МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА ИНФОРМАЦИОННО-СЕТЕВЫХ ТЕХНОЛОГИЙ

КУРСОВАЯ РАБОТА (ПРОЕКТ)   
ЗАЩИЩЕНА С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| доцент |  |  |  | А.А. Попов |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОЙ РАБОТЕ |
| ТЕМА |
| по дисциплине: ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4134К |  |  |  | Столяров Н.С. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2023

# Задание на курсовое проектирование

Разработка системы обработки данных гироскопа MPU6050 с использованием фильтров для улучшения точности измерений.

**Описание проекта:** Система должна обеспечивать считывание данных с гироскопа MPU6050 и их обработку с использованием различных фильтров (например, фильтр Калмана, низкочастотный фильтр и скользящее среднее).

**Основные требования к системе:**

* **Гироскоп MPU6050:** Считывание данных о угловой скорости и ускорении в трех осях.
* **Программа:** Отображение текущих значений угловой скорости и обработанных данных в режиме реального времени.
* **Программное обеспечение контроллера:** Микропрограмма для Arduino или esp8266, обеспечивающая обработку данных гироскопа, применение фильтров и вывод информации на дисплей.

**Функциональность:**

* **Считывание данных с MPU6050 и их первичная обработка.**
* **Применение различных фильтров для улучшения точности измерений (фильтр Калмана, низкочастотный фильтр, скользящее среднее).**
* **Отображение текущих значений угловой скорости и обработанных данных.**
* **Сохранение данных для последующего анализа**

**Дополнительные возможности:**

* **Реализация функции калибровки гироскопа для устранения смещения.**
* **Ведение логов данных для последующего анализа и визуализации.**
* **Возможность настройки параметров фильтров через интерфейс пользователя.**

# Техническое задание на прибор

Цель: Разработка устройства для обработки данных гироскопа MPU6050 с использованием фильтров для улучшения точности измерений.

**Гироскоп MPU6050:**

* Считывает данные о угловой скорости и ускорении в трех осях.
* Обеспечивает высокую точность измерений.

**Микроконтроллер esp8266:**

* **Обрабатывает данные, полученные от гироскопа.**
* Реализует алгоритмы фильтрации (например, фильтр Калмана, низкочастотный фильтр, скользящее среднее).
* Отправляет данные для отображения по seral порту

**Исполнители (актеры) системы:**

* **Пользователь: Наблюдает за результатами работы устройства через дисплей.**
* **Микроконтроллер: Обрабатывает данные от гироскопа и управляет выводом информации на дисплей.**
* **Гироскоп: Снимает данные о угловой скорости и ускорении.**

# Схемы и настройка проекта

Для реализации проекта потребуется подключить гироскоп MPU6050 и микроконтроллер Arduino или esp8266. Ниже представлена схема подключения:

Компоненты:

* Микроконтроллер (например, Arduino Nano или esp8266)
* Гироскоп MPU6050
* Провода для подключения

|  |
| --- |
| уыз8266 MPU6050  -------------------------  VCC ------> VCC (3.3V или 5V)  GND ------> GND  SDA ------> SDA  SCL ------> SCL |

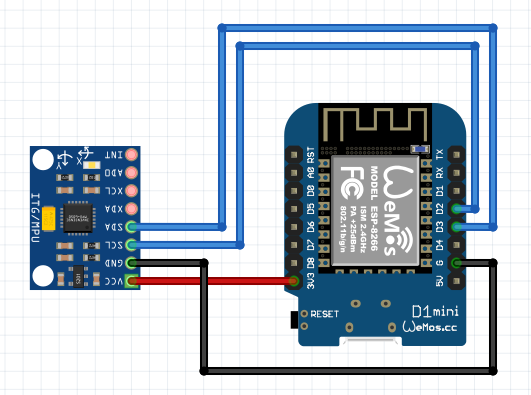


Рисунок 1. Схема подключения дисплея к esp8266

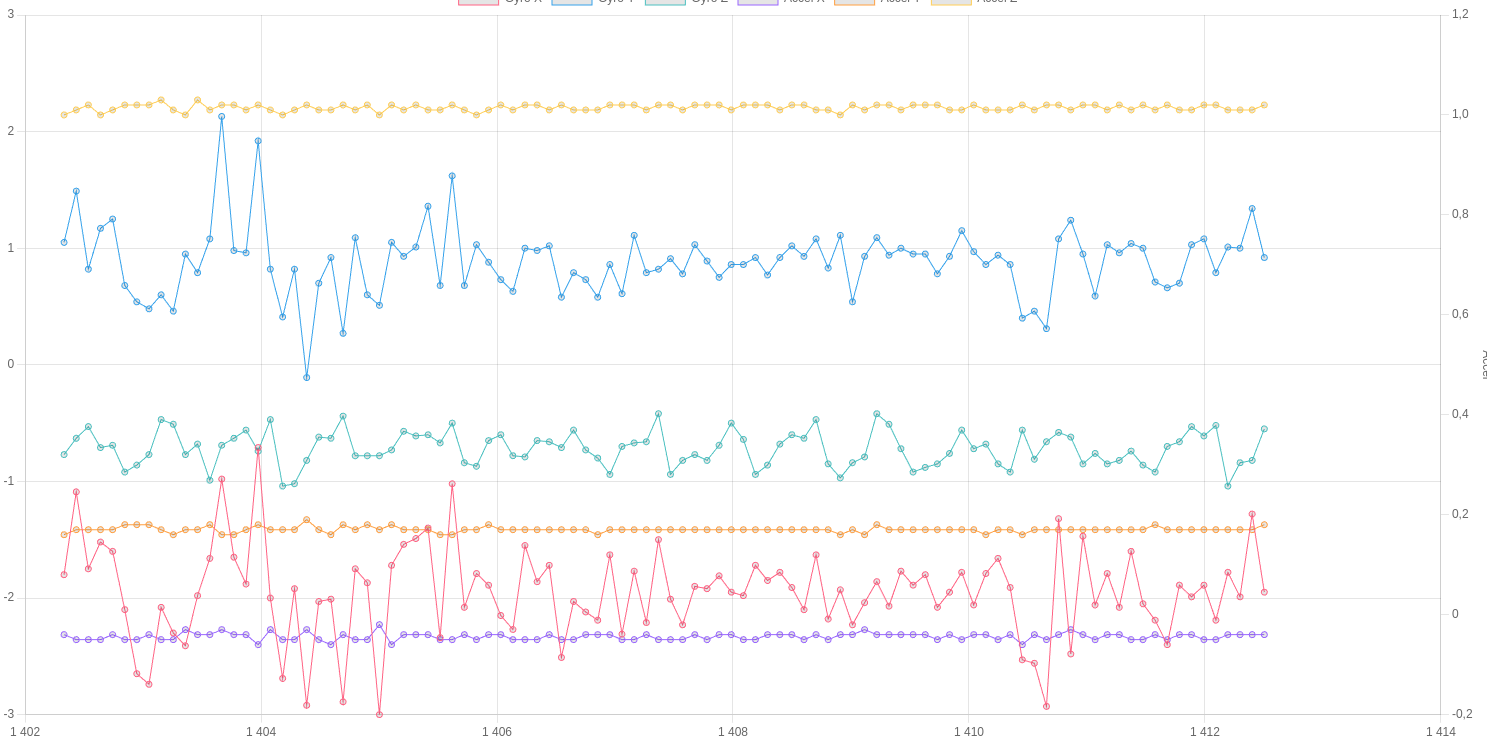
**Настройка программного обеспечения**

Для работы с проектом потребуется установить необходимые библиотеки и настроить среду разработки Arduino:

1. **Установка Arduino IDE:  
    Скачайте и установите Arduino IDE с официального сайта** [Arduino](https://www.arduino.cc/en/software)**.**
2. **становка библиотек:  
    Откройте Arduino IDE и перейдите в меню Скетч -> Подключить библиотеку -> Управление библиотеками.  
    Установите следующие библиотеки: Wire (для работы с I2C)**
3. **Пример кода:**

|  |
| --- |
| #include <Wire.h>  #include <EEPROM.h>  #define MPU6050\_ADDR 0x68  #define PWR\_MGMT\_1 0x6B  #define ACCEL\_XOUT\_H 0x3B  #define GYRO\_XOUT\_H 0x43  void setup() {  Serial.begin(115200);  Wire.begin(D2, D1);  // Инициализация MPU6050  Wire.beginTransmission(MPU6050\_ADDR);  Wire.write(PWR\_MGMT\_1);  Wire.write(0); // Выключаем спящий режим  Wire.endTransmission();  }  void loop() {  // Считываем данные акселерометра  int16\_t accelX = readMPU6050(ACCEL\_XOUT\_H);  int16\_t accelY = readMPU6050(ACCEL\_XOUT\_H + 2);  int16\_t accelZ = readMPU6050(ACCEL\_XOUT\_H + 4);  // Считываем данные гироскопа  int16\_t gyroX = readMPU6050(GYRO\_XOUT\_H);  int16\_t gyroY = readMPU6050(GYRO\_XOUT\_H + 2);  int16\_t gyroZ = readMPU6050(GYRO\_XOUT\_H + 4);  // Преобразуем данные акселерометра в физические величины  float ax = (accelX / 16384.0);  float ay = (accelY / 16384.0);  float az = (accelZ / 16384.0);  // Преобразуем данные гироскопа в физические величины  float gx = gyroX / 131.0;  float gy = gyroY / 131.0;  float gz = gyroZ / 131.0;  // Отправляем данные  Serial.printf("%.2f, %.2f, %.2f, %.2f, %.2f, %.2f\n", ax, ay, az, gx, gy, gz);  }  // Функция для чтения данных с MPU6050  int16\_t readMPU6050(uint8\_t reg) {  Wire.beginTransmission(MPU6050\_ADDR);  Wire.write(reg);  Wire.endTransmission(false);  Wire.requestFrom(MPU6050\_ADDR, 2);  int16\_t value = Wire.read() << 8 | Wire.read();  return value;  } |

1. **Калибровка гироскопа**Для повышения точности измерений рекомендуется выполнить калибровку гироскопа. Это можно сделать, собрав несколько измерений в состоянии покоя и вычислив средние значения для каждого из осей. Эти значения затем можно использовать для коррекции данных.

Рисунок 2. Пример данных с mpu6050

**Мёртвая зона**

Так как качество китайских модулей оставляет желать лучшего, пришлось ограничивать данные которые собирает контроллер