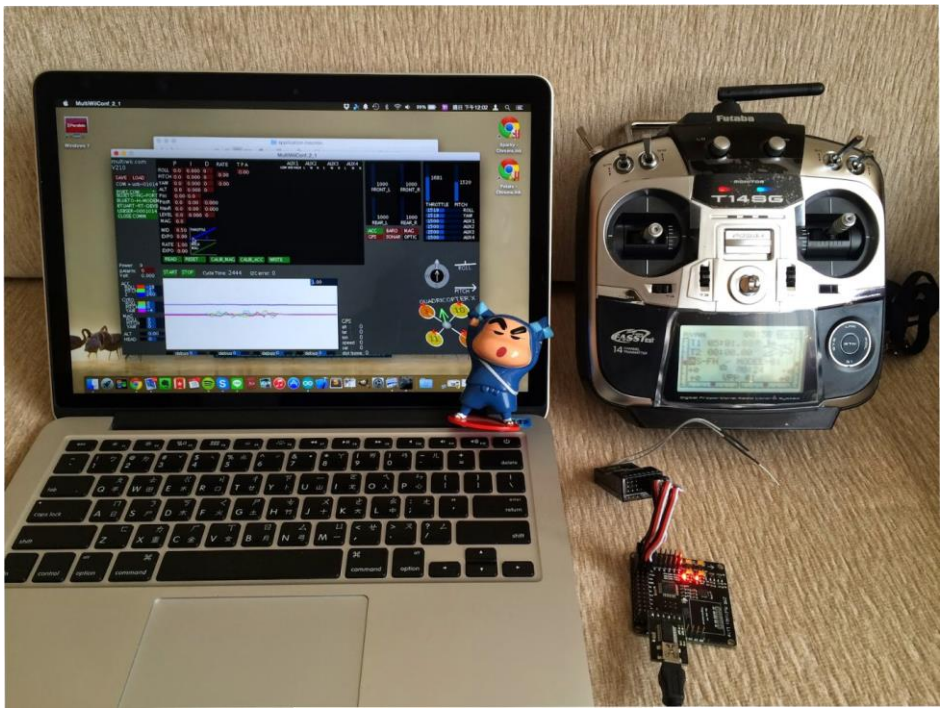
UNIVERSITÉ TOULOUSE III PAUL SABATIER — Université de Toulouse

PROJET BE
SIMULATION DRONE



Réalisé par :

ELALAOUI Abdelhak

ELMAAZOUZ Najlae

Encadré par :

THIERRY PERISSE

M1 SME
2022 -- 2023

SOMMAIRE

INTRODUCTION

Dans le cadre de ce TP, nous allons approfondir nos ressources et connaissances en matière de capteurs de température et de gyroscope, en utilisant la carte de développement STM32 pour le traitement des données et leur implémentation dans une solution plus complexe.

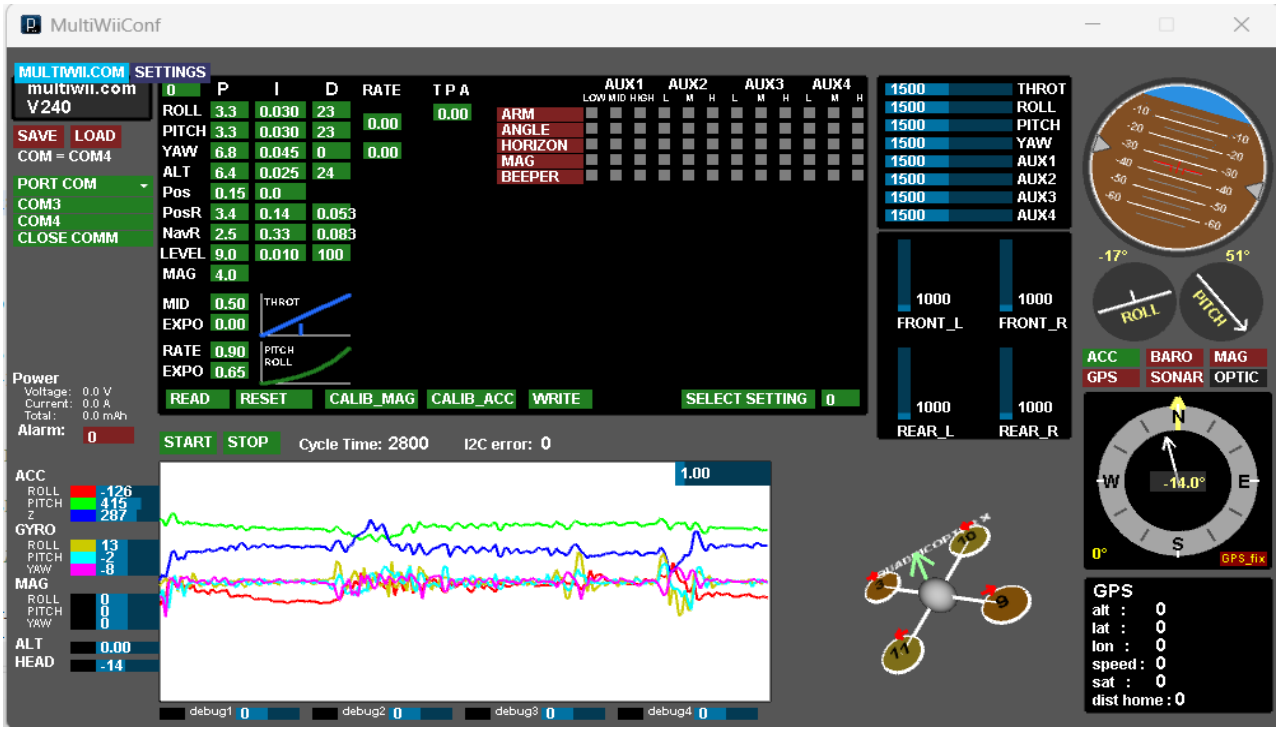
Concrètement, nous allons utiliser ces composants pour la simulation d'un drone, en utilisant l'interface graphique MultiWii. Cette dernière nous aidera à recevoir les données de différentes manières, telles que la communication radio, le protocole SPI ou une connexion WIFI.

Grâce à cette expérience, nous pourrons obtenir des résultats plus précis et des données plus détaillées sur les capteurs de température et d'humidité, ainsi que sur les valeurs d'accélération et de gyroscope, tout en observant les changements sur l'interface graphique.

Finalement, nous serons en mesure de transférer ces données entre la carte STM32 et la carte Arduino via le protocole WIFI ou SPI, afin d'optimiser la qualité de nos résultats et de mieux comprendre le fonctionnement des capteurs et de leurs interactions avec l'interface graphique.

I) Description du système:

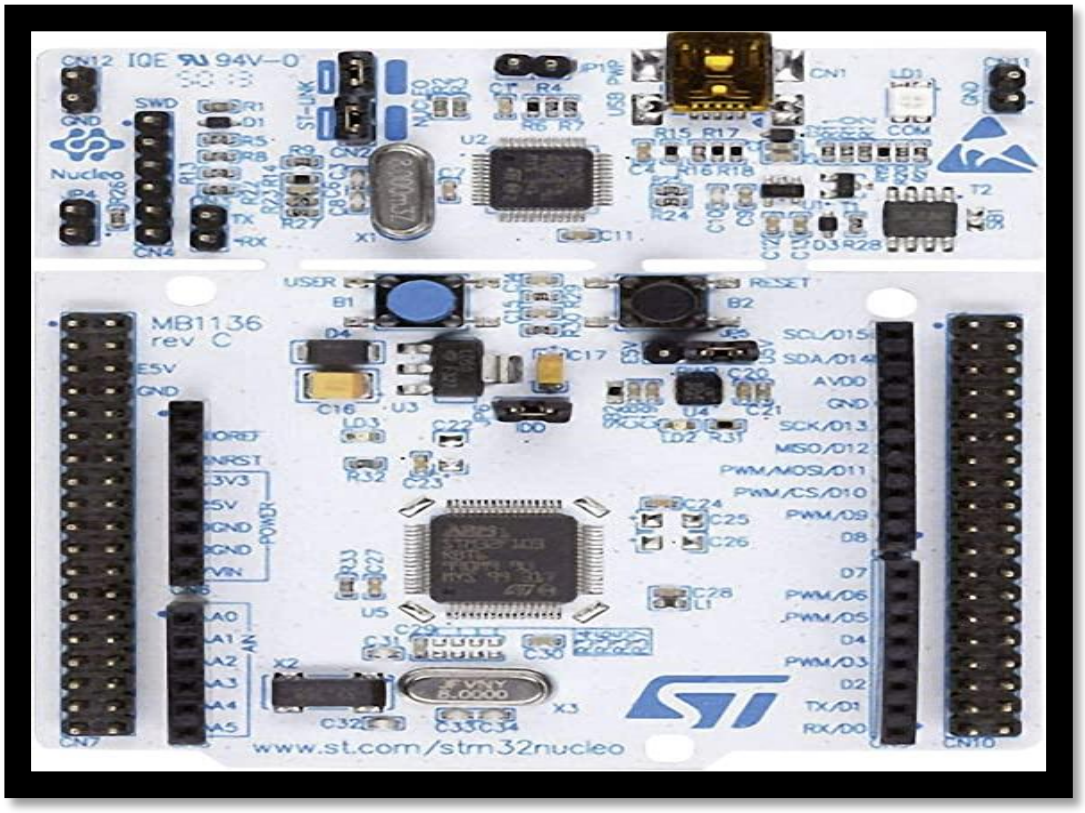
Le système que nous allons mettre en place est très sophistiqué et utilise deux capteurs de haute qualité pour collecter et transmettre des données. Le premier capteur, le SHT31, est spécialement conçu pour mesurer la température et l'humidité avec une précision extrême. Ce capteur est capable de fournir des données météorologiques en temps réel, ce qui peut être très utile dans de nombreuses applications telles que l'agriculture, la construction et la surveillance environnementale. Le deuxième capteur, l'IMU 9DOF, est un capteur d'accélération et de gyroscope qui peut mesurer la position et les mouvements avec une extrême précision. Cela permet de surveiller les vibrations, les chocs et les mouvements avec précision, ce qui peut être extrêmement utile dans des applications telles que les véhicules autonomes, les drones et les robots.



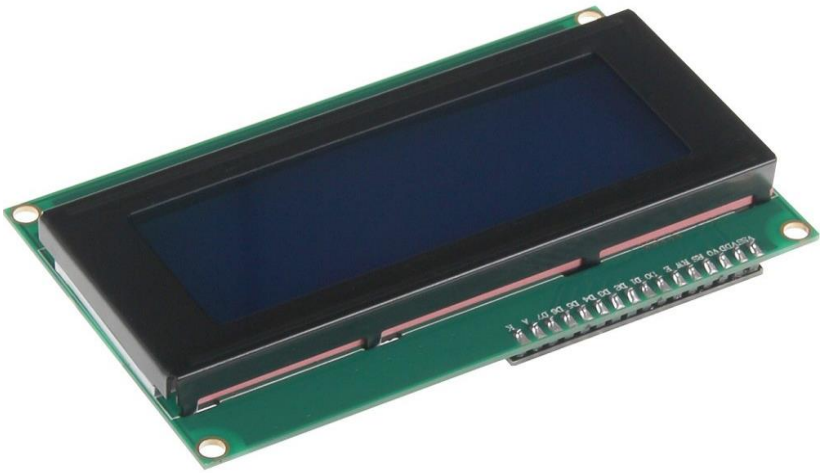
Grâce à toutes ces données, nous serons en mesure de lire les paramètres du drone avec une grande précision et de les afficher clairement sur l'écran LCD. Cela nous permettra de visualiser en temps réel le mouvement du drone sur l'interface graphique, et de mieux comprendre son comportement et sa trajectoire. En utilisant ces informations, nous pourrons également ajuster et améliorer les réglages du drone pour optimiser ses performances et répondre aux besoins spécifiques de chaque mission. En somme, cette approche nous permettra d'obtenir une vue d'ensemble complète et détaillée de l'état du drone, et de prendre des décisions éclairées en temps réel pour garantir la réussite de chaque mission.

II) Matériels utilisés :

1) Carte Nucléo STM32:

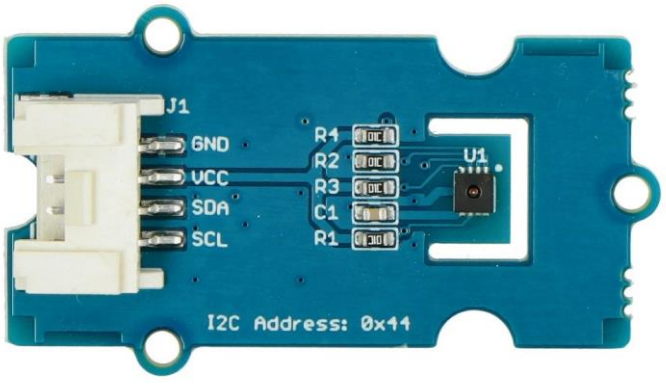


2) LCD 20X4 :

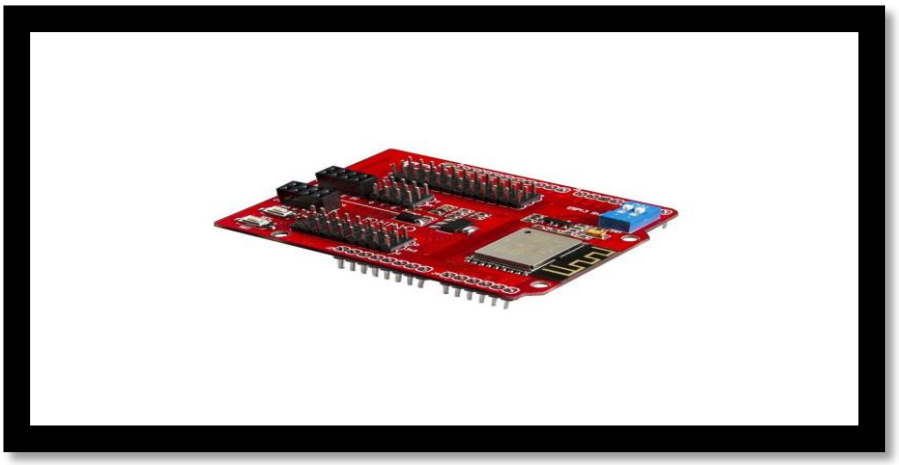


3) CAPTUER IMU 9DOF :

4) CAPTUER SHT31:



5) Module WIFI ESP8266 :



Le module ESP8266 est un microcontrôleur Wi-Fi à faible coût et à faible consommation d'énergie qui est largement utilisé dans les projets IoT (Internet des objets). Lorsqu'il est associé à une carte Arduino, il permet à votre projet de se connecter à un réseau Wi-Fi et de communiquer avec d'autres appareils connectés.

Voici quelques informations importantes sur le module ESP8266 :

1. Fonctionnalités : L'ESP8266 est capable de se connecter à un réseau Wi-Fi en tant que client et peut également agir en tant que point d'accès (AP) pour permettre à d'autres appareils de se connecter à lui. Il dispose d'une pile TCP/IP intégrée, ce qui lui permet d'établir des connexions réseau et de communiquer avec des serveurs Web.

2. Programmation : L'ESP8266 peut être programmé à l'aide de l'IDE Arduino, ce qui facilite le développement de projets en utilisant le langage de programmation Arduino. Vous pouvez écrire du code Arduino standard pour contrôler le module ESP8266 et exploiter ses fonctionnalités Wi-Fi.

3. Bibliothèques : Plusieurs bibliothèques sont disponibles pour faciliter la programmation de l'ESP8266 avec Arduino. L'une des plus populaires est la bibliothèque ESP8266WiFi, qui fournit des fonctions pour gérer les connexions Wi-Fi, envoyer des requêtes HTTP, etc. Il existe également des bibliothèques spécifiques pour des fonctionnalités telles que la transmission de données MQTT (par exemple, PubSubClient).

4. Modes de fonctionnement : L'ESP8266 offre différents modes de fonctionnement, tels que le mode Station (STA), le mode Point d'accès (AP) et le mode Station + Point d'accès. En mode STA, il se connecte à un réseau Wi-Fi existant. En mode AP, il crée un réseau Wi-Fi auquel d'autres appareils peuvent se connecter. Le mode Station + Point d'accès permet de combiner les deux modes.

5. Pins et alimentation : Le module ESP8266 dispose de plusieurs broches d'E/S (Entrée/Sortie) qui peuvent être utilisées pour se connecter à d'autres composants ou capteurs. Il peut être alimenté à partir d'une tension de 3,3 V, mais il est important de noter que les broches d'E/S sont également limitées à cette tension. Veillez donc à ne pas y connecter de composants qui nécessitent une tension de 5 V.

6. Communauté et ressources : L'ESP8266 est soutenu par une grande communauté d'utilisateurs et de développeurs. Vous pouvez trouver de nombreux tutoriels, exemples de code et forums en ligne où vous pouvez obtenir de l'aide et partager vos projets. Les sites populaires tels que GitHub et Instructables regorgent de projets basés sur l'ESP8266.

6) Module radio NRF24 :

L Le module radio NRF24 est un module de communication sans fil à faible coût et à faible consommation d'énergie, conçu pour la transmission de données à courte distance. Il est souvent utilisé dans les projets d'IoT (Internet des objets) et de domotique pour établir des communications sans fil entre différents appareils.

Voici quelques informations importantes sur le module radio NRF24 :

1. Fonctionnalités : Le module NRF24 est basé sur la puce nRF24L01+ de Nordic Semiconductor. Il prend en charge la communication bidirectionnelle à faible débit de données sur des distances relativement courtes (quelques dizaines de mètres). Il fonctionne sur la bande de fréquences 2,4 GHz, ce qui signifie qu'il peut potentiellement rencontrer des interférences provenant d'autres appareils utilisant la même fréquence.

2. Protocole de communication : Le module NRF24 utilise un protocole de communication propriétaire basé sur la modulation de fréquence à décalage de phase (FSK) pour transmettre les données. Il permet de configurer des réseaux sans fil à plusieurs nœuds, ce qui signifie que plusieurs modules NRF24 peuvent être utilisés dans un même système de communication.

3. Bibliothèques et programmation : Pour utiliser le module NRF24 avec une carte Arduino, vous pouvez utiliser des bibliothèques telles que la bibliothèque RF24. Ces bibliothèques fournissent des fonctions et des exemples de code qui facilitent la communication entre les modules NRF24. Vous pouvez programmer les fonctionnalités de communication à l'aide du langage Arduino.

4. Configuration et paramètres : Le module NRF24 peut être configuré à l'aide de différents paramètres, tels que la puissance de transmission, la fréquence de fonctionnement, le canal de communication et les adresses des nœuds. La configuration de ces paramètres est nécessaire pour établir une communication fiable entre les modules NRF24.

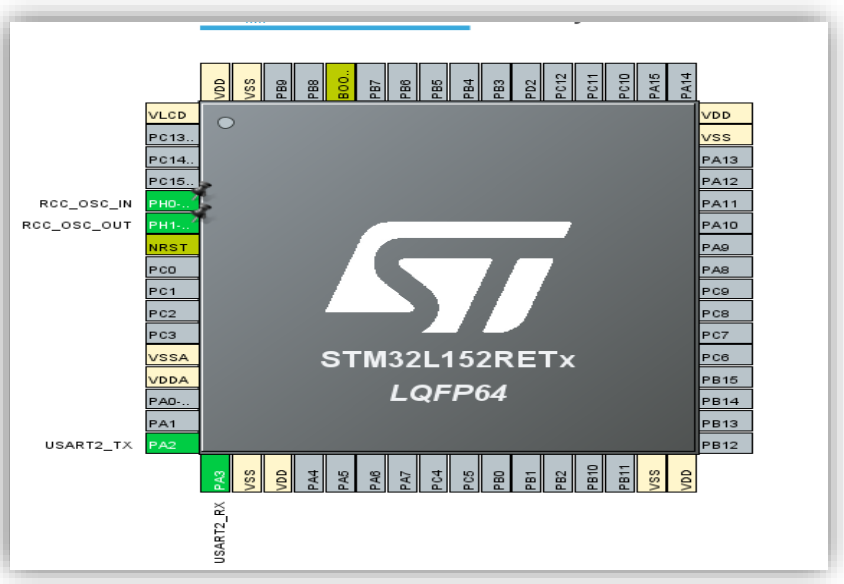
5. Alimentation et broches : Le module NRF24 fonctionne généralement à une tension de 3,3 V. Il dispose de plusieurs broches d'E/S (Entrée/Sortie) pour se connecter à la carte Arduino ou à d'autres microcontrôleurs. Il est important de noter que les broches du module NRF24 fonctionnent à une tension de 3,3 V et ne doivent pas être connectées directement à des broches d'E/S Arduino qui fonctionnent à 5 V. Une conversion de niveau logique peut être nécessaire.

Le module radio NRF24 offre une solution abordable et pratique pour établir des communications sans fil à courte distance dans vos projets Arduino. Que vous souhaitiez créer un réseau de capteurs sans fil, contrôler des appareils à distance ou mettre en place une solution de domotique, le module NRF24 peut être une option intéressante.

III) Implémentation du projet BE:

1) Wifi ESP8266 + STM32 :

ce code initialise le microcontrôleur, configure l'horloge et les périphériques GPIO et USART, initialise un module ESP et démarre un serveur pour gérer les communications. Il s'exécute ensuite dans une boucle infinie pour maintenir le fonctionnement du serveur.



Programmation :

Une fois le câblage réalisé, nous avons utilisé le logiciel STM32CubeMX pour configurer les broches nécessaires à notre projet et générer le code en C en suivant le programme disponible dans notre dépôt GitHub.

Des petits détails du code :

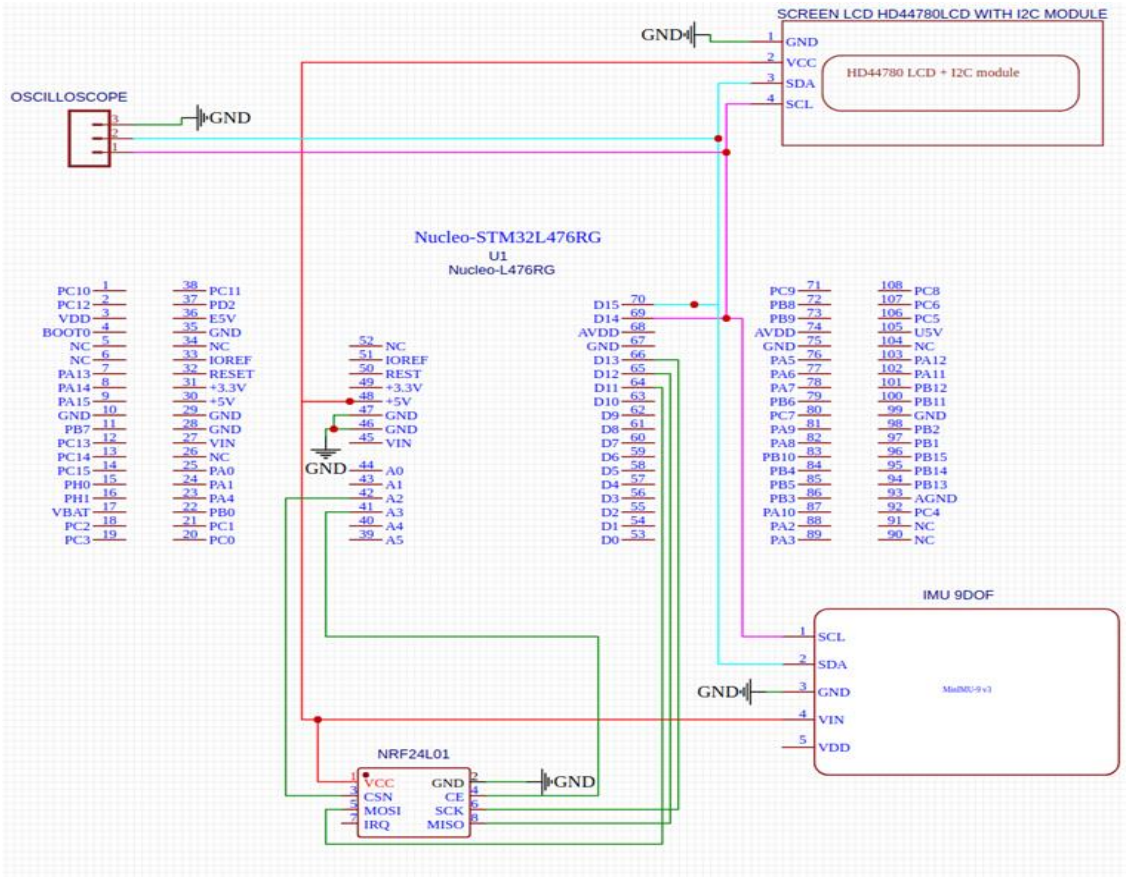
1. Les fonctions MX_GPIO_Init() et MX_USART2_UART_Init() sont appelées pour initialiser les périphériques GPIO (General-Purpose Input/Output) et USART2 (Universal Synchronous/Asynchronous Receiver/Transmitter) respectivement. Ces fonctions sont générées automatiquement par le programme de génération de code STM32CubeMX en fonction de la configuration matérielle.
2. La fonction ESP_Init() est appelée pour initialiser le module ESP8266. Elle prend trois paramètres : le nom du réseau Wi-Fi, le mot de passe du réseau Wi-Fi et l'adresse IP du serveur ESP8266.
3. À l'intérieur de la boucle infinie while(1), la fonction Server_Start() est appelée. Cette fonction est spécifique à l'application et est probablement responsable de la gestion des connexions au serveur.

L'intérêt de ce code est de configurer et d'initialiser les périphériques du microcontrôleur STM32, d'initialiser le module ESP8266 et de démarrer le serveur. Le microcontrôleur peut ainsi communiquer avec d'autres appareils via le module ESP8266 et exécuter des tâches spécifiques en fonction des connexions et des données reçues.

REMARQUE : Nous avons tenté initialement d'intégrer la carte STM32 avec le nouveau module Wifi ESP8266 pour récupérer les données souhaitées, mais nous n'avons pas réussi à mener le projet à terme. Donc On était censé de d'essayer avec un autre module, le "**Radio NRF24**".

2) Transmetteur (STM32) :

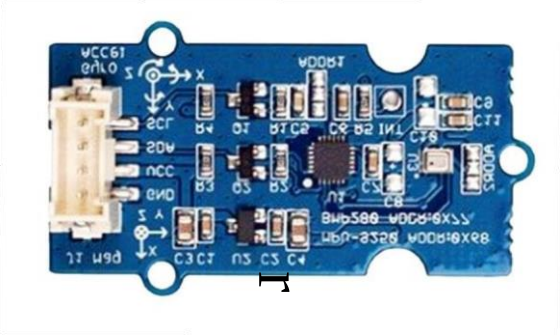
a) Schéma électrique :



b) Présentation de schéma électrique :

- **Utilisation des capteurs et LCD :**

Nous avons opté pour l'utilisation d'un seul capteur, l'IMU 9DOF, plutôt que deux, car cela simplifie grandement la méthode. En utilisant le protocole de communication I2C, il suffit de modifier l'adresse du capteur IMU 9DOF pour obtenir une variété de données supplémentaires, telles que les données de température et d'humidité, en plus des valeurs du gyroscope et de l'accélération.



Ce choix nous a permis de réduire la complexité de notre approche sans compromettre la qualité des données recueillies. De plus, cette méthode nous a permis d'économiser des coûts en n'utilisant qu'un seul capteur plutôt que deux, tout en améliorant l'efficacité de notre processus de collecte de données. En somme, cette stratégie innovante nous a permis d'améliorer notre approche et de réduire les coûts tout en garantissant une qualité de données élevée.

Pour l'adresse de l'accélération et gyroscope : **0x68**

Pour l'adresse de la température et l'humidité : **0x0C**

Pour l'adresse de l'écran LCD : **0x4E**

D'où le capteur IMU et l'écran seront liés à la carte STM32 par la communication I2C et à l'aide de mettre la bonne adresse lors de la communication ou le rappel dans le code :

→ **#define MPU6050_ADDR 0x68**

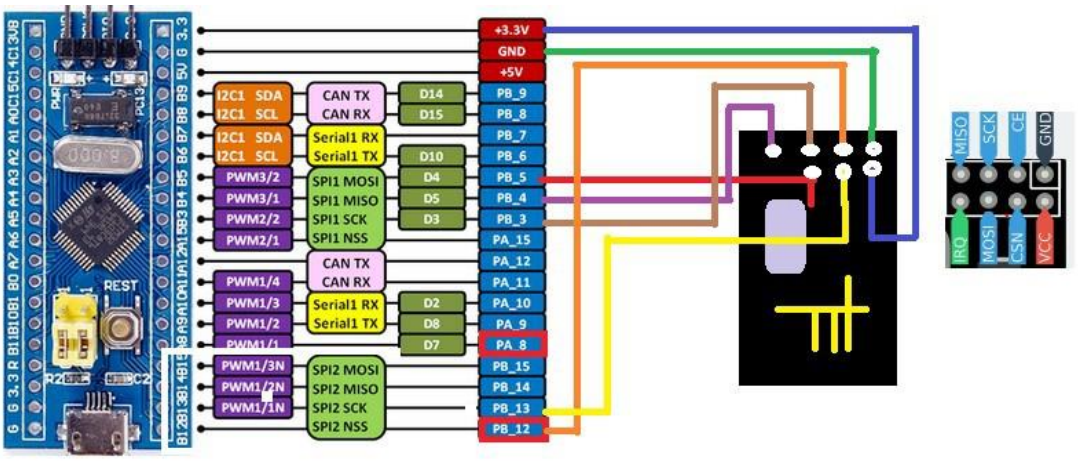
→ **#define SLAVE_ADDRESS_LCD 0x4E**

→ **HAL_I2C_Mem_Read (&hi2c1, MPU6050_ADDR,WHO_AM_I_REG,1, &check, 1, 1000);**

→ **HAL_I2C_Mem_Read (&hi2c1, temp_ADDR,2, &check, 1, 1000);**

.Bloc de communication SPI:

Pour la communication entre les deux cartes, entre le STM32 et l'Arduino Uno ça sera avec le protocole radio ou SPI qui sert a envoyer et recevoir les données dans les ondes radio à l'aide d'ajuster les paramètres de radio dans le code.



c) Programmation :

- ## Comment ça marche le module NRF24 :

Le module NRF24 est un module de communication sans fil très populaire. Il utilise la technologie RF pour transmettre des données à travers un signal de radiofréquence. Pour pouvoir l'utiliser, il est nécessaire de le configurer de manière appropriée. Ce module prend en charge les modes émission et réception, ce qui le rend très polyvalent. Il utilise un protocole de communication simple qui est basé sur l'envoi et la réception de paquets de données.

- Le module NRF24 est équipé d'une adresse unique pour permettre de différencier les modules entre eux. Cela permet de gérer plusieurs modules simultanément sans les confondre ou les mélanger. Il est également équipé d'une antenne intégrée, ce qui facilite grandement son utilisation.

- Pour les utilisateurs, il est important de savoir que le module NRF24 dispose de bibliothèques logicielles pour faciliter son utilisation. Ces bibliothèques permettent aux utilisateurs d'utiliser le module de manière plus simple et plus efficace.

À l’aide de logiciel STM32IDE nous allons programmer la carte STM32 pour arriver a se communiquer avec la carte Arduino.

- **Modes de transmission et de réception SPI STM32**

Cette section répertorie les différentes façons de gérer les transactions SPI dans les applications firmware. Pour des exemples de code et des tests, cliquez sur le bouton du prochain tutoriel et continuez à travers la série de tutoriels. Il y aura beaucoup d'exemples et de bibliothèques basés sur la communication SPI.

- **SPI avec polling et interruptions**

Le polling pour la ressource matérielle jusqu'à ce qu'elle soit prête à passer à l'étape suivante des instructions de votre programme est la façon la moins efficace de faire les choses et le processeur perdra beaucoup de temps dans un état d'attente occupé. De même, avec les interruptions SPI, bien que cela permette de gagner beaucoup de temps et soit considéré comme la meilleure façon de gérer des événements de ce type, nous ne pouvons pas prédire quand elles arriveront ou pendant quelle tâche, ce qui peut perturber le comportement de synchronisation du système, surtout avec un bus de communication rapide comme le SPI qui peut bloquer le processeur avec de gros blocs de données à un taux élevé.

- **SPI intégration dans le logiciel :**

Voici l'onglet de configuration pour le périphérique SPI dans l'IDE et voici les modes possibles pour SPI :

- **Les fonctions de SPI dans le logiciel :**

→ Pour la transmission :

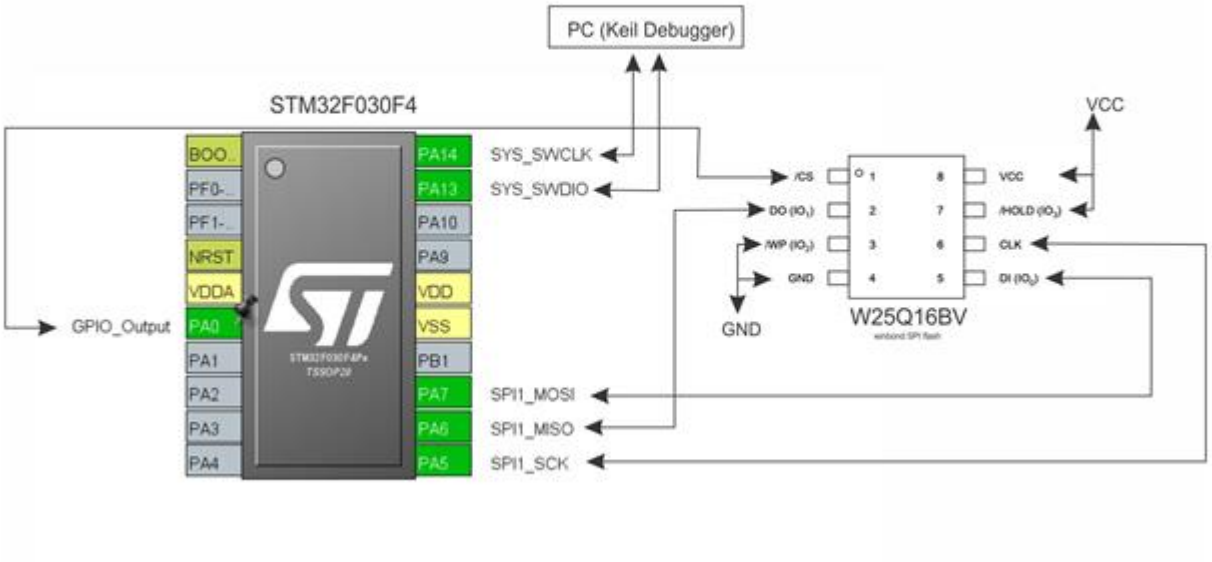
HAL_SPI_Transmit(SPI_HandleTypeDef * hspi, uint8_t * pData, uint16_t Size, uint32_t Timeout);

→ Pour la réception :

HAL_SPI_Receive(SPI_HandleTypeDef * hspi, uint8_t * pData, uint16_t Size, uint32_t Timeout);

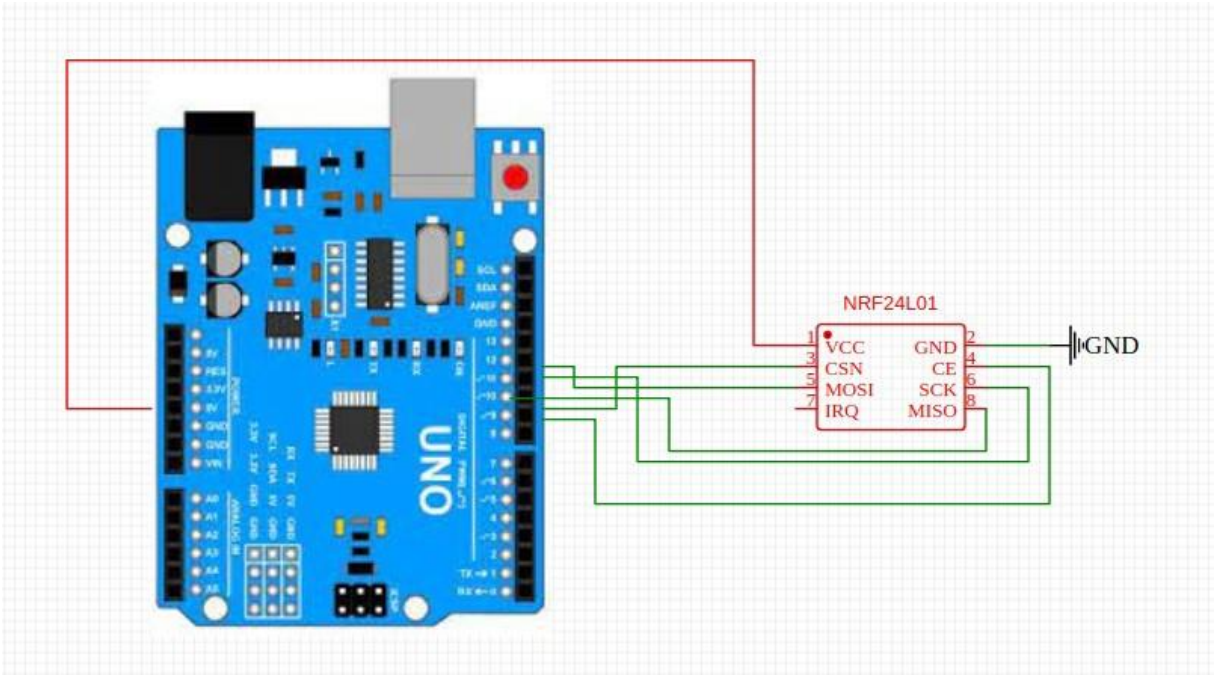
→ Pour les deux fonctions :

HAL_SPI_TransmitReceive(SPI_HandleTypeDef * hspi, uint8_t * pTxData, uint8_t * pRxData, uint16_t Size, uint32_t Timeout);



3) Récepteur (Arduino UNO):

a) Schéma électrique:



Le schéma est assez simple. Tout d'abord, vous devez connecter la carte Arduino Uno au module radio pour que la carte STM32 puisse communiquer avec elle et recevoir les données des capteurs IMU 9DOF. Une fois que vous avez ces données de capteurs, vous pouvez les utiliser pour afficher et implémenter votre capteur en tant que drone sur la plateforme MultiWii.

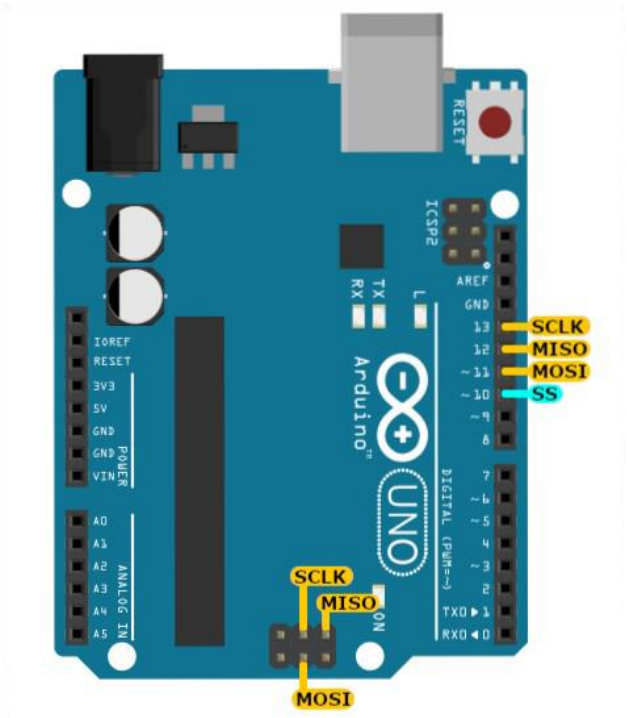
Il est important de noter que le module radio est essentiel pour communiquer avec la carte STM32. Sans lui, votre projet ne fonctionnera pas correctement. De plus, une fois que vous avez connecté les composants, vous pouvez commencer à collecter des données à partir des capteurs IMU 9DOF. Ces données peuvent être utilisées pour une variété de projets intéressants en plus de la plateforme MultiWii.

Enfin, il est important de comprendre que l'implémentation de votre capteur en tant que drone sur la plateforme MultiWii peut être un processus complexe, mais qui vaut certainement la peine d'être exploré. Avec un peu de patience et de pratique, vous pouvez créer des projets passionnants et innovants qui utilisent ces technologies fascinantes.

b) Programmation:

- **L'interface SPI de l'Arduino UNO:**

L'Arduino Uno dispose d'une prise en charge matérielle intégrée pour la communication SPI. Les broches SS/CS, MOSI, MISO et SCLK sont indiquées dans le diagramme ci-dessous :



Les broches 10 à 13 sont généralement utilisées, mais il y a aussi des broches MOSI, MISO et SCLK sur l'en-tête ICSP (près de la puce ATMEGA). Les broches MOSI, MISO et SCLK sur les broches 11 à 13 et sur l'en-tête ICSP sont identiques, donc l'utilisation de l'en-tête ICSP ne libère pas les broches 11 à 13 pour d'autres fins.

Dans le code il suffit d'ajuster la même onde de radio comme le STM32 pour bien recevoir les données de l'autre carte :

→ radio.begin();

→ radio.setChannel(115);

→ radio.setDataRate(RF24_250KBPS); F24_PA_MAX);

4) L'interface MultiWii :

MultiWii est un contrôleur de vol gyroscopique chargé de fonctionnalités. Cette version de MultiWii qui est compatible des dispositifs d'un gyroscope MEMS à 3 axes ITG-3205, d'un accéléromètre à 3 axes Bosch BMA180 ou d'un MCU ATmega 328P-AU.

ntenant utiliser divers capteurs mais a été initialement développé pour prendre en charge l

Ce projet a été l'occasion de développer mon propre logiciel sur une plateforme Arduino.

La stabilité atteinte est excellente pour les vols en FPV et permet toutes sortes d'acrobaties.

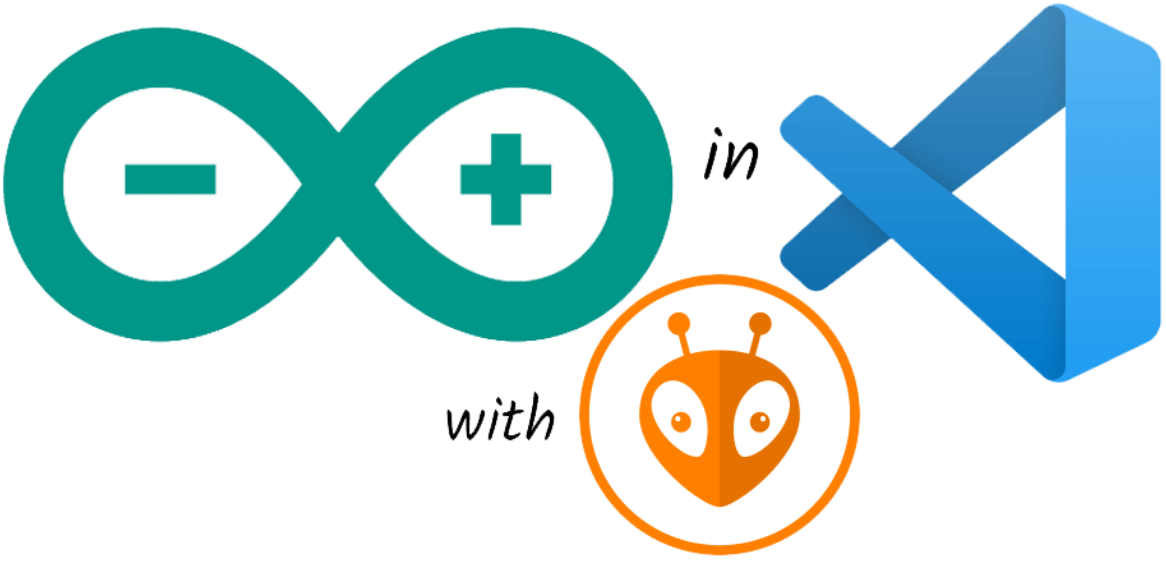
Le logiciel est pour le moment capable de contrôler un tricopter, un quadricopter ou un hexacopter.

Le tricopter mentionné dans cet article est principalement le premier tricopter que j'ai fabriqué, basé sur une structure en fibre.

La structure a été renforcée avec des fibres de carbone/kevlar. Des LEDs ont également été ajoutées pour une meilleure visibilité en vol.

5) Projet final :

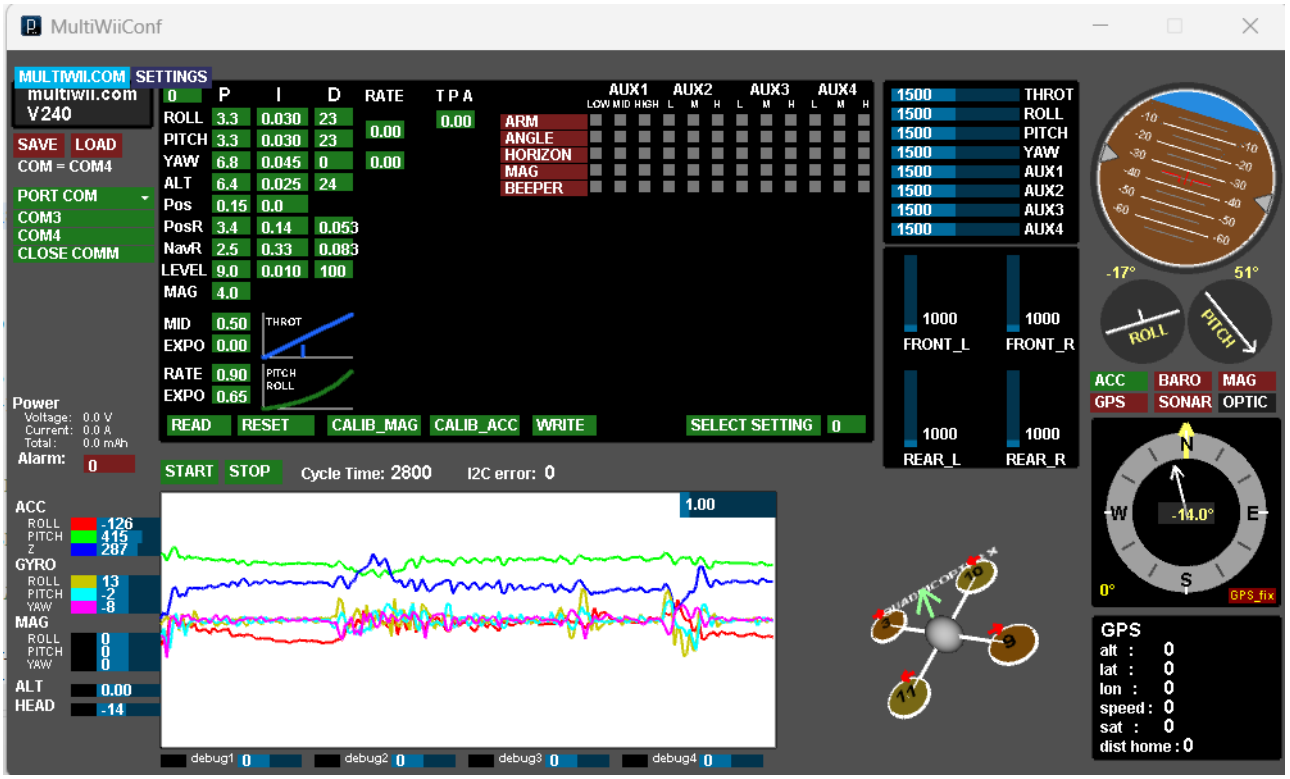
Pour le projet final, nous avons réussi à recevoir des données de la carte STM32 vers l'Arduino en utilisant la plateforme platforme.io dans Vscode, qui est un logiciel de programmation pour cartes électroniques avec une interface utilisateur différente. Cette plateforme nous a permis de configurer les paramètres de communication entre les deux cartes de manière plus efficace, en utilisant des protocoles de communication tels que I2C ou SPI. De plus, nous avons effectué des tests rigoureux pour nous assurer que les données étaient transférées de manière fiable et en temps réel. Tout cela a été possible grâce à notre équipe dévouée et à notre approche méthodique pour résoudre les problèmes techniques rencontrés lors du développement de notre projet.



On a eu l'opportunité de coder la carte STM32 avec le Firmware Arduino et avec une possibilité d'une simulation réele de projet et au même temps avec le STM32 IDE qui concerne les cartes de nucleo.

Nous avons mis tous les codes sur le répertoire de github pour plus d'information qt qui concerne le transmetteur et le recepteur.

Voici une image réele de l'implémentation de l'interface graphique Multiwii avec les données reçus sur la carte Arduino :

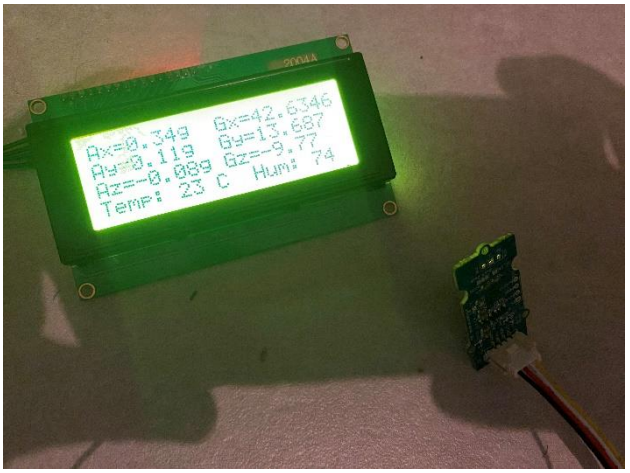


Danc cette interface graphique on peut visualiser plusieurs informations au même temps qui concerne :

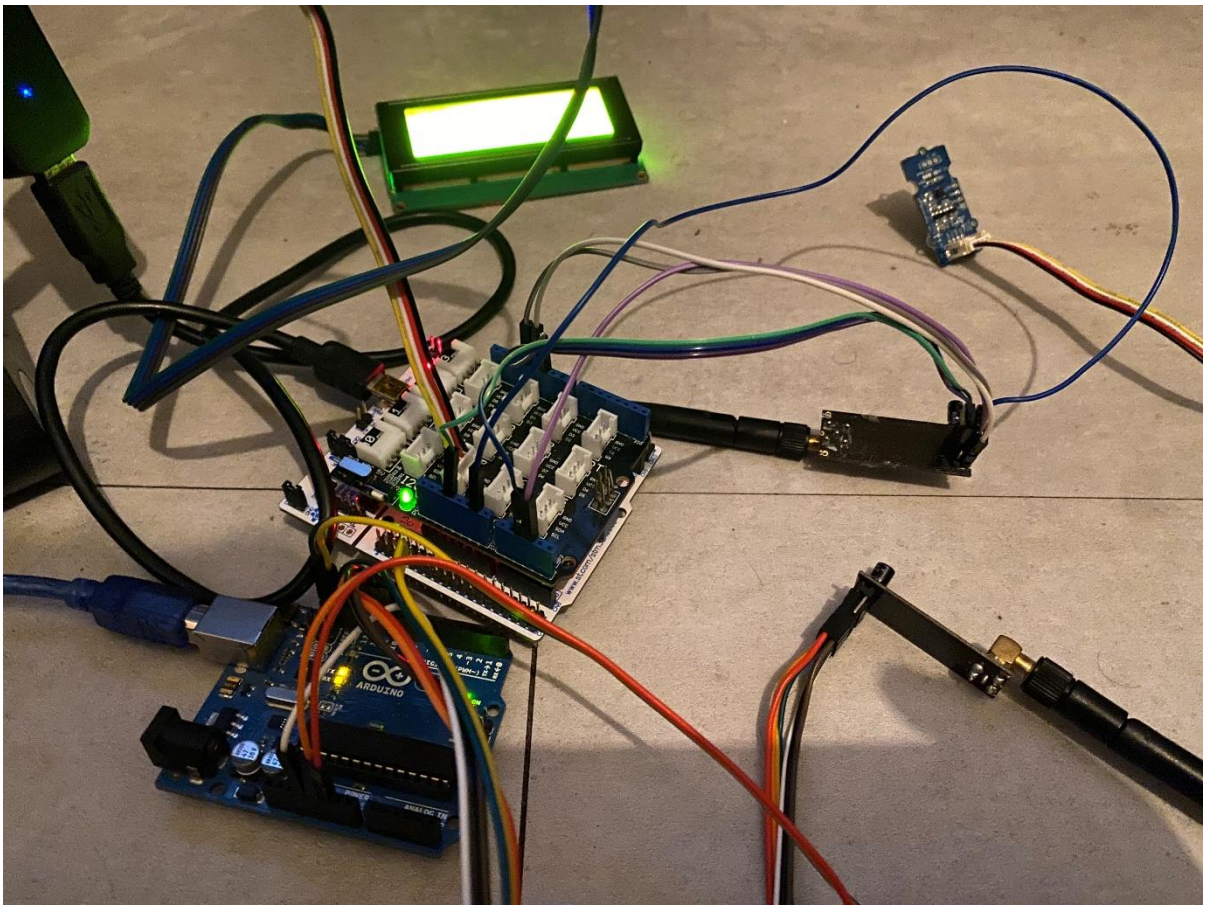
- Les données des gyroscope et de l'accéleration (ROLL / PITCH / YAW)
- La vitesse si y a une simualtion réele d'un drone
- Le GPS et le sens de mouvement
- Les informations de l'altitude

Avec cette interface il faut utiliser leur bibliothèque qui concerne la configuration de port, des données, de type de communication, de type de simulaiton....

Pour l'écran LCD il y a l'affichage de données de l'accéleration et le gyroscope ainsi pour la température et l'humidité :



Voici le projet général avec les deux cartes et le capteur :



Pour observer bien le résultat on vous laisse une vidéo sur Youtube avec le résultat de projet avec la simulation de projet, voici le lien de la video ainsi que le lien de la video est sur Github :

https://www.youtube.com/watch?v=d5eQsANhUu0

CONCLUSION

À l'issue de ce BE, nous avons acquis des compétences essentielles pour notre avenir professionnel. Parmi celles-ci, nous pouvons mentionner notre maîtrise de l'utilisation de la carte STM32 NUCLEO ainsi de produire un code compatible avec l'analyse des capteurs utilisés et le schéma électrique proposé. Ces compétences nous permettent de répondre aux diverses exigences spécifiées dans le cadre de projets industriels ou universitaires.

Contraintes : Au départ, nous avons tenté d'utiliser la carte STM32 avec le nouveau module Wifi ESP8266 pour collecter les données souhaitées, mais nous n'avons pas réussi à le finaliser. C'est pourquoi nous avons décidé d'essayer avec un autre module, le "Radio NRF24". Nous avons divisé le projet en deux parties, l'une utilisant le module WIFI ESP8266 et l'autre utilisant le module Radio NRF24.