#include "arduinoFFT.h"

#define SAMPLES 128             //Must be a power of 2
#define SAMPLING_FREQUENCY 1000 //Hz, must be less than 10000 due to ADC
int numberOfLed = 12;

int maxLow1 = 100;  //range of each LED in low frequency bar
int maxLow2 = 200;
int maxLow3 = 600;
int maxLow4 = 800;

int maxMid1 = 100;  //range of each LED in mid frequency bar
int maxMid2 = 200;
int maxMid3 = 600;
int maxMid4 = 800;

int maxHigh1 = 100; //range of each LED in high frequency bar
int maxHigh2 = 200;
int maxHigh3 = 600;
int maxHigh4 = 800;

int LowRange = 0;  //number of LEDs that will light up per bar, with values 0,1,2,3,4
int MidRange = 0;  //0,1,2,3,4
int HighRange = 0; //0,1,2,3,4
long int currentTime;

arduinoFFT FFT = arduinoFFT(); // using the arduinoFFT
library to calculate the FFT

unsigned int sampling_period_us; // delay variable for
analog reads
unsigned long microseconds;

double vReal[SAMPLES]; // Array of frequency samples
double vImag[SAMPLES];

void setup() {
    // LED start from Digital Pin 27 to 27 + NumberOfLed
    for (int i = 27; i < numberOfLed + 27; i++) {
        pinMode(i, OUTPUT);
    }

    Serial.begin(115200);

    sampling_period_us = round(1000000 * (1.0 /
SAMPLING_FREQUENCY)); // calculates the desired
sampling period
}

void loop() {
    currentTime = millis(); // time elapsed since
beginning of program

    /*SAMPLING*/
for (int i = 0; i < SAMPLES; i++)
{
    microseconds = micros(); // Overflows after around 70 minutes!

    vReal[i] = analogRead(0); // reads data from analog pin 0
    Serial.println(vReal[i]); // Send data via serial output for Matlab
    vImag[i] = 0;

    while (micros() < (microseconds + sampling_period_us)) { // delays the program with a sampling period period of time
    }
}

/* Running the FFT */

// Prevents smearing by using a weighting function
FFT.Windowing(vReal, SAMPLES, FFT_WIN_TYP_HAMMING, FFT_FORWARD);

// Computes the FFT calculations
FFT.Compute(vReal, vImag, SAMPLES, FFT_FORWARD);

// Returns the complex numbers as magnitudes
FFT.ComplexToMagnitude(vReal, vImag, SAMPLES);

// Returns the major peak values for the magnitude values
double peak = FFT.MajorPeak(vReal, SAMPLES, SAMPLING_FREQUENCY);

/* Compressing Data into 3 bins */
int maxLow = 0;
int maxMid = 0;
int maxHigh = 0;

/* find max value for each bin */
for (int i = 0; i < 19; i++) {
    maxLow = max(maxLow, vReal[i + 10]);
}
for (int i = 0; i < 18; i++) {
    maxMid = max(maxMid, vReal[i + 29]);
}
for (int i = 0; i < 17; i++) {
    maxHigh = max(maxHigh, vReal[i + 47]);
}

/* For debugging */
//    Serial.print("MaxLow: ");
//    Serial.println(maxLow);
//    Serial.print("MaxMid: ");
//    Serial.println(maxMid);
//    Serial.print("MaxHigh: ");
//    Serial.println(maxHigh);

/* Takes the maximum amplitude of the lower frequencies and converts it into LED range for the first four LEDs */
if (maxLow1 > maxLow) {
    LowRange = 0;
}
else if (maxLow2 > maxLow) {
    LowRange = 1;
}
else if (maxLow3 > maxLow) {
    LowRange = 2;
}
else if (maxLow4 > maxLow) {
    LowRange = 3;
}
else
    LowRange = 4;

/* Takes the maximum amplitude of the mid range frequencies
   Converts max-es into LED range for the second four LEDs
   */
if (maxMid1 > maxMid) {
    MidRange = 0;
}
else if (maxMid2 > maxMid) {
    MidRange = 1;
}
else if (maxMid3 > maxMid) {
    MidRange = 2;
}
else if (maxMid4 > maxMid) {
    MidRange = 3;
/* Takes the maximum amplitude of the higher frequencies
   Converts max-es into LED range for the last four LEDs
 */

if (maxHigh1 > maxHigh) {
    HighRange = 0;
}
else if (maxHigh2 > maxHigh) {
    HighRange = 1;
}
else if (maxHigh3 > maxHigh) {
    HighRange = 2;
}
else if (maxHigh4 > maxHigh) {
    HighRange = 3;
}
else
    HighRange = 4;

/* Sets the digital pin output corresponding to the range
The data has been compressed to 3 bin values: LowRange, MidRange, and HighRange */
setLed(LowRange, MidRange, HighRange);
/* Plots a graph of the FFT output.
   Commented out because not necessary for this program */

    //    for(int i=0; i<(SAMPLES/2); i++)
    //    {
    //        /*View all these three lines in serial
terminal to see which frequencies has which amplitudes*/
    //
    //        //Serial.print((i * 1.0 *
SAMPLING_FREQUENCY) / SAMPLES, 1);
    //        //Serial.print(" ");
    //        Serial.println(vReal[i], 1);    //View only
this line in serial plotter to visualize the bins
    //    }

/* PRINT RESULTS FOR DEBUGGING */
//Serial.println("time: ");
//Serial.println(millis() - currentTime);

/* Sets the digital pins to HIGH or LOW values (turns
   LEDs ON or OFF).
   Uses cases to distinguish between the different
   frequencies */
void setLed(int Low, int Mid, int High) {
    /* Low frequencies */
    switch (Low) {
case 0: // all OFF
    digitalWrite(24, LOW);
    digitalWrite(25, LOW);
    digitalWrite(26, LOW);
    digitalWrite(27, LOW);
    break;

case 1: // one ON
    digitalWrite(24, HIGH);
    digitalWrite(25, LOW);
    digitalWrite(26, LOW);
    digitalWrite(27, LOW);
    break;

case 2: // two ON
    digitalWrite(24, HIGH);
    digitalWrite(25, HIGH);
    digitalWrite(26, LOW);
    digitalWrite(27, LOW);

case 3: // three ON
    digitalWrite(24, HIGH);
    digitalWrite(25, HIGH);
    digitalWrite(26, HIGH);
    digitalWrite(27, LOW);

default: //optional all ON
    digitalWrite(24, HIGH);
    digitalWrite(25, HIGH);
    digitalWrite(26, HIGH);
    digitalWrite(27, HIGH);
}

/* Mid frequencies */
switch (Mid) {

case 0: // all OFF
    digitalWrite(28, LOW);
    digitalWrite(29, LOW);
    digitalWrite(30, LOW);
    digitalWrite(31, LOW);
    break;

case 1: // one ON
    digitalWrite(28, HIGH);
    digitalWrite(29, LOW);
    digitalWrite(30, LOW);
    digitalWrite(31, LOW);
    break;

case 2: // two ON
    digitalWrite(28, HIGH);
    digitalWrite(29, HIGH);
    digitalWrite(30, LOW);
    digitalWrite(31, LOW);

case 3: // three ON
    digitalWrite(28, HIGH);
    digitalWrite(29, HIGH);
    digitalWrite(30, HIGH);
    digitalWrite(31, LOW);

default: //optional all ON
    digitalWrite(28, HIGH);
    digitalWrite(29, HIGH);
    digitalWrite(30, HIGH);
    digitalWrite(31, HIGH);
}

/* High frequencies */
switch (High) {
case 0: // all OFF
    digitalWrite(32, LOW);
digitalWrite(33, LOW);
digitalWrite(34, LOW);
digitalWrite(35, LOW);
    break;
case 1: // one ON
    digitalWrite(32, HIGH);
digitalWrite(33, LOW);
digitalWrite(34, LOW);
digitalWrite(35, LOW);
    break;
case 2: // two ON
    digitalWrite(32, HIGH);
digitalWrite(33, HIGH);
digitalWrite(34, LOW);
digitalWrite(35, LOW);
case 3: // three ON
    digitalWrite(32, HIGH);
digitalWrite(33, HIGH);
digitalWrite(34, HIGH);
digitalWrite(35, LOW);
default: //optional all ON
    digitalWrite(32, HIGH);
digitalWrite(33, HIGH);
digitalWrite(34, HIGH);
digitalWrite(35, HIGH);