La PWM : qu'est-ce ?

(A partir d'un article publié sur Locaduino.org)

Jusqu'à présent, nous avons vu l'émission de signaux numériques par le biais des broches numériques et de `digitalWrite(...)` et l'allumage et l'extinction d'une Led par exemple. Ces signaux numériques ont soit une valeur égale à LOW, c'est à dire 0V, soit une valeur égale à HIGH, ce qui correspond à la tension d'alimentation de l'Arduino, VDD, soit 5V pour les Arduino à base de micro-contrôleurs AVR ou bien 3,3V pour les Arduino à base de micro-contrôleurs ARM.

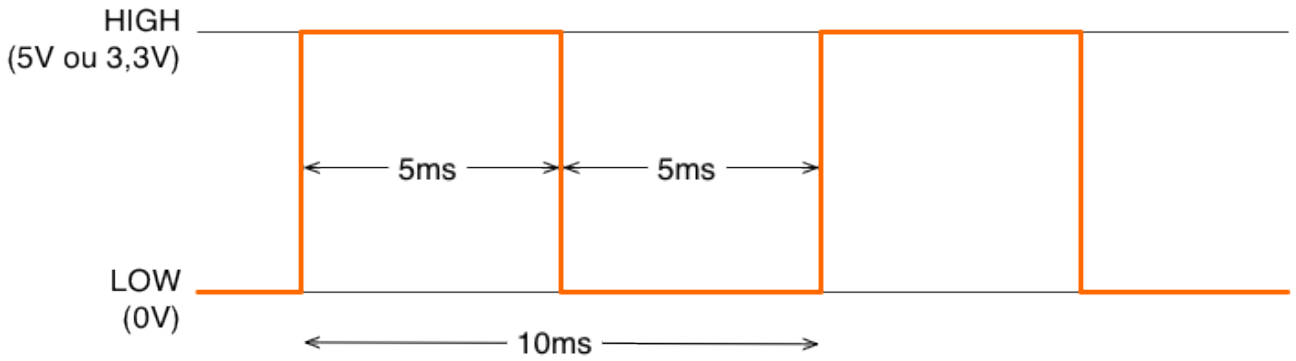
Mais les sorties numériques ne nous permettent pas, par exemple, de régler la luminosité de la Led, ou de faire varier la vitesse de rotation d'un moteur. Pour pouvoir faire cela, il serait nécessaire de pouvoir disposer de sorties permettant des tensions intermédiaires entre LOW et HIGH.

Mais ceci n'existe pas sur les Arduino. L'alternative est d'utiliser une **PWM**, pour Pulse Width Modulation, ou **MLI** en français pour Modulation de largeur d'impulsion.

Qu'est ce que la PWM ?

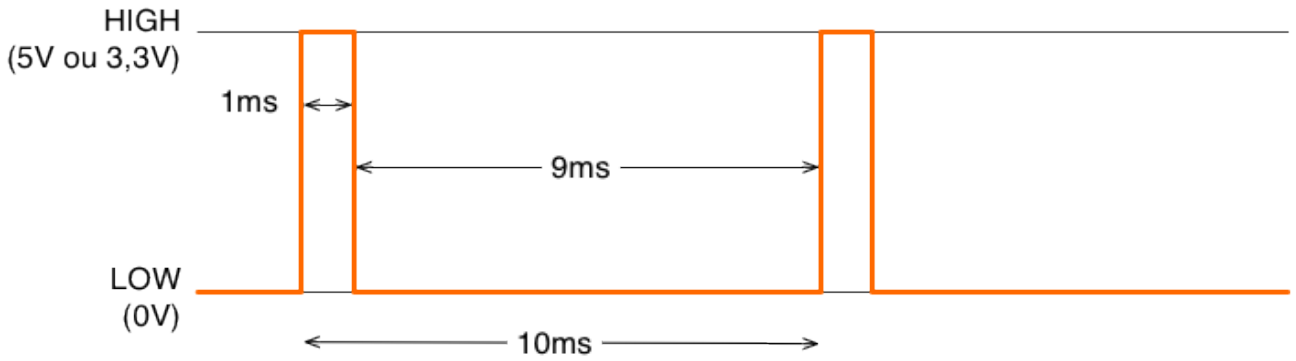
On reste en numérique, les signaux ont toujours une valeur LOW ou HIGH et le principe est de construire un signal qui est alternativement LOW et HIGH et de répéter très vite cette alternance. La Led est donc alternativement allumée et éteinte mais le cycle est tellement rapide que la persistance rétinienne nous donne l'illusion d'une Led allumée en permanence.

Prenons par exemple une période de 10ms, soit une fréquence de 100Hz. Si la Led est allumée pendant 5ms et éteinte pendant 5ms, comme sur la figure ci-dessous, l'impression sera une luminosité de 50% de la luminosité maximum.



PMW à 50%
La fréquence est de 100Hz, le rapport cyclique de 50%

Si la Led est allumée pendant 1ms et éteinte pendant 9ms, l'impression sera une luminosité de 10% comme sur la figure ci-dessous.



PWM à 10%
La fréquence est de 100Hz et le rapport cyclique de 10%.

Le pourcentage de temps passé à l'état HIGH sur la période du signal est appelé le *rapport cyclique*. Il varie donc de 0%, le signal est tout le temps LOW, à 100%, le signal est tout le temps HIGH.

Les capacités PWM des Arduino

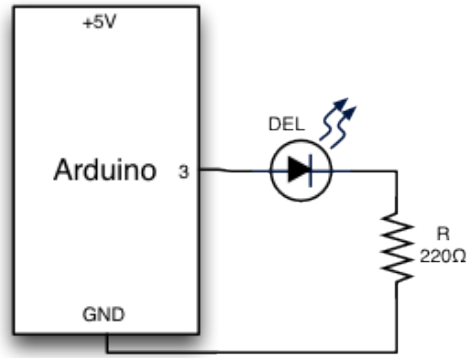
Les micro-contrôleurs proposent donc des dispositifs permettant de disposer de PWM autonomes où le signal voulu est engendré indépendamment de l'exécution du programme. Toutes les broches ne sont pas utilisables. Sur l'Arduino Uno, les broches concernée portent le symbole ' '. Sur le Uno et les Arduino à base d'ATMega 328, 6 broches sont disponibles, sur le Mega, 15 broches sont disponibles. Le tableau ci-dessous résume la disponibilité des PWM sur les cartes Arduino.

Modèle d'Arduino	Broches PWM
Uno, Pro Mini, Nano	3, 5, 6, 9, 10 et 11
Mega	2 à 13, 44 à 46
Leonardo	3, 5, 6, 9, 10, 11 et 13
Due	2 à 13
Zero	2 à 13

La fréquence de la PWM est prédéterminée sur l'Arduino. Il est possible de la changer comme nous le verrons dans un futur article mais ce n'est pas une possibilité accessible très simplement. La fréquence n'est pas la même selon les broches. Sur le Uno, la fréquence est de **490Hz** sur les broches **3, 9, 10 et 11** et de **980Hz** sur les broches **5 et 6**.

Exemple de mise en œuvre

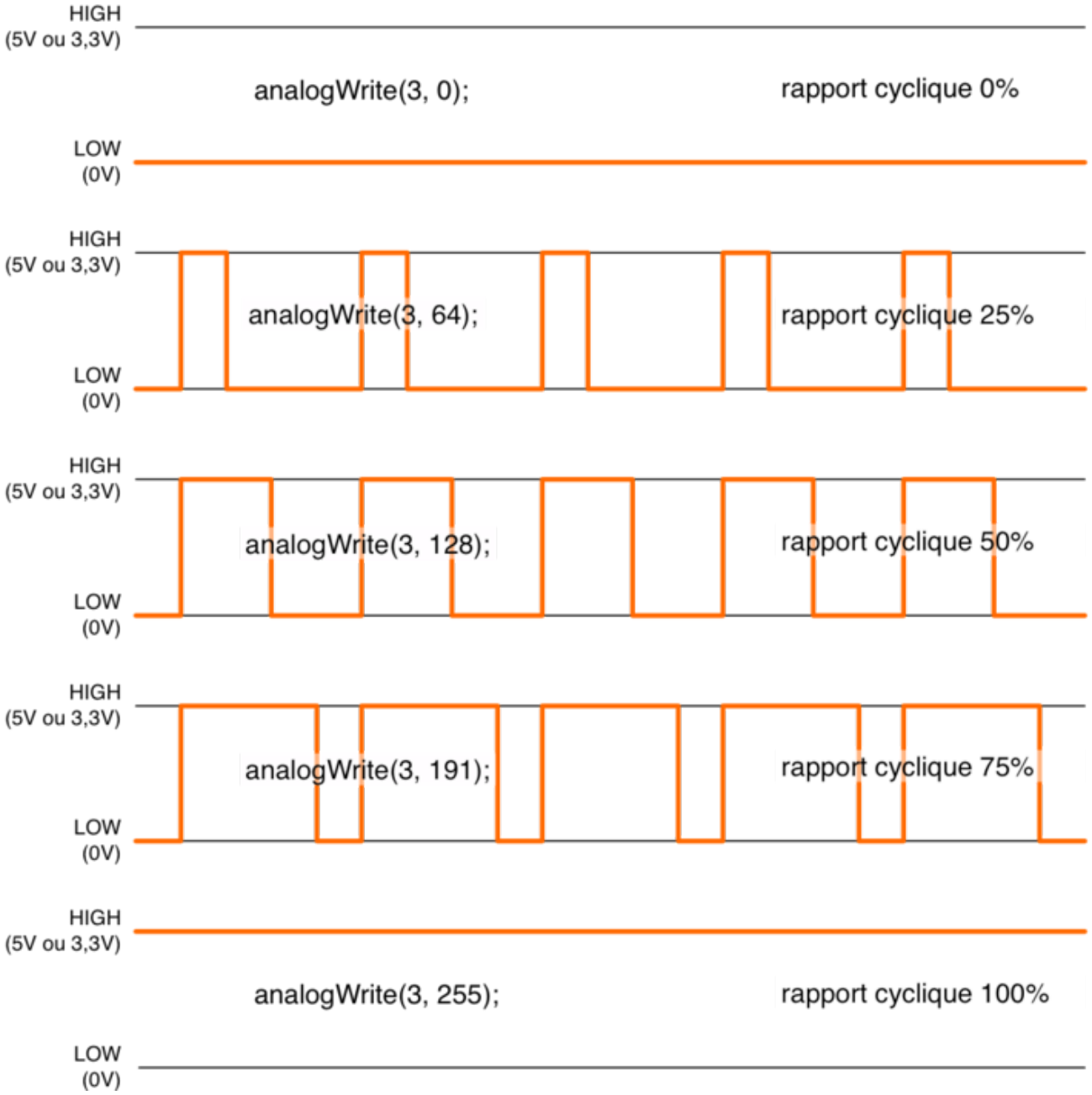
Commande d'une Led par PWM.



En ce qui concerne la programmation, la fonction permettant de fixer le rapport cyclique de la PWM est la fonction `analogWrite(...)`. Nom bien mal choisi puisque, comme on l'a vu, tout est numérique. Le premier argument de `analogWrite(...)` est la broche concernée et le second argument le rapport cyclique. Le rapport cyclique **n'est pas** donné de **0 à 100** mais de **0 à 255**. Il faut donc faire une règle de 3 pour calculer la valeur à appliquer pour le rapport cyclique voulu. Par exemple, la valeur à appliquer pour un rapport cyclique de 75% sera égal à 0,75 x 255 = 191.

```
analogWrite(3, 191); // éclaire notre DEL à 75%
```

La figure ci-dessous montre différents réglages de PWM.



Voici pour commencer un programme très simple qui augmente progressivement, de 1 en 1, la luminosité de la DEL de 0, extinction totale, à 255, luminosité maximum, puis recommence à 0. On va fixer le temps de ce cycle à 1000ms. Le délai entre chaque augmentation est donc de 1000 / 255 = 3,922 ms. Arrondissons à 4ms. En utilisant un `byte` pour stocker la valeur, nous pouvons profiter du débordement qui se produit

lorsque, étant à 255, l'ajout de 1 fait repasser la valeur 0. Vous noterez également qu'il n'est pas nécessaire de programmer la broche en sortie pour son utilisation en tant que sortie PWM. Cette programmation s'effectue automatiquement lors du premier `analogWrite()`.

```
const byte pinDEL = 3;
byte luminosite = 0;

void setup()
{
}

void loop()
{
   analogWrite(pinDEL, luminosite);
   luminosite++;
   delay(4);
}
```