

# Compte rendu : Le Simon

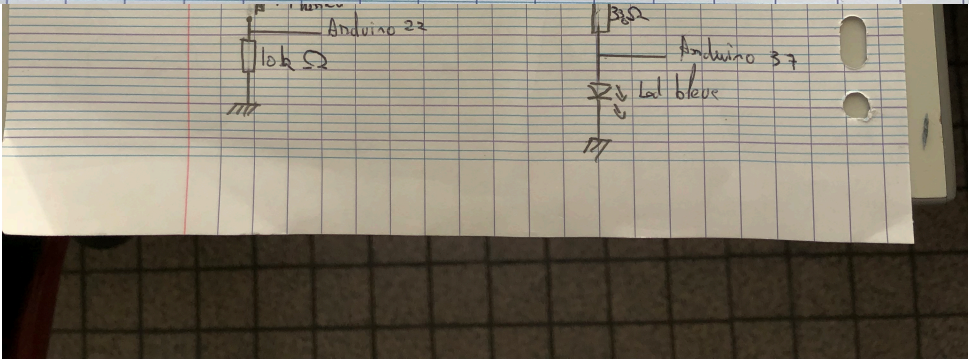
Pour ce TP, nous avons été chargé de reproduire le jeu Simon , un jeux de mémorisation avec des couleur et des bouton poussoir , mais tout sa doit être fait avec arduino en y incluant un écran LCD

## Ce dont on a besoin :

- Une carte Arduino ( Méga 2560 )
- Un écran LCD DFrobot
- 4 led rgb
- 4 bouton poussoir
- Des résistance (4  $10K\Omega$  et 4  $330\Omega$ )
- Des câbles ( beaucoup de mâle - mâle et un mâle - femelle )
- Une Bread-board

## Le câblage :

Le câblage de notre carte se divise en 3 partie , les led , les bouton e  
t l'écran lcd , qui seront mieux définie lors du code



# Les led

Pour commander les led , j'ai raccorder chacune des led individuellement sur une broche et paramétrer via le code , il y a en tout 4 led

- Une Rouge , alimenter seulement par la broche rouge de cette led
- Une Jaune , alimenter par la broche rouge et verte de cette led
- Une Verte , alimenter seulement par la broche verte de cette led
- Une Bleu , alimenter seulement pas la broche bleu de cette led

## Les bouton poussoir :

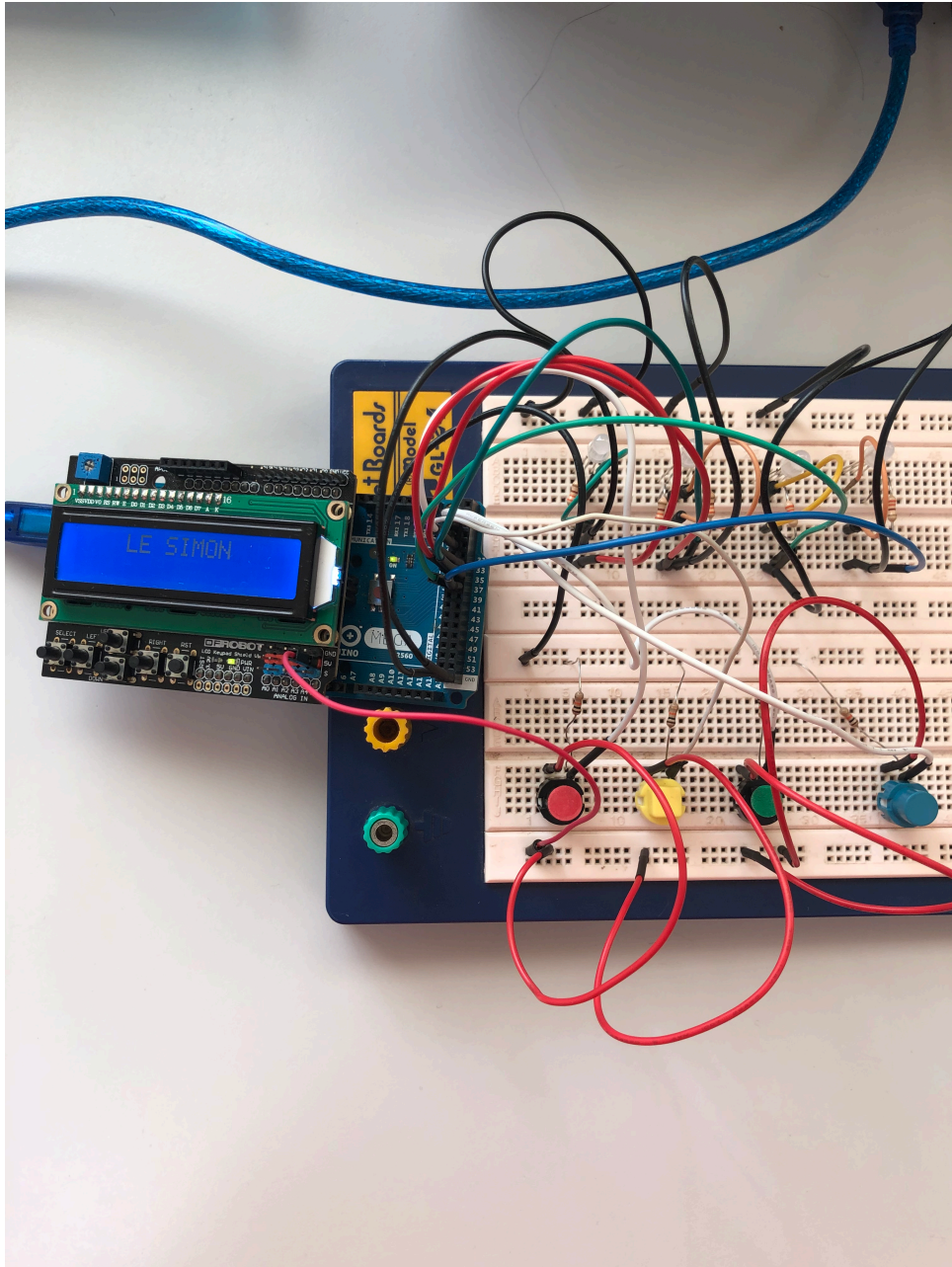
Pour actionner faire fonctionner les bouton je l'ai ai placer les une a cote des autres , en face des led sur la Bread board

Chaque bouton est relié indépendamment sur la carte et sera lier à chaque led via le code

## L'écran LCD :

L'écran est seulement un shield qu'on pose sur la carte et qui s'aligne avec n'importe quel carte arduino

Elle dispose de nombreux bouton et aussi d'un écran led de 16 cases sur 2 qui elle même comporte 5 pixel par 8  
Et voici la photo de notre câblage complet



## Le code arduino :

Le code fait peut paraître un peu difficile , mais une fois compris il est très simple

Je vais expliquer certaine partie du code qui sont plus difficile

```
// Fonction pour vérifier l'entrée de l'utilisateur  
void checkUserInput() {
```



```

if (userInput[userInputIndex] == sequence[userInputIndex]) {
    userInputIndex++; // Passer à l'entrée suivante de l'utilisateur
    if (userInputIndex == sequenceLength) {
        score++; // Augmenter le score si la séquence est correcte
        if (score % 5 == 0) { // Tous les 5 niveaux, augmenter la difficulté
            level++;
            lcd.clear();
            lcd.setCursor(0, 0);
            lcd.print("Niveau : ");
            lcd.print(level);
            delay(1000);
        }
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Correct! Score: ");
        lcd.print(score);
        delay(1000);
        nextRound(); // Passer à la prochaine round
    }
} else {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Jeu Terminé!");
    lcd.setCursor(0, 1);
    lcd.print("Score Final: ");
    lcd.print(score);
    delay(2000);
    startGame(); // Redémarrer le jeu
}
}

```

Ce code si compte est une fonction pour initier le jeux et les variable aléatoire pour les led , c'est assez simple mais complet

Code complet :

```
#include <LiquidCrystal.h>
```

```
// Identification des leds
```

```
int redpin1 = 33;
```

```
int redpin2 = 34;
```

```
int greenpin2 = 35;
```

```

int greenpin3 = 36;
int bluepin4 = 37;

// Identification des boutons
int redbtn = 28;
int yellowbtn = 26;
int greenbtn = 24;
int bluebtn = 22;

// Variables pour la gestion des boutons
bool test1, test2, test3, test4;

// Définition de l'écran LCD
const int rs = 8, en = 9, d4 = 4, d5 = 5, d6 = 6, d7 = 7;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

// Tableau pour la séquence de Simon (les indices sont les LEDs)
int sequence[100]; // Séquence maximale de 100 couleurs
int sequenceLength = 0; // Longueur actuelle de la séquence
int userInput[100]; // Réponse de l'utilisateur
int userInputIndex = 0; // Index de l'entrée de l'utilisateur
int score = 0; // Score du joueur
int level = 1; // Niveau du jeu (difficulté)

// Fonction pour démarrer un nouveau jeu
void startGame() {
    score = 0;
    sequenceLength = 0;
    level = 1; // Commencer au niveau 1
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Commencez!");
    delay(1000);
    nextRound(); // Démarrer la première round
}

// Fonction pour passer à la prochaine round
void nextRound() {
    userInputIndex = 0; // Réinitialiser l'entrée utilisateur
    sequenceLength++; // Augmenter la longueur de la séquence
    sequence[sequenceLength - 1] = random(0, 4); // Ajouter une couleur aléatoire à la
séquence
    // Afficher la séquence
    for (int i = 0; i < sequenceLength; i++) {

```

```
int ledIndex = sequence[i];
lightUp(ledIndex);
delay(500 - (level * 50)); // Diminuer le délai à chaque niveau (niveau 1: 500ms,
niveau 2: 450ms...)
turnOffAllLEDs();
delay(250); // Attente de 0.25s avant d'afficher la LED suivante
}
```

```
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("A votre tour!");
}
```

// Fonction pour allumer la LED correspondant à l'indice

```
void lightUp(int ledIndex) {
switch (ledIndex) {
case 0:
digitalWrite(redpin1, HIGH);
lcd.setCursor(0, 1);
lcd.print("ROUGE");
break;
case 1:
digitalWrite(redpin2, HIGH);
digitalWrite(greenpin2, HIGH);
lcd.setCursor(0, 1);
lcd.print("JAUNE");
break;
case 2:
digitalWrite(greenpin3, HIGH);
lcd.setCursor(0, 1);
lcd.print("VERT");
break;
case 3:
digitalWrite(bluepin4, HIGH);
lcd.setCursor(0, 1);
lcd.print("BLEU");
break;
}
}
```

// Fonction pour éteindre toutes les LEDs

```
void turnOffAllLEDs() {
digitalWrite(redpin1, LOW);
digitalWrite(redpin2, LOW);
digitalWrite(greenpin2, LOW);
}
```

```

digitalWrite(greenpin3, LOW);
digitalWrite(bluepin4, LOW);
}

// Fonction pour vérifier l'entrée de l'utilisateur
void checkUserInput() {
  if (userInput[userInputIndex] == sequence[userInputIndex]) {
    userInputIndex++; // Passer à l'entrée suivante de l'utilisateur
    if (userInputIndex == sequenceLength) {
      score++; // Augmenter le score si la séquence est correcte
      if (score % 5 == 0) { // Tous les 5 niveaux, augmenter la difficulté
        level++;
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Niveau : ");
        lcd.print(level);
        delay(1000);
      }
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Correct! Score: ");
      lcd.print(score);
      delay(1000);
      nextRound(); // Passer à la prochaine round
    }
  } else {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Jeu Terminé!");
    lcd.setCursor(0, 1);
    lcd.print("Score Final: ");
    lcd.print(score);
    delay(2000);
    startGame(); // Redémarrer le jeu
  }
}

void setup() {
  // Initialisation des LEDs et des boutons
  pinMode(redpin1, OUTPUT);
  pinMode(redpin2, OUTPUT);
  pinMode(greenpin2, OUTPUT);
  pinMode(greenpin3, OUTPUT);
  pinMode(bluepin4, OUTPUT);
}

```



```
pinMode(redbtn, INPUT);
pinMode(yellowbtn, INPUT);
pinMode(greenbtn, INPUT);
pinMode(bluebtn, INPUT);
```

```
Serial.begin(9600);
lcd.begin(16, 2);
lcd.print(" LE SIMON");
delay(2000);
lcd.clear();
// Démarrer le jeu
startGame();
}
```

```
void loop() {
// Afficher la séquence à l'utilisateur
if (userInputIndex == sequenceLength) {
delay(500); // Attente de 0.5s après que l'utilisateur ait entré sa séquence
nextRound(); // Passer à la prochaine round
}
```

```
// Vérifier la réponse de l'utilisateur
test1 = digitalRead(redbtn);
test2 = digitalRead(yellowbtn);
test3 = digitalRead(greenbtn);
test4 = digitalRead(bluebtn);
```

```
if (test1 == 1) {
lightUp(0); // Allumer la LED rouge quand le bouton est pressé
delay(500); // Afficher la LED rouge pendant 300ms
turnOffAllLEDs();
userInput[userInputIndex] = 0; // Rouge
checkUserInput();
}
if (test2 == 1) {
lightUp(1); // Allumer la LED jaune quand le bouton est pressé
delay(500); // Afficher la LED jaune pendant 300ms
turnOffAllLEDs();
userInput[userInputIndex] = 1; // Jaune
checkUserInput();
}
if (test3 == 1) {
lightUp(2); // Allumer la LED verte quand le bouton est pressé
delay(500); // Afficher la LED verte pendant 300ms
turnOffAllLEDs();
```

```
userInput[userInputIndex] = 2; // Vert
checkUserInput();
}
if (test4 == 1) {
lightUp(3); // Allumer la LED bleue quand le bouton est pressé
delay(500); // Afficher la LED bleue pendant 300ms
turnOffAllLEDs();
userInput[userInputIndex] = 3; // Bleu
checkUserInput();
}
}
```

Liam Le Huérou et Hugo Astruc TSTI2D2 SIN