//CODE FOR FLEX SENSOR

const int flexPin1 = A0;

const int flexPin2 = A1;

const int flexPin3 = A2;

const int flexPin4 = A3;

const int flexPin5 = A6;

void setup()

{

Serial.begin(9600);

}

void loop()

{

int flexValue1;

int flexValue2;

int flexValue3;

int flexValue4;

int flexValue5;

flexValue1 = analogRead(flexPin1);

Serial.print("sensor 1: ");

Serial.println(flexValue1);

delay(3000);

flexValue2 = analogRead(flexPin2);

Serial.print("sensor 2: ");

Serial.println(flexValue2);

delay(3000);

flexValue3 = analogRead(flexPin3);

Serial.print("sensor 3: ");

Serial.println(flexValue3);

delay(3000);

flexValue4 = analogRead(flexPin4);

Serial.print("sensor 4: ");

Serial.println(flexValue4);

delay(3000);

flexValue5 = analogRead(flexPin5);

Serial.print("sensor 5: ");

Serial.println(flexValue5);

delay(3000);

}

CODE FOR GYROSCOPE

#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("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));

// the following equation was taken from the documentation [MPU-6000/MPU-6050 Register Map and Description, p.30]

Serial.print(" | tmp = "); Serial.print(temperature/340.00+36.53);

Serial.print(" | gX = "); Serial.print(convert\_int16\_to\_str(gyro\_x));

Serial.print(" | gY = "); Serial.print(convert\_int16\_to\_str(gyro\_y));

Serial.print(" | gZ = "); Serial.print(convert\_int16\_to\_str(gyro\_z));

Serial.println();

// delay

delay(1000);

}

CODE FOR YAW,PITCH AND ROLL

/\*

MPU6050 Triple Axis Gyroscope & Accelerometer. Pitch & Roll & Yaw Gyroscope Example.

Read more: http://www.jarzebski.pl/arduino/czujniki-i-sensory/3-osiowy-zyroskop-i-akcelerometr-mpu6050.html

GIT: https://github.com/jarzebski/Arduino-MPU6050

Web: http://www.jarzebski.pl

(c) 2014 by Korneliusz Jarzebski

\*/

#include <Wire.h>

#include <MPU6050.h>

MPU6050 mpu;

// Timers

unsigned long timer = 0;

float timeStep = 0.01;

// Pitch, Roll and Yaw values

float pitch = 0;

float roll = 0;

float yaw = 0;

void setup()

{

Serial.begin(115200);

// Initialize MPU6050

while(!mpu.begin(MPU6050\_SCALE\_2000DPS, MPU6050\_RANGE\_2G))

{

Serial.println("Could not find a valid MPU6050 sensor, check wiring!");

delay(500);

}

// Calibrate gyroscope. The calibration must be at rest.

// If you don't want calibrate, comment this line.

mpu.calibrateGyro();

// Set threshold sensivty. Default 3.

// If you don't want use threshold, comment this line or set 0.

mpu.setThreshold(3);

}

void loop()

{

timer = millis();

// Read normalized values

Vector norm = mpu.readNormalizeGyro();

// Calculate Pitch, Roll and Yaw

pitch = pitch + norm.YAxis \* timeStep;

roll = roll + norm.XAxis \* timeStep;

yaw = yaw + norm.ZAxis \* timeStep;

// Output raw

Serial.print(" Pitch = ");

Serial.print(pitch);

Serial.print(" Roll = ");

Serial.print(roll);

Serial.print(" Yaw = ");

Serial.println(yaw);

// Wait to full timeStep period

delay((timeStep\*1000) - (millis() - timer));

}

CODE WITH GESTURES

#include "math.h"

#include "Wire.h"

#include <SoftwareSerial.h>

const int MPU=0x68;

int16\_t AcX,AcY,AcZ,Tmp,GyX,GyY,GyZ;

String newprint,prevprint;

SoftwareSerial EEBlue(10,11);

void setup()

{

const int vcc=5;

Wire.begin();

Wire.beginTransmission(MPU);

Wire.write(0x6B);

Wire.write(0);

Wire.endTransmission(true);

Serial.begin(9600);

EEBlue.begin(9600);

}

void loop()

{

int a;

int b;

int f1; // variable for analog value from pinky finger

int f2; // variable for analog value from ring finger

int f3 ; // variable for analog value from middle finger

int f4;// variable for analog value from index finger

int f5;

int flexPin1 = A0; //for flex sensor on pinky finger

int flexPin2 = A1; //for flex sensor on middle finger

int flexPin3 = A2; //for flex sensor on index finger

int flexPin4 = A3; //for flex sensor on thumb

int flexPin5 = A6;

f1=analogRead(flexPin1);

f2=analogRead(flexPin2);

f3=analogRead(flexPin3);

f4=analogRead(flexPin4);

f5=analogRead(flexPin5);

Wire.beginTransmission(MPU);

Wire.write(0x3B);

Wire.endTransmission(false);

Wire.requestFrom(MPU,14,true);

AcX=Wire.read()<<8|Wire.read(); // 0x3B (ACCEL\_XOUT\_H) 0x3C (ACCEL\_XOUT\_L)

AcY=Wire.read()<<8|Wire.read(); // 0x3D (ACCEL\_YOUT\_H) 0x3E (ACCEL\_YOUT\_L)

AcZ=Wire.read()<<8|Wire.read(); // 0x3F (ACCEL\_ZOUT\_H) 0x40 (ACCEL\_ZOUT\_L)

GyX=Wire.read()<<8|Wire.read(); // 0x43 (GYRO\_XOUT\_H) 0x44 (GYRO\_XOUT\_L)

GyY=Wire.read()<<8|Wire.read(); // 0x45 (GYRO\_YOUT\_H) 0x46 (GYRO\_YOUT\_L)

GyZ=Wire.read()<<8|Wire.read(); // 0x47 (GYRO\_ZOUT\_H) 0x48 (GYRO\_ZOUT\_L)

//Code for the gesture Hello

if(f1<195 && f2<220 && f3>300 && f4>290 && f5>200 )

{

if( GyX <1700 && GyY >300 && GyZ>220)

{

a=1;

}

if(AcX > 12000 && AcY > 1000 && AcZ<7000)

{

b=1;

}

if(a==1 && b==1)

{

Serial.println("Hello");

a=0;

b=0;

}

}

//Code for the gesture GOOD BYE

if(f1<200 && f2>200 && f3>290 && f4>270 && f5>200 )

{

if( GyX <1700 && GyY >300 && GyZ>220)

{

a=1;

}

if(AcX > 12000 && AcY > 1000 && AcZ<7000)

{

b=1;

}

if(a==1 && b==1)

{

Serial.println("GOOD BYE");

a=0;

b=0;

}

}

//CODE FOR GESTURE WHERE

if(f1<180 && f2<190 && f3>180 && f4>250 && f5>170 )

{

if( GyX <1700 && GyY >300 && GyZ>220)

{

a=1;

}

if(AcX > 12000 && AcY > 1000 && AcZ<7000)

{

b=1;

}

if(a==1 && b==1)

{

Serial.println("WHERE");

a=0;

b=0;

}

}

//CODE FOR GESTURE YES

if(f1<135 && f2<170 && f3<170 && f4>200 && f5<150 )

{

if( GyX <1700 && GyY >300 && GyZ>220)

{

a=1;

}

if(AcX > 12000 && AcY > 1000 && AcZ<7000)

{

b=1;

}

if(a==1 && b==1)

{

Serial.println("Hello");

a=0;

b=0;

}

}

delay(500);

Serial.write(EEBlue.read());

}