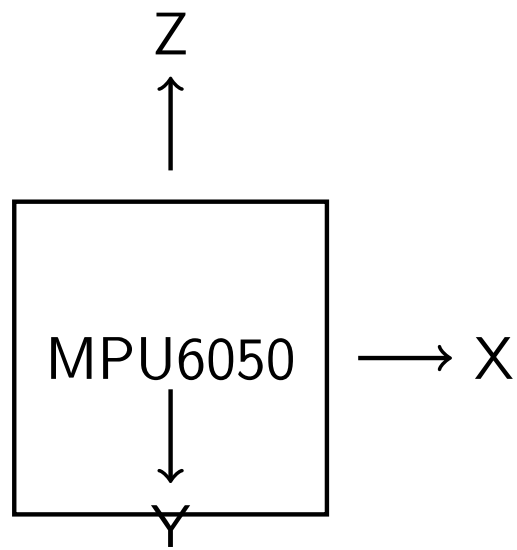
Utilisation du capteur MPU6050 avec ESP32

Houssem Lahmer

Plan du cours

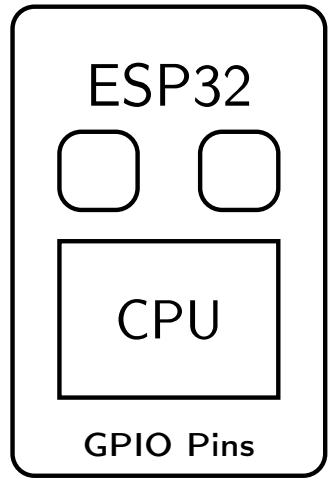
Introduction au MPU6050

- Capteur de mouvement 6 axes (6-DoF)
- Composants intégrés :
  - Accéléromètre 3 axes (X, Y, Z)
  - Gyroscope 3 axes (X, Y, Z)
- Communication via protocole I^2C
- Utilisations : drones, robotique, détection de mouvements, stabilisation d'image...



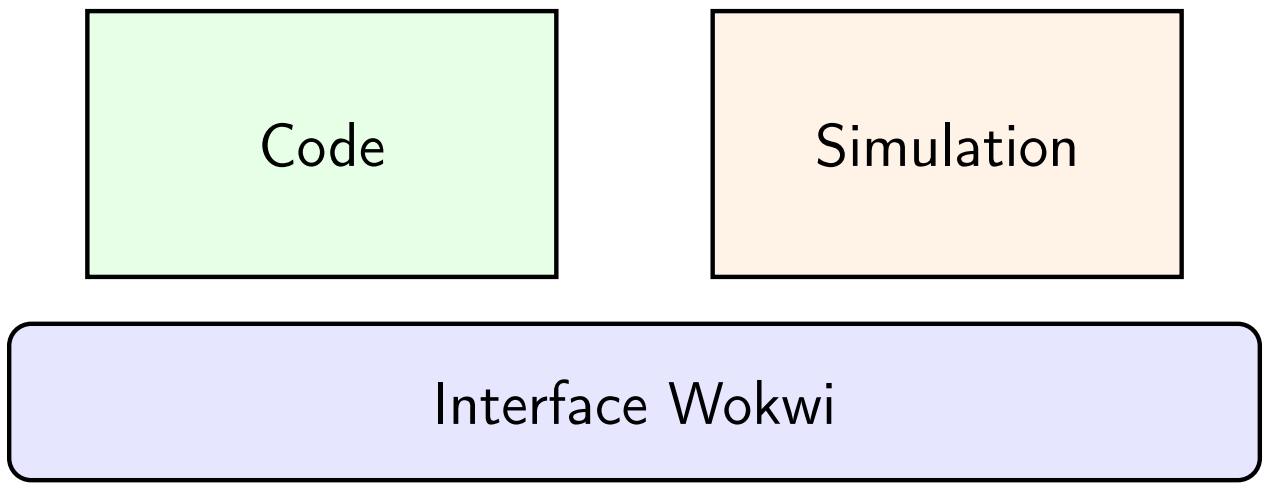
Introduction à l'ESP32

- Microcontrôleur puissant et polyvalent
- Processeur dual-core 32-bit
- Wi-Fi et Bluetooth intégrés
- Nombreuses entrées/sorties
- Support de multiples interfaces (I^2C, SPI, UART...)
- Parfait pour les projets IoT



Présentation de Wokwi

- Simulateur en ligne pour microcontrôleurs
- Permet de tester des circuits et du code sans matériel physique
- Support pour Arduino, ESP32, ESP8266, etc.
- Interface graphique intuitive
- Possibilité de simuler divers capteurs et actuateurs
- Idéal pour l'apprentissage et le prototypage



Fonctionnement du MPU6050

Accéléromètre

- Mesure l'accélération linéaire selon 3 axes
- Utilise des structures micromécaniques (MEMS)
- Mesure l'accélération gravitationnelle (permet de déterminer l'orientation)
- Unité standard : g ($1g = 9.81\ m/s^2$)

Gyroscope

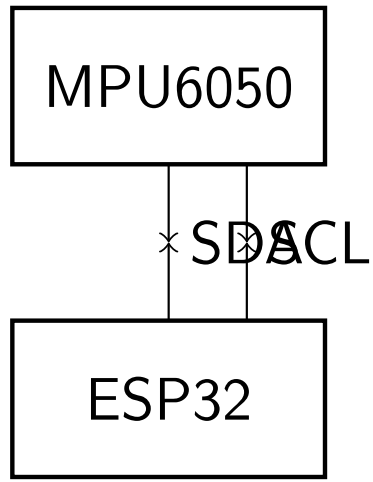
- Mesure la vitesse angulaire selon 3 axes
- Détecte les rotations et changements d'orientation
- Unité standard : degrés par seconde ($^\circ/s$)

DMP (Digital Motion Processor)

- Processeur embarqué pour traitement des données
- Calculs d'orientation (quaternions, angles d'Euler...)
- Allège la charge du microcontrôleur principal

Communication I²C

- Protocole de communication série
- Utilise 2 lignes :
  - SDA (Serial Data) - Données
  - SCL (Serial Clock) - Horloge
- Mode maître-esclave
- Adresse I²C du MPU6050 : 0x68 (par défaut)
- Communication bidirectionnelle
- Vitesse standard : 100-400 kHz



Bibliothèques nécessaires

Pour utiliser le MPU6050 avec l'ESP32, nous avons plusieurs options :

1. **Adafruit MPU6050** - Simple et bien documentée
   - Installation : Gestionnaire de bibliothèques → "Adafruit MPU6050"
   - Dépendance : Adafruit Unified Sensor
2. **I2Cdevlib MPU6050** - Plus complète, accès à toutes les fonctionnalités
   - Téléchargement : GitHub "jrowberg/i2cdevlib"
   - Installation manuelle nécessaire
3. **ESP32 Wire** - Bibliothèque native pour la communication I^2C
   - Déjà incluse dans l'environnement ESP32

Dans ce cours, nous utiliserons principalement Adafruit MPU6050 pour sa simplicité.

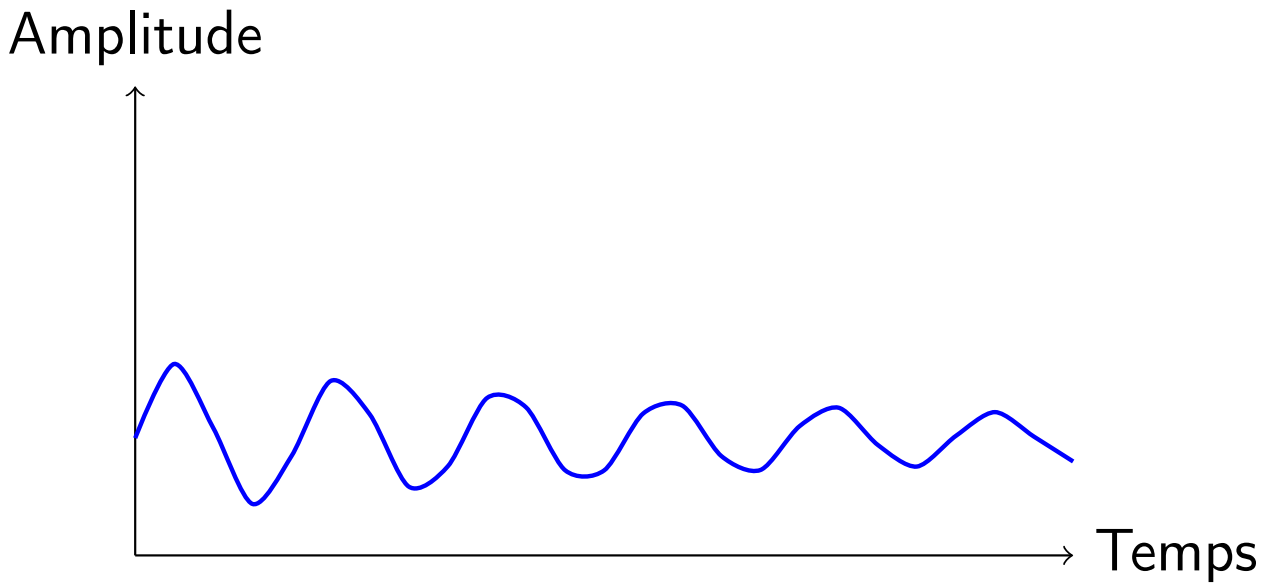
Détection de mouvements

- Détection de secousses, chutes ou mouvements rapides
- Surveillance de mouvements spécifiques
- Calcul de l'accélération totale : $|a| = \sqrt{a_x^2 + a_y^2 + a_z^2}$

```
1 // Calcul de l'acclration totale float accelMagnitude =
  sqrt(pow(a.acceleration.x, 2) + pow(a.acceleration.y, 2) +
  pow(a.acceleration.z, 2))
2
3 // Seuil de d;tection (par exemple 12 m/s      = 1.2g) if
  (accelMagnitude > 12)  return true // Mouvement d~;tect  return false
```

Surveillance de vibrations

- Mesure de l'amplitude et de la fréquence des vibrations
- Applications :
  - Surveillance de machines industrielles
  - Détection de tremblements
  - Analyse de structures
- Nécessite un échantillonnage à haute fréquence
- Peut être associé à une analyse FFT pour l'analyse fréquentielle



Interface avec écran OLED

Sur Wokwi, nous pouvons ajouter un écran OLED I2C pour visualiser les données :

```
#include <Adafruit_MPU6050.h>
#include <Adafruit_SSD1306.h>

Adafruit_MPU6050 mpu;
Adafruit_SSD1306 display(128, 64, &Wire, -1);

void setup() {
  Wire.begin(21, 22);

  // Initialize MPU6050
  mpu.begin();

  // Initialize OLED
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
}

void loop() {
```

Problèmes courants et solutions

1. **Dérive du gyroscope**
   - Problème : Accumulation d'erreurs dans le temps
   - Solutions :
     - Filtre complémentaire
     - Filtre de Kalman
     - Étalonnage régulier
2. **Bruit dans les mesures**
   - Problème : Fluctuations aléatoires
   - Solutions :
     - Moyenne mobile
     - Filtre passe-bas
     - Augmenter le temps d'échantillonnage
3. **Étalonnage incorrect**
   - Problème : Offset dans les mesures
   - Solutions :
     - Procédure d'étalonnage au démarrage
     - Stockage des valeurs d'étalonnage en EEPROM

Projet : Niveau à bulle numérique

Objectif : Créer un niveau à bulle numérique utilisant le MPU6050 et l'ESP32

Fonctionnalités :

- Affichage de l'inclinaison sur deux axes
- Indication visuelle de l'horizontalité
- Alarme sonore quand le niveau est parfait
- Calibration utilisateur

Composants sur Wokwi :

- ESP32
- MPU6050
- Écran OLED SSD1306
- Buzzer (optionnel)
- LED RGB (optionnel)

Structure du projet

```
// Biblioth ques
#include <Wire.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_SSD1306.h>

// Constantes et d finitions
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET -1
#define SCREEN_ADDRESS 0x3C
#define BUZZER_PIN 25
#define BUTTON_PIN 15
#define LEVEL_THRESHOLD 2.0  // Degr s

// Objets
Adafruit_MPU6050 mpu;
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT,
                         &Wire, OLED_RESET);

// Variables
float angleX = 0, angleY = 0;
```

Configuration et calibration

```
void setup() {
  Serial.begin(115200);
  Wire.begin(21, 22);  // SDA, SCL
  pinMode(BUZZER_PIN, OUTPUT);
  pinMode(BUTTON_PIN, INPUT_PULLUP);

  // Initialisation du MPU6050
  if (!mpu.begin()) {
    Serial.println("Failed to find MPU6050 chip");
    while (1) {
      delay(10);
    }
  }
  Serial.println("MPU6050 Found!");

  // Configuration du MPU6050
  mpu.setAccelerometerRange(MPU6050_RANGE_2_G);
  mpu.setGyroRange(MPU6050_RANGE_250_DEG);
  mpu.setFilterBandwidth(MPU6050_BAND_21_HZ);

  // Initialisation de l' cran  OLED
  if (!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
```

Fonction de calibration

```
1  void calibrateSensor() {
2    display.clearDisplay();
3    display.setCursor(0, 0);
4    display.println("Calibrating...");
5    display.println("Keep device flat");
6    display.println("and still");
7    display.display();
8
9    float sumX = 0, sumY = 0;
10   int samples = 100;
11
12   for (int i = 0; i < samples; i++) {
13     sensors_event_t a, g, temp;
14     mpu.getEvent(&a, &g, &temp);
15
16     // Calculer les angles bruts
17     float rawAngleX = atan2(a.acceleration.y,
18                             sqrt(a.acceleration.x * a.acceleration.x
       +
19                                  a.acceleration.z * a.acceleration.z)
       )
20                             * 180.0 / PI;
```

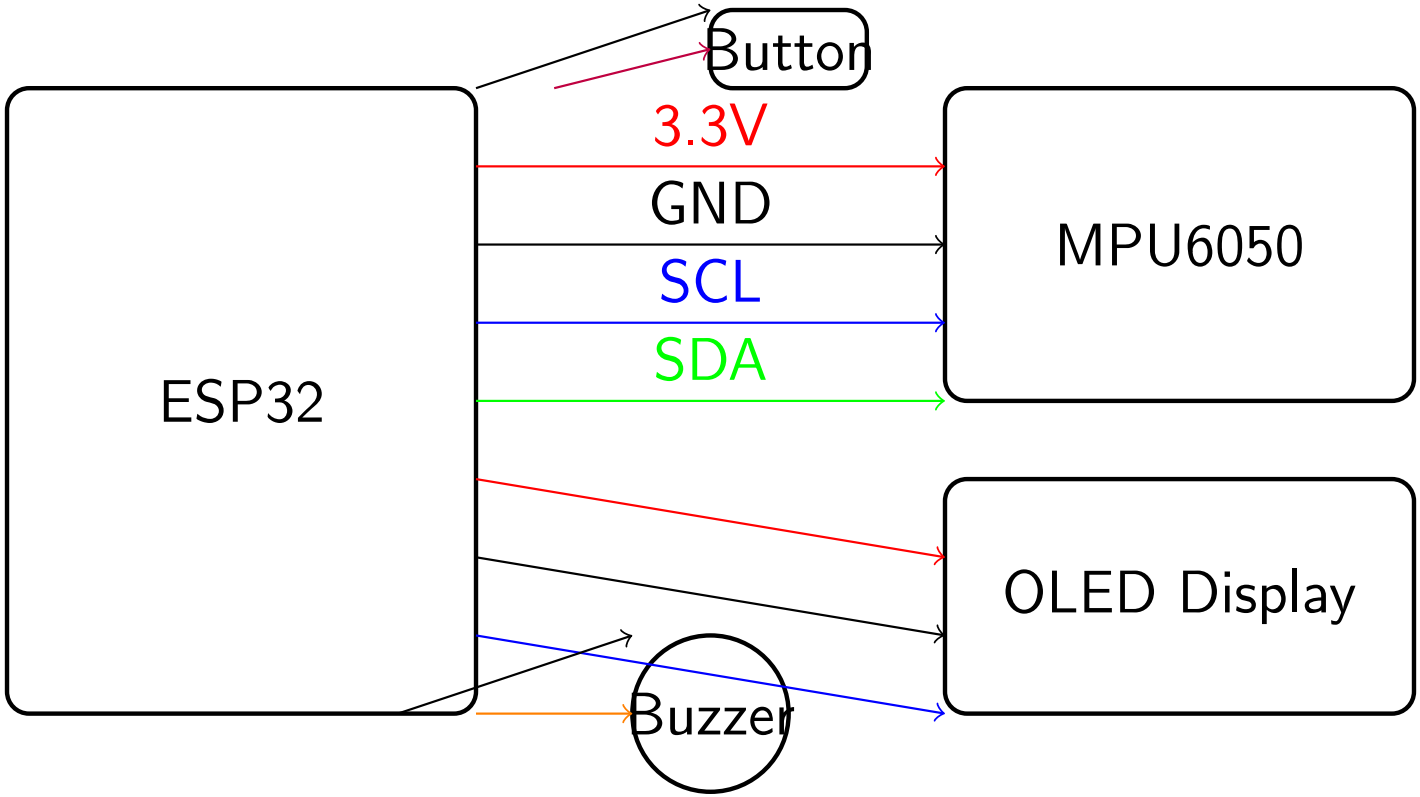
Boucle principale

```
void loop() {
  // V rifier si le bouton de calibration est press
  if (digitalRead(BUTTON_PIN) == LOW) {
    delay(50);  // Debounce
    if (digitalRead(BUTTON_PIN) == LOW) {
      calibrateSensor();
    }
  }

  // Lire les donn es du capteur
  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);

  // Calculer les angles corrig s
  angleX = atan2(a.acceleration.y,
          sqrt(a.acceleration.x * a.acceleration.x +
               a.acceleration.z * a.acceleration.z))
          * 180.0 / PI - accelXoffset;
  angleY = atan2(-a.acceleration.x,
          sqrt(a.acceleration.y * a.acceleration.y +
               a.acceleration.z * a.acceleration.z))
          * 180.0 / PI - accelYoffset;
```

Fonction d'affichage

```
1  void updateDisplay(float x, float y) {
2    display.clearDisplay();
3
4    // Afficher les valeurs num riques
5    display.setCursor(0, 0);
6    display.println("Digital Spirit Level");
7    display.print("X: ");
8    display.print(x, 1);
9    display.println(" deg");
10   display.print("Y: ");
11   display.print(y, 1);
12   display.println(" deg");
13
14   // Dessiner le niveau    bulle
15   int centerX = SCREEN_WIDTH / 2;
16   int centerY = 45;
17   int radius = 15;
18
19   // Cercle ext rieur
20   display.drawCircle(centerX, centerY, radius, SSD1306_WHITE
       );
21
```

Schéma de branchement sur Wokwi



ESP32	Connecté à
GPIO21 (SDA)	MPU6050-SDA et OLED-SDA
GPIO22 (SCL)	MPU6050-SCL et OLED-SCL
GPIO25	Buzzer
GPIO15	Bouton de calibration
3.3V	MPU6050-VCC et OLED-VCC
GND	MPU6050-GND, OLED-GND, Buzzer-GND, Bouton-GND

Exercice 1 : Mesure d'angle simple

Objectif : Afficher les angles d'inclinaison sur le moniteur série
Instructions :

1. Créer un nouveau projet sur Wokwi
2. Connecter le MPU6050 à l'ESP32
3. Écrire un programme qui :
   - Lit les données de l'accéléromètre
   - Calcule les angles d'inclinaison (roll et pitch)
   - Affiche les angles sur le moniteur série
4. Tester en faisant pivoter virtuellement le MPU6050

Conseil : Utilisez la fonction `atan2()` pour calculer les angles à partir des données de l'accéléromètre

Exercice 2 : Détection de mouvements

Objectif : Créer un système de détection de mouvement/secousse
Instructions :

1. Utiliser le projet précédent comme base
2. Ajouter une LED connectée à l'ESP32
3. Créer un algorithme qui :
   - Détecte les accélérations soudaines
   - Active la LED lorsqu'un mouvement rapide est détecté
   - Désactive la LED après un délai
4. Rendre le système paramétrable (seuil de sensibilité)

Conseil : Calculez la magnitude de l'accélération avec $\sqrt{x^2 + y^2 + z^2}$ et comparez-la à un seuil

Exercice 3 : Podomètre simple

Objectif : Créer un podomètre basique qui compte les pas
Instructions :

1. Ajouter un écran OLED au projet précédent
2. Développer un algorithme qui :
   - Détecte les pics d'accélération verticale
   - Filtre les fausses détections
   - Compte les pas et les affiche sur l'écran
3. Inclure un bouton pour réinitialiser le compteur

Conseil : Utilisez une fenêtre temporelle pour éviter les comptages multiples pour un même pas

Résumé du cours

1. **Introduction au MPU6050 et ESP32**
   - Caractéristiques techniques
   - Principes de fonctionnement
2. **Configuration matérielle et logicielle**
   - Branchement des composants
   - Installation des bibliothèques
3. **Acquisition et traitement des données**
   - Lecture des valeurs brutes
   - Calcul des angles d'orientation
   - Filtrage et calibration
4. **Applications pratiques**
   - Niveau à bulle numérique
   - Détection de mouvements
   - Communication sans fil

Ressources complémentaires

Sites et documentation :

- Wokwi - Simulateur en ligne
- TDK InvenSense - Documentation officielle du MPU6050
- ESP32 Documentation - Documentation officielle d'Espressif
- I2Cdevlib - GitHub de la bibliothèque I2Cdevlib
- Adafruit MPU6050 - Documentation de la bibliothèque Adafruit

Livres recommandés :

- "Capteurs et Arduino" par Martin Evans, Joshua Noble et Jordan Hochenbaum
- "Practical Arduino Engineering" par Harold Timmis
- "ESP32 for Beginners" par Thomas Alva

Perspectives d'approfondissement

1. **Fusion de capteurs avancée**
   - Filtre de Kalman étendu
   - Quaternions pour représenter les rotations
   - Intégration avec des magnétomètres
2. **Interfaces utilisateur avancées**
   - Applications mobiles connectées
   - Réalité virtuelle/augmentée
   - Interface gestuelle
3. **Intelligence artificielle**
   - Reconnaissance de gestes
   - Détection d'anomalies
   - Apprentissage supervisé pour applications spécifiques

Questions et contact

Merci pour votre attention!

Avez-vous des questions ?

Pour tout complément d'information :

formation@iot-capteurs.com
www.iot-capteurs.com