

stroboscope RVB

avec Arduino

un instrument utile,
instructif et distrayant

Roel Arits (Pays-Bas)

Le dictionnaire nous apprend que le stroboscope (du grec *strobos*, tournant) est un instrument destiné à faire apparaître immobiles ou animés d'un mouvement lent des objets animés d'un mouvement périodique rapide. Une succession d'éclairs à une fréquence appropriée permettent de ne voir qu'une phase déterminée du phénomène : l'objet mobile paraît immobile ou il semble se mouvoir au ralenti.

Il y a quelques décennies, le stroboscope avait trouvé une application très courante, dans les garages automobiles. Pour qu'un moteur à combustion interne à essence fonctionne correctement, les bougies d'allumage de chaque cylindre doivent produire une étincelle à un moment précis. Cette distribution était assurée par un dispositif mécanique (sensible à l'humidité dans certaines marques de voitures), qui devait être synchronisé avec précision avec le moteur. Et c'est là que notre stroboscope entre en jeu.

Le nombre d'interrupteurs du distributeur correspondait à celui des cylindres du moteur. Chaque fois qu'un tel contact s'ouvrait, l'énergie électrique emmagasinée dans la bobine d'allumage était déchargée à travers la bougie d'allumage dans la chambre de combustion sous la forme d'une étincelle qui enflammait le mélange air-carburant. Et cela devait se produire au moment précis où le piston dans le cylindre venait de quitter son point mort haut (c'est-à-dire que le mélange avait atteint sa compression maximale). Les pistons du moteur sont solidaires du vilebrequin, au bout duquel est arrimée la poulie de la courroie trapézoïdale. La position de la poulie est synchrone avec la position du piston. Sur cette poulie se trouvait une marque blanche et à côté, sur un support métallique, une deuxième marque fixe.

Le stroboscope utilisé pour régler l'allumage était constitué d'un tube à éclats au xénon ou au néon et d'un câble de déclenchement relié à la bougie de référence par un capteur inductif, de sorte que chaque fois que la bougie de référence s'allumait, le stroboscope produisait un éclair.

Puisque le stroboscope était déclenché par la bougie de référence, les deux marques devaient apparaître exactement en face l'une de l'autre en raison de l'effet stroboscopique. En cas de décalage entre les deux marques, le distributeur devait être ajusté. Dans cette application, le stroboscope était synchronisé avec le mouvement de rotation, mais cela fonctionne aussi sans signal de déclenchement !

Sans déclenchement ?

Pour mesurer une rotation avec un stroboscope non synchronisé, il faut ajuster la fréquence des éclairs de sorte que la marque sur le disque tournant semble quasi immobile et qu'une seule marque soit visible. Si la roue tourne dans le sens horaire et que le marqueur semble se déplacer lentement dans le sens horaire, c'est que la fréquence de clignotement est un peu faible. L'éclair arrive un peu trop tard, et la marque semble se déplacer dans le sens de rotation. Si la roue tourne

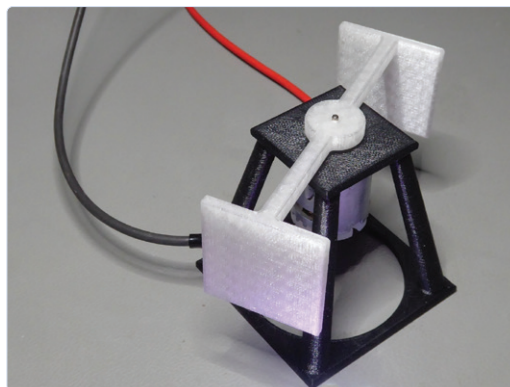


Figure 1. Le cadre avec les réflecteurs fixés à l'hélice.

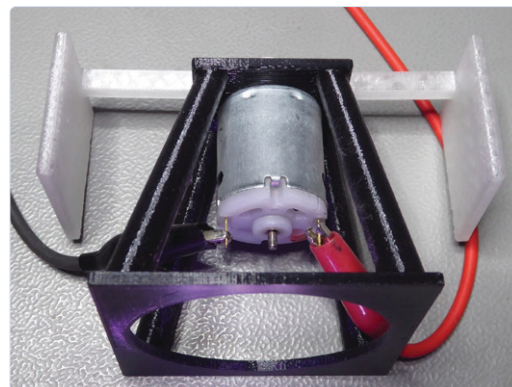


Figure 2. Gros plan du moteur.

dans le sens antihoraire et que le marqueur semble se déplacer lentement dans le sens horaire, c'est que la fréquence des éclairs est un peu trop élevée : chaque éclair arrive un peu trop tôt. Si plusieurs marques quasi fixes apparaissent, c'est que la fréquence des éclairs est un (sous-)multiple de la vitesse de rotation.

La vitesse peut être mesurée en ajustant la fréquence de clignotement du stroboscope de manière à immobiliser le marqueur. La durée de chaque éclair doit être assez courte pour obtenir une marque nette. Si la durée du flash est trop longue par rapport à sa fréquence, la marque bave.

Éclairs bariolés...

Et si notre stroboscope produisait plusieurs éclairs, de même fréquence mais de couleur différente, et avec un déphasage entre les différents éclairs ? Et si nous faisons tourner un objet blanc sur lui-même tout en variant fréquence, déphasage et largeur des éclairs colorés ?

Ce n'est pas difficile avec un microcontrôleur. Pour produire les éclairs de couleur, nous pouvons utiliser une LED RGB – ou mieux encore trois LED obtenir des éclairs plus lumineux. Pour commander la fréquence, le déphasage et la durée des éclairs, ce sera un Arduino Pro Mini, une carte qui dispose de suffisamment de ports d'entrée/sortie et largement assez rapide pour cette tâche. Les LED seront commandées par des signaux modulés en largeur d'impulsion (PWM).

Nous aurions pu utiliser trois modules PWM spéciaux de l'Arduino Pro Mini, mais ceux-ci utilisent trois temporisateurs différents. Il est donc plus délicat de les synchroniser et de programmer un déphasage précis. La fréquence des signaux PWM requis est relativement basse et une résolution de 16 bits hors de proportion avec nos besoins. Des signaux PWM produits par du logiciel dit *softPWM* feront amplement l'affaire. Le *softPWM* émet ses signaux sur des sorties numériques normales. Celles-ci sont commandées et initialisées à l'aide d'un compteur. Une interruption est émise lorsqu'un certain compte est atteint. De cette façon, on obtient un intervalle suffisamment court pour ajuster la largeur d'impulsion ou le déphasage du signal PWM avec une précision suffisante.

Pratique

Pour faire tourner l'hélice, nous avons un moteur 12 V à courant continu (MOT3N) pour lequel j'ai confectionné un petit support. L'hélice est prolongée par deux réflecteurs disposés verticalement. Ce dispositif

(fig. 1 et 2) a été produit avec une imprimante 3D au moyen de fichiers téléchargeables [3].

Note : à vide, le moteur 12 V utilisé tourne à environ 11500 tr/min, ce qui s'est excessif pour l'hélice imprimée en 3D. Lors du premier essai, elle s'est brisée en mille morceaux sous la force centrifuge. Au cours de telles expérimentations, le port de lunettes de protection est recommandé !

Pour le ralentir, le moteur fonctionne donc sous seulement 3 V environ. La tension est réglable pour synchroniser la vitesse. Sous 3 V, la vitesse est d'environ 2100 tr/min, ce qui correspond à 35 Hz. Comme l'hélice est équipée de deux réflecteurs (ce qui en facilite l'équilibrage), la fréquence d'éclair est deux fois plus élevée, c'est-à-dire 70 Hz (durée de la période environ 14 ms).

Le circuit (fig. 3) est la simplicité même et peut facilement être construit sur une plaque d'essais (fig. 4). Les trois LED RVB sont pilotées par de modestes BC547 sous les ordres de l'Arduino. La tension d'alimentation du circuit est de 5 V continus, la consommation est d'environ 500 mA. Le stroboscope est commandé par une minuterie d'interruption appelée toutes les 100 µs. Cet intervalle de 0,1 ms est la résolution avec laquelle on peut faire varier la fréquence de l'éclair. Cela laisse assez de temps pour modifier l'image stable produite par l'hélice.

```
// Configuration d'une minuterie de 16 bits en
// fonctionnement normal avec interruption à 100 µs
// 16 MHz/1 = 6,25 ns. Pour obtenir 100µs, nous
// devons donc laisser le minuteur compter 1600
// ticks.
TCCR1A = 0;
TCCR1B = _BV(CS10); //prédiviseur - diviser par 1
TCNT1 = 0xFFFF - 1600; // dépassemnt à 65535 = 0xFFFF
TIMSK1 = _BV(TOIE1); // interruption de dépassemnt
TCNT1 = 0;
sei();
}
ISR (TIMER1_OVF_vect)
{
TCNT1 = 0xFFFF - 1600; //100 us interruption
...

```

La période d'éclair est codée en dur dans le logiciel à 144 – c'est le nombre d'interruptions de la minuterie entre deux éclairs consécutifs.

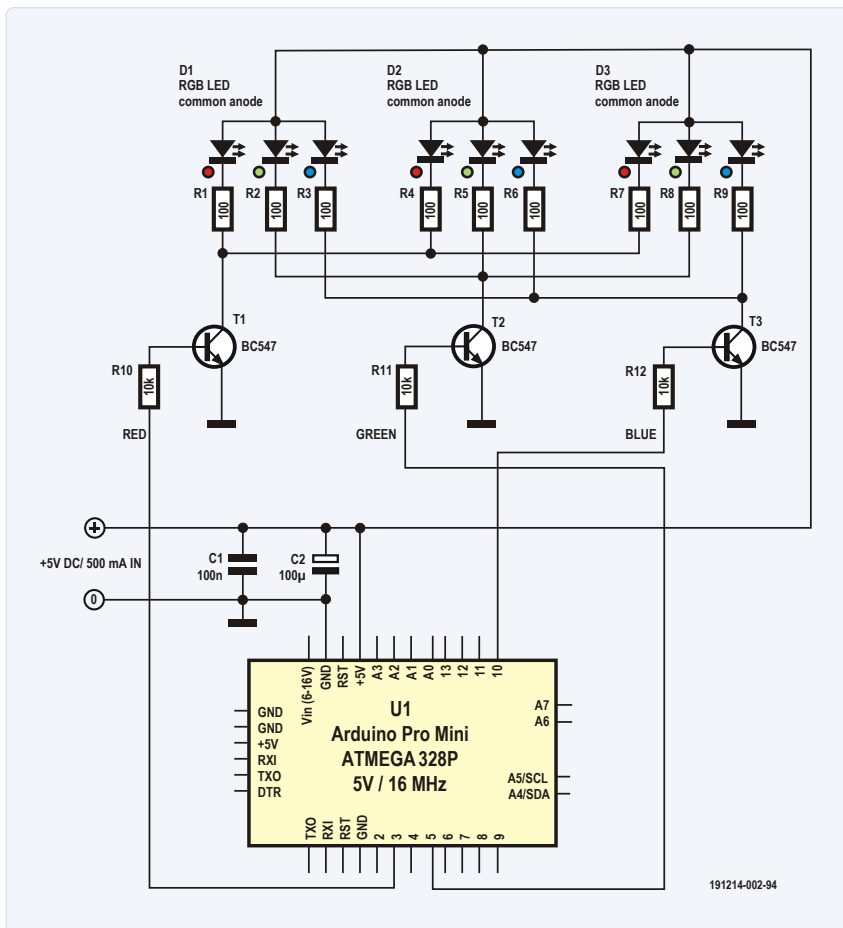


Figure 3. Pas trop compliqué, mon circuit ?

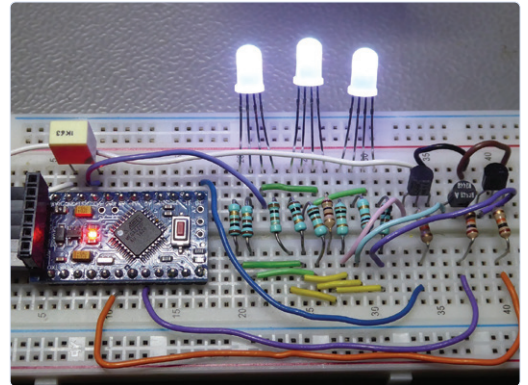


Figure 4. Pas trop critique, tout ça. Pour les expériences, une plaque d'expérimentation fera l'affaire.

tifs. La période est donc de $144 \times 100 \mu s = 14,4 \text{ ms}$. La durée d'éclair de chaque LED est fixée à 8, ce qui correspond à $8 \times 100 \mu s = 800 \mu s$. Le cycle d'utilisation (rapport impulsion/pause) est donc de $800 \mu s / 14,4 \text{ ms} = 5,5 \%$.

```
#define STROBE_PERIOD 144
```

```
TSoftPwm Pwm[] = {TSoftPwm(ID_RED, 0, 8,
    STROBE_PERIOD),
    TSoftPwm(ID_GREEN, 0, 8, STROBE_PERIOD),
    TSoftPwm(ID_BLUE, 0, 8, STROBE_PERIOD)};
```

Avec ce rapport cyclique, nous obtenons, une fois le régime moteur établi de manière optimale, une belle image stable.

Le circuit, y compris le logiciel téléchargeable sur la page de cet article [4], n'est pas un projet achevé. C'est pourquoi il n'y a pas (encore) d'interface confortable pour l'utilisateur. Cependant le logiciel montre bien les possibilités en jouant avec les paramètres : fréquence d'éclair, durée d'éclair et différence de phase. Le logiciel est commenté généreusement, inutile d'entrer dans les détails ici.

Un stroboscope non synchronisé n'est pas vraiment optimal. On pourrait monter une barrière lumineuse sur l'hélice, et en utiliser le signal pour déclencher le stroboscope. À vous l'initiative !

Mes deux vidéos sur Youtube permettent d'admirer le fonctionnement du stroboscope [1] et le rapport des signaux de commande des LED sur l'écran d'un oscilloscope [2] avec leur effet sur l'image stroboscopique perçue. Le projet peut également être consulté sur le site d'Elektor Labs [3].

191214-04



Produits

- > **livre en français avec kit : Tuto Arduino Uno**
www.elektor.fr/tuto-arduino-uno
- > **livre en français : Arduino – Faites-le jouer au train**
www.elektor.fr/arduino-faites-le-jouer-au-train
- > **e-book en français :**
36 Expériences de Physique avec Arduino
www.elektor.fr/36-experiences-de-physique-avec-arduino-e-book

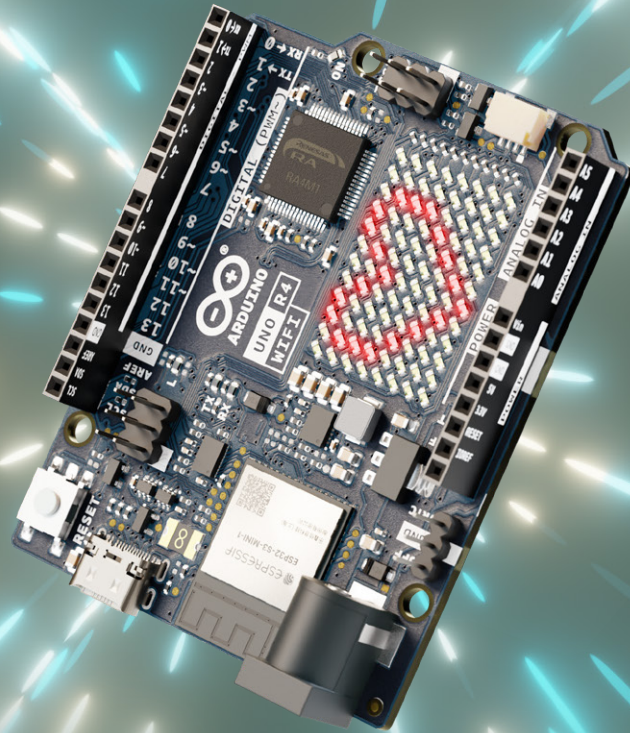
LIENS

- [1] Vidéo de démonstration 1 : <https://youtu.be/WHv6DCWM82k>
- [2] Vidéo de démonstration 2 : <https://youtu.be/3tYqUin4-vQ>
- [3] Projet sur Elektor Labs : www.elektormagazine.fr/labs/arduino-rgb-color-stroboscope
- [4] Page en ligne de cet article : www.elektormagazine.fr/191214-04



ARDUINO

UNO R4



La nouvelle dimension des Makers

Repoussez toutes vos limites grâce à un microcontrôleur convivial et accessible. Avec l'Arduino UNO R4, embarquez dans un univers en pleine expansion!



Mémoire plus rapide et plus grande - 48 MHz, 256 kB Flash, 32 kB RAM - parfait pour effectuer des calculs plus précis et gérer des projets plus complexes.



Nouveaux périphériques intégrés - Le RA4M1 (Arm® Cortex®-M4 Core avec FPU) comprend un DAC, un CAN-BUS, un support HID et un OpAMP incorporé permettant de réaliser des projets plus avancés.



Développez vos projets avec le shield UNO (compatible 5 V) et le vaste catalogue de nœuds *Qwiic compatibles.



Connectivité *Coprocesseur ESP32-S3 - connectivité Wi-Fi® et Bluetooth® Low Energy, laissant le microcontrôleur RA4M1 libre.



Robuste de par sa conception - avec une plus grande plage de tension, 6 - 24 V, vous pouvez maintenant utiliser l'UNO R4 en combinaison avec des moteurs, des bandes LED ou d'autres actionneurs à partir d'une seule source d'alimentation.

UNO R4 WiFi



UNO R4 Minima



*Note - ces fonctionnalités ne sont disponibles que sur le modèle UNO R4 WiFi.