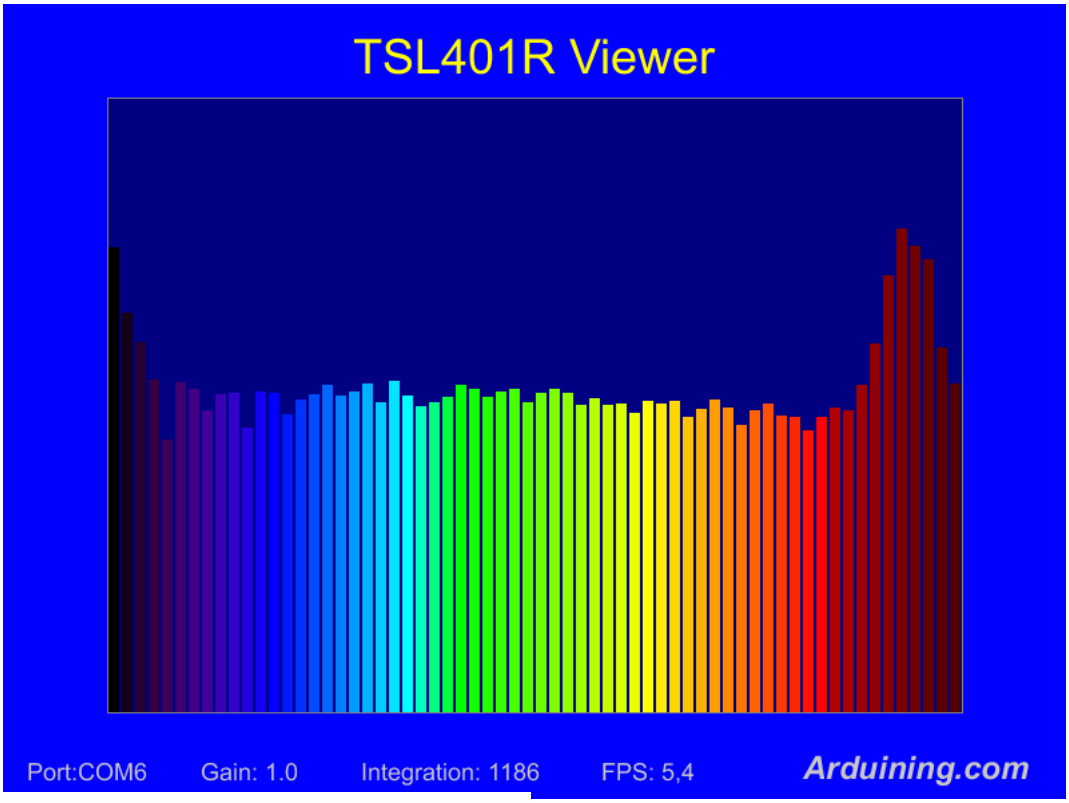
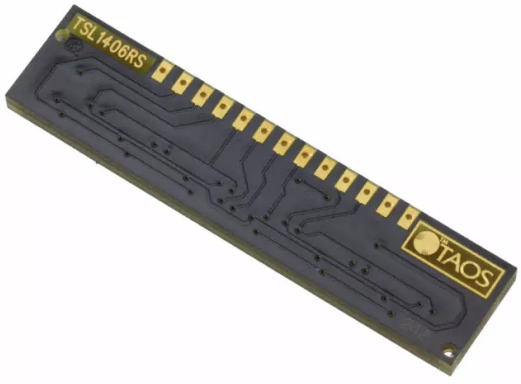
Compte-rendu

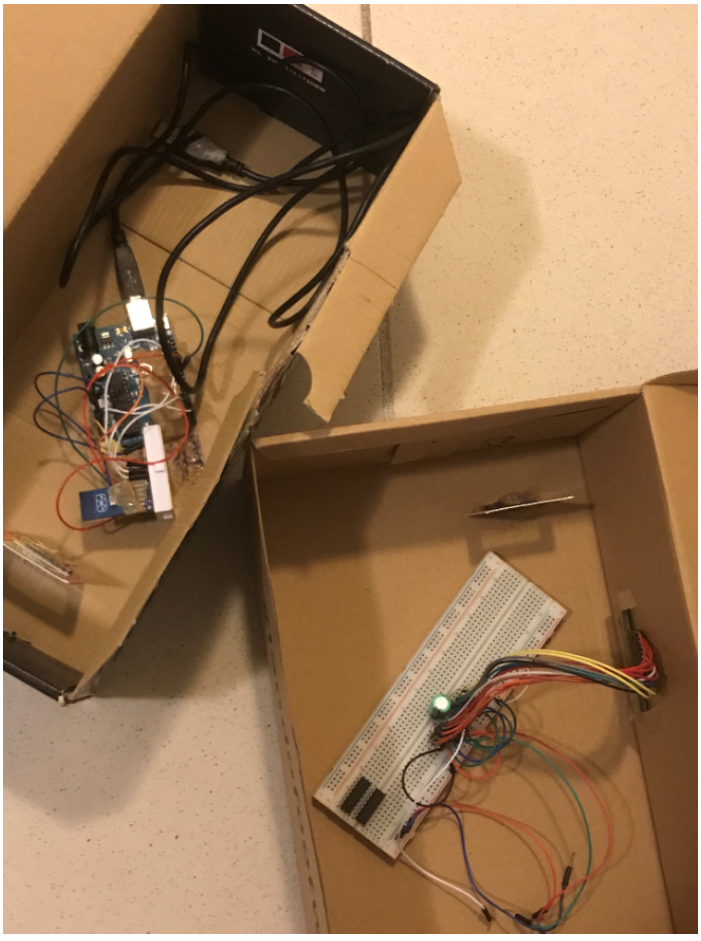
Spectrophotomètre





Par Hugo Marquette
By Hugo Marquette

Sommaire



Présentation

Fr :

Ce projet est comme son nom l'indique, un spectrophotomètre. Ce dernier va permettre l'analyse d'un spectre lumineux issu d'une lampe à incandescence. Tout ce projet est réalisable grâce à Arduino. Une lumière blanche est envoyé sur réseau de diffraction qui va « exploser » cette dernière en un ensemble de couleur multicolore. Cet ensemble de couleur multicolore va constituer un spectre de lumière. On pourra ainsi l'afficher en se servant d'un dispositif TSL1401R ou TSL1406R et d'Arduino. Les données récupérées par l'Arduino seront transmises par le biais d'un d'un module Bluetooth sur une application Androïde d'un appareil externe.

En :

This project is as its name indicates it, a spectrophotometer. This one is going to allow the analysis of a bright spectre stemming from a light bulb. All this project is practicable thanks to Arduino. A white light is sent on linear diffraction gratting which is going "to explode" this light in a set of multicolored color. This whole multicolored color is going to establish a spectre of light. We can so show it by means of a TSL1401R or TSL1406R and of Arduino. The data got back by Arduino will be transmitted thanks to a module Bluetooth on an application Android of an external device.

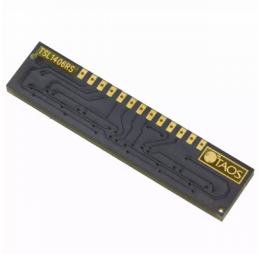
Objectifs initiaux :

-Créer un spectrophotomètre
-Utiliser pleinement les fonctionnalités d'Arduino
-Pouvoir transmettre des données en utilisant un module Bluetooth
-Pouvoir avancer d'un pas dans le domaine du professionnalisme de l'ingénieur.

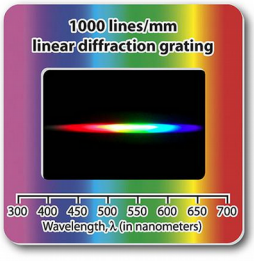
Matériel :

-2 boîtes de chaussures

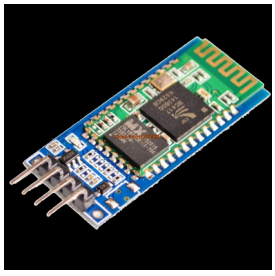
-1 TSL1401R
-1 TSL1406R



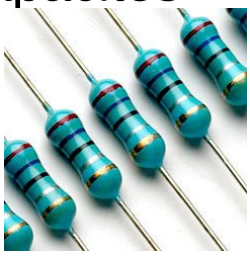
-2 réseaux de diffraction (500/1000 lines/mm)



-1 module HC-06



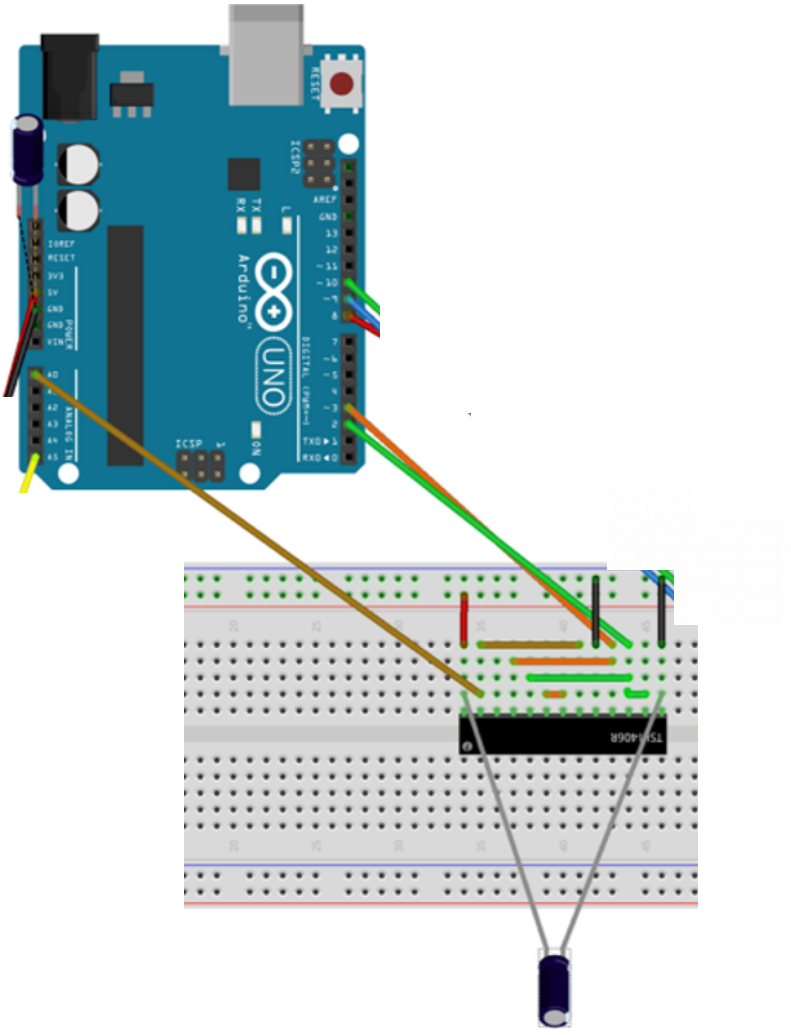
-Résistances/Capacités



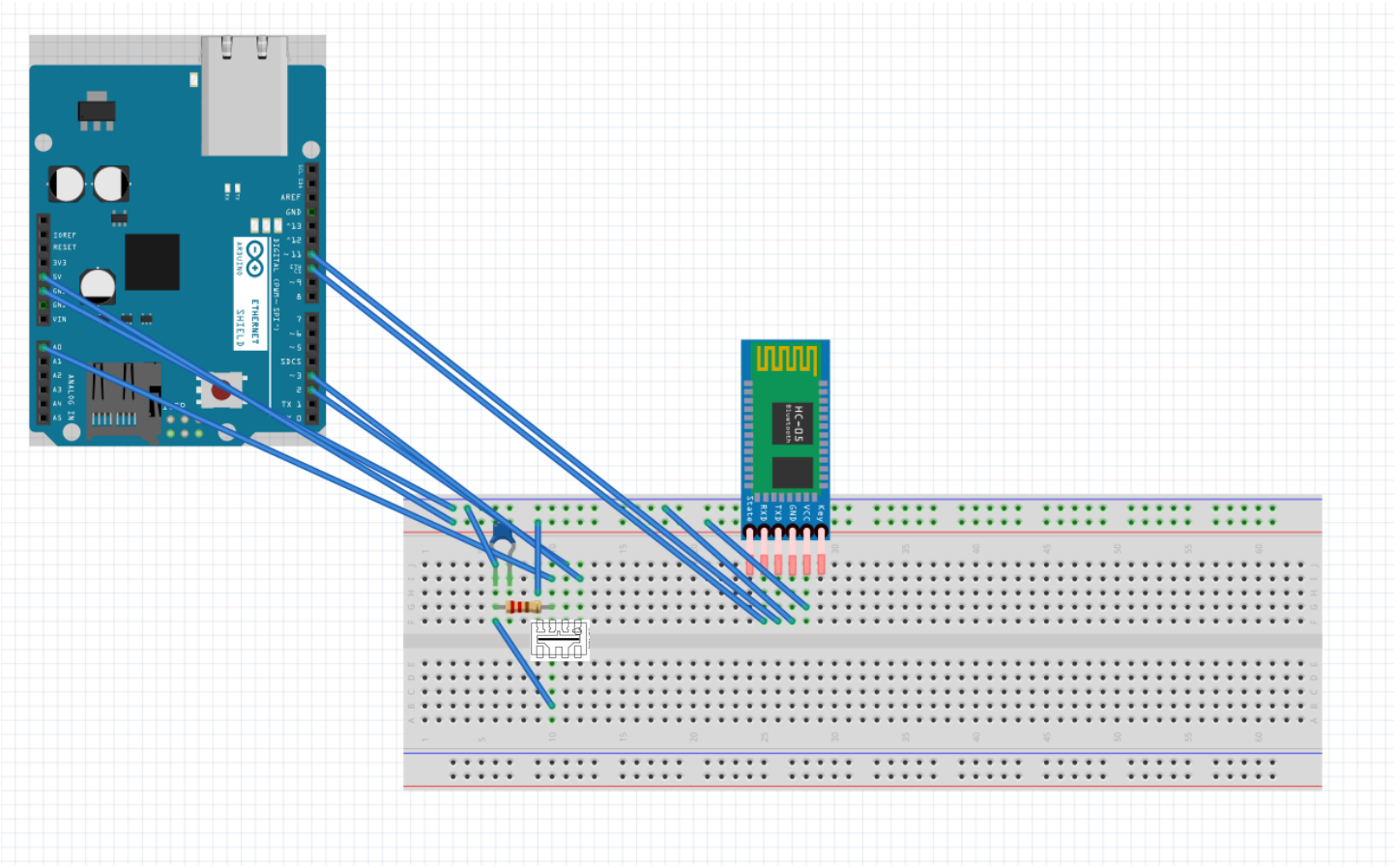
-Câbles mâles-mâles/mâle
-Scotch

Schémas:

-TSL1406R



-TSL1401R



Difficultés rencontrées/Solutions apportées:

-Problèmes de ports pour le programme Procesing
→ Pour remédier à cela, j'ai effectué un changement de carte en prenant une Arduino Uno
-Difficultés de compréhension des dispositifs utilisés
→ Pour remédier à cela, il m'a fallut étudier le datasheet de chaque dispositif et j'ai dû être aidé par un professeur d'électronique sur les semi-conducteurs en cycle ingénieur.

-Difficultés dans l’avancement du projet, un cerveau s’avère être souvent inadapté à des projets ayant des demandes intellectuelles un peu plus poussées. Cela a été mon cas ici, les demandes théoriques en optique et informatique étaient assez conséquentes. De plus, n’étant pas quelqu’un de manuel, j’ai eu beaucoup de mal à établir la boîte.
-Aucune solution trouvée pour envoyer les données récoltées par Arduino sur l’appareil externe.

Futurs (perspectives)

-Ce projet va rester en suspens après la deuxième année.
-Si plus tard, je décide de le continuer, il faudrait améliorer la connectivité avec le Bluetooth, pouvoir faire apparaître sur un appareil externe le même graphe que sur l’ordinateur
-Confectionner une meilleure boîte pour le spectrophotomètre
-Améliorer le code pour le TSL1406R afin d’obtenir un spectre plu facilement exploitable.

Codes :

1)TSL1401R

-Arduino

```
#define CLK     2
#define SI      3
#define VOUT    0   //pixel intensity in the analog channel 0
#define INTVAL  1   //integration time adjust in the analog channel 1.
#define PIXELS  64

int intDelay;       //Integration Period = (intDelay + 535 ) microseconds.
int Value;          //pixel intensity value.
//------------------ Send the intensity of the pixel serially -----------------
void readPixel(int pixel){
  digitalWrite(CLK, LOW);
  digitalWrite(SI, HIGH);
  digitalWrite(CLK, HIGH);
  digitalWrite(SI, LOW);


  for(int i=0;i<pixel; i++){    //Clock pulses before pixel reading
    digitalWrite(CLK, LOW);
    digitalWrite(CLK, HIGH);    //Select next pixel and reset integrator of actual pixel.
  }

  Value = analogRead(VOUT);

  Serial.print("i= ");
   if (pixel<10)Serial.print("0");
  Serial.print(pixel);
   Serial.print("   int= ");
   if (Value<100) Serial.print("0");
  Serial.println(Value);
//  delay(8);       //used with 9600 bauds
  delay(1);         //used with 115200 bauds
  for(int i=0;i<=(PIXELS-pixel); i++){    //Clock pulses after pixel reading.
    digitalWrite(CLK, LOW);
    digitalWrite(CLK, HIGH);    //Select next pixel and reset integrator of actual pixel.
  }
 }
void loop(){
  intDelay=1+analogRead(INTVAL)*10;  //read integration time from potentiometer.
  Serial.print(">");                 //Mark the start of a new frame
  Serial.println(intDelay + 535);    //Send Integration Period in microseconds.
// delay(8);      // used with 9600 bauds
  delay(2);      // used with 115200 bauds

  readPixel(0);                      //the first reading will be discarded.
 delayMicroseconds(intDelay);

  for(int i=0;i<PIXELS;i++){
    readPixel(i);
    delayMicroseconds(intDelay);     //Delay added to the integration period.
  }
 // delay(100);
}
```

-Procesing

```
//                Labels
String    heading = "TSL401R Viewer";
String    credit  = "Arduining.com";
//------------------------- Graph dimensions (constants) ----------------------
//System variables width and height are defined with size(width,height) in setup().

int       leftMargin=   80;   //from Graph to the left margin of the window
int       rightMargin=  80;   //from Graph to the right margin of the window
int       topMargin=    70;   //from Graph to the top margin of the window.
int       bottonMargin= 70;   //from Graph to the botton margin of the window.
int       rouge = 0 ;//;
int       vert = 0;//;
int       bleu = 0 ;// ;
//-------------------- Colors ---------------------------------------------------
final  color  frameColor =  color(0,0,255);    // Main window background color.(Blue)
final  color  graphBack =   color(0,0,128);    // Graphing window background color. (Dark b
final  color  borderColor=  color(128);        // Border of the graphing window.(grey)
final  color  barColor=     color(55,255,0);  // Vertical bars color. (yellow)
final  color  titleColor=   color(255,255,0);  // Title color. (yellow)
final  color  textColor=    color(200);        // Text values color. (grey)
//-------------------- Serial port library and variables ----------------------
import    processing.serial.*;

Serial    myPort;
boolean   COMOPENED = false;      // to flag that the COM port is opened.
final  char  LINE_FEED=10;        //mark the end of the frame.
final  int  numRecords=3;         //Number of data records per frame.
String[]  vals;                   //array to receibe data.
String    dataIn= null;           //string received
int       pixelCount= 64;
int       lightIntensity;
int       wavelength;
int       dataIn1;
int       Integration= 0;         //Integration period in milliseconds
```

[...]

```
//--------------------------------------------------------------------------------
void PortManager(){

//------------- List the available COM ports in this system ---------------------
  String[] portlist = Serial.list();  //Create a list of available ports.
  println("There are " + portlist.length + " COM ports:");
  //println(portlist);

//----------- If COM port in "settings.txt" is available, open it  -------------
  for (int i=0; i < portlist.length; i++) {
      if(comPortName.equals(portlist[i]) == true){
          myPort = new Serial(this, portlist[0], 115200); //open serial port.
          COMOPENED = true;
          println(portlist[i]+" opened");
      }
  }

//-------- Procedure when COM port is not available --------------------
  if(!COMOPENED){
      println(comPortName + " is not available");    //Anounce the problem
//      SelComPort();          //Procedure to select another port...
  }
}
```

2)TSL1406R

-Arduino

```
class TSL1406
{
public:
  byte pin_si, pin_clk, pin_analogOut;
  int exposureTime;

  TSL1406(byte pin_si, byte pin_clk, byte pin_ao) {
    this->pin_si = pin_si;
    this->pin_clk = pin_clk;
    this->pin_analogOut = pin_ao;

    pinMode(pin_si, OUTPUT);
    pinMode(pin_clk, OUTPUT);
    pinMode(pin_ao, INPUT);

    exposureTime = 5;
  }

  void clockPulse() {
    digitalWrite(pin_clk,1);
    digitalWrite(pin_clk,0);
  }

  void read(int* output)
  {
    digitalWrite(pin_si,1);
    clockPulse();
    digitalWrite(pin_si,0);
```

-Procesing

```
Serial port; // The serial port

int exposureTime = 1; // 1 ms
float updateSpeed = 2000; // 2000 ms
int lastSpectrumUpdate=0; // keep track of time in this variable

// LED toggle variables
boolean ledState=true;
boolean rledState=false;
boolean gledState=false;
boolean bledState=false;

// Variables containing the fonts
PFont defaultFont;
PFont titleFont;

// Serial communication buffer
String buffer;

//Mathematically the spectrometer should operate in the 400-900 nm w
//For this specific box, pixel one corresponds with 900 nm, pixel 76
//But should be calibrated..
int SpectrumSize = 768;
float[] rawSpectrumData, correctedSpectrumData;
float[] darkReadout, whiteReadout;
int spectrumValueIndex=0;
int spectraCount=0;
```