

Sortie analogique isolée pour Arduino Uno

Giovanni Carrera (Italie)

Dans ce projet, j'utilise la sortie PWM de l'Arduino Uno pour réaliser une sortie analogique complètement isolée avec une plage de 0 à 5 V ou plus. L'utilisation d'un signal PWM moyenné est incomparable avec un vrai DAC, mais offre plus de simplicité puisqu'il suffit d'utiliser un optocoupleur pour isoler le signal PWM numérique.

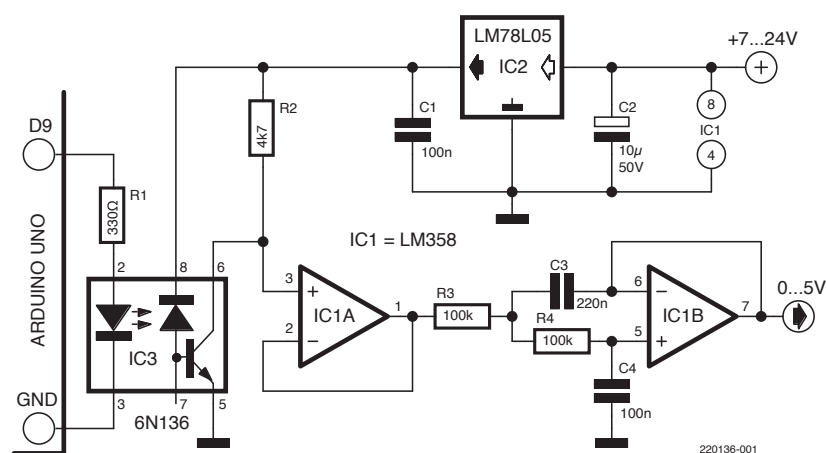


Figure 1. Circuit simple.

Dans les systèmes de contrôle des installations industrielles, il est toujours conseillé d'isoler à la fois les entrées et les sorties du secteur. Cela permet d'éviter les perturbations causées par des surtensions, des coups de foudre ou d'autres sources d'interférences électromagnétiques, ainsi que par des différences de potentiel de terre.

L'Arduino Uno ou tout autre système établi avec un ATmega328 ne dispose pas d'une véritable sortie analogique, mais il est possible d'en réaliser une avec une sortie PWM, moyennée avec un filtre passe-bas. L'utilisation d'un signal PWM moyenné avec un paramétrage de 8 bits n'est pas comparable à un véritable DAC. Cependant, avec une isolation, elle présente des avantages incontestables de simplicité puisqu'il suffit d'utiliser un optocoupleur pour isoler le signal numérique PWM.

Le PWM de l'Arduino

Un Arduino Uno a plusieurs broches (D3, D5, D6, D9, D10 et D11) qui peuvent être configurées en sortie PWM. Pour ce projet, j'ai utilisé la broche D9 car les autres étaient déjà utilisées par divers appareils (LCD, SD et RTC).

Le signal PWM des broches D9 et D10 est généré par Timer1 de

l'ATmega328. Il possède un prescaler qui divise par 1, 8, 64, 256, 1024, piloté par les trois bits de poids faible du registre TCCR1B. La valeur par défaut du prescaler définie par l'EDI Arduino est égale à $N_p = 64$ (TCCR1B, bits 2..0 = 110), ce qui donne une fréquence de sortie :
Fréquence PWM = $\text{CPUClock} / (2 \times N_p \times \text{TOP}) = 16000000 / (2 \times 64 \times 255) = 490,196 \text{ Hz}$

où TOP est la valeur maximale du timer/compteur.

Si vous utilisez un optocoupleur commun avec un phototransistor, comme le 4N25, la fréquence sera limitée à cause des temps de transition élevés. J'ai donc utilisé un optocoupleur plus rapide avec photodiode et avec une sortie à collecteur ouvert, tel que le 6N136. Pour éliminer le bruit de sortie, j'ai utilisé un filtre Sallen-key, passe-bas actif du second ordre, avec une fréquence de coupure d'environ 11,2 Hz.

Circuit

Le circuit représenté sur la **figure 1** est assez simple. L'isolation est réalisée à l'aide d'un optocoupleur ; bien entendu, vous devez utiliser pour ce circuit une alimentation différente de celle utilisée pour l'Arduino. Si vous ne voulez pas d'isolation, les choses deviennent encore plus

simples. Connectez le filtre à la sortie PWM. Dans ce cas, même la source de référence IC2 n'est pas nécessaire.

Je vous recommande d'utiliser un amplificateur opérationnel double compatible avec une alimentation mono-rail pour IC1, par exemple le LM358. Le circuit LM358 doit être alimenté avec une tension supérieure à 7 V (et inférieure à 32 V) pour avoir une tension de sortie de 5 V. Et notez bien : le régulateur a également une chute de tension de 2 V. Le collecteur ouvert de l'optocoupleur a un avantage : il permet d'obtenir facilement une plage de sortie différente. Par exemple, en utilisant une tension de référence de 10 V et une résistance $R2 = 10\text{ k}\Omega$, la plage de sortie devient 0... 10 V. Dans ce cas, vous devez remplacer le LM78L05 par un LM317 avec un montage approprié.

Il est indispensable de mesurer les condensateurs utilisés pour le filtre avec un capacimètre. Pour mon prototype, j'ai choisi pour C3 des condensateurs de 220 nF pour tenter d'obtenir une valeur approchant les 200 nF. Et pour C4, j'ai choisi une valeur égale à la moitié de C3.

Test

La **figure 2** montre les résultats de la régression linéaire sur les 14 points de mesure effectués sur mon prototype. Les conditions de test sont :

- Fréquence PWM = 490,196 Hz
- $V_{in} = 12\text{ V}$
- $V_{ref} = 5,00\text{ V}$

L'erreur type est d'environ 6,1 mV. Les résultats sont donc très bons avec la fréquence PWM par défaut.

J'ai également testé le système avec une fréquence de 3921,569 Hz, mais avec une erreur type de 39 mV. Les erreurs les plus importantes sont constatées pour des valeurs de rapport cyclique élevées. Dans cette plage, les impulsions sont étroites et le temps de montée est élevé ; ce phénomène crée une non-linéarité. La période est : $T = 1/3921,569 = 255\text{ }\mu\text{s}$. L'impulsion plus étroite a une durée d'environ 1 μs , soit approximativement la même valeur que le temps de montée des impulsions. Ce phénomène est à l'origine de la non-linéarité. En utilisant la fréquence par défaut de 490,196 Hz, l'impulsion minimale a une durée huit fois plus grande, ce qui améliore grandement la linéarité.

Programme Arduino

Pour tester le système, j'ai utilisé un Arduino Uno avec un LCD et l'entrée analogique A0 connectée à un potentiomètre

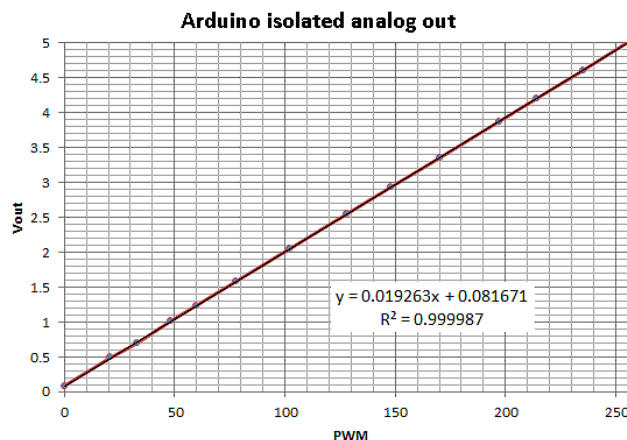


Figure 2. Courbe d'étalonnage.



Listage 1 : code source [1]

```
// program to test Arduino Uno PWM at 3.9 kHz
// G. Carrera 30 sept 2016

#include <LiquidCrystal.h>

int PWMpin = 9;          // PWM out on digital pin 9
int analogPin = 0;       // potentiometer connected to A0
int val = 0;             // variable to store the read value
char spacestring[17] = "                ";

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(7, 6, 5, 4, 3, 2);

void setup() {
  pinMode(PWMpin, OUTPUT); // sets the pin as output
  lcd.begin(16, 2);        // set up number of columns and rows
  lcd.setCursor(0, 0);     // set the cursor to column 0, line 0
  lcd.print("Stalker PWM"); // Print a message to the LCD
}

void loop() {
  val = analogRead(analogPin) >> 2; // read the potentiometer as 8 bit
  analogWrite(PWMpin, val);
  val = 255-val;                  // complement
  lcd.setCursor(0, 1);
  lcd.print(spacestring);
  lcd.setCursor(0, 1);
  lcd.print(val);
  delay(500);
}
```



LISTE DES COMPOSANTS

Résistances

R1 = 330 Ω, ± 5 %

R2 = 5.1 kΩ, ± 5 %

R3 = 100 kΩ, ± 5 %

R4 = 100 kΩ, ± 1 % film métallique

Condensateurs

C1 = 100 nF, céramique

C2 = 10 μF, 50 V, électrolytique

C3 = 200 nF, Mylar ± 2 %

C4 = 100 nF, Mylar ± 2 %

Semi-conducteurs

IC1 = LM358

IC2 = LM78L05

IC3 = 6N136

Divers

Carte Arduino Uno

pour faire varier le rapport cyclique du signal PWM. Le code source du croquis Arduino est donné dans le **listage 1**.

(220136-04) — VF : Asma Adhimi

Des questions, des commentaires ?

Envoyez un courriel à l'auteur (g.carrera@elettronicaemake.it) ou contactez Elektor (redaction@elektor.fr).



PRODUITS

> **Arduino Uno Rev3 (SKU 15877)**
www.elektor.fr/15877

> **Livre en anglais « Ultimate Arduino Uno Hardware Manual », W. A. Smith (Elektor 2021, SKU 19678)**
www.elektor.fr/19678

LIEN

[1] Code source : www.elektormagazine.fr/summer-circuits-2022

Vous souhaitez publier votre montage dans le magazine ?

Rendez-vous sur la page du labo d'Elektor : www.elektormagazine.fr/labs pour y enregistrer votre projet.

Cliquez sur « Créer un projet ». Connectez-vous (créez un compte gratuit si vous n'en avez pas encore). Remplissez les différents champs du formulaire.

Votre proposition de montage sera examinée par l'ensemble des rédacteurs du magazine. Si votre projet est retenu pour sa publication dans le magazine, un rédacteur prendra contact avec vous pour vous accompagner dans la rédaction de l'article.



Labo d'Elektor :
www.elektormagazine.fr/labs
créer > partager > vendre

