École nationale supérieure de
l'électronique et de ses applications

Projet 1A : Gyroscope

Élèves :
Chentouf Rachid
Mayimona Zandu Merveille

Enseignant :
Jean-Michel Dumas

9 juin 2023

Table des matières

1 Résumé

Ce rapport présente le projet de conception et de réalisation d'un gyroscope ADXRS450. L'objectif principal du projet était de développer un système de capteur gyroscopique capable de mesurer et de fournir des informations précises sur la vitesse angulaire.

Le projet s'est déroulé en plusieurs étapes clés, notamment la conception de la carte de circuit imprimé (PCB), le choix et la connexion des composants, ainsi que la programmation du microcontrôleur Arduino pour communiquer avec le gyroscope ADXRS450 via le protocole SPI.

La conception du PCB a été réalisée en utilisant Kicad, avec une attention particulière accordée aux connexions électriques et à la disposition optimale des composants pour réduire les interférences électromagnétiques.

La phase de réalisation comprenait le soudage des composants sur la PCB, la connexion de la PCB au microcontrôleur Arduino, ainsi que la programmation du microcontrôleur pour envoyer des commandes au gyroscope ADXRS450 et récupérer les données de vitesse angulaire.

Les tests et les résultats ont démontré que le système fonctionnait correctement, fournissant des mesures précises de la vitesse angulaire lorsque le gyroscope était en mouvement. Les résultats ont été comparés à des références externes pour valider la précision des mesures.

Le projet a permis de développer des compétences en conception de circuits imprimés, en sélection de composants, en programmation embarquée et en intégration de systèmes électroniques. Il offre également des perspectives d'amélioration pour l'ajout de fonctionnalités supplémentaires et l'optimisation des performances.

Mots clés : gyroscope, ADXRS450, capteur de vitesse angulaire, PCB, Arduino, SPI.

2 Introduction

Le gyroscope ADXRS450 est un capteur de vitesse angulaire largement utilisé dans divers domaines tels que la robotique, la navigation inertielle et d'autres applications nécessitant des mesures précises de l'orientation et du mouvement.

Ce projet consistait à développer un système complet intégrant le gyroscope ADXRS450 à l'aide d'une carte de circuit imprimé (PCB) et d'un microcontrôleur Arduino. Le système devait être capable de lire les données de vitesse angulaire fournies par le gyroscope et de les traiter pour obtenir des informations exploitables.

Les objectifs principaux de ce projet sont les suivants :

- Comprendre le fonctionnement et les caractéristiques du gyroscope ADXRS450.
- Concevoir et réaliser un circuit imprimé (PCB) pour connecter le gyroscope ADXRS450 à un microcontrôleur.
- Programmer le microcontrôleur pour communiquer avec le gyroscope et récupérer les données de vitesse angulaire.
- Effectuer des tests et des mesures pour évaluer la précision et la fiabilité du système.

Ce projet revêt une grande importance car il permet d'acquérir des connaissances en matière de conception de circuits imprimés, de sélection de composants électroniques, de programmation de microcontrôleurs et de validation expérimentale.

Le rapport présentera les différentes étapes du projet, notamment la conception du circuit imprimé, le choix des composants, la programmation du microcontrôleur, les tests et les résultats obtenus. Il mettra en évidence les défis rencontrés et les solutions apportées, ainsi que les perspectives d'amélioration et d'exploitation futures du système.

3 Conception et spécifications

3.1 Description du système et de ses composants

Le système conçu comprend principalement deux composants clés : le gyroscope ADXRS450 et le microcontrôleur Arduino. Le gyroscope ADXRS450 est un capteur de vitesse angulaire qui mesure la rotation autour d'un axe spécifique. Il est doté de quatre broches de sortie, à savoir CS (Chip Select), MISO (Master In Slave Out), SCLK (Serial Clock) et MOSI (Master Out Slave In), qui permettent la communication avec le microcontrôleur.//

3.2 Les spécifications techniques du gyroscope ADXRS450 incluent

— Gyroscope de taux complet sur une seule puce.
— Mesure de la vitesse angulaire de ±300°/sec.
— Excellente stabilité de la déviation de zéro de 25°/heure.
— Capacité de survie aux chocs jusqu'à 2000 g.
— Sortie numérique SPI avec un mot de données de 16 bits.
— Faible bruit et faible consommation d'énergie pour une alimentation de 3,3 V et 5 V.
— Fonctionnement de -40°C à +105°C

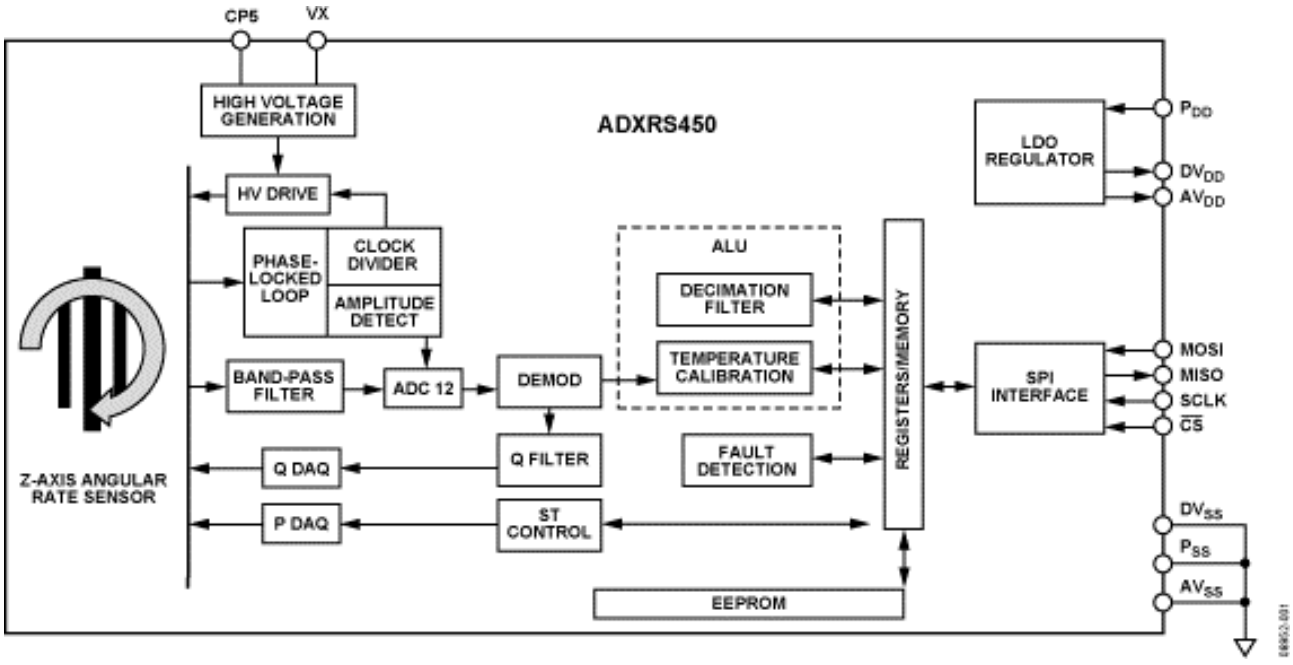


FIGURE 1 – ADXRS450 Datasheet

3.3 L'architecture du système comprend les éléments suivants

— Gyroscope ADXRS450 : Il mesure la vitesse angulaire et envoie les données au microcontrôleur via l'interface SPI.
— Microcontrôleur Arduino : Il communique avec le gyroscope ADXRS450 via l'interface SPI, reçoit les données de vitesse angulaire et les traite.
— Circuit imprimé (PCB) : Il permet de connecter le gyroscope ADXRS450 et le microcontrôleur Arduino de manière fiable et sécurisée.
— Alimentation : Fournit la tension requise pour alimenter le gyroscope et le microcontrôleur.

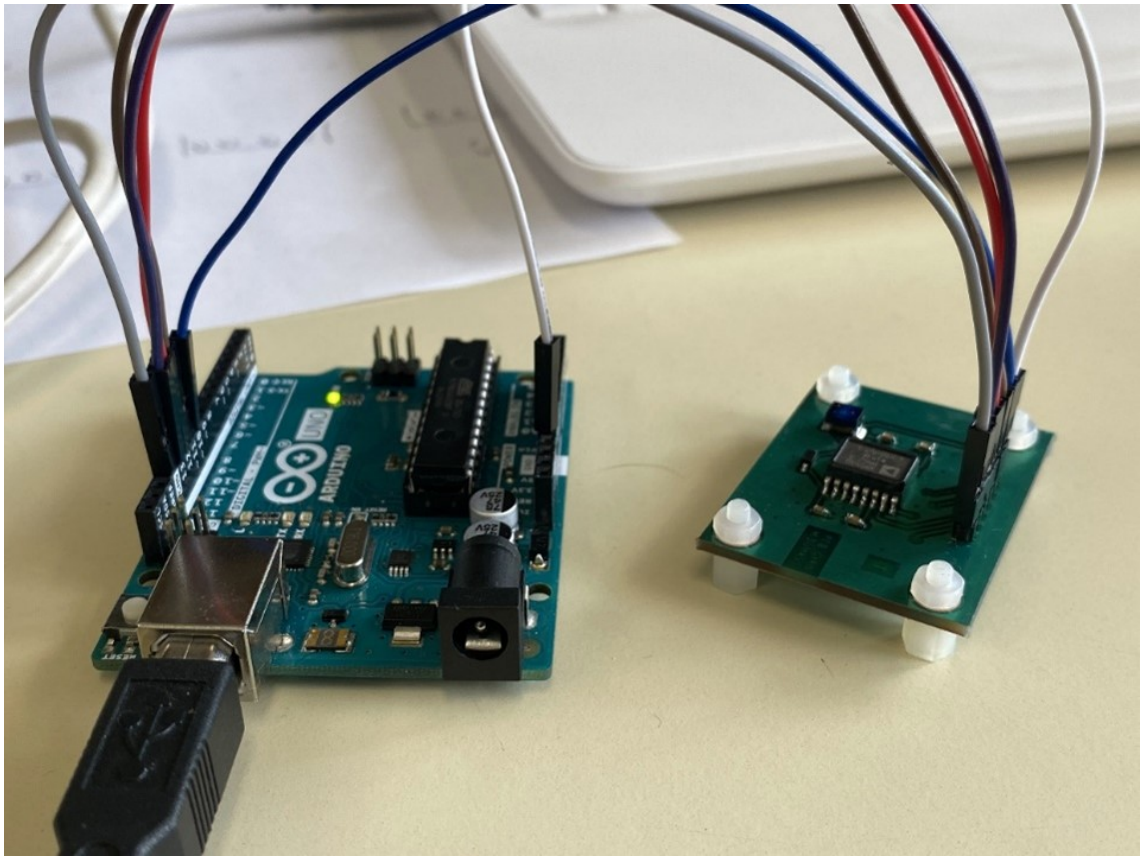


FIGURE 2 – Arduino avec notre PCB

5

4 Réalisation du PCB

Avant de commencer la conception de notre carte de circuit imprimé (PCB), nous avons consacré du temps à étudier attentivement la fiche technique de notre composant ADXRS450 afin de pouvoir le manipuler correctement. Cela nous a permis de comprendre les spécifications techniques, les broches de connexion et les fonctionnalités du gyroscope.

En utilisant le logiciel de conception de PCB Kicad, nous avons réalisé le schéma du circuit en plaçant les composants nécessaires et en les connectant de manière appropriée. Cependant, nous avons rencontré une difficulté lors de la recherche des bibliothèques de composants spécifiques dont nous avions besoin pour notre montage. Nous avons donc dû effectuer des recherches supplémentaires pour trouver les bibliothèques adéquates ou créer nos propres empreintes de composants.

Ce schéma nous a permis de réaliser la schématique sur Kicad.
Dans le monde du travail actuel, une pratique courante consiste à utiliser des composants électroniques de petite taille. Dans notre projet, nous avons suivi cette tendance en utilisant des condensateurs de taille 0603. Cela permet d'optimiser l'espace sur la carte PCB et de faciliter l'assemblage des composants. De plus, nous avons utilisé une bobine de taille 0804, qui est légèrement plus grande que les composants de type 0603.

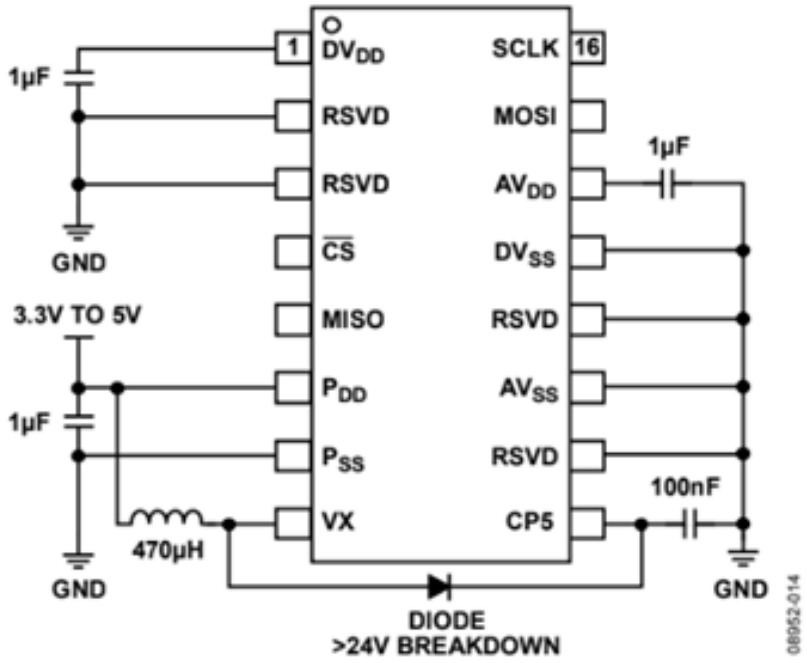


Figure 19. Recommended Applications Circuit, SOIC_CAV Package

FIGURE 3

Une fois le schéma terminé, nous sommes passés à l'étape du routage, où nous avons tracé les pistes de connexion entre les différents composants sur le PCB. Nous avons veillé à respecter les règles de conception et à optimiser l'agencement des composants pour minimiser les interférences et assurer un bon fonctionnement du circuit.

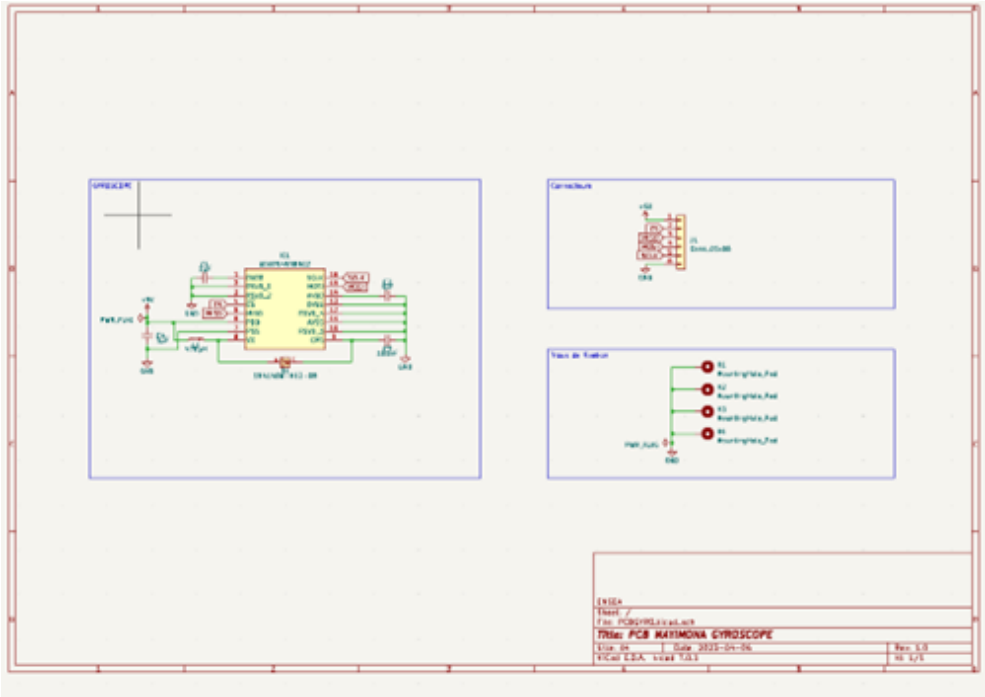
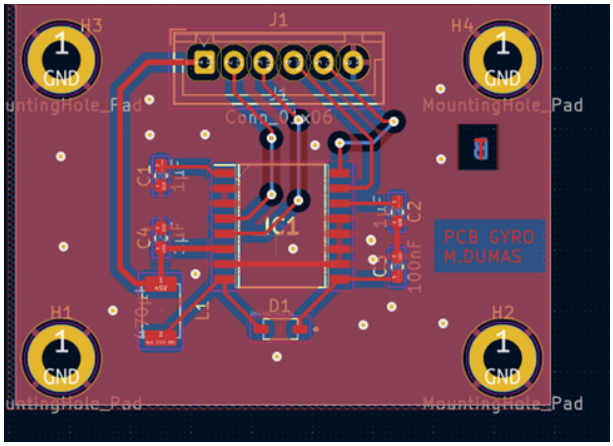


FIGURE 4 – Scematic du PCB

Une fois le PCB conçu, nous l'avons soumis à la validation de notre professeur. Après avoir obtenu son approbation, nous avons procédé à l'impression du PCB et à la soudure des composants. Cette étape a exigé de la précision et du soin pour assurer des connexions solides et fiables.

Les plans de masse isolés peuvent poser des problèmes de continuité et de référence de tension dans un circuit imprimé. L'ajout de vias permet de connecter les deux plans de masse, créant ainsi une liaison électrique entre eux

Une fois le PCB conçu, nous l'avons soumis à la validation de notre professeur. Après avoir obtenu son approbation, nous avons procédé à l'impression du PCB et à la soudure des composants. Cette étape a exigé de la précision et du soin pour assurer des connexions solides et fiables.



Les plans de masse isolés peuvent poser des problèmes de continuité et de référence de tension dans un circuit imprimé. L'ajout de vias permet de connecter les deux plans de masse, créant ainsi une liaison électrique entre eux.

7

5 Programmation du microcontrôleur

5.1 Protocole SPI

5.1.1 Expérience 1

Cette partie sera comme introduction au protocole SPI. Dans un premier temps, on essaye d'envoyer un int par exemple 145 et essayer de visualiser sur l'oscilloscope.
Tout d'abord, nous avons inclus la bibliothèque SPI et configuré les paramètres SPI nécessaires. Nous avons choisi un clockDivider de DIV64 pour ralentir le signal envoyé, ce qui nous permet de mieux visualiser les signaux sur l'oscilloscope.

```
void setup() {
  // Initialize SPI communication
  SPI.begin();

  // Configure SPI settings
  SPI.setBitOrder(MSBFIRST); // Most Significant Bit First
  SPI.setDataMode(SPI_MODE0); // Clock Polarity 0, Clock Phase 0
  SPI.setClockDivider(SPI_CLOCK_DIV64); // Set SPI clock frequency to 4MHz

  // Configure SS pin as output (master mode)
  pinMode(SS_PIN, OUTPUT);

  // Initialize the Serial Monitor
  Serial.begin(9600);
}
```

FIGURE 5

SPI.setBitOrder(MSBFIRST) : Le bit le plus significatif de chaque octet sera transmis en premier.
SPI.setDataMode(SPI_MODE0) : L'horloge démarre à un état bas (polarité 0) et les données sont échantillonnées sur le front montant de l'horloge.

8

Ici, on utilise la fonction SPI.transfer16() fonction pour envoyer des données à l'appareil via le bus SPI. Ensuite, dans la fonction loop(), nous sélectionnons l'appareil avec lequel nous voulons communiquer en définissant la broche SS à un niveau bas. Enfin, nous désélectionnons l'appareil en mettant la broche SS au niveau haut et attendons un peu avant d'envoyer plus de données.

```
void loop() {
  // Select the device (set SS pin low)
  digitalWrite(SS_PIN, LOW);

    // Wait for a moment before sending again
  delayMicroseconds(5);

  // Send data to the device
  int bit=145;
  for (int i = 0; i < 1; i++) {
    SPI.transfer16(bit); // Send ASCII code of character
  }

  // Deselect the device (set SS pin high)
  digitalWrite(SS_PIN, HIGH);

  // Print the sent data to the Serial Monitor
  //Serial.print("Sent: ");
  Serial.println(bit);


}
```

FIGURE 6

9

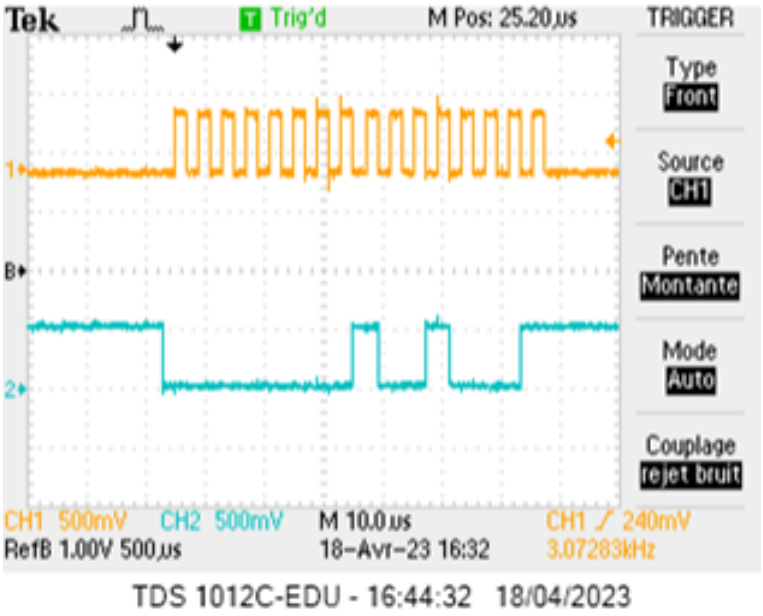


FIGURE 7 – SCLK en jaune, MOSI en bleu

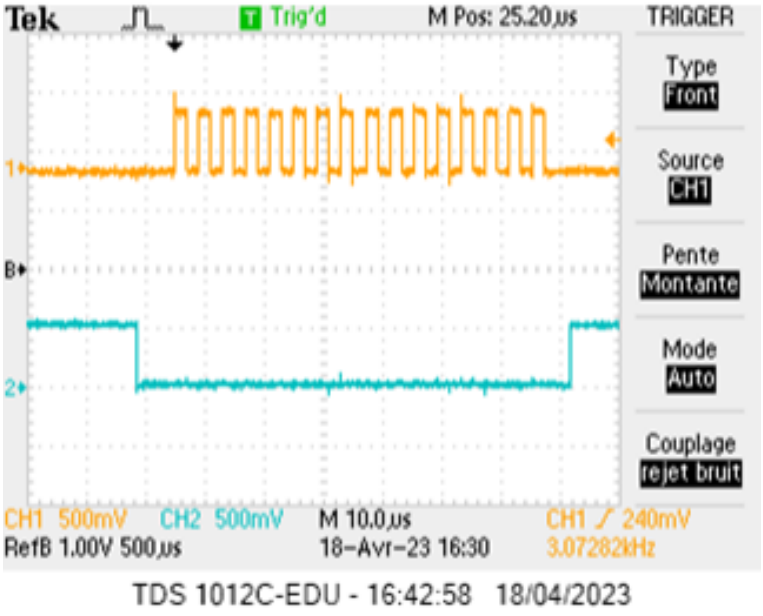


FIGURE 8 – SLCK en jaune,CS en bleu

10

5.1.2 Expérience 2

Dans un deuxième temps, nous avons exploré la communication entre deux cartes Arduino en utilisant le protocole SPI pour transmettre une chaîne de caractères de la carte émettrice à la carte réceptrice. Notre objectif était de réussir à envoyer la chaîne de caractères "abcdef" d'une carte Arduino à une autre

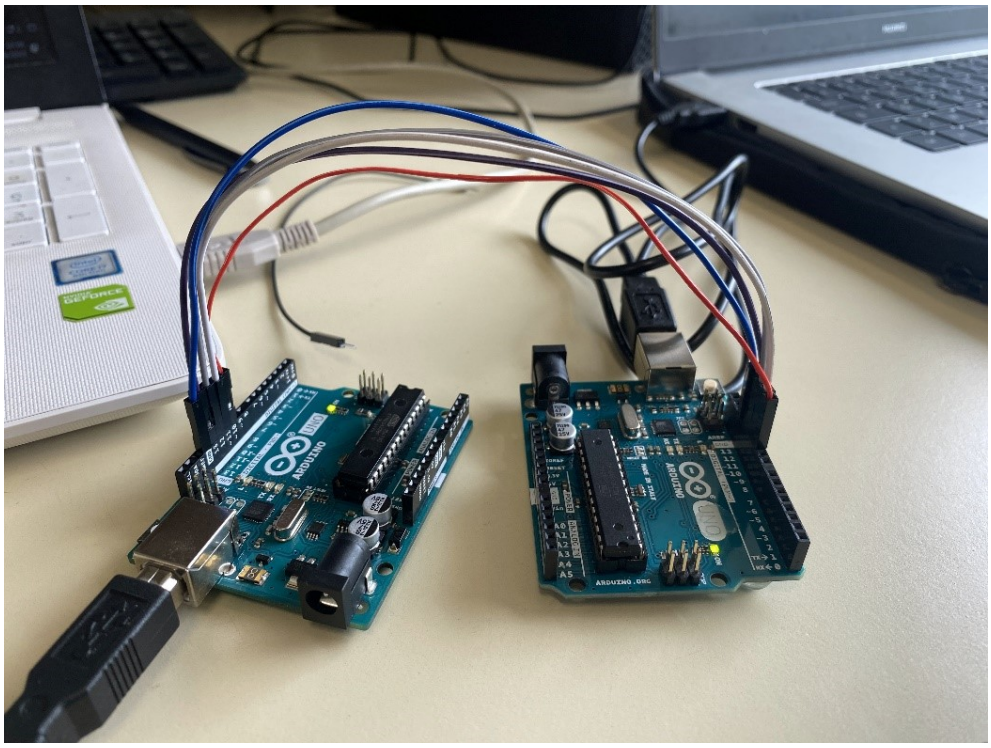


FIGURE 9 – Deux cartes Arduino branchées avec deux différents pcs

Pour cela, nous avons défini une variable de type string contenant la chaîne de caractères à envoyer. Ensuite, nous avons utilisé la fonction SPI.transfer pour transmettre les données. La taille de la chaîne de caractères "abcdef" est de 6, mais lorsque nous utilisons sizeof(data), nous obtenons une taille de 4.

11

```
void loop() {
  // Select the device (set SS pin low)
  digitalWrite(SS_PIN, LOW);

    // Wait for a moment before sending again
  delay(1000);

  // Send data to the device
 char data[]={"abc"};
  for (int i = 0; i < sizeof(data)-1; i++) {
    SPI.transfer(data[i]); // Send ASCII code of character
  }

  // Deselect the device (set SS pin high)
  digitalWrite(SS_PIN, HIGH);

  // Print the sent data to the Serial Monitor
  Serial.print("Sent: ");
  Serial.println(data);
```

FIGURE 10

Dans notre console de l'appareil émetteur, nous avons affiché le message "abc"

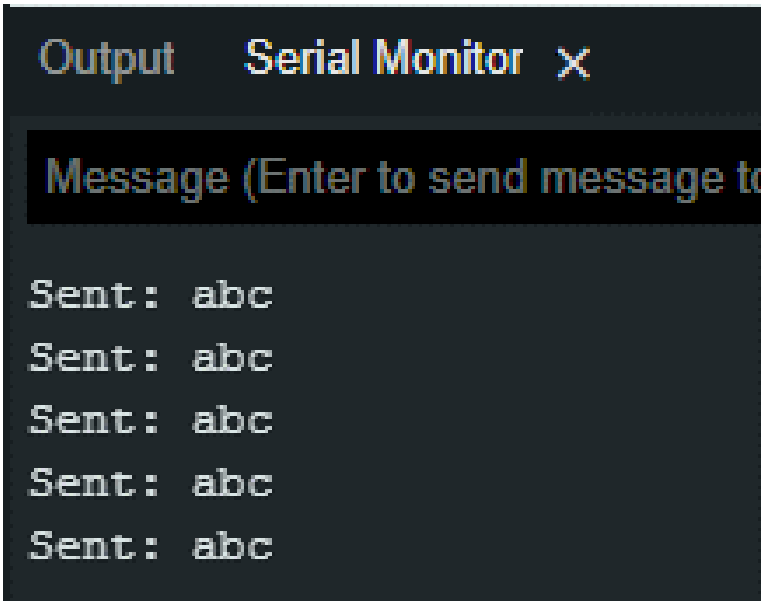


FIGURE 11

12

De même, dans la console de l'appareil récepteur, nous avons affiché le message reçu. Cependant, nous avons remarqué qu'il y avait un problème intermittent, où le message "abc" était reçu correctement une fois sur deux.

```
Received: abc
Received: bca
Received: abc
Received: bca
Received: abc
Received: bca
```

FIGURE 12

Malgré ce problème intermittent, nous avons réussi à envoyer avec succès le message "abc" d'une carte Arduino à une autre en reliant les 5 broches (SS, MOSI, MISO, SCLK, GND). Cela démontre la faisabilité de la communication entre les deux cartes Arduino via le protocole SPI.

Les deux signaux qu'on obtient sur l'oscilloscope :

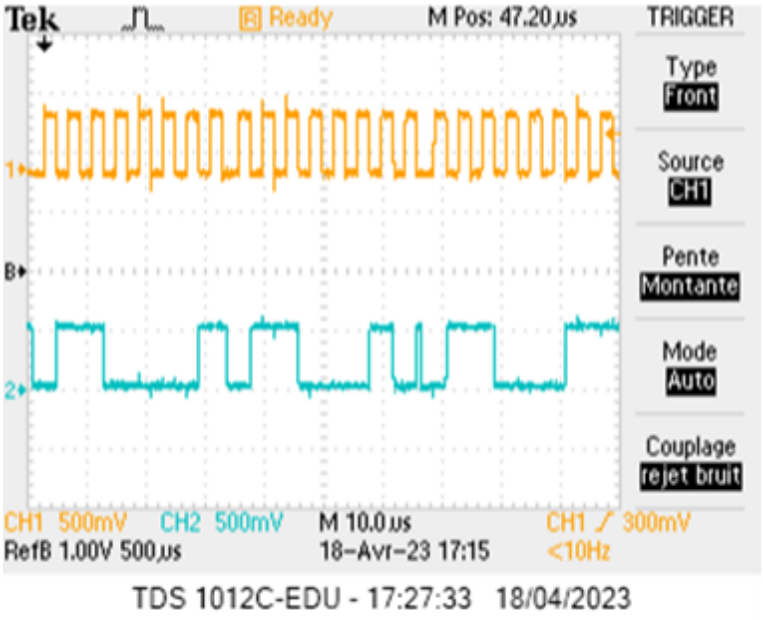


FIGURE 13 – SCLK en jaune, MOSI en bleu

13

5.2 Programmation du microcontrôleur

Dans cette partie essentielle de notre projet, nous nous sommes concentrés sur la programmation du microcontrôleur pour lire les données de vitesse angulaire du gyroscope ADXRS450 en utilisant le protocole SPI et les afficher sur le moniteur série à l'aide du microcontrôleur Arduino.
Nous avons choisi le mode SPI_MODE0 car notre configuration d'horloge nécessite que les données soient échantillonnées sur le front montant de l'impulsion d'horloge et décalées sur le front descendant. Cette configuration est illustrée par la trace d'horloge jaune dans le diagramme.
Le message que nous envoyons dans un premier temps via MOSI est 0x2000 0003 puis des 0x2000 0000. Étant donné que nous travaillons avec des données de 16 bits, nous envoyons d'abord 0x2000, suivi de 0x0000.
En binaire, 0x2000 est représenté par 0010 0000 0000 0000. C'est exactement ce que nous observons pendant le cycle de SCLK sur l'oscilloscope. Ces résultats sont conformes aux données de la séquence de démarrage (Start-Up Sequence) du gyroscope, ce qui confirme que nous sommes sur la bonne voie.

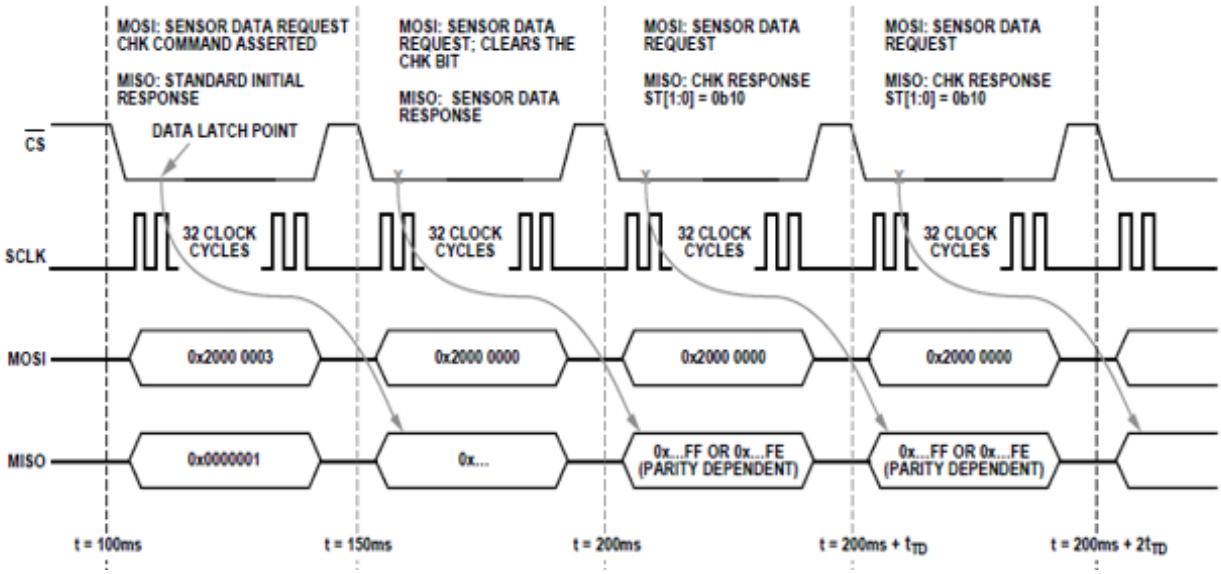


FIGURE 14 – Séquence de démarrage recommandée

14

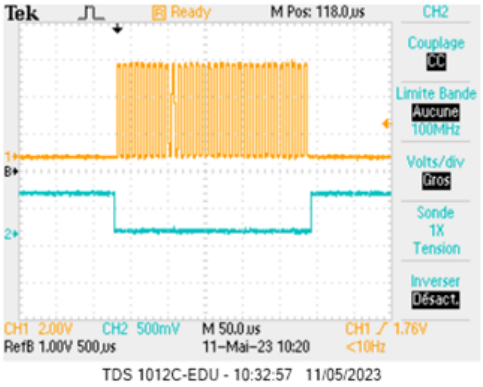


FIGURE 15 – SCLK en jaune, CS en bleu

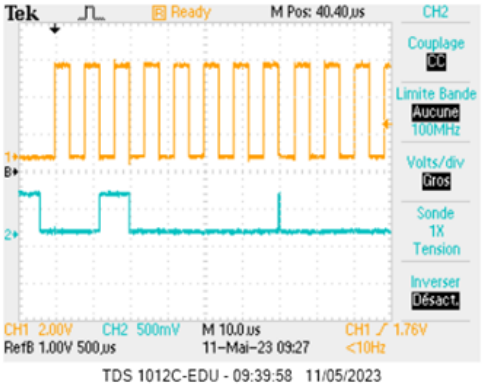


FIGURE 16 – SCLK en jaune, MOSI en bleu

Pour récupérer la vitesse de rotation, il est nécessaire d'extraire les données contenues dans les registres du gyroscope.

Table 9. SPI Responses

	Bit																													
Command	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Sensor Data	SQ2	SQ1	SQ0	P0	ST1	ST0	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0			PLL	Q	NVM	POR	PWR	CST	CHK	P1
Read	0	1	0	P0	1	1	1	0	SM2	SM1	SM0	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0					P1
Write	0	0	1	P0	1	1	1	0	SM2	SM1	SM0	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0					P1
R/W Error	0	0	0	P0	1	1	1	0	SM2	SM1	SM0	0	0	SPI	RE	DU									PLL	Q	NVM	POR	PWR	CST	CHK	P1

FIGURE 17

En tournant le gyroscope dans le sens positif, nous constatons une augmentation claire de la vitesse, et cette variation de vitesse est également reflétée sur notre console.

En revanche, lorsque nous tournons le gyroscope dans le sens inverse, la vitesse change de signe. Ces résultats sont cohérents avec la réalité et démontrent le bon fonctionnement du gyroscope et de notre programme de lecture des données.

15

Les premières vitesses (0.007 rad/s en moyenne) représentent l'état initial où le gyro était fixe avant de le faire tourner.
Quand on le fait tourner dans le sens direct, la vitesse augmente. Après il revient à sa position initiale.

```
Angular Velocity: 0.0079685 rad/s
Angular Velocity: 0.0069513 rad/s
Angular Velocity: 0.0089735 rad/s
Angular Velocity: 0.0075411 rad/s
Angular Velocity: 0.0061267 rad/s
Angular Velocity: 0.0075139 rad/s
Angular Velocity: 0.0257481 rad/s
Angular Velocity: 0.1831309 rad/s
Angular Velocity: 0.8621283 rad/s
Angular Velocity: 1.3258070 rad/s
Angular Velocity: 1.1125876 rad/s
Angular Velocity: 0.7642499 rad/s
Angular Velocity: 0.9078181 rad/s
Angular Velocity: 0.1092065 rad/s
Angular Velocity: -0.9589968 rad/s
Angular Velocity: -0.4783639 rad/s
Angular Velocity: -0.4029811 rad/s
Angular Velocity: -0.2841319 rad/s
Angular Velocity: -0.1391861 rad/s
Angular Velocity: -0.0661897 rad/s
Angular Velocity: -0.0304768 rad/s
Angular Velocity: -0.0121841 rad/s
Angular Velocity: -0.0032123 rad/s
Angular Velocity: 0.0019718 rad/s
Angular Velocity: 0.0045638 rad/s
Angular Velocity: 0.0062962 rad/s
Angular Velocity: 0.0089949 rad/s
```

Cette courbe représente la variation de la vitesse quand le gyroscope est fixe sans le toucher. Les valeurs oscillent autour de la valeur 0.07.
Bref, notre gyroscope est maintenant opérationnel et les valeurs qui nous sort sont bien réelles dans les deux sens.

Figure 18

16

Ce qui serait intéressant est d'introduire une moyenne glissante en plus d'un filtre passe-bas pour mieux visualiser les résultats. Les variations de notre courbe sont devenues moins importantes, en d'autres termes on a bien réduit le bruit.

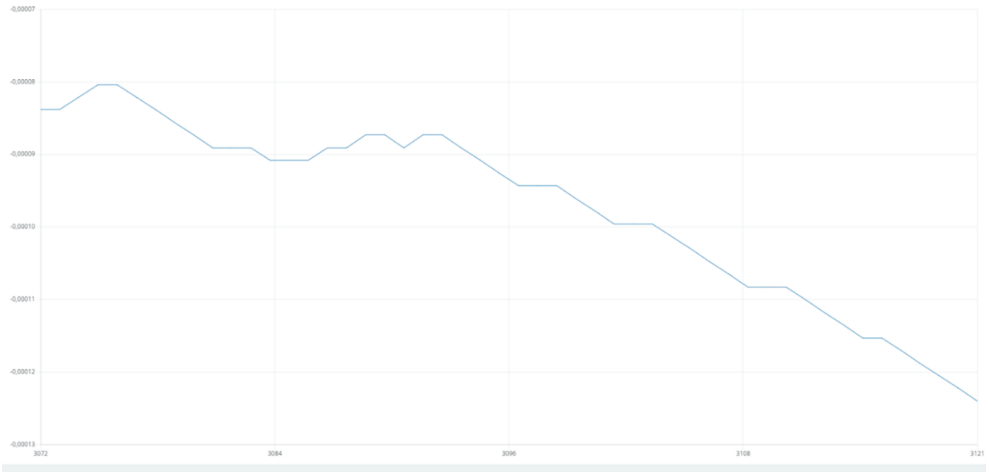


FIGURE 19

17

6 Améliorations proposées

6.1 Utilisation d'un servo-moteur à courant continu

L'étape suivante consiste à utiliser d'un servo-moteur à courant continu pour contrôler la rotation du support de notre gyroscope. Initialement, nous avions utilisé un servo-moteur standard, mais sa vitesse était trop élevée et non réglable. Cela ne correspondait pas à nos besoins spécifiques.

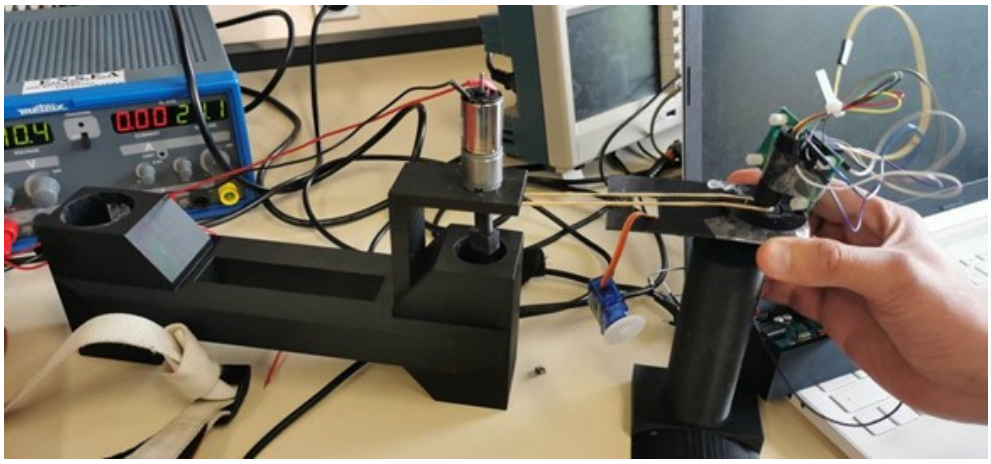


FIGURE 20

Pour remédier à cela, nous avons opté pour l'utilisation d'un servo-moteur à courant continu, qui offre une plus grande flexibilité en termes de vitesse de rotation. Ce type de servo-moteur peut être contrôlé en ajustant la tension appliquée à ses bornes. En augmentant la tension, la vitesse de rotation augmente également.

6.2 Connexion Bluetooth avec le HC-05

Dans le cadre de l'amélioration de notre projet, nous avons décidé de remplacer la connexion filaire entre notre Arduino et le gyroscope ADXRS450 par une connexion sans fil utilisant le module HC-05 Bluetooth.

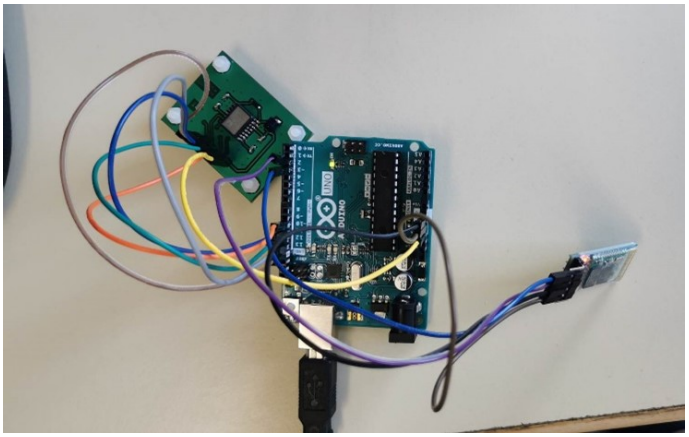


FIGURE 21

18

Cela nous permettra d'établir une communication bidirectionnelle entre le gyroscope et l'Arduino, sans avoir besoin d'un câble physique.
Nous avons intégré le module Bluetooth HC-05 fourni par le professeur. Le branchement physique du module (VCC, GND, RX, TX) a été réalisé conformément aux spécifications.

Pour établir la connexion entre le module HC-05 et notre ordinateur, nous avons procédé à l'appairage Bluetooth. Une fois l'appairage réussi, nous avons utilisé le logiciel PuTTY pour afficher les données échangées.

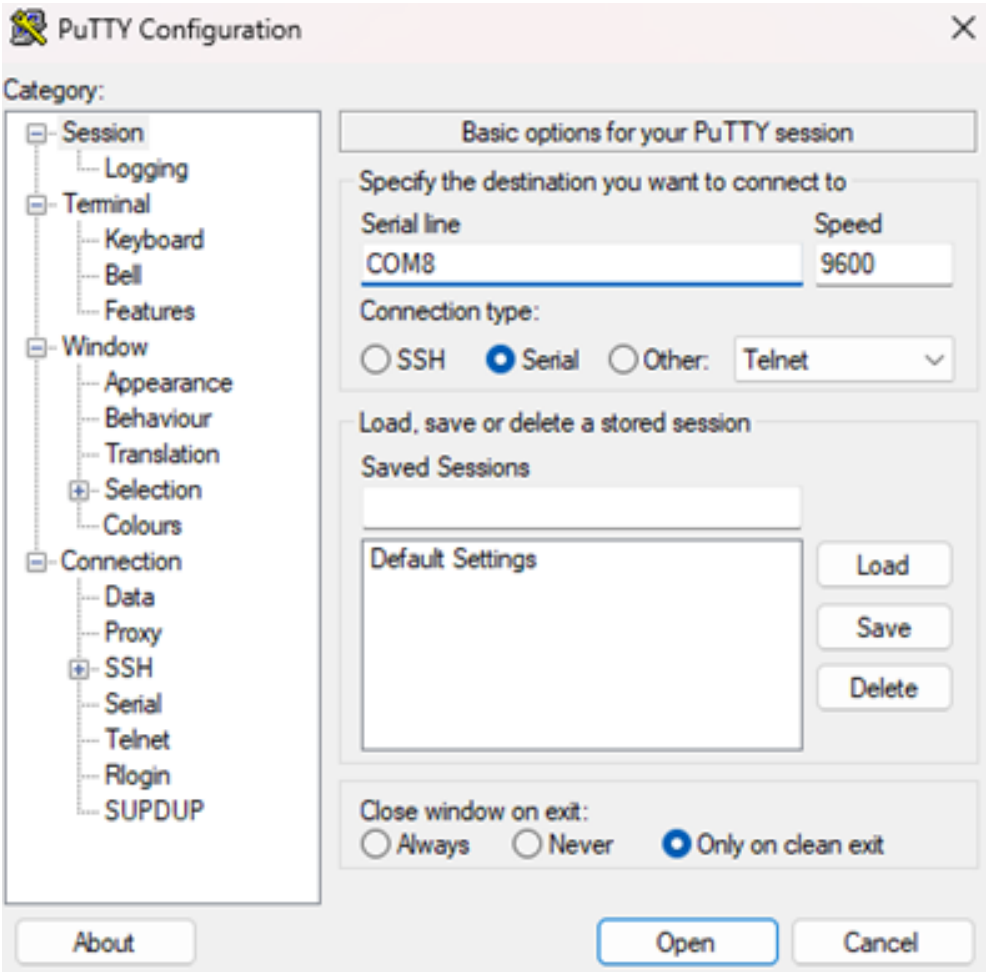


FIGURE 22

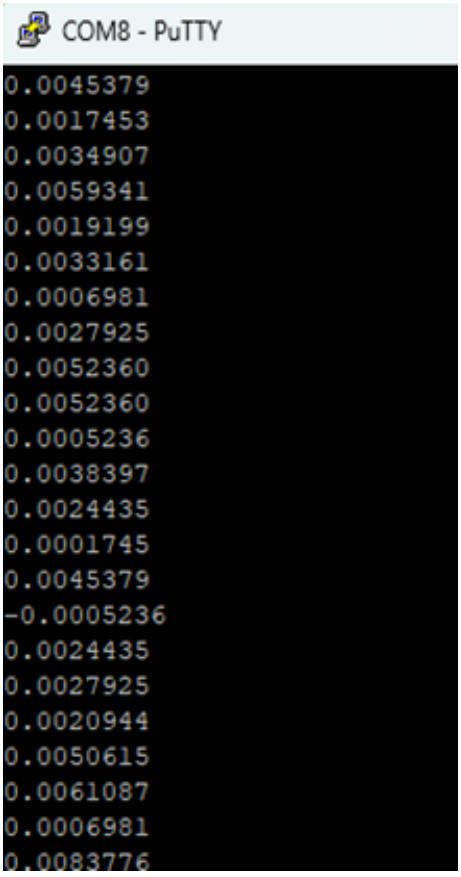
Dans notre code Arduino, nous avons inclus la bibliothèque "SoftwareSerial.h" pour prendre en charge la communication série avec le module HC-05. Nous avons écrit un code simple qui réagit à la réception de lettres spécifiques via la connexion Bluetooth.

Par exemple, lorsque nous envoyons la lettre 'a', une LED s'allume, et lorsque nous envoyons la lettre 'b', la LED s'éteint. Ces résultats démontrent avec succès l'établissement d'une connexion entre le module HC-05 et notre ordinateur.

FIGURE 23

On modifie notre code pour que l'on envoie les données du gyroscope.

Voilà notre moniteur série sur PuTTY qui nous affiche les vitesses angulaires qui proviennent du gyroscope.



Nous avons progressé dans notre projet en plaçant l'ensemble comprenant la carte Arduino, le gyroscope ADXRS450, le module HC-05 et la pile sur l'objet tournant. Cette étape nous permet d'obtenir des mesures de vitesse angulaire en temps réel pendant la rotation de l'objet.

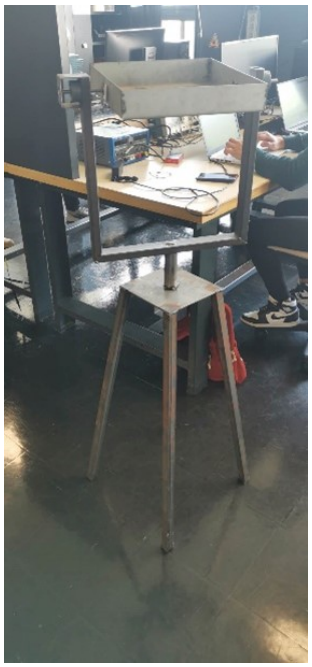


FIGURE 24

FIGURE 25

Pour commencer, nous avons effectué quelques modifications sur notre code initial. Nous avons redéfini les broches RX et TX pour le module HC-05, et nous avons remplacé les appels à Serial.println() par mySerial.println() afin d'afficher les valeurs de vitesse angulaire du gyroscope sur le moniteur PuTTY via la communication série avec le module HC-05. Nous avons ajouté la ligne mySerial.begin(9600) dans la fonction setup() pour initialiser la communication série avec le module.

Une fois le code téléversé sur la carte Arduino, nous avons déconnecté la carte de l'ordinateur et l'avons alimentée avec la pile que nous avons fabriquée. Nous avons ensuite fixé l'ensemble sur l'objet tournant et l'avons laissé immobile.

On obtient ceci :

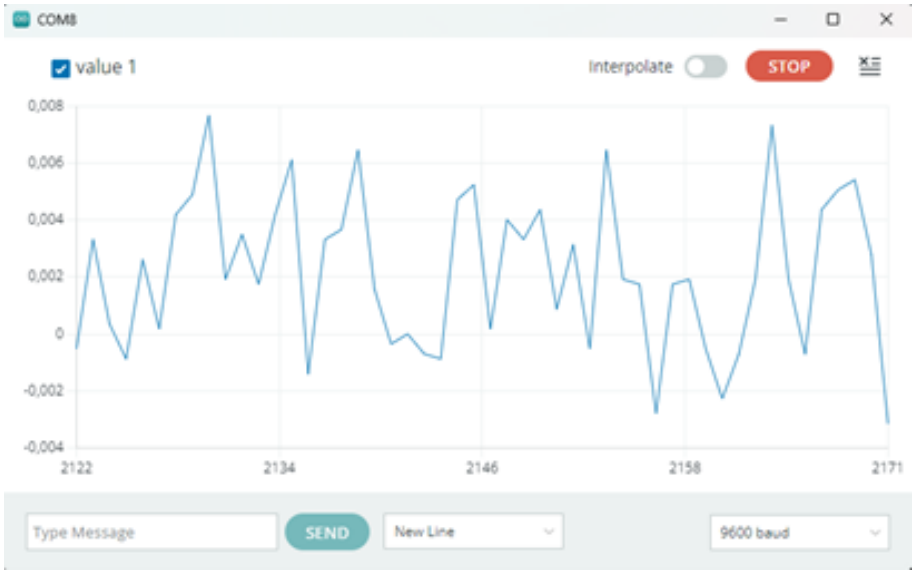


FIGURE 26

Après nous l'avons fait tourner, nous avons observé que le gyroscope détectait cette augmentation de vitesse angulaire. Une fois que la vitesse de rotation a atteint un certain seuil, nous avons observé une décélération.

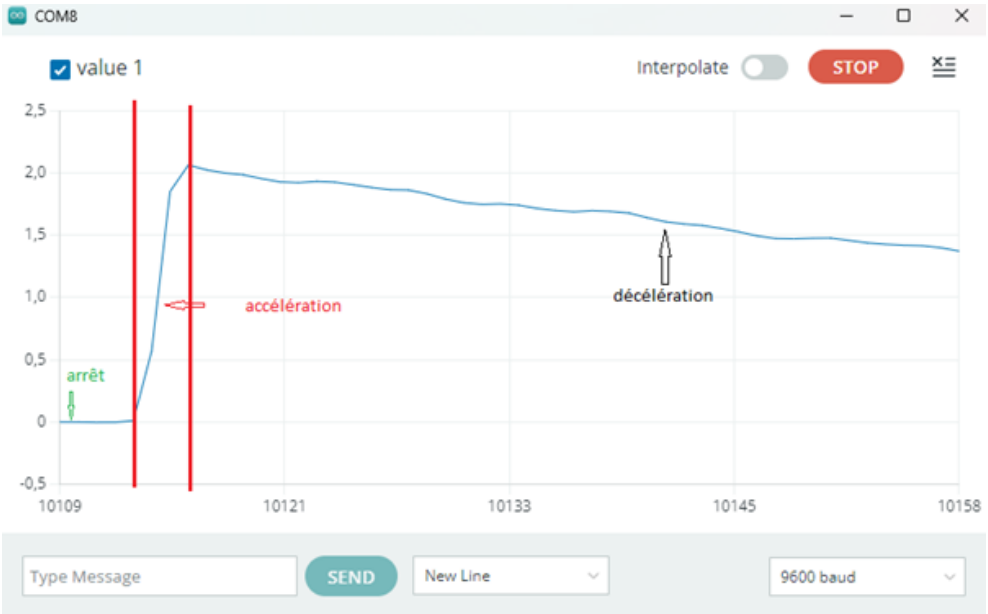


FIGURE 27

Ces observations nous ont permis de constater que le gyroscope ADXRS450 était sensible aux changements de vitesse de rotation et était capable de détecter à la fois l'accélération et la décélération de l'objet.

22

Nous avons effectué un zoom pour mieux visualiser ralentissement de la vitesse au cours du temps en enlevant la moyenne glissante car nous voyons mal la décélération, vue que la MG est la somme des 100 premiers termes.

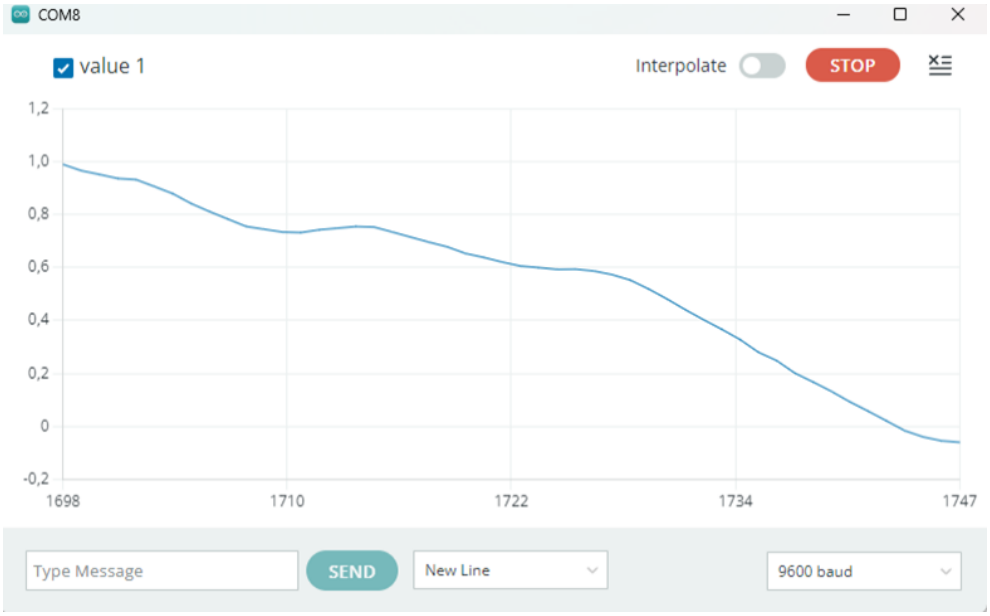


FIGURE 28

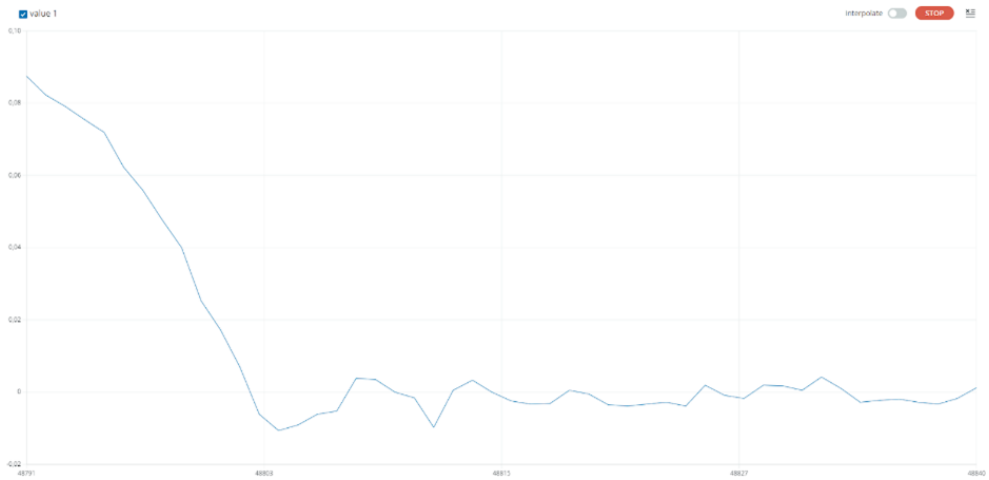


FIGURE 29

La vitesse de rotation minimum est 0,01 rad/s après on a principalement du bruit.

23

7 Conclusion

En conclusion, ce projet nous a permis de réaliser plusieurs étapes clés dans l'utilisation et la programmation du gyroscope ADXRS450 avec un microcontrôleur Arduino. Nous avons réussi à établir une connexion SPI entre le gyroscope et l'Arduino, à lire les données de vitesse angulaire et à les afficher sur le moniteur série. De plus, nous avons intégré un module Bluetooth HC-05 pour permettre une communication sans fil avec un ordinateur. Les résultats obtenus sont en accord avec les spécifications techniques du gyroscope ADXRS450, et nous avons pu observer le comportement du gyroscope en fonction de la vitesse de rotation appliquée. Nous avons constaté que le gyroscope réagissait de manière appropriée aux changements de vitesse, ce qui démontre son efficacité et sa précision.
Pour évaluer et vérifier la réalisation du projet par rapport aux objectifs fixés, nous avons fourni Ce tableau de critères fournit ce qui permettra d'assurer la qualité et la conformité du résultat final.

N°	Fonction/Contraintes	Critère	Niveau d'exigence	Moyen de contrôle
FP1	Mesurer la Vitesse angulaire	Le plus précis possible	F0	Gyroscope ADXRS450
FP2	Etre autonome	Fonctionne à l'aide d'une pile	F3	Pile
FP3	Envoyer des valeurs sans fil	Envoie continue de la vitesse angulaire du capteur	F2	Module bluetooth HC05
FC1	Respecter les consignes	PCB obligatoire	F0	Lire les consignes
FC2	Précision	Précision de mesure	F0	moniteur série
FC3	Sensibilité aux vibrations	Réduction du bruit	F1	Filtre passe-bas, moyenne glissante
FC4	Dérive	Stabilité, Répétabilité	F1	Calibration
FC5	Respecter les échéances	6 Juin 2023	F0	Calendrier

FIGURE 30

24