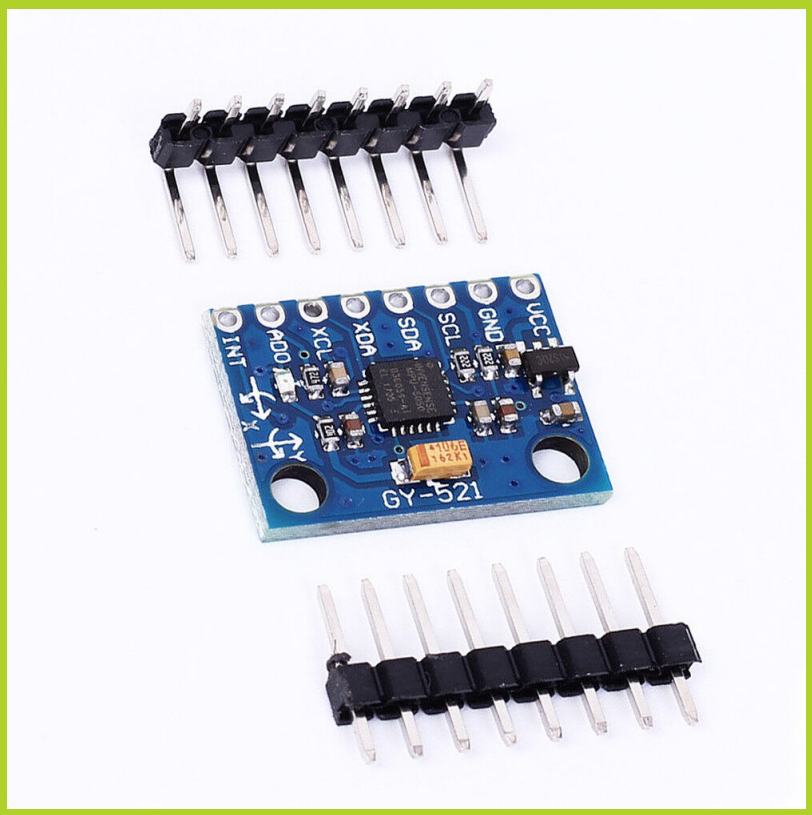
ACELEROMETRO CON GIROSCOPIO GY-521

OKY3234

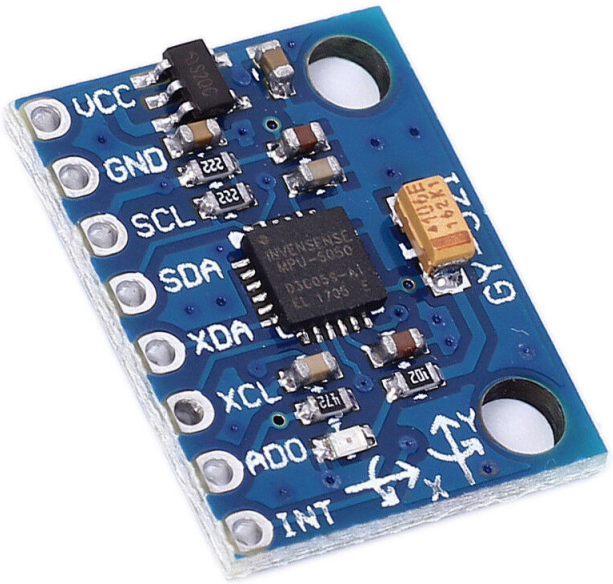


OKYSTAR

AG Electrónica

ACELEROMETRO CON GIROSCOPIO GY-521

OKY3234



DESCRIPCIÓN

El MPU6050 es una unidad de medición inercial, debido a que combina un acelerómetro de 3 ejes y un giroscopio de 3 ejes. La combinación de giroscopio y acelerómetros se conoce comúnmente como Unidad de Medición Inercial o IMU es capaz de medir la velocidad, orientación y aceleración de un sistema. Este módulo es muy utilizado en navegación, goniometría, estabilización.

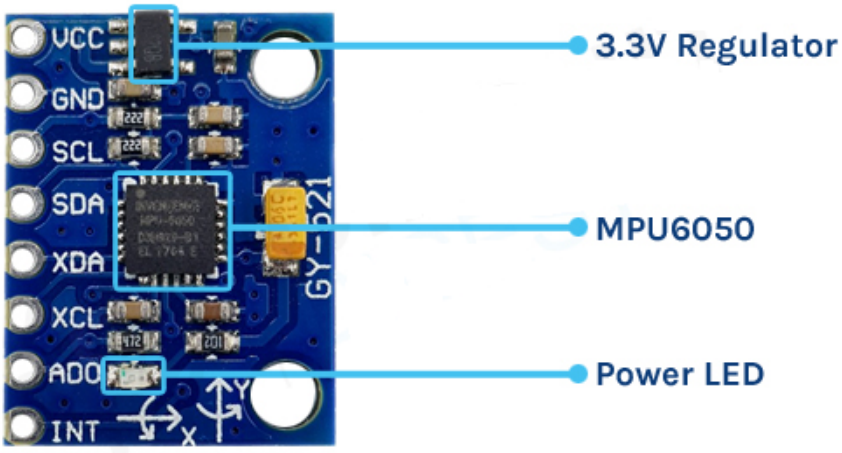
ESPECIFICACIONES

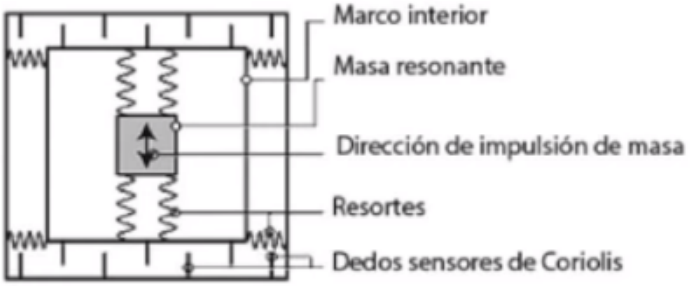
Sensor	MPU6050
Voltaje de funcionamiento	3V~5V
Voltaje de entrada (recomendado)	3.3V~5V
Voltaje de entrada (max y min)	3V~5V
Grados de libertad (DoF)	6
Rango Acelerómetro	2g/4g/8g/16g
Rango Giroscopio	250Grad/Seg, 500Grad/ Seg, 1000Grad/Seg, 2000Grad/Seg
Sensibilidad Giroscopio	131 LSBs/dps
Interfaz	I2C
Conversor AD	16 Bits (salida digital)
Dimensiones	20mm x 16mm x 3mm
Peso	3 g

ELEMENTOS DE LA TARJETA

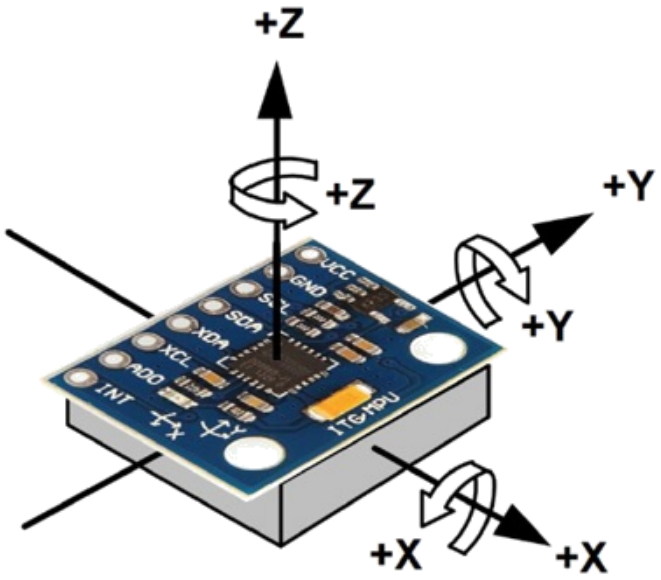
Regulador AP2112K 3.3V: Reguladores de voltaje lineal de una salida fija positiva 600mA encapsulado SOT-25.

MPU6050: es un circuito integrado que combina un giroscopio de 3-ejes y un acelerómetro de 3-ejes en el mismo chip, teniendo 6 grados de libertad (DoF).
LED de encendido: Es el indicador de que el modulo esta encendido.

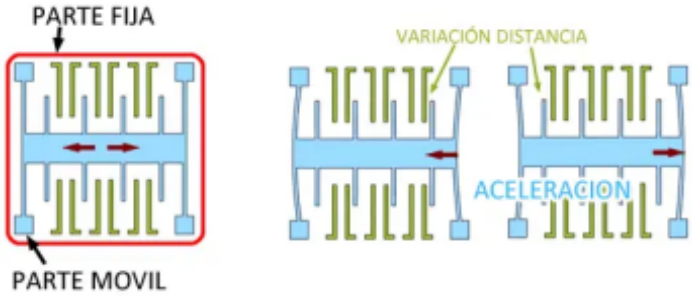




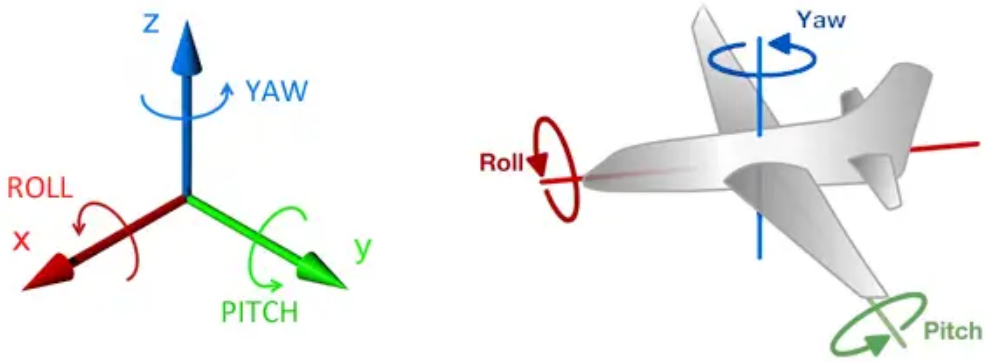
El MPU6050 es un sistema micro electromecánico (MEMS) que consta de un acelerómetro de 3 ejes y un giroscopio de 3 ejes en su interior. Esto nos ayuda a medir la aceleración, la velocidad, la orientación, el desplazamiento y muchos otros parámetros relacionados con el movimiento de un sistema u objeto. Este módulo también tiene un procesador de movimiento digital (DMP) en su interior que es lo suficientemente potente como para realizar cálculos complejos y así liberar el trabajo del microcontrolador.



Acelerómetro: Al aplicar una aceleración al conjunto la masa suspendida ejercerá una fuerza sobre los muelles causando que uno se contraiga y otro se elongue, por lo que la posición relativa de la masa dentro del sensor variará. Este desplazamiento de la masa libre interior puede ser medido para determinar la magnitud de la aceleración. El desplazamiento será proporcional a la aceleración soportada, y se mantendrá constante mientras la aceleración sea constante.



Giroscopio: para determinar la distribución de un objeto respecto a una base en un espacio tridimensional deberemos establecer su posición y orientación. Se requieren tres parámetros para determinar la posición relativa y otros tres para determinar su orientación. Seguramente la más extendida e intuitiva son los ángulos de navegación, o ángulos de Tait-Bryan, en los que la orientación se representa como tres rotaciones ortogonales en torno al eje X (roll), Y (pitch), y Z (yaw) lo cual permite determinar la magnitud y dirección de la rotación. Los ángulos de Tait-Bryan son una variación de los ángulos de Euler, introducidos por el matemático Leonhard Euler durante su estudio sobre la mecánica del sólido rígido.



El filtro complementario se comporta como un filtro de paso alto para la medición del giroscopio y un filtro de paso bajo para la señal del acelerómetro. Es decir, la señal del giroscopio manda a corto plazo, y la del acelerómetro y medio y largo, que es exactamente lo que queremos para compensar sus ventajas y defectos.

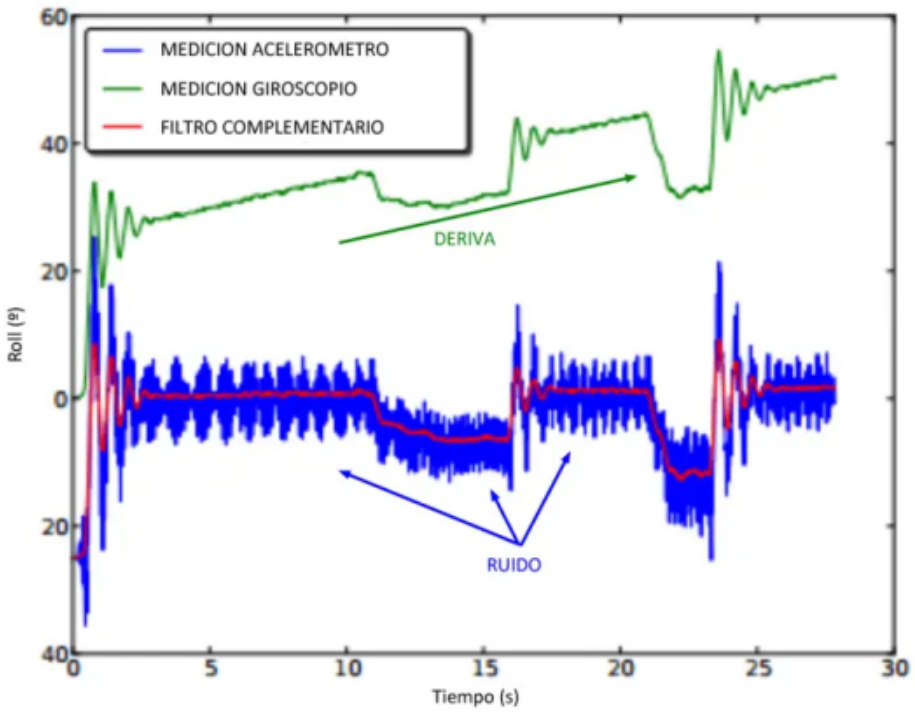
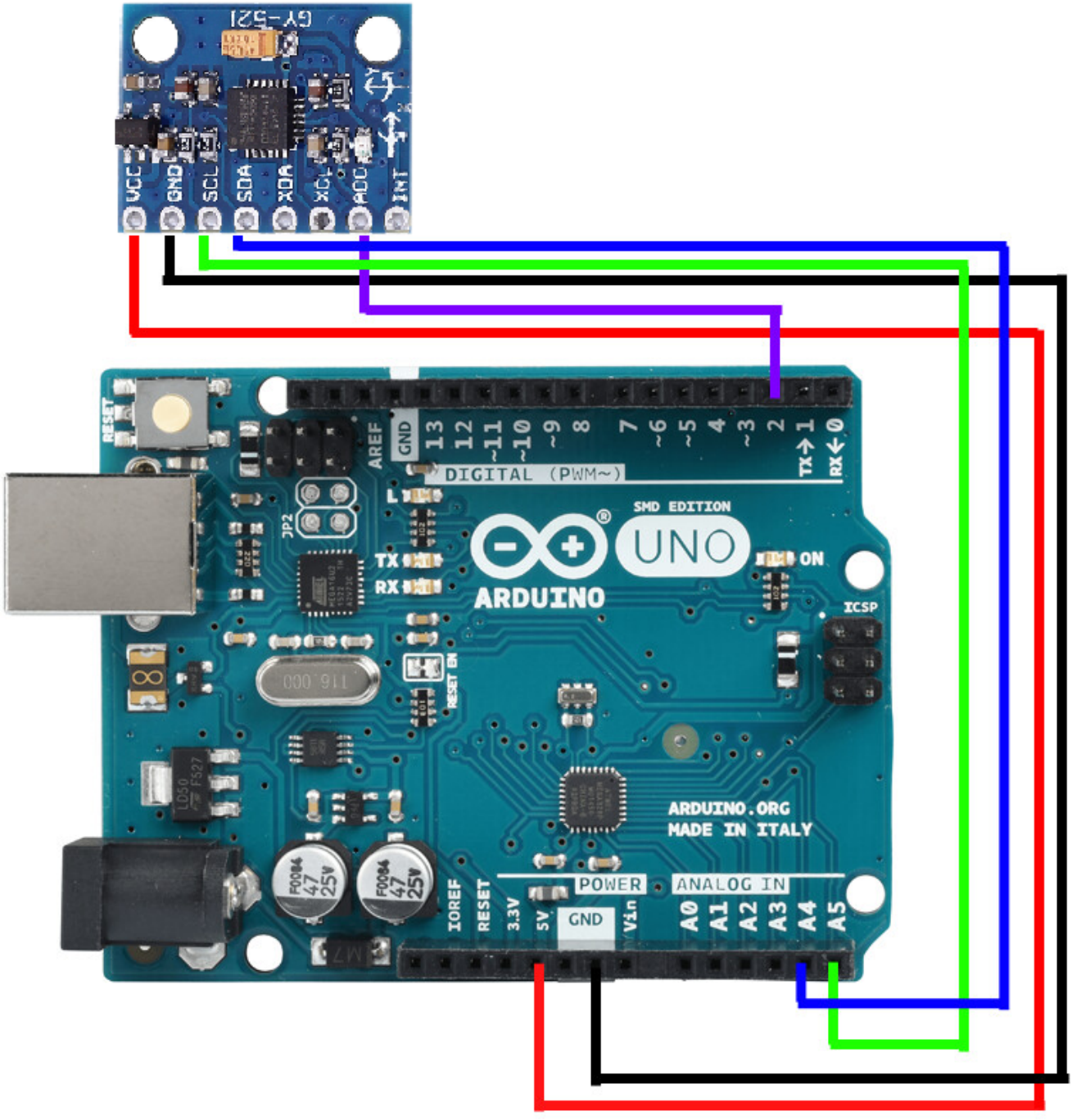
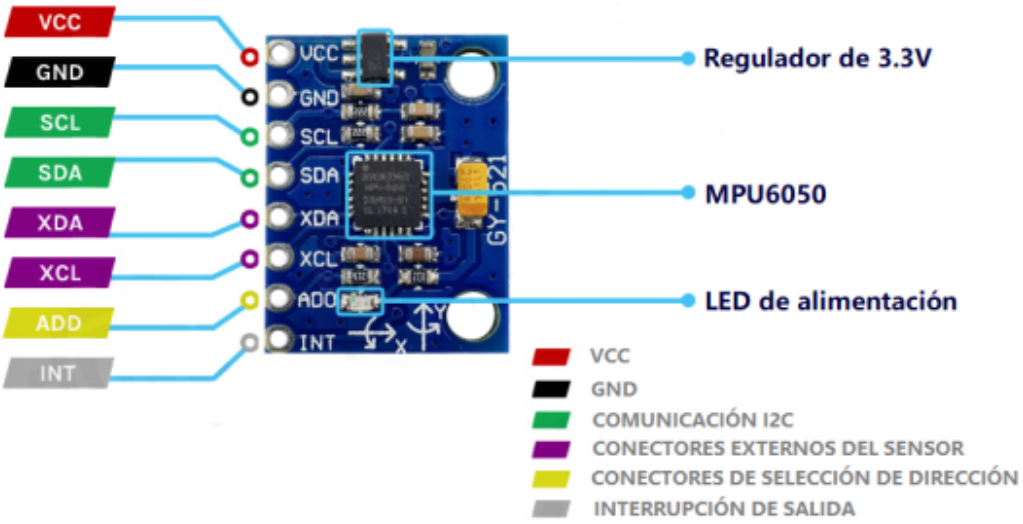


DIAGRAMA DE CONEXIÓN



PINOUT



CCV Proporciona energía para el módulo, conéctese al pin de 5V del Arduino.

TIERRA Conexión a tierra al pin de tierra del Arduino.

SCL Reloj en serie Se utiliza para proporcionar un pulso de reloj para la comunicación I2C.

ASD Datos en serie Se utilizan para transferir datos a través de la comunicación I2C.

XDA Datos seriales auxiliares: se pueden usar para interconectar otros módulos I2C con MPU6050.

XCL Reloj serie auxiliar: se puede utilizar para interconectar otros módulos I2C con MPU6050.

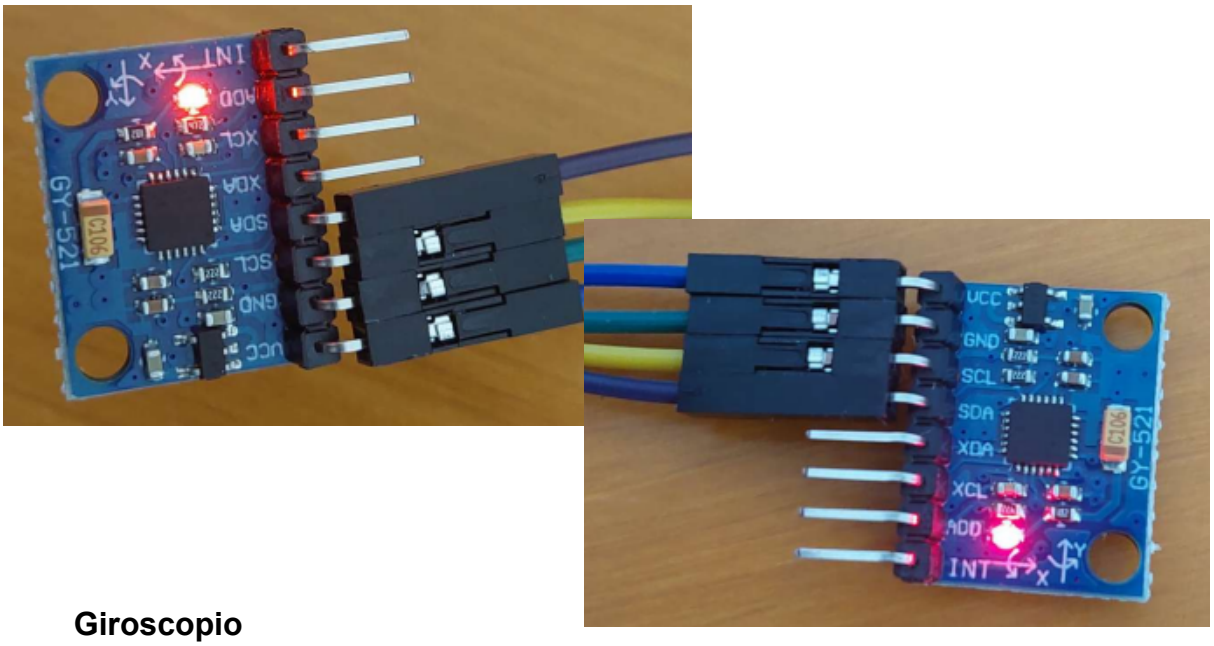
AGREGAR/ADO Pin de selección de dirección si se utilizan varios módulos MPU6050.

ENT Pin de interrupción para indicar que los datos están disponibles para que MCU los lea.

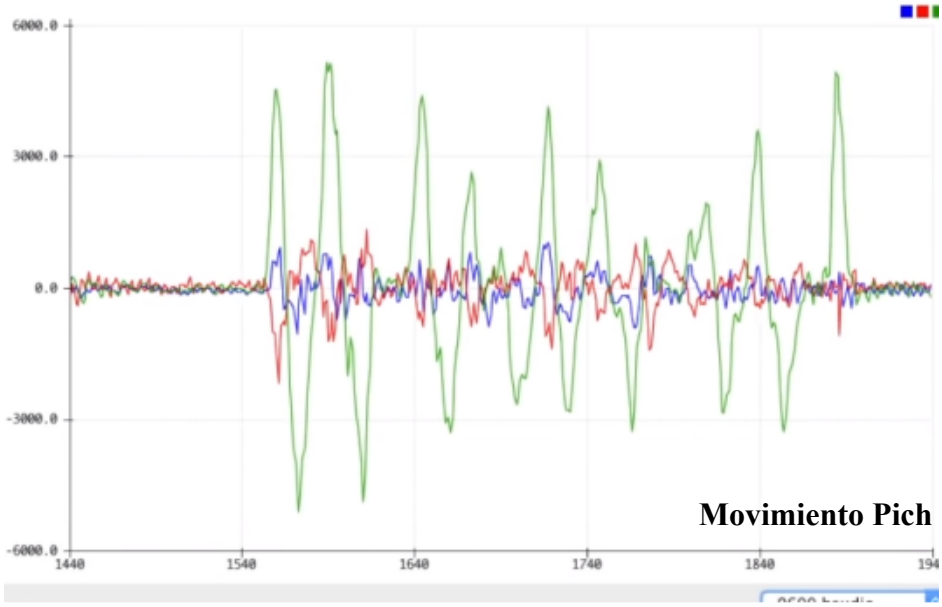
CÓDIGO DE EJEMPLO

```
// (c) Michael Schoeffler 2017, http://www.mschoeffler.de
#include "Wire.h" // This library allows you to communicate with I2C
devices.
const int MPU_ADDR = 0x68; // I2C address of the MPU-6050. If AD0 pin
is set to HIGH, the I2C address will be 0x69.
int16_t accelerometer_x, accelerometer_y, accelerometer_z; //
variables for accelerometer raw data
int16_t gyro_x, gyro_y, gyro_z; // variables for gyro raw data
int16_t temperature; // variables for temperature data
char tmp_str[7]; // temporary variable used in convert function
char* convert_int16_to_str(int16_t i) { // converts int16 to string.
Moreover, resulting strings will have the same length in the debug
monitor.
  sprintf(tmp_str, "%6d", i);
  return tmp_str;
}
void setup() {
  Serial.begin(9600);
  Wire.begin();
  Wire.beginTransmission(MPU_ADDR); // Begins a transmission to the
I2C slave (GY-521 board)
  Wire.write(0x6B); // PWR_MGMT_1 register
  Wire.write(0); // set to zero (wakes up the MPU-6050)
  Wire.endTransmission(true);
}
void loop() {
  Wire.beginTransmission(MPU_ADDR);
  Wire.write(0x3B); // starting with register 0x3B (ACCEL_XOUT_H)
[MPU-6000 and MPU-6050 Register Map and Descriptions Revision 4.2,
p.40]
  Wire.endTransmission(false); // the parameter indicates that the
Arduino will send a restart. As a result, the connection is kept
active.
  Wire.requestFrom(MPU_ADDR, 7*2, true); // request a total of 7*2=14
registers
```

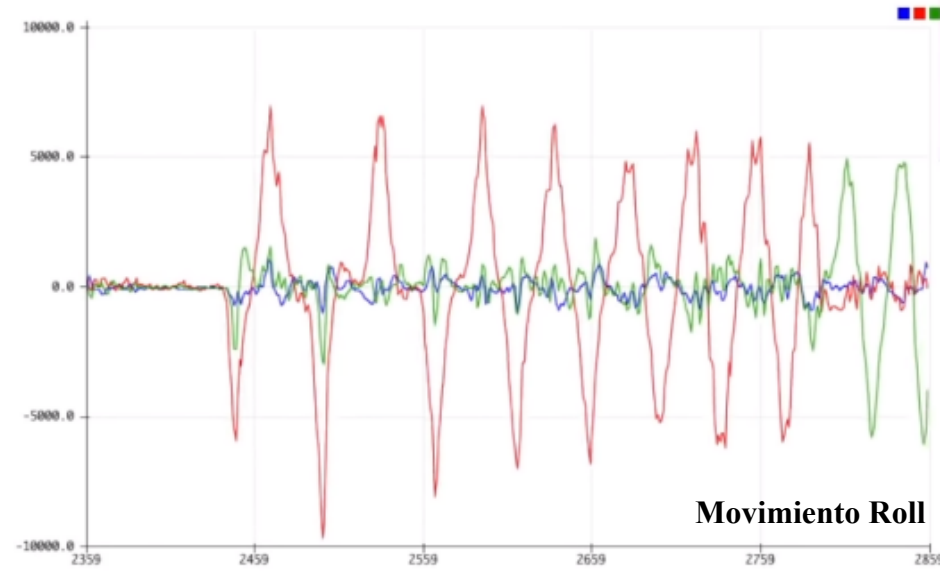
```
// "Wire.read()<<8 | Wire.read();" means two registers are read and
stored in the same variable
  accelerometer_x = Wire.read()<<8 | Wire.read(); // reading
registers: 0x3B (ACCEL_XOUT_H) and 0x3C (ACCEL_XOUT_L)
  accelerometer_y = Wire.read()<<8 | Wire.read(); // reading
registers: 0x3D (ACCEL_YOUT_H) and 0x3E (ACCEL_YOUT_L)
  accelerometer_z = Wire.read()<<8 | Wire.read(); // reading
registers: 0x3F (ACCEL_ZOUT_H) and 0x40 (ACCEL_ZOUT_L)
  temperature = Wire.read()<<8 | Wire.read(); // reading registers:
0x41 (TEMP_OUT_H) and 0x42 (TEMP_OUT_L)
  gyro_x = Wire.read()<<8 | Wire.read(); // reading registers: 0x43
(GYRO_XOUT_H) and 0x44 (GYRO_XOUT_L)
  gyro_y = Wire.read()<<8 | Wire.read(); // reading registers: 0x45
(GYRO_YOUT_H) and 0x46 (GYRO_YOUT_L)
  gyro_z = Wire.read()<<8 | Wire.read(); // reading registers: 0x47
(GYRO_ZOUT_H) and 0x48 (GYRO_ZOUT_L)

  // print out data
  //
Serial.print(convert_int16_to_str(accelerometer_x));Serial.print(",");
  //
Serial.print(convert_int16_to_str(accelerometer_y));Serial.print(",");
  //Serial.print(convert_int16_to_str(accelerometer_z));
  // the following equation was taken from the documentation
[MPU-6000/MPU-6050 Register Map and Description, p.30]
  //Serial.print(temperature/340.00+36.53);
  Serial.print(convert_int16_to_str(gyro_z));Serial.print(",");
  Serial.print(convert_int16_to_str(gyro_y)); Serial.print(",");
  Serial.print(convert_int16_to_str(gyro_x));
  Serial.println();

  delay(20);
}
```



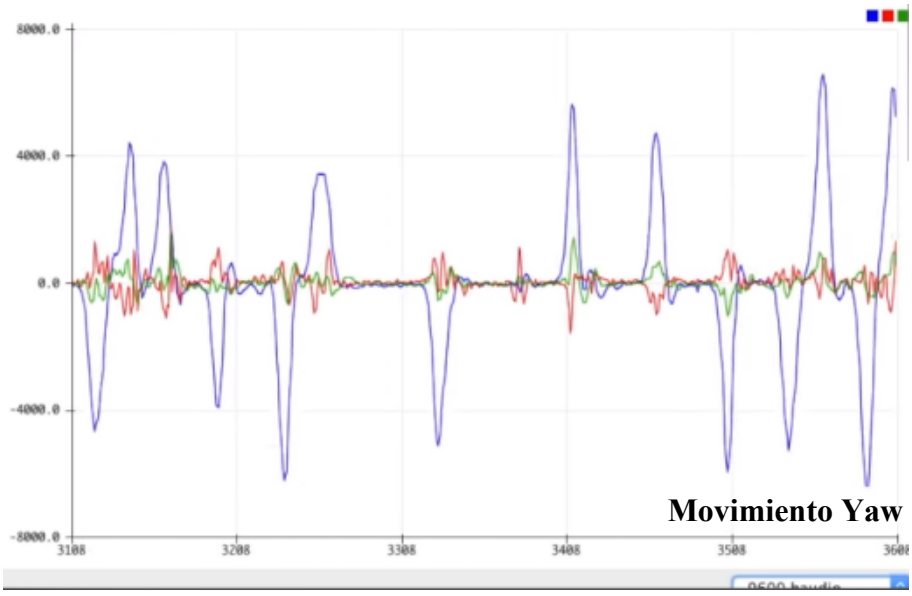
Giroscopio



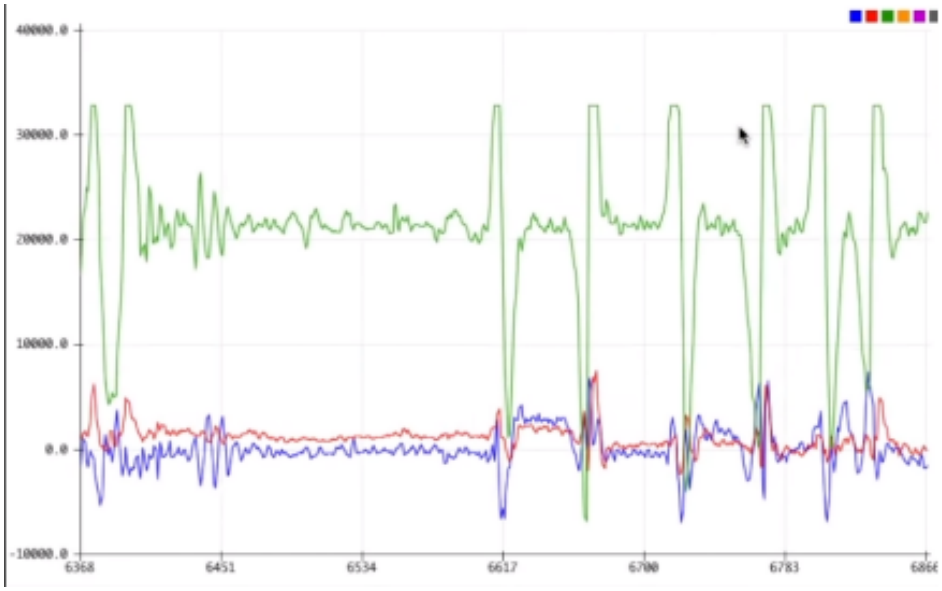
Movimiento Pich

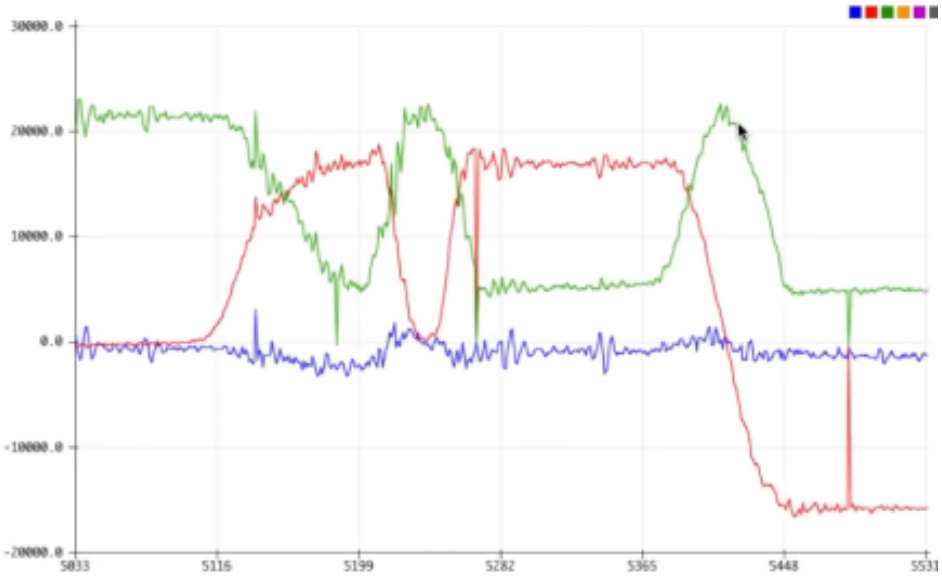


Movimiento Roll



Acelerómetro





REALIZÓ:GAC

REVISÓ: GAC