

Sorties numériques :

amplification de puissance, « interrupteur optique », protection opto-couplée avec Arduino.



Ateliers Arduino

par X. HINAULT

www.mon-club-elec.fr



Tous droits réservés – 2012.

Ce document légèrement payant est soumis au droit d'auteur et est réservé à l'usage personnel.

Afin d'encourager la production de supports didactiques de qualité, ce document est légèrement payant.

La licence d'utilisation est attribuée pour un usage personnel uniquement, dans le cercle familial. Mise en ligne et diffusion non autorisées.

Si vous n'êtes pas le détenteur de la licence attribuée pour l'usage de ce document, soyez sympa, merci d'acheter votre exemplaire personnel ici : <https://monclubelec.dpdcart.com/>

Pour tout problème lié à l'utilisation de ce document, veuillez envoyer une copie ici : support@mon-club-elec.fr

Pour obtenir tout autres types de licence d'utilisation (enseignement, commercial, etc...), veuillez contacter l'auteur ici : support@mon-club-elec.fr

Vous avez constaté une erreur ? une coquille ? N'hésitez pas à nous le signaler à cette adresse : support@mon-club-elec.fr

Truc d'utilisation : visualiser ce document en mode diaporama dans le visionneur PDF. Navigation avec les flèches HAUT / BAS ou la souris.

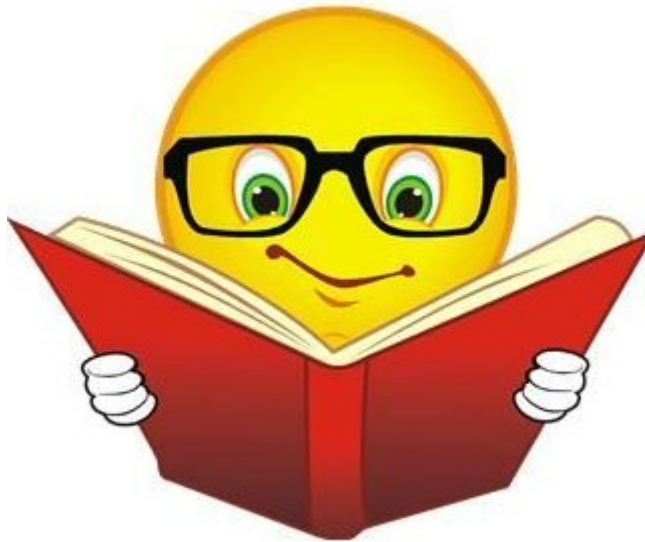
En mode fenêtre, activer le panneau latéral vous facilitera la navigation dans le document. Bonne lecture !

Lancer également le logiciel Arduino et connecter votre carte Arduino afin de pouvoir tester au fur et à mesure les codes d'exemples !

1. Intro

L'objectif ici est d'apprendre des choses importantes à connaître en ce qui concerne les broches numériques en sortie:

- de rappeler les caractéristiques électriques de la carte Arduino
- de poser la problématique de « l'amplification de puissance » ON/OFF avec Arduino
- de présenter les solutions simples existantes et des exemples d'utilisation
- de présenter la protection opto-couplée et son principe d'utilisation



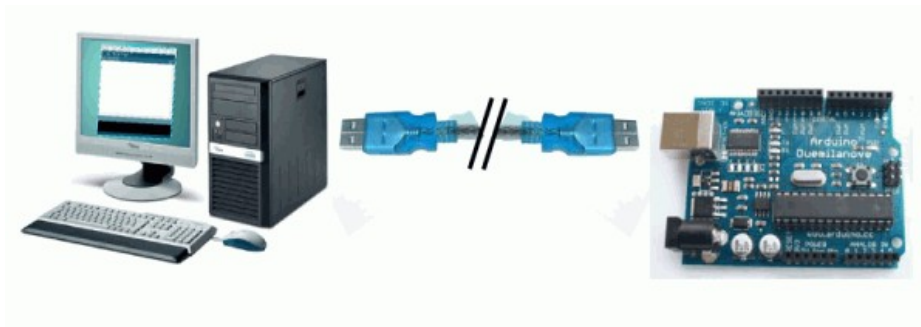
Remarque :

Les informations contenues dans cet atelier sont essentiellement de nature technique, même si quelques codes d'exemple sont bien sûr fournis.
N'en soyez pas surpris et prenez le temps de bien comprendre plusieurs notions importantes présentées ici.

2. Matériel nécessaire pour les ateliers Arduino

Pour cet atelier, vous aurez besoin de tout ou partie des éléments suivants pour pouvoir réaliser les exemples proposés :

De l'espace de développement Arduino

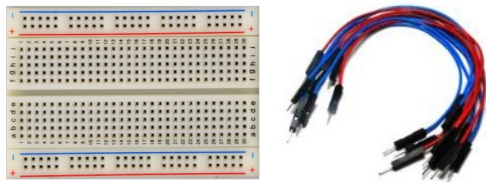


L'espace de développement Arduino associe :

- un ordinateur sous Windows, Mac Os X ou Gnu/Linux (Ubuntu)
- avec le logiciel Arduino installé (voir : <http://www.arduino.cc/>)
- un câble USB
- une carte Arduino UNO ou équivalente.

disponible chez : <http://shop.snootlab.com/> ou <http://www.gotronic.fr/>

Du nécessaire pour réaliser des montages sans soudure

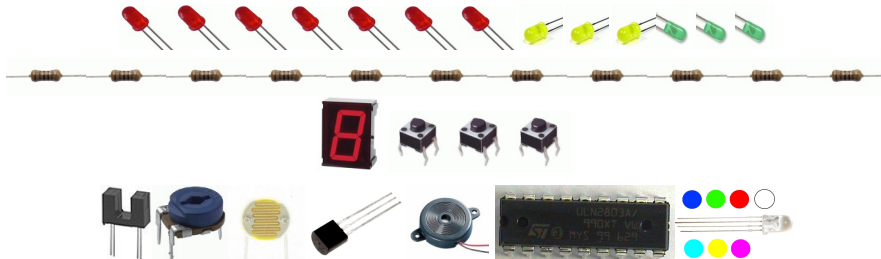


Pour réaliser des montages sans soudure, vous aurez besoin :

- d'une plaque d'essai ou breadboard moyenne (450 points)
- de quelques câbles souples (ou jumpers) mâle/mâle

disponible chez : <http://www.gotronic.fr/>

De quelques composants de base



Pour vous simplifier la vie, nous avons négocié ce kit pour vous !

Vous pouvez commander ce kit complet directement en 1 clic chez notre partenaire

<http://www.gotronic.fr/> avec le code express **701710**

GO TRONIC
ROBOTIQUE ET COMPOSANTS ÉLECTRONIQUES

Pour plus de détails, voir : http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ATELIERS

Pour les ateliers Arduino niveau débutant, vous devrez idéalement disposer des composants suivants :

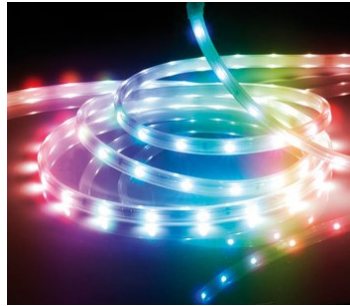
- des LEDs 5mm Rouges(x20), Vertes (x5) et 3 Jaunes (x5)
- digit à cathode commune rouge 13mm (x1)
- Résistances (1/4w - 5%) de 270 Ohms (x20), 4,7K Ohms (x1), 1K Ohms (x1)
- mini bouton-poussoir (x3)
- Opto-fourche (x 1)
- Résistance variable linéaire 10K (x 1)
- Photo-résistance 7mm (x 1)
- Capteur de température LM35DZ (-55/+150°C - 10mV/°C) (x 1)
- Capsule son piézoélectrique (x 1)
- ULN 2803A (CI amplificateur 8 voies, 500mA/ voie) (x 1)
- LED 5mm multicolore RVB cathode commune (x 1)

3. *Matériel spécifique nécessaire pour cet atelier*

Pour cet atelier vous pouvez avoir besoin également, selon les exemples que vous mettrez en oeuvre :

D'un ruban à LEDs RVB 12V

Disponible en magasin de bricolage ou chez fournisseur électronique. 25€ le mètre environ.



D'un relais 1 RT (bobine 12V DC)

En magasin de bricolage ou chez fournisseur électronique. Prendre modèle Finder type 41 ou équivalent. 4€ environ.



D'un bloc secteur 12V/2A



D'un moteur CC standard (6-12V)



4. Notion d'électricité élémentaire à bien connaître

Pour faire de l'électronique numérique avec Arduino, il n'y a pas beaucoup de règles d'électricité à connaître, mais il y en a une qui est très importante à connaître (elle vous servira tout le temps !!) :

TOUT « DISPOSITIF » ELECTRIQUE SE CARACTERISE PAR 2 CHOSES :

UNE TENSION D'ALIMENTATION mesurée en VOLTS

UNE INTENSITE DE FONCTIONNEMENT mesurée en AMPERES

La règle qui lui est associée et qui est tout aussi importante :

TOUTE « ALIMENTATION » ELECTRIQUE SE CARACTERISE PAR 2 CHOSES :

UNE TENSION FOURNIE mesurée en VOLTS

UNE INTENSITE MAXIMALE disponible mesurée en AMPERES

Ainsi, avant de connecter un dispositif sur une alimentation, il va falloir systématiquement se poser 2 questions :

- la tension fournie par l'alimentation correspond-elle à la tension d'alimentation du dispositif ?
- l'intensité de fonctionnement du dispositif est-elle bien inférieure à l'intensité maximale possible ?

Il faudra toujours répondre OUI à ces 2 questions. Exemples :

- puis-je connecter un appareil fonctionnant en 12V et consommant 1A sur une prise de courant de 220V / 10A ? => **NON (danger!)**
- puis-je connecter un moteur fonctionnant en 6V et consommant 2A sur un bloc secteur 6V/500mA ? => **NON**
- puis-je connecter une lampe consommant 5V/20mA sur une broche pouvant fournir 5V/40mA ? => **OUI**

Exemple de fiche technique d'un moteur.



Moteur: RE280/1
Alimentation: 12 à 24 Vcc
Consommation à vide: 0,10 A à 12 Vcc
Consommation en charge: 0,30 A à 12 Vcc

Exemple de fiche technique d'un bloc secteur

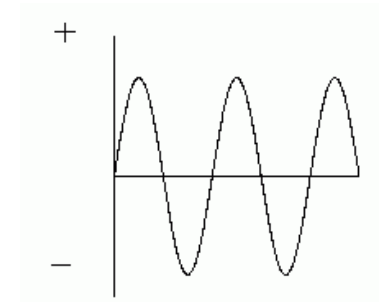


Tension d'entrée: 230 Vac - 50 Hz
Tension de sortie régulée: 12 Vcc
Courant maxi : 1000 mA

5. Notion de tension alternative, continue, régulée..

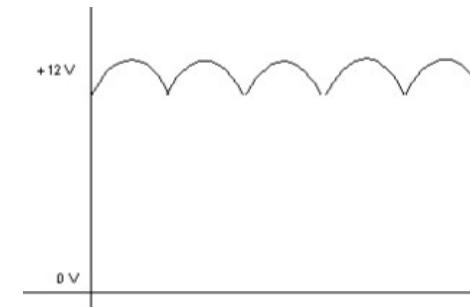
Tension alternative = celle du secteur 220V

La tension du secteur est une tension qui varie tout le temps en faisant des « vagues » : la tension est dite sinusoïdale ou alternative.



Tension continue non régulée = celle d'un bloc secteur 12V

La plupart des dispositifs « basse tension » tels que les moteurs et autres petits appareils alimentés par une alimentation externe (bloc secteur) utilise une alimentation continue mais qui n'est pas parfaitement « plane » avec quelques variations possibles autour de la tension théorique prévue.



Tension continue régulée = celle des circuits numériques

Les circuits numériques tels qu'une carte Arduino, ou la carte-mère (interne) d'un ordinateur, utilisent et nécessitent des tensions continues **PARFAITEMENT STABLES** qui sont obtenues à l'aide d'un composant appelé régulateur : ce sont des **alimentations dites régulées**.

Les tensions régulées les plus utilisées en électronique numérique sont le **5V**, le **12V** et le 3.3V. La carte Arduino standard fonctionne en 5V régulé.



La tension V_{in} d'une carte Arduino pourra être de type continu régulé ou non, la carte Arduino disposant d'un régulateur 5V intégré.

6. Caractéristiques électriques globales de la carte Arduino

Les deux façons d'alimenter la carte Arduino :

La carte Arduino standard peut être alimentée de 2 façons :

- soit directement par le port USB (qui fournit 5V/500mA régulé)
- soit par une alimentation continue externe via le connecteur d'alimentation (cette source s'appelle Vin) :
 - tension continue, régulée ou non, entre 7 et 12V conseillé (jusqu'à 20V maxi) (*La carte Arduino dispose d'un régulateur 5V intégré*)
 - pouvant fournir au moins 500mA, ou plus...

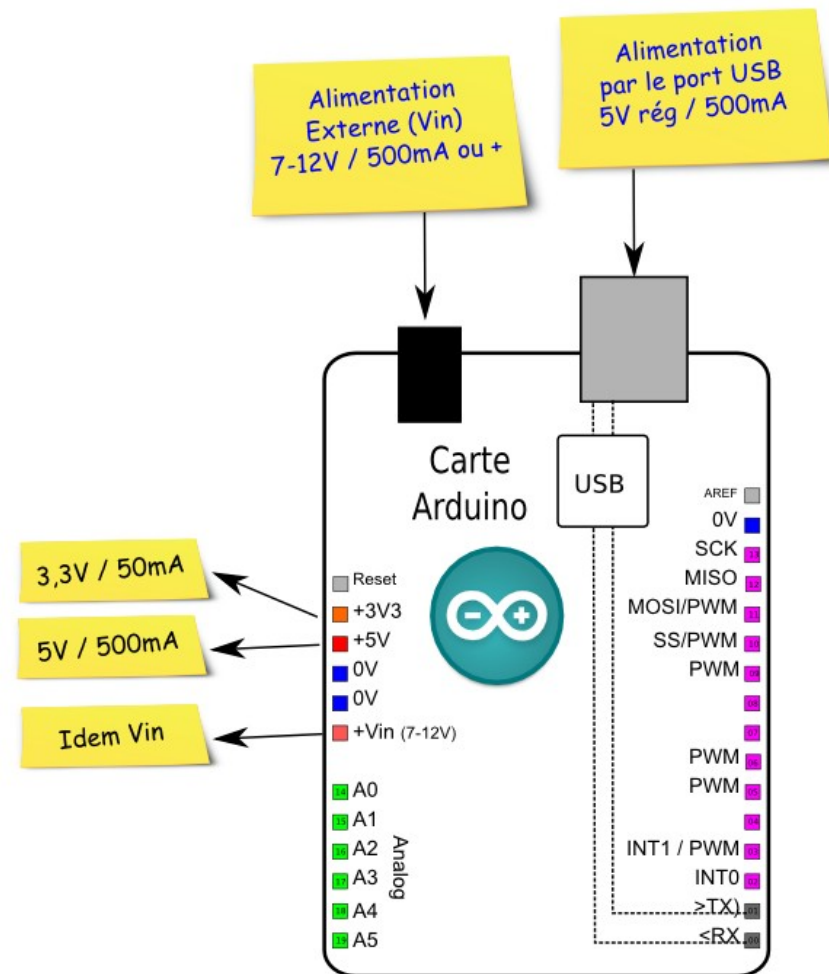
Le connecteur d'alimentation

La carte Arduino dispose de d'un connecteur d'alimentation qui fournit plusieurs tensions :

- **Vin** qui est une reprise de la tension Vin connectée au connecteur d'alimentation externe et a les mêmes caractéristiques que l'alimentation utilisée. A noter que Vin est régulé si l'alimentation utilisée est régulée (Alimentation de PC)
- **5V** qui est du 5V régulé jusqu'à 500mA maximum, utilisable pour alimenter les capteurs, modules utilisés avec la carte Arduino,
- **3V3** régulé jusqu'à 50mA (attention faible intensité max) et utilisable pour alimenter certains composants nécessitant du 3V3. Peu utilisé en pratique... mais il est là si besoin...

La tension Vin d'une carte Arduino pourra être de type continu régulé ou non, la carte Arduino disposant d'un régulateur 5V intégré.

En pratique, pour alimenter votre carte Arduino avec une alimentation externe, utiliser un bloc 220V => 6-12V cc /500mA ou mieux 6-12V cc / 1A.



7. Caractéristiques électriques d'une broche Numérique Arduino en sortie

Une broche numérique Arduino individuelle en sortie peut-être considérée **comme une « mini »-alimentation** :

- fournissant une tension de **5V régulé** au niveau HAUT et 0V au niveau BAS.
- pouvant fournir **au maximum 40mA** pour une seule broche en sortie.

Pour l'ensemble des broches numériques :

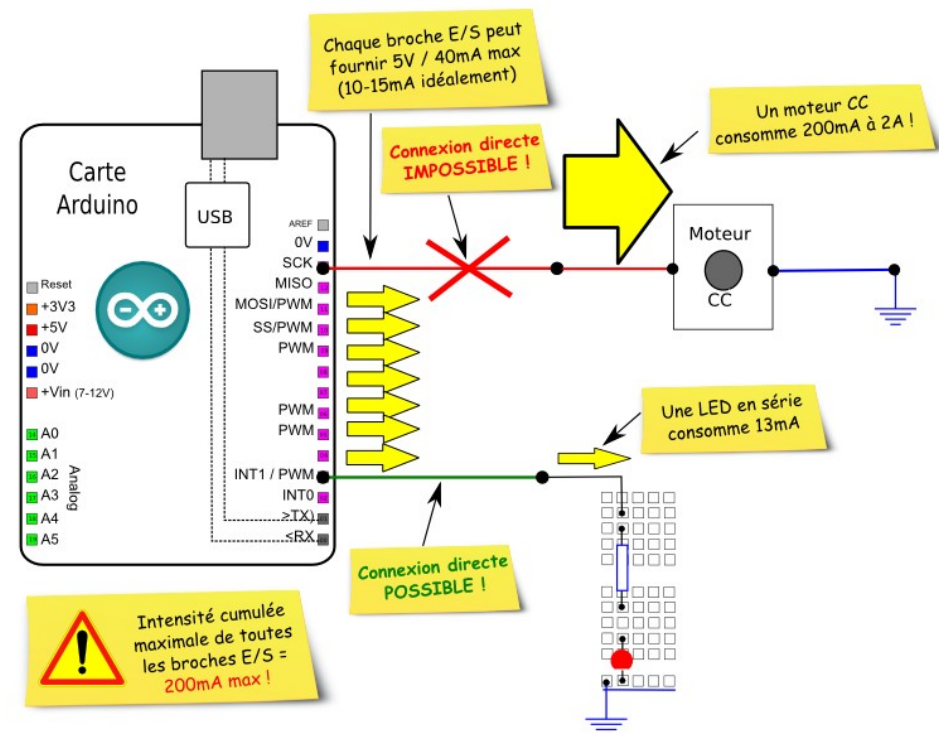
- **l'intensité maximale cumulée ne doit pas dépasser les 200mA.**
- Ainsi, pour éviter les soucis, dans l'hypothèse où l'on utilisera au maximum une 15aine de broches en sortie simultanément, considérer que l'on peut utiliser sans danger $200\text{mA}/15 = 13\text{mA}$ / broche soit en pratique 10 à 15mA par broche.

En pratique : considérer qu'une broche Arduino fournit 5V / 10-15mA

Voici quelques exemples, pour se faire une idée de ce que l'on peut connecter directement sur une broche numérique d'une carte Arduino en sortie :

- une broche d'un autre CI numérique consommera dans les 1 mA, => **connexion directe POSSIBLE !**
- une LED en série avec sa résistance consommera dans les 10-20mA selon la résistance utilisée, => **connexion directe POSSIBLE !**
- la broche de commande d'un servomoteur consommera dans les 5mA, => **connexion directe POSSIBLE !**
- un moteur CC consommera dans les 250mA => **connexion directe IMPOSSIBLE !** (Dans ce cas, on devra utiliser une interface de puissance comme nous le verrons)

En pratique : TOUJOURS se demander « quelle intensité va être utilisée ? » par l'élément que l'on va connecter sur la broche.



8. Notion d'amplification de puissance et d'interface de puissance

Le problème...

A présent, vous allez comprendre facilement le concept d'amplification de puissance. C'est amusant d'allumer des LEDs, mais on va rapidement avoir envie de commander avec une carte Arduino plein d'autres choses et notamment des dispositifs de la vie réelle...

Si on utilise la carte Arduino telle quelle, les possibilités sont assez limitées : on ne pourra allumer/contrôler que des dispositifs :

- fonctionnant en 5V (1V = 1 Volt = 1000 millivolts)
- ne nécessitant pas plus de 40 mA (1 mA = 1 milliampère = 0,001 Ampère)

Or de nombreux dispositifs ont caractéristiques bien différentes :

- un moteur à courant continu nécessitera 6 à 12V et 200mA à 2A
- un relais fonctionnera en 12V et nécessitera 100mA par exemple
- un ruban à LED fonctionnera en 12V et jusqu'à plusieurs A
- etc...

Pour prendre une image, c'est comme si on voulait brancher une lance à incendie sur un robinet de cuisine !

La solution...

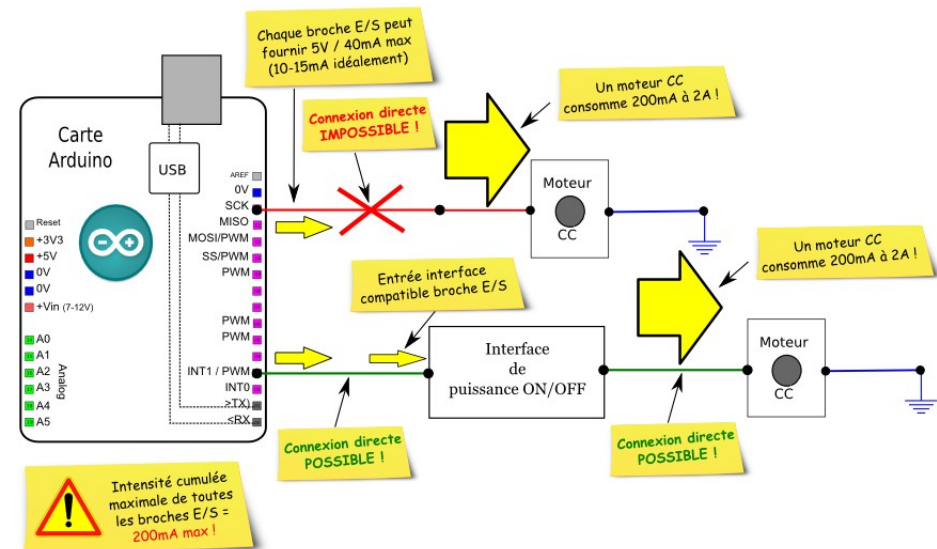
La solution passe par une « interface de puissance » : c'est un circuit électronique ou une carte électronique selon les cas, qui va assurer un double rôle :

- adapter la tension
- adapter l'intensité

De cette façon, à partir de la sortie numérique 5V/10mA, on va pouvoir contrôler des appareils :

- entre 5V et 12V voire 30V (adaptation en tension)
- consommant 100mA, 300mA voire plusieurs A (adaptation en intensité)

On pourra de cette façon potentiellement contrôler toutes sortes de dispositifs à partir d'une simple carte Arduino !

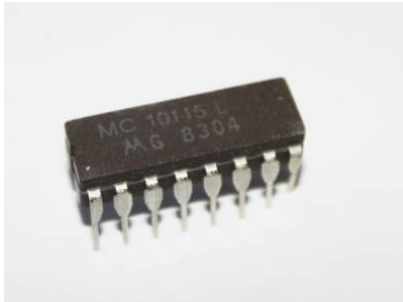


9. Info technique : les circuits intégrés en boîtier DIL

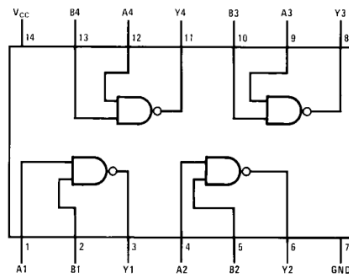
Présentation des circuits intégrés DIL...

En fait, il s'agit d'un format de boîtier de circuit intégré courant :

- petit boîtier noir en plastique rigide
- présentant toute une série de broches métalliques latérales
- avec un numéro dessus correspondant à la référence du circuit



Un circuit intégré est un circuit électronique miniaturisé et ayant une fonction particulière fixe : un compteur, une bascule JK, portes logiques, etc...



Exemple de schéma fonctionnel – 74LS00 – 4 portes NAND

Un circuit intégré dispose de plusieurs broches :

- Chaque broche du circuit intégré DIL va avoir une fonction précise qui est décrite dans une fiche technique appelée « datasheet » en anglais.
- Dans le cas d'un circuit intégré numérique, les broches métalliques seront des entrées ou sorties numériques.
- Un circuit intégré type dispose également, comme tout dispositif électrique de 2 broches d'alimentation : le 0V et le +5V classiquement.

Les circuits intégrés numériques existent en plusieurs familles :

- circuits analogiques, numériques, mixtes,
- chaque famille est dénommée avec des lettres et des nombres
- l'une d'entre elle est courante : les 74LSxx où xx est un nombre.
- il en existent pleins d'autres, les lettres utilisées étant souvent associées à un fabricant et le nombre à une fonction

74LS00	4 portes NAND à 2 entrées	DIL 14
74LS02	4 portes NOR à 2 entrées	DIL 14
74LS03	4 NAND à 2 entrées col. ouv.	DIL 14
74LS04	Sextuple inverseur	DIL 14
SN7406	Sextuple inverseur buffer	DIL 14
74LS06	Sextuple inverseur buffer	DIL 14
74LS07	Sextuple buffer	DIL 14
74LS08	4 portes AND à 2 entrées	DIL 14
74LS09	4 AND à 2 entrées col. ouv.	DIL 14
74LS09D-CMS	4 AND à 2 entrées col. ouv.	SO14
74LS12	3 NAND à 3 entrées col. ouv.	DIL 14
74LS14	Sextuple inverseur trigger	DIL 14
74LS15	3 AND à 3 entrées col. ouv.	DIL 14

Exemple de nom de circuit et leur fonction



Position type des broches 0V et +5V d'un CI logique.

Noter le sens du circuit indiqué par une marque sur le boîtier.

Avec Arduino, on aura très peu besoin de circuits intégrés externes, la plupart des fonctions pouvant être réalisées par programmation. Un programme élaboré peut être l'équivalent de plusieurs dizaines de circuits intégrés associés !

10. Exemple de CI d'amplification de puissance ON/OFF : le ULN 2803A

Fonction

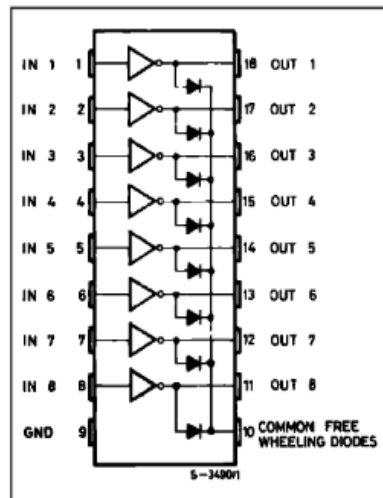
- Ce circuit très répandu et bon marché (<2 Euros) est au format DIL 18 est constitué de 8 étages d'amplification ON/OFF inverseurs
- chaque étage :
 - peut-être commandé par une entrée numérique
 - et peut fournir 500mA en sortie, sous une tension pouvant aller jusqu'à 50V.



Brochage

Le brochage de ce CI est très simple et très pratique :

- 2 broches d'alimentation : 0V (broche 9) et V+ (broche 10).
- 8 broches d'entrées de commande numériques (1 à 8) face chacune à une broche de sortie de puissance (10 à 18)



Principe de fonctionnement

Le fonctionnement est très simple. Pour chaque étage, on a :

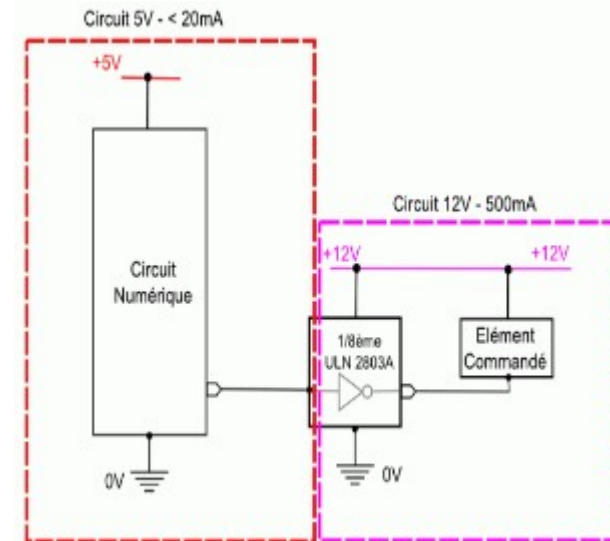
entrée numérique	Tension Sortie	Effet
HAUT	0V	MARCHE
BAS	V+(+5V à +50V !)	ARRET

Ainsi, la sortie de puissance est le reflet inversé de l'entrée de commande et l'élément électrique commandé est en MARCHE sur le niveau HAUT.

Circuit de fonctionnement type

Les éléments électriques à commander doivent être connectés entre le V+ et une broche de sortie :

- c'est le niveau HAUT, et uniquement le niveau HAUT, qui peut commander la mise sous tension de l'élément commandé.
- Ainsi, il n'est pas possible de commander l'élément électrique par un niveau BAS, en connectant l'élément entre la sortie et le 0V. Ceci limite le fonctionnement de l'ULN2803 à un fonctionnement ON/OFF simple.



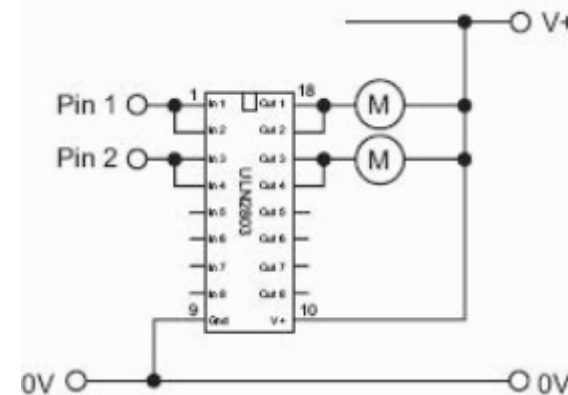
11. Exemple de CI d'amplification de puissance ON/OFF : le CI 2803A (suite)

Plusieurs avantages :

- amplification ON/OFF sur 8 voies !
- entrée numérique en face de la sortie amplifiée !
- petite taille (simple boîtier DIL)
- pas cher et répandu
- [adaptation en tension](#) de 6V à 30V
- [adaptation en intensité](#) jusqu'à 500mA/voie
- possibilité de coupler les voies 2 à 2
- rapide = laisse passer impulsions

Truc d'utilisation

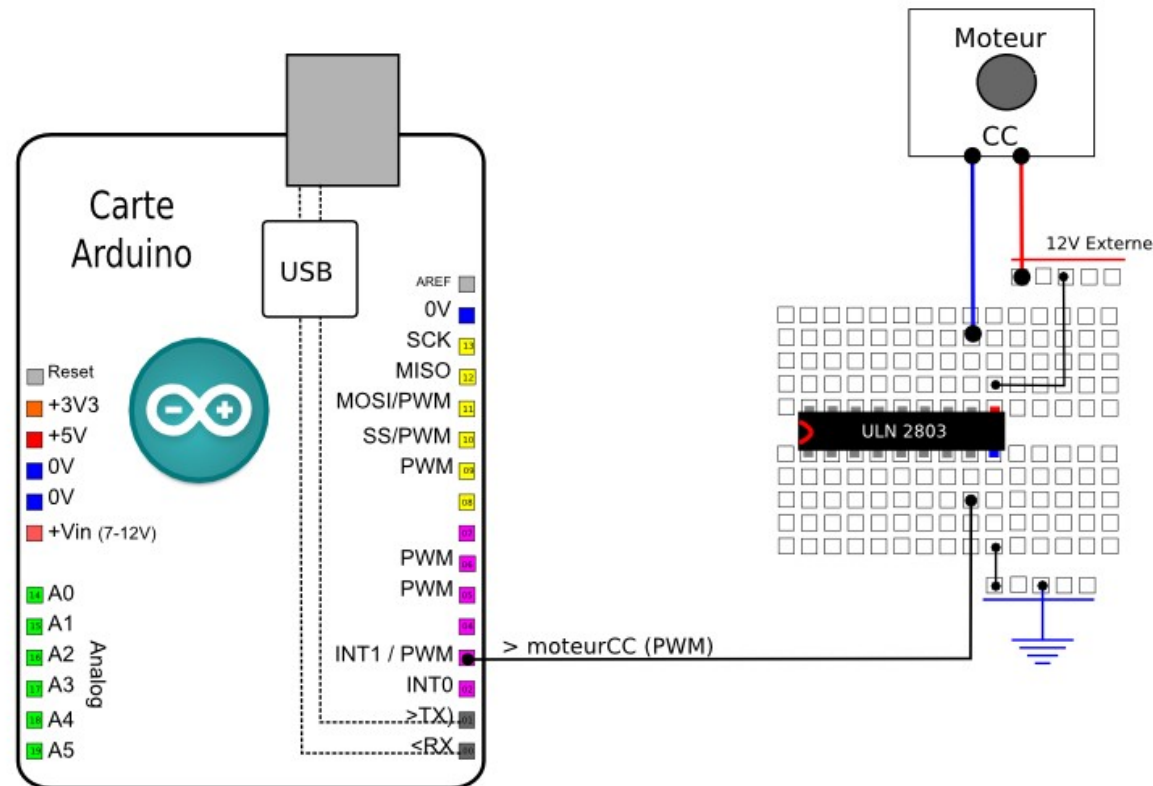
Si l'on souhaite disposer d'une intensité de plus de 500mA, on peut coupler plusieurs entrées et les sorties correspondantes :



Bien comprendre que ce type de circuit « ON/OFF » ne permet pas d'inverser la polarité aux bornes du dispositif connecté :
pour un moteur, l'ULN2803 ne pourra donc pas contrôler le sens de rotation mais seulement la vitesse. Le moteur tournera toujours dans le même sens.

12. Exemple d'amplification ON/OFF : Contrôler un moteur à courant continu : le montage

- On monte le ULN2803A à cheval sur le rail central de la plaque d'essai. On connecte un moteur CC simple (consommation < 500mA!) entre une broche en sortie de l'ULN 2803A et le V+. La tension doit être comprise entre 6 et 12V. La broche en entrée correspondante de l'ULN 2803A est connectée à l'Arduino.



TRUC TECHNIQUE : Pour utiliser facilement du 12V avec la plaque d'essai, alimenter votre carte Arduino via le connecteur Jack d'alimentation en 12V (à l'aide d'un bloc secteur par exemple) puis reprenez le 12V sur la broche Vin du bornier d'alimentation comme vous le feriez pour le 5V, tout simplement !

Remarquer également la souplesse du ULN2803A : si vous voulez utiliser du 6V, le montage est le même. Alimenter simplement la carte en 6V au lieu de 12V.

Noter que ce montage ne permet pas de contrôler le sens de rotation du moteur, mais uniquement sa vitesse.

Pour contrôler également le sens de rotation, il faudra utiliser une interface « moteur » capable à la fois de contrôler le sens ET la vitesse du moteur.

Tout cela est présenté dans les ateliers dédiés aux motorisations (servomoteur standard, à rotation continue, moteur CC et moto-réducteur, moteur pas à pas).

13. Exemple d'amplification ON/OFF : Contrôler la mise sous tension d'un moteur CC : le programme

Ce qu'on va faire ici...

- Le programme présenté fait la même chose que l'un des premiers programmes que vous avez écrit et qui fait clignoter une LED. La nouveauté, ici, ne se situe pas au niveau du code mais au niveau du montage, puisque l'on utilise un circuit qui amplifie l'intensité de la broche numérique en sortie utilisée.
- Autrement dit, le contrôle de dispositifs tels qu'un moteur CC ou des lampes, relais, etc...n'est pas plus compliqué à coder que l'allumage d'une simple LED !

Entête déclarative

- On déclare simplement une constante de broche désignant la broche utilisée pour le moteur.

Fonction setup()

- On configure la broche utilisée en sortie à l'aide de l'instruction `pinMode()`

Fonction loop()

- On met le moteur sous tension en appliquant un niveau HAUT sur la broche à l'aide de l'instruction `digitalWrite()`
- Ensuite, on réalise une pause à l'aide de l'instruction `delay()`
- Puis on éteint le moteur en appliquant un niveau BAS à l'aide de l'instruction `digitalWrite()`
- on termine par une pause plus longue, et ainsi de suite.

```
//--- entete déclarative
// = déclarer ici variables et constantes globales
const int MOTEUR=3; // constante désignant la broche de la LED

//--- la fonction setup() : exécutée au début et 1 seule fois
void setup() {

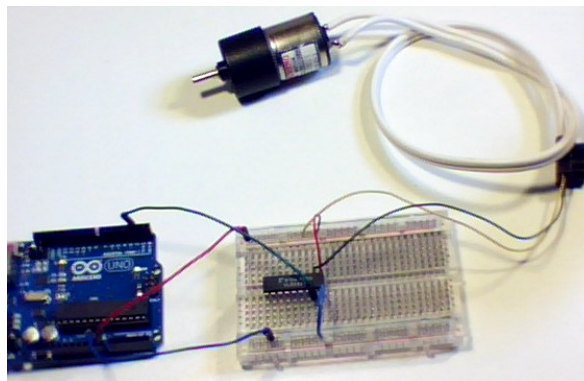
    pinMode(MOTEUR, OUTPUT); // met la broche en sortie
} // fin de la fonction setup()

//--- la fonction loop() : exécutée ensuite en boucle sans fin
void loop() {

    digitalWrite(MOTEUR,HIGH); // allume la LED
    delay(1000); // pause de n millisecondes
    digitalWrite(MOTEUR,LOW); // éteint la LED
    delay(3000); // pause de n millisecondes
} // fin de la fonction loop()
```

Fonctionnement du programme

- Une fois la carte Arduino programmée : le moteur tourne pendant 1 seconde puis s'arrête 3 secondes et ainsi de suite. C'est juste pour montrer le principe.



Purchased by Franck Ourion, franck.ourion@univ-lorraine.fr #6280170

Atelier Arduino : amplification de puissance, « interrupteur optique », protection opto-couplée avec Arduino.

14. Rappel : Le concept de modulation de largeur d'impulsion (MLI)

Rappelons à présent comment « simuler » des comportements analogiques à partir d'une broche d'E/S numérique en sortie. Rappelez-vous :

- numérique (ou ON/OFF) : tout ou rien, pas de niveau intermédiaire.
- analogique : variation progressive avec tous les niveaux intermédiaires.

Par exemple, dans le cas d'une lampe :

- en fonctionnement ON/OFF : la lampe est allumée ou éteinte
- en fonctionnement analogique : la lampe peut voir sa luminosité varier d'aucune lumière à luminosité maximale.

Simuler de l'analogique avec une broche numérique ...

Il serait pratique de pouvoir réaliser un comportement « analogique » avec une E/S broche numérique... mais comment faire ? La broche numérique ne peut en effet prendre que 2 niveaux : HAUT ou BAS...

Imaginez à présent que l'on définisse une période de temps donnée assez courte et que l'on allume la LED que pendant une partie de cette durée, par exemple 25% puis qu'on la laisse éteinte les 75% restant et que l'on recommence ensuite très rapidement. Que va-t-il se passer ? Vous verrez la LED allumée au quart de sa luminosité maximale ! Le tour est joué...

En effet, la tension moyenne au fil du temps vaudra 25% de 5V soit 1,25V !

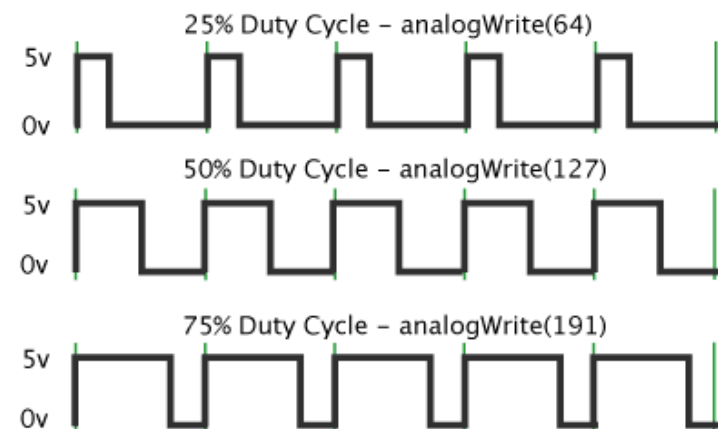
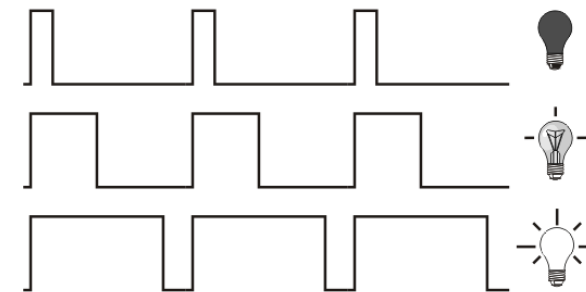
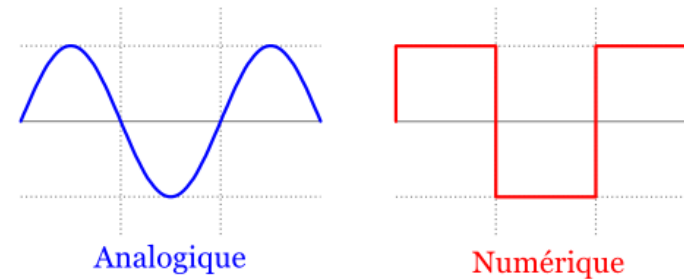
Le concept de Modulation de Largeur d'Impulsion (ou PWM)

Vous avez compris ? Alors, vous venez de saisir ce qui se cache derrière le concept de « Modulation de Largeur d'Impulsion » ou MLI (Pulse Width Modulation en anglais, ou PWM) !

En pratique :

- on appelle « duty-cycle » la période de temps de base, qui correspond à 100%
- on fixera la largeur de l'impulsion (ou « pulse width ») de 0 à 100%,
- ce qui en binaire s'exprimera par un octet dont la valeur sera entre 0(0%) et 255 (=100%).

Avec le langage Arduino, on générera une telle impulsion grâce à l'instruction `analogWrite()` **disponible seulement sur les broches dites PWM** (sigle ~ à côté soit les broches 3,5,6,9,10,11).



15. Contrôle simple de la vitesse (seule) d'un moteur (ou moto-réducteur CC) : le programme

Ce qu'on va faire ici...

- Le programme que je vous propose est à nouveau assez simple : on va simplement faire varier la vitesse du moteur de 0 (=arrêt) à 100% de la vitesse.
- Ce programme ressemble point pour point à celui que nous avons utilisé pour faire varier la luminosité d'une LED. La différence fondamentale est ici « technique » (utilisation d'un ULN 2803) mais au niveau programmation, c'est exactement la même chose ou presque.

Entête déclarative

- On déclare :
 - une constante de broche de contrôle du moteur. Cette broche doit être de type PWM (la 3,5,6,9,10 ou 11 – signalée par ~ devant)
 - une variable **int** fixant la vitesse
 - une variable **int** appelée largeur et fixant la largeur de l'impulsion

```
//--- entete déclarative
// = déclarer ici variables et constantes globales

const int MOTEUR=3; // constante désignant la broche du moteur
int vitesse=50; // variable fixant la durée de la pause en ms

int largeur=0; // variable pour la largeur d'impulsion
```

Fonction **setup()**

- On commence par configurer en sortie la broche de contrôle du moteur avec l'instruction **pinMode**(broche, sens),
- puis on réalise un test initial du moteur :
 - on met la broche du moteur à 1 pendant 1 seconde, le moteur tourne alors à fond.
 - puis on arrête le moteur pendant 1 seconde.

```
//--- la fonction setup() : exécutée au début et 1 seule fois
void setup() {

    pinMode(MOTEUR, OUTPUT); // met la broche en sortie

    //-- test du moteur --
    digitalWrite(MOTEUR,HIGH); // moteur à fond
    delay(1000); // 1 seconde
    digitalWrite(MOTEUR,LOW); // moteur à l'arrêt
    delay(1000);
} // fin de la fonction setup()
```

Fonction `loop()`

- A l'aide d'une première boucle `for()`, on augmente progressivement la vitesse de 0 à 100% en élargissant progressivement l'impulsion PWM générée avec l'instruction `analogWrite(broche, largeur)`
- De la même façon, on utilise une seconde boucle `for()` pour réduire la vitesse de 100% à 0%.
- Noter l'utilisation de la fonction `map()` qui permet de transposer la valeur en % (échelle 0-100) en la valeur correspondante sur l'échelle PWM 0-255.
- Des instructions `delay()` fixent la pause entre chaque changement de vitesse.

```
//--- la fonction loop() : exécutée ensuite en boucle sans fin
void loop() {

    for (int i=0; i<=100; i++) { // boucle croissante

        largeur= map(i,0,100, 0,255); // ré-échelonne i (0-100%) vers (0-255)
        analogWrite(MOTEUR,largeur); // génère impulsion PWM voulue sur la broche
        delay(vitesse); // pause

    } // fin for i

    for (int i=100; i>=0; i--) { // boucle décroissante

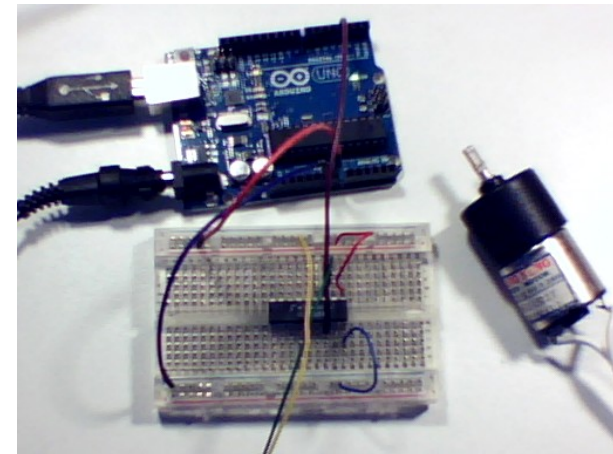
        largeur= map(i,0,100, 0,255); // ré-échelonne i (0-100%) vers (0-255)
        analogWrite(MOTEUR,largeur); // génère impulsion PWM voulue sur la broche
        delay(vitesse); // pause

    } // fin for i

} // fin de la fonction loop()
```

Fonctionnement du programme

- Une fois la carte Arduino programmée :
 - le moteur tourne à fond 1 seconde puis s'arrête,
 - puis le moteur se remet à tourner progressivement de plus en plus vite de 0% à 100%
 - puis la vitesse diminue à nouveau de 100% jusqu'à l'arrêt (0%)
 - et ainsi de suite...

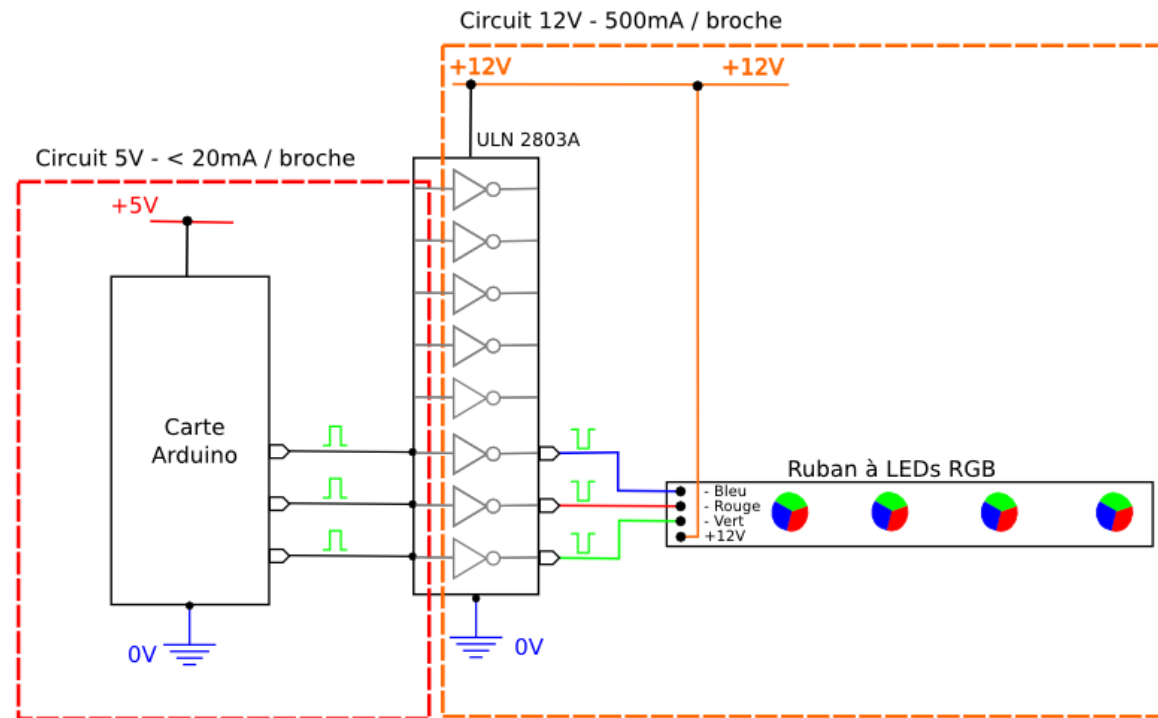


Ce programme fonctionne car le ULN 2803 répercute quasi instantanément l'état de la broche en entrée sur la sortie amplifiée : **l'ULN 2803 est donc un circuit « transparent » aux impulsions, notamment PWM (ou MLI).**

16. Exemple d'amplification ON/OFF : contrôler un ruban à LEDs RGB avec Arduino : le schéma théorique.

Le ruban à LEDs RVB dispose de 4 broches :

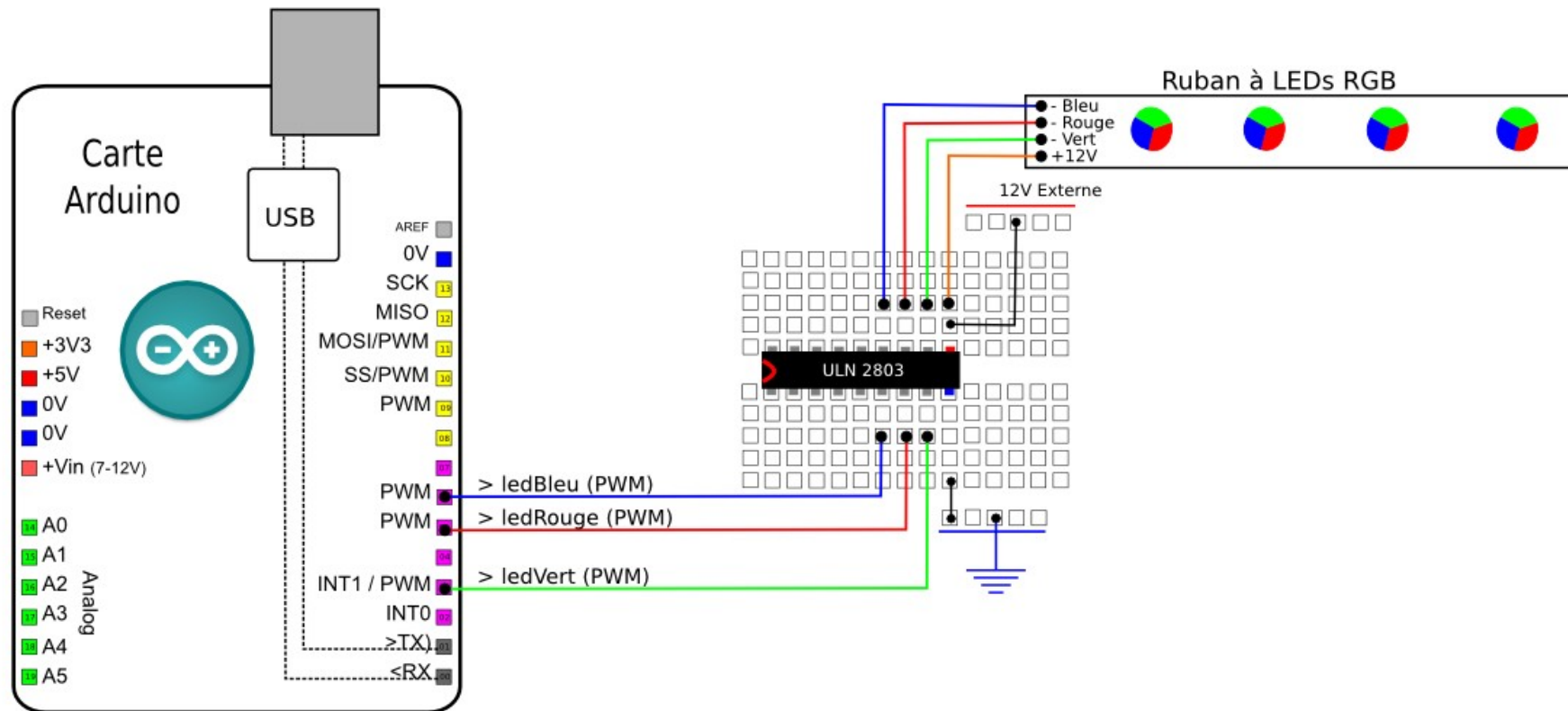
- une broche de 12V
- une broche de commande du ROUGE (active sur 0V)
- une broche de commande du VERT (active sur 0V)
- une broche de commande du BLEU (active sur 0V)



17. Exemple d'amplification ON/OFF : contrôler un ruban à LEDs RGB avec Arduino : le montage à réaliser

- Noter que les broches RVB du ruban à LED sont actives sur 0V : comme on utilise un ULN2803 qui a pour effet d'inverser le niveau d'entrée, un niveau numérique HAUT donnera du 0V sur la broche LED

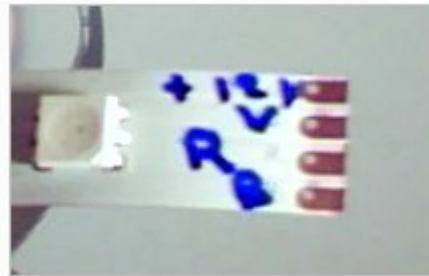
ATTENTION : Un Ruban à LEDs RVB peut demander une intensité significative si il est long : vous devrez donc disposer d'une alimentation capable de fournir du 12V en au moins 2 Ampères pour être à l'aise... Une alimentation de PC fera l'affaire.



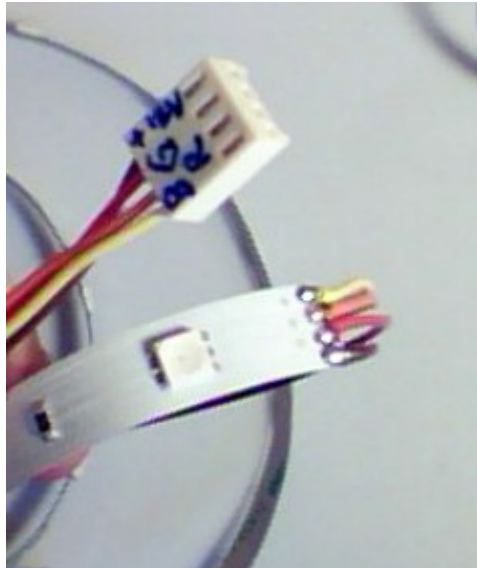
18. *Truc technique : utiliser facilement un ruban RGB*

Le ruban à LEDs RVB dispose de 4 broches :

- une broche d'alimentation de 12V / 2 ou 3A voire plus selon le nombre de LEDs
- une broche de commande du ROUGE (active sur 0V)
- une broche de commande du VERT (active sur 0V)
- une broche de commande du BLEU (active sur 0V)



On pourra souder un connecteur sur fils à 4 broches sur l'extrémité du ruban pour faciliter les connexions avec la plaque d'essai.



19. Exemple d'amplification ON/OFF : contrôler un ruban à LEDs RGB avec Arduino : le programme

Ce qu'on va faire ici...

- Ce programme teste l'affichage des couleurs sur un ruban à LED colorées RVB (ou ruban à LED RGB, c'est pareil). Les LEDs sont alimentées en 12V : on utilisera donc un ULN 2803 (octuple amplificateur inverseur 500mA) en guise d'interface de puissance.
- Voir également : www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ArduinoInitiationLedsDiversTestRubanRVB

Entête déclarative

- On déclare 3 constantes `int` désignant les 3 broches PWM utilisées pour contrôler les leds RGB. Ces broches doivent être de type PWM (parmi les broches 3,5,6,9,10 ou 11 – signalée par ~ devant).
- On déclare par ailleurs des constantes et variables `int` utiles dans ce programme.

```
//--- entete déclarative
// = déclarer ici variables et constantes globales

const int ledR=3; // constante désignant la broche de la LED
const int ledV=5; // constante désignant la broche de la LED
const int ledB=6; // constante désignant la broche de la LED

const int R=1; //--- constante binaire couleur R
const int V=1; //--- constante binaire couleur G
const int B=1; //--- constante binaire couleur B

int vitesse=10; // variable fixant la durée de la pause
```

Fonction `setup()`

- On configure en sortie les broches utilisées.

```
//--- la fonction setup() : exécutée au début et 1 seule fois
void setup() {

    pinMode(ledR, OUTPUT); // met la broche en sortie
    pinMode(ledV, OUTPUT); // met la broche en sortie
    pinMode(ledB, OUTPUT); // met la broche en sortie

} // fin de la fonction setup()
```

Fonction `loop()`

- On commence par allumer successivement chaque couleur en mettant au niveau HAUT la broche voulue pendant 1 seconde.

Bien noter l'enchaînement des niveaux de tension avec l'ULN 2803 :
broches Arduino Haut => entrée ULN 2803 Haut => sortie ULN à 0V =>
LEDs de la couleurs allumées (actif sur 0V).

- Ensuite, on appelle la fonction `ledRVB()` (voir ci-après) qui permet de fixer le niveau de chaque couleur :
 - tout d'abord, à l'aide de boucles pour faire varier l'intensité de chaque couleur
 - puis en combinant les couleurs 2 à 2
 - puis de façon aléatoire



```
//--- la fonction loop() : exécutée ensuite en boucle sans fin  
void loop() {
```

```
    //---- vert ---  
    digitalWrite(ledV,HIGH); // allume la couleur voulue  
    delay(1000); // pause  
    digitalWrite(ledV,LOW); // allume la couleur voulue  
    delay(1000); // pause
```

```
    //---- rouge ---  
    digitalWrite(ledR,HIGH); // allume la couleur voulue  
    delay(1000); // pause  
    digitalWrite(ledR,LOW); // allume la couleur voulue  
    delay(1000); // pause
```

```
    //---- bleu ---  
    digitalWrite(ledB,HIGH); // allume la couleur voulue  
    delay(1000); // pause  
    digitalWrite(ledB,LOW); // allume la couleur voulue  
    delay(1000); // pause
```

```
    //--- couleur de base ---  
    for (int i=0; i<=255; i++) ledRVB(i,0,0); // variation de rouge  
    for (int i=0; i<=255; i++) ledRVB(0,i,0); // variation de vert  
    for (int i=0; i<=255; i++) ledRVB(0,0,i); // variation de bleu
```

```
    //---- mix ---  
    for (int i=0; i<=255; i++) ledRVB(i,0,255-i); // variation de rouge +  
    bleu  
    for (int i=0; i<=255; i++) ledRVB(255-i,i,0); // variation de rouge +  
    vert  
    for (int i=0; i<=255; i++) ledRVB(0,255-i,i); // variation de vert +  
    bleu
```

```
    //---- aléatoire ----  
    for (int i=0; i<=50; i++) {  
        ledRVB(random(0,255),random(0,255),random(0,255)); // variation au  
        hasard des 3 couleurs  
        delay(200);  
    }
```

```
} // fin de la fonction loop()
```

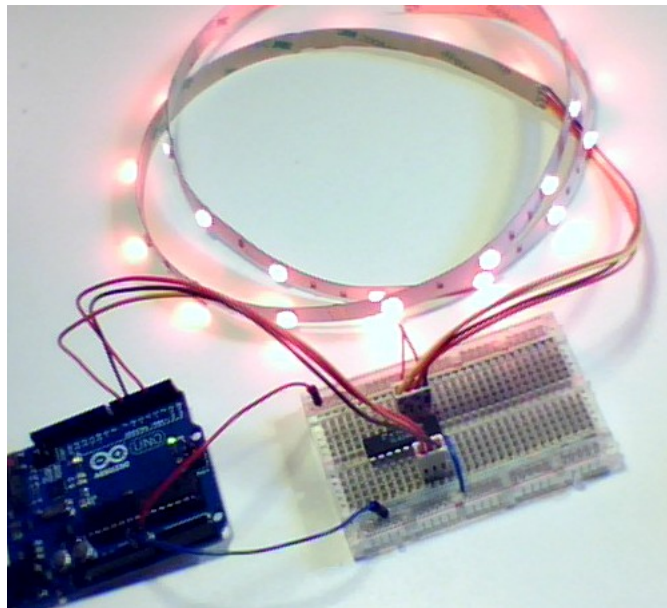

Fonction **ledRVB()**

- Cette fonction :
 - ne renvoie rien (type void)
 - reçoit en paramètre 3 valeurs int correspondant chacune au niveau de chaque couleur Rouge, Vert, Bleu.
- Cette fonction applique l'impulsion PWM voulue pour chaque couleur permettant la combinaison facile de toutes les couleurs.

```
//----fonction pour combiner couleurs----  
  
void ledRVB(int Rouge, int Vert, int Bleu) {  
  
  analogWrite(ledR,Rouge); // proportion de rouge  
  analogWrite(ledV,Vert); // proportion de vert  
  analogWrite(ledB,Bleu); // proportion de bleu  
  
  delay(vitesse); // pause de n millisecondes  
  
}
```

Fonctionnement du programme

- Une fois la carte Arduino programmée, le ruban à LED s'allume :
 - tout d'abord successivement des couleurs rouge, vert bleu
 - puis dans diverses combinaisons de couleurs.



L'intérêt premier de ce montage est l'utilisation de l'ULN 2803 comme interface d'amplification ON/OFF.
Pour plus détails sur les impulsions PWM et les LEDs RVB, voir l'atelier consacré aux sorties analogiques et impulsions.

Purchased by Franck Ourion, franck.ourion@univ-lorraine.fr #6280170

Atelier Arduino : amplification de puissance, « interrupteur optique », protection opto-couplée avec Arduino.

20. Technique : Informations pratiques sur les relais

Description

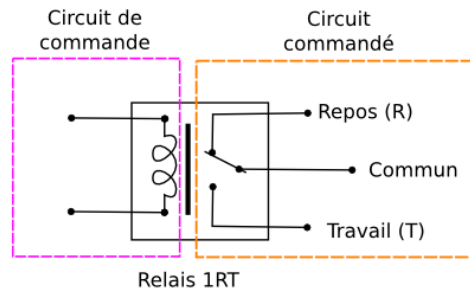
- Un relais est un dispositif à la fois mécanique et électrique (on dit « électro-mécanique ») constitué :
 - d'une bobine d'électro-aimant alimentée par une tension de commande,
 - d'un « interrupteur » mécanique commandé par l'électro-aimant



- En un mot, un relais est un « bouton poussoir » à commande électrique : au lieu d'appuyer sur un bouton, on applique une tension sur la bobine et le contact reste fermé tant que la tension est présente sur la bobine.

Schéma électrique

- Electriquement, un relais est constitué :
 - d'une bobine d'électro-aimant d'une part, alimentée par la tension de commande
 - d'un contacteur simple ou double d'autre part :
 - le contact en position « hors tension » est appelé repos ou R
 - le contact en position « sous tension » est appelé travail ou T



- En pratique, avec un relais, vous pouvez contrôler une machine à laver avec Arduino si vous voulez !

Principe de fonctionnement

- Lorsque la tension est présente sur la bobine, le contact Travail est actif.
- Lorsque la tension est absente sur la bobine, le contact Repos est actif.
- Le relais assure simultanément :
 - l'**adaptation en tension** (avec du 5Vdc on commande du 12Vdc qui commande du 220Vac..)
 - l'**isolation** des circuits électriques de commande et de charge

Caractéristiques électriques

Les relais sont caractérisés :

- par la **tension d'alimentation** de la bobine, qui peut être soit continue (DC) ou alternative (AC) :
 - 6V ou 12V -DC : ceux qu'on peut facilement utiliser avec Arduino via un ULN 2803
 - 230V AC : utilisables en « capteurs » de présence de tension 220V (voir atelier BP en entrée)
- par la tension / intensité supportées par le contact : 230VAC/ 10A, 230VAC/16A, etc...
- le type de contact : 1 RT (rpos/travail), 2 RT (2 repose/travail), etc..

Familles de relais

- Plusieurs familles standard de relais existent type 40, type 55, etc... ayant des caractéristiques variées. En pratique, les relais compacts de la série Finder type 41 sont pratiques

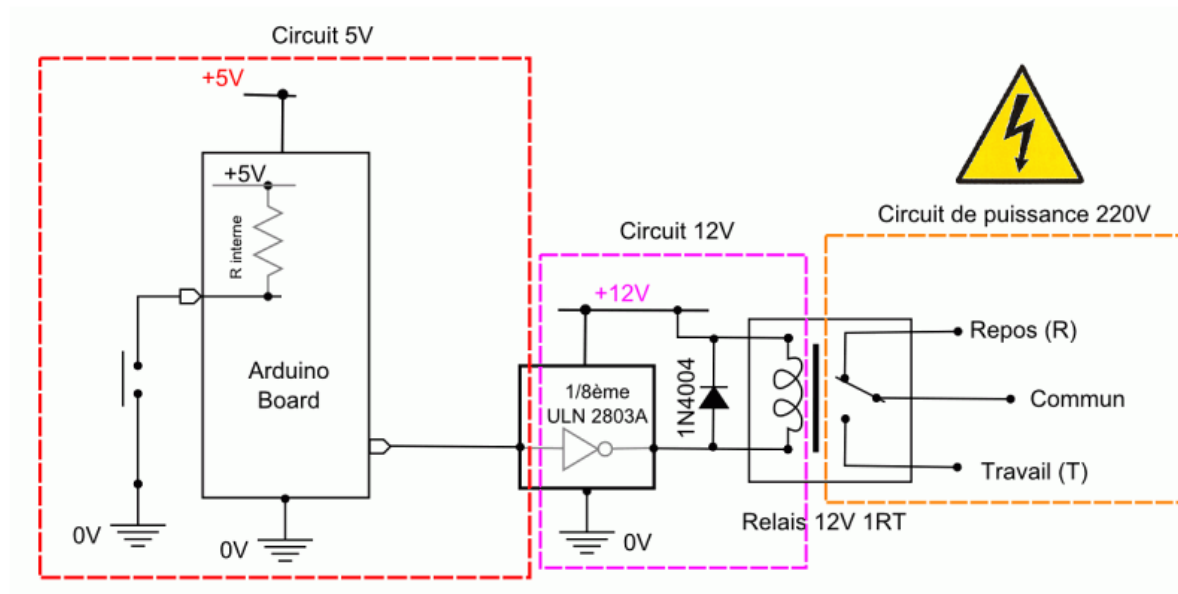
Truc pratique

- Les relais de type 41 peuvent être facilement utilisés au sein d'un tableau électrique à l'aide de supports pour rail DIN, avec borniers à vis, que l'on trouvera en ligne. (Par exemple, chez Gotronic)

Pour info : autre usage possible d'un relais

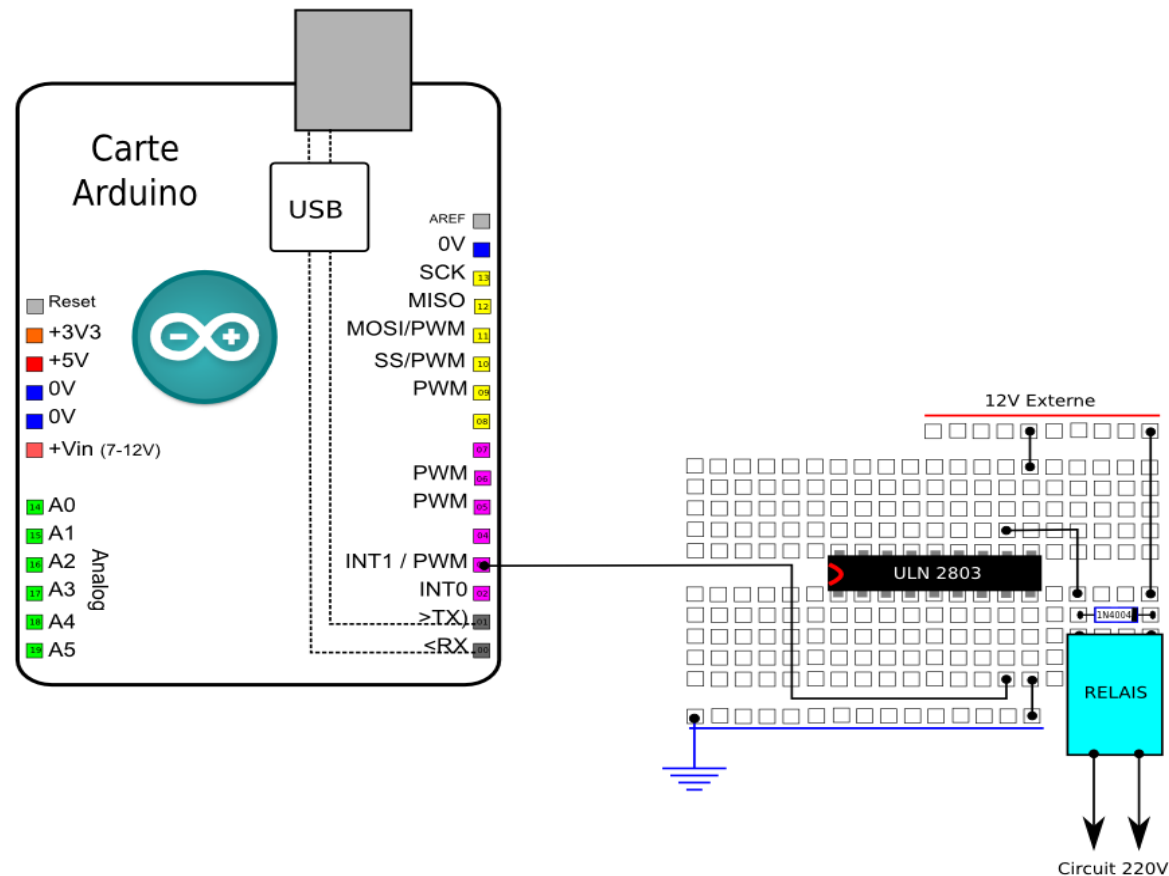
- Il existe des relais 220V (alimentation de la bobine) : de tels relais peuvent donc être utilisés comme des « détecteurs » de 220V, en connectant le contact Travail sur une broche en entrée de la carte Arduino. Le relais se comportera alors comme un « bouton poussoir » commandé par le 220V.

21. Pour info : Contrôler un relais via un ULN 2803 avec Arduino : le schéma théorique



ATTENTION : L'utilisation du 220V est dangereuse et nécessite impérativement de respecter des règles de sécurité précises. Les mineurs ne doivent en aucun cas utiliser du 220V sans la présence d'un adulte.

22. Pour info : Contrôler un relais via un ULN 2803 avec Arduino : le montage à réaliser



Bon à savoir :

Plusieurs autres dispositifs sont l'équivalent de la bobine d'un relais ou d'un moteur et pourront être contrôlés également via un ULN 2803, notamment : les électro-aimants, les électro-vannes, etc... et plus généralement, tout dispositif utilisant une bobine alimentée en 12V/500mA max. Pour tous les dispositifs nécessitant des **tensions** d'alimentation plus élevées ou alternatives et/ou des **intensités** d'alimentation plus élevées, on pourra simplement mettre un relais entre l'ULN 2803 et le dispositif à contrôler.

En un mot, avec un relais, vous pouvez contrôler une machine à laver ou une pompe 220V avec Arduino si vous le voulez !

23. Interrupteur optique et isolation opto-couplée

Le problème à résoudre dans certaines situations...

- Dans un certain nombre de situations, on pourra avoir besoin de simplement réaliser un « contact » à partir d'une broche Arduino pour déclencher un circuit secondaire...
- ... mais sans avoir besoin d'utiliser la « lourdeur » d'un ULN 2803 éventuellement suivi d'un relais.
- Dans ces situations, le circuit secondaire ne nécessite qu'une intensité modérée et il existe une alternative plus légère à l'utilisation du relais...

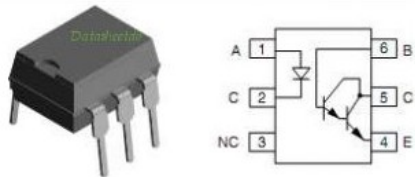
Une idée lumineuse...



- Une nouvelle fois, la solution est élégante et utilise par un mécanisme ingénieux :
 - dans le circuit numérique de commande, on utilise le niveau HAUT d'une broche en sortie pour allumer une LED
 - dans le circuit de charge, que l'on souhaite activer, on place un photo-transistor en regard de la LED : ainsi, lorsque la LED est allumée, le photo-transistor laisse passer le courant, ce qui correspond à un simple contact.

En pratique

- Il existe des composants qui rassemblent dans un même boîtier une LED et un photo-transistor : ce sont les opto-coupleurs.
- Un modèle par exemple est le 4N32 ou le 4N27 :

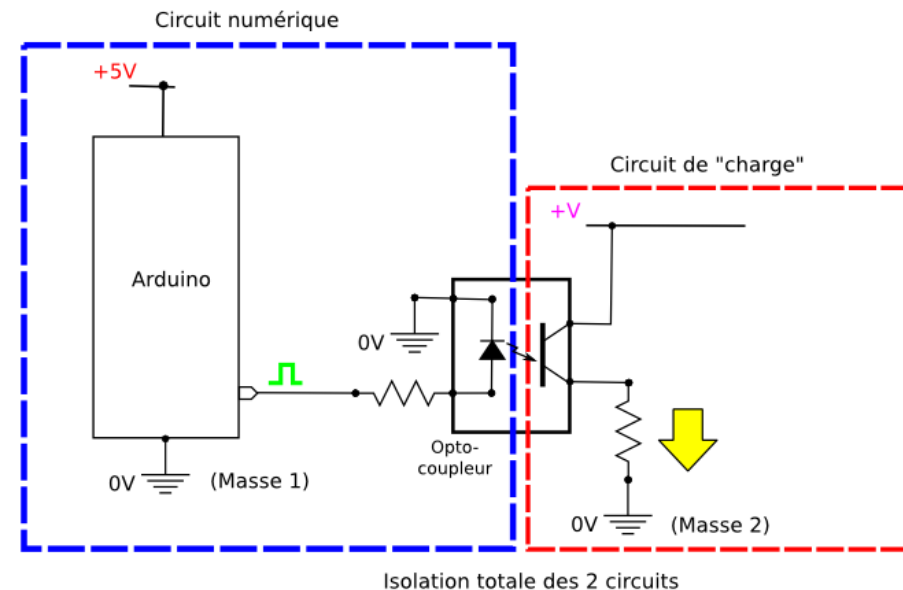


Différence entre relais et opto-coupleur optique

- Le relais nécessite un ULN 2803 (ou équiv.) pour alimenter la bobine à partir d'une broche numérique, mais peut contrôler un circuit de forte intensité ou de tension élevée.
- L'opto-coupleur peut se connecter directement sur une broche numérique d'Arduino (la LED), mais ne pourra contrôler un circuit nécessitant une faible intensité de fonctionnement.

Notion de protection opto-couplée :

- Un autre intérêt majeur d'un opto-coupleur est qu'il permet d'isoler totalement le circuit de commande électrique et le circuit de charge : on parle à ce sujet d'isolation des masses.
- Ceci est très pratique et intéressant lorsque le circuit de charge risque de perturber le fonctionnement du circuit de commande numérique.



Bon à savoir :

Il existe plusieurs variantes d'opto-coupleur : les opto-fourches, les opto-coupleurs en boîtier, ou encore réflectif (détection de ligne, trou).

24. Rappel : Fiche composant : découvrir le transistor et le photo-transistor

Description

En électronique, le transistor est un composant semi-conducteur (il en existe 2 types dits PNP ou NPN) dans un petit boîtier qui dispose de 3 broches :

- la base B qui reçoit une intensité de déclenchement I_b
- le collecteur C qui laisse entrer une intensité I_c proportionnelle à I_b
- l'émetteur E qui laisse sortir une intensité valant $I_e = I_c + I_b$



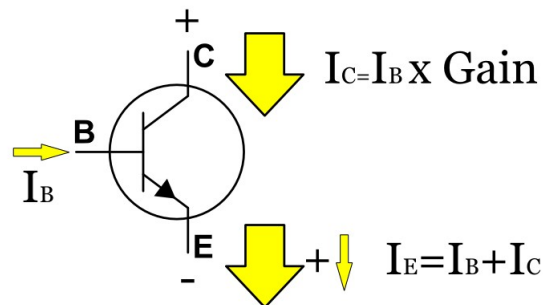
A savoir :

Le transistor est un composant essentiel, qui date des années 40, et qui a révolutionné l'électronique et permis l'apparition de l'électronique numérique et des ordinateurs. Les processeurs des ordinateurs actuels possèdent des millions de transistors miniaturisés !!

Le micro-contrôleur de votre carte Arduino lui-même intègre environ 500 000 transistors !

Principe de fonctionnement

Le principe fondamental de fonctionnement d'un transistor est le suivant : une petite intensité circulant sur la broche de la base va provoquer la circulation d'une intensité importante proportionnelle entre le collecteur et l'émetteur.



Le Transistor NPN

Pour faire simple, on peut dire qu'un transistor est un « multiplicateur » d'intensité : il multiplie l'intensité de la base et l'intensité résultante circule entre le collecteur et l'émetteur. Le coefficient multiplicateur est appelé **gain**.

Purchased by Franck Ourion, franck.ourion@univ-lorraine.fr #6280170

Atelier Arduino : amplification de puissance, « interrupteur optique », protection opto-couplée avec Arduino.

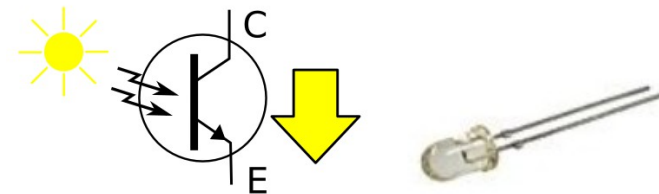
Modes de fonctionnement d'un transistor

Le transistor est un composant qui peut être utilisé aussi bien en mode analogique que « numérique » :

- **en mode analogique**, la variation d'intensité sur la base se répercute immédiatement en variation d'intensité du collecteur. C'est ce principe qui est à la base des amplificateurs audio et autres appareils de radio (dont lui vient d'ailleurs le nom de transistor).
- **en mode « numérique » ou « ON/OFF » appelé également mode saturé** : dès qu'une intensité est présente sur la base, le courant de collecteur est d'emblée maximal. L'absence de courant sur la base ne laisse passer aucun courant de collecteur. C'est une sorte d'interrupteur à commande électrique. C'est ce mode de fonctionnement qui est à la base de tous les circuits logiques et numériques.

Une variante du transistor : le photo-transistor

Dans ce composant, la broche de la base est remplacée par une zone sensible à la lumière infra-rouge. Le photo-transistor n'a donc que 2 broches :



Le principe de fonctionnement est le suivant : une intensité lumineuse présente sur la zone photo-sensible va provoquer la circulation d'un courant de collecteur qui sera proportionnel à l'intensité lumineuse reçue.

Le photo-transistor pourra être utilisé soit en mode analogique ou saturé.

Les transistors avec Arduino en pratique

Afin de ne pas compliquer inutilement les montages, en pratique, on n'utilisera quasiment pas les transistors « bruts » avec Arduino, mais plutôt des circuits les utilisant tels que le circuit intégré ULN 2803 qui intègre 8 étages d'amplification ON/OFF et ne nécessite aucun composant externe.

Par contre, on utilisera le photo-transistor, utilisé au sein des opto-coupleurs, comme nous allons le voir par la suite.

Remarque : l'étude des transistors et de leur utilisation est un domaine passionnant et qui peut faire l'objet de livres entiers. Ici, nous en parlons uniquement pour introduire le photo-transistor. Si vous voulez approfondir, voir notamment : <http://fr.wikipedia.org/wiki/Transistor>

25. Rappel : Fiche composant : découvrir l'opto-coupleur en fourche

Description

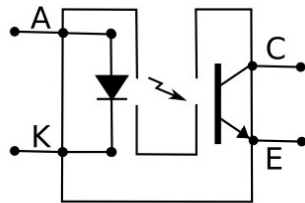
L'opto-coupleur en fourche est un composant qui associe en fait 2 composants différents qui sont positionnés face à face dans un même boîtier (2 broches par composant soit 4 broches en tout) :

- d'une part une photo-diode ou LED infra-rouge qui fonctionne comme une LED classique et émet une lumière invisible dite infra-rouge,
- d'autre part un photo-transistor infra-rouge utilisé ici pour détecter la présence de la lumière infra-rouge (patte courte = Emetteur).



Schéma interne

Le schéma théorique de l'optocoupleur est le suivant :



Opto-coupleur fourche
(Ex: LTH301-7)

- On retrouve d'une part la **LED infra-rouge signalée par les lettres A et K** sur le boîtier de l'opto-coupleur correspondant à l'anode (A = +) et la cathode (K = - = patte courte)
- On retrouve d'autre part le **photo-transistor signalé par les lettres C et E** sur le boîtier de l'opto-coupleur correspondant au collecteur (C = +) et à l'émetteur (E = - = patte courte).

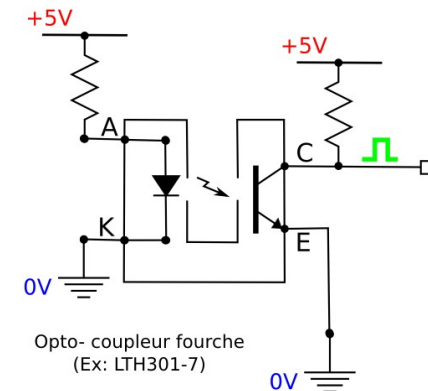
Principe de fonctionnement

- Lorsque la LED infra-rouge est allumée, la base du photo-transistor est éclairée et le photo-transistor laisse passer le courant.
- Lorsque la LED infra-rouge est éteinte, ou si un objet se trouve dans la fente, la base du photo-transistor ne laisse passer aucun courant.

Le montage type

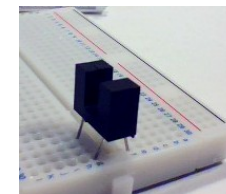
L'utilisation de ce composant nécessite en fait la réalisation de 2 circuits :

- tout d'abord, le circuit de la LED infra-rouge, qui s'utilise comme une LED standard. On pourra donc se contenter de mettre une résistance en série avec LED pour qu'elle soit allumée. **Comme vu précédemment, si on désire une intensité de 13mA dans la LED, on utilisera, d'après la loi d'ohm, une résistance de $R = U/I = 3,5V/0,013A = 270 \text{ Ohms}$.**
- le circuit du photo-transistor qui sera ici utilisé en mode saturé, autrement dit :
 - si pas d'objet dans la fente = lumière IR présente, alors la tension du collecteur vaudra 0V
 - si objet présent dans la fente = pas de lumière IR, alors la tension du collecteur vaudra 5V
 - pour obtenir ce résultat, on se contente d'utiliser une résistance de quelques milliers d'Ohms entre le collecteur et le +5V. **En pratique, on utilisera 4,7KOhms avec un LTH301-7.**

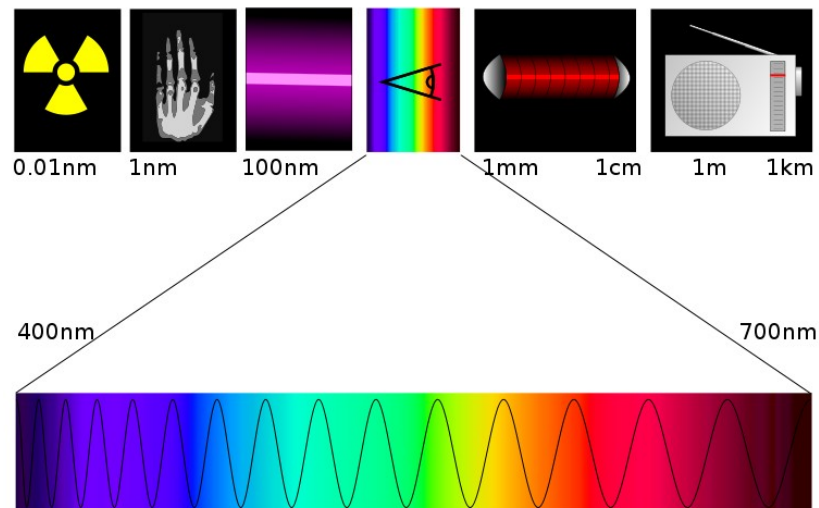


Principe d'utilisation sur une plaque d'essai

L'opto-coupleur s'utilise simplement, à cheval sur le rail central :

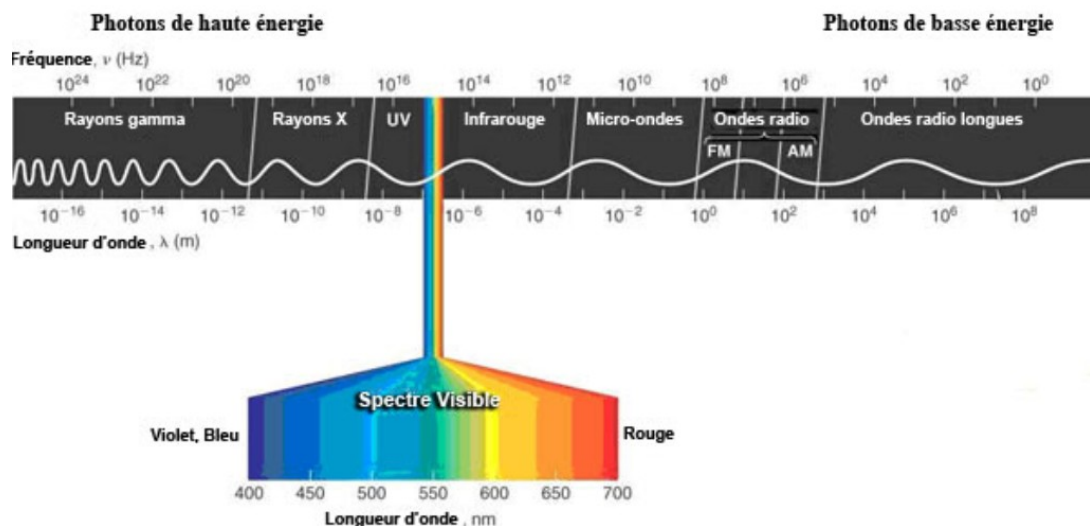


26. Pour info : le spectre des ondes électro-magnétiques et de la lumière visible



<http://fr.wikipedia.org/wiki/Fichier:Spectre.svg>

La lumière, tout comme les ondes radio ou les micro-ondes sont des ondes dites « électro-magnétiques »



source : <http://www.lampexpress.fr/images/ampoules-fiche-technique/spectre-lumiere.jpg>

La lumière visible ne représente qu'une toute petite partie de l'ensemble des ondes électro-magnétiques.

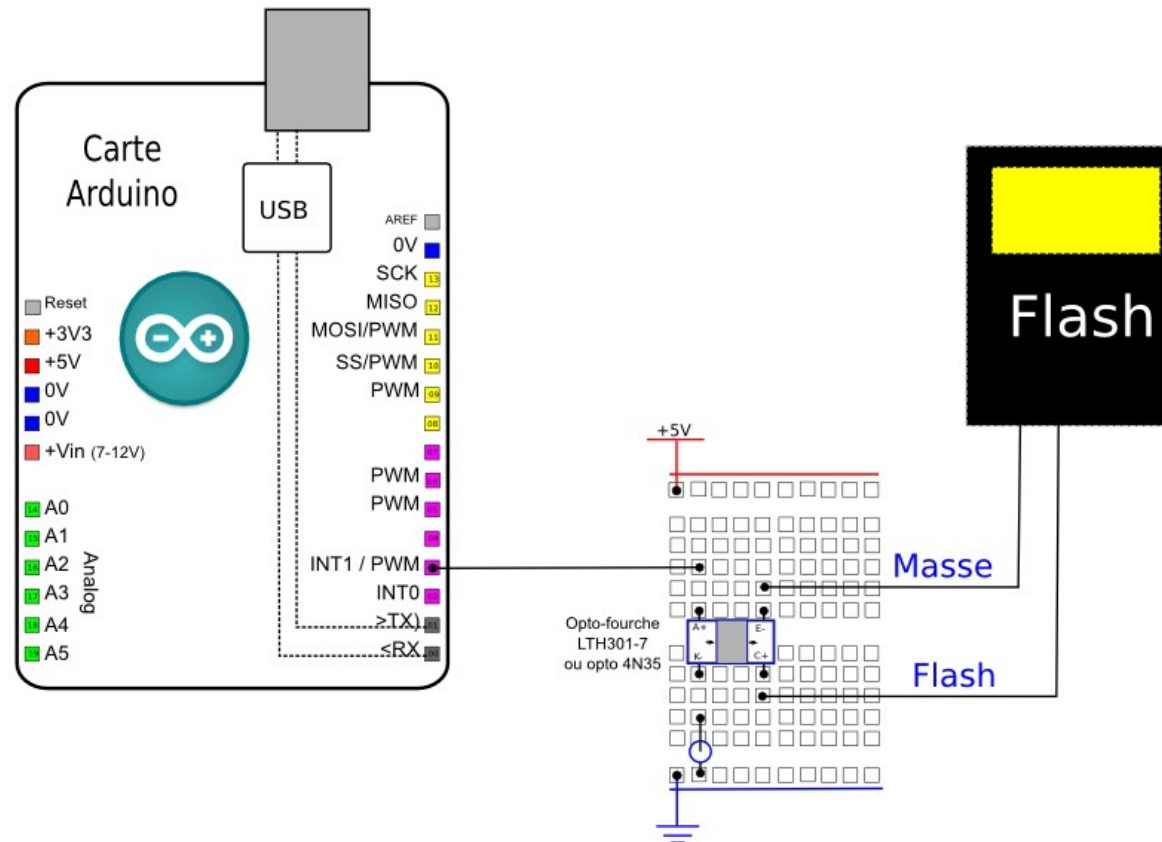
La lumière infra-rouge, à laquelle est sensible le photo-transistor, est une lumière invisible à l'oeil nu, de même que la lumière ultra-violette.

Purchased by Franck Ourion, franck.ourion@univ-lorraine.fr #6280170

Atelier Arduino : amplification de puissance, « interrupteur optique », protection opto-couplée avec Arduino.

27. Exemple : Contrôler un flash photographique avec un opto-coupleur : le montage

- Voici un montage d'exemple d'utilisation d'un opto-coupleur en tant que dispositif de commande : ici, l'opto-coupleur permet de contrôler un Flash photographique à partir d'une broche Arduino.
- Lorsque la broche sera mise à HAUT, la LED s'allumera et le Flash sera déclenché.



Ce montage est donné à titre d'illustration du principe. Si vous souhaitez le réaliser, se reporter au préalable à la documentation du flash utilisé ou à un exemple de montage concret que vous trouverez sur internet. On pourra notamment utiliser un buzzer piézo-électrique en tant que détecteur de choc pour déclencher le flash. Cette technique est notamment utilisée pour capturer des mouvements rapides.



source : http://www.beantownboogiedown.com/storage/high-speed-5.jpg?_SQUARESPACE_CACHEVERSION=1246942167717

Purchased by Franck Ourion, franck.ourion@univ-lorraine.fr #6280170

Atelier Arduino : amplification de puissance, « interrupteur optique », protection opto-couplée avec Arduino.

28. A présent, vous devriez être capable :

- D'utiliser des dispositifs nécessitant une alimentation continue de 6 à 12V voire plus et une intensité de fonctionnement jusqu'à 1A ou plus
- De contrôler la vitesse de rotation d'un moteur ou les couleurs d'un ruban à LEDs avec Arduino,
- D'avoir quelques notions de base sur l'utilisation des relais avec Arduino,
- De comprendre le principe de l'interrupteur optique et de l'isolation opto-couplée.

Table des matières

NIVEAU INTERMEDIAIRE |

Sorties numériques :

amplification de puissance, « interrupteur optique », protection opto-couplée avec Arduino.

Intro |

|

Matériel nécessaire pour les ateliers Arduino |

Matériel spécifique nécessaire pour cet atelier |

Notion d'électricité élémentaire à bien connaître |

Notion de tension alternative, continue, régulée.. |

Caractéristiques électriques globales de la carte Arduino |

Caractéristiques électriques d'une broche Numérique Arduino en sortie |

Notion d'amplification de puissance et d'interface de puissance |

Info technique : les circuits intégrés en boîtier DIL |

Exemple de CI d'amplification de puissance ON/OFF : le ULN 2803A |

Exemple de CI d'amplification de puissance ON/OFF : le CI 2803A (suite) |

Exemple d'amplification ON/OFF : Contrôler un moteur à courant continu : le montage |

Exemple d'amplification ON/OFF : Contrôler la mise sous tension d'un moteur CC : le programme |

Rappel : Le concept de modulation de largeur d'impulsion (MLI) |

Contrôle simple de la vitesse (seule) d'un moteur (ou moto-réducteur CC) : le programme |

Exemple d'amplification ON/OFF : contrôler un ruban à LEDs RGB avec Arduino : le schéma théorique. |

Exemple d'amplification ON/OFF : contrôler un ruban à LEDs RGB avec Arduino : le montage à réaliser |

Truc technique : utiliser facilement un ruban RGB |

Exemple d'amplification ON/OFF : contrôler un ruban à LEDs RGB avec Arduino : le programme |

Technique : Informations pratiques sur les relais |

Pour info : Contrôler un relais via un ULN 2803 avec Arduino : le schéma théorique |

Pour info : Contrôler un relais via un ULN 2803 avec Arduino : le montage à réaliser |

Interrupteur optique et isolation opto-couplée |

Rappel : Fiche composant : découvrir le transistor et le photo-transistor |

Rappel : Fiche composant : découvrir l'opto-coupleur en fourche |

Pour info : le spectre des ondes électro-magnétiques et de la lumière visible |

Exemple : Contrôler un flash photographique avec un opto-coupleur : le montage |

A présent, vous devriez être capable : |

Bravo !
vous avez terminé cet atelier Arduino !



Prêt pour la suite ? Retrouvez de nombreux autres thèmes d'ateliers Arduino ici :

http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ATELIERS