#include "I2Cdev.h"

#include "MPU6050.h"

// Arduino Wire library is required if I2Cdev I2CDEV\_ARDUINO\_WIRE implementation

// is used in I2Cdev.h

#if I2CDEV\_IMPLEMENTATION == I2CDEV\_ARDUINO\_WIRE

#include "Wire.h"

#endif

MPU6050 accelgyro;

//MPU6050 accelgyro(0x69); // <-- use for AD0 high

int16\_t ax, ay, az;

int16\_t gx, gy, gz;

#define OUTPUT\_READABLE\_ACCELGYRO

#define LED\_PIN 13

bool blinkState = false;

int input = 50; // the FSR and cap are connected to pin50

int reset = 49;

int \*fsrReading; // the digital readings

int myPins[] = {27, 28, 31, 32, 35, 36, 39, 40, 43, 44}; // The pins used as VCC for the FSRs

int minmax[] = {23, 24}; // The resistors used to find a min and max count

int \*rcounts; // The counts from the resistors

void setup(void) {

// We'll send debugging information via the Serial monitor

#if I2CDEV\_IMPLEMENTATION == I2CDEV\_ARDUINO\_WIRE

Wire.begin();

#elif I2CDEV\_IMPLEMENTATION == I2CDEV\_BUILTIN\_FASTWIRE

Fastwire::setup(400, true);

#endif

Serial.begin(9600);

for (int counter = 0; counter<10; counter++){//set FSR VCC pins to output mode and low state

pinMode(myPins[counter], OUTPUT);

digitalWrite(myPins[counter], LOW);

}

for (int counter = 0; counter<2; counter++){//set resistor VCC pins to output mode and low state

pinMode(minmax[counter], OUTPUT);

digitalWrite(minmax[counter], LOW);

}

pinMode(reset,OUTPUT);

pinMode(LED\_PIN, OUTPUT);

}

void loop(void) {

// read the resistor using the RCtime technique

// read raw accel/gyro measurements from device

blinkState = !blinkState;

digitalWrite(LED\_PIN, blinkState);

accelgyro.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);

fsrReading = RCtime(input, myPins, reset);

for (int count = 0; count<10; count++){//4th fsr is working fsr INDEX FINGER

Serial.print(fsrReading[count]); // the raw analog reading

Serial.print(",");

}

// these methods (and a few others) are also available

//accelgyro.getAcceleration(&ax, &ay, &az);

//accelgyro.getRotation(&gx, &gy, &gz);

#ifdef OUTPUT\_READABLE\_ACCELGYRO

// display tab-separated accel/gyro x/y/z values

// Serial.print("a/g:\t");

Serial.print(ax+3400); Serial.print(",");

Serial.print(ay-14800); Serial.print(",");

Serial.print(az-4100); Serial.print(",");

Serial.println();

//Serial.print(gx); Serial.print("\t");

//Serial.print(gy); Serial.print("\t");

//Serial.println(gz);

#endif

#ifdef OUTPUT\_BINARY\_ACCELGYRO

Serial.write((uint8\_t)(ax >> 8)); Serial.write((uint8\_t)(ax & 0xFF));

Serial.write((uint8\_t)(ay >> 8)); Serial.write((uint8\_t)(ay & 0xFF));

Serial.write((uint8\_t)(az >> 8)); Serial.write((uint8\_t)(az & 0xFF));

//Serial.write((uint8\_t)(gx >> 8)); Serial.write((uint8\_t)(gx & 0xFF));

//Serial.write((uint8\_t)(gy >> 8)); Serial.write((uint8\_t)(gy & 0xFF));

//Serial.write((uint8\_t)(gz >> 8)); Serial.write((uint8\_t)(gz & 0xFF));

#endif

/\*Serial.print("Resistors: ");

Serial.print(rcounts[0]); // the raw analog reading

Serial.print(",");

Serial.print(rcounts[1]);

Serial.print(" FSRs: ");\*/

//rcounts = getminmax(minmax, input, reset);

//delay(1000);

Serial.println();

}

// Uses a digital pin to measure a resistor (like an FSR or photocell!)

// We do this by having the resistor feed current into a capacitor and

// counting how long it takes to get to Vcc/2 (for most arduinos, thats 2.5V)

int \*RCtime(int RCpin, int pins[], int reset) {

int reading[] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0}; // start with 0s

for (int count=0; count<10; count++){

digitalWrite(reset, HIGH); //set reset to high

// set the pin to an output and pull to LOW (ground)

pinMode(RCpin, OUTPUT);

digitalWrite(RCpin, LOW);

// Now set the pin to an input and...

pinMode(RCpin, INPUT);

digitalWrite(pins[count], HIGH);//Set VCC to FSR input

while (digitalRead(RCpin) == LOW) { // count how long it takes to rise up to HIGH

reading[count]++; // increment to keep track of time

if (reading[count] == 30000) {

// if we got this far, the resistance is so high

// its likely that nothing is connected!

break; // leave the loop

}

}

digitalWrite(pins[count], LOW);//Set VCC pin back Sto low for next iteration

digitalWrite(reset, LOW);//discharge capacitor

//delay(1000); //ADDED TO TEST

}

// OK either we maxed out at 30000 or hopefully got a reading, return the count after resetting FSR Vin pin

return reading;

}

int \*getminmax(int minmax[], int RCpin, int reset){

int mmcount[] = {0, 0};

for (int counter = 0; counter<2; counter++){

digitalWrite(reset, HIGH);

pinMode(RCpin, OUTPUT);

digitalWrite(RCpin, LOW);

// Now set the pin to an input and...

pinMode(RCpin, INPUT);

digitalWrite(minmax[counter], HIGH);//Set VCC to resistor input

while (digitalRead(RCpin) == LOW) { // count how long it takes to rise up to HIGH

mmcount[counter]++; // increment to keep track of time

}

digitalWrite(minmax[counter], LOW);//Set resistor input low

digitalWrite(reset, LOW); //discharge capacitor

}

return mmcount;

}