#include "Wire.h" // This library allows you to communicate with I2C devices.

const int flexPin1 = A0;

const int flexPin2 = A1;

const int flexPin3 = A2;

const int flexPin4 = A3;

const int flexPin5 = A6;

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 Ax, Ay, Az; // variables for accelerometer raw 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

Ax = Wire.read()<<8 | Wire.read(); // reading registers: 0x3B (ACCEL\_XOUT\_H) and 0x3C (ACCEL\_XOUT\_L)

Ay = Wire.read()<<8 | Wire.read(); // reading registers: 0x3D (ACCEL\_YOUT\_H) and 0x3E (ACCEL\_YOUT\_L)

Az = Wire.read()<<8 | Wire.read(); // reading registers: 0x3F (ACCEL\_ZOUT\_H) and 0x40 (ACCEL\_ZOUT\_L)

// print out data

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

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

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

Serial.println();

delay(1000);

int f1;

int f2;

int f3;

int f4;

int f5;

f1 = analogRead(flexPin1);

//Serial.print("sensor 1: ");

Serial.println(f1);

delay(1000);

f2 = analogRead(flexPin2);

//Serial.print("sensor 2: ");

Serial.println(f2);

delay(1000);

f3 = analogRead(flexPin3);

//Serial.print("sensor 3: ");

Serial.println(f3);

delay(1000);

f4 = analogRead(flexPin4);

//Serial.print("sensor 4: ");

Serial.println(f4);

delay(1000);

f5 = analogRead(flexPin5);

//Serial.print("sensor 5: ");

Serial.println(f5);

delay(1000);

if((f1<195)&&(f1>180) && (f2<265)&&(f2>250) && (f3<170)&&(f3>150) && (f4<200)&&(f4>185) && (f5<230)&&(f5>215) )

{

Serial.println("WHY") ;

}

//delay(1000);

if((f1<170)&&(f1>160) && (f2<220)&&(f2>200) && (f3<125)&&(f3>115) && (f4<175)&&(f4>160) && (f5<215)&&(f5>200) )

{

Serial.println("SORRY") ;

}

//delay(1000);

if((f1<180)&&(f1>170) && (f2<280)&&(f2>260) && (f3<150)&&(f3>135) && (f4<190)&&(f4>175) && (f5<230)&&(f5>215) )

{

Serial.println("ALWAYS") ;

}

//delay(1000);

if((f1<185)&&(f1>170) && (f2<245)&&(f2>230) && (f3<150)&&(f3>140) && (f4<170)&&(f4>160) && (f5<215)&&(f5>200) )

{

Serial.println("NO") ;

}

//delay(1000);

if((f1<190)&&(f1>180) && (f2<245)&&(f2>240) && (f3<155)&&(f3>140) && (f4<210)&&(f4>180) && (f5<220)&&(f5>200) )

{

if((Ax<14500)&&(Ax>13000 ) && (Ay<9000)&&(Ay>7000) && (Az<-8000)&&(Az>-6000))

{

Serial.println("HELLO") ;

}

if((Ax<17500)&&(Ax>16000 ) && (Ay<-4000)&&(Ay>-2500) && (Az<-3800)&&(Az>-2200))

{

Serial.println("GOOD BYE") ;

}

if((Ax<16000)&&(Ax>15000 ) && (Ay<-7000)&&(Ay>-5200) && (Az<-4500)&&(Az>-3000))

{

Serial.println("THANKS") ;

}

if((Ax<13500)&&(Ax>11500 ) && (Ay<-10300)&&(Ay>-9000) && (Az<-6000)&&(Az>-4000))

{

Serial.println("PLEASE") ;

}

}

//delay(1000);

}