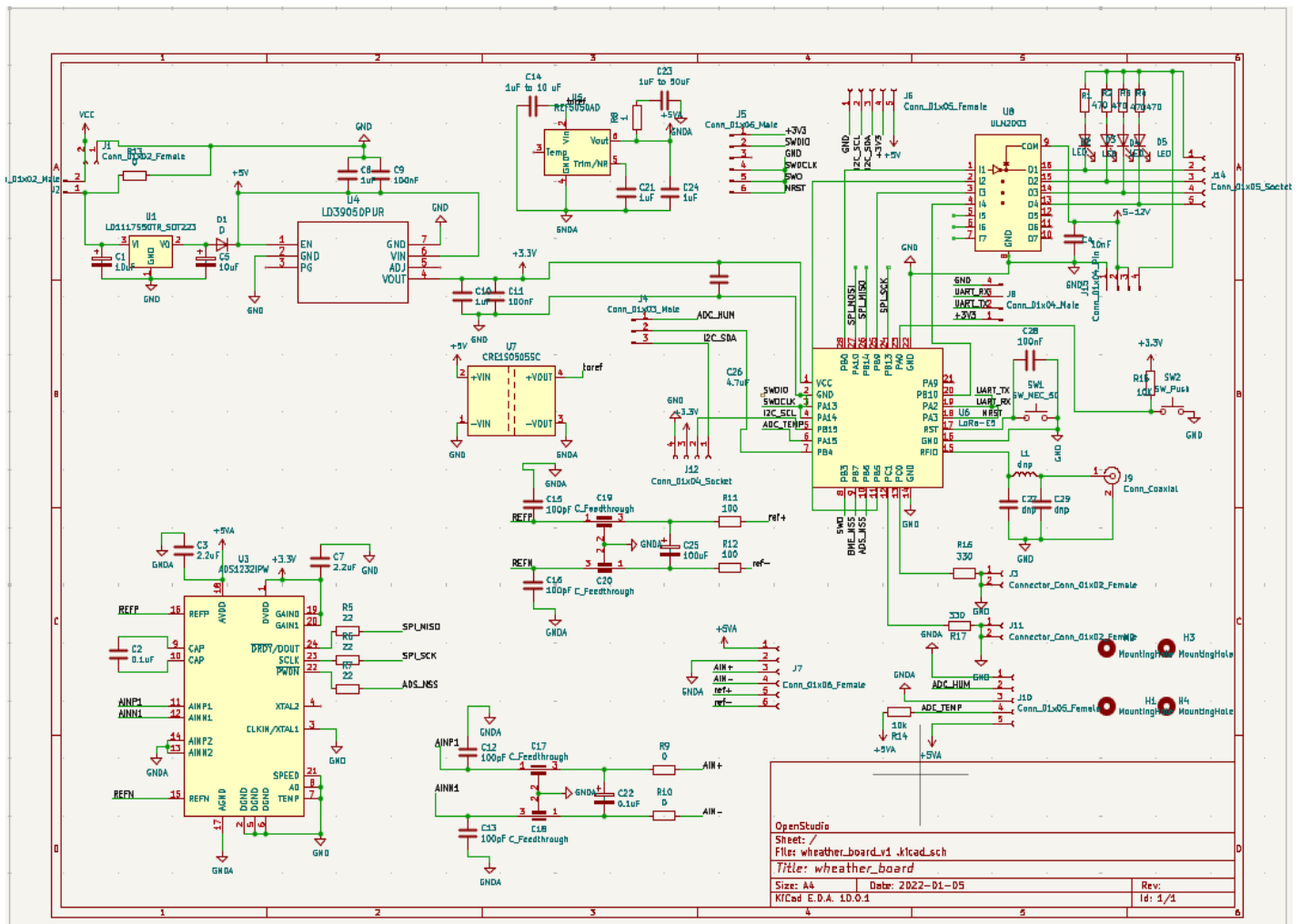


Figure 14 : Schéma électronique

2.3.2 Conception du PCB :

À partir du schéma électronique validé, une première phase de conception du circuit imprimé a été réalisée sous KiCad. Cette étape a consisté à placer les principaux composants de la carte en tenant compte des contraintes fonctionnelles et mécaniques du projet.

Une attention particulière a été portée à l'intégration du driver moteur ainsi qu'au positionnement des différents connecteurs destinés au moteur, au capteur de fin de course, au bouton de tare, aux LEDs et à l'interface I2C [24].

Cependant, en raison de la durée limitée du stage, la conception du PCB n'a pas pu être menée à son terme. Les travaux réalisés ont permis d'établir l'implantation générale des composants et de préparer les étapes de routage, qui pourront être poursuivies lors de futurs développements.

- Ce chapitre a présenté la conception et le développement de la solution proposée. Les choix matériels et logiciels réalisés ont permis de mettre en place une plateforme fonctionnelle destinée à valider les principes de mesure, de recherche d'équilibre et de pesée.

Chapitre III : Tests et validation

3.1 Mise en place de l'environnement de test :

Avant l'intégration du programme sur la carte électronique finale, les développements et les premiers tests ont été réalisés sur une carte de développement **STM32 Nucleo-F411RE** [6]. Cette étape a permis de valider le fonctionnement des différents périphériques, de vérifier les interfaces de communication et de tester les algorithmes de commande dans un environnement plus accessible et facilement débogable.

L'utilisation de cette carte de prototypage a également facilité les phases de mise au point grâce aux outils de débogage intégrés et à la compatibilité avec l'environnement de développement **STM32CubeIDE** [23]. Les différentes fonctionnalités ont ainsi pu être validées progressivement avant leur intégration sur la carte issue de la fusion entre Mellia et OpenBeeLab.

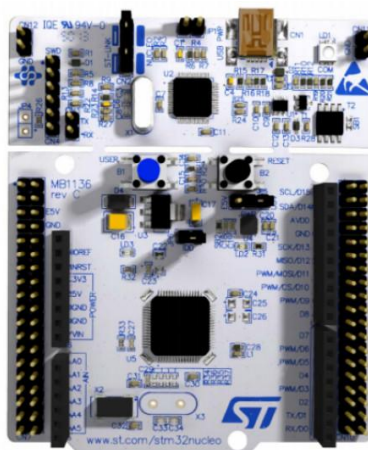


Figure 15 : Carte de test (Nucleo-F411RE)

3.2 Configuration du microcontrôleur STM32 :

La configuration du microcontrôleur a été réalisée sous **STM32CubeMX** avant la génération du projet sous **STM32CubeIDE**. Une configuration de base a été définie afin de disposer des ressources nécessaires aux premiers essais.

L'interface **USART2** a été activée sur les broches **PA2** et **PA3** pour la communication série et l'affichage des informations de débogage. La broche **PA5**, reliée à la LED intégrée de la carte Nucleo, a été configurée en sortie afin de réaliser les premiers tests de commande. Plusieurs broches, notamment **PB4**, **PB5**, **PB6**, **PB10** et **PA10**, ont également été configurées en sorties numériques pour piloter les différents éléments du système. Les broches **PA0** et **PC0** ont été utilisées comme entrées pour les essais d'acquisition et de détection d'événements.

Les interfaces de programmation et de débogage **SWD** ont été conservées via les broches **PA13** et **PA14**, permettant le téléchargement du programme et le suivi de son exécution.

Cette configuration constitue la base de développement utilisée durant l'ensemble des essais. Selon les besoins des différents tests réalisés par la suite, certains paramètres et périphériques ont été adaptés, tout en conservant la même architecture générale de configuration.

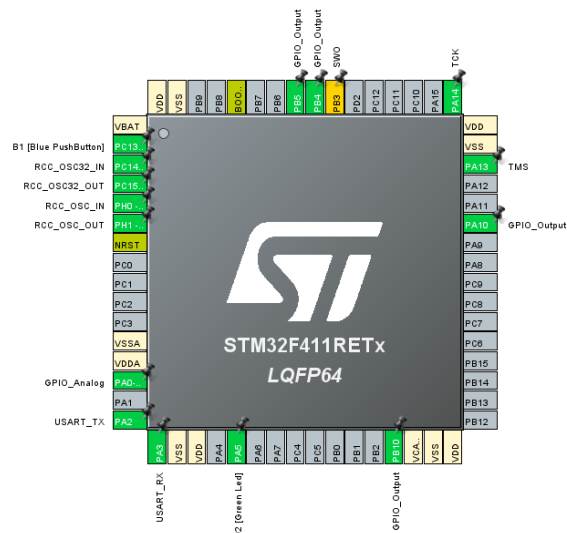


Figure 16 : Configuration de la carte test

3.3 Tests de validation des fonctionnalités :

3.3.1 Test des LEDs de signalisation :

Le premier test réalisé a consisté à faire clignoter la LED intégrée de la carte STM32 Nucleo-F411RE reliée à la broche **PA5**. L'objectif était de vérifier la bonne configuration des GPIO ainsi que le bon fonctionnement du microcontrôleur.

Pour cela, l'horloge du port A a été activée puis la broche PA5 a été configurée en sortie numérique. La LED est ensuite allumée et éteinte périodiquement à l'aide d'une boucle infinie et d'un délai logiciel.

Ce test a permis de valider la configuration des sorties numériques avant le développement des fonctionnalités suivantes.

➤ « Voir Annexe A : Algorithme du test de clignotement des LEDs. »

3.3.2 Test de pilotage du moteur pas à pas :

Après la validation des sorties numériques à l'aide de la LED, un test de pilotage du moteur pas à pas a été réalisé. L'objectif était de vérifier le bon fonctionnement du driver moteur ainsi que la capacité du STM32 à générer la séquence de commande nécessaire à la rotation du moteur.

Le moteur a été commandé en mode demi-pas à travers les quatre sorties reliées au driver. Une séquence de 1024 demi-pas a d'abord été exécutée dans un sens, puis dans le sens inverse après une courte temporisation. La LED intégrée de la carte a été utilisée pour indiquer les phases de rotation.

Ce test a permis de confirmer le bon câblage du système et de valider le déplacement du moteur dans les deux sens avant de poursuivre avec les essais liés au capteur de fin de course et à la recherche d'équilibre.

➤ « Voir Annexe B : Algorithme du pilotage du moteur pas à pas. »

3.3.3 Test des capteurs de début et de fin de course :

Afin de vérifier la répétabilité du déplacement du contrepoids, deux capteurs de fin de course à interrupteur ont été installés aux extrémités de la balance. Un support conçu sous FreeCAD [22] puis fabriqué par découpe laser a permis leur intégration sur la structure mécanique. Le contrepoids actionne directement ces capteurs lorsqu'il atteint les positions extrêmes de déplacement.

L'objectif de ce test était de vérifier que le moteur retrouve la même position après plusieurs cycles aller-retour et de déterminer si l'ajout d'un encodeur était nécessaire pour compenser une éventuelle perte de pas.

Plusieurs déplacements entre le début et la fin de course ont été réalisés, puis la position du moteur a été relevée à chaque retour au point de référence. Les mesures obtenues présentent un écart maximal de 105 demi-pas, soit une erreur relative de 0,28 % et une répétabilité de 99,72 %.

Ces résultats montrent une très bonne répétabilité du système et une perte de pas négligeable. Ils permettent également de valider la plage de déplacement correspondant à une capacité de mesure d'environ 0 à 120 kg. L'ajout d'un encodeur n'a donc pas été jugé nécessaire dans le cadre de ce projet, les performances obtenues étant suffisantes pour poursuivre le développement de la balance. Toutefois, l'intégration d'un encodeur pourrait constituer une perspective d'amélioration intéressante afin d'augmenter la précision du suivi de position et de vérifier l'absence totale de perte de pas. Cette évolution pourra faire l'objet de travaux futurs ou d'un prochain sujet de stage. Les résultats détaillés des essais sont présentés en [Annexe C](#).



Figure 17 : Capteur de fin de course

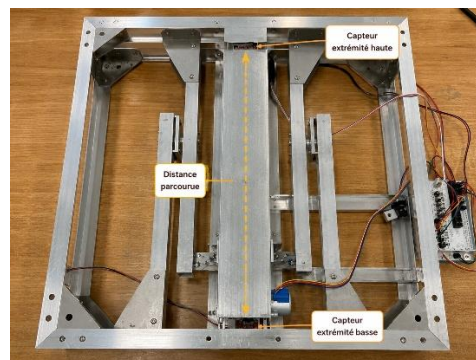


Figure 18 : Course du contrepoids

➤ « Voir Annexe D : Algorithme du test des fins de course. »

3.3.4 Test de lecture du capteur optique par ADC :

Un test de lecture ADC a été réalisé afin de valider le fonctionnement du capteur optique utilisé pour détecter la position du bras de levier. Les valeurs acquises ont été affichées sur le terminal série sous forme de valeurs ADC et de tensions.

Les mesures montrent une variation importante du signal selon la position du bras. Lorsque le faisceau lumineux est totalement coupé, la valeur ADC est proche de **4095** ($\approx 3,3$ V). À l'inverse, lorsque le faisceau est libre, la valeur descend autour de **650** ($\approx 0,5$ V).

Cette plage de mesure importante permet de suivre précisément le passage du bras dans le faisceau optique et constitue la base de l'algorithme de recherche d'équilibre.

➤ « Voir Annexe E : Algorithme du test de lecture d'ADC. »

➤ « Voir Annexe F : Résultats de lecture d'ADC. »

3.3.5 Caractérisation de la résolution du système :

Afin de caractériser la résolution du système, trois essais ont été réalisés à vide, c'est-à-dire sans charge appliquée sur la balance. Les courbes obtenues sont présentées en [Annexe G](#).

L'objectif était de déterminer la variation du signal ADC associée à un déplacement minimal du moteur autour de la position d'équilibre. Les résultats montrent qu'un **demi-pas** moteur correspond à une variation d'environ **100 points ADC**, ce qui permet de distinguer clairement deux positions successives du contrepoids.

Ces essais ont également permis de comparer les performances de différents convertisseurs ADC. La figure 19 présente les résolutions théoriques obtenues avec un ADC 8 bits, l'ADC 12 bits intégré à la carte STM32 Nucleo utilisée pour les essais et l'ADC 24 bits ADS1232 prévu sur la version finale. Dans la zone d'équilibre retenue pour la pesée, comprise entre **1800 et 3000 points ADC**, l'ADC 12 bits offre déjà une résolution théorique d'environ **3,2 mg**.

Les résultats montrent qu'un ADC 24 bits apporterait une précision largement supérieure aux besoins réels de l'application. Compte tenu des performances obtenues et des limitations mécaniques du système, l'utilisation d'un ADC 12 bits apparaît suffisante pour atteindre la précision recherchée sur une plage de mesure comprise entre **0 et 120 kg**.

Résolution ADC	Nb niveaux ADC	Points ADC / demi-pas	Précision par pas (g)	Prix typique
8 bits	256	25	0,0128 g	5 €
12 bits	4096	100	0,0032 g	2 €
24 bits (ADS1232)	16 777 216	419 430,4	$7,63 \times 10^{-7}$ g	10 €

Figure 19 : Comparaison des convertisseurs ADC utilisés et envisagés

➤ « Voir Annexe H : Algorithme du test de résolution autour de l'équilibre. »

3.3.6 Détermination du temps de stabilisation :

Un test de stabilisation a été réalisé avec un déplacement de **10 demi-pas moteur** afin de déterminer le délai optimal entre deux pas. Cinq essais ont été effectués avec des délais de **2 ms, 5 ms, 10 ms, 15 ms et 25 ms**. Les courbes obtenues sont présentées en annexe.

Les résultats montrent que le meilleur compromis entre rapidité et stabilité est obtenu avec un délai de **5 ms** entre deux demi-pas. Dans cette configuration, le système se stabilise en moyenne après **134,1 ms**, ce qui a conduit à retenir ce paramètre pour l'algorithme de recherche d'équilibre.

➤ « Voir Annexe I : Algorithme du test de stabilisation. »

➤ « Voir Annexe J : Résultats du test de stabilisation. »

3.3.7 Test de la recherche d'équilibre :

Le principe de la recherche d'équilibre repose sur le déplacement automatique du contrepoids à l'aide du moteur pas à pas en fonction de la valeur mesurée par le capteur optique. Lorsque le bras de levier est éloigné de sa position d'équilibre, le moteur se déplace rapidement afin de réduire le temps de recherche.

Une zone d'équilibre a été définie entre **1800 et 3000 points ADC**, avec une valeur centrale située autour de **2400 points ADC**. Lorsque le système dépasse cette zone, le sens de déplacement du moteur est inversé. Des mouvements successifs de va-et-vient sont alors effectués autour de la position d'équilibre.

Afin d'améliorer la stabilité de la mesure et de limiter l'influence des jeux mécaniques, le moteur est progressivement ralenti à chaque changement de direction. Pour cela, un délai supplémentaire de **200 ms** est ajouté entre deux déplacements à chaque retour. L'amplitude des oscillations diminue ainsi progressivement jusqu'à ce que la valeur ADC se stabilise dans la zone d'équilibre.

Lorsque cette condition est atteinte, le moteur est arrêté et la position finale du contrepoids est enregistrée. Cette position est ensuite utilisée pour déterminer la masse appliquée sur la balance.

- « Voir Annexe K : Algorithme de la recherche d'équilibre. »
- « Voir Annexe L : Résultats de la recherche d'équilibre. »

3.3.8 Test de la pesée :

3.3.8.1 Pesée basée sur la différence de position

Le premier algorithme de pesée repose sur une méthode classique utilisant deux recherches d'équilibre successives. Une première recherche est réalisée à vide afin de déterminer la position de référence du système, appelée **tare**. Cette position correspond au zéro de mesure.

Une seconde recherche d'équilibre est ensuite effectuée après l'ajout d'une charge sur la balance. Le poids est calculé à partir de la différence entre la position obtenue sous charge et la position de tare. La conversion en masse est réalisée à l'aide du coefficient d'étalonnage déterminé expérimentalement, soit **0,32 g par demi-pas moteur**.

3.3.8.2 Pesée avec correction de la position dans la zone d'équilibre

Le second algorithme reprend le même principe mais ajoute une correction liée à la position exacte du bras dans la zone d'équilibre. La plage d'équilibre retenue pour la pesée est comprise entre **1800 et 3000 points ADC**, avec une valeur centrale fixée à **2400 points ADC**.

Lorsque le système s'arrête dans cette zone, l'écart entre la valeur ADC mesurée et la valeur centrale est calculé. Cette correction est prise en compte lors de la tare puis lors de la mesure sous charge afin d'affiner le calcul de la masse.

Cette approche permet d'exploiter l'information disponible à l'intérieur de la zone d'équilibre et d'améliorer la précision de la pesée. Les essais réalisés ont montré qu'environ **100 points ADC** correspondent à **un demi-pas moteur**, ce qui permet d'estimer plus finement la masse réelle mesurée.

- « Voir Annexe M : Algorithme de la pesée. »

Les essais de pesée avec des masses étalons n'ont pas pu être réalisés durant la période du stage en raison du temps disponible limité. Une partie importante du temps de développement a en effet été consacrée à l'identification et à la correction de plusieurs problèmes mécaniques affectant le déplacement du contrepoids et la stabilité du système. Ces ajustements étaient nécessaires pour garantir un fonctionnement fiable de la balance avant d'envisager les essais de pesée.

Les différents tests menés ont néanmoins permis de valider le fonctionnement de la recherche d'équilibre, la répétabilité du système, la résolution de la mesure ainsi que la méthode de calcul du poids. Les algorithmes de pesée développés constituent ainsi une base fonctionnelle qui pourra être complétée et validée expérimentalement lors de futurs travaux.

3.4 Bilan du stage :

Ce stage m'a permis de participer à un projet concret de développement d'une balance connectée destinée au suivi du poids des ruches. la majorité des objectifs du cahier des charges a été atteinte et plusieurs travaux complémentaires ont également été réalisés au cours du projet.

Sur le plan technique, j'ai acquis de nouvelles compétences en programmation sur microcontrôleur STM32, notamment pour la gestion des entrées/sorties, de l'ADC, du moteur pas à pas et du traitement des données issues des capteurs. J'ai également approfondi mes connaissances en conception électronique grâce à l'utilisation de KiCad pour la réalisation du schéma électronique et les premières étapes de conception du PCB.

Ce stage m'a aussi permis de découvrir et d'utiliser différents outils de fabrication numérique. J'ai notamment conçu plusieurs pièces mécaniques sous FreeCAD pour l'impression 3D et la découpe laser, puis participé à leur intégration sur le prototype. Ces travaux m'ont permis de mieux comprendre les interactions entre les aspects mécaniques, électroniques et logiciels d'un système embarqué complet.

Au cours du projet, plusieurs difficultés mécaniques ont été identifiées et corrigées afin d'améliorer le fonctionnement de la balance. Une part importante du travail a également consisté à réaliser de nombreux essais expérimentaux afin de déterminer plusieurs paramètres et constantes qui n'avaient pas été définis dans les versions précédentes du projet. Ces tests ont notamment permis de caractériser la répétabilité du système, de définir la zone d'équilibre, d'évaluer la résolution de la mesure et d'optimiser les paramètres des algorithmes de recherche d'équilibre et de pesée.

Enfin, ce stage m'a permis de développer mon autonomie, ma rigueur expérimentale et ma capacité à analyser et résoudre des problèmes techniques. Cette expérience a constitué une première immersion concrète dans un projet d'ingénierie, tout en renforçant mes compétences en électronique, en programmation embarquée et en conception mécanique.

- Les essais réalisés ont permis de caractériser le comportement du système et de valider les principales fonctionnalités développées. Les résultats obtenus constituent une base solide pour la poursuite du développement et l'amélioration des performances de la balance connectée.

Conclusion et perspectives

Ce stage a permis de concevoir et de développer un prototype de balance motorisée reposant sur un système de recherche d'équilibre commandé par moteur pas à pas. Les différents travaux réalisés ont conduit à la validation de l'architecture mécanique retenue, du pilotage du moteur, de l'acquisition des mesures ainsi que des algorithmes de recherche d'équilibre et de calcul du poids.

Les essais effectués ont également permis de déterminer plusieurs paramètres et constantes nécessaires au bon fonctionnement du système, qui n'étaient pas connus au début du projet. Ces travaux expérimentaux ont joué un rôle essentiel dans la compréhension du comportement du système et dans l'ajustement des algorithmes développés. Malgré certaines difficultés rencontrées lors de la mise au point mécanique du prototype, les objectifs principaux du cahier des charges ont été atteints et plusieurs améliorations ont même été apportées au cours du projet.

Les essais complets de pesée n'ont pas pu être réalisés durant la période du stage en raison du temps consacré à la résolution des problèmes mécaniques et à la caractérisation du système. De même, la conception du PCB a été initiée mais n'a pas pu être finalisée avant la fin du stage. Néanmoins, l'ensemble des développements réalisés constitue une base solide pour la poursuite du projet.

Pour la suite, plusieurs améliorations pourront être envisagées. L'ajout d'un encodeur de position permettrait de connaître la position réelle du moteur et d'améliorer encore la précision des mesures. Les algorithmes de pesée pourront également être affinés afin d'exploiter plus précisément les variations du signal ADC autour du point d'équilibre. Par ailleurs, la carte électronique développée intègre déjà un bouton de tare. Une évolution naturelle du système consistera à utiliser directement ce bouton pour effectuer la mise à zéro de la balance, sans lancer automatiquement une recherche d'équilibre à vide au démarrage. Enfin, la finalisation du PCB conçu sous KiCad, suivie de sa fabrication et de sa validation, constituera une étape importante pour poursuivre l'intégration du système et aboutir à une version plus compacte et mieux adaptée à une utilisation sur le terrain.

Bibliographie

Documentation des projets

- [1] FabLab Coh@bit, *Dépôt Git du projet RucheConnectée*.
https://git.cohabit.fr/Ruches_Connectees/Hive_Melia_OpenBeeLab
- [2] Nicolas Breard, Mellia – Dépôt GitHub du projet Mellia.
<https://github.com/nbreard/mellia>
- [3] *Dépôt Git du projet OpenBeeLab*.
<https://gitlab.com/openbeelab>
- [4] OpenHiveScale, *Code source et documentation du système de pesée OpenHiveScale*.
<https://github.com/openhivescale>
- [5] FabLab Coh@bit, *Wiki OpenBeeLab*.
<https://projets.cohabit.fr/redmine/projects/openbeelab/wiki/Wiki>

Microcontrôleurs et modules de communication

- [6] STMicroelectronics, *STM32F411RE Datasheet et Documentation Technique*.
<https://www.st.com/en/evaluation-tools/nucleo-f411re.html>
- [7] STMicroelectronics, *Carte Nucleo-F411RE User Manual*.
<https://www.st.com/en/evaluation-tools/nucleo-f411re.html>
- [8] Espressif Systems, *ESP8266EX Datasheet*.
<https://www.espressif.com/en/products/socs/esp8266ex/resources>
- [9] Seeed Studio, *LoRa-E5 (STM32WLE5JC) Documentation*.
https://wiki.seeedstudio.com/LoRa-E5_STM32WLE5JC_Module
- [10] LoRa Alliance, *LoRaWAN Specification*.
https://lora-alliance.org/resource_hub/lorawan-specification-v1-0-3/

Capteurs et acquisition de données

- [11] Texas Instruments, *ADS1232 – 24-Bit Analog-to-Digital Converter for Weigh Scale Applications* :
<https://www.ti.com/lit/ds/symlink/ads1232.pdf>
- [12] Texas Instruments, *REF5050 – Precision Voltage Reference Datasheet* :
<https://www.ti.com/lit/ds/symlink/ref5050.pdf>
- [13] Bosch Sensortec, *BME680 Environmental Sensor Datasheet* :
<https://www.bosch-sensortec.com/products/environmental-sensors/gas-sensors/bme680/>

[14] Sensirion, *SHT20 Humidity and Temperature Sensor Datasheet* :
<https://sensirion.com/products/catalog/SHT20>

[15] Vishay, *TCST2103 Optical Reflective Sensor Datasheet* :
<https://www.vishay.com/docs/83760/tcst2103.pdf>

Alimentation et régulation

[16] STMicroelectronics, *LD39050 Ultra-Low Noise Voltage Regulator Datasheet* :
<https://www.st.com/en/power-management/ld39050.html>

[17] STMicroelectronics, *LD1117 Voltage Regulator Datasheet* :
<https://www.st.com/resource/en/datasheet/ld1117.pdf>

Motorisation et électronique de puissance

[18] 28BYJ-48, *Stepper Motor Datasheet* :
<https://www.makerguides.com/wp-content/uploads/2019/04/28byj48-Stepper-Motor-Datasheet.pdf>

[19] Texas Instruments, *ULN2003A Darlington Transistor Array Datasheet* :
<https://www.ti.com/lit/ds/symlink/uln2003a.pdf>

[20] Toshiba, *TBD62003AFG Driver Array Datasheet* :
<https://toshiba.semicon-storage.com/info/docget.jsp?did=30523>

Outils logiciels et conception

[21] KiCad Development Team, *KiCad EDA Documentation* :
<https://docs.kicad.org>

[22] FreeCAD Community, *FreeCAD Documentation* :
<https://wiki.freecad.org>

[23] STMicroelectronics, *STM32CubeIDE User Guide* :
<https://www.st.com/en/development-tools/stm32cubeide.html>

Protocoles et interfaces

[24] NXP Semiconductors, *I²C-bus Specification and User Manual* :
<https://www.nxp.com/docs/en/user-guide/UM10204.pdf>

[25] SparkFun Electronics, *Serial Peripheral Interface (SPI) Documentation* :
<https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all>

[26] Analog Devices, *UART Communication Protocol Overview* :
<https://www.analog.com/en/analog-dialogue/articles/uart-a-hardware-communication-protocol.html>

Annexes

Annexe A – Algorithme du test de clignotement des LEDs

```

Début
|
|   Activer l'horloge du port GPIOA
|   Configurer la broche PA5 en sortie numérique
|   Répéter indéfiniment
|   |   Mettre la broche PA5 à l'état haut
|   |   Allumer la LED
|   |   Attendre un délai
|   |   Mettre la broche PA5 à l'état bas
|   |   Éteindre la LED
|   |   Attendre un délai
|   Fin Répéter
|
Fin
  
```

Annexe B – Algorithme du pilotage du moteur pas à pas

```

Début
|
|   Initialiser l'horloge du système
|   Configurer les broches du moteur en sortie
|   |   PA10 → IN1
|   |   PB4  → IN2
|   |   PB5  → IN3
|   |   PB10 → IN4
|
|   Initialiser la séquence de demi-pas
|   Arrêter le moteur
|
|   Répéter indéfiniment
|   |   Allumer la LED de test
|   |
|   |   Faire avancer le moteur de 1024 demi-pas
|   |   |   Appliquer la séquence de demi-pas
|   |   |   Attendre 3 ms
|   |
|   |   Éteindre la LED
|   |   Attendre 1 seconde
|   |
|   |   Allumer la LED de test
|   |
|   |   Faire reculer le moteur de 1024 demi-pas
|   |   |   Appliquer la séquence inverse
|   |   |   Attendre 3 ms
|   |
|   |   Éteindre la LED
  
```

- └─ Attendre 1 seconde
- └─ Fin Répéter

Annexe C – Résultats des essais de la détection de la course

Mesure	Nombre de demi-pas
1	37764
2	37724
3	37731
4	37741
5	37751
6	37732
7	37739
8	37761
9	37783
10	37776
11	37795
12	37804
13	37811
14	37810
15	37829

Figure 20 : Répétabilité de la mesure de la course moteur

Annexe D – Algorithme du test des fins de course

```

Début
├─ Initialiser le microcontrôleur
├─ Configurer l'horloge du système
├─ Configurer la liaison UART
├─ Configurer les broches du moteur en sortie
│   ├── PA10 → IN1
│   ├── PB4  → IN2
│   ├── PB5  → IN3
│   └── PB10 → IN4
├─ Configurer les capteurs de fin de course en entrée
│   ├── PA0 → capteur de début de course
│   └── PA1 → capteur de fin de course
├─ Arrêter le moteur
├─ Afficher "Recherche début de course A0"
└─ Tant que le capteur A0 n'est pas activé

```

```

├── Reculer le moteur d'un demi-pas
├── Répéter la lecture du capteur A0
├── Arrêter le moteur
├── Attendre 500 ms
├── Afficher "Début de course détecté"
├── Initialiser le compteur de demi-pas à 0
├── Afficher "Recherche fin de course A1"
├── Tant que le capteur A1 n'est pas activé
│   ├── Avancer le moteur d'un demi-pas
│   ├── Incrémenter le compteur de demi-pas
│   └── Répéter la lecture du capteur A1
├── Arrêter le moteur
├── Afficher "Fin de course détectée"
├── Afficher le nombre total de demi-pas parcourus
├── Maintenir le moteur à l'arrêt
└── Fin

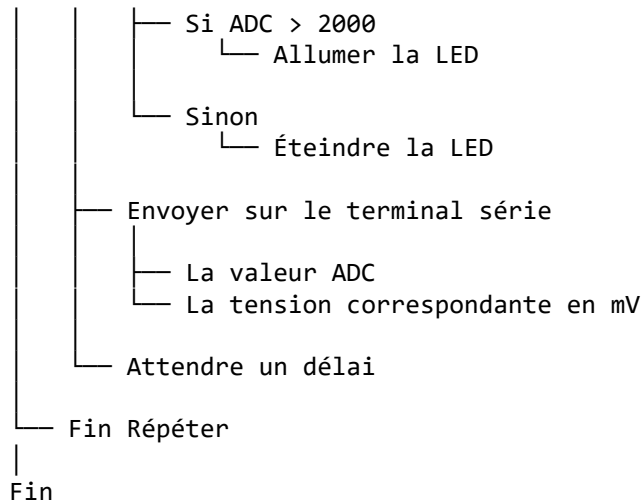
```

Annexe E – Algorithme du test de lecture d'ADC

```

Début
├── Initialiser les GPIO
│   ├── Configurer PA5 en sortie numérique
│   │   └── LED intégrée de la carte
│   └── Configurer PA0 en mode analogique
│       └── Entrée ADC du capteur
├── Initialiser l'ADC
│   ├── Activer l'horloge de l'ADC1
│   ├── Sélectionner le canal 0 correspondant à PA0
│   ├── Régler le temps d'échantillonnage
│   └── Activer l'ADC1
├── Initialiser l'UART
│   ├── Configurer PA2 en transmission UART
│   ├── Régler la vitesse à 9600 bauds
│   └── Activer USART2
├── Afficher "Démarrage lecture ADC"
├── Répéter indéfiniment
│   ├── Lire la valeur ADC sur PA0
│   ├── Convertir la valeur ADC en tension
│   │   └── Tension = ADC × 3300 / 4095
│   └── Comparer la valeur ADC au seuil défini

```

Annexe F– Résultats de lecture d'ADC

ADC = 4067	Tension = 3277 mV
ADC = 4067	Tension = 3277 mV
ADC = 4067	Tension = 3277 mV
ADC = 4067	Tension = 3277 mV
ADC = 4067	Tension = 3277 mV
ADC = 4067	Tension = 3277 mV
ADC = 4063	Tension = 3274 mV
ADC = 4067	Tension = 3277 mV
ADC = 4063	Tension = 3274 mV
ADC = 4069	Tension = 3279 mV
ADC = 4063	Tension = 3274 mV
ADC = 4067	Tension = 3277 mV
ADC = 4071	Tension = 3280 mV
ADC = 4071	Tension = 3280 mV
ADC = 4075	Tension = 3283 mV
ADC = 4075	Tension = 3283 mV
ADC = 586	Tension = 472 mV
ADC = 599	Tension = 482 mV
ADC = 599	Tension = 482 mV
ADC = 591	Tension = 476 mV
ADC = 591	Tension = 476 mV
ADC = 586	Tension = 472 mV
ADC = 680	Tension = 547 mV
ADC = 586	Tension = 472 mV

Figure 21 : Acquisition des valeurs ADC lors du test du capteur optique

Annexe G – Résultats expérimentaux de la caractérisation de la résolution du système

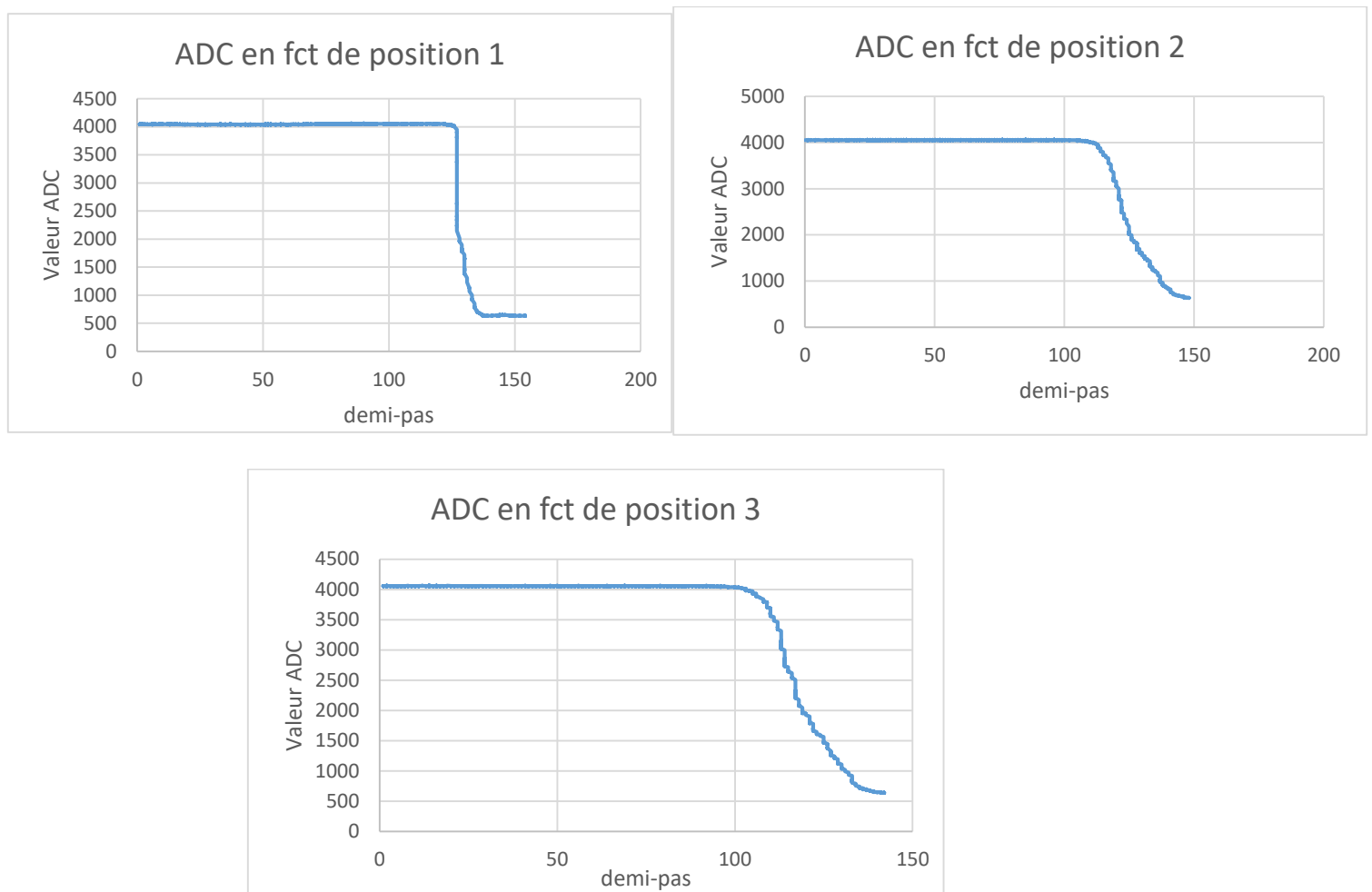


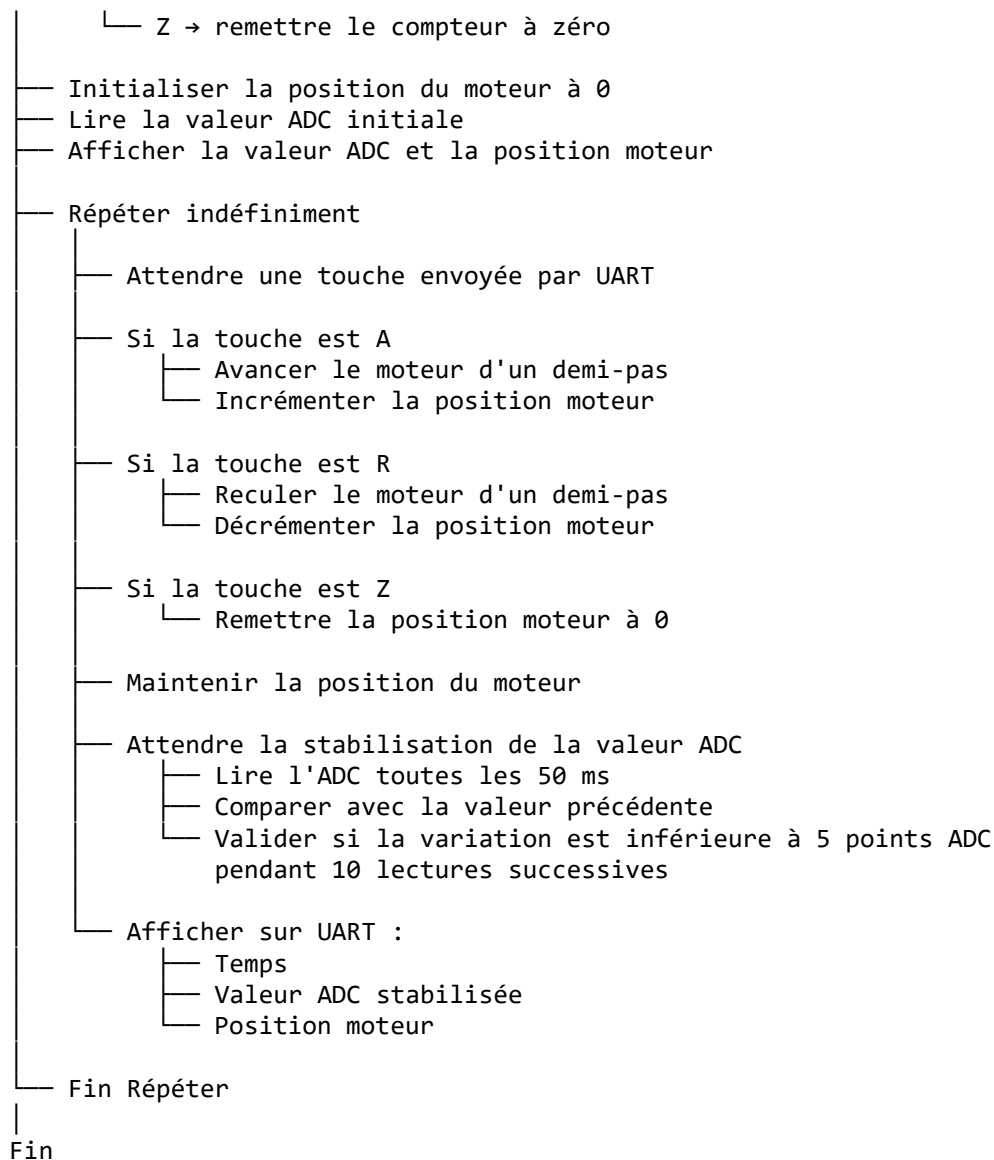
Figure 22 : Résultats expérimentaux de la caractérisation de la résolution du système

Annexe H– Algorithme du test de résolution autour de l'équilibre

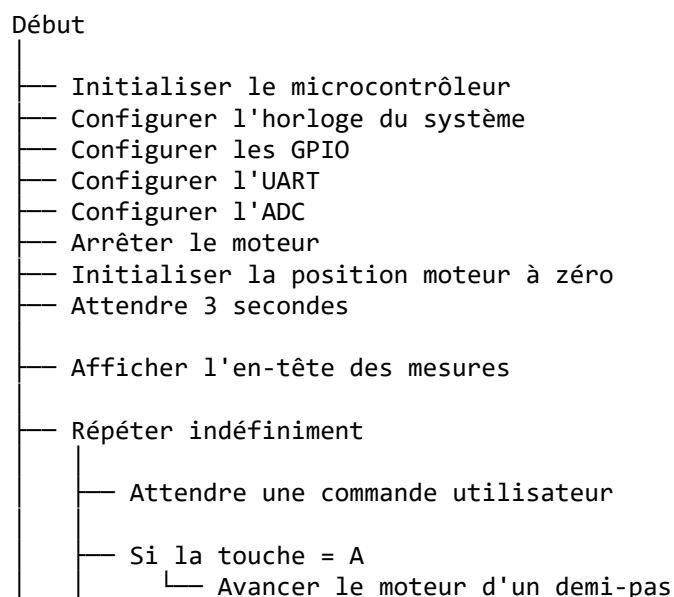
```

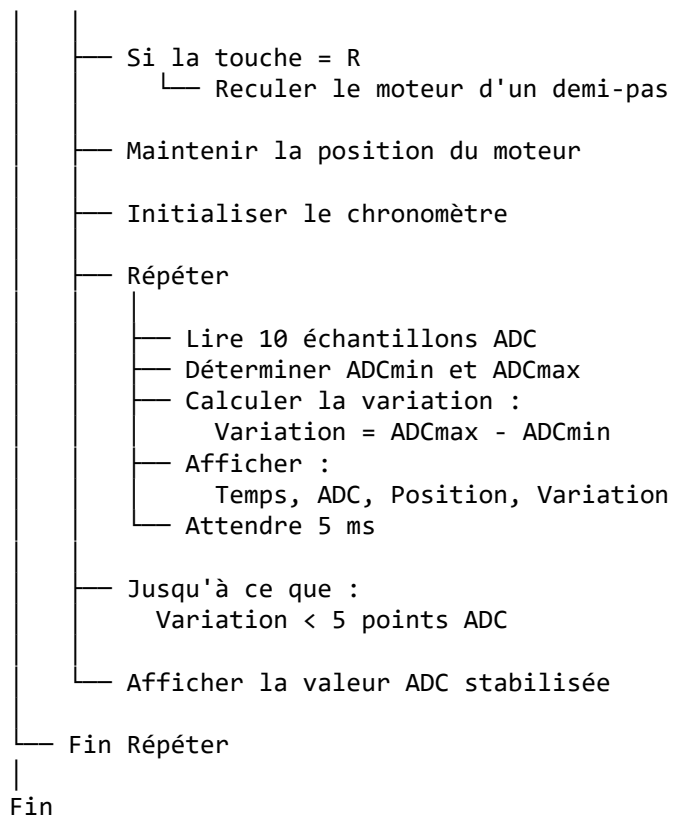
Début
├── Initialiser le microcontrôleur
├── Configurer l'horloge du système
├── Configurer la liaison UART à 115200 bauds
├── Configurer les broches du moteur en sortie
│   ├── PA10 → IN1
│   ├── PB4  → IN2
│   ├── PB5  → IN3
│   └── PB10 → IN4
├── Configurer PA0 en entrée analogique
│   └── Lecture du capteur optique par ADC
├── Initialiser l'ADC
├── Arrêter le moteur
├── Attendre 3 secondes
├── Afficher le mode manuel sur le terminal série
│   ├── A → avancer d'un demi-pas
│   └── R → reculer d'un demi-pas

```



Annexe I– Algorithme du test de stabilisation





Annexe J– Résultats du test de stabilisation

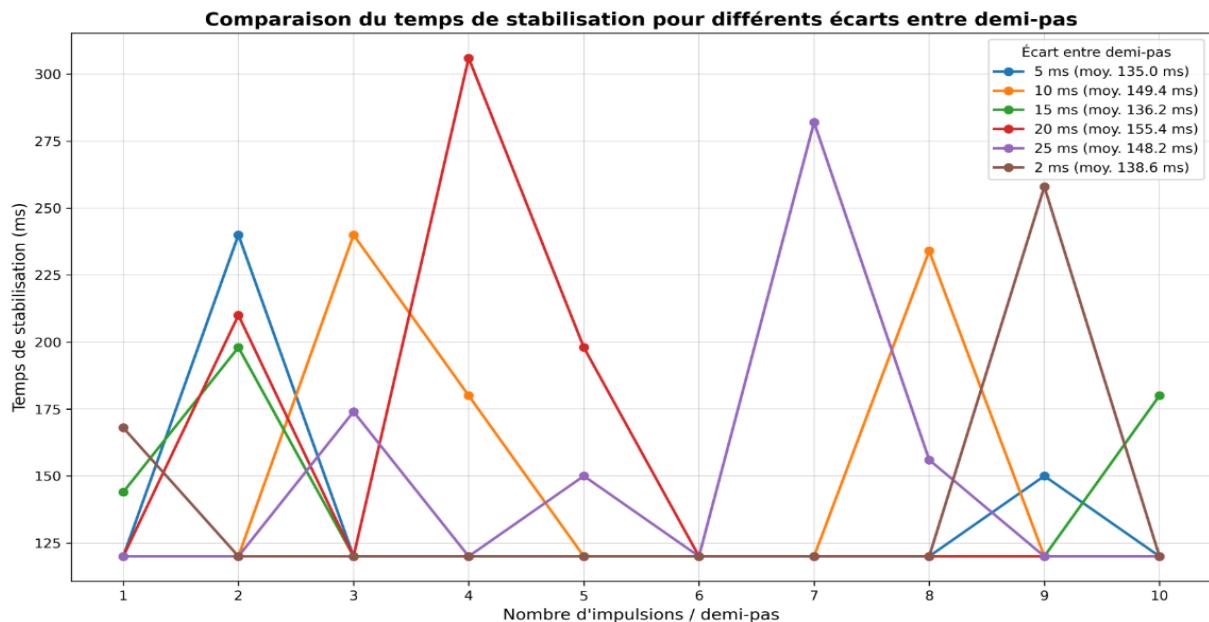
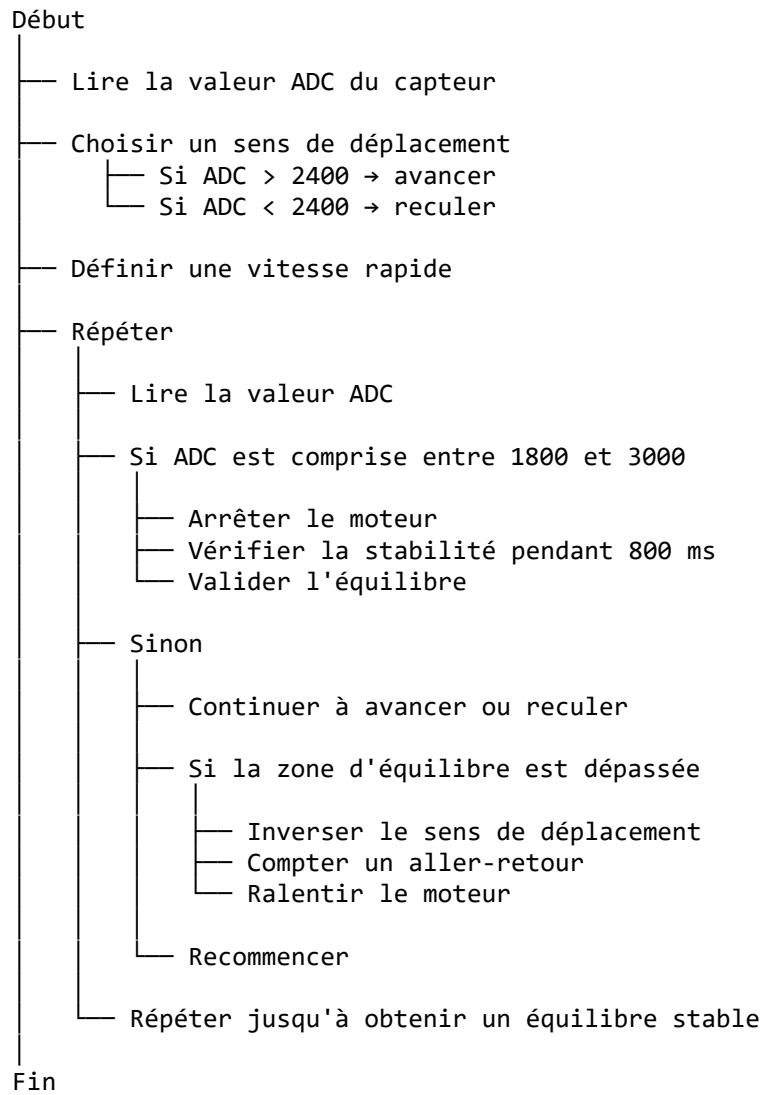


Figure 23 : Comparaison du temps de stabilisation pour différents écarts entre demi-pas

Annexe K– Algorithme de la recherche d'équilibre



Annexe L – Résultats de la recherche d'équilibre

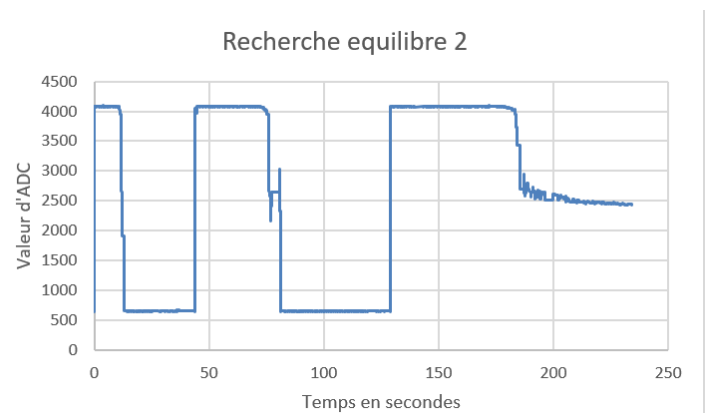
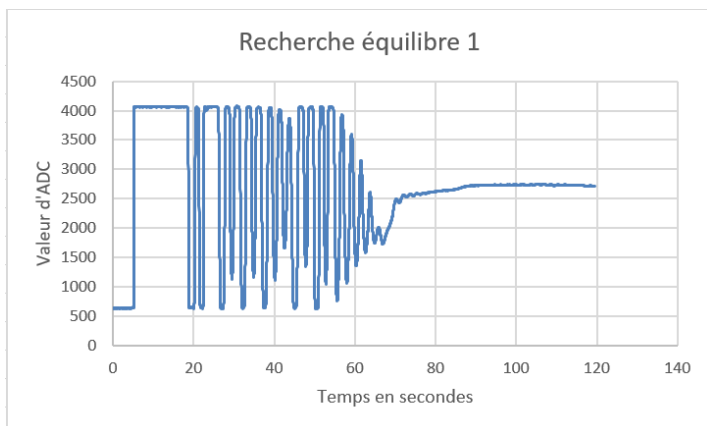


Figure 24 : Résultats de la recherche d'équilibre

Annexe M – Algorithmes de la pesée

Algorithme 1 : Pesée par différence de positions d'équilibre

```

Début
├── Effectuer une première recherche d'équilibre à vide
├── Lorsque l'équilibre est trouvé
│   ├── Enregistrer la position moteur
│   │   └── comme position de tare
├── Déposer la charge à mesurer
├── Effectuer une seconde recherche d'équilibre
├── Lorsque l'équilibre est trouvé
│   └── Enregistrer la nouvelle position moteur
├── Calculer la différence de position
│   └──  $\Delta\text{Pas} = \text{PositionPoids} - \text{PositionTare}$ 
├── Convertir les demi-pas en grammes
│   └──  $\text{Poids} = \Delta\text{Pas} \times 0,32 \text{ g}$ 
├── Afficher le poids mesuré
└── Fin
  
```

Algorithme 2 : Pesée avec correction ADC

```

Début
├── Effectuer une recherche d'équilibre à vide
├── Enregistrer
│   ├── la position de tare
│   └── la valeur ADC de fin d'équilibre
├── Déposer la charge à mesurer
├── Effectuer une nouvelle recherche d'équilibre
├── Enregistrer
│   ├── la position avec charge
│   └── la valeur ADC finale
├── Calculer la différence de position
├── Calculer l'écart ADC par rapport
│   └── au centre de la zone d'équilibre
│       └── Centre ADC = 2400
├── Corriger la mesure à partir
│   └── de cet écart ADC
└── Convertir la position corrigée
  
```

