/\* Basic IO tester

Tests basic inputs (accel-pot, brake switches, current) and outputs (pwm drive)

The circuit:

\* accel-pot connected to A7

\* motor pwm connected to pin D6

created 2005

by David A. Mellis

modified 8 Feb 2010

by Paul Stoffregen

modified 12Aug2017

by Sujay Sirur

This code is NOT in the public domain.

\*/

// constants won't change. Used here to

// set pin numbers:

const long interval = 100; // sample time of the main task (milliseconds)

const int sensePin1 = A7; // LDR is connected here

const int sensePin2 = A6; // LDR is connected here

const int sensePin3 = A5; // LDR is connected here

const int sensePin4 = A4; // LDR is connected here

const int sensePin5 = A3; // LDR is connected here

const int ledPin1 = 2; // the number of the LED pin

const int ledPin2 = 3; // the number of the LED pin

const int ledPin3 = 4; // the number of the LED pin

const int ledPin4 = 5; // the number of the LED pin

const int ledPin5 = 6; // the number of the LED pin

const int stayAlivePin = 9; // high on this pin keeps controller alive (low powers off)

const unsigned int turnOnThreshold = 350; // 3V on LDR

//const unsigned int turnOffThreshold = 2; // 0.01V on LDR

const unsigned int flickerPercent = 10; // in percentage

const unsigned int flickerDuration = 10; // in msec

const unsigned int minOnTimeMins = 5; // minimum ON time in minutes i.e. wait for LED's to be turned ON

const unsigned int maxOnTimeMins = 180; // maximum ON time in minutes i.e. power off even if LED's are ON

const unsigned int switchOffDelay = 20; // wait time after forced switch OFF before enabling switch ON

const int switchOffLPF\_N = 10; // LPF filter factor - higher value means slower filter

const int switchOffLPF\_Thd = 300; // HPF threshold to turn OFF

// Variables will change:

int ledState1 = LOW; // ledState used to set the LED

int ledState2 = LOW; // ledState used to set the LED

int ledState3 = LOW; // ledState used to set the LED

int ledState4 = LOW; // ledState used to set the LED

int ledState5 = LOW; // ledState used to set the LED

long previousMillis = 0; // will store last time LED was updated

long noOfmilliSecs = 0; // variables to record time since power ON

long noOfSecs = 0;

long noOfMins = 0;

int light1VoltLPF=0, light1VoltHPF=0, light1VoltHPFPrev=0, light1SwitchOffDelay=0;

int light2VoltLPF=0, light2VoltHPF=0, light2VoltHPFPrev=0, light2SwitchOffDelay=0;

int light3VoltLPF=0, light3VoltHPF=0, light3VoltHPFPrev=0, light3SwitchOffDelay=0;

int light4VoltLPF=0, light4VoltHPF=0, light4VoltHPFPrev=0, light4SwitchOffDelay=0;

int light5VoltLPF=0, light5VoltHPF=0, light5VoltHPFPrev=0, light5SwitchOffDelay=0;

void setup() {

unsigned int i;

unsigned int selfTestDelay=50;

// initialize serial communication at 9600 bits per second:

Serial.begin(9600);

// set the stay alive pin to latch the power

pinMode(stayAlivePin, OUTPUT);

digitalWrite(stayAlivePin, HIGH);

// set the digital pin as output:

pinMode(ledPin1, OUTPUT);

pinMode(ledPin2, OUTPUT);

pinMode(ledPin3, OUTPUT);

pinMode(ledPin4, OUTPUT);

pinMode(ledPin5, OUTPUT);

randomSeed(analogRead(sensePin1)+analogRead(sensePin2)+analogRead(sensePin3)+analogRead(sensePin4)+analogRead(sensePin5));

// powerup self check

digitalWrite(ledPin1, HIGH);

for(i=1;i<10;i++) {

delay(selfTestDelay);

digitalWrite(ledPin2, HIGH);

digitalWrite(ledPin1, LOW);

delay(selfTestDelay);

digitalWrite(ledPin3, HIGH);

digitalWrite(ledPin2, LOW);

delay(selfTestDelay);

digitalWrite(ledPin4, HIGH);

digitalWrite(ledPin3, LOW);

delay(selfTestDelay);

digitalWrite(ledPin5, HIGH);

digitalWrite(ledPin4, LOW);

delay(selfTestDelay);

digitalWrite(ledPin1, HIGH);

digitalWrite(ledPin5, LOW);

}

digitalWrite(ledPin1, LOW);

digitalWrite(ledPin2, LOW);

digitalWrite(ledPin3, LOW);

digitalWrite(ledPin4, LOW);

digitalWrite(ledPin5, LOW);

}

void loop()

{

// here is where you'd put code that needs to be running all the time.

// check to see if it's time to blink the LED; that is, if the

// difference between the current time and last time you blinked

// the LED is bigger than the interval at which you want to

// blink the LED.

unsigned long currentMillis = millis();

unsigned int light1RawDec,light1RawVolt;

unsigned int light2RawDec,light2RawVolt;

unsigned int light3RawDec,light3RawVolt;

unsigned int light4RawDec,light4RawVolt;

unsigned int light5RawDec,light5RawVolt;

if(currentMillis - previousMillis > interval) {

// save the last time you blinked the LED

previousMillis = currentMillis;

// count the time in mS,Sec,Mins since power up

noOfmilliSecs+=interval;

if(noOfmilliSecs>=1000) {

noOfSecs+=1;

noOfmilliSecs-=1000;

}

if(noOfSecs>=60) {

noOfMins+=1;

noOfSecs-=60;

}

// Read LDR for each light and set the state of that light

light1RawDec = analogRead(sensePin1);

light1RawVolt = map(light1RawDec, 0, 1023, 500, 0); // 5V --> 1023

if((light1RawVolt>turnOnThreshold)&&(light1SwitchOffDelay==0)) ledState1=HIGH;

//if(light1RawVolt<turnOffThreshold) ledState1=LOW;

light1VoltLPF = light1VoltLPF + ((int)light1RawVolt - light1VoltLPF)/switchOffLPF\_N;

light1VoltHPFPrev = light1VoltHPF;

light1VoltHPF = (light1VoltLPF - (int)light1RawVolt);

if (light1VoltHPF - light1VoltHPFPrev > switchOffLPF\_Thd) {

light1SwitchOffDelay = switchOffDelay;

ledState1=LOW;

}

if(light1SwitchOffDelay>0) light1SwitchOffDelay--;

light2RawDec = analogRead(sensePin2);

light2RawVolt = map(light2RawDec, 0, 1023, 500, 0); // 5V --> 1023

if((light2RawVolt>turnOnThreshold)&&(light2SwitchOffDelay==0)) ledState2=HIGH;

//if(light1RawVolt<turnOffThreshold) ledState1=LOW;

light2VoltLPF = light2VoltLPF + ((int)light2RawVolt - light2VoltLPF)/switchOffLPF\_N;

light2VoltHPFPrev = light2VoltHPF;

light2VoltHPF = (light2VoltLPF - (int)light2RawVolt);

if (light2VoltHPF - light2VoltHPFPrev > switchOffLPF\_Thd) {

light2SwitchOffDelay = switchOffDelay;

ledState2=LOW;

}

if(light2SwitchOffDelay>0) light2SwitchOffDelay--;

light3RawDec = analogRead(sensePin3);

light3RawVolt = map(light3RawDec, 0, 1023, 500, 0); // 5V --> 1023

if((light3RawVolt>turnOnThreshold)&&(light3SwitchOffDelay==0)) ledState3=HIGH;

//if(light1RawVolt<turnOffThreshold) ledState1=LOW;

light3VoltLPF = light3VoltLPF + ((int)light3RawVolt - light3VoltLPF)/switchOffLPF\_N;

light3VoltHPFPrev = light3VoltHPF;

light3VoltHPF = (light3VoltLPF - (int)light3RawVolt);

if (light3VoltHPF - light3VoltHPFPrev > switchOffLPF\_Thd) {

light3SwitchOffDelay = switchOffDelay;

ledState3=LOW;

}

if(light3SwitchOffDelay>0) light3SwitchOffDelay--;

light4RawDec = analogRead(sensePin4);

light4RawVolt = map(light4RawDec, 0, 1023, 500, 0); // 5V --> 1023

if((light4RawVolt>turnOnThreshold)&&(light4SwitchOffDelay==0)) ledState4=HIGH;

//if(light1RawVolt<turnOffThreshold) ledState1=LOW;

light4VoltLPF = light4VoltLPF + ((int)light4RawVolt - light4VoltLPF)/switchOffLPF\_N;

light4VoltHPFPrev = light4VoltHPF;

light4VoltHPF = (light4VoltLPF - (int)light4RawVolt);

if (light4VoltHPF - light4VoltHPFPrev > switchOffLPF\_Thd) {

light4SwitchOffDelay = switchOffDelay;

ledState4=LOW;

}

if(light4SwitchOffDelay>0) light4SwitchOffDelay--;

light5RawDec = analogRead(sensePin5);

light5RawVolt = map(light5RawDec, 0, 1023, 500, 0); // 5V --> 1023

if((light5RawVolt>turnOnThreshold)&&(light5SwitchOffDelay==0)) ledState5=HIGH;

//if(light1RawVolt<turnOffThreshold) ledState1=LOW;

light5VoltLPF = light5VoltLPF + ((int)light5RawVolt - light5VoltLPF)/switchOffLPF\_N;

light5VoltHPFPrev = light5VoltHPF;

light5VoltHPF = (light5VoltLPF - (int)light5RawVolt);

if (light5VoltHPF - light5VoltHPFPrev > switchOffLPF\_Thd) {

light5SwitchOffDelay = switchOffDelay;

ledState5=LOW;

}

if(light5SwitchOffDelay>0) light5SwitchOffDelay--;

// Serial.print(light1RawVolt);

// Serial.print(" ");

// Serial.print(light1VoltLPF);

// Serial.print(" ");

// Serial.println(light1VoltHPF);

// create random flicker

if(ledState1) {

if(random(100)<flickerPercent) digitalWrite(ledPin1, LOW);

}

if(ledState2) {

if(random(100)<flickerPercent) digitalWrite(ledPin2, LOW);

}

if(ledState3) {

if(random(100)<flickerPercent) digitalWrite(ledPin3, LOW);

}

if(ledState4) {

if(random(100)<flickerPercent) digitalWrite(ledPin4, LOW);

}

if(ledState5) {

if(random(100)<flickerPercent) digitalWrite(ledPin5, LOW);

}

// wait for LED's to go off & then restore

delay(flickerDuration);

digitalWrite(ledPin1, ledState1);

digitalWrite(ledPin2, ledState2);

digitalWrite(ledPin3, ledState3);

digitalWrite(ledPin4, ledState4);

digitalWrite(ledPin5, ledState5);

// give 5 mins to light the lamp

if((noOfMins>=minOnTimeMins)&&(ledState1+ledState2+ledState3+ledState4+ledState5==0))

digitalWrite(stayAlivePin, LOW);

// if lamp is not turned off then auto shutdown after 3 hours

if(noOfMins>=maxOnTimeMins)

digitalWrite(stayAlivePin, LOW);

}

}