

Harpe Laser
DAC
&
Convertisseur Unipolaire-Bipolaire

Frédéric Wauters 2019
Belgique

Table des matières

Introduction	3
Besoin.....	3
A propos de l'Arduino	3
A propos de la conversion Unipolaire-Bipolaire	3
A propos de l'utilisation de plusieurs galvanomètres	3
Le circuit	4
1. Brancher le circuit	4
a. À l'Arduino.....	4
b. À l'alimentation	4
c. Au contrôleur	5
d. Rappel des fils et des connexions.....	5
2. L'offset	5
3. Le gain	6
Le programme de réglage.....	7
1. Prérequis.....	7
2. Protocole de test	7
3. Le montage de test	7
4. Le code	7
Le programme de test	9
1. Prérequis.....	9
2. Protocole de test	9
3. Le montage de test	9
4. Le code	9

DAC – Convertisseur Unipolaire-Bipolaire

Introduction

Pour les besoins du développement de la harpe laser, nous devons fournir au contrôleur du galvanomètre une tension analogique allant de -5V à +5V.

Besoin

Le contrôleur du galvanomètre fonctionne au standard ILDA¹. Dans cette optique, il est nécessaire de lui fournir une tension de -5V représentant la position la plus à gauche, ou au contraire, +5V représentant la position la plus à droite. Dans le cas de l'application de 0V, le galvanomètre se placera au milieu de l'angle qu'il couvre.

Le contrôleur nécessite également la même tension inverse sur son autre sortie. Par exemple, lorsque sur la première connexion -5V sont appliqués, il est nécessaire d'appliquer +5V sur l'autre et inversement. Si +1V est appliqué sur la première entrée, -1V sera appliqué sur l'autre entrée.

Note : Le GND des deux circuits doivent être connectés ensemble ! Les mesures de tensions indiquées ci-avant sont mesurées par rapport à la masse (GND).

A propos de l'Arduino

L'Arduino ne fournit pas de tension analogique mais bien PWM². Il est impossible de nous en servir en l'état. Nous devons donc utiliser un DAC³ (Digital to Analog Converter). Celui-ci communique avec l'Arduino en utilisant un protocole I2C⁴. Ce protocole ne requiert que trois fils et permet de communiquer une valeur allant jusqu'à 12 bits dans notre cas. Ceci signifie que nous pouvons programmer une vraie tension analogique de 0 à 5V avec une précision de 4096 paliers. Chacun de ces paliers représente donc 1,2 mV !

L'Arduino MEGA (2560) dispose en standard des connexions nécessaires pour le I2C.

A propos de la conversion Unipolaire-Bipolaire

Grâce au DAC, nous disposons d'une tension analogique précise allant de 0 à 5V. Le convertisseur Unipolaire-Bipolaire va, non seulement, convertir cette tension en -5V à +5V, mais aussi fournir son inverse (+5V à -5V) nécessaire au contrôleur.

A propos de l'utilisation de plusieurs galvanomètres

En standard, les DAC répondent à la même adresse I2C. Dans le cas où plusieurs circuits de ce type doivent être utilisés, il faudra opérer une très légère modification du circuit et du programme de sorte que chacun dispose d'une adresse distincte.

Note : durant les tests, suivez bien ce que le programme de l'Arduino vous indique dans le moniteur série.

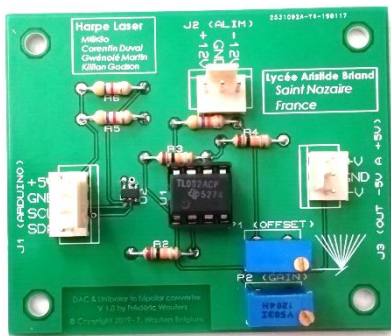
¹ ILDA : International Laser Display Association : www.ilda.com

² PWM : Pulse Width Modulation : fr.wikipedia.org/wiki/Modulation_de_largeur_d%27impulsion

³ DAC : Convertisseur digital/analogique : MCP 4724 (Microchip®)

⁴ I2C : Inter-Integrated Circuit : fr.wikipedia.org/wiki/I2C

Le circuit



Le circuit imprimé a été réalisé en Chine pour sa qualité et son prix de revient. Sa conception est telle qu'il est protégé contre les parasites extérieurs à l'aide de plans de masse.

A gauche se trouve un connecteur permettant de dialoguer avec l'Arduino, en haut, un autre connecteur permet l'alimentation en + et -12V continu, enfin, le connecteur de droite est à relier au contrôleur.

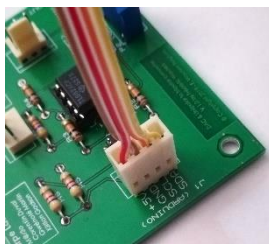
Les circuits sont livrés pré-réglés. Ils disposent d'un réglage d'offset et d'un réglage de gain. Grâce à ces potentiomètres multi tours de précision, le circuit peut être parfaitement réglé à tous moments. Après livraison et avant mise en œuvre dans la harpe, il est recommandé de vérifier si les réglages n'ont pas bougé.

Le DAC est le petit circuit intégré à gauche (U2). L'autre circuit intégré consiste en 2 amplificateurs opérationnels (U1).

1. Brancher le circuit

Chaque connecteur dispose des indications nécessaires, cependant, voici comment brancher le circuit dans la harpe laser :

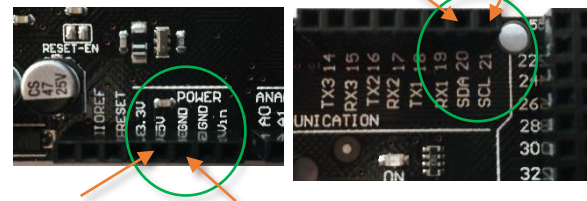
a. À l'Arduino



Utilisez le connecteur à quatre broches fourni (fils marron, rouge, orange, jaune) et branchez-le à gauche de la plaque (connecteur J1 nommé « Arduino »). Grâce au détrompeur, vous évitez tout branchement erroné (sens).

La nomenclature de chacun des fils est indiquée sur le PCB à hauteur de chacune des connexions.

Connectez le fil rouge au 5V de l'Arduino, le fil marron au GND de l'Arduino, le fil orange à la borne SCL (21) de l'Arduino et le fil jaune à la borne SDA (20) de l'Arduino.



b. À l'alimentation



Utilisez le connecteur à trois broches fourni (blanc, noir gris). ATTENTION : un autre câble à trois fils est également fourni. Vérifiez bien les couleurs.

Utilisez ce câble pour connecter le circuit à votre alimentation stabilisée. Le fil blanc doit être connecté au +12V, le fil noir à la masse (GND) et le fil gris au -12V.

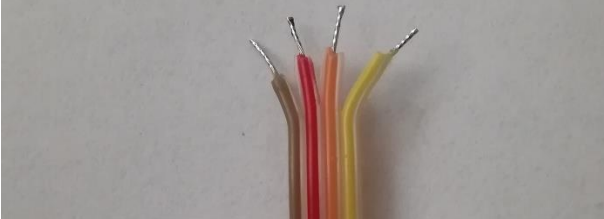

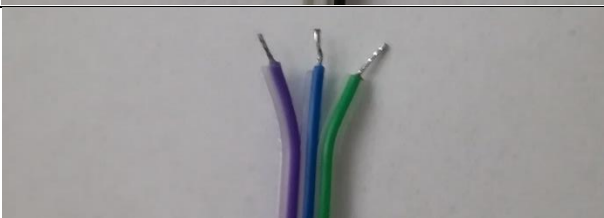
c. Au contrôleur



Utilisez le connecteur à trois broches fourni (violet, bleu, vert) pour connecter le circuit au contrôleur de galvanomètre.

Connectez le fil vert (+V) au « + » du contrôleur, le fil GND au GND du contrôleur et le fil (-V) au « - » du contrôleur. Attention : vous devez connecter les fils au connecteur de signal d'entrée du contrôleur. Il ne s'agit pas des + et - d'alimentation du contrôleur, celui-ci doit être alimenté avec l'alimentation fournie avec le galvanomètre et le contrôleur.

d. Rappel des fils et des connexions

	<u>Vers Arduino :</u> MARRON : GND ROUGE : +5V ORANGE : SCL (21) JAUNE : SDA (20)
	<u>Vers alimentation stabilisée :</u> GRIS : -12V BLANC : +12V NOIR : GND
	<u>Vers le contrôleur du galvanomètre :</u> VIOLET : -V BLEU : GND VERT : +V

2. L'offset

Il permet de régler le circuit de sorte que les tensions extrêmes soient identiques mais inverses. Pour bien comprendre, voici un tableau :

Tension d'entrée	Tension de sortie	A REGLER
0V	-1V	
2.5V	+3V	
5V	+8V	

Tension d'entrée	Tension de sortie	A REGLER
0V	-9V	
2.5V	-2V	
5V	+5V	

Tension d'entrée	Tension de sortie	PARFAIT
0V	-5V	
2.5V	0V	
5V	+5V	

3. Le gain

Il permet de régler la plage de tension en sortie. Par exemple, la sortie doit aller de -5V à +5V (soit une plage de 10V). Dans ce cas, le gain est parfaitement réglé. Voici de nouveaux tableaux pour la compréhension :

Tension d'entrée	Tension de sortie	A REGLER (plage : 20V)
0V	-10V	
2.5V	0V	
5V	+10V	

Tension d'entrée	Tension de sortie	A REGLER (plage : 6V)
0V	-3V	
2.5V	0V	
5V	+3V	

Tension d'entrée	Tension de sortie	PARFAIT (plage : 10V)
0V	-5V	
2.5V	0V	
5V	+5V	

Note : Les réglages (offset et gain) mesurés sur la sortie « + » ne doivent pas être répétés sur la seconde sortie (« - »). D'autre part, le réglage du gain peut avoir une influence sur l'offset. Pendant le réglage du gain, vérifiez donc régulièrement si l'offset est toujours bon et, au besoin, corrigez le réglage.

Le programme de réglage

1. Prérequis

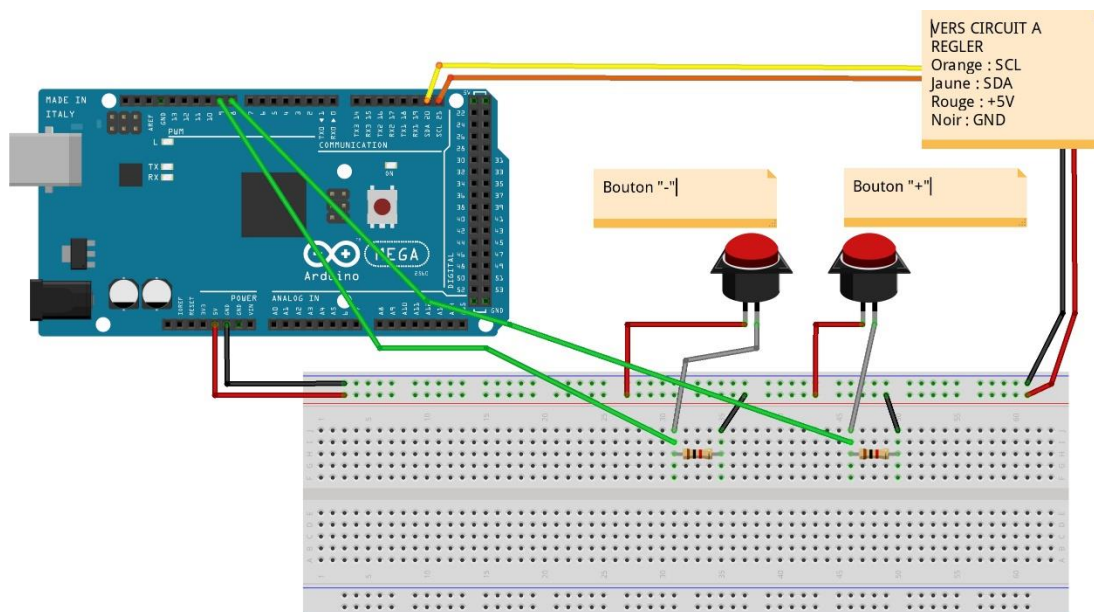
Comme le circuit utilise un DAC (MCP4725), il est nécessaire d'utiliser une librairie adaptée nommée « MCP4725.h ». Cette librairie est téléchargeable sur le site de *Rinky-Dink Electronics*⁵.

2. Protocole de test

Il s'agit dans ce test de régler l'offset et le gain. Dans cet ordre d'idée, le programme permet, à l'aide des boutons poussoir + et – de passer respectivement à 5V ou à 0V.

Lorsque la tension est de 0V, vous devez obtenir -5V en sortie V+. A l'inverse, lorsque la tension de sortie est de 5V, vous devez obtenir +5V en sortie V-.

3. Le montage de test



4. Le code

Le programme est livré sans garantie et permet des réglages simples.

```
// Librairie pour le DAC
#include <MCP4725.h>

// Declarations
MCP4725 dac(SDA, SCL);
int laserPin = 52;
int plusPin = 8;
int moinsPin = 9;
bool Status=false;

// Configuration avant programme principal
void setup() {
  // Initialisation du DAC
```

⁵ Rinky-Dink Electronics : www.rinkydinkelectronics.com/library.php?id=75

```

dac.begin();

// Initialisation des pins
pinMode(laserPin, OUTPUT);
digitalWrite(laserPin, HIGH);

// Pour les boutons tests
pinMode(plusPin, INPUT);
pinMode(moinsPin, INPUT);

// Ouverture du port serie
Serial.begin(9600);
while (!Serial) {

}
}

// Programme principal (boucle)
void loop() {
  // Si Plus
  if (digitalRead(plusPin)==HIGH) {
    // Si on n'etait pas encore dans cet etat
    if (Status !=true) {
      // On a change
      Serial.println("5V");
    }

    // Monter a 5V
    dac.setValue(4095);

    // Memoriser
    Status = true;
  }

  // Si Moins
  if (digitalRead(moinsPin)==HIGH) {
    if (Status !=false) {
      // On a change
      Serial.println("GND");
    }

    // Descendre a 0V
    dac.setValue(0);

    // Memoriser
    Status = false;
  }
}
}

```


Le programme de test

Il s'agit à présent de placer à présent le contrôleur du galvanomètre dans le circuit.

1. Prérequis

Comme le circuit utilise un DAC (MCP4725), il est nécessaire d'utiliser une librairie adaptée nommée « MCP4725.h ». Cette librairie est téléchargeable sur le site de *Rinky-Dink Electronics*⁶.

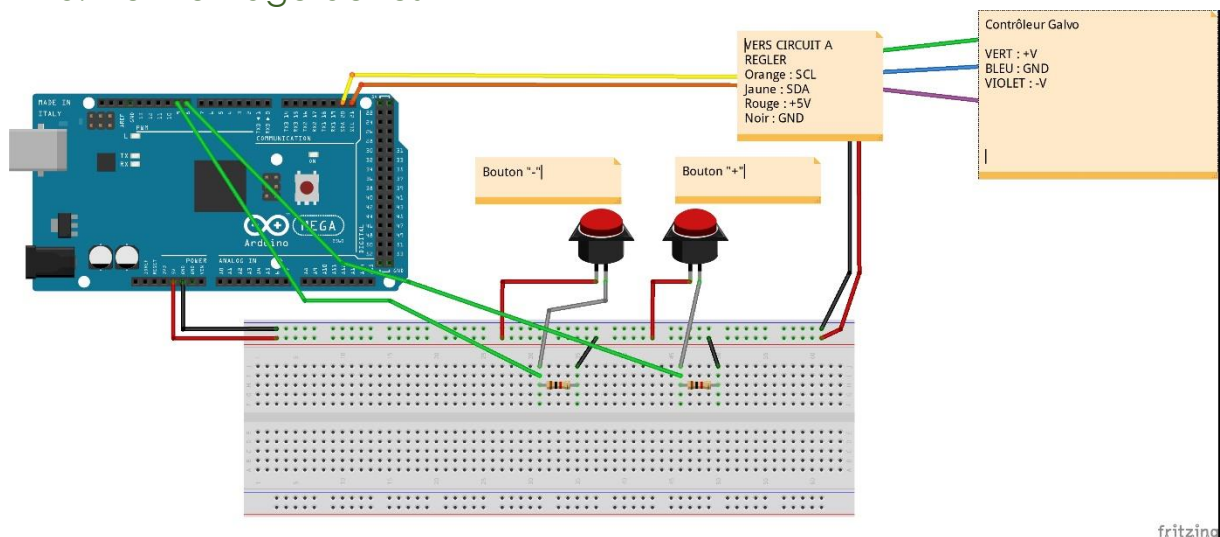
2. Protocole de test

Il s'agit dans ce test de vérifier que tout fonctionne correctement. A l'aide du bouton + et du bouton -, on peut passer d'une phase de test à une autre.

Le galvanomètre se positionne d'abord à l'extrême gauche. A chaque pression du bouton +, le galvanomètre se déplace à la position suivante (vers la droite). Le programme est configuré pour 12 faisceaux équidistants.

La phase de test suivante allume les douze faisceaux simultanément (mode harpe) sans éteindre le laser. Il vous revient d'ajouter cette fonctionnalité. Dès lors, il est naturel qu'une traînée soit visible entre les faisceaux.

3. Le montage de test



4. Le code

```
// Librairie pour le DAC
#include <MCP4725.h>

// Declarations
MCP4725 dac(SDA, SCL); // DAC
int nbCordes = 12;      // Nombre de cordes
int laserPin = 52;      // Laser
int plusPin = 8;        // Bouton Plus
int moinsPin = 9;       // Bouton Moins
float Status=0;         // Variable de travail

// *****
// * Setup * (Configuration avant programme principal)
```

⁶ Rinky-Dink Electronics : www.rinkydinkelectronics.com/library.php?id=75

```

// *****
void setup() {
  // Ouverture du port serie (pour le debuggage dans le moniteur serie)
  Serial.begin(9600); // Initialisation
  while (!Serial) { // Attente initialisation ok
  }

  // Moniteur
  Serial.println("Moniteur série : OK");

  // Initialisation du DAC
  dac.begin();

  // Moniteur
  Serial.println("DAC initialisé : OK");

  // Initialisation des pins
  pinMode(laserPin, OUTPUT); // Laser
  pinMode(plusPin, INPUT); // Bouton Plus
  pinMode(moinsPin, INPUT); // Bouton Moins

  // Moniteur
  Serial.println("Boutons initialisés : OK");

  // S'assurer que le laser est bien eteint
  digitalWrite(laserPin, LOW);

  // Placer le scanner au milieu
  // (2048 est la moitié de 4096 => entre 0 et 5v : 2.5V => entre -5 et +5 :
0v
  dac.setValue(2048);

  // Moniteur
  Serial.println("Laser éteint et scanner au centre : OK");
}

// *****
// * Loop * (Programme principal (boucle))
// *****
void loop() {
  // Variables de travail
  int cordeCourante=0;

  // PREMIER TEST : Positionner
  dac.setValue(Status);

  // Attente de l'opérateur
  //Serial.println("Test de positionnement - Appuyez sur 'PLUS' pour
démarrer...");

  // Si Plus
  if ((digitalRead(plusPin)==HIGH) && (Status < 4095)) {
    // Position suivante
    Status = Status + (4096/nbCordes);
    Serial.println(Status);
    dac.setValue(Status);
  }
}

```

```

// Moniteur
Serial.println("PLUS - corde :");
Serial.println(Status, DEC);

// Pause
delay(1000);

// Eviter les debordement
if (Status > 4095) Status = 4095;
}

// Si Moins
if ((digitalRead(moinsPin)==HIGH) && (Status > 0)) {
    // Position suivante
    Status = Status - (4096/nbCordes);
    dac.setValue(Status);

    // Moniteur
    Serial.println("MOINS - corde :");
    Serial.println((4096/Status), DEC);

    // Pause
    delay(1000);

    // Eviter les debordement
    if (Status < 0) Status = 0;
}

// SECOND TEST : toutes les cordes
if (Status == 4095) {
    // Status a 0
    Status=0;

    // Attente de l'operateur
    Serial.println("Appuyer sur 'PLUS' pour le test suivant");

    // Boucler tant qu'on appuie pas sur PLUS
    while (digitalRead(plusPin)==LOW) {
    }

    // Pour revenir au test precedent, appuyer sur MOINS
    Serial.println("Test des cordes - Appuyez sur MOINS pour revenir au test
précédent");

    // Boucler tant qu'on appuie pas sur MOINS
    while (digitalRead(moinsPin)==LOW) {
        // Positionner la corde suivante
        while (cordeCourante < nbCordes) {
            cordeCourante = cordeCourante + 1;

            // Positionner le scanner
            dac.setValue(cordeCourante*(4096/nbCordes));

            // Stabilisation
            delay(3);

```

```
    }  
  
    while (cordeCourante > 0) {  
        cordeCourante = cordeCourante - 1;  
  
        // Positionner le scanner  
        dac.setValue(cordeCourante*(4096/nbCordes));  
  
        // Stabilisation  
        delay(3);  
    }  
}  
}
```