**Rapport de Séance du 15 décembre**

**Objectifs de la séance :**

1. Déterminer le matériel nécessaire pour la séance
2. Effectuer les branchements de l’accéléromètre
3. Récupérer et traiter les données de l’accéléromètre (partie code)
4. Illustrer les mouvements de l’accéléromètre

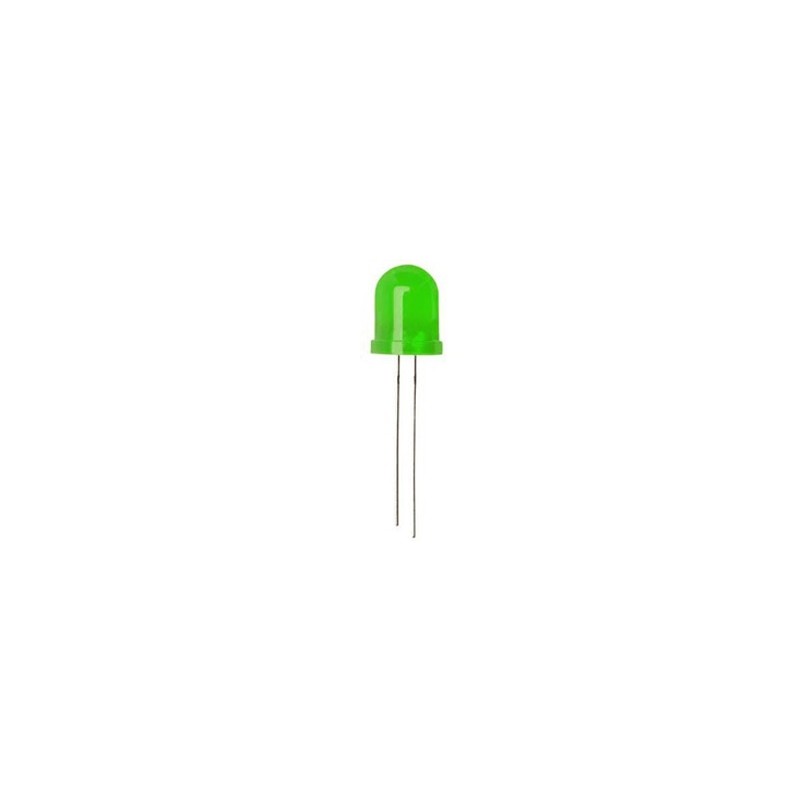
**Déroulement de la séance :**

1. Définition du matériel nécessaire pour la séance

*Pour nous lancer correctement dans la séance, j’ai tout d’abord effectué un bref inventaire du matériel nécessaire au déroulement de la séance du jour. La séance du jour consiste à récupérer puis traiter les données acquises par l’accéléromètre, c’est pourquoi j’aurai besoin de :*

* *1 Carte Arduino Uno*
* *1 Accéléromètre (le* GY-521 qui est basé sur le MEMS MPU-6050)
* *4 LED (2 oranges, 2 vertes)*
* 1 Plaque d’essai petite





1. Branchements de l’accéléromètre :

*Pour récupérer les données acquises par l’accéléromètre il faut tout d’abord brancher correctement ce dernier. Par chance un cours complet qui explique comment se servir de l’accéléromètre est à notre disposition. C’est pourquoi il m’a été facile de brancher le module selon le montage suivant :*

Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Description générée automatiquement

*Le montage est simple puisqu'il n’utilise que l’alimentation + 5 V de l’Arduino et les connections SCL et SDA du bus I2C :*

* *SDA pour Serial Data (I/O n° A5)*
* *SCL pour Serial Clock (I/O n° A4)*

1. Récupération et traitement des données de l’accéléromètre :

*Maintenant que notre module GY-521 est correctement branché, il faut coder le nécessaire pour pouvoir récupérer les données enregistrer par l’accéléromètre.*

*En m’informant sur le cours disponible je suis arriver à réaliser le code suivant :*

#include "Wire.h"

const int MPU\_ADDR = 0x68;

int16\_t accelerometer\_x, accelerometer\_y, accelerometer\_z;

char tmp\_str[7];

char\* convert\_int16\_to\_str(int16\_t i) {

  sprintf(tmp\_str, "%6d", i);

  return tmp\_str;

}

void setup() {

  Serial.begin(9600);

  Wire.begin();

  Wire.beginTransmission(MPU\_ADDR);

  Wire.write(0x6B);

  Wire.write(0);

  Wire.endTransmission(true);

}

void loop() {

  Wire.beginTransmission(MPU\_ADDR);

  Wire.write(0x3B);

  Wire.endTransmission(false);

  Wire.requestFrom(MPU\_ADDR, 7\*2, true);

  accelerometer\_x = Wire.read()<<8 | Wire.read();

  accelerometer\_y = Wire.read()<<8 | Wire.read();

  accelerometer\_z = Wire.read()<<8 | Wire.read();

  Serial.print("aX = "); Serial.print(convert\_int16\_to\_str(accelerometer\_x));

  Serial.print(" | aY = "); Serial.print(convert\_int16\_to\_str(accelerometer\_y));

  Serial.print(" | aZ = "); Serial.print(convert\_int16\_to\_str(accelerometer\_z));

delay(1000);

}

***Vous trouverez la version finale, détaillée et commentée du code dans le fichier « accelerometre.ino » joint au rapport de séance !***

*Ce code me permet donc d’afficher sur le moniteur série de l’IDE toutes les données enregistrées par l’accéléromètre, ce qui représentera par la suite les mouvement de la main :*

aX = *Valeur* | aY = *Valeur* | aZ = *Valeur*

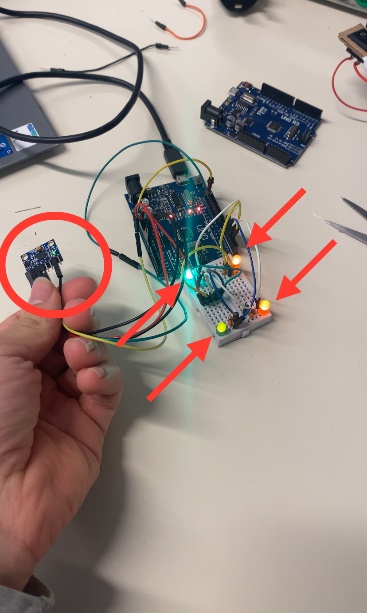
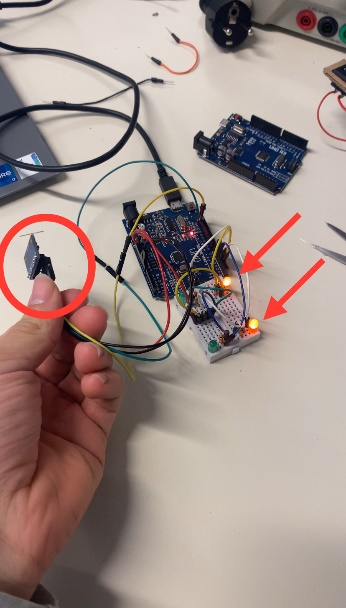
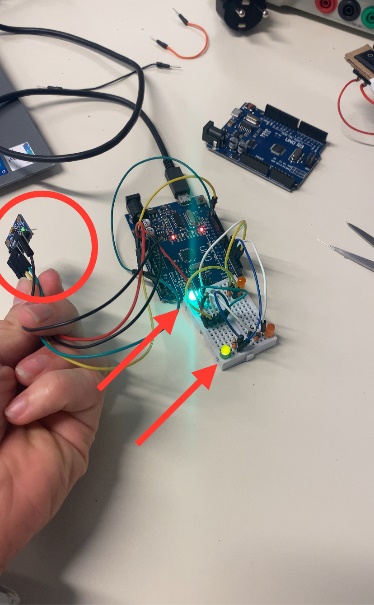
1. Illustrer les mouvements de l’accéléromètre :

*Pour finir la séance j’ai souhaité illustrer les mouvements de l’accéléromètre de manière plus visuel que l’affichage décrit précédemment. C’est pour quoi j’ai réalisé un montage avec 4 LED. Ces LED s’allument en fonction de la position de l’accéléromètre.*

*Voici quelques photos pour illustrer :*

*Accéléromètre à plat Accéléromètre penche à gauche Accéléromètre penche à droite*

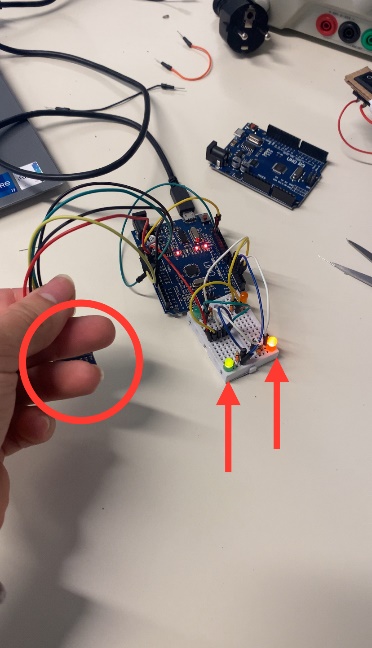
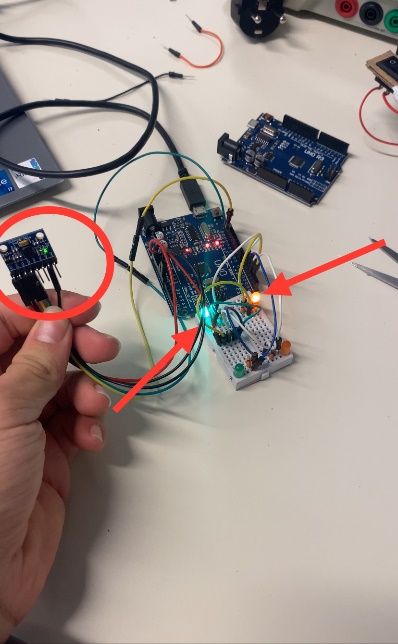
*=> les 4 LED s’allument => les 2 LED gauches s’allument => les 2 LED droites s’allument*

**

*Accéléromètre penche devant Accéléromètre penche derrière Accéléromètre à plat*

*=> les 4 LED arrière s’allument => les 2 LED avant s’allument => les 4 LED s’allument*

*Une image contenant Ingénierie électronique, câble, fils électriques, Appareils électroniques

Description générée automatiquement*

*Pour parvenir à ce résultat j’ai ajouté quelques ligne de code simple à mon code originale :*

// On indique sur quelle I/O on a branché les LED :

const int LED\_orange\_1=2;

const int LED\_orange\_2=3;

const int LED\_verte\_1=4;

const int LED\_verte\_2=5;

void setup() { // On ajoute dans le setup le code suivant :

// On met les I/O en sortie :

pinMode(LED\_orange\_1,OUTPUT);

  pinMode(LED\_orange\_2,OUTPUT);

  pinMode(LED\_verte\_1,OUTPUT);

  pinMode(LED\_verte\_2,OUTPUT);

// On met allumme les 4 LED au début :

  digitalWrite(LED\_orange\_1,LOW);

  digitalWrite(LED\_orange\_2,LOW);

  digitalWrite(LED\_verte\_1,LOW);

  digitalWrite(LED\_verte\_2,LOW);

}

void loop() {// On ajoute dans le loop le code suivant :

// Si l’accéléromètre penche à gauche :

if (accelerometer\_x < 1000 && accelerometer\_y < - 8000) {

  digitalWrite(LED\_orange\_1, HIGH) ;

  digitalWrite (LED\_orange\_2, HIGH);

  digitalWrite (LED\_verte\_1,LOW) ;

  digitalWrite (LED\_verte\_2, LOW);

}

// Si l’accéléromètre penche à droite :

else if (accelerometer\_x < 1000 && accelerometer\_y > 8000) {

  digitalWrite (LED\_orange\_1, LOW);

  digitalWrite (LED\_orange\_2, LOW) ;

  digitalWrite (LED\_verte\_1, HIGH);

  digitalWrite (LED\_verte\_2, HIGH);

}

// Si l’accéléromètre penche à devant :

else if (accelerometer\_x > 8000 && accelerometer\_y < 1000) {

  digitalWrite (LED\_orange\_1, LOW);

  digitalWrite (LED\_orange\_2, HIGH) ;

  digitalWrite (LED\_verte\_1, HIGH) ;

  digitalWrite (LED\_verte\_2, LOW);

}

// Si l’accéléromètre penche à derrière :

else if (accelerometer\_x < -8000 && accelerometer\_y < 1000) {

  digitalWrite (LED\_orange\_1, HIGH) ;

  digitalWrite (LED\_orange\_2, LOW);

  digitalWrite (LED\_verte\_1, LOW);

  digitalWrite (LED\_verte\_2, HIGH);

}

else {

  digitalWrite (LED\_orange\_1, LOW); digitalWrite (LED\_orange\_2, LOW);

digitalWrite (LED\_verte\_1, LOW); digitalWrite (LED\_verte\_2, LOW); }}