Processing Code

```
import processing.opengl.*;
import processing.serial.*;

Serial serial;
int WIDTH = 1440;
int analog_in = 0;
PFont font;

void setup() {
  size(WIDTH,900,OPENGL);
  font = loadFont("Helvetica-Bold-20.vlw");
  textFont(font,20);
  textAlign(TOP,LEFT);
  serial = new Serial(this, Serial.list()[0], 115200);
  background(255);
  smooth();
}

int val = 0;

void draw() {
  background(255);
  fill(0);
  stroke(0);
  while (serial.available() > 0) {
    int c = serial.read();
    if (c >= '0' && c <= '9') {
      val = val * 10 + ((int)c - (int)'0');
    } else if (c == '\n') {
      next(val);
      val = 0;
    }
  }


  text("input: " + analog_in,40,140);

  renderBuffer();
  text("max-noise: " + maxInBuffer(),340,90);
  text("min-noise: " + minInBuffer(),340,40);
  text("initial min: " + maxInBuffer2(),40,40);
  text("initial max: " + minInBuffer2(),40,90);

}

float minInBuffer() {
  float min = 99999;
  for (int i = 0; i < WIDTH; i++) {
    if (min > buffer[i])
    min = buffer[i];
  }
  return min;
}

float maxInBuffer() {
  float max = 0;
  for (int i = 0; i < WIDTH; i++) {
    if (max < buffer[i])
    max = buffer[i];
  }
  return max;
}

float minInBuffer2() {
  float min = 99999;
  for (int i = 0; i < WIDTH; i++) {
    if (min > buffer2[i])
    min = buffer2[i];
  }
  return min;
}

float maxInBuffer2() {
  float max = 0;
  for (int i = 0; i < WIDTH; i++) {
    if (max < buffer2[i])
    max = buffer2[i];
  }
  return max;
}

int index = 0;
float buffer[] = new float[WIDTH];
float buffer2[] =  new float[WIDTH];

float min = 509.0;
float max = 506.0;
float m = 2.0/(max - min);
float b = 1.0 - m * max;

void next(int val) {

   analog_in = val;
float acc = (analog_in - 508)/2;



  //println(val);
  if (pause)
    return;

  float gz = m * val + b;
  //float gz = 5.f*val*1000.f/(8.f*1024.f);
  buffer[index] = gz;
  buffer2[index] = acc;
  index = (index + 1) % WIDTH;
}

boolean pause = false;

void keyPressed() {
  if (key == 's') {
    saveFrame("snapshot.png");
    pause = true;
  }
  if (key == 'p' || key == ' ') {
    pause = !pause;
  }
}

void renderBuffer() {
  int in = WIDTH-1;
  float last = 0;
  for (int i = index; i < WIDTH; i++, in--) {
    line(in,height/2+buffer[i],in+1,last);
    last = height/2+buffer[i];
  }
  for (int i = 0; i < index; i++, in--) {
    line(in,height/2+buffer[i],in+1,last);
    last = height/2+buffer[i];
  }
}
```