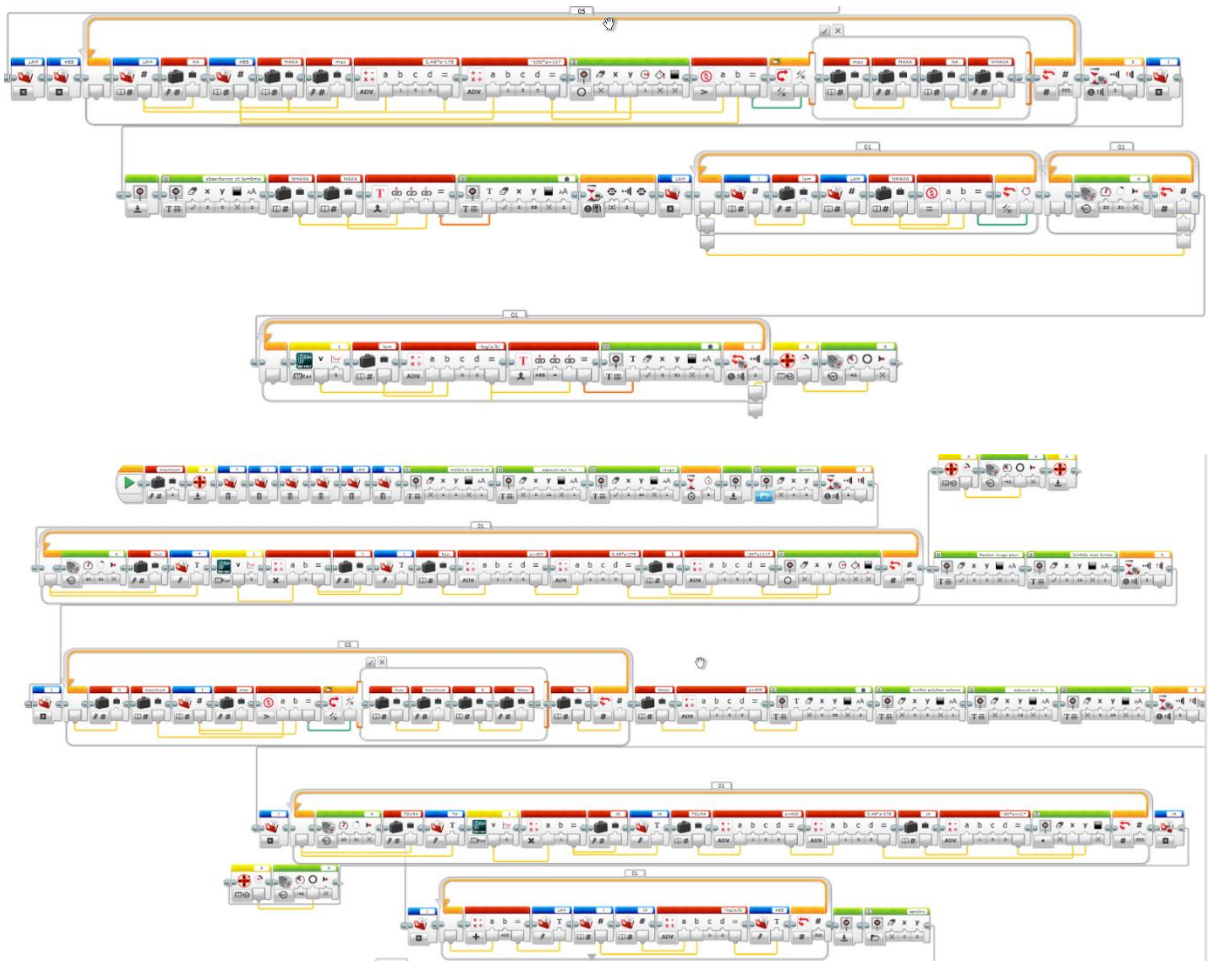
ANNEXE DU PROGRAMME QUI UTILISAIT NOTRE PREMIERE VERSION DU SPECTROFACILE AVEC DES LEGO MINDSTORMS.



ANNEXE : UNE BONNE PARTIE DU PROGRAMME QU'UTILISE ACTUELLEMENT NOTRE SPECTROFACILE :

```
// lecture parallèle d'un capteur CCD TSL1402R (de 256 photodiodes) pour exploiter l'absorbance d'une solution colorée entre 400 et 700 nm
//------------------------------------------------------------------------------
#include <TFT.h> // bibliothèques nécessaires
#include <SPI.h>
#include <math.h>
// définition des sorties
#define cs   10
#define dc   9
#define rst 8
// paramètre pour changer la fréquence de calcul du microcontrôleur :
const unsigned char PS_32 = (1 << ADPS2) | (1 << ADPS0);
const unsigned char PS_128 = (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0);
int CLKpin = 4;  // signal de décharge (clock pulse)
int SIpin = 5;  // SI (serial-input) entrée série
int AOpin1 = 1; // sortie analogique 1
int AOpin2 = 2; // sortie analogique 2
int IntArray[256];// stockage des valeurs de chaque photodiodes dans une liste sous forme d'entiers de 0 à 1023 pour le solvant
int IntArrayab[256];// stockage des valeurs de chaque photodiodes dans une liste sous forme d'entiers de 0 à 1023 pour la solution
int a = 0; // paramètre de boucle
int spectobscur = 40; //valeur donnée par le capteur quand il est dans l'obscurité
float I0vmaxi = 0;// vrai maximum I0
int LmaxI0 = 0; // Lambda max de I0
float amax = 0; //Absorbance max en indice
float vamax = 0; // Absorbance max absolue
float absor = 0; // l'absorbance en temps réel
float I0maxi = 0;
float Ic = 0;
float I0c = 0;
const int bouton = 12;
int etatBouton;
int imax = 0;
int facteurintegration = 0;
int xPos = 0; //avant 10
TFT TFTscreen = TFT(cs, dc, rst);
void setup()
{
 pinMode(bouton, INPUT); //le bouton est une entrée
 etatBouton = HIGH;
 Serial.begin(9600);
 TFTscreen.setTextSize(2);
```

```
TFTscreen.begin();
TFTscreen.background(0, 0, 0);
pinMode(CLKpin, OUTPUT);
pinMode(SIpin, OUTPUT);
ADCSRA &= ~PS_128;
ADCSRA |= PS_32;
for ( int i = 0; i < 7; i++ )
{
        digitalWrite(i, LOW);
}
for (int i = 0; i < 260; i++)
{
        ClockPulse();
}
digitalWrite(SIpin, HIGH);
ClockPulse();
digitalWrite(SIpin, LOW);
for (int i = 0; i < 260; i++)
{
 ClockPulse();
}
do {
 delayMicroseconds(facteurintegration); // c'est ici que l'on modifie le temps d'intégration
 digitalWrite(SIpin, HIGH);
        ClockPulse();
        digitalWrite(SIpin, LOW);
        for (int i = 0; i < 128; i++)
        {
        delayMicroseconds(20);
        IntArray[i] = analogRead(AOpin1) - spectobscur; // vidage des photodiodes auxquelles on
retranche spetrobscure
   IntArray[i + 128] = analogRead(AOpin2) - spectobscur;
        //Serial.print(IntArray[i]); Serial.print(";");
        //Serial.println("");
        ClockPulse();
        }
        digitalWrite(SIpin, HIGH);
        ClockPulse();
        digitalWrite(SIpin, LOW);
        for (int i = 0; i < 256; i++)
        {
   I0maxi = IntArray[i]; // lecture de la liste I0
   a = i * 1.172;
        if (I0maxi > I0vmaxi) { // condition sur valeur max de I0
        I0vmaxi = I0maxi; // stockage de la valeur max dans une variable
    LmaxI0 = a + 400;
        //Serial.println(IntArray[i]);
```

```
        }
        if (xPos >= 160) {
        xPos = 0;
        }
        else {
        xPos++;
        }
        }
        TFTscreen.background(0, 0, 0);
        for (int i = 0; i < 260; i++)
        {
        if (i == 18)
        {
        }
        ClockPulse();
        }
        facteurintegration = facteurintegration + 200;
 } while (I0vmaxi < 940 - spectobscur);
}
void loop()
{
 delayMicroseconds(facteurintegration);
 digitalWrite(SIpin, HIGH);
 ClockPulse();
 digitalWrite(SIpin, LOW);
 for (int i = 0; i < 128; i++)
 {
        delayMicroseconds(20);
        IntArray[i] = analogRead(AOpin1) - spectobscur;
   IntArray[i + 128] = analogRead(AOpin2) - spectobscur;
        ClockPulse();
 }
 digitalWrite(SIpin, HIGH);
 ClockPulse();
 digitalWrite(SIpin, LOW);
 for (int i = 0; i < 256; i++)
 {
        int drawHeight = map(IntArray[i], 0, 900 , 0, TFTscreen.height());
        float xPosdec = 0.625 * i;
        xPos = floor(xPosdec);
        TFTscreen.stroke(250, 250, 250);
        TFTscreen.line(xPos, TFTscreen.height() - drawHeight, xPos, TFTscreen.height() -
drawHeight + 1);
        I0maxi = IntArray[i]; // lecture de la liste I0
   a = i * 1.172;
        if (I0maxi > I0vmaxi) { // condition sur valeur max de I0
```