Rapport de Projet

Main Robotisée

Étudiants : Chériaux Ludovic

Gomes-Vicheney Renaud

Professeur référent : Mme Maillefert

Projet dans le cadre de la licence professionnelle MECSE parcours SESAM

UNIVERSITÉ PARIS SUD | IUT DE CACHAN

université PARIS-SACLAY

Sommaire

Notre équipe !

I – Présentation

Projet Main Robotisée :

Ce projet date de 2016 et s'améliore au cours des années ;

Le projet nous a été donné selon ce synoptique :

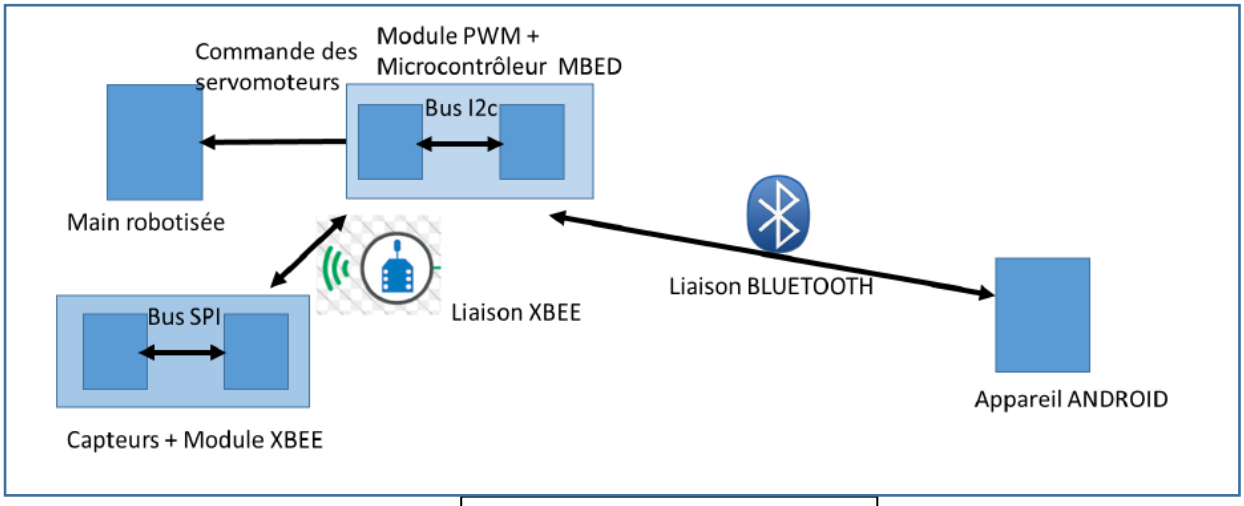


Figure 1 : Synoptique du circuit

Cependant, La supervision par appareil Android a été retirée du projet en raison de la complexité et du manque de temps.

1.1) Principe du projet

Il s'agit en fait d'un gant où des capteurs (accéléromètres) sont collés à chaque phalange, ces capteurs envoient, en fonction de la position des doigts des données et la main robotisée est censée imiter de la manière la plus fidèle possible les positions de chaque doigt de l'utilisateur à l'aide de servomoteurs.

Pour ce faire, trois protocoles interviennent dans notre projet :

- Le bus SPI à travers les capteurs
- Le Xbee car les données transitent d'une carte compatible mbed à une autre en liaison radio
- Le PWM avec nos servomoteurs utilisés dans la main robotisée

1.2) Notre Objectif

- L'objectif principal est de rendre fonctionnel deux nouveaux gants commencés par des étudiants de l'année dernière qui n'ont pas été finalisés.
- Un programme fonctionnel pour piloter la main robotisée avec ces deux gants
- Bonus : Adapter ce système à une autre utilisation de la main robotisée (contrôle de LED par exemple)

Nous nous sommes basés sur ce planning pour appréhender ce projet :

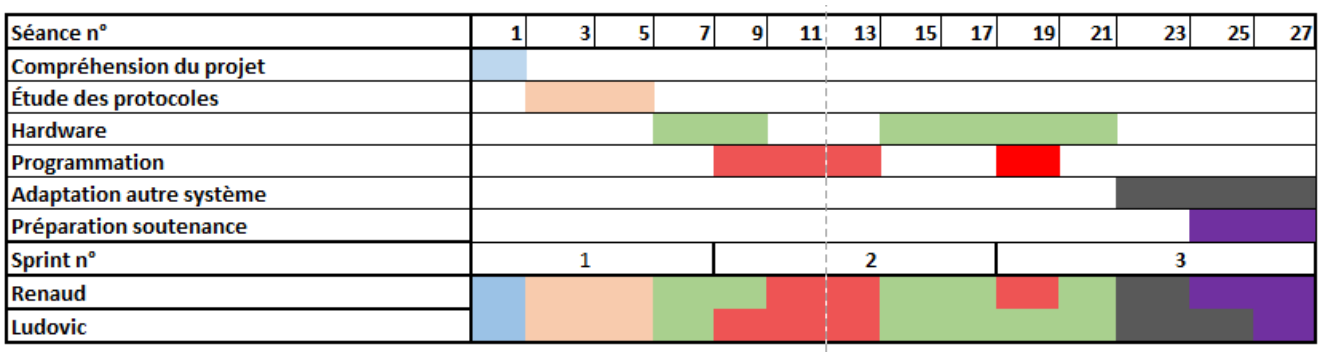


Figure n°2 : Planning du projet

Nous pouvons voir sur ce planning comment nous étions organisés tout au long de ce projet, certaines parties ont été faites en commun et d'autres travaux ont été effectués individuellement, nous voyons également la répartition des tâches en bas du diagramme ci-dessus.

Afin de mener à bien ce projet, nous nous sommes basés sur un gant prototype existant notamment au niveau de la programmation :

Figure n°3 : Prototype initial

II – Hardware

2.1) Spécifications techniques

Dans un premier temps, nous avons fait un schéma de câblage sur le papier afin de repérer toute les liaisons sur le câblage et tout particulièrement pour la partie émission, le hardware concernant la réception étant déjà fonctionnel.

Nous avons fait ce schéma mettant en avant chacun des composants intervenants dans le système :

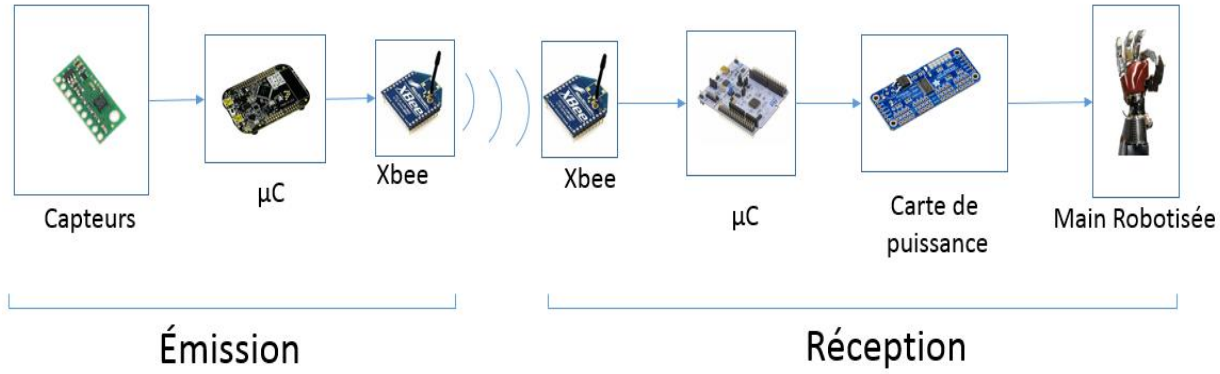


Figure n°4 : Synoptique du projet n°2

Nous nous sommes également penchés sur chacun des protocoles intervenants dans le projet cités en présentation pour connaître la raison du choix de ces composants et les principales caractéristiques :

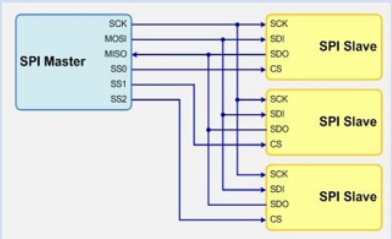
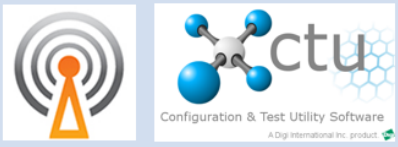
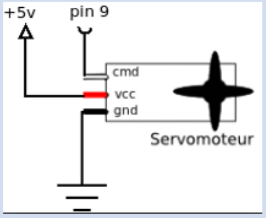
SPI	XBEE	PWM
Bus de données synchrone  - Le maître génère l'horloge - L'esclave répond aux requêtes du maître	Mode de communication sans fil  - Fréquence porteuse: 2,4GHz - Faible consommation et sécurité fiable - Peu couteux et fonctionnel pour notre projet	Synthétise les signaux analogiques  - Changement du rapport cyclique en fonction de la commande

Figure n°5 : Protocoles électroniques utilisés

2.2) Partie émission

Cette partie a constitué une partie non négligeable du projet ;

En effet nous nous sommes heurtés à un gant non fonctionnel dû à de nombreux faux contacts.

Nous avons éprouvé quelques difficultés à résoudre tous les contacts, heureusement, nous avions à notre disposition un fer à souder avec une panne fine nous permettant de souder avec précision.

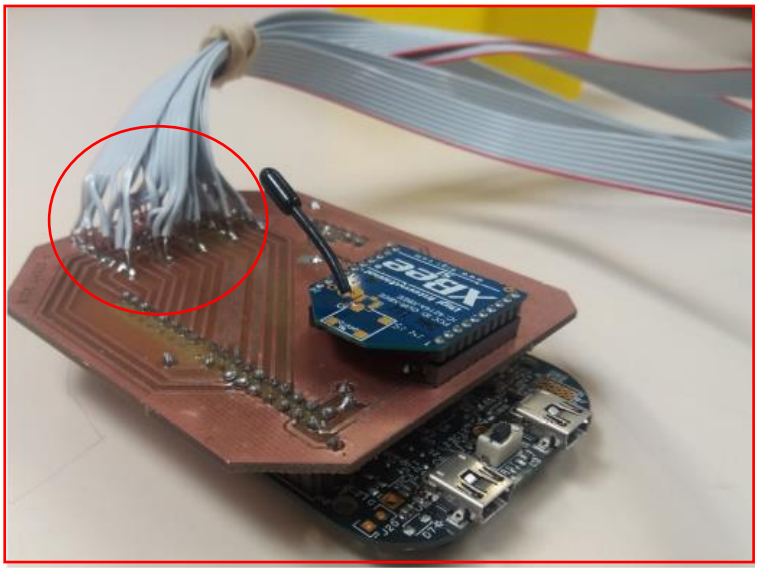


Figure n°6 : Partie technique soudures

Nous avons grâce au matériel à disposition des appareils nous permettant de valider le bon transfert des données ainsi que la continuité entre les différentes pistes, nous avons notamment eu accès à un oscilloscope et des multimètres.

Figure n°7 : outils maintenance

Nous en sommes arrivés à deux gants fonctionnels respectant le schéma de câblage suivant :

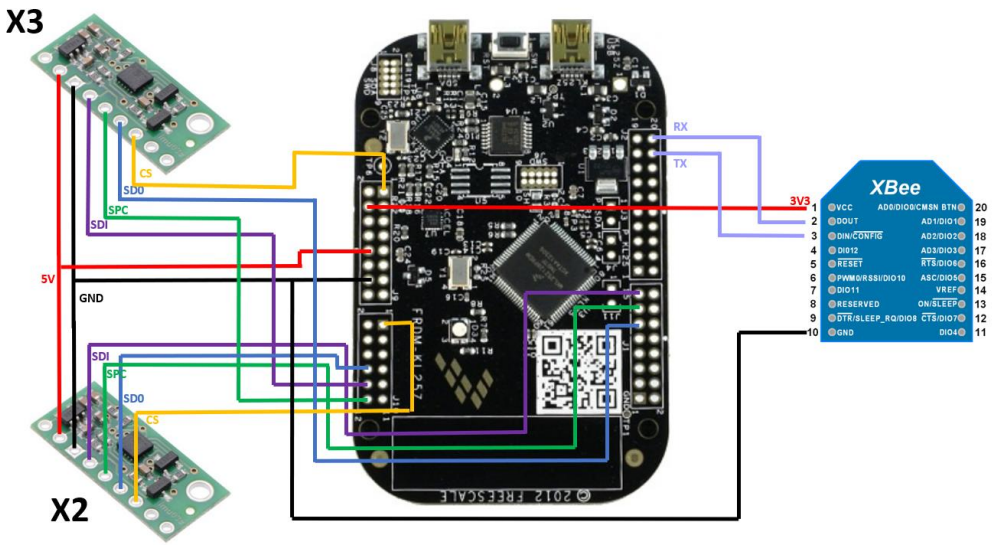


Figure n°8 : Schéma de câblage émission

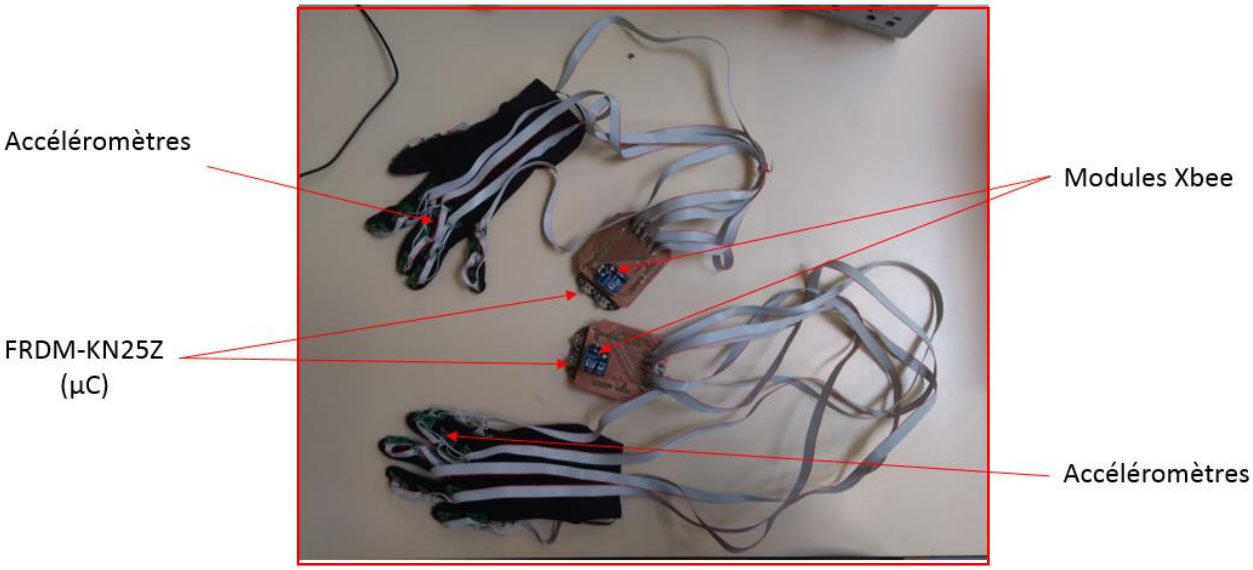


Figure n°8 : Les deux gants fonctionnels

Ci-dessus les deux gants fonctionnels câblés à nos PCB.

Nous avons effectué la validation du hardware notamment avec un programme nous disant si tous les capteurs étaient reconnus.

```
void Capteur_present(void){// verification presence capteurs
int present;
    for (int i=0;i < 6;i++){
        present = Read_Register(i, 0x0F);  // registre "Who am I" de l'accéléromètre
        if( present == 0b01001001){      // valeur contenue par defaut 0x49
          pc.printf("capteur2 %X = OK\r\n", i);
        } else{
           pc.printf("capteur2 %X = Error\r\n", i);
        }
    }
    for (int i=6;i < nbCapteur;i++){
        present = Read_Register(i, 0x0F);
        if( present == 0b01001001){
           pc.printf("capteur_1 %X = OK\r\n", i);
        } else{
           pc.printf("capteur_1 %X = Error\r\n", i);
        }
    }
}
```

Figure n°9 : Extrait code validation capteurs

Après avoir compilé sur la carte ce programme avec Mbed, il nous restait qu'à vérifier sur l'hyperterminal du PC si tous les accéléromètres étaient présents

2.3) Partie réception

Concernant le Hardware, la partie réception était déjà fonctionnelle suivant le câblage ci-dessous :

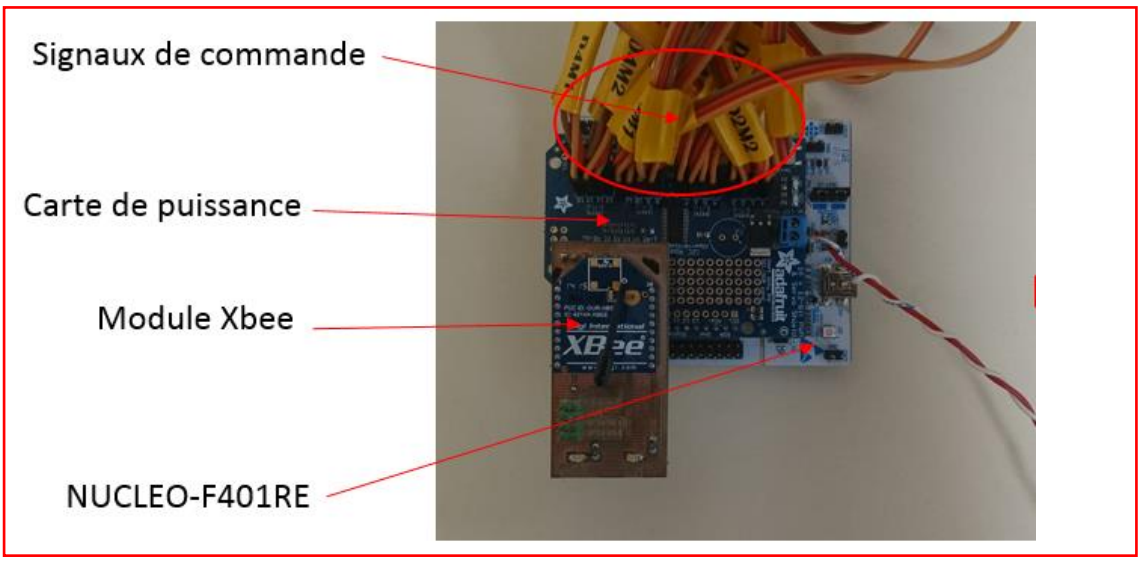


Figure n°10 : Hardware réception

A noter que nous avons utilisé une alimentation de puissance pour faire fonctionner la main étant donné que celle-ci consomme jusqu'à 6A.

III – Programmation

3.1) Principe

Notre solution pour manipuler la main avec le gant a été la suivante :

Nous savons que le capteur envoie une donnée analogique, la méthode la plus efficace pour nous était de mettre en place des seuils ;

Selon le seuil de la donnée transmise, un caractère propre à ce seuil est envoyé.

Une position est attribuée aux servomoteurs propre au caractère reçu.

3.2) Émission

```
LSM303D lsm1(capteur2,PTE31);//pouce haut
LSM303D lsm6(capteur2,PTA17);//pouce m
LSM303D lsm11(capteur2,PTA16);//pouce b
```

```
while(1)
 {
    // pouce haut
     if ((accz(lsm1)) < 0.1 ){
        xbee.printf("p");
     }
     //pouce milieu
     if ((accz(lsm1))>0.1 && (accz(lsm1))<0.4){
         xbee.printf("o");
         }
     // pouce bas
     if ((accz(lsm1)) > 0.4  ){
        xbee.printf("q");
     }
```

Figure n°11 : Code émission

Voici un extrait du programme nous montrant comment un doigt est initialisé et comment les données sont envoyées et converties en caractères.

3.3) Réception

Comme expliqué précédemment, caractère est ensuite associé à une position PWM.

```
while(1)
{

       if(xbee.readable()==1) //Si un caractère peut être lu
      {
          reception = xbee.getc(); //la variable reception prend la valeur du caractère
          pc.printf("reception : %c\n", reception);


          if(reception=='p'){           //Si le caractère reçu est égal à p
             PCA_SERVO.set_pwm_pw(1,2200);
             PCA_SERVO.set_pwm_pw(2,2200);
             PCA_SERVO.set_pwm_pw(3,2400);
          }

          if(reception=='o'){           //Si le caractère reçu est égal à o
              //PCA_SERVO.set_pwm_pw(1,1700);
              PCA_SERVO.set_pwm_pw(2,1700);
              //PCA_SERVO.set_pwm_pw(3,1800);
              }

          if(reception=='q'){           //Si le caractère reçu est égal à q
             PCA_SERVO.set_pwm_pw(1,1300);
             PCA_SERVO.set_pwm_pw(2,1300);
             PCA_SERVO.set_pwm_pw(3,1300);
          }
```

Figure n°12 : Code réception

Le code ci-dessus concerne également le pouce.

IV – Tests et Résultats

Les tests sont concluants et les deux nouveaux gants sont fonctionnels tout du moins avec la main robotisée ;

Cependant on décerne quelques problématiques ou points à améliorer :

- L'alimentation ne permet pas de faire bouger un certain nombre de moteurs en même temps (problème venant peut-être de la carte de puissance)
- On s'est limités à seulement trois positions par doigts donc la précision et la fidélité des mouvements ne sont pas optimales.
- La visserie et le support de la main sont relativement instable notamment lorsque la main est en mouvement

V – Adaptation à un autre système

Nous avons décidé pour améliorer le projet, d'adapter celui-ci à un autre projet de la classe, en l'occurrence le projet « Synthèse musicale ».

5.1) Principe

On reprendra le même principe que pour la main robotisée, notamment au niveau programmation où le principe de conversion de caractère est repris.

A noter que nous nous occupons seulement de la partie émission, la partie réception étant gérée par l'autre groupe.

Les données seront envoyées en Bluetooth.

5.2) Hardware

On utilisera le module HC-05 respectant le schéma de câblage suivant :

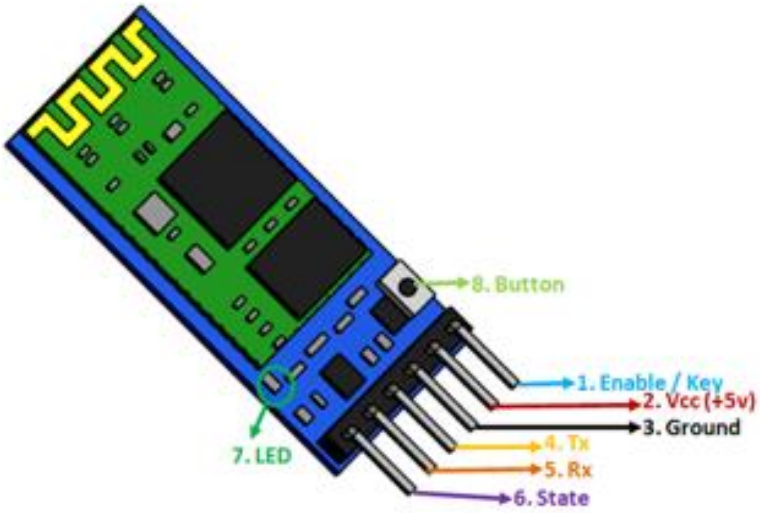


Figure n°13 : HC-05

Ce qui nous donne sur notre système :

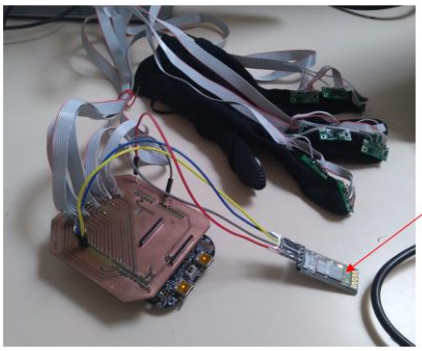


Figure n°14 : Câblage complet

Nous avons adapté comme ceci les notes par rapport à la position des doigts :

57	Do	Pouce
58	Dod	pouce et index
59	re	index
60	red	index et maj
61	mi	majeur
62	fa	annulaire
63	fad	majeur et annulaire
64	sol	Auriculaire
65	sold	Auriculaire et pouce
66	la	annulaire et pouce
67	lad	pouce et majeur
68	si	Index et Auri

```
R
A6000
R
R6200
A6300
R
R6500
R6600
R
R6800
R
R5800
R
A6000
R0
R200
A6300
R6600
R
R6800
R5700
R
R5900
```

Figure n°15 : Attribution note / doigt

Lorsqu'on effectue des tests d'envoi on reçoit des données comme on peut le voir à droite.

5.3) Programmation

Voici les principales syntaxes à retenir pour le code (le système de seuil est toujours utilisé) :

```
LSM303D lsm1(capteur2,PTE31);//pouce haut
LSM303D lsm6(capteur2,PTA17);//pouce m
LSM303D lsm11(capteur2,PTA16);//pouce b
```

```
while(1) {
   // pouce haut DO
     if ((accz(lsm1)) < 0.1 ){
        bt.printf("A5700\r\n");
        pc1.printf("A5700/r/n");
     }

     // pouce bas
     if ((accz(lsm1)) > 0.4  ){
        bt.printf("R5700\r\n");
        pc1.printf("R5700\r\n");
     }

     // pouce et index haut DOd
     if ((accz(lsm1))<0.1 && (acc(lsm2))>0.5){
        bt.printf("A5800\r\n");

     }
```

Figure n°16 : Code émission Bluetooth

5.4) Conclusion

Cette adaptation est largement améliorable mais encourageante ;

Effectivement, nous arrivons à envoyer de bonnes données au récepteur et les sons que l'on entend correspondent à la position des doigts.

Cependant le débit du capteur n'étant pas adapté (il envoie des valeurs en continu), nous n'arrivons pas à envoyer tout ce que l'on veut.

Ce projet pourrait être amélioré dans le futur en sachant comment adapter le débit avec les capteurs.

Lien Github :

https://github.com/CheriauxLudovic/ProjetMainRobot?fbclid=IwAR1SmHhF4aFPCKonnJOD4G0li4yUUMzaMsUJIYmqt2NZdnuIyQeUUn26FT8