

# PROJET : *Variophuino*

## Rapport de séance n°3 du 03/01/2022

Durant les vacances de Noël puis lors de cette troisième séance mon but était de:

- Réaliser un montage avec au moins 2 pistes de carbone sur la carte Arduino puis vérifier si le projet était réalisable avec un nombre de notes plus important,
- Fabriquer le clavier du *Variophuino*,
- Améliorer la production du son (montage de 2 haut-parleurs à l'aide de transistors),
- Fabriquer le « Takstylo »,
- Modifier le programme afin de régler certains problèmes,
- Commander, recevoir et tester la carte Arduino MEGA, les transistors, les potentiomètres et les haut-parleurs,
- Fabriquer le second clavier du *Variophuino*,
- Modifier le programme de l'effet Pitchbend.

### ➤ Montage avec plusieurs pistes de carbone → Révision du projet.

Après avoir étendu le projet avec un deuxième potentiomètre un problème est apparu : Le projet ne fonctionnait que sur une seule note (soit une seule piste de carbone de potentiomètre à glissière). En effet, pour produire un changement de fréquence sur chaque note nous avons l'intention de placer un potentiomètre par touche (soit 13 pistes de carbone au total) pour effectuer l'effet pitchbend directement sur la note souhaitée et non sur toutes les notes à l'aide d'une molette (comme sur un synthétiseur classique).

Le problème que nous avons rencontré est que nous sortions, pour 2 touches, 2 fils soit 2 Takstylo. Il y avait donc autant de Takstylo que de touches sur le clavier ! Cela étant contraignant et difficile à régler nous avons choisi de nous rabattre sur l'idée de ne mettre qu'un seul potentiomètre qui agirait sur les 13 touches (8 non altérées et 5 altérées).

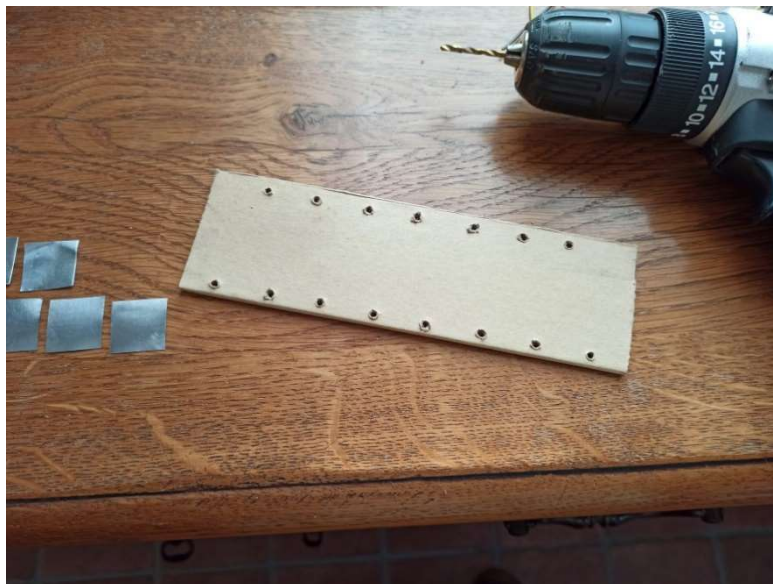
➤ **Fabrication du premier clavier : le « clavier main droite » pour jouer la mélodie**

J'ai fabriqué les 2 claviers du *Variophuino* et câbler le potentiomètre ainsi que les 13 entrées du clavier.

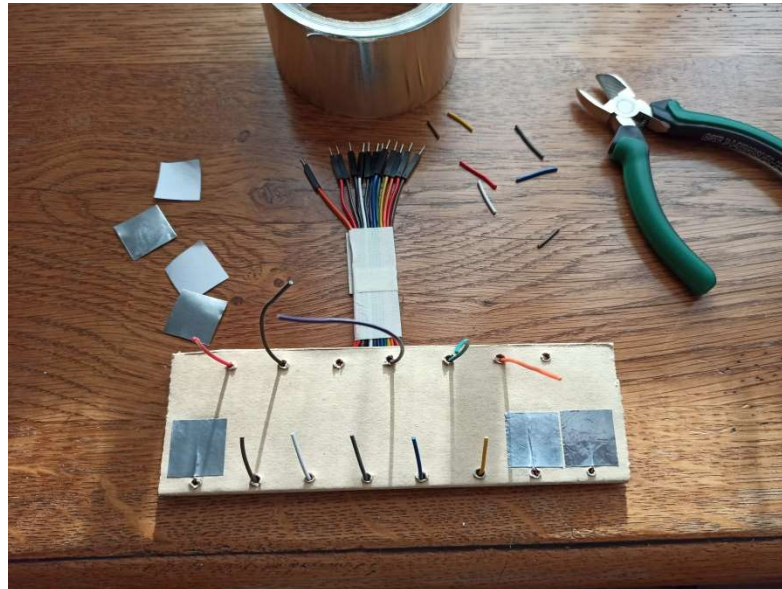
Perçage du support



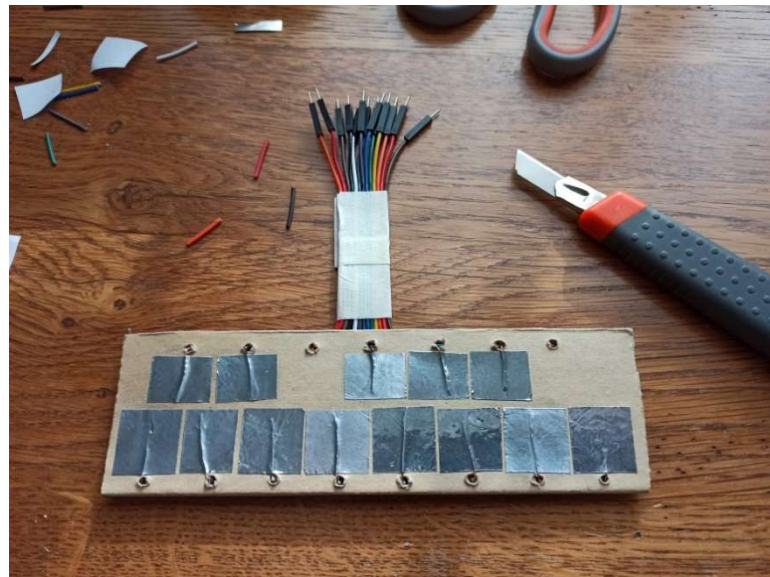
Perçage terminé



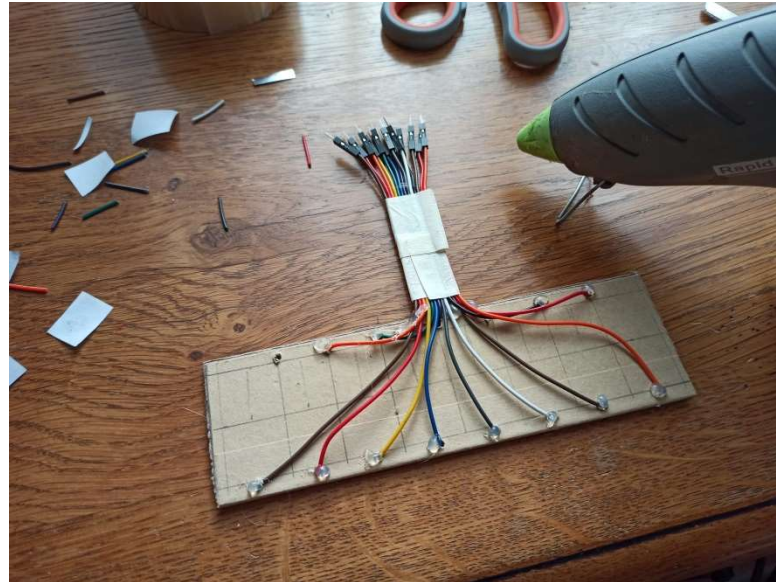
Mise en place des fils et des surfaces de contacts



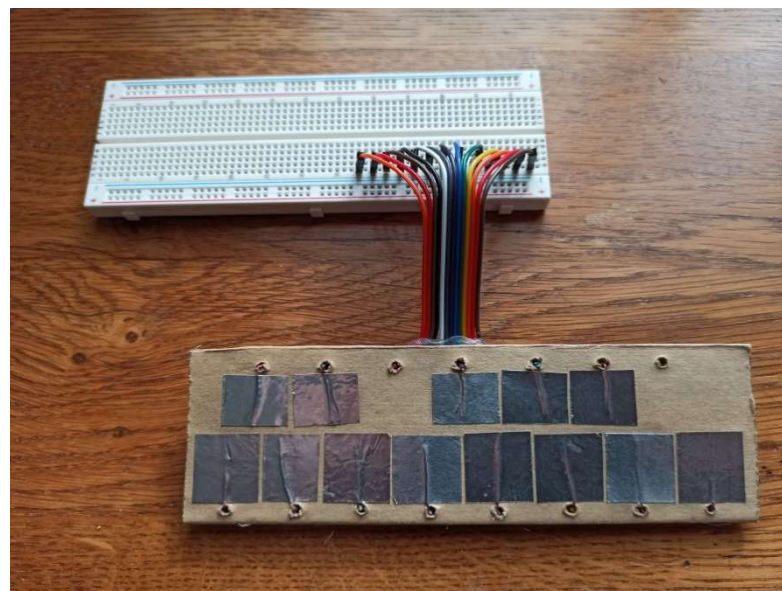
Face supérieure terminée



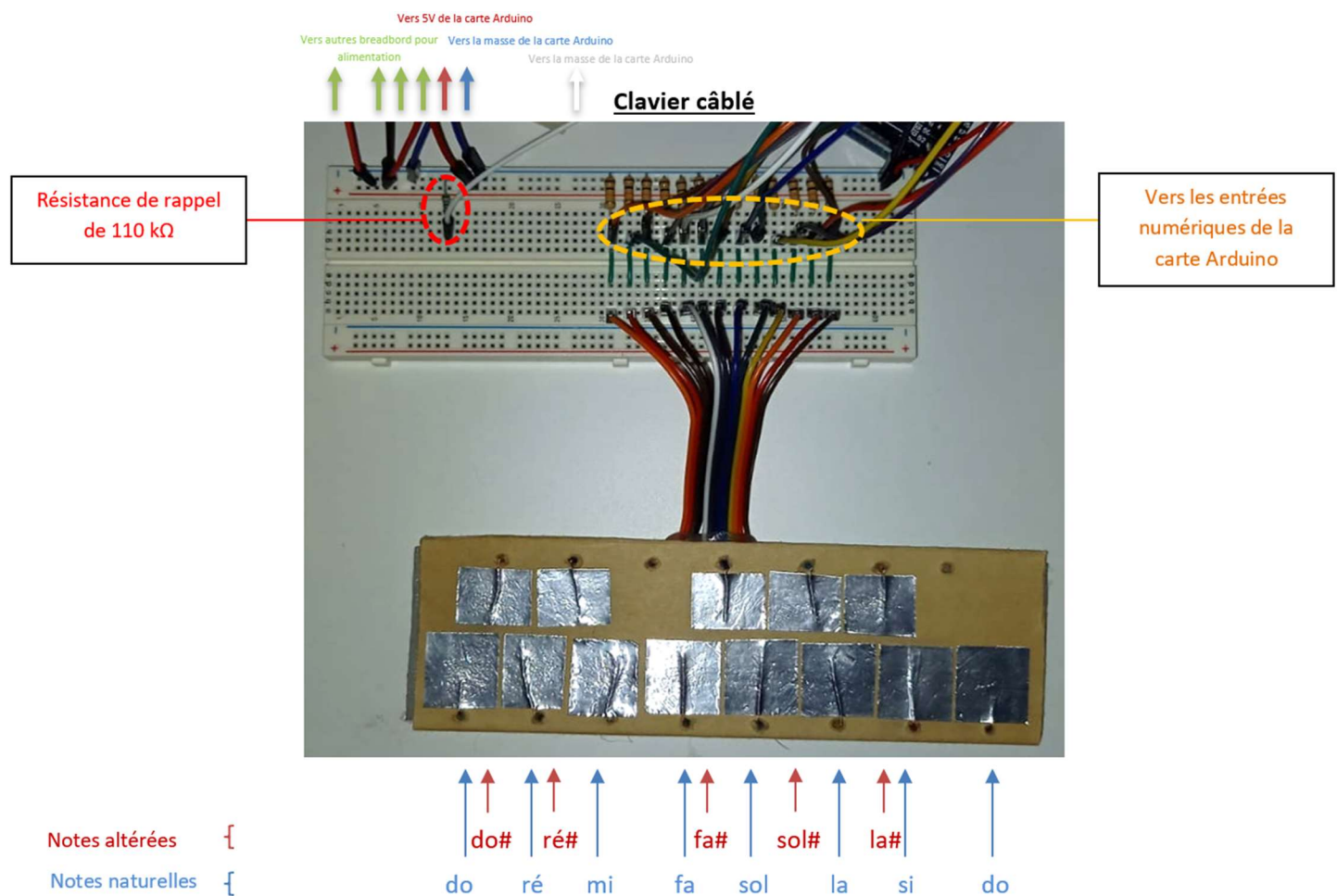
Collage des fils sur la face intérieure



Clavier terminé







Ce clavier servira à jouer de la main droite la mélodie.

### ➤ Module « production sonore »

Pour produire un son au moyen de la carte Arduino j'ai utilisé la commande "tone()" qui permet de faire osciller un signal de n'importe quelle sortie à la fréquence désirée. Pour transformer cette oscillation électrique en onde sonore, j'ai essayé différentes possibilités :

Possibilité 1 (déjà traitée dans mes comptes rendus précédents):

- Un buzzer piézoélectrique (de forte impédance)

Possibilité 2 : (déjà traitée dans mes comptes rendus précédents):

- Un module Amplificateur Haut-Parleur Grove

Possibilité 3 : (mis en œuvre pendant les congés de Noël)○ **Un haut-parleur****a) Branchement en « direct »**

Les haut-parleurs électrodynamiques ont une faible impédance (généralement 4 ou 8  $\Omega$ ). Si on branche un haut-parleur de 8  $\Omega$  directement entre l'une des sorties numériques de l'Arduino et la masse, cette sortie oscillera entre une tension de 0 volt et une tension de 5 volts.

En appliquant la loi d'Ohm on peut déterminer l'intensité du courant circulant dans le haut-parleur. Pour  $R = 8 \Omega$  et  $U = 5 \text{ V}$  on obtient  $I = U/R = 5/8 = 0,625 \text{ A} = 625 \text{ mA}$ .

Mais il n'est pas conseillé de faire circuler dans une sortie de l'Arduino un courant supérieur à 20 mA. Ce branchement « direct » risque donc de griller le microcontrôleur.

**b) Branchement avec une résistance en série**

Une solution possible est d'ajouter en série avec le haut-parleur une résistance de façon à limiter le courant à 20 mA.

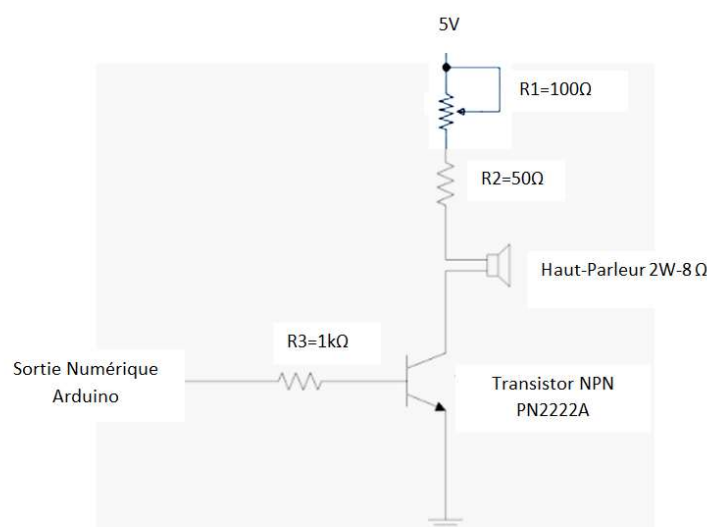
Nouvelle application de la loi d'Ohm :  $R = U/I = 5/0,020 = 250 \Omega$

La résistance du haut-parleur n'étant que de 8 Ohm, celui-ci n'émettra alors qu'un son de très faible intensité.

**c) Branchement via un transistor :**

Pour éviter que le courant circulant dans la sortie de l'Arduino alimentant le haut-parleur soit de trop forte intensité, j'ai commandé celui-ci via un transistor.

Un faible courant électrique circulera dans la sortie de l'Arduino et dans la base du transistor qui contrôlera le courant circulant dans le haut-parleur. (Voir le montage ci-dessous)



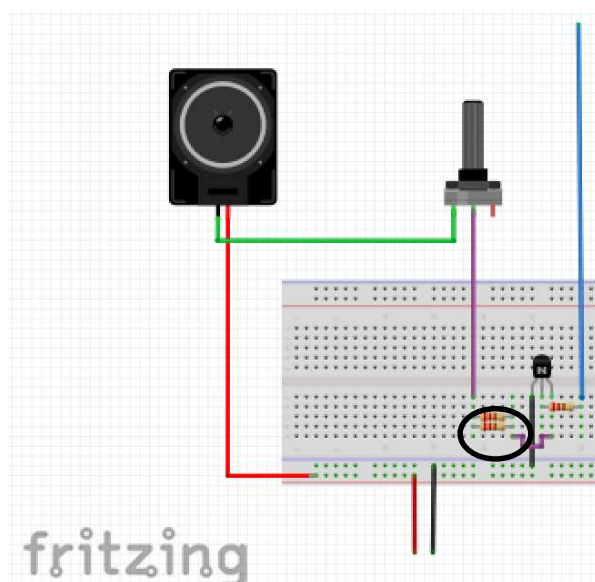
- Le potentiomètre R1 permet de limiter l'intensité du courant circulant dans le haut-parleur donc de régler le volume sonore de celui-ci.
- La résistance R2 placée avant le haut-parleur a pour but de limiter le courant circulant dans le haut-parleur lorsque potentiomètre R1 est équivalent à 0  $\Omega$ .
- La résistance R3 placée sur l'émetteur du transistor a pour but de limiter les courants d'appel, se produisant à chaque commutation du transistor.

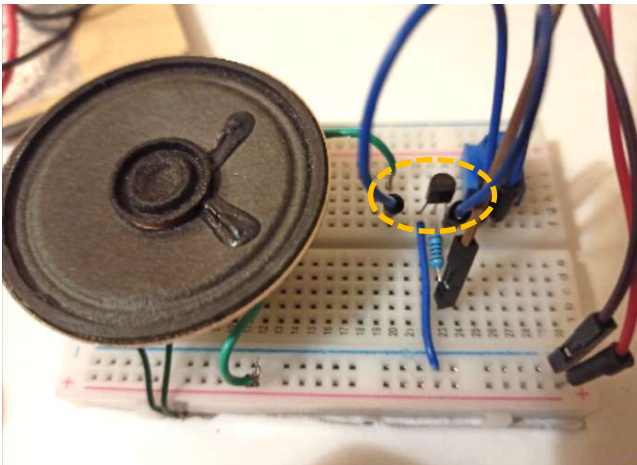
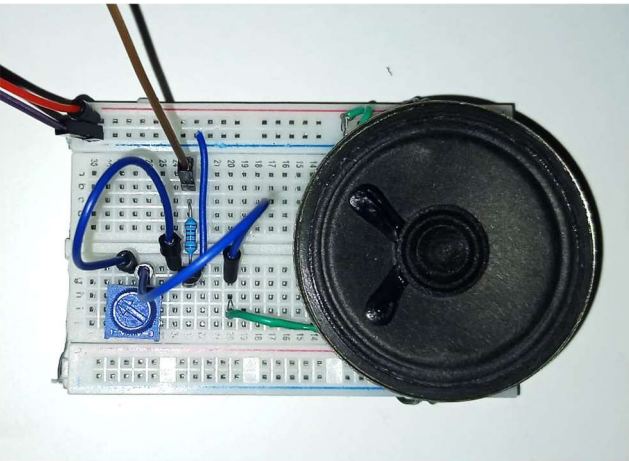
#### Choix des composants :

Le courant maximal pouvant circuler dans les sorties 5 volts de l'Arduino peut atteindre près de 500 mA et le calcul précédent a montré que le haut-parleur de 8  $\Omega$  soumis à une tension de 5 volts sera traversé par un courant de 625 mA. Cette intensité est un peu trop élevée (même si le transistor absorbera une petite fraction de ces 5 volts). J'ai donc ajouté une résistance de 50  $\Omega$  en série avec le haut-parleur (en réalité 2 résistances de 100  $\Omega$  branchées en parallèles).

La valeur du potentiomètre a été déterminée expérimentalement (ajouts successifs de résistances pour obtenir une modulation du volume sonore correcte).


Sortie Numérique  
Arduino






Photos du montage d'un haut-parleur « récupéré »

J'ai choisi et j'ai utilisé un transistor **PN2222** car il supporte des tensions supérieures à 6V et car le courant maximal du collecteur est de de 600 mA.

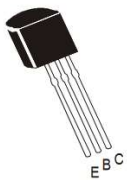


Continental Device India Limited  
An ISO/TS16949 and ISO 9001 Certified Company



PN2222  
PN2222A  
CBE  
TO-92

**NPN SILICON PLANAR SWITCHING TRANSISTORS**

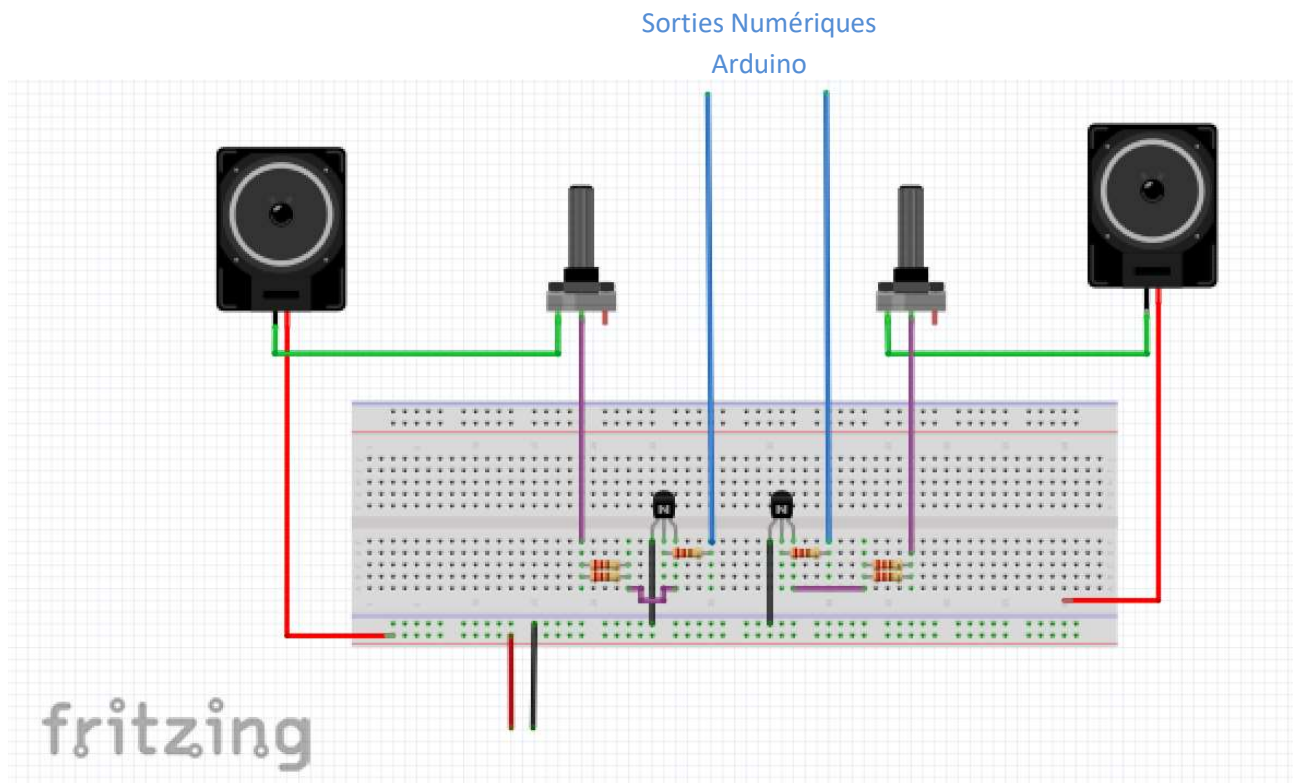


**Complementary Silicon Transistors For Switching And Linear Applications  
DC Amplifier & Driver For Industrial Applications.**

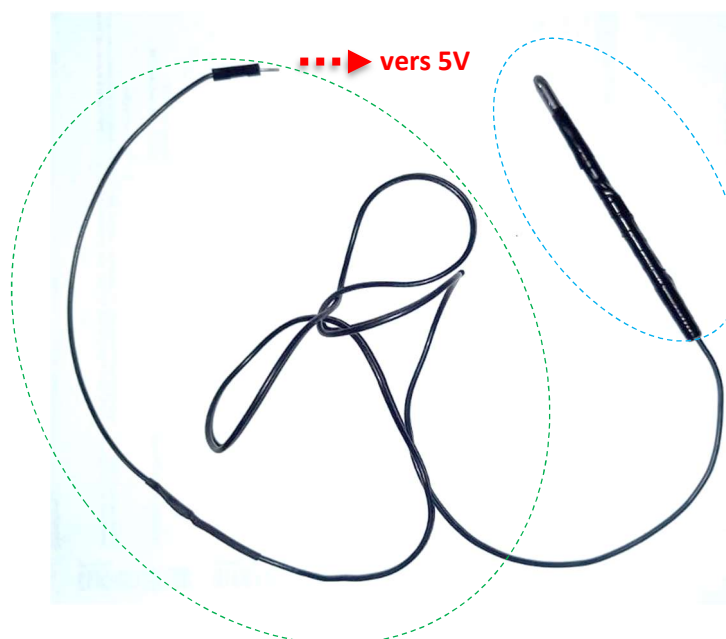
**ABSOLUTE MAXIMUM RATINGS**(Ta=25deg C unless otherwise specified)

DESCRIPTION	SYMBOL	2222	2222A	UNIT
Collector -Emitter Voltage	VCEO	30	40	V
Collector -Base Voltage	VCBO	60	75	V
Emitter -Base Voltage	VEBO	5.0	6.0	V
Collector Current Continuous	IC	600	625	mA
Power Dissipation @Ta=25 degC	PD	5	5	mW
Derate Above 25deg C				mW/deg C
@ Tc=25 degC	PD	1.5	1.5	W
Derate Above 25deg C				mW/deg C
Operating And Storage Junction Temperature Range	Tj, Tstg	-55 to +150	-55 to +150	deg C



Schéma du montage « stéréo » sur deux sorties➤ **Takstylo :**

Pour la fabrication du « Takstylo » du prototype du *Variophuino* j'ai utilisé un simple **clou de diamètre 4 mm** que j'ai ensuite relié à un **fil électrique** au Takstylo.



### ➤ Modification de la programmation :

Une fois les claviers terminés et les branchements faits, il m'a fallu modifier la programmation pour que le haut-parleur n'émette un son que lorsque le « Takstylo » était en contact avec la touche du clavier (partie argentée métallique). J'ai donc, pour un premier essai, écrit un programme basique composé uniquement de *if* et de *else* : Si l'entrée branchée à la touche N°1 (soit le do) lisait 5V, le buzzer émettait une fréquence de 262 Hz (soir un do<sub>3</sub>) sinon il ne produisait aucun son. J'ai effectué ce système de boucle sur 6 touches (car il me manquait la carte MEGA pour augmenter mon nombre d'entrées et j'ai préféré coder petit à petit).

### ➤ Carte Arduino Mega 2560



Nous avons changé notre carte **Arduino UNO** pour une carte **Arduino Elegoo Mega 2560**.

Cette carte comporte 54 I/O ce qui règle notre manque d'entrées numériques. En effet, il nous fallait 14 I/O de libres au total (13 pour les 13 touches, 1 pour le buzzer) sans compter la N°0 et 1.

Nous n'avons donc plus besoin d'utiliser le Multiplexeur/Démultiplexeur.

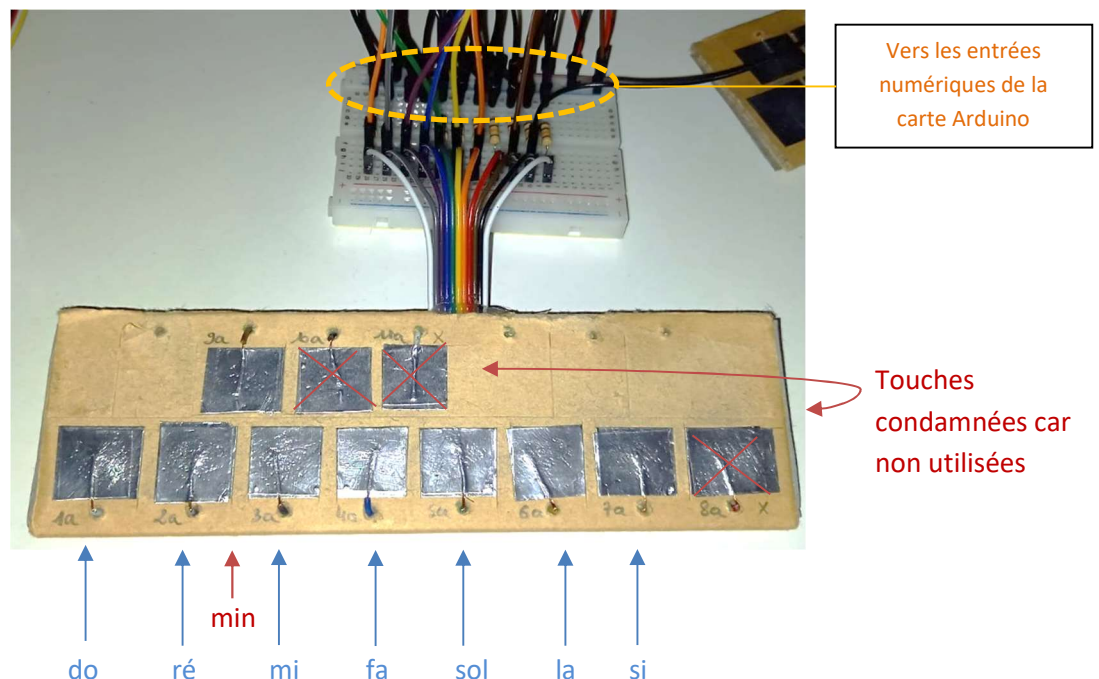
La carte MEGA a permis de régler un autre problème. Le programme pour jouer les notes et changer la fréquence à l'aide du potentiomètre est très coûteux en mémoire. Comme le montre le tableau ci-dessous, la carte MEGA a 4 fois plus de mémoire que la UNO.

NOM	PROCESSEUR	VOLTAGE	VITESSE CPU	ENTREES/ SORTIES/ ANALOG.	E-S DIGITALES/ PWM	EEPROM (kB)	SRAM (kB)	FLASH	UART
UNO	ATMEGA 328	5V/7-12V	16MHz	6/0	14/6	1	2	32	1
DUE	AT91SAM3X8E	3.3V/7-12V	84MHz	12/2	54/12	-	96	512	4
LEONARDO	ATmega32u4	5V/7-12V	16MHz	12/0	20/7	1	2.5	32	1
MEGA 2560	ATmega2560	5V/7-12V	16MHz	16/0	54/15	4	8	256	4
MEGA ADK	ATmega2560	5V/7-12V	16MHz	16/0	54/15	4	8	256	4
MICRO	ATmega32u4	5V/7-12V	16MHz	12/0	20/7	1	2.5	32	1
MINI	ATMEGA 328	5V/7-9V	16MHz	8/0	14/6	1	2	32	-
NANO	ATMEGA 328/ ATMEGA 168	5V/7-9V	16MHz	8/0	14/6	0.512	1/2	16/32	1

FIGURE 5 – Tableau comparatif de quelques cartes Arduino

source : <https://fablabutc.fr>

- **Fabrication du deuxième clavier en complément du premier : le « clavier main gauche » pour jouer un accompagnement**



Ce clavier servira à jouer l'accompagnement à la main gauche (soit des accords majeurs et mineurs en arpège). Il a été réalisé sur le même principe que le « clavier main droite » mais avec 7 touches « notes fondamentales » et 1 touche « effet ».

Il n'y aura pas d'effet pitchbend appliqué à ce clavier.

Lorsque l'on appuie sur la touche 1a (do) uniquement, le haut-parleur émet les 3 notes qui composent l'accord de fondamentale do en majeur (do mi sol) en arpège (soit do mi sol à la suite).

Lorsque l'on appuie sur la touche 1a (do) et 9a (min), le haut-parleur émet les 3 notes qui composent l'accord de fondamentale do en mineur (do ré# sol) en arpège (soit do ré# sol à la suite).

- **Effet Pitchbend :**

Pour avoir une idée globale du rendu, j'ai calculé, pour chaque note naturelle et altérée de l'octave 3 et 4, 3 fréquences intermédiaires. En effet, les potentiomètres à glissières que nous avons commandés étaient exponentiels. L'ancien étant linéaire, j'ai donc du coder en sorte que la fréquence change tous les  $\frac{1}{4}$  de longueur de la piste (soit tous les 15 mm). Je me suis aidée du moniteur série en affichant les différentes tensions en fonction de la longueur mais cela reste peu précis. Cependant, ce programme m'a permis d'avoir une première impression de l'effet pitchbend.

De plus, cela était long à calculer et coder. En effet, l'écart entre la note naturelle et altérée n'était jamais le même et plus je montais dans les octaves, plus celui-ci augmentait dans les hautes fréquences (*voir captures d'écran de code ci-dessous*).

C'est pour cela qu'il faudra, par la suite, écrire une fonction permettant de calculer automatiquement plusieurs fréquences intermédiaires en fonction de la note donnée.

do<sub>3</sub> → do#<sub>3</sub>

la#<sub>4</sub> → si<sub>4</sub>

```
if( b1 == 1 ) {
  int f = 262;

  if(real_tension < 0.35){
    tone(buzzer, f);
    Serial.println("DO3 : 262 Hz");
  }

  if((real_tension >= 0.35) && (real_tension < 0.8)){
    tone(buzzer, f+4);
    Serial.println("Modulation : 266 Hz");
  }

  if((real_tension >= 0.8) && (real_tension < 3)){
    tone(buzzer, f+7);
    Serial.println("Modulation : 269 Hz");
  }

  if((real_tension >= 3) && (real_tension < 4.8)){
    tone(buzzer, f+11);
    Serial.println("Modulation : 273 Hz");
  }

  if((real_tension >= 4.8) && (real_tension <= 5)){
    tone(buzzer, f+19);
    Serial.println("DO#3 : 277 Hz");
  }
}
```

```
if( b10 == 1){
  int f = 932;

  if(real_tension < 0.35){
    tone(buzzer, f);
    Serial.println("LA#4 : 932 Hz");
  }

  if((real_tension >= 0.35) && (real_tension < 0.8)){
    tone(buzzer, f+14);
    Serial.println("Modulation : 946 Hz");
  }

  if((real_tension >= 0.8) && (real_tension < 3)){
    tone(buzzer, f+28);
    Serial.println("Modulation : 960 Hz");
  }

  if((real_tension >= 3) && (real_tension < 4.8)){
    tone(buzzer, f+42);
    Serial.println("Modulation : 974 Hz");
  }

  if((real_tension >= 4.8) && (real_tension <= 5)){
    tone(buzzer, f+56);
    Serial.println("SI4 : 988 Hz");
  }
}
```

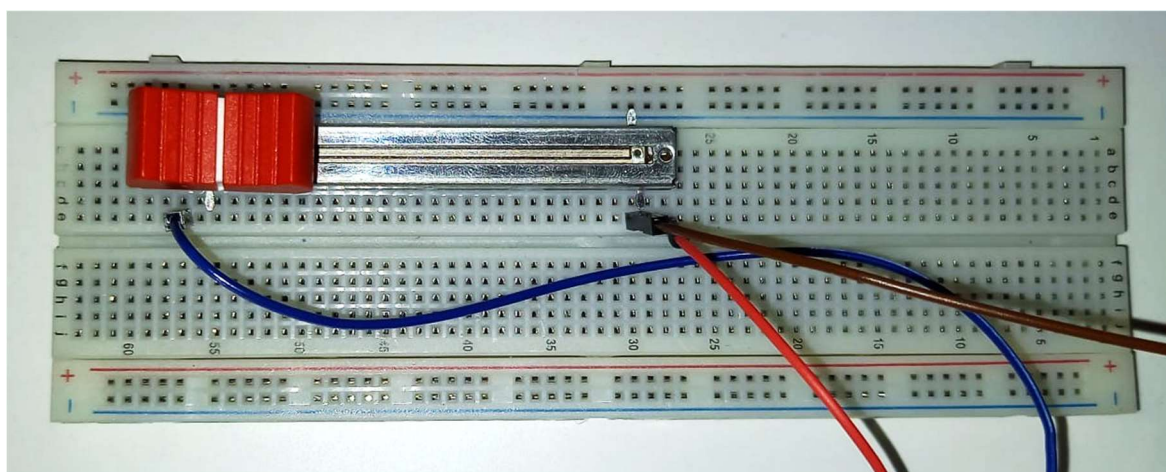


Photo du potentiomètre à glissière commandé

(le câble rouge est relié au 5V, le bleu au GND et le marron à l'entrée analogique A0 de la carte Arduino)



➤ **Prochains travaux :**

- Réaliser la partie « Production sonore Stéréo » (hardware et coder la partie stéréo),
- Finaliser les tests du prototype,
- Apprendre à souder des composants et des fils sur une plaque perforée,
- Commencer la réalisation du produit final (soudures),
- Etudier les différentes manières de jouer et d'accéder à tous les effets sonores (stylo, gants ou autre).