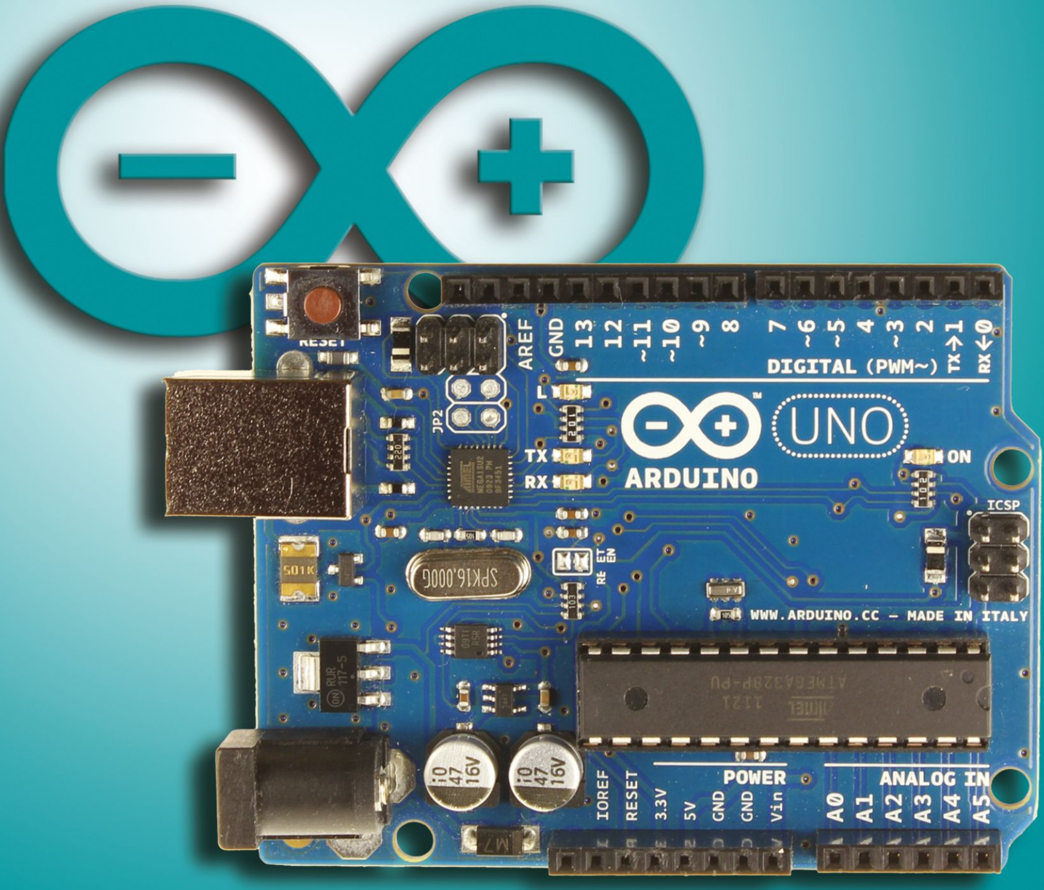
45 PROJETS ÉLECTRONIQUES ORIGINAUX

TUTO ARDUINO UNO



elektor

DÉCOUVRIR | CRÉER | PARTAGER

Bert van Dam

Traduction : Pascal Duchesnes

Tuto Arduino Uno

45 projets électroniques originaux

Téléchargement du code présenté dans ce livre :

www.elektor.fr/livres-arduino

L'éditeur remercie d'avance le lecteur qui prendra la peine de lui signaler les erreurs éventuelles qui auront échappé à sa vigilance (écrire à `webmaster@elektor.fr`).

1ère édition, 1er tirage

Mise en page : Mariline THIEBAUT-BRODIER
Coordination : Denis MEYER
Imprimé aux Pays-Bas par Wilco (04/2016)

ISBN : 978-2-86661-202-3

Sommaire

Votre premier programme

Pour notre premier test, nous allons faire clignoter la LED jaune de votre Arduino Uno. Pour cela, nous allons nous servir du programme de démonstration qui se trouve dans l'Arduino IDE sous : *Fichier, Exemples, 01.Basics, Blink* (en français « clignotant »).

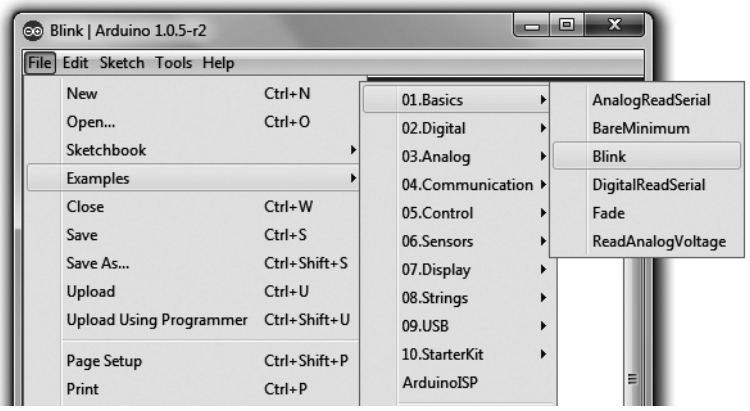


Figure 9 - Programme de démonstration Blink (clignotant)

Le programme est chargé automatiquement. Ne vous précipitez pas une fois que vous avez lancé Arduino IDE, car l'affichage de tous les programmes de la rubrique « Exemples » prend du temps. Si vous êtes allé trop vite, fermez le menu, attendez un instant puis réessayez ; la structure et le mode de fonctionnement du programme ne nous intéressent pas, nous y reviendrons dans le chapitre 4, page 23. Maintenant, le programme doit être traduit dans le langage Arduino avant d'être envoyé à la carte. Votre Arduino Uno étant relié à votre ordinateur, cliquez sur le bouton avec la flèche pointant vers la droite (voir numéro 1 dans la figure 11). Apparaît alors le message « Compilation du croquis en cours… » dans la barre de progression située au-dessous du programme.

Figure 10 - L'Arduino IDE cherche un autre port COM.

```
    b=t*5;
    c=3*b-a;
    Serial.write(27);
    Serial.print("[H");
    Serial.print("t= ");
    Serial.println(t);
    Serial.print("a= ");
    Serial.println(a);
    Serial.print("b= ");
    Serial.println(b);
    Serial.print("c= ");
    Serial.println(c);
    while(!Serial.available());
    dummy=Serial.read();
    t++;
}
```

À l'aide des données que nous avons dans la fenêtre du terminal, nous voyons ce qui ne colle pas avec les calculs. Un problème fréquent est que les calculs sont exécutés avec des valeurs qui ne conviennent pas au type de variable prévu à cet effet, c'est-à-dire lorsque « word » est utilisé alors qu'on aurait besoin de « int » ou lorsque les valeurs dépassent celles qui sont escomptées pour ce type de variable. Faites le calcul en vous servant d'une calculatrice pour dépister l'erreur.

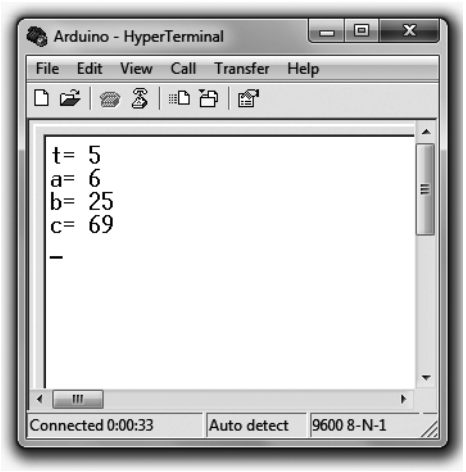


Figure 19 - Toutes les valeurs figurent les unes après les autres ; dans l'attente qu'on appuie sur *Entrée*.

Cette méthode permet également d'engendrer plusieurs colonnes à l'écran. Pour cela, on laisse le compteur « t » déterminer les colonnes. On voit bien que les différentes variables modifient leur valeur respective. Dans le prochain *sketch*, nous allons décaler

Nous utilisons le même matériel que dans le projet précédent. Vous pouvez tester le circuit en dirigeant la lumière d'une lampe de poche vers la photorésistance.

5.6 LED à luminosité réglable (MLI)

Je vous propose ici de régler la luminosité d'une LED en nous servant d'un potentiomètre. Nous allons utiliser pour cela la modulation de largeur d'impulsion (MLI ou *PWM* en anglais).

En principe, pour régler la luminosité d'une lampe ou la vitesse d'un moteur, il suffirait de faire varier sa tension d'alimentation. En pratique, ce n'est pas aussi simple et une commande par variation de tension ne fonctionne pas bien. Pour qu'ils commencent à tourner, la plupart des moteurs ont besoin d'une tension minimale. D'autre part, ils cessent de tourner en-dessous d'une certaine vitesse de rotation. Un deuxième inconvénient est leur faible couple, c'est-à-dire la force du moteur, à basse vitesse. Il en va de même pour les LED, qui elles aussi ne s'allument qu'au-dessus d'un seuil de tension minimale. En résumé, pour commander la luminosité d'une LED ou la vitesse d'un moteur, il ne suffit pas d'en faire varier la tension.

Une meilleure solution consiste à maintenir la tension constamment au maximum (dans notre cas, +5 V) et d'interrompre cette tension plus ou moins souvent ou longtemps. Plus la tension reste longtemps présente, plus le moteur tourne vite et plus la lumière de la LED est forte. Nous maintenons constante la fréquence d'interruption de la tension d'alimentation du moteur ou de la LED, mais nous modifions la durée (ou plutôt la largeur) des impulsions. Cette technique est appelée « modulation de largeur d'impulsion ».

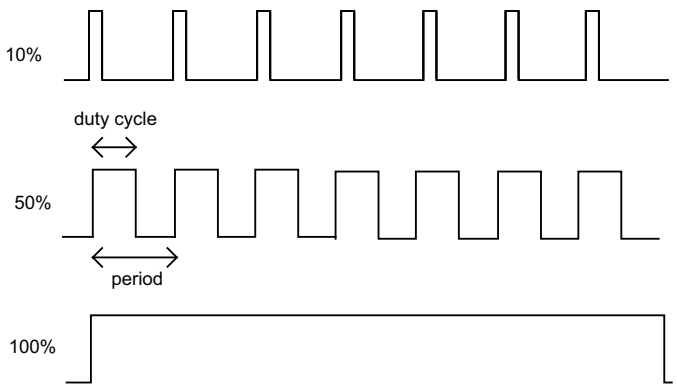


Figure 35 - Signal à MLI avec trois rapports cycliques différents

Figure 51 - Contacts et leur placement dans le pot de fleurs

Dès que le terreau est trop sec et donc que la valeur de mesure analogique est trop élevée, la LED s'allume pour signaler que la plante doit être arrosée. Les valeurs exactes dépendent naturellement de vos capteurs et de leur écartement. J'ai obtenu les résultats suivants :

humidité	ma mesure
insuffisante	env. 500
suffisante	env. 300
excessive	env. 200

Tableau 7 - Mesures de l'humidité

Vous pouvez dresser ce tableau en vous servant du *sketch* présenté au paragraphe 5.2, page 66. Suite à mes mesures, j'ai noté une valeur de seuil de 400 dans mon *sketch*.

```
int plant = A3;
int led=2;
int button = 8;
int threshold=400, value;

void setup() {
   pinMode(led, OUTPUT);
   pinMode(button, INPUT);
}

void loop(){
   value = analogRead(plant);
   // la configuration de la LED est requise
   if (value>threshold+20){
      digitalWrite(led, HIGH);
```

Suite à cela, mon microcontrôleur s'est mis en grève et mon robot refusait de bouger.

En me servant d'un émulateur d'oscilloscope, j'ai constaté que ces capteurs provoquaient des perturbations dans le bloc d'alimentation. J'ai résolu le problème en me servant d'un condensateur de 100 µF connecté aux broches d'alimentation de chaque capteur. Ici, vous n'avez pas besoin d'un tel condensateur. Ce dernier n'est requis que lorsqu'on utilise des fils de connexion longs dans un environnement perturbateur (moteurs).

6.7 Capteur à ultrasons (télémètre)

Dans ce projet, nous connectons le capteur à ultrasons (en anglais *Ultrasonic Range Finder*) SRF05 à l'Arduino pour mesurer la distance qui le sépare des objets qui l'entourent.

Figure 72 - Capteur à ultrasons, modèle *Ultrasonic Range Finder SRF05*

Le SRF05 de Daventech est un capteur à ultrasons économique qui permet de mesurer la distance qui le sépare d'un objet qui émet des ondes acoustiques.

La figure 73 illustre que le cycle de détection de l'Arduino requiert une impulsion d'une longueur d'au moins 10 µs sur l'entrée impulsion de déclenchement (en anglais *pulse trigger*) pour pouvoir démarrer. Dès que cette borne repasse à l'état bas, le module envoie une série de huit impulsions ultrasoniques d'une fréquence de 40 kHz. Pour parer à une rétroaction entre l'émetteur et le récepteur, le module attend un instant et met la sortie impulsion d'écho (en anglais *echo pulse*) à l'état haut (HIGH). C'est le

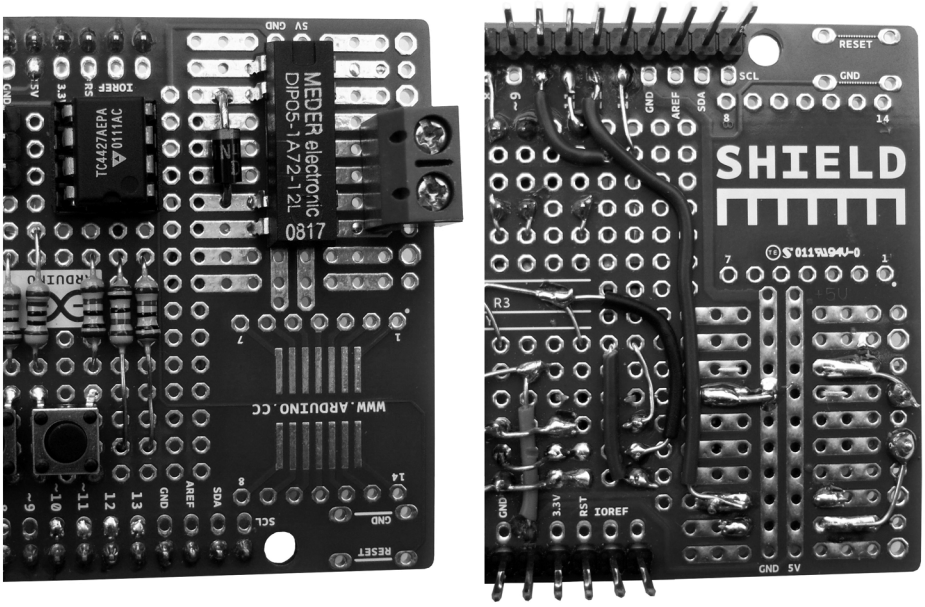


Figure 88 - Relais et diode sur le *shield*

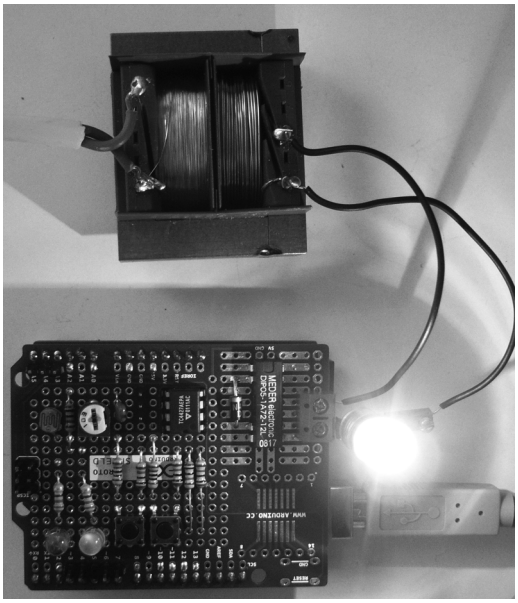


Figure 89 - Lampe de 10 V clignotante

change d'état (elle s'allume si elle était éteinte et vice versa). Et on attend une nouvelle fois 100 ms. Si la deuxième LED s'allume, c'est que le temps d'allumage est faussé ou qu'il ne correspond plus à ce qui avait été prévu. Si cela vous dérange, vous pouvez insérer après le *delay*, l'instruction *t=t+100*. Le compteur fait alors un bond en avant de 100 ms.

Et voilà comment nous allons nous en servir :

1. Tapotez l'élément piézoélectrique pour activer la LED connectée à la broche 2.
2. Si la LED s'allume, tapotez l'élément piézoélectrique pour allumer et éteindre la LED connectée à la broche 5.

Il serait facile de réaliser une gâche électrique à combinaison secrète en incorporant un compteur de tocs dans la boucle de comptage et en commutant un relais lorsque le compteur a atteint la valeur de consigne.

7.6 Casse-pieds

Sur l'internet j'ai découvert un montage proposé par le bien nommé Zach le terrible.[8] En effet, son projet a pour objet de déranger les gens, la nuit, durant leur sommeil.

Il s'agit d'un petit circuit que l'on cache dans la chambre de sa victime. Dès qu'il fait noir, le circuit se met en route et produit un bip irritant. Ce bip s'arrête dès que la victime allume la lumière. La personne croit avoir rêvé, elle éteint donc la lumière pour se recoucher. Mais le cirque recommence.

Dès qu'il est sous tension, le circuit mesure la luminosité. Si celle-ci se trouve au-dessous d'une certaine valeur de seuil (c'est-à-dire dès qu'il fait sombre), le bip se fait entendre. Dès que la luminosité dépasse la valeur de seuil, le circuit reste silencieux, il attend entre 30 et 90 s, puis il vérifie si la lumière a été éteinte. Vous pouvez bien sûr allonger le temps d'attente, mais pas trop longtemps pour que la victime ne se rendorme pas.

```
int speaker = A1;
int ldr = A2;
int pot = A0;
int ldrm,potm;

void setup(){
   randomSeed(analogRead(5));
   Serial.begin(9600);
}
```

8. Ce montage est fait de manière traditionnelle sans Arduino et sans microcontrôleur.

!

Numérique

A

B

C

D

E

F

G

I

L

M

N

O

P

Q

R

S

T

V

W

X