CÓDIGO ARDUINO Y PROCESSING OSCILOSCOPIO

```
#define CHANNEL_A_PIN 0
void setup()
{
Serial.begin(115200); //velocidad máxima para comunicarse con el ordenador
}
void loop()
{
int value =  analogRead(CHANNEL_A_PIN);
value = (value >> 2)& 0xFF;
Serial.print(value);
}
```

```
import processing.serial.*;  //importamos la librería serial


Serial MyPort;  //nombre del puerto serie
int value;  //desde el puerto serie
int[] valores;
float zoom;




void setup()  //Función de arranque
{
  size(1600, 720);         //Tamaño de la ventana en el que se dispondrá la gráfica
  MyPort = new Serial(this, Serial.list()[2], 9600);
  valores = new int[width];
}
int valorbyte() {   //Esta función devuelve un número entre 0 y 255 que se refiere al próximo byte esperando en el buffer
  int valor;
  valor = -1;
  while (MyPort.available() >= 3) {        //En primer lugar comprobamos si hay espacio libre
    if (MyPort.read() == 0xff) {
      //valor= myPort.read( < < 8) | (myPort.read());    //Al igual que en el código de arduino cogemos ignoramos los bytes que no sean los últimos ocho
      valor = (MyPort.read() << 8) | (MyPort.read());
    }


  }
```

```
   return valor;
}




void Grid() {
  stroke(250, 0, 0);   //Color
  line(0, height/2, width, height/2);  //Define los limites del grid, se extiende por todo el ancho del display y por la mitad de la altura
}


int coordenadaY(int value) {   //Transformamos value que nos  llega desde el puerto  para representarlo en el eje y
  return (int)(height - value / 1023.0f * (height - 1));
}


void dibujarGrafico() {        //Representamos el gráfico con esta función
  stroke(200);   //Color de los bordes
  int i;
  i = valores.length - width;
  int x0 = 0;
  int y0 = coordenadaY(valores[i]); //Llamada a la función anterior, en la que mandamos los valores que nos llegan desde el  puerto ya adaptados
  for (int j=1; j<width; j++) {
    j++;
    int x1 = (int) (j * (width-1) / (width-1));
    int y1 = coordenadaY (valores[i]);
    line(x0, y0, x1, y1);  //Se dibujará una recta entre los puntos X0 e Y0
    x0 = x1;       //lógicamente el nuevo valor de X0 es el que tenía X1
    y0 = y1;      //lo mismo que en el caso de x
  }
}
```

```
void siguienteValor(int value) {      //Como indica su nombre, se encarga de dar el valor siguiente para la grafica
  int i;
  for (i=0; i<width-1; i++)
    valores[i] = valores[i+1];
  valores[width-1] = value;
}

void grafico()      //función definitiva con la que ya se une todas las funciones que se encargaban de diseño de display con las que representaban las rectas con los valores captados en arduino
{
  background(51);  //Color del fondo de pantalla de la ventana de processing
  Grid();          //Llamada a la función que define el display
  value = valorbyte ();   //el int value esta declarado al principio del main, no confundir con valor
  if (value != -1) {
    siguienteValor(value);      //Mientras haya un siguiente valor, llamamos a la función que hemos declarado justo antes
  }
  dibujarGrafico();
}
```