Montage Arduino pour la Fête de la Science

Antoine Bérut

8 décembre 2014

1 Présentation générale

Un petit montage à base de micro-contrôleur Arduino UNO permet d'illustrer l'utilisation de quelques capteurs et transducteurs courants (et bon marché), en servant d'interface entre le monde « physique » et le monde « électronique ».

Dans la version actuelle du montage, trois capteurs sont branchés :

- Un capteur de température (TMP36 de chez Analog Device)
- Un capteur capacitif (Standalone Toggle Capacitive Touch Sensor Breakout AT42QT1012 de chez Adafruit)
- Un capteur de lumière (Log-scale Analog Light Sensor GA1A12S202 de chez Adafruit)

Le capteur de température contrôle en ensemble de DELs [1] qui s'allument progressivement au fur et à mesure que la température s'accroît. Le capteur capacitif permet d'allumer ou d'éteindre un mini-buzzer (Piezo Electric Sound Component PKM17EPP-4001-B0 de chez muRata). Et enfin le capteur de lumière change la note de musique qui va être jouée par le buzzer en fonction de l'éclairement reçu, créant ainsi un « theremin lumineux ».

2 Fonctionnement pratique

Pour que le montage fonctionne, le micro-contrôleur Arduino doit être alimenté (soit en 5 V via son câble USB, soit via son entrée « power jack » avec une tension continue entre 7 et 12 V.)

Lorsque le montage vient d'être branché, ou lorsque l'on presse le bouton RESET, seul le capteur de lumière est actif pendant 5 s pour sa calibration [2] : le capteur enregistre en continu l'éclairement et les valeurs extrémales prises durant ce laps de temps deviendront ses bornes de sensibilité. Lors du fonctionnement normal, toute valeur d'éclairement supérieure à la valeur maximale enregistrée pendant la phase de calibration sera considérée comme égale à cette valeur maximale (idem pour les valeurs inférieures à la valeur minimale mesurée pendant la calibration). Si les conditions d'éclairement changent durant l'utilisation du montage, il est recommandé de recalibrer le capteur.

Les quatre DELs s'allument respectivement lorsque la température au niveau du capteur dépasse : 20 °C (DEL bleue), 24 °C (DEL jaune), 28 °C (DEL rouge) et 30 °C (DEL rouge). Il est donc assez facile de faire s'allumer les lumière en le prenant dans ses mains.

Le capteur capacitif fonctionne comme un interrupteur pour le buzzer piézo-électrique (qui s'allume ou s'éteint lorsqu'on touche le capteur).

Le capteur de lumière transforme une valeur d'éclairement en note de musique jouée par le buzzer piézo-électrique (en tout il y a 2 octaves, donc 14 notes, réparties non-linéairement [3] entre les valeurs seuils mesurées pendant la phase de calibration).

1. Diode électroluminescente.
2. La calibration est indiquée par le clignotement d'une DEL située sur la carte Arduino.
3. Car le capteur n'est pas linéaire.

Dans une pièce bruyante, le buzzer n'est pas très audible, il serait sans doute préférable de brancher le signal sortant de l'Arduino (qui fait entre 0 et 5 V, mais n'est pas capable de débiter beaucoup de courant) sur un amplificateur, lui-même relié à un vrai haut-parleur pour avoir quelque chose de plus convaincant.

3 Description des branchements

Les capteurs sont reliés à une plaquette d'essai via des connecteurs « faits maison ». Toutes les connexions à l'Arduino se font via la plaquette d'essai, sans soudure.

Les connecteurs sont marqués d'une lettre permettant de les identifier (T pour le capteur de température, C pour le capteur capacitif et L pour le capteur de lumière). Ils disposent tous d'une petite encoche triangulaire sur leur partie plastique qui indique le fil servant de fil de masse.

Les deux capteurs de chez Adafruit comportent des inscriptions claires (GND pour la masse, VCC pour l'alimentation 5 V, OUT pour le signal). Pour le capteur thermique, *lorsque l'on a la partie plate en face de soi*, la masse est la patte de droite [4], le signal est la patte centrale, et l'alimentation la patte de gauche.

Les masses (GND) sont toutes reliées entre elles, et les alimentations (VCC) des capteurs sont assurées par la sortie régulée à 5V de l'Arduino.

Les trois sorties (OUT) des capteurs sont branchées sur les entrées 0 à 2 de l'Arduino (dans l'ordre : température, capacitif, lumière). Les quatre DELs sont branchées sur les sorties 2 à 5 de l'Arduino, et reliées à la masse via des résistances d'une centaine d'Ohms (pour éviter de les griller en leur imposant un courant trop important). Le buzzer piézo-électrique est branché entre la sortie 12 de l'Arduino et la masse.

Le montage est schématisé figure 1.

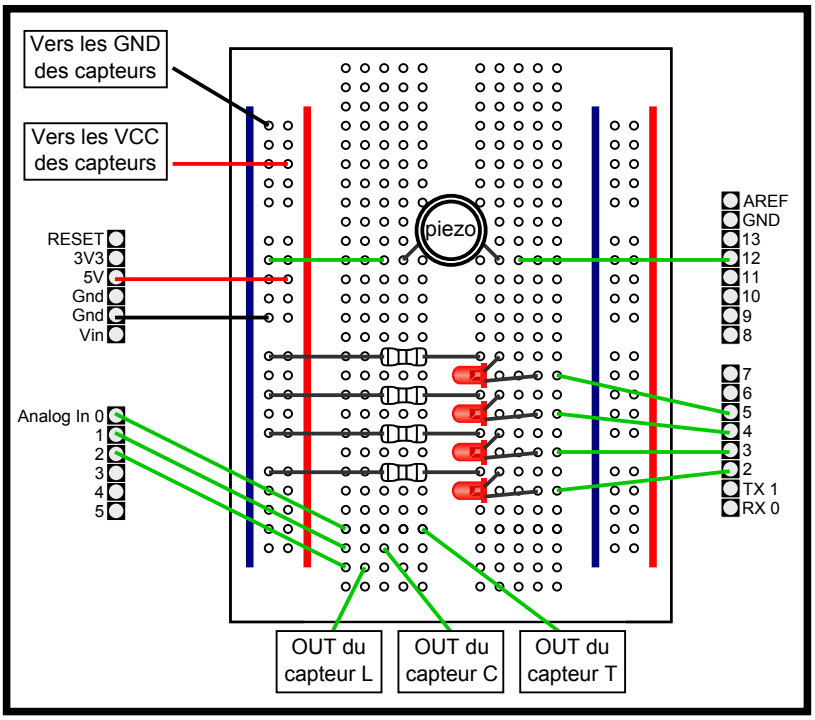

FIGURE 1 – Schéma équivalent des branchements entre l'Arduino et la plaquette d'essai (le vrai montage est plus compact).

4. Elle est normalement marquée au Tipp-Ex, mais je doute que cela tienne très longtemps.

4 Description du code Arduino

Le micro-contrôleur Arduino est programmable facilement dans un langage ressemblant à du C, via le logiciel spécifique (et libre) Arduino.

Il suffit de le brancher en USB à un ordinateur sur lequel le logiciel est installé pour qu'il soit reconnu. On peut ensuite (après avoir sélectionné le modèle de carte utilisée et le port COM sur lequel on est branché) lui « téléverser » un script qu'il gardera en mémoire même hors tension.

Ci-dessous le script commenté (la plupart des fonctions ont des noms transparents, mais vous pouvez trouver leur description détaillée sur la page de référence).

Définition des constantes et variables :

```
// On definit les capteurs et les sorties
const int TempInPin = A0;
const int CapaInPin = A1;
const int LumiInPin = A2;
const int SonOutPin = 12;
const int ledPin = 13;

// On definit la temperature ambiante
const float TempAmb = 20.0;

// On definit les notes via un arrondi de la frequence leur correspondant
const int B_g = 245; // si grave
const int C = 261; // do
const int D = 294; // re
const int E = 329; // mi
const int F = 349; // fa
const int G = 392; // sol
const int A = 440; // la
const int B = 493; // si
const int c = 523; // do aigu
const int d = 587; // re aigu
const int e = 659; // mi aigu
const int f = 698; // fa aigu
const int g = 784; // sol aigu
const int a = 880; // la aigu
const int b = 988; // si aigu
const int R = 0; // silence

// On definit la gamme
int gamme[16] = { R, B_g, C, D, E, F, G, A, B, c, d, e, f, g, a, b };

// On definit les variables pour la detection de lumiere
int LumiValue;
int LumiLow = 1023;
int LumiHigh = 0;
```

Boucle d'initialisation (qui est lue au démarrage de l'Arduino, ou lorsqu'on appuie sur le bouton RESET) :

```
void setup() {
  // Eteint les diodes de temperature
  for(int pinNumber = 2; pinNumber < 6; pinNumber++){
    pinMode(pinNumber, OUTPUT);
    digitalWrite(pinNumber, LOW);
  }

  // Allume la diode qui signale la calibration
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, HIGH);
```

```
11
   // Mesure l'intensite lumineuse pendant 5 s pour calibrer
13 // Les valeurs extremales seront LumiHigh et LumiLow
   while (millis() < 5000) {
15   LumiValue = analogRead(LumiInPin);
     if (LumiValue > LumiHigh) {
17     LumiHigh = LumiValue;
     }
19   if (LumiValue < LumiLow) {
       LumiLow = LumiValue;
21   }
   }
23 // Eteint la diode qui indique la calibration
   digitalWrite(ledPin, LOW);
25 }
```

Boucle infinie qui s'enclenche dès que la phase d'initialisation est terminée (et dure jusqu'à ce que l'Arduino soit arrêté ou que le bouton RESET soit pressé) :

```
1  void loop(){

3    // On mesure la valeur de temperature
     int TsensorVal = analogRead(TempInPin);
5    float Tvoltage = (TsensorVal/1024.0) * 5.0;
     float temperature = (Tvoltage - .5) * 100;
7
     // On allume ou éteint les DELs en consequence
9    if(temperature >= TempAmb + 0){
       digitalWrite(2, HIGH);
11   }
     else{
13       digitalWrite(2, LOW);
     }
15   if(temperature >= TempAmb + 4){
       digitalWrite(3, HIGH);
17   }
     else{
19     digitalWrite(3, LOW);
     }
21   if(temperature >= TempAmb + 8){
       digitalWrite(4, HIGH);
23   }
     else{
25     digitalWrite(4, LOW);
     }
27   if(temperature >= TempAmb + 10){
       digitalWrite(5, HIGH);
29   }
     else{
31     digitalWrite(5, LOW);
     }
33
     // On attend une milli-seconde juste pour ne pas risquer de faire 2 mesures trop
       proches
35   delay(1);

37   // On mesure la valeur sur le detecteur capacitif
     int CsensorVal = analogRead(CapaInPin);
39   // Si le detecteur est actif, on mesure la valeur sur le detecteur de lumiere
     if(CsensorVal>512){
41     LumiValue = analogRead(LumiInPin);

43     // On bloque les valeurs qui depassent les seuils de calibration
       if(LumiValue>LumiHigh){
```

```
45          LumiValue=LumiHigh;
        }
47      if(LumiValue<LumiLow){
          LumiValue=LumiLow;
49      }

51      // On convertit la valeur de l'eclairement en valeur comprise entre 1 et 15 qui
        correspond a une note de musique
        int note = map(LumiValue, LumiLow, LumiHigh, 1, 15);
53
        // On joue la note de musique pendant 200 milli-secondes
55      tone(SonOutPin, gamme[note], 200);
        // On attend 150 milli-secondes
57      delay(150);
    }
59  }
```

Pour plus d'informations sur le code Arduino, je vous renvoie vers leur site internet qui comporte un tutoriel et de nombreux exemples !