PROGRAMMEZ! #209 - juillet/août 2017 le magazine des développeurs

Créer son Makey Makey !

Jean Paul DELCROIX
Animateur multimédia et Fabmanager

• Nicolas THILL *stagiaire en 1ère SEN.*

Connaissez-vous la Makey Makey ? Cette carte électronique servant d'interface entre un PC et tous objets conducteurs, leur permettant de remplacer certaines touches du clavier. Vous n'avez jamais joué du piano sur des bananes ? Ou encore joué à Pacman avec des touches faites en pâte à modeler ?
Plus d'info dans cette vidéo : https://www.youtube.com/watch?v=rfQqh7iCcOU.

Je vous propose donc dans un premier temps de transformer votre Arduino Uno en pseudo Makey Makey ; puis dans un second temps d'adapter le code pour créer une Makey Makey à partir d'une Arduino Leonardo !
Pour réaliser une véritable Makey Makey, il faut utiliser une Arduino Leonardo, qui à la différence de la Uno, peut être considérée par le PC comme une souris ou un clavier. La Uno n'est détectée par l'ordinateur que comme un port série et non comme un périphérique USB.
Alors à quoi peut servir cette pseudo Makey Makey ? Eh bien elle n'est pas d'utilité avec un ordinateur, mais elle peut très facilement servir de lien entre un clou et une led, ou encore une cuillère et un servomoteur, ou bien de la pâte à modeler conductrice et un moteur continu…

Source

De nombreux sites nous annoncent que la Makey Makey et l'Arduino sont compatibles, ou encore que la Makey Makey est une dérivée de l'Arduino, mais même si nous trouvons quelques vidéos de personnes ayant réussi à le faire, ou à créer un shield, on ne trouve rien sur la méthode… La seule chose que nous ayons trouvée c'est cette vidéo en arabe dont mon tuto est une adaptation : https://www.youtube.com/watch?v=7rHi2XcFqgw.
Nous avons essayé de comprendre le principe, de plus une partie du code n'était pas apparente sur la vidéo, il nous a fallu la trouver… J'ai également modifié un peu le code supprimant certaines parties et changeant le nom des variables. Le code d'origine faisait s'allumer 3 leds. Je vous propose d'appuyer sur une banane et d'actionner une led témoin et un servomoteur !!!

Matériel

- Une carte Arduino et son câble USB
- 1 mini servomoteur 9g
- 9 câbles / jumpers
- 1 résistante de 1 MΩ et 1 résistance de 220
- 1 led de 5 mm
- 2 pinces crocodiles

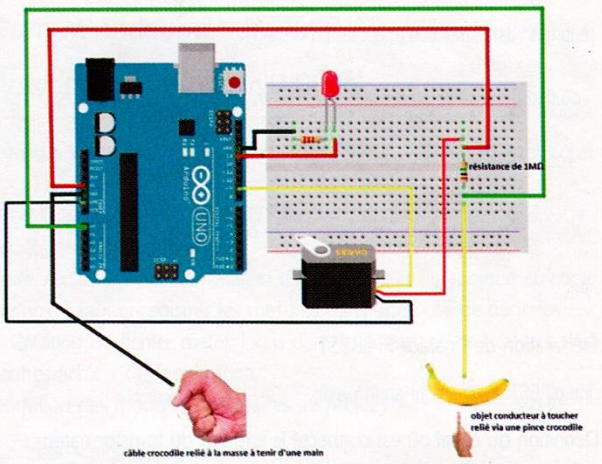
Budget : environ 32€10 sur Gotronic

Etape 1 : le montage

Il s'agit ici en fait de 3 montages distincts, le premier pour la Makey Makey (objet conducteur à toucher), le second pour le servomoteur, et le dernier (optionnel) pour une led témoin :

- Pour obtenir le ***montage Makey Makey***, le ground de l'Arduino doit être branché à une pince croco et tenu à la main.

La broche 5V, doit être connectée à une résistance de 1MΩ, qui elle-même est branchée à la fois à la broche A0 de l'Arduino et à une pince croco pour le relier à l'objet conducteur.



- Le ***servomoteur*** est branché d'un côté au ground, au 5V de l'Arduino, et à la broche 9 de l'Arduino. On peut imaginer remplacer le servomoteur par d'autres composants.
- La ***led témoin*** est branchée d'un côté au ground, via une résistance, ici de 220 , et de l'autre à la broche 13 de l'Arduino. Elle est optionnelle, mais permet de visualiser lorsque le courant passe.

Etape 2 : le code : Principe de base

L'Arduino envoie un courant de 5V qui va diminuer à cause de la résistance de 1MΩ et de la "résistance" rencontrée entre les 2 pinces crocos, donc entre l'objet conducteur et le corps de la personne touchant cet objet. Ce qui sera donc variable en fonction d'une part des personnes et des objets reliant les 2 pinces crocos.
Dans notre code, on appellera cette résistance entre les 2 pinces "résistivité".
Si cette dernière est inférieure au seuil qu'on définit (c'est à dire qu'il y a suffisamment de courant qui passe) alors la led s'allumera et le servomoteur se positionnera à l'angle défini.
Par exemple : les différentes personnes ayant servi de cobaye avaient une résistivité variant entre 2200 et 3700.
Dans le cas d'un dessin fait à la mine de crayon gris (graphite) la valeur de "résistivité" était un peu moins de 5000, une banane également testée tournait entre 5000 et 6000. De plus, cette résistivité à tendance à diminuer avec le temps.

Etape 3 : Déclaration des variables et des constantes

1) Pour la fonction Makey Makey

```
int sensorPin = A0;
int sensorValue = 0;
```

```
int values[10];
int k=0;
int resistivite=0;
```

Explication : on crée une variable de type "int" nommée "sensorPin" qui est affectée à la broche analogique A0 de l'Arduino.
On crée une variable de type "int" nommée "sensorValue" permettant d'y stocker la valeur que la carte détectera, elle est au départ initialisée à 0.
On crée un tableau nommé "values" contenant 10 cellules pour y stocker des variables de type "int". Les cellules sont pour l'instant vides.
On crée une variable de type "int" nommée "k", initialisée à 0, et qui va nous permettre de calculer la "résistivité".
On crée une variable de type "int" nommée "résistivité" qui symbolise en quelque sorte la résistivité électrique. Plus elle est élevée, moins le courant passe.

2) Led et servomoteur

```
int ledPin = 13;
```

```
#include <Servo.h>
Servo monservo;
```

Etape 4 : Void Setup

```
Serial.begin(9600);
```

```
pinMode(ledPin, OUTPUT);
```

```
monservo.attach(9);
```

Etape 5 : Void Loop

1) Pour la fonction makey

```
sensorValue = analogRead(sensorPin);
 for (k=0;k<10; k++){
   values[k]    = values[k+1]; }
   values[9]    = sensorValue;
   resistivite = 0;

for (k=0; k<10; k++) {
resistivite = resistivite + values [k]; }

 Serial.println(resistivite);
```

Explication : on définit la valeur de la variable "sensorValue" comme étant ce que l'on lit sur la broche analogique où sensorPin est affectée (donc ici A0).
On crée une fonction for, où la variable "k" est initialisée à la valeur 0, et où si elle est inférieure à 10, elle est incrémentée (augmentée) de 1 en 1.
On y stocke dans le tableau "values" les valeurs de la variable "k", chaque cellule du tableau prenant la valeur suivante de "k".
La dernière cellule du tableau, la cellule 9, prendra la valeur de la variable "sensorValue, donc de ce que la carte détecte sur la broche A0.
On initialise la variable "résistivité" à 0.
On relance la boucle for pour déterminer la valeur de la variable "résistivité".
La variable "résistivité" aura comme valeur "résistivité" + la valeur présente dans les cellules du tableau "values".
On affiche dans le moniteur série la valeur de la variable "résistivité". Plus la valeur est basse, plus l'objet est conducteur. Lorsqu'on fait toucher les 2 pinces crocos, le courant passe normalement, il n'y a pas de résistance la valeur de la variable "résistivité" est à 0.
Dans mon cas, lorsque rien ne touche les pinces crocos, donc seul l'air ambiant fait passer l'électricité entre les 2 pinces, le moniteur série affiche une valeur d'environ 10 200.
Si je touche les 2 pinces crocos, donc seul mon corps sert de conducteur ; "résistivité" est à environ 3600, pour d'autres personnes l'ayant testée, cela varie entre 2200 et 3700.
Dans le cas d'un dessin fait à la mine de crayon gris (graphite) la valeur de "résistivité" est un peu moins de 5000, une banane également testée tourne entre 5000 et 6000. Il faudra donc peut être modifier la valeur dans la condition if en fonction de l'objet à « cliquer ».

2) Pour la led et le servomoteur

```
if (resistivite < 5000){
   digitalWrite(ledPin, HIGH);
   }

 if (resistivite < 5000){
   monservo.write(90);
   delay(500);
   monservo.write(0);
   }

 else{
digitalWrite(ledPin, LOW);
   }
```

Explication : si la valeur de la variable "résistivité" est inférieure à 5000 (cette valeur peut être changée pour améliorer la sensibilité), alors on allume la led.
De même, si la valeur de la variable "résistive" est inférieure à 5000, on positionne le servomoteur à l'angle choisit ici 90°.
On crée un délai d'une demi seconde avant de refaire la boucle. Attention durant ce délai la led reste allumée.
On repositionne le servomoteur à l'angle choisit ici à 0°.
Sinon, c'est à dire que la valeur de la variable "résistive" est supérieure à 5000, on éteint la led. Pas besoin de commande pour le servomoteur, puisqu'il est déjà repositionné.

RÉALISATION D'UNE MAKEY MAKEY À PARTIR D'UNE ARDUINO LEONARDO

Pour ce cas de figure, il nous faut donc une carte Arduino Leonardo qui pourra émuler les touches du clavier et la souris. Pour rappel également seules 6 touches peuvent être émulées puisque la carte ne possède que 6 entrées analogiques.

Etape 1 : le montage (visualisation sur Fritzing)

Le montage est identique au précédent, si ce n'est qu'il ne se compose que de 2 montages distincts, le premier pour la Makey Makey (objet conducteur à toucher) que l'on reproduit 6 fois pour pouvoir connecter 6 objets conducteurs différents, le second (optionnel) pour des leds témoins :
• pour la ***Makey Makey***, même câblage. Pour rappel, le ground de

l'Arduino doit être branché à une pince croco et tenu à la main. La broche 5V, doit être connectée à une résistance de 1M , qui elle-même est branchée à la fois à la broche A0 de l'Arduino et à une pince croco pour le relier à l'objet conducteur. L'opération étant répétée pour brancher les différents objets sur les 6 broches analogiques de la Leonardo, allant de A0 à A5.

- pour les ***leds témoins,*** on branche chaque Led d'un côté au ground, via une résistance, ici de 220 , et de l'autre à une broche digitale de l'Arduino, par exemple de la PIN 8 à la PIN 13. Ce montage est toujours optionnel, mais permet à la fois de visualiser lorsque le courant passe, et en y affectant des leds de couleurs différentes, de pouvoir s'y retrouver plus facilement.

Etape 2 : Principe de base de code

Le principe de base de ce montage reste le même, mais les réglages doivent cependant être plus fins, car le taux de résistivité fluctue sans cesse, et la carte fait de nombreux relevés lors de la phase d'appui sur l'objet conducteur.

En d'autres termes, durant le temps que l'on passe à appuyer sur notre banane, même s'il ne dure qu'une seconde, la carte fera des dizaines de relevés de la résistivité.

Cela n'est pas gênant en soi, lorsqu'il s'agit de faire tourner un servomoteur comme dans notre premier montage, puisque la fonction « delay » liée au servomoteur empêche toute autre action. Le servomoteur tournant pour atteindre l'angle voulu et restant en position pendant toute la durée du « delay ». Aucune autre action ne peut être entreprise.

Mais lorsqu'il s'agit d'émuler une touche du clavier ou un clic cela peut s'avérer problématique. En effet, nous ne pouvons pas associer de « delay » à une touche pressée, car celle-ci exécuterait un nombre conséquent de « pressions » durant toute la durée du delay. Par exemple, si en appuyant sur une banane cela représente la pression de la touche espace ; associé un delay de 1 seconde reviendrait à appuyer sur cette touche pendant une seconde, ce qui créerai un nombre conséquent d'espaces.

De même, une autre problématique entre en ligne de compte : une fois l'appui sur la banane effectué, en relâchant celle-ci, la résistivité ne remonte pas directement à son niveau ambiant (par exemple à 10230 pour mon cas), ce qui peut également être considéré par l'Arduino comme un nouvel appui de la touche espace.

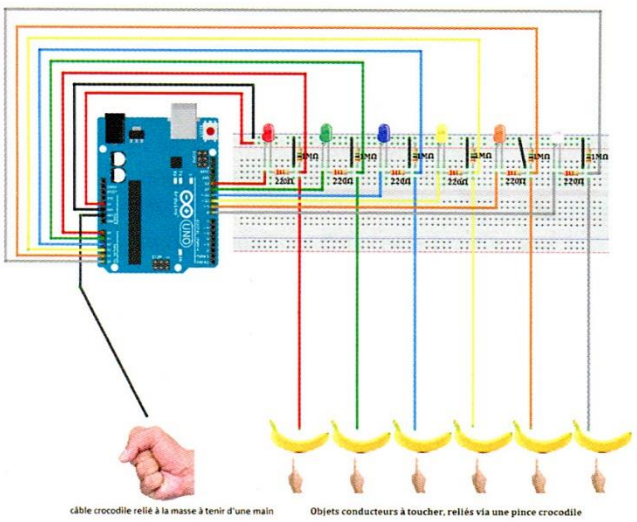
Il est donc important d'une part de rajouter une temporisation dans le loop, lorsque l'Arduino calcule la résistivité. Dans mon cas j'ai choisi un lapse de temps de 1500 millisecondes, afin de ne pas avoir des relevés en cascade, mais uniquement tous les 1500 millisecondes.

D'autre part, il vous faudra choisir un seuil de résistivité assez proche de la résistivité ambiante. En effet, l'appui sur notre objet conducteur doit être comme celle d'une touche : rapide et non maintenue, pour que lorsqu'on appuie sur notre banane la résistivité passe directement en dessous du seuil choisi et qu'elle remonte immédiatement après relâchement pour atteindre sa valeur par défaut qui est celle de l'air ambiant.

Pour affiner ces réglages il vous faudra vous aider du moniteur série du logiciel Arduino IDE, pour connaitre d'une part la résistivité ambiante, donc celle qui s'affichera lorsque vous ne cliquez pas sur votre objet, mais également connaître la résistivité rencontrée lorsque vous touchez l'objet conducteur, afin de définir dans votre code le seuil optimal.

Attention ! Cette opération peut s'avérer fastidieuse, puisque la résistivité ambiante dépend de nombreux paramètres (humidité, température ambiante ...). De même la résistivité obtenue lorsqu'une personne clique sur l'objet conducteur est également variable d'une part en fonction de la personne, mais également de l'objet conducteur. Et dans ce domaine une banane n'est pas équivalente à une autre !

Entrons maintenant dans le code.



câble crocodile relié à la masse à tenir d'une main — Objets conducteurs à toucher, reliés via une pince crocodile

Etape 3 : Déclaration des variables et des constantes

1) Pour la fonction Makey Makey

Voir code complet sur www.programmez.com

Explication : aucun changement par rapport à notre précédent montage, nous utilisons le même principe que nous déclinons en 5 fois, puisque la makey makey possède 5 fonctions. Pour plus de facilité chaque objet conducteur est associée à une led de couleur différente.

Pour rappel, nous créons des variables de type int, que l'on nomme sensor_Pin suivies de la couleur de la led et que l'on associe à une des entrées analogiques de la carte Arduino. Ainsi l'objet branché à la PIN A0 sera relié à la led rouge ; celui sur la PIN A1, à la led verte ; la A2 à la led Bleue ; la A3 à la led Jaune ; la A4 à la led Orange et finalement la A5 à la led Blanche.

Sur la même idée nous créons également des variables de type int nommées sensorValue suivies également de la couleur de la led, que l'on initialise à 0, qui nous permettront d'y stocker la valeur que la carte détectera.

S'ensuit pour chaque objet la création d'un tableau de 10 cellules pour y stocker les variables, et la création de variables de type "int" nous permettant de calculer la "résistivité" et dont les noms seront représentés par des lettres, que j'ai choisies arbitrairement allant de k à p.

On finit par la création de variable de type "int" nommée "résistivité et le nom de la couleur" qui symbolise en quelque sorte la résistivité électrique. Plus elle sera élevée, moins le courant passera.

2) Pour les leds

```
int ledPin_Rouge = 13;
int ledPin_Verte = 12;
int ledPin_Bleue = 11;
int ledPin_Jaune = 10;
int ledPin_Orange = 9;
int ledPin_Blanche = 8;
```

Explication : on crée une suite de variables de type "int" nommée "ledpin suivie de la couleur de la led" et affectée à l'une des broches de l'Arduino, pour cela on utilisera les PIN 8 à 13 de la carte.

3) Pour le clavier et la souris

```
#include <Keyboard.h>
#include <Mouse.h>
```

Explication : afin d'émuler un clavier et la souris nous faisons appel à deux bibliothèques.

Etape 4 : Void Setup

1) Pour la fonction makey

```
Serial.begin(9600);
```

Explication : on définit le taux de transfert de données à 9600 bauds.

2) Pour les leds

```
pinMode(ledPin_Rouge, OUTPUT);
pinMode(ledPin_Verte, OUTPUT);
pinMode(ledPin_Bleue, OUTPUT);
pinMode(ledPin_Jaune, OUTPUT);
pinMode(ledPin_Orange, OUTPUT);
pinMode(ledPin_Blanche, OUTPUT);
```

Explication : on définit les broches connectées à la variable "ledPin suivie de la couleur de la led" comme étant des sorties.

3) Pour le clavier et la souris

```
Mouse.begin();
 Keyboard.begin();
}
```

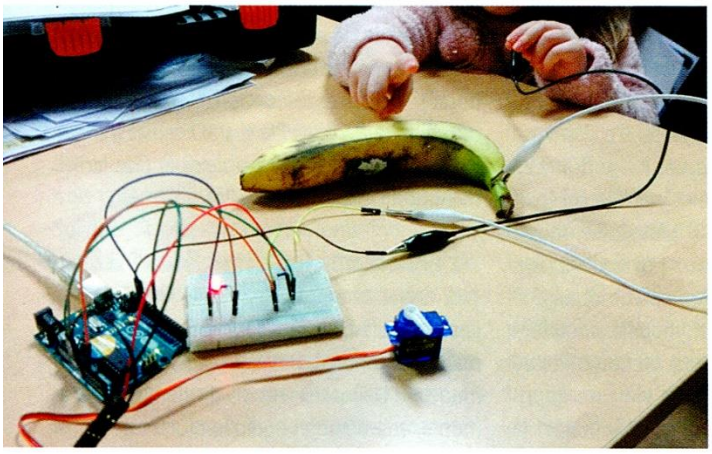
Explication : on initialise la communication avec le clavier et la souris. Et on n'oublie pas de fermer l'accolade du void setup.
https://www.arduino.cc/en/Reference/KeyboardModifiers
https://www.arduino.cc/en/Reference/MouseClick

Etape 5 : Void Loop

1) Pour la fonction makey

Voir code complet sur www.programmez.com

Explication : Encore une fois on se retrouve avec 6 blocs, chacun représentant un objet connecté symbolisé par une led de couleur différente. Dans chaque bloc, on définit comme pour le premier code les "sensorValue" et des variables nommées par des lettres allant par exemple de K à p pour nous permettre de calculer la "résistivité".



On affiche dans le moniteur série la valeur de la variable "résistivité". Plus la valeur est basse, plus l'objet sera conducteur. Pour rappel, lorsqu'on fait toucher les 2 pinces crocos, le courant passe normalement, il n'y a pas de résistance ; la valeur de la variable "résistivité" est à 0.
Dans mon cas, l'air ambiant donne une valeur d'environ 10 200 à 10 300, un corps humain varie entre 2200 et 3700, de la mine de crayon gris (graphite) est à un peu moins de 5000, et une banane tournera entre 5000 et 6000. Il faudra donc peut être modifier la valeur dans la condition if en fonction de l'objet à « cliquer ».
Finalement on temporise par une fonction « delay » ici à 1500 millisecondes, mais dont vous devrez certainement modifier la valeur en fonction de vos besoins. Pour rappel, ce lapse de temps est nécessaire pour éviter que l'Arduino ne fasse une cascade de relevés et n'enchaîne ce qu'elle considérera comme de multiples appuis.

2) Pour les leds témoins

Voir code complet sur www.programmez.com

Explication : si la valeur de la variable "résistivité" liée à chaque couleur est inférieure à 5000 (cette valeur peut être changée pour améliorer la sensibilité), alors on allume la led de couleur correspondante.
Sinon, c'est à dire que la valeur de la variable "resistive" est supérieure à 5000, on éteint la led.

3) Pour les touches du clavier et le clic gauche de la souris

Voir code complet sur www.programmez.com

Explication : si la valeur de la variable "résistivité" liée à chaque couleur est inférieure à 5000, alors on fait appel à la commande Keyboard.press(chiffre) ; de la bibliothèque Keyboard pour émuler la touche correspondante au chiffre saisi dans les parenthèses. Par exemple 218 correspond à la flèche du haut, alors que le 178 correspond à la barre espace. On utilise ensuite la commande Keyboard.releaseAll() ; pour simuler la fin l'émulation de toutes les touches.
De même pour émuler le clic gauche de la souris on utilise la commande Mouse.click() ; de la bibliothèque Mouse, que l'on arrête par la commande Mouse.release() ;
Et on n'oublie pas de fermer l'accolade du loop.

4) Et si on émulait d'autres touches ?

Le principe reste le même il suffit juste de choisir le chiffre correspondant à la touche. Par exemple :

128 pour Ctrl gauche
129 pour Shift gauche
130 pour Alt gauche
131 pour Touche Windows
132 pour Ctrl droit
133 pour Shift droit etc.

Adapter, modifier et vous l'approprier !

Ce projet est une base améliorable en fonction de vos besoins et de vos envies. Vous pourrez vous en inspirer et les modifier.

D'autres exemples sur mon site https://ardwinner.jimdo.com/arduino/ •