Rapport de Projet IMA4
P32: Robe augmentée

Réalisé par: MUZAKARE Brinda

YAN Xuelu

Encadré par: Mr Redon Xavier

Mr Boé Alexandre

Mr Vantroys Thomas

Année scolaire: 2018/2019

Remerciements

Nous adressons nos sincères remerciements à nos encadrants qui nous ont accompagné, aidé, et prodigué des conseils tout au long de ce projet.

On remercie également nos collègues les autres étudiants , et les autres professeurs qui nous ont aidé de près ou de loin pour l'avancée de notre projet.

Table des matières

I. Introduction

les vêtements font partie intégrantes du quotidien de l'homme. Le projet robe augmentée part d'un intemporel de la garde robe pour lui donné une autre dimension et un tout autre regard en lui ajoutant du dynamisme, de la lumière et de l'interaction avec celui qui la porte. Même si nous on a choisi une robe, ce projet peut être adaptée à tout types de vêtements, de tissus et d'accessoires.

Dans ce rapport , on va commencer par une analyse générale de notre projet en vous présentant les objectifs et le cahier des charges , son positionnement par rapport à la concurrence, ensuite présenter le matériels que nous avons utilisé et enfin décrire les grandes étapes de réalisations de notre projet qui sont la partie électronique et la partie programmation .

II. Présentation du projet

1. Objectif

La robe augmentée est une robe basique à laquelle on a ajouté de la luminosité et de l'éclat mais aussi qui s'illumine selon le rythme cardiaque. Cela veut dire que selon un rythme cardiaque normale , la robe s'illumine normalement mais dès qu'un certain seuil de la fréquence cardiaque est dépassé la luminosité de la robe change également.

Nous avons également la tâche de garantir le confort et la sécurité du porteur ainsi que l'élégance de la robe.

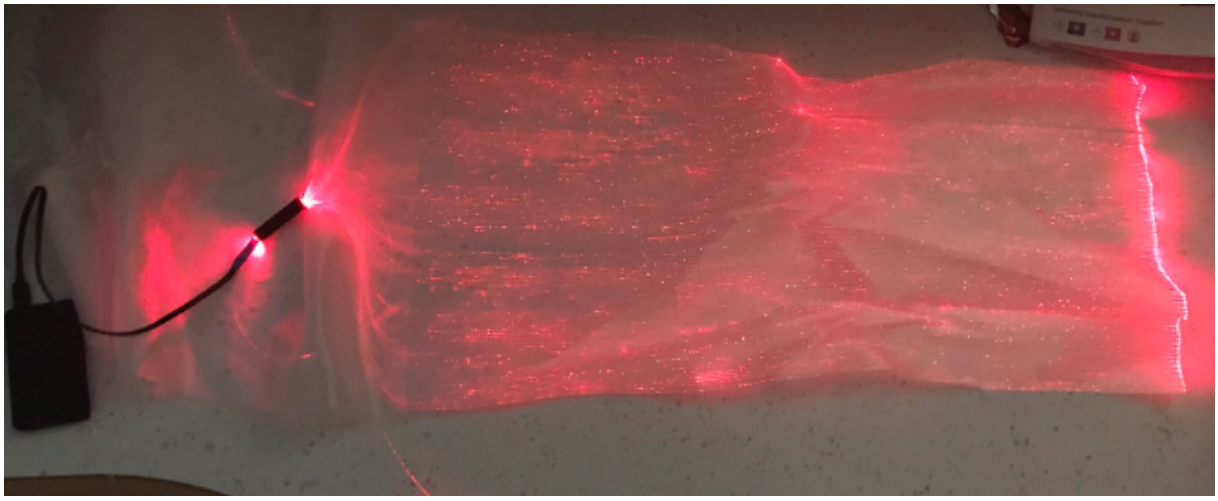
2. Analyse concurrentielle

Avant de commencer ce projet nous nous sommes renseigner sur ce qui ce fait déjà comme vêtement lumineux pour nous en inspirer et voir comment l'adapter sur notre projet.

A. **LUMIGRAM**

LUMIGRAM est un fabricant de produits textiles à base de tissus lumineux en fibres optiques. Le tissu lumineux est disponible dans une douzaine de couleurs.

Leurs domaines d'applications sont : L'habillement ,la décoration, l' architecture intérieure, les spectacles (costumes, scène). Nous avons pu nous procurer un échantillon de leur tissu est nous avons trouvé leur technique très simple. Il s'agit d'un tissu de fibre optique dont les extrémités sont groupées en anneau et éclairé par une led alimentée par une batterie rechargeable.



B. **Brochier Technologies**

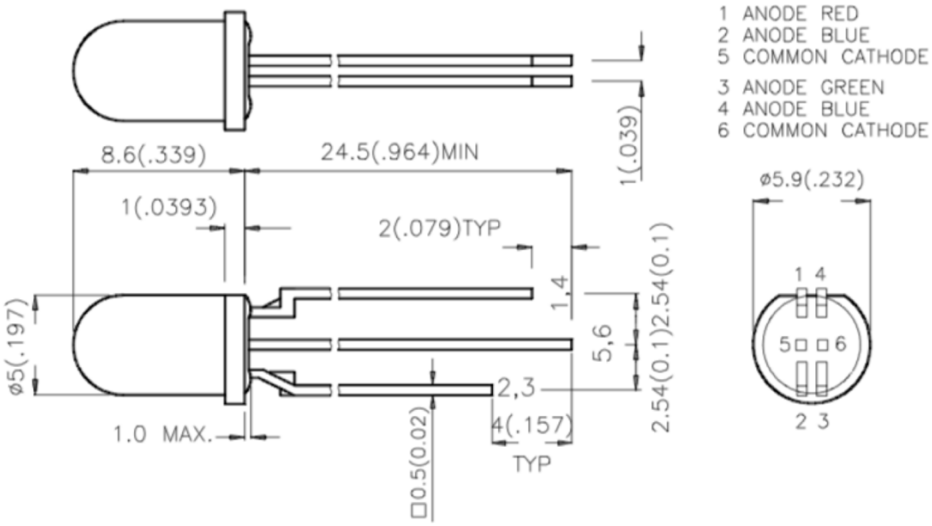
Brochier Technologies est une entreprise spécialisé dans les textiles lumineux servant avant tout à être vu pour des questions de sécurité, de publicité ou d'esthétique

Ils utilisent la technologie Lightex particulièrement adaptée pour les vêtements de sécurité. Dans le domaine de la santé ils ont mis au point une couverture lumineuse capable de traiter la jaunisse du nourrisson. Sur leur site on a pu voir qu'ils utilisent des technologies plus développées comme le pilotages des leds par bluetooth, WIFI, DMX, RF ,DALI, DES LEDS puissante d'une durée de vie de 50 000 à 100 000h.

Figure 1: couverture lumineuse capable de traiter la jaunisse d'un nourrisso

3. Choix du matériel

A. LED RGB



Principales caractéristiques

Couleur de LED: Rouge, Vert, Bleu

Montage LED: Traversant

Taille de l'ampoule: T-1 3/4 (5mm)

Angle, vision: 30°

Forme de lentille: Rond

Courant If / Couleur: R 30mA, G 25mA, B 30mA

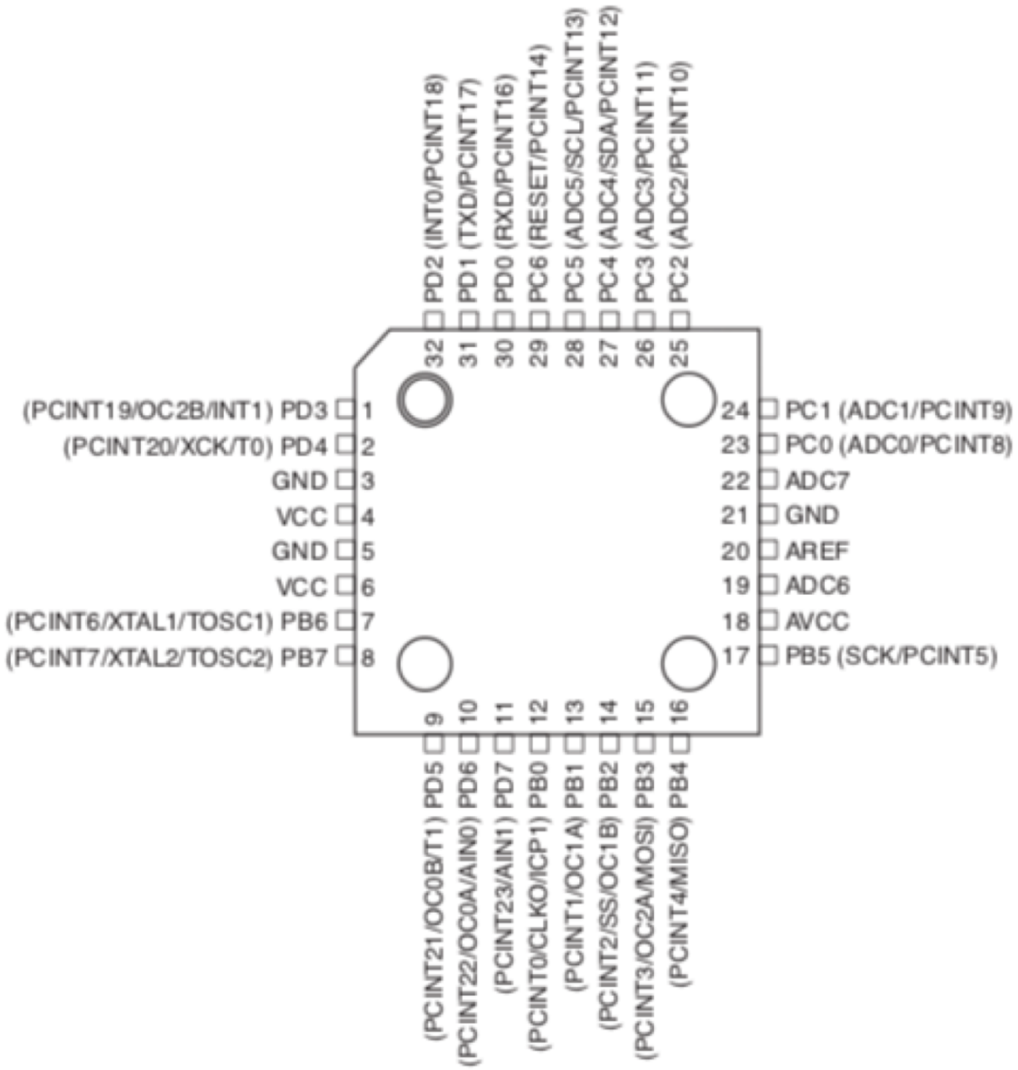
Intensité lumineuse / couleur: R 50mcd, G 30mcd, B 40mcd

Tension Vf / Couleur: R 2V, G 2V, B 4V

Longueur d'onde / Couleur: R 625nm, G 568nm, B 455nm

Ce sont des Leds RGB à cathode commune (Red, Green Blue). Nous avons choisi celle ci pour avoir un grand choix de couleur de la led mais aussi parce qu'elles ont une grande intensité lumineuse . A partir du rouge, du vert et du bleu , nous pouvons avoir huit combinaisons possibles de couleur. Elle possède également un angle de diffusion de 30°, ce qui nous convient pour notre projet , car ainsi le maximum de la lumière pourra être diffusé dans les fibres optiques.

B. ATmega328p



Principales caractéristiques

1) Microprocesseur AVR 8 bits haute performance et basse consommation.

2) Architecture RISC avancée

-131 instructions - la plupart du temps d'exécution des instructions est un cycle d'horloge unique

-32 registres de travail généraux 8 bits

- Travail statique complet

- Performances jusqu'à 20 MIPS à 20 MHz

- un multiplicateur matériel ne nécessitant que deux cycles d'horloge

3) Programme non volatile et mémoire de données

-32K octets de Flash programmable intégré au système

- Zone de code d'amorçage facultative avec bits de verrouillage indépendants

-1024 octets de mémoire EEPROM

-2K octets de mémoire SRAM intégrée

- Les bits de verrouillage peuvent être programmés pour chiffrer le programme utilisateur

4) Caractéristiques périphériques

- Deux compteurs / compteurs 8 bits avec fonctions de pré-décompression et de comparaison indépendantes

- compteur / minuterie 16 bits avec pré-échelisseur, fonction de comparaison et fonction de capture

- Compteur temps réel RTC avec oscillateur indépendant

- PWM à six canaux

-8 ADC 10 bits

- Série USART programmable

- Interface série SPI pouvant fonctionner en mode maître / esclave

- Interface série à 2 fils basée sur les octets

- Horloge de surveillance programmable avec oscillateur indépendant sur puce

- Comparateur analogique sur puce

- Un changement de niveau des broches peut provoquer une interruption et le réveil du MCU

5) Fonctions spéciales du microcontrôleur

- réinitialisation à la mise sous tension (POR) et détection de baisse de tension programmable (DBO)

- Oscillateur RC calibré sur puce

- Source d'interruption sur puce, hors puce

-6 modes de veille: mode veille, mode de suppression du bruit de l'ADC, mode d'économie d'énergie, mode mise hors tension, mode veille et mode veille étendu

6) I / O et emballage

-23 ports d'E / S programmables

-PDIP 28 broches, boîtier TQFP 32 broches, boîtier QFN / MLF 28 broches et boîtier QFN / MLF 32 broches

Statut de travail

Tension de travail

1.8 - 5.5V

Plage de température de fonctionnement:

-40 ° C à 85 ° C

Grade de vitesse de travail

0 - 20 MHz @ 1,8 - 5,5V

Consommation d'énergie ultra faible

-Mode normal:

1 MHz, 1,8 V, 25 ° C: 0,2 mA

-Mode de mise hors tension:

1,8 V, 0,1 μA

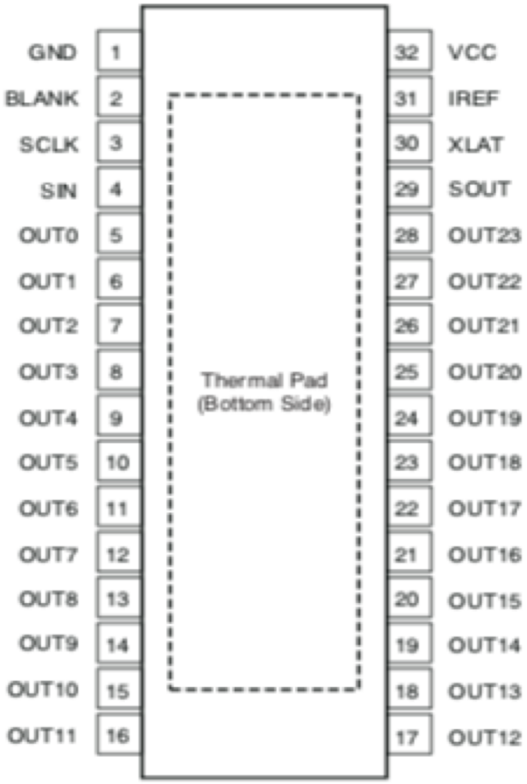
-Mode économie d'énergie:

1,8 V, 0,75 μA

Motif de la sélection

Tout d'abord, l'atmega328p est plus rapide que d'autres composants à la même fréquence. Ensuite , les périphériques atmega328p sont intéressants à utiliser dans les fonctions , tels que le timer, la conversion analogique numérique , que nous allons réutiliser par la suite dans la partie programmation. L' atmega328p a une bonne stabilité et une forte anti-ingérence. Enfin , le lecteur d'atmega328p des entrées/sorties est flexible, il est possible de choisir arbitrairement un collecteur ouvert, push-up, et ouvert.

C. **TLC5947**



Principales caractéristiques

Tension, entrée min: 3V

tension, entrée max..: 5.5V

Tension, sortie max..: 30V

Courant, sortie max:30mA

Fréquence de commutation: 30MHz

Nombre de canaux de sortie: 24Sortie(s)

Montage CI: CMS

Le TLC5947 est un circuit d'attaque de LED de puits à courant constant et à 24 canaux. Chaque canal est réglable individuellement avec 4096 pas modulés en largeur d'impulsion (PWM). Le contrôle PWM est répété automatiquement avec les données en niveaux de gris programmées (GS). Les données GS sont écrites via un port d'interface série. La valeur actuelle des 24 canaux est définie par une seule résistance externe.

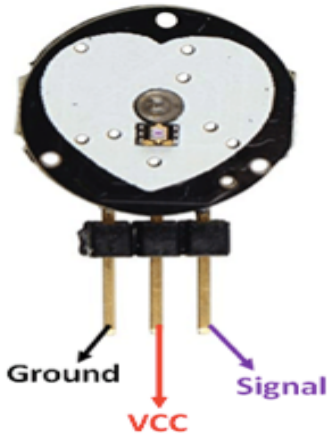
Le TLC5947 dispose d'une fonction d'arrêt thermique qui désactive.
Tous les pilotes de sortie en cas de surchauffe. Tous les pilotes de sortie redémarrent automatiquement lorsque la température revient à des conditions normales.

D. **Fibres optiques**

On a choisi les fibres optiques striée. c'est à dire avec des points lumineux tous les 2 cm le long de la fibre optique et possède un diamètre de 1mm. Au bout de ces fibres nous allons ajouter des kits de terminaisons trouvés sur le site sur lequel on a commandé les fibres optique pour recueillir la lumière diffuse au bout des fibres.

`

E. Capteur de fréquence cardiaque



Principe

Nous utilisons un capteur de pouls pour détecter le signal lié au pouls. Le fonctionnement du capteur de pulsation / rythme cardiaque est très simple. Le capteur a deux côtés, d'un côté la LED est placée avec un capteur de lumière ambiante et de l'autre côté nous avons des circuits. Ce circuit est responsable des travaux d'amplification et d'annulation du bruit. La LED située à l'avant du capteur est placée sur une veine de notre corps humain. Cela peut être soit le bout de votre doigt, soit le bout de votre oreille, mais il devrait être placé directement sur une veine. Maintenant, la LED émet une lumière qui va tomber directement sur la veine. Le sang ne circule dans les veines que lorsque le cœur pompe, alors si nous surveillons le flux de sang, nous pouvons également surveiller les battements du cœur. Si le flux sanguin est détecté, le capteur de lumière ambiante capte plus de lumière puisqu'il sera réfléchi par le sang. Ce changement mineur de la lumière reçue est analysé au fil du temps pour déterminer les battements de notre cœur.

Configuration

Pin Number	Pin Name	Wire Colour	Description
1	Ground	Black	Connected to the ground of the system
2	Vcc	Red	Connect to +5V or +3.3V supply voltage
3	Signal	Purple	Pulsating output signal.

III. Réalisation du projet

1. Partie électronique

A. Design de la carte électronique

On a commencé par réfléchir à la disposition des cartes électroniques sur la robe de façon a l'illuminer complètement. On a décidé de partir sur une carte principale sur laquelle on va mettre le microcontrôleur et 4 autres cartes qui porteront les leds et que nous disposerons des 4 côtés de la robe de façon à former une ceinture. En effet, l'ensemble des cartes electroniques avec les fibres optiques se portera comme une ceinture autour de la robe . Pour la réalisation des cartes on a utilisé Fritzing , un logiciel de conception de circuit imprimé gratuit et facile d'utilisation.

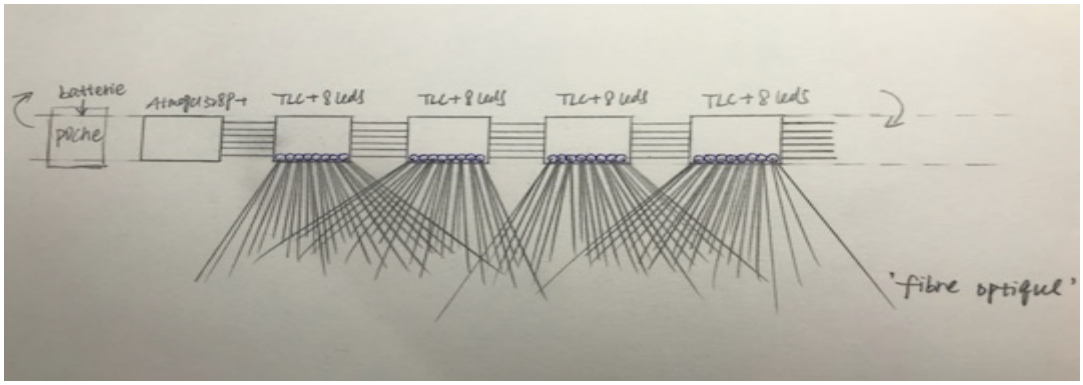


Figure 2: ceinture de leds

Figure 3: prototype de la robe à la fin du projet

B. La carte principale

Sur la carte principale se trouve 3 parties :

i. *Partie microcontrôleur:*

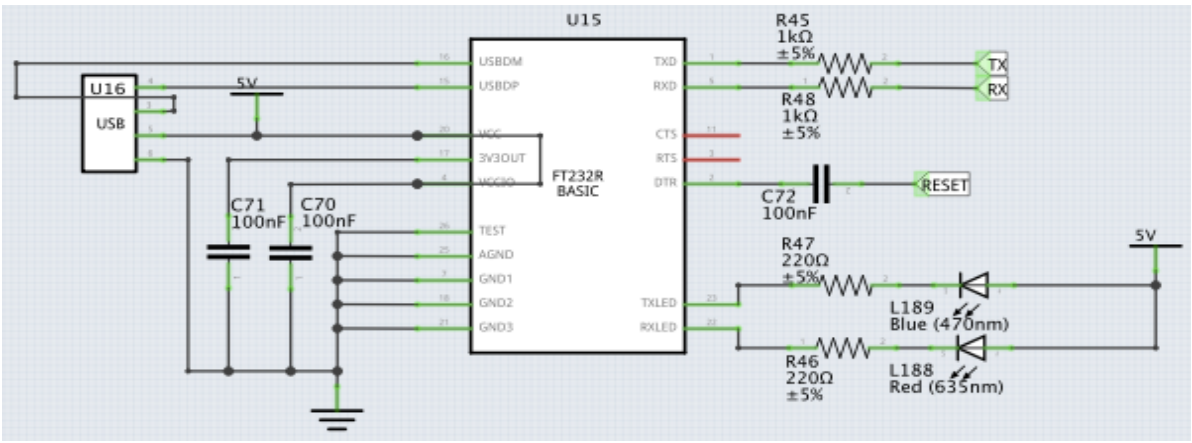
Cette partie est constitué d'un atmega 328p. Mais avant de le rendre fonctionnel on a besoin d'autres composant pour le configurer. Parmi les éléments de configuration de l'atmega on a un bus spi que l'on va utiliser dans un premier temps pour lancer un bootloader l'atmega328p avant de lui envoyer des instructions par liaison série. Cette communication passera par les interfaces SPI (MOSI, SCK ,MISO) que l'on retrouve aussi sur le microcontrôleur sur les port PB3,PB4,PB5.

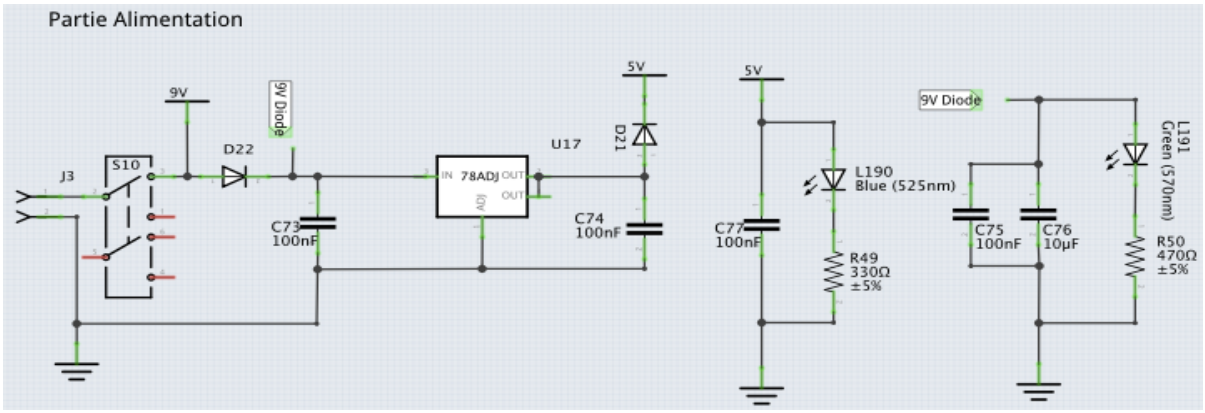
ii. *Partie USB:*

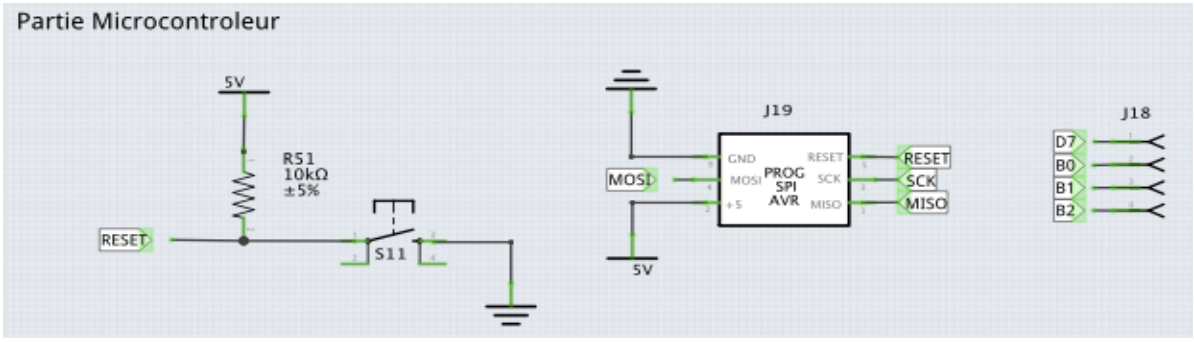
Elle est constitué d'un port USB et d'un FTDI pour programmer l'atmega 328p . En effet le code pour programmer l'interaction des lumières avec le capteur de fréquence cardiaque sera programmer dans l'atmega 328p après la soudure du composant sur la carte car il s'agit d'un composant cms donc il ne peut être programmer avant.

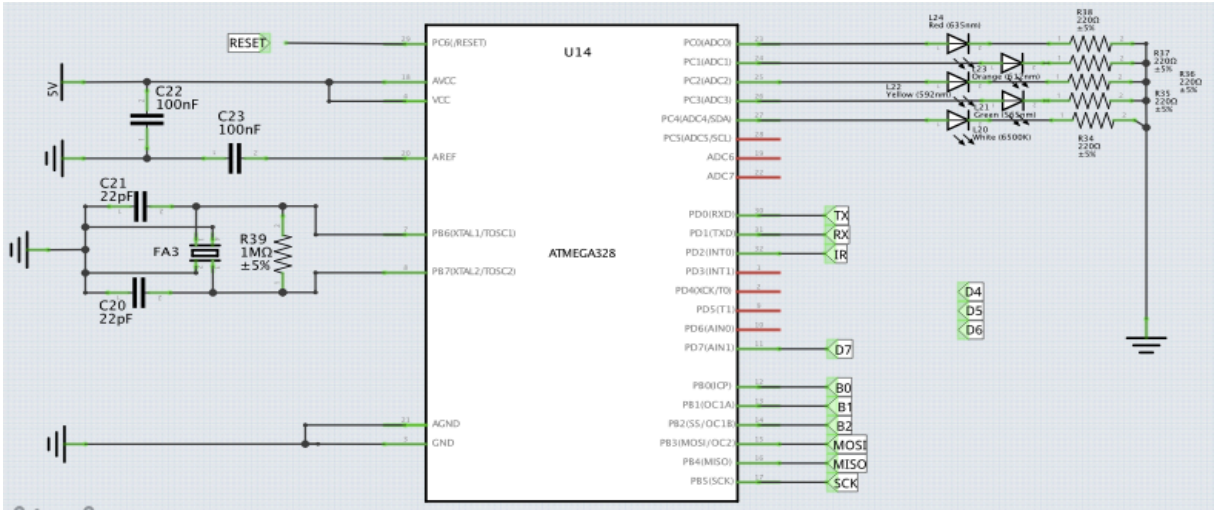
iii. *Partie d'alimentation:*

Constituée d'une batterie de 9 Volts, d'un interrupteur , d'un régulateur de tension pour obtenir une tension de 5volt afin de limiter les pertes de chaleurs. On y a ajouté deux leds de test , une led bleu pour s'assurer qu'o a bien une tension de 5 volt et une led verte pour la tension de 9 volts.



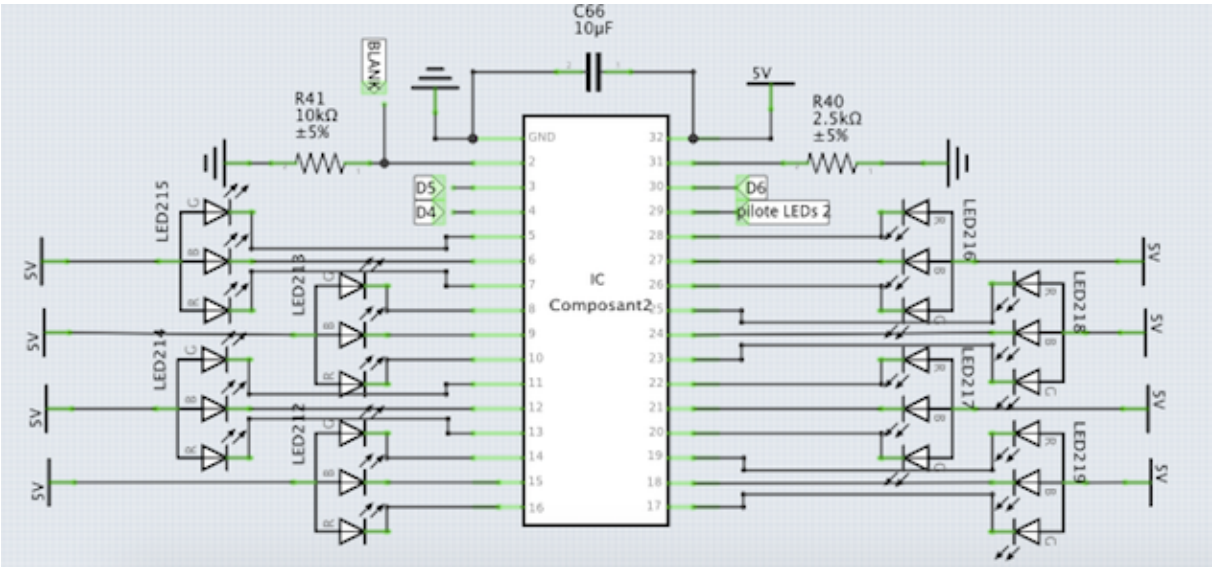






C. La carte des Leds

On a trois cartes de leds toutes identiques. Sur chaque carte se trouve un TLC5947 un pilote de 24 leds. Mais comme on utilise des leds RGB on ne pourra mettre sur chaque carte que 8 leds. On y a ajouter quelques résistances et un condensateur pour limiter le courant dans les leds.



D. Le routage et le soudage

Après le schématique , il fallait réaliser le routage qui était la partie la plus délicate de notre projet car on avait beaucoup de composant à router. Une fois le routage vérifiée et validée par notre encadrant, il l'a commandé à une entreprise extérieure pour un rendu propre où nous avons pu choisir la couleur et la forme des cartes .

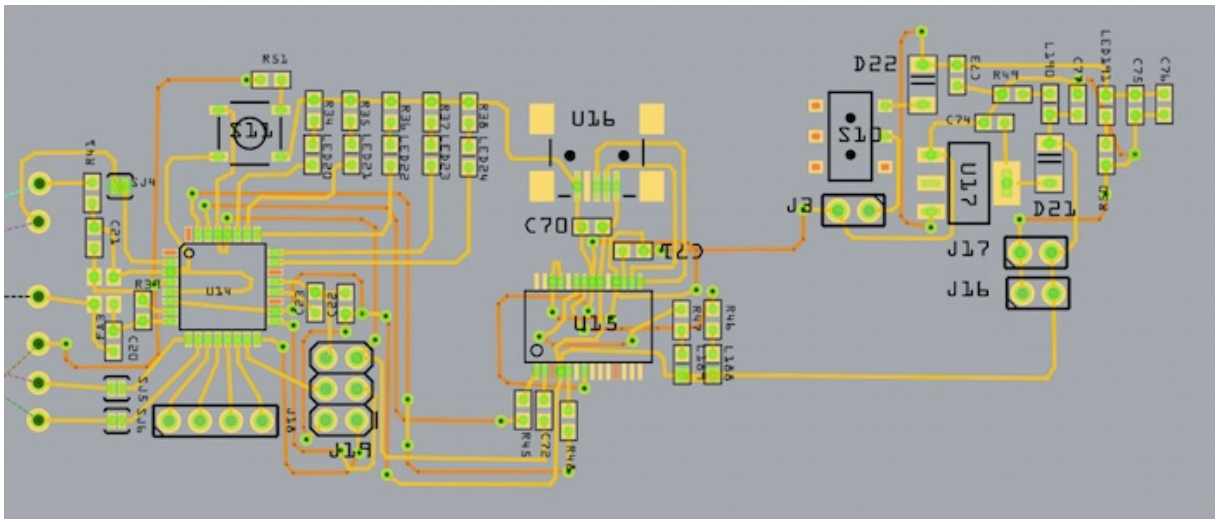


Figure 4: Carte principale

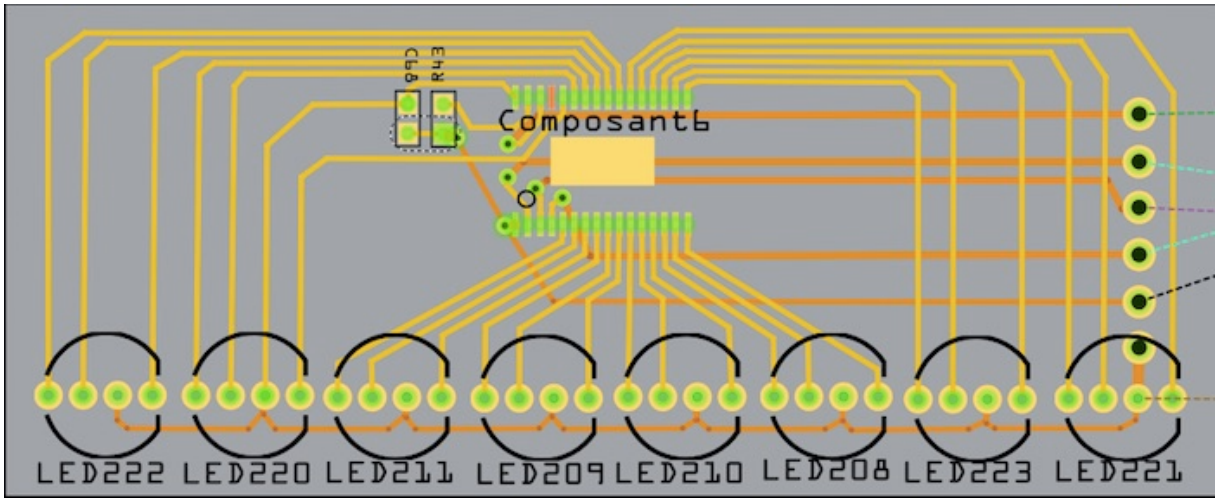


Figure 5: carte des Leds

Une fois toutes les cartes réceptionnées, nous avons soudé les composants sur les cartes. La soudure s'est fait progressivement. Nous avons commencé par soudé la carte principale, ensuite nous l'avons testé afin de s'assurer que les composants sont bien soudé et bien fonctionnels. Après validation, on a soudé les pilotes de leds ainsi que les Leds.

2. Partie programmation

L'interaction des leds en fonction de la fréquence cardiaque nécessite de programmer le microcontrôleur à cet effet. Le programme a été réalisé en langage C et comprend 4 grandes fonctions. Voir en annexes toutes les fonctions.

A. Les fonctions analogues

Tout d'abord on commence par lire la valeur du signal issue du capteur de fréquence cardiaque . Ce dernier nous l'avons branché sur le port PD7 de l'atmega et on le porte tout en haut sur le doigt majeur. Pour pouvoir lire le signal issu du capteur on a utilisé les fonctions convertisseurs analogique numérique.

```
#include <avr/io.h>

/** Fonctions pour le convertisseur analogique / numerique **/

void ad_init(unsigned char channel) {

    ADCSRA |= (1<<ADPS2) | (1<<ADPS1) | (1<<ADPS0); //Clock_ADC = Clock / 128 = 125 000 Hz
    ADMUX |= (1<<REFS0) | (1<<ADLAR); //AVCC with external capacitor at AREF pin
    ADMUX = (ADMUX&0xf0) | channel; //Channel = ADC0, ADC1,...,ADCN
    ADCSRA |= (1<<ADEN);

}

unsigned int ad_sample(void) {

    ADCSRA |= (1<<ADSC); //Start conversion
    while(bit_is_set(ADCSRA, ADSC)); //Attend la fin de la conversion

    return ADCH; //Retourne le registre 8 bits en virant les 2 de poids faible

}
```

B. Les fonctions serial et TLC 5947

Les fonctions serial permettent la communication par liaison série avec l'atmega328p. Dans les fonctions TLC5947 , nous avons commencé par inclure la bibliothèque TLC5747.h , ensuite on a une fonction d'initialisation des pilotes de leds où nous configurons les 24 canaux en sorties. On a une fonctions set_LED_RGB qui allume les leds dans le bon ordre et une fonction set_LED_Drivers qui va demander aux autres leds d'ignorer un signal qui ne leur est pas destiné en restant éteinte.

C. La fonction principale

Nous allons appeler toutes les fonctions précédemment décrites ci haut. Dans le tableau indirect nous avons mis l'ordre dans lequel les leds se suivent , le tableau d cycle contient les 8 combinaison de couleur possible que l'on peut faire avec du rouge, du vert et du bleu et le black correspond à l'état éteint.

```
int indirect[]={20,19,18,23,22,21,2,1,0,5,4,3,8,7,6,11,10,9,14,13,12,17,16,15};
color LED_state[NB_LEDS];
color cycle[]={{64,0,0},{0,64,0},{0,0,64},{64,64,0},{0,64,64},{64,0,64},{64,64,64}};
color black={0,0,0};

int pulse=0;//input signal
int pulse_hold=100;//determine which signal to count

/*initialisation du timer*/
```

Nous avons configuré un timer et autorisé des interruptions chaque seconde pour avoir en tout temps une valeur précise du capteur de fréquence cardiaque. Nous avons ensuite créer deux types d'animations selon que la valeur de la fréquence cardiaque soit inférieure ou supérieure à 100.

```
/*initialisation du timer*/
void init_timer(void){
    TCCR1A =0x02;
    TCCR1B =0x03;
    OCR1AL =0x09;
    OCR1AH =0x3D;
    TIMSK1 =0x02;
}

/*interruption après une seconde*/
ISR(TIMER1_COMPA_vect){

    ad_init(7);
    pulse=ad_sample();


}
```

En effet, si la fréquence est inférieure à 100 , les leds vont s'allumer une à une dans l'ordre , c'est à dire que quand une led est allumée, la led suivante s 'allume et la précédente s'éteint et ainsi de suite . Quand le tour de toutes les leds et fini, elles changent de couleur et refont le même tour. Le temps d'éclairage équivaut à la moitié de la valeur du signal en millisecondes.

```
    /*fonction à executer si la frequence cardiaque est inférieur à 100*
else{
    for(c=0;c<NB_COLORS;c++){
          if(pulse>pulse_hold) goto suite;
            for(l=0;l<NB_LEDS;l++){

                for(i=0;i<NB_LEDS;i++) {
                 LED_state[i]=(i==l)?cycle[c]:black;
                    set_LED_RGB(LED_state,indirect,NB_LEDS);
                    _dyn_delay_ms(pulse);
                }
            }
       }
    }
```

Lorsque la fréquence est supérieur à 100 nous allumons toutes les leds consécutivement et une fois toutes allumées , elles s'éteignent aussi consécutivement passe à une autre couleur. Le temps d'éclairage équivaut à la valeur du signal en millisecondes.

```
/*fonction à executer si la frequence cardiaque est supérieure à 100*/
    if (pulse> pulse_hold){
       for(c=0;c<NB_COLORS;c++){
           if(pulse<pulse_hold) goto suite;
          for(i=0;i<NB_LEDS;i++) {
            LED_state[i]=cycle[c];
            set_LED_RGB(LED_state,indirect,NB_LEDS);
            _dyn_delay_ms(pulse/2);}

            for(i=0;i<NB_LEDS;i++) {
                LED_state[i]=black;
                set_LED_RGB(LED_state,indirect,NB_LEDS);
                _dyn_delay_ms(pulse/2);}

       }
    }
    /*fonction à executer si la frequence cardiaque est inférieur à 100*/
```

3. Assemblage sur la robe

Après avoir soudé toutes les cartes et implémenté les fonctions de lecture du capteur et de configuration de l'éclairage en fonction de la fréquence cardiaque, nous avons ensuite tout assembler sur la robe . Nous avons mesuré le tour de taille de la jupe et avons décidé d'utiliser seulement 3 carte des Leds. Ils sont placés à l'avant, à gauche et à droite de la jupe. Nous avons placé la carte principale derrière la robe .

Enfin, nous avons fixé les fibres optiques sur les leds au moyen des gaines rétro rétractables que nous avons adapter sur les dimension des leds et des fibres optiques. Nous avons aussi poncer les fibres optiques pour leurs apporter beaucoup plus d'éclat.

IV. Conclusion

Nous avons trouvé le projet très intéressant car il nous a permis de mettre en pratique les connaissances acquises au cours de notre formations IMA , notamment la conception et la réalisation des cartes électroniques, la programmation des microcontrôleurs mais aussi comment lire et manipuler les données issues d'un capteur.

Nous avons également expérimenté le travail en groupe et la gestion d'un projet sur le long terme . Une bonne organisation , la répartition des tâches ,la collaboration et l'investissement de chacun sont les valeurs que nous retenons de cette expérience.

V. Annexes

```c
#include <avr/io.h>

/** Fonctions pour le convertisseur analogique / numerique **/

void ad_init(unsigned char channel) {

    ADCSRA |= (1<<ADPS2) | (1<<ADPS1) | (1<<ADPS0); //Clock_ADC = Clock / 128 = 125 000 Hz
    ADMUX |= (1<<REFS0) | (1<<ADLAR); //AVCC with external capacitor at AREF pin
    ADMUX = (ADMUX&0xf0) | channel; //Channel = ADC0, ADC1,...,ADCN
    ADCSRA |= (1<<ADEN);

}

unsigned int ad_sample(void) {

    ADCSRA |= (1<<ADSC); //Start conversion
    while(bit_is_set(ADCSRA, ADSC)); //Attend la fin de la conversion

    return ADCH; //Retourne le registre 8 bits en virant les 2 de poids faible

}
```

Figure 6:analog.c

```c
/** Functions for serial port **/

#include <avr/io.h>
#include <stdio.h>

#include "serial.h"

static int send_serial_printf(char c,FILE *stream);
static FILE mystdout=FDEV_SETUP_STREAM(send_serial_printf,NULL,_FDEV_SETUP_WRITE);

void init_serial(int speed){
/* Set baud rate */
UBRR0 = F_CPU/(((unsigned long int)speed)<<4)-1;

/* Enable transmitter & receiver */
UCSR0B = (1<<TXEN0 | 1<<RXEN0);

/* Set 8 bits character and 1 stop bit */
UCSR0C = (1<<UCSZ01 | 1<<UCSZ00);

/* Set off UART baud doubler */
UCSR0A &= ~(1 << U2X0);
}

void send_serial(char c){
loop_until_bit_is_set(UCSR0A, UDRE0);
UDR0 = c;
}

char get_serial(void){
loop_until_bit_is_set(UCSR0A, RXC0);
return UDR0;
}
```

Figure 7: Serial.c

```
void init_printf(void){
init_serial(9600);
stdout=&mystdout;
}

static int send_serial_printf(char c,FILE *stream){
if(c=='\n') send_serial('\r');
send_serial(c);
return 0;
}
```

Figure 8: Serial.c

```
/** TLC5947 LEDs driver **/

// Include files

#include <avr/io.h>
#include <stdio.h>

#include "tlc5947.h"

// Functions

void init_LED_Drivers(int nb){
                    // LED drivers I/O as outputs

  //printf("init %d\n",nb);
  DDR_DLED |= (1<<PIN_DLED_CLOCK) | (1<<PIN_DLED_DATA) | (1<<PIN_DLED_LATCH);
                    // Set LATCH output low
  PORT_DLED &= ~(1<<PIN_DLED_LATCH);
}

void set_LED_RGB(color *leds,int *indirect,int nb){
int pwm[3*nb];
int i;
//printf("RGB %d\n",nb);
for(i=0;i<nb;i++){
  if(!indirect){ pwm[3*i]=leds[i].red<<4; pwm[3*i+1]=leds[i].green<<4; pwm[3*i+2]=leds[i].blue<<4; }
  else{
    int low=7-(i&0x07),high=(i>>3)*24;
   // printf("%d,%d ",high,low);
    pwm[high+indirect[3*low]]=leds[i].red<<4; pwm[high+indirect[3*low+1]]=leds[i].green<<4; pwm[high+indirect[3*low+2]]=leds[i].blue<<4; }
  }
```

Figure 9: TLC5947.c

```
void set_LED_Drivers(int pwm[],int nb){
  int c,b;

  //printf("LED %d\n",nb);
  // Set LATCH output low
  PORT_DLED &= ~(1<<PIN_DLED_LATCH);
  // 24 channels per TLC5974
  for(c=DLED_CHANNELS*nb-1;c>=0;c--){
    // 12 bits per channel, send MSB first
    for(b=11;b>=0;b--){
                                    // Set CLOCK output low
      PORT_DLED &= ~(1<<PIN_DLED_CLOCK);

                                    // Set DATA as stated by bit #b of c
      if(pwm[c] & (1<<b))
        PORT_DLED |= (1<<PIN_DLED_DATA);
      else
        PORT_DLED &= ~(1<<PIN_DLED_DATA);

                                    // Set CLOCK output HIGH
      PORT_DLED |= (1<<PIN_DLED_CLOCK);
    }
  }
                                    // Set CLOCK output low
  PORT_DLED &= ~(1<<PIN_DLED_CLOCK);

                                    // Set LATCH output high
  PORT_DLED |= (1<<PIN_DLED_LATCH);
                                    // Set LATCH output low
  PORT_DLED &= ~(1<<PIN_DLED_LATCH);
}
```

Figure 10:TLC4947.c

```
#include <stdio.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include "serial.h"
#include "tlc5947.h"
#include "analog.h"

#define NB_DRIVERS  3
#define NB_COLORS   7
#define NB_LEDS     (NB_DRIVERS*DLED_CHANNELS/3)


int indirect[]={20,19,18,23,22,21,2,1,0,5,4,3,8,7,6,11,10,9,14,13,12,17,16,15};
color LED_state[NB_LEDS];
color cycle[]={{64,0,0},{0,64,0},{0,0,64},{64,64,0},{0,64,64},{64,0,64},{64,64,64}}
color black={0,0,0};

int pulse=0;//input signal
int pulse_hold=100;//determine which signal to count

/*initialisation du timer*/
void init_timer(void){
    TCCR1A =0x02;
    TCCR1B =0x03;
    OCR1AL =0x09;
    OCR1AH =0x3D;
    TIMSK1 =0x02;
}

/*interruption après une seconde*/
```

Figure 11:leds.c

```
/*interruption après une seconde*/
ISR(TIMER1_COMPA_vect){

    ad_init(7);
    pulse=ad_sample();


}

/* fonction de delay*/
void _dyn_delay_ms(int32_t time){
    int32_t i;
    for(i=0;i<time;i++) _delay_us(1000);
}
```

```
int main(void)
{

init_printf();//configuration de la laison série

init_timer();//on commence à compter le temps

printf("Init. drivers\n");
init_LED_Drivers(NB_DRIVERS);
int i,c,l;

sei(); //autorisation des interruptions
while(1){

     printf("pulse is:%d\n",pulse);

/*fonction à executer si la frequence cardiaque est supérieure à 100*/
    if (pulse> pulse_hold){
       for(c=0;c<NB_COLORS;c++){
           if(pulse<pulse_hold) goto suite;
          for(i=0;i<NB_LEDS;i++) {
            LED_state[i]=cycle[c];
            set_LED_RGB(LED_state,indirect,NB_LEDS);
            _dyn_delay_ms(pulse/2);}



            for(i=0;i<NB_LEDS;i++) {
                LED_state[i]=black;
                set_LED_RGB(LED_state,indirect,NB_LEDS);
                _dyn_delay_ms(pulse/2);

            }
```

```
            for(i=0;i<NB_LEDS;i++) {
                LED_state[i]=black;
                set_LED_RGB(LED_state,indirect,NB_LEDS);
                _dyn_delay_ms(pulse/2);

            }

        }
    }
    /*fonction à executer si la frequence cardiaque est inférieur à 100*/
else{
    for(c=0;c<NB_COLORS;c++){
        if(pulse>pulse_hold) goto suite;
          for(l=0;l<NB_LEDS;l++){

              for(i=0;i<NB_LEDS;i++) {
               LED_state[i]=(i==l)?cycle[c]:black;
                  set_LED_RGB(LED_state,indirect,NB_LEDS);
                  _dyn_delay_ms(pulse);
              }
          }
      }
    }

    suite:printf("coucou\n");
    }

    |
return 0;

}
```

Figure 12:fonction principale