

Club de Robotique et d'Electronique Programmable de Ploemeur

Atelier du 6 mai 2023

Nicolas Le Guerroué

8 mai 2023

Table des matières

	Page
Glossaire	2
Liste des figures	2
Préambule	3
0.1 Conventions	4
1 La conversion des grandeurs	6
1.1 Les Convertisseurs Analogiques-Numériques	6
1.1.1 Résolution de l'ADC	7
1.1.2 Fréquence d'échantillonnage	8
1.1.3 Le multiplexage	10
1.1.4 Les ADC externes	12
1.1.5 Protection des ADC	13
1.2 Questionnaire	14
I Les interfaces de puissance	15
2 Introduction	16
3 Les transistors bipolaires	17
3.1 Présentation	17
3.2 Conventions	17
3.2.1 Les familles de transistors bipolaires	18
3.3 Les paramètres de sélection du transistor	18
3.4 Le principe	19
3.5 Exemple	20
3.6 Mise en pratique	21
3.6.1 Branchements	21
3.6.2 Dimensionnement de la résistance	22
3.6.3 Exemple de programme Arduino	22
4 Les transistors MOSFET	24
4.1 Présentation	24
4.2 Conventions	24
4.2.1 Les familles de transistors MOSFET	25
4.3 Les paramètres de sélection du transistor	25
4.4 Le principe	26
4.5 Comparaison avec les transistors bipolaires	26
4.6 Mise en pratique	26
4.6.1 Branchements	26
4.6.2 Exemple de programme Arduino	27

4.6.3	Pour aller plus loin	28
4.7	Les fiches techniques	29
4.8	Les relais électromagnétiques	30
4.8.1	Principe	30
4.8.2	Symbole	31
4.8.3	La diode de roue libre	31
4.8.4	En pratique	34
4.9	L'isolation galvanique	34
5	Conclusion	39
5.1	Ce qu'il faut retenir	39
5.2	Questionnaire	39
6	Le simulateur TinkerCAD	41
	Bibliographie	49

Glossaire

ADC *Analog to Digital Converter* - Convertisseur Analogique-Numérique.

I2C *Inter-Integrated Circuit (Bus)* - Bus de communication.

LED *Light Emitting Diode* - Diode Electro-Luminescente (DEL).

MOSFET *Metal Oxide Semiconductor Field Effect Transistor* - Transistor à effet de champ à structure métal-oxyde-semiconducteur.


SPI *Serial Peripheral Interface* - Interface Série pour Périphérique.


Liste des figures

1.1	Un signal analogique et numérique	6
1.2	Un signal analogique et numérique échantillonnés	7
1.3	L'échantillonnage d'un signal	9
1.4	Un sous-échantillonnage	10
1.5	Un multiplexeur	11
1.6	Un multiplexeur avec un ADC	12
1.7	Un ADC ADS1115	12
1.8	Un pont diviseur de tension	13
3.1	La représentation du transistor bipolaire	17
3.2	Conventions du transistor bipolaire	18
3.3	Transistors NPN et PNP	18
3.4	Le rôle du transistor	19
3.5	Courant de commande et de puissance	20
3.6	Branchement du transistor bipolaire	21
4.1	La représentation du transistor MOSFET	24
4.2	Transistors à canal N et P	25
4.3	Branchement du transistor MOSFET	27
4.4	Le montage typique d'un transistor MOSFET	29
4.5	Extrait n°1 du IRF520	30
4.6	Extrait n°2 du IRF520	30
4.7	Un relais électro-mécanique	31
4.8	Le symbole du relais	31
4.9	Utilisation d'un relais	32
4.10	Une simulation LTSpice	33
4.11	Une surtension sur le transistor	33
4.12	Une surtension plus faible	34
4.13	Un relai avec une interface de contrôle	34
4.14	Un transistor sans isolation galvanique	35
4.15	Un optocoupleur	35
4.16	Le schéma d'un optocoupleur	36
4.17	L'architecture à base de l'optocoupleur	36
4.18	L'architecture complète	38


6.1	Bouton d'inscription	41
6.2	Choix de création d'un compte	41
6.3	Création d'un compte personnel	42
6.4	Saisie de l'adresse et du mot de passe Tinkercad	42
6.5	Menu latéral	42
6.6	La liste de vos projets et réalisations	43
6.7	On clique sur New Circuit	43
6.8	La page d'édition TinkerCAD	44
6.9	La recherche de composants	44
6.10	Une carte Arduino	45
6.11	Un MOSFET	45
6.12	Ajouter un câble	45
6.13	Ajouter un câble	46
6.14	Notre montage d'exemple	46
6.15	Bouton du code	47
6.16	Code en texte	47
6.17	Confirmation du passage en mode Texte	47
6.18	Code par défaut	48
6.19	Lancement de la simulation	48
6.20	Simulation en cours	49
6.21	Simulation du moteur	49


Préambule

 Document réalisé en \LaTeX par Nicolas Le Guerroué pour le Club de Robotique et d'Électronique Programmable de Ploemeur (CREPP)


 Version du 8 mai 2023

 Taille de police : 11pt (carlito)

 N'hésitez pas à faire des retours sur le document, cela permettra de l'améliorer

 nicolasleguerroue@gmail.com

 <https://github.com/CREPP-PLOEMEUR>¹

 Permission vous est donnée de copier, distribuer et/ou modifier ce document sous quelque forme et de quelque manière que ce soit.

 **Dans la mesure du possible, évitez d'imprimer ce document si ce n'est pas nécessaire. Il est optimisé pour une visualisation sur un ordinateur et contient beaucoup d'images (50 images)**

Conventions










Les commandes à saisir sont dans des encadrés similaires :

```
sudo apt-get update
```

Code 1 - Exemple de commande

Parfois, ces encadrés contiendront des instructions qu'il faudra placer dans certains fichiers.

1. Click-droit et **Copier l'adresse du lien**

- Les fichiers sont indiqués par le repère  fichier
- Les dossiers sont indiqués par le repère  dossier
- Les logiciels sont indiqués par le repère  logiciel²
- Les adresses IP sont indiquées par le repère  Adresse IP
- Les adresses MAC sont indiquées par le repère  Adresse MAC
- Les liens sont indiqués par le repère  Lien
- Les broches génériques des composants sont indiquées par le repère  Broche et se scindent en deux parties :
 - Les broches d'entrée par  Broche
 - Les broches de sortie par  Broche

Versions

octobre 2021	Fusion des supports d'ateliers
novembre 2021	Ajout de l'atelier sur les servomoteurs
décembre 2021	Ajout de l'atelier sur les moteurs pas-à-pas
janvier 2022	Ajout de l'annexe pour l'installation des bibliothèques ESP8266
février 2022	Ajout de l'atelier pour le serveur Web ESP8266 NodeMCU
mars 2022	Ajout de l'atelier pour la partie Réseaux (Adressage) et de l'atelier Capteurs
mai 2022	Ajout de l'atelier sur les modules de communication
janvier 2023	Ajout de l'atelier sur les écrans
mars 2023	Ajout de l'atelier sur les redirections des ports

2. Sont également concernés les paquets Linux et les bibliothèques des langages

Section 1

La conversion des grandeurs

Les Convertisseurs Analogiques-Numériques

Les Convertisseur Analogiques-Numériques, plus familièrement appelés CAN ou ADC¹, permettent de faire la jonction entre le monde physique qui est essentiellement analogique (température, pression...) et le monde des microcontrôleurs qui fonctionne en numérique.



Pour rappel, un signal numérique est un signal qui prend un nombre fini de valeur (0 et 1 typiquement) alors qu'un signal analogique est définie dans un intervalle avec une infinité de valeur entre les bornes.

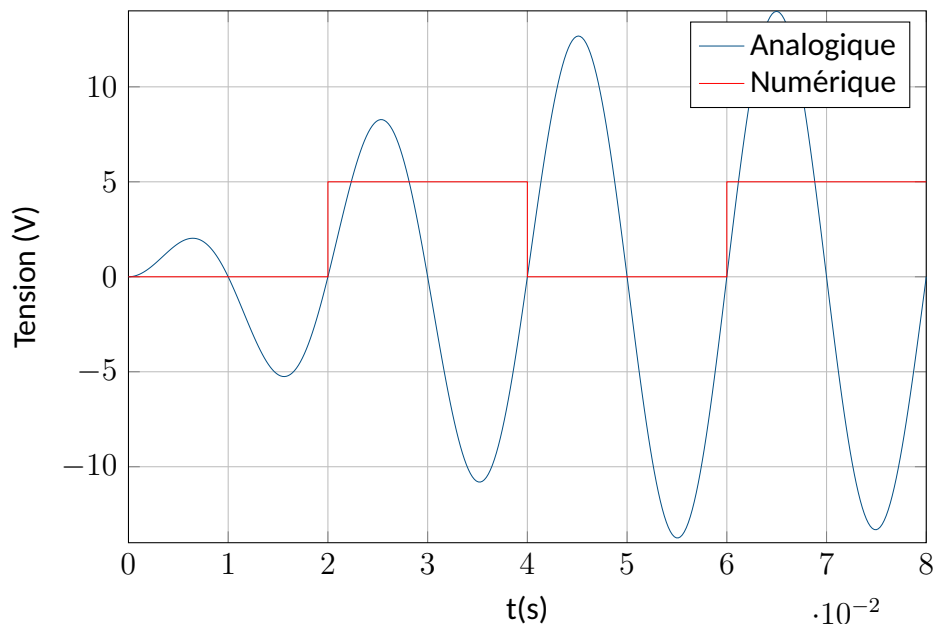


Figure 1.1 – Un signal analogique et numérique

Un ADC va donc devoir échantillonner, c'est à dire prélever une valeur de la grandeur² à un instant t puis numériser la valeur analogique.

1. ADC: *Analog to Digital Converter* - Convertisseur Analogique-Numérique

2. Dans notre cas une tension.

La figure suivante représente les échantillons des deux signaux.

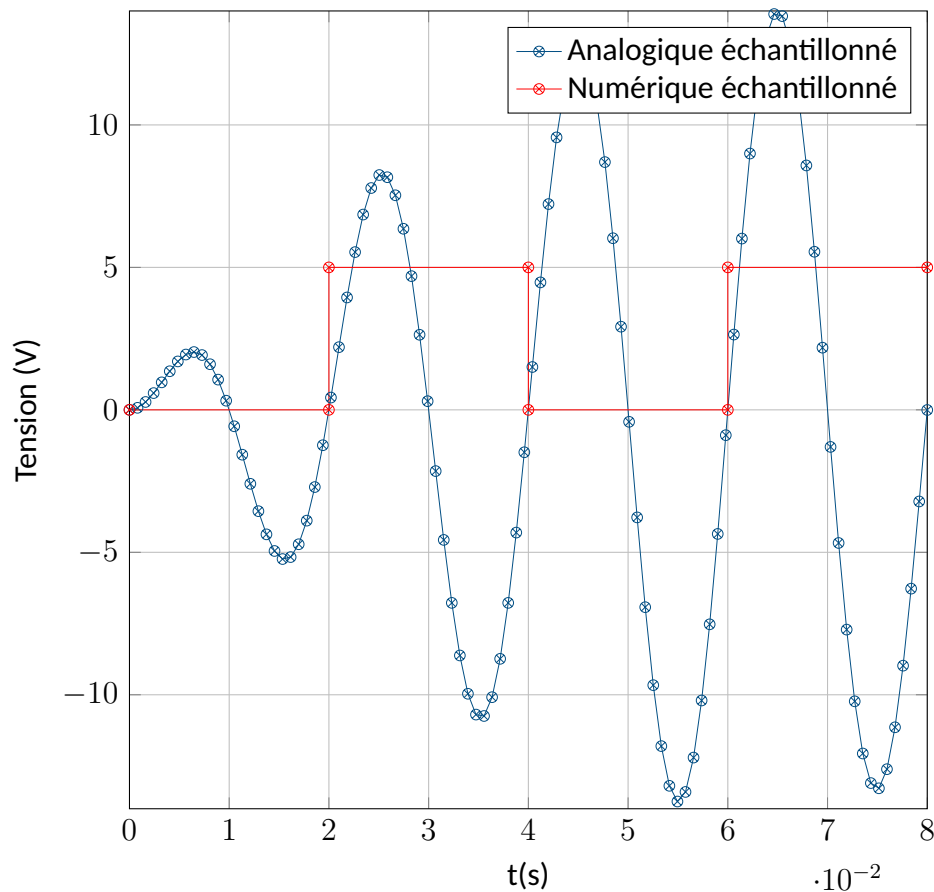


Figure 1.2 – Un signal analogique et numérique échantillonnés

Résolution de l'ADC

Pour notre signal analogique, bien que les valeurs de tension soient infinies, la valeur en sortie de l'ADC est finie et la précision de la valeur lue dépend de la résolution de l'ADC. Cela correspond au nombre de bits sur lequel la valeur est enregistrée.

La précision dépendra de la plage de tension de l'ADC.

Si nous souhaitons obtenir un ADC précis, il faut minimiser l'étendue de la plage de tension et augmenter le nombre de bits de résolution.

Il existe différents types d'ADC, tels que l'ADC à rampe, l'ADC à approximation successive ou encore l'ADC à sigma-delta. Chaque type d'ADC a ses propres caractéristiques et avantages en fonction de l'application pour laquelle il est utilisé.

Considérons un ADC avec une résolution de 8 bits. Cet ADC est alimenté en 5V dans notre exemple.

Avec une résolution de 8 bits, le nombre en sortie de l'ADC est compris entre 0 et 255 ($2^8 - 1$).

Ainsi, l'ADC avec N bits de résolution aura une résolution de tension ΔV valant :

$$\Delta V = \frac{V_{ref}}{2^N - 1} = \frac{5}{2^8 - 1} = 19.6 \text{ mV}$$

Typiquement, une tension de 19 *mV* en entrée de l'ADC sera interprétée avec la valeur 0 car c'est le premier intervalle de tension mesurable. Le tableau suivant représente les valeurs de l'ADC en fonction de la tension en entrée :

Tension (mV)	Valeur ADC
0	0
...	...
18	0
...	...
21	1
...	...
100	5
...	...
5000	255
...	...
6000	255

Table 1.1 – Tableau ADC



On constate que pour les tension en dehors des intervalles de tension de l'ADC, la sortie sera égale à la valeur minimale ou maximale de l'ADC.

Fréquence d'échantillonnage

Sur cette autre figure, nous voyons les différents niveau de tension lus par l'ADC. La période entre deux échantillons permet de déterminer la **fréquence d'échantillonnage**, exprimée en *Hz*.

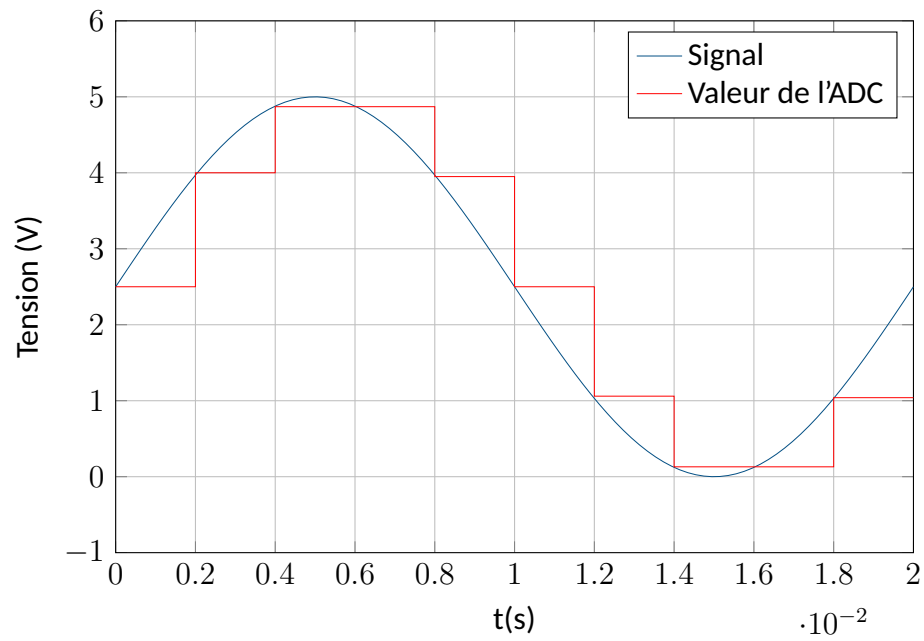


Figure 1.3 – L'échantillonnage d'un signal

Dans ce cas, la fréquence vaut $\frac{1}{T} = \frac{1}{0.002} = 500 \text{ Hz}$

Lors de l'échantillonnage de signaux, il convient de respecter un théorème bien connu : **le théorème de Shannon**.

Ce théorème stipule que la fréquence d'échantillonnage d'un signal doit être égale à au moins deux fois la fréquence maximale du signal.

Ce rapport de 2 entre les 2 fréquences permet de restituer le signal par la suite.

Ainsi, si nous souhaitons échantillonner un signal audio de 15 kHz, il nous faudra faire un échantillonnage d'au moins 30 kHz. La figure suivante représente un signal sous-échantillonné :

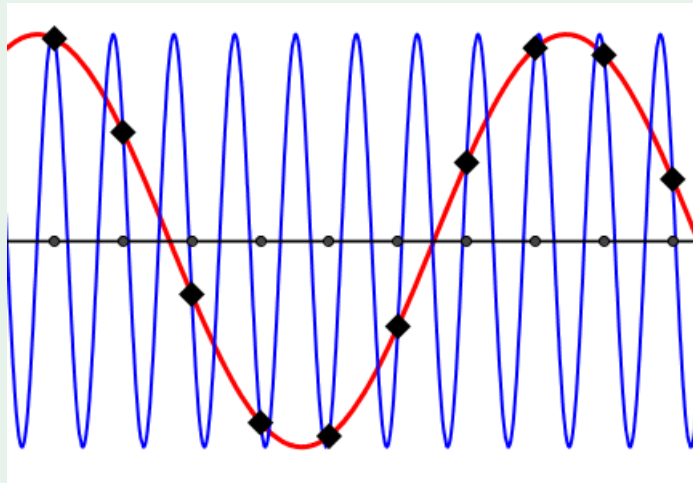


Figure 1.4 – Un sous-échantillonnage

Si nous abaissons la fréquence d'échantillonnage, le nouveau signal reconstruit (rouge) ne représente pas le signal d'origine.

Les cartes Arduino UNO intègrent un CAN avec 6 canaux, c'est à dire que l'on peut mesurer jusqu'à 6 signaux différents.

Les broches utilisées par le CAN sont numérotées de **PIN A0** à **PIN A5**.

Cependant, sur d'autres cartes, comme les ESP12 NodeMCU, il n'y a qu'un CAN avec un seul canal. Si nous souhaitons lire les valeurs de 3 signaux, nous ne pouvons pas le faire.

La section suivante présente quelques solutions pour lire des signaux.

Le multiplexage

Il est possible de créer virtuellement des ADC en faisant du multiplexage, c'est à dire en redirigeant physiquement les signaux.

Un multiplexeur est vu comme un aiguilleur de train : en fonction d'une commande, une seule des entrées est redirigée vers la broche commune :

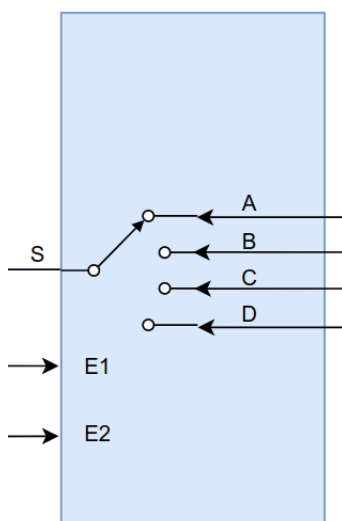


Figure 1.5 – Un multiplexeur

L'entrée qui doit être redirigée est choisie en fonction des valeurs sur l'entrée E1 et E2 :

E1	E2	S
0	0	A
0	1	B
1	0	C
1	1	D

Table 1.2 – Sélection des entrées du multiplexeur



La sélection des entrées en fonction du code sur E1 et E2 n'est pas absolue et dépend du composant choisi. Il a été mis en place pour illustrer le principe du multiplexage.

Avec deux broches numériques, nous pouvons lire quatre signaux avec un seul ADC. Il existe également des multiplexeurs avec 2^N entrées avec N broches de commande.

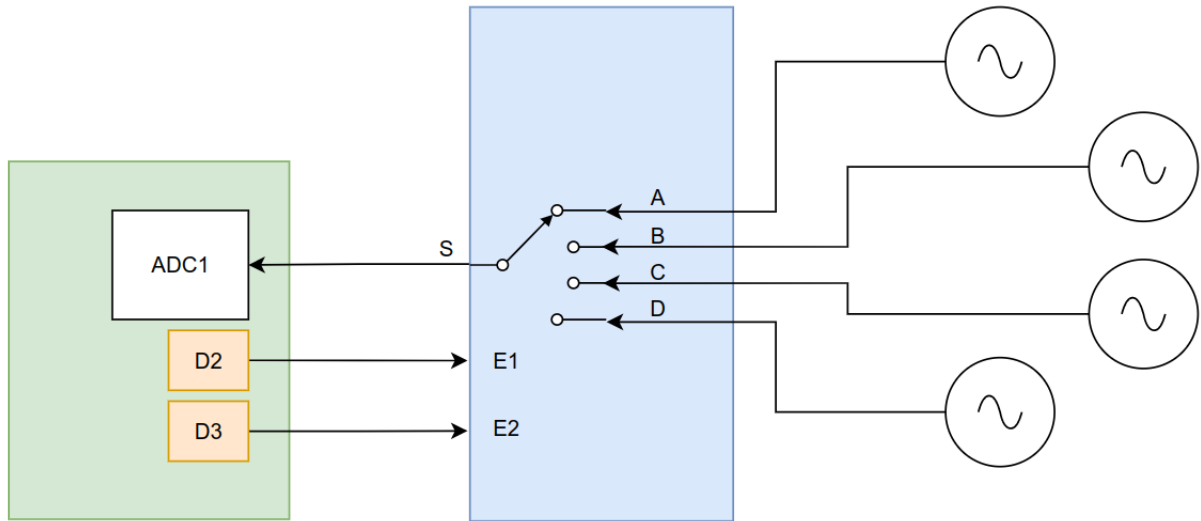


Figure 1.6 – Un multiplexeur avec un ADC

Les ADC externes

Il est également possible d'utiliser des ADC externes à notre carte (ESP, Arduino). Ils existent en circuits dédiés et permettent de lire les signaux en utilisant un protocole de communication standard, généralement de l'I2C³ ou du SPI⁴.

Un ADC assez répandu est l'**ADS1115**.

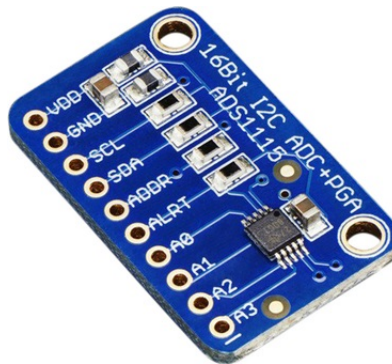


Figure 1.7 – Un ADC ADS1115

Cet ADC possède 4 canaux et sa résolution est de 16 bits, ce qui permet de lire des signaux avec des variations de 0.05 mV ⁵

Protection des ADC

3. I2C: *Inter-Integrated Circuit (Bus)* - Bus de communication

4. SPI: *Serial Peripheral Interface* - Interface Série pour Périphérique

5. Alimenté sous 3.3V

Lors de la lecture des signaux, nous souhaitons que la valeur maximale de notre signal soit inférieure à la valeur maximale de l'ADC.

Supposons que la tension maximale de notre ADC soit de 5V (Typiquement un ADC sur les cartes Arduino). Et que nous souhaitons lire un signal dont la valeur maximale est de 7V. Une protection bien adaptée impose une tension de 5V quand la tension d'entrée est de 7V. Il va falloir faire une adaptation de tension avec un **pont diviseur de tension**.

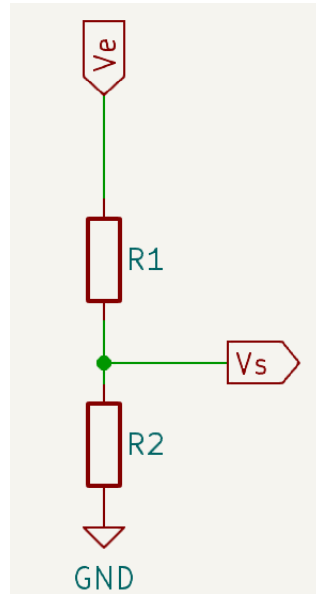


Figure 1.8 – Un pont diviseur de tension

Soit le pont diviseur de tension formé par les deux résistances R_1 et R_2 .

Le pont est alimenté avec la tension V_e .

La tension de sortie V_s va en entrée de l'ADC d'une carte Arduino (pour notre exemple)

La tension en sortie d'un pont diviseur de tension vaut :

$$V_s = V_e \cdot \frac{R_2}{R_1 + R_2}$$

Nous allons donc calculer R_1 après avoir fixé R_2 valant $100\text{ k}\Omega$:

$$R_1 = \frac{R_2 \cdot (V_e - V_s)}{V_s} = \frac{100 \cdot (7 - 5)}{5} = 40\text{ k}\Omega$$

La valeur normalisée la plus proche est $39\text{ k}\Omega$.

Ainsi, en prenant $R_1 = 40\text{ k}\Omega$ et $R_2 = 100\text{ k}\Omega$ et une tension $V_e = 7\text{ V}$, nous avons une tension en sortie de pont valant 5 V .

Question 1. Et quel est l'ordre de grandeur de la valeur de résistance à choisir

>>> **1.** On prend généralement des valeurs comprises entre $10\text{ k}\Omega$ et $100\text{ k}\Omega$, cela évite de consommer trop de courant et cela permet de rester en dessous de l'impédance d'entrée⁶ des ADC.

6. Résistance en entrée de l'ADC

Questionnaire

Quiz

1. Un ADC permet de :

Générer une tension
analogique

Lire une tension et la
convertir en un
nombre fini

Filtrer un signal
numérique

2. A quelle fréquence devons-nous au minimum échantillonner un signal de 25 kHz ?

12.5 kHz
250 kHz

25 kHz

50 kHz

100 kHz

3. La résolution d'un ADC dépend de :

La fréquence
d'échantillonnage

Du nombre de bits
de l'ADC

La plage de tension
de l'ADC

La consommation
électrique de l'ADC

4. Quelle structure permet de protéger les ADC :

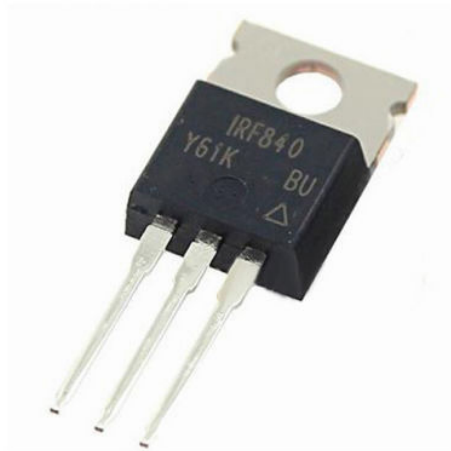
Un pont de Graëtz

Un pont diviseur de
tension

Un pont de Wien

Première partie

Les interfaces de puissance



Section 2

Introduction

Pour certains projets plus évolués, on souhaite utiliser des composants tels que des moteurs ou des résistances (chauffage, ventilation).

Or, on constate rapidement que le branchement direct de ces éléments sur une carte Arduino va se révéler impossible.

En effet, la carte Arduino est prévue pour délivrer de **faibles courants** et **faibles tensions** . Nous allons donc créer un circuit où la puissance et la commande sont dissociés.

Section 3

Les transistors bipolaires

Présentation

Une des moyens pour créer notre circuit de puissance est le transistor bipolaire. . Ce composant possède trois broches :

- ▶ Le collecteur (C)
- ▶ la base (B)
- ▶ l'émetteur (E)

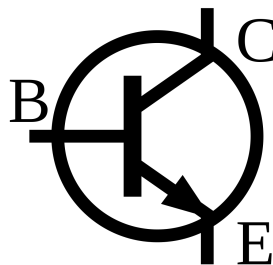


Figure 3.1 – La représentation du transistor bipolaire

Conventions

Afin de simplifier les calculs par la suite, posons les normes suivantes :

- ▶ Le courant entrant dans le Collecteur est appelé I_C
- ▶ Le courant entrant dans la Base est appelé I_B
- ▶ Le courant sortant de l'émetteur est appelé I_E
- ▶ La tension entre la Base et l'Émetteur est appelée V_{be}

- La tension entre le Collecteur et l'Émetteur est appelée V_{ce}

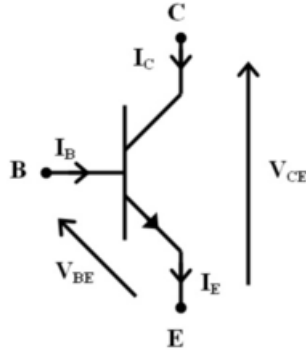


Figure 3.2 – Conventions du transistor bipolaire

Les flèches au sein du transistor indiquent le sens de déplacement du courant sur les broches.

Les familles de transistors bipolaires

Les transistors bipolaires sont classés en deux catégories :

- Les transistors NPN¹
- Les transistors PNP

Le principe de fonctionnement est similaire entre ces deux familles, seul le branchement et le niveau de commande diffère.

Dans ce document, nous utiliserons essentiellement des transistors NPN car ces derniers utilisent des grandeurs positives.

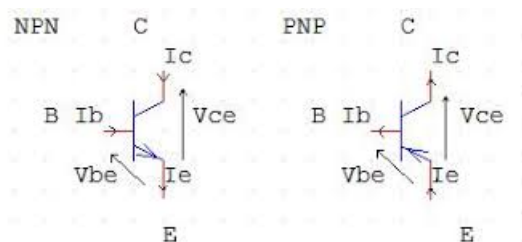


Figure 3.3 – Transistors NPN et PNP

Les paramètres de sélection du transistor

Notre transistor doit dans un premier temps répondre à deux contraintes :

1. Le nom de ces familles provient du type de jonction utilisé en interne. Pour plus de renseignements, consulter les diodes et semi-conducteurs

- ▶ La tension admissible sur V_{ce}^2 doit être supérieure à la tension d'alimentation de notre circuit. Concrètement, si notre circuit est alimenté en 48V mais que le transistor ne supporte pas plus de 30V, il va être détruit.
- ▶ Le transistor doit supporter un courant plus élevé que le courant maximal transitant dans notre circuit. Pour contrôler un moteur consommant 1 Ampère, je dois donc choisir un transistor pouvant contrôler au moins 2 Ampère.

Pour la suite de la présentation, on supposera que notre transistor a été dimensionné pour répondre à ces deux contraintes.

Le principe

Ce type de transistor fonctionne comme une vanne pour une canalisation. Il est possible de réguler le débit de la canalisation avec la vanne.

Le transistor bipolaire permet de contrôler un courant important avec un faible courant.

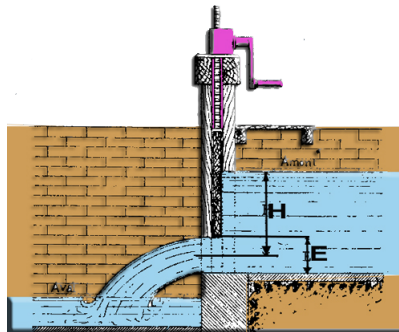


Figure 3.4 – Le rôle du transistor

Ici, notre transistor joue le rôle de la vanne et permet de bloquer le courant (électrons) ou bien de les laisser passer.

Le courant de l'élément à contrôler (moteur, résistance de puissance) transite entre le collecteur et l'émetteur et le courant de commande passe par la base, comme l'illustre la figure suivante.

2. Cette tension est indiquée dans les documentations techniques

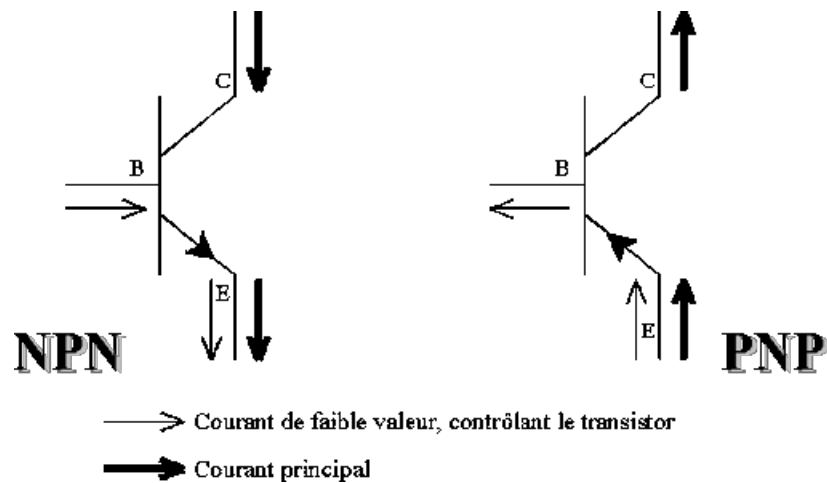


Figure 3.5 – Courant de commande et de puissance

La relation fondamentale reliant le courant de puissance et de commande est la suivante :

$$I_C = \beta \cdot I_B$$

Le paramètre β , appelé **gain du transistor**³ est une caractéristique interne de notre transistor, c'est à dire qu'il dépend du type de transistor que nous choisissons.

Les courants I_C, I_B, I_E sont exprimés dans la même unité (ampère, milliampères..) pour une formule homogène.

Les transistors de puissance possèdent des gains de l'ordre de la dizaine alors que les transistors de signal (faibles courants) ont un gain pouvant facilement atteindre 200 ou 300.



Plus notre β est faible, plus il va falloir injecter un courant important dans notre base.

Question 2. Et que devient notre broche "Émetteur" ?

>>> **2.** Notre émetteur est relié à la masse du circuit et permet de le fermer pour que les électrons puissent circuler.

Le courant circulant dans l'émetteur est simplement la somme des courants entrant dans le transistor. d'où :

$$I_E = I_B + I_C$$

Exemple

On souhaite commander l'arrêt et la marche d'un moteur consommant au maximum 0.5A et alimenté avec une tension de 9V.

Nous choisissons un transistor permettant de commuter 1A (sécurité) avec $\beta = 30$

3. le gain est sans dimension (unité) et est appelé h_{fe} dans les documentations

Question 3. Quel doit-être le courant injecté dans la base ?

>>> 3. On applique la formule précédent et on obtient :

$$I_B = \frac{I_C}{\beta} = \frac{0.5}{30} = 16mA$$

Mise en pratique

Branchements

Maintenant que nous connaissons les tensions et courants nécessaires à notre transistor et à notre moteur, nous allons le commander avec une carte Arduino.

Tout d'abord, il convient de placer le moteur entre notre alimentation et le collecteur.



Toutes les charges à contrôler avec ce type de transistor se placent entre l'alimentation et le collecteur.

Enfin, il ne nous reste plus qu'à relier une sortie numérique de l'Arduino vers notre base par l'intermédiaire d'une résistance.

La résistance va servir à imposer le courant dans la base de notre transistor .

Nous obtenons donc le schéma suivant.

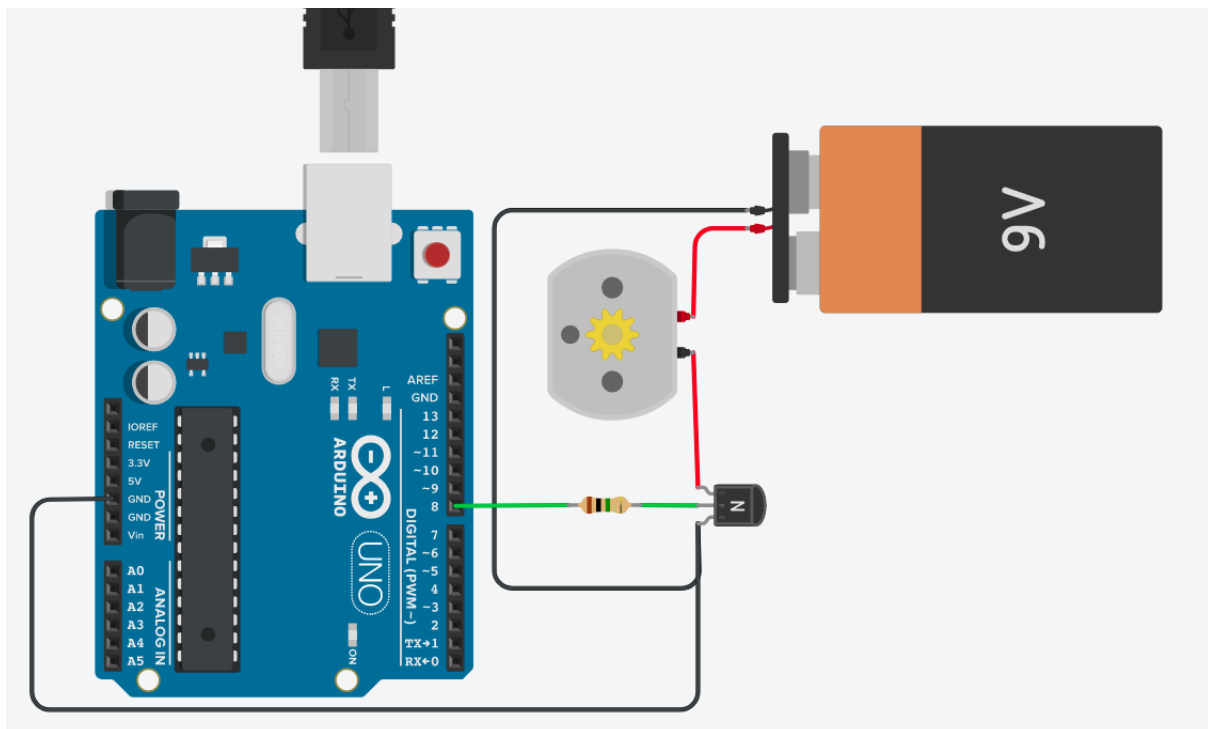


Figure 3.6 – Branchement du transistor bipolaire

Dimensionnement de la résistance

On souhaite obtenir un courant de $16mA$ dans notre base et on sait que l'Arduino délivre du $5V$ en sortie.

$$\text{Nous sommes donc tentés de dire que } R_b = \frac{U_{arduino}}{I_B} = \frac{5}{0.016} = 312\Omega^4$$

Hélas, il y a peu de chance que votre moteur tourne dans les conditions optimales. Il convient d'avoir à l'esprit que notre β trouvé dans la documentation n'est que théorique et qu'il peut être en réalité inférieur.



Une des conventions non officielles admet que pour de la commutation en Tout ou Rien, on divise la valeur théorique de notre β par 2. Nous allons donc prendre donc un β valant 15.

On refait donc les calculs.

$$I_B = \frac{I_C}{\beta} = \frac{0.5}{15} = 32mA$$



On remarquera que la carte Arduino doit fournir un courant qui s'approche de la limite de courant pour une seule broche, à savoir 40 mA !

Une dernière chose : les transistors bipolaires entraînent une chute de tension entre la base et l'émetteur (V_{be}).

Cette chute de tension dépend de la technologie des transistors bipolaires :

- ▶ $0.7V$ pour les transistors au silicium
- ▶ $0.3V$ pour les transistors au germanium

Dans l'extrême majorité des cas, on utilisera des transistors au silicium. La tension disponible aux bornes de la résistance est donc de $4.3V$ ($5 - 0.7$)

D'où :

$$R_b = \frac{U_{arduino} - V_{be}}{I_b} = \frac{4.3}{0.032} = 134\Omega$$

Exemple de programme Arduino

Voici un code permettant de faire tourner le moteur périodiquement pendant 5 secondes puis de l'arrêter pendant 5 secondes.

```
#define D8 8 //Broche 8 de l'Arduino

void setup() {
```

4. On part de la loi de Ohm qui dit que $U = R.I$

```
pinMode(D8, OUTPUT); //Mise en sortie de la broche

} //End setup

void loop() {

    digitalWrite(D8, HIGH);    //Déclencher la rotation du moteur
    delay(5000);               //Délai de 5s
    digitalWrite(D8, LOW);    //Fin de la rotation du moteur
    delay(5000);               //Délai de 5s

} //End loop
```

Code 2 - Code Arduino avec transistors NPN

Section 4

Les transistors MOSFET

Présentation

Nous avons vu l'utilisation des transistors bipolaires.

Ces derniers sont assez contraignants à mettre en oeuvre car ils sont commandés en courant.

Nous allons utiliser cette fois-ci la technologie des MOSFET¹ car ces derniers ont l'avantage d'être contrôlés en **tension**.

Ce composant possède trois broches :

- ▶ Le drain (D)
- ▶ la porte (G)²
- ▶ la source (S)

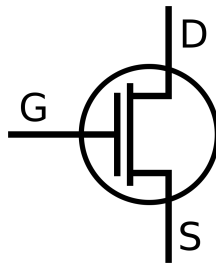


Figure 4.1 – La représentation du transistor MOSFET

Conventions

Afin de simplifier les calculs par la suite, posons également les normes suivantes :

1. MOSFET: *Metal Oxide Semiconductor Field Effect Transistor* - Transistor à effet de champ à structure métal-oxyde-semiconducteur

2. G pour Gate

- ▶ Le courant entrant dans le Drain est appelé I_D
- ▶ Le courant entrant dans la Porte est appelé I_G
- ▶ Le courant sortant de la Source est appelé I_S
- ▶ La tension entre la Porte et la Source est appelée V_{GS}
- ▶ La tension entre le Drain et la Source est appelée V_{DS}

Les familles de transistors MOSFET

Les transistors MOSFET sont classés en deux catégories :

- ▶ Les transistors MOSFET à canal N³
- ▶ Les transistors MOSFET à canal P

Le principe de fonctionnement est similaire entre ces deux familles, seul le branchement et le niveau de commande diffère.

Dans ce document, nous utiliserons essentiellement des transistors MOSFET à canal N car ces derniers utilisent des grandeurs positives.

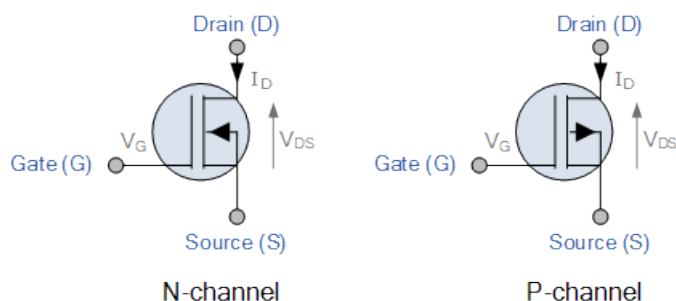


Figure 4.2 – Transistors à canal N et P

Les paramètres de sélection du transistor

Les paramètres de sélection de nos transistors MOSFET sont identiques aux transistors bipolaires, c'est à dire :

- ▶ La tension admissible sur V_{DS} du transistor
- ▶ Le courant admissible entre le Drain et la Source.

Pour la suite de la présentation, on supposera que notre transistor a été dimensionné pour répondre à ces deux contraintes.

3. Le nom de ces familles provient du type de jonction utilisé en interne. Pour plus de renseignement, consulter les diodes et semi-conducteurs

Le principe

Ce type de transistor fonctionne comme les transistors bipolaires mais est commandé en tension et non en courant.

Par analogie, le drain joue le rôle du collecteur, la source celui de l'émetteur et la porte celui de la base.

Le courant de l'élément à contrôler (moteur, résistance de puissance) transite entre le drain et la source et la tension de commande est aux bornes de la porte.

Les transistors MOSFET deviennent passant⁴ lorsque la tension sur la porte dépasse une tension de déclenchement appelée $V_{GS_{th}}$. Cette valeur est généralement comprise entre 2 et 4 Volts.

Lorsque cette tension $V_{GS_{th}}$ est atteinte, notre transistor peut être remplacé d'un point de vue électrique entre le drain et la source par une résistance de très faible valeur, appelée $R_{DS_{on}}$.

Comparaison avec les transistors bipolaires

Par nature, la porte du MOSFET est vue comme un condensateur. Le transistor ne consomme pas de courant, excepté pendant les commutations.

Ainsi, le courant est nul dans la porte pour maintenir le moteur en marche alors que pour un bipolaire, il faut maintenir un courant dans la base.

Les MOSFET sont donc plus économes en énergie que les bipolaires.
De plus, ils peuvent généralement supporter des courants plus importants que les bipolaires.

En revanche, en hautes fréquences, les MOSFET sont moins réactifs du fait de leur capacité en entrée.

Mise en pratique

Nous souhaitons faire tourner le même moteur que celui utilisé avec notre transistor bipolaire. Nous allons le commander avec une carte Arduino.

Branchements



Pour connaître le branchement du transistor, il suffit de consulter sa fiche technique (datasheet). Par exemple, la datasheet du MOSFET représente le branchement de la façon suivante :

Tout d'abord, il convient de placer le moteur entre notre alimentation et le drain (D).

4. Le transistor laisse passer tout le courant autorisé.

Enfin, il ne nous reste plus qu'à relier une sortie numérique de l'Arduino vers notre porte **sans** résistance (Broche G). Nous obtenons donc le schéma suivant.

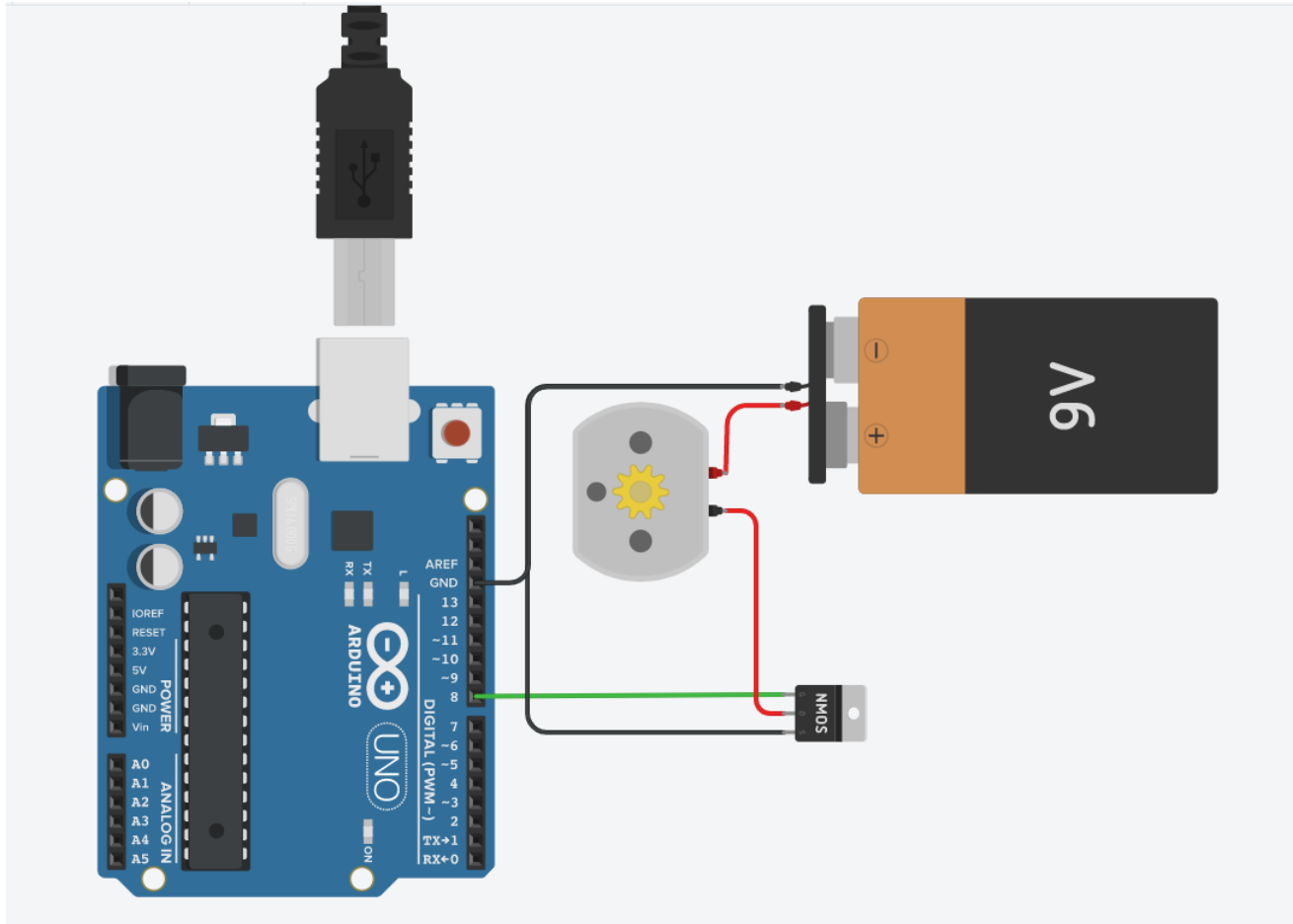


Figure 4.3 – Branchement du transistor MOSFET



Toutes les charges à contrôler avec ce type de transistor se placent entre l'alimentation et le drain.

Exemple de programme Arduino

Voici un code permettant de faire tourner le moteur périodiquement pendant 5 secondes puis de l'arrêter pendant 5 secondes. Il s'agit du même code que pour le transistor bipolaire.

```
#define PIN 8      //GATE du transistor

void setup() {

    pinMode(PIN, OUTPUT); //Mise en sortie de la broche
```

```
//Fin setup

void loop() {

    digitalWrite(PIN, HIGH);    //Mise en route du transistor
    delay(5000);                //Délai de 5s
    digitalWrite(PIN, LOW);     //Arret du transistor
    delay(5000);                //Délai de 5s

} //Fin loop
```

Code 3 - Code Arduino avec MOSFET

Pour aller plus loin

En pratique, on rajoute une résistance entre la Gate et la source des MOSFET.

Pourquoi donc ? Tout simplement parce que la broche Gate du MOSFET possède une capacité⁵ entre la Gate et la Source.

Cela engendre plusieurs problèmes :

- ▶ Le MOSFET peut s'activer avec un simple toucher de la main sur la Gate.
- ▶ Lorsque les broches de contrôles sont mises en haute-impédance⁶, la grille du MOSFET reste au potentiel haut et le MOSFET reste passant.

La résistance entre la Gate et la Source permet donc de décharger le condensateur parasite afin de ramener au potentiel BAS.

- ▶ Cette capacité parasite réduit le temps de commutation du transistor

Nous réalisons donc le montage suivant :

5. c'est à dire un condensateur

6. physiquement, la broche n'est ni à un potentiel HAUT ni BAS, elle est vu comme si elle était déconnecté du circuit

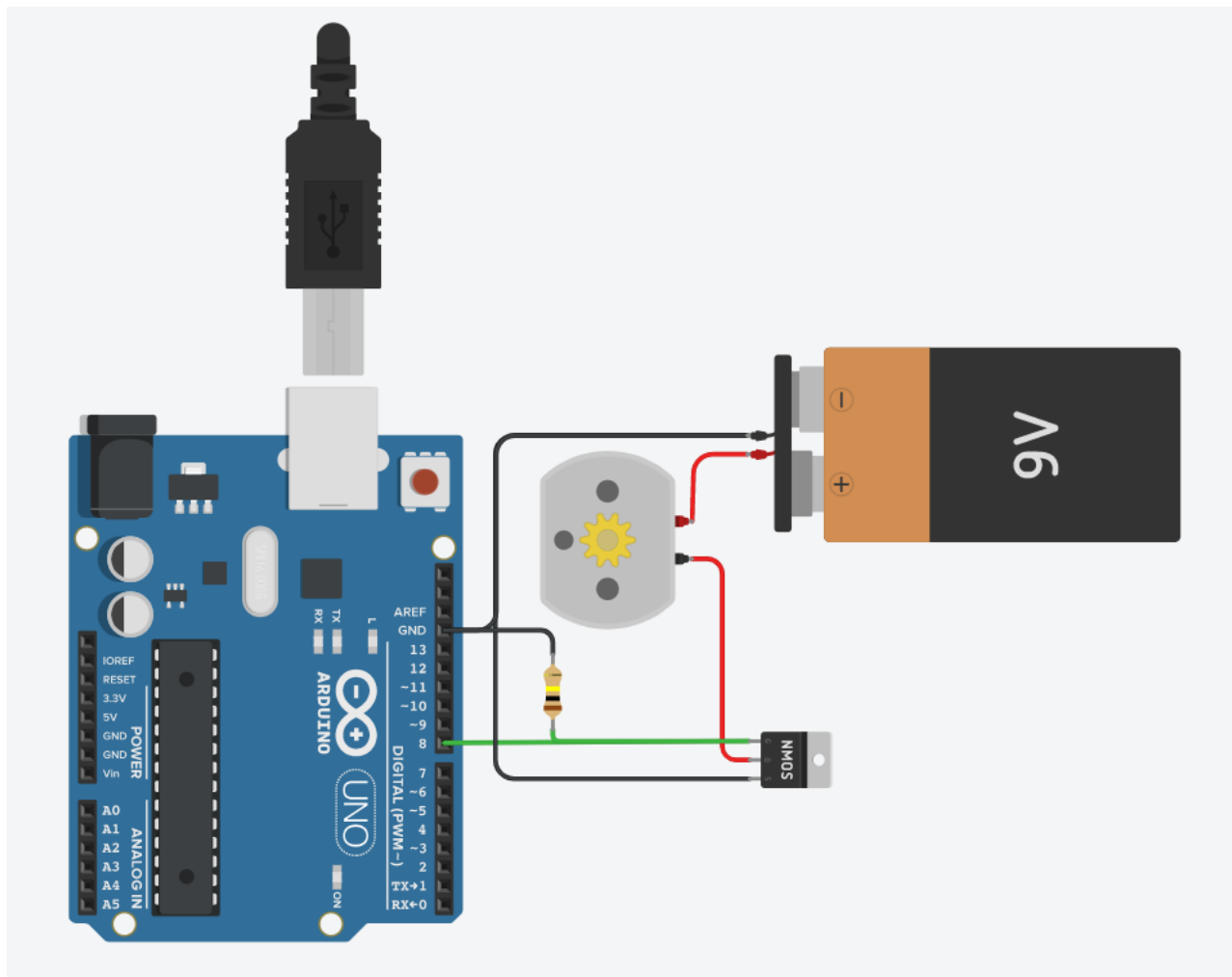


Figure 4.4 – Le montage typique d'un transistor MOSFET

Une valeur souvent utilisée est $100\text{ k}\Omega$.

Les fiches techniques

L'intégralité des informations disponibles pour un transistor sont disponibles dans un document complet appelé **Datasheet**.

Ce document détaille les broches, les caractéristiques électriques, propose des schémas d'exemples....

Par exemple, voici quelques extraits de la documentation du transistors IRF520⁷ :

7. Transistor de puissance

IRF520NPbF

HEXFET® Power MOSFET

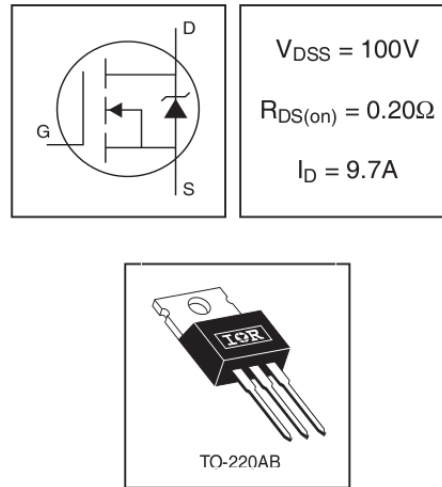


Figure 4.5 – Extrait n°1 du IRF520

Electrical Characteristics @ $T_J = 25^\circ\text{C}$ (unless otherwise specified)

	Parameter	Min.	Typ.	Max.	Units	Conditions
$V_{(BR)DSS}$	Drain-to-Source Breakdown Voltage	100	—	—	V	$V_{GS} = 0V, I_D = 250\mu A$
$\Delta V_{(BR)DSS}/\Delta T_J$	Breakdown Voltage Temp. Coefficient	—	0.11	—	V/°C	Reference to 25°C , $I_D = 1mA$
$R_{DS(on)}$	Static Drain-to-Source On-Resistance	—	—	0.20	Ω	$V_{GS} = 10V, I_D = 5.7A$ ④
$V_{GS(th)}$	Gate Threshold Voltage	2.0	—	4.0	V	$V_{DS} = V_{GS}, I_D = 250\mu A$

Figure 4.6 – Extrait n°2 du IRF520

On retrouve sur cette figure la valeur de $R_{DS_{on}}$ et de $V_{GS_{th}}$

Les relais électromagnétiques

Principe

Les relais sont des interrupteurs commandés électriquement. Une bobine alimentée sous faible tension (5 à 24 V) génère un champ électromagnétique qui fait déplacer une membrane qui va ouvrir ou fermer le circuit.

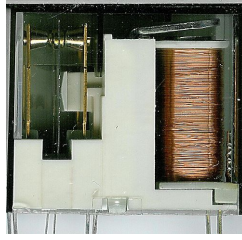


Figure 4.7 – Un relais électro-mécanique

Le relais offre une isolation galvanique entre le circuit de commande et de puissance, c'est à dire qu'il n'y a aucune liaison conductrice entre ces deux circuits.

Un relais est caractérisé par :

- ▶ La tension de commande : 5 à 24V
- ▶ Le courant de coupure : Ex 30A
- ▶ La tension maximale dans le circuit de puissance : Ex 230V
- ▶ Sa position au repos⁸ : Normalement Ouvert (NO) ou Normalement Fermé (NF)
- ▶ Sa durée de vie : les relais sont garantis pour un nombre de commutation, par exemple 1 million.

Symbole

Le relais se symbolise de la façon suivante :

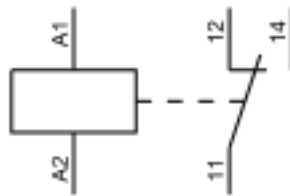


Figure 4.8 – Le symbole du relais

On distingue clairement la partie de commande (rectangle) et le circuit qui s'ouvre ou se ferme pour laisser passer le courant.

La diode de roue libre

La bobine de commande du relais nécessite un certain courant⁹ qu'une broche de microcontrôleur ne peut pas fournir. Pour cela on utilise un composant qui fera l'interface entre le microcontrôleur et le relais : le transistor.

8. Position en absence de tension sur la commande

9. De l'ordre de la centaine de mA

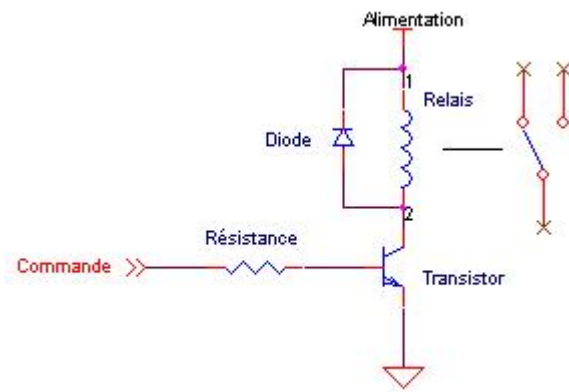


Figure 4.9 – Utilisation d'un relais

Cependant, lors de la fermeture du relais¹⁰, le courant est brutalement coupé, or, les bobines s'opposent aux variations de courant.

Cela engendre une surtension qui va se répercuter sur le transistor.

Cette surtension vaut :


$$U = L \cdot \frac{dI}{dt}$$

Avec :

- ▶ U la tension en Volt aux bornes de la bobine
- ▶ L la valeur en Henry de l'inductance¹¹
- ▶ I la variation de courant en Ampère dans la bobine

On met donc une diode en parallèle de la bobine pour que l'énergie accumulée dans la bobine passe dans la diode.

Cette diode est appelé **diode de roue libre**.

En exemple, une simulation sans diode est faite avec  LTSpice :

¹⁰. Mise à 0 de la broche de commande de transistor

¹¹. Bobine

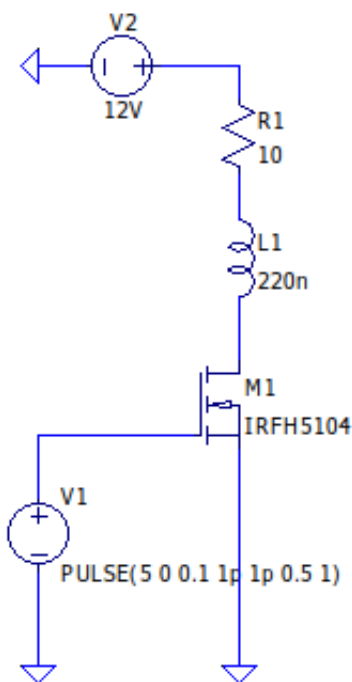


Figure 4.10 – Une simulation LTSpice

On active le transistor pendant 100 ms puis on le désactive et on observe la tension aux bornes du transistor.



Figure 4.11 – Une surtension sur le transistor

On constate un pic à 24V, c'est à dire le double de l'alimentation 12V.

Maintenant, faisons la même simulation avec une diode de roue libre.
Pour ces diodes, on privilégie des diodes Schottky à commutation rapide.



Figure 4.12 – Une surtension plus faible

La surtension ne vaut plus que quelques dizaines de mV.

En pratique

Pour utiliser des relais avec des microcontrôleurs, on utilise le plus souvent des relais qui intègrent une interface de contrôle.

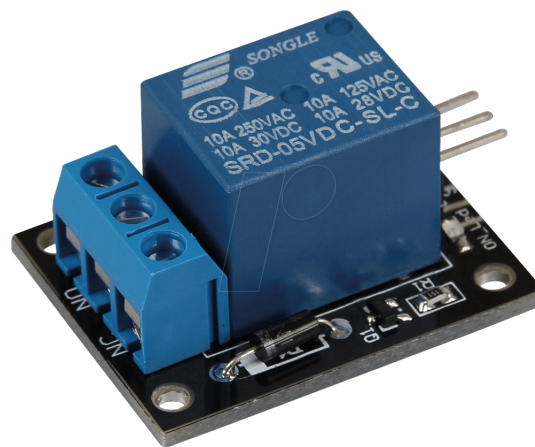


Figure 4.13 – Un relais avec une interface de contrôle

Il suffit généralement d'alimenter le relais en 5V constant et une broche active le relais si elle passe au niveau logique HAUT.

L'isolation galvanique

Nous avons vu qu'avec un relais, nous avons une isolation entre l'interface de puissance et de commande.

Cependant, lorsque nous utilisons un transistor, la masse est commune et cela pose un problème.

Considérons le circuit suivant comportant un transistor commandé par un signal V_e .

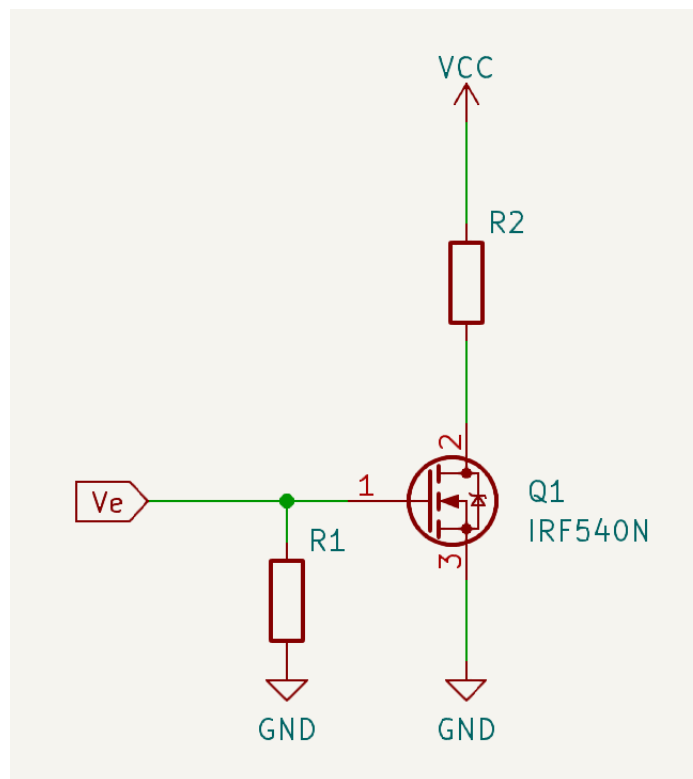


Figure 4.14 – Un transistor sans isolation galvanique

Si par malheur notre transistor est grillé, il y a un risque pour que le circuit de puissance soit injecté dans le circuit de commande entre le drain et la gate du transistor.

La tension **VCC** peut aller dans le signal **Ve** et abîmer le circuit en amont. Pour cela nous voulons donc ajouter une isolation. Il existe un composant qui permet de le faire : **l'optocoupleur**.



Figure 4.15 – Un optocoupleur



Le principe est d'isoler les 2 circuits avec, non pas un courant qui fait l'interface mais une onde lumineuse

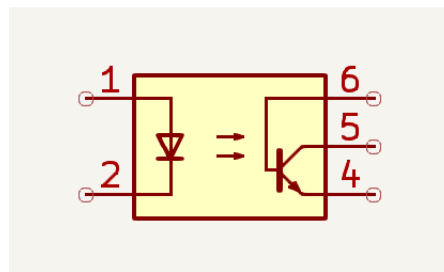


Figure 4.16 – Le schéma d'un optocoupleur

Nous pouvons scinder l'optocoupleur en 2 parties :

- Un émetteur
- Un phototransistor

Du côté de l'émetteur, nous avons une LED¹² généralement dans le domaine de l'infrarouge.

De l'autre côté, un phototransistor qui se comporte comme un transistor lorsqu'il est soumis à un rayon lumineux. Le rayon lumineux rend passant le courant entre le collecteur et l'émetteur du transistor.



Dans notre application, nous prendrons un optocoupleur 4N35 ainsi qu'un transistor IRF540.

Le schéma de principe d'utilisation de l'optocoupleur utilisé avec un transistor est le suivant :

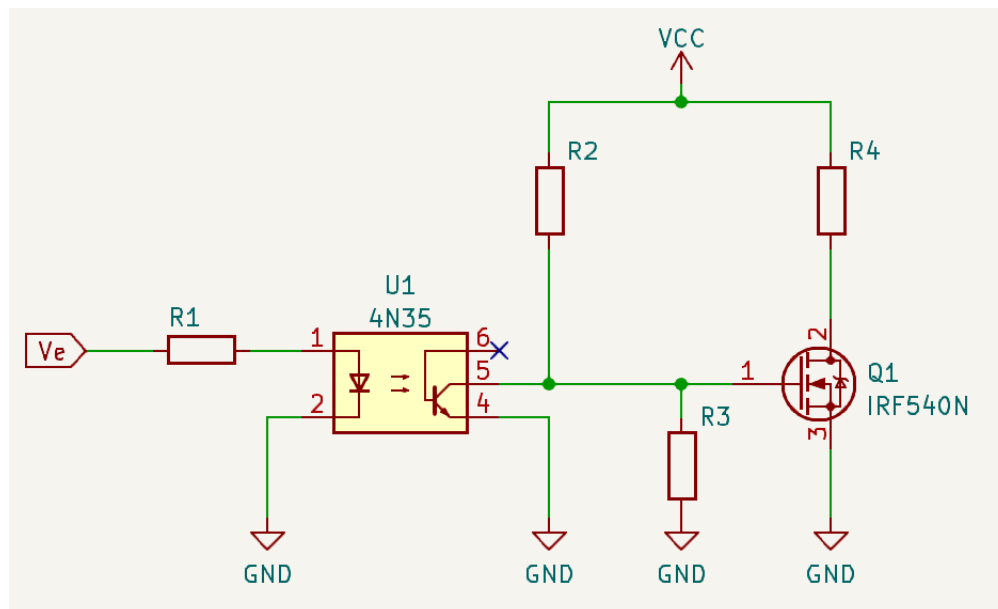


Figure 4.17 – L'architecture à base de l'optocoupleur

Nous allons donc réaliser le dimensionner de cette interface.

12. LED: *Light Emitting Diode* - Diode Electro-Luminescente (DEL)

La documentation nous indique que le phototransistor laisse passer un courant de 4 mA quand un courant de 10 mA parcourt la LED.

Nous allons donc dimensionner les résistances de telle sorte que ces courants soient respectés

Commençons par le courant dans la LED. Il s'agit d'une loi d'Ohm mais il nous faut connaître la chute de tension aux bornes de la LED : la documentation nous indique une chute de tension entre 0.8 et 1.5 V.

Nous prendrons la valeur la plus critique, à savoir 1.5V.

$$\text{Avec } I_{LED} = 10 \text{ mA}$$

$$R_1 = \frac{V_e - V_d}{I_{LED}} = \frac{5 - 1.5}{10 \cdot 10^{-3}} = 350\Omega$$

Nous prendrons la valeur normalisée la plus proche au dessus, à savoir 390Ω

Nous allons ensuite dimensionner la résistance entre notre alimentation **VCC** et le collecteur de notre optocoupleur.

Il nous faut connaître le courant maximal : la datasheet nous indique un courant de 100 mA, nous prendrons 10% de cette valeur.¹³

Maintenant que nous avons notre valeur de courant (10 mA), nous pouvons déterminer la résistance :

$$R_2 = \frac{V_{CC}}{I_{collecteur}} = \frac{12}{0.01} = 1200\Omega$$

Il ne nous reste plus qu'à fixer la valeur de notre résistance R_3 .

Il nous faut connaître cette fois ci la tension maximale et minimale de commutation de notre transistor MOS : la tension V_{GStH}

Dans notre cas, elle vaut minimum 2.5V et au maximum, elle est de 20V. Nous pouvons donc considérer que le transistor sera totalement passant à partir de 5V. La résistance R_2 et R_3 forment un pont diviseur de tension sur la grille du MOSFET, ainsi :

$$V_{GS} = \frac{R_3}{R_2 + R_3} \cdot V_{CC}$$

En fixant R_2 à 1.2kΩ et R_3 à 1kΩ, nous obtenons une tension V_{GS} valant 5.45V¹⁴ :

$$V_{GS} = \frac{1}{1 + 1.2} \cdot 12 = 5.45V$$

La résistance R_4 représente quant à elle la charge que commande le MOSFET (relais, moteur...) : Nous obtenons donc le schéma suivant :

13. Il ne vaut pas prendre une valeur de courant trop faible (< 5% sous peine de voir la tension de déchet V_{CE} augmenter)

14. Puisque le pont diviseur de tension représente un rapport de résistance, nous pouvons faire le calcul en kΩ, le résultat sera valide.

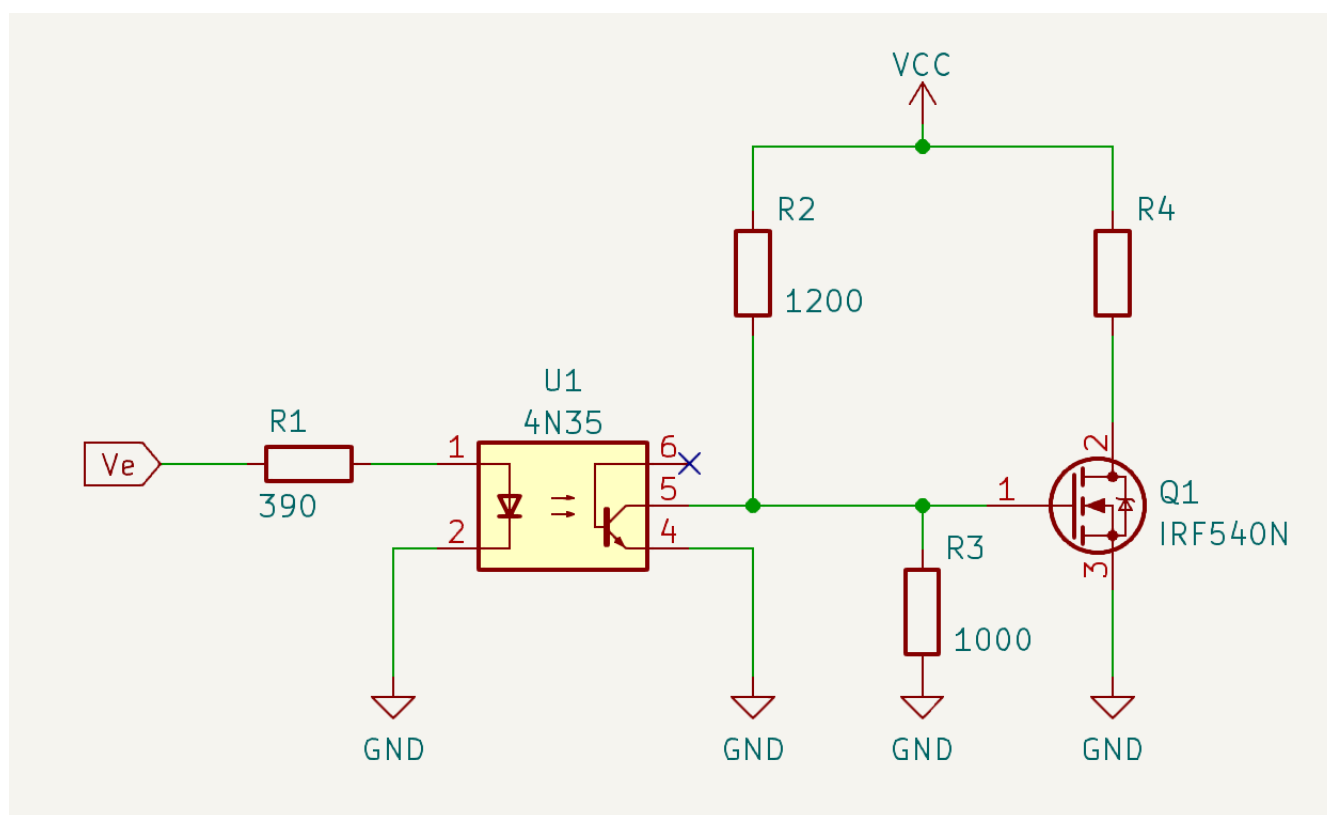


Figure 4.18 – L'architecture complète

Section 5

Conclusion

Ce qu'il faut retenir

Nous avons à notre disposition tout un ensemble de technologies pour contrôler la partie puissance.

Les transistors ne sont pas adaptés pour commuter une charge sur secteur (230V), cette partie sera donc réservée aux relais.

En revanche, pour toutes les tensions continues, les transistors sont adaptés et prennent moins de place en encombrement.

Questionnaire

Quiz

1. Les transistors MOSFET sont contrôlés avec une certaine valeur de :

Courant

Tension

2. Les transistors bipolaires sont contrôlés avec une certaine valeur de :

Courant

Tension

3. Quel composant fournit une isolation galvanique ?

La diode

Le triac

Le transistor
bipolaire

Le relais

Le transistor MOSFET

L'optocoupleur

4. Les transistors MOSFET, lorsqu'ils sont passant, sont vus comme :

Une résistance de
faible valeur

Un condensateur de
valeur élevée

Une bobine de valeur
élevée

5. Pour un transistor MOSFET, comment s'appelle la tension à partir de laquelle le transistor devient passant ? :

 V_{ON} $V_{GS_{th}}$ V_{DS}

6. Les diodes de roues libres permettent de :

Faire tourner les
moteurs dans un seul
sens

Augmenter la vitesse
de commutation de
l'organe de
commande
(transistor)

Protéger le circuit de
commutation des
surtensions

Section 6

Le simulateur TinkercAD

L'objectif de ce chapitre est de présenter un outil de simulation Arduino : TinkercAD.

Le simulateur est disponible à l'adresse suivante : <https://www.tinkercad.com>

Il vous faudra tout d'abord créer un compte en cliquant en haut à droite sur [Sign Up](#)¹ :

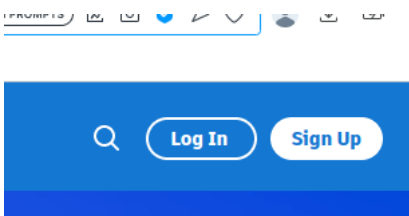


Figure 6.1 – Bouton d'inscription

Veuillez cliquer sur [Create a personal account](#)

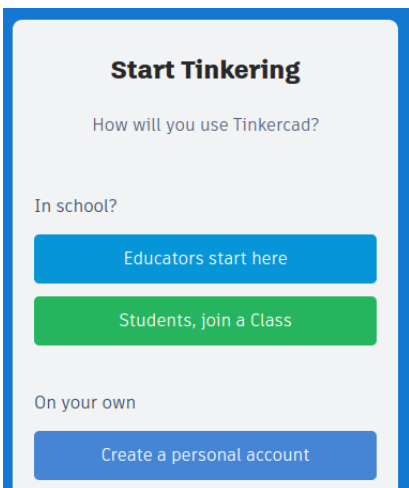
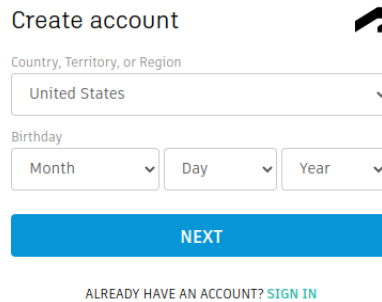


Figure 6.2 – Choix de création d'un compte

Il vous faudra ensuite renseigner vos informations personnelles puis faire [Next](#) .

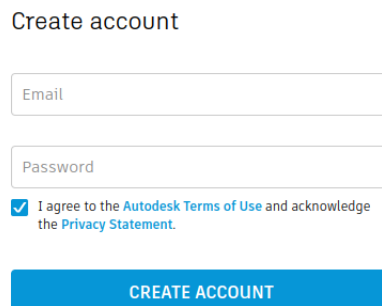
1. Ou bien [Inscription](#)



The screenshot shows the 'Create account' page on the Tinkercad website. At the top, there's a title 'Create account' and a small Tinkercad logo. Below the title, there's a dropdown menu for 'Country, Territory, or Region' with 'United States' selected. Underneath, there's a 'Birthday' section with three dropdown menus for 'Month', 'Day', and 'Year'. A large blue button labeled 'NEXT' is positioned below the birthday fields. At the bottom, there's a link that says 'ALREADY HAVE AN ACCOUNT? SIGN IN'.

Figure 6.3 – Création d'un compte personnel

Après la saisie de votre nouveau mot de passe, cliquer sur .



This screenshot shows the 'Create account' page with the 'Email' and 'Password' fields filled out. Below the password field, there's a checkbox that is checked, with the text 'I agree to the Autodesk Terms of Use and acknowledge the Privacy Statement.' A large blue button labeled 'CREATE ACCOUNT' is at the bottom.

Figure 6.4 – Saisie de l'adresse et du mot de passe Tinkercad

Il se peut que vous receviez un mail de confirmation à l'adresse mail fournie. Suivez la procédure du mail puis lors de votre première connexion à TinkerCAD, vous tombez sur une interface avec un menu latéral :

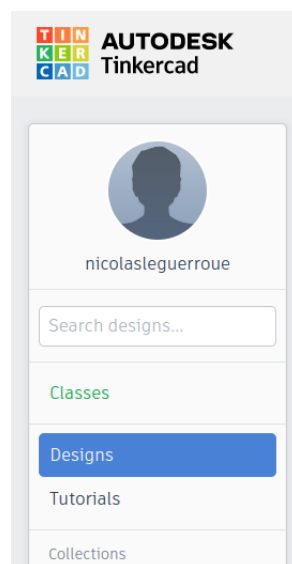


Figure 6.5 – Menu latéral

Sur la page centrale, vous pourrez retrouver vos réalisations :

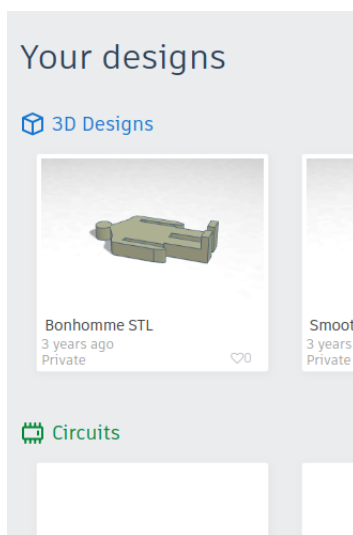


Figure 6.6 – La liste de vos projets et réalisations



Tinkercad permet également de réaliser des projets de CAO 3D.

Dans notre cas, nous souhaitons éditer une simulation Arduino, nous allons donc dans la section **Your Designs** puis **+ New**.

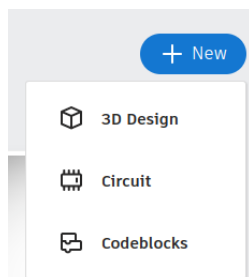


Figure 6.7 – On clique sur New Circuit

Vous arrivez ensuite sur la zone d'édition de circuits.

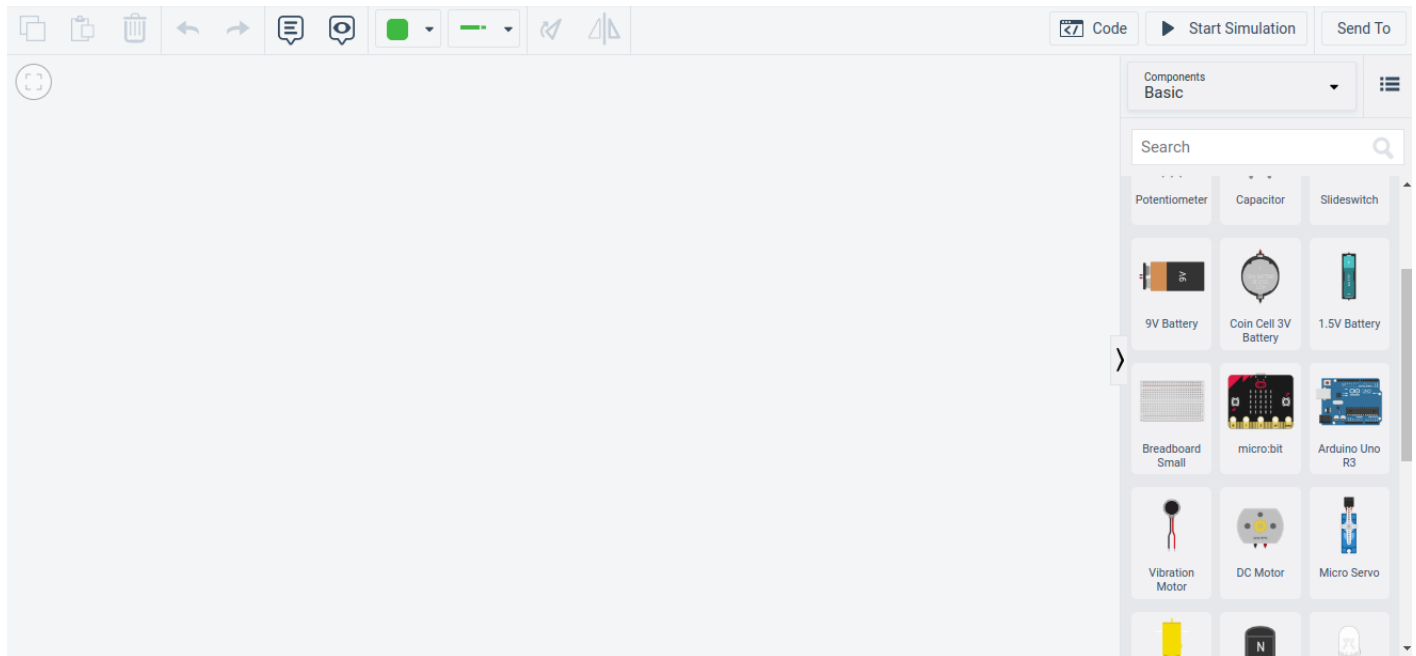


Figure 6.8 – La page d'édition TinkercAD

Nous allons réaliser une simulation avec une carte Arduino et un transistor MOSFET. Le menu de droite permet de faire des recherches de composants.

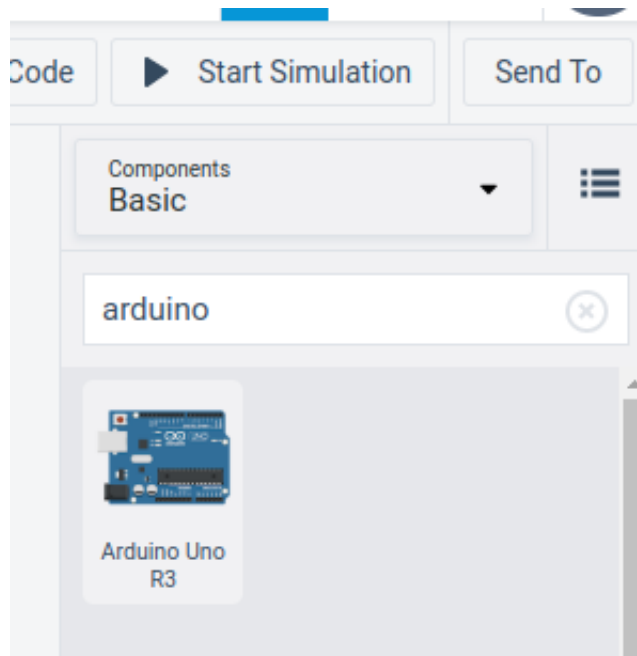


Figure 6.9 – La recherche de composants

Il vous suffit de faire un glisser-déposer de la carte Arduino vers la zone centrale pour qu'elle soit affichée dans la zone.

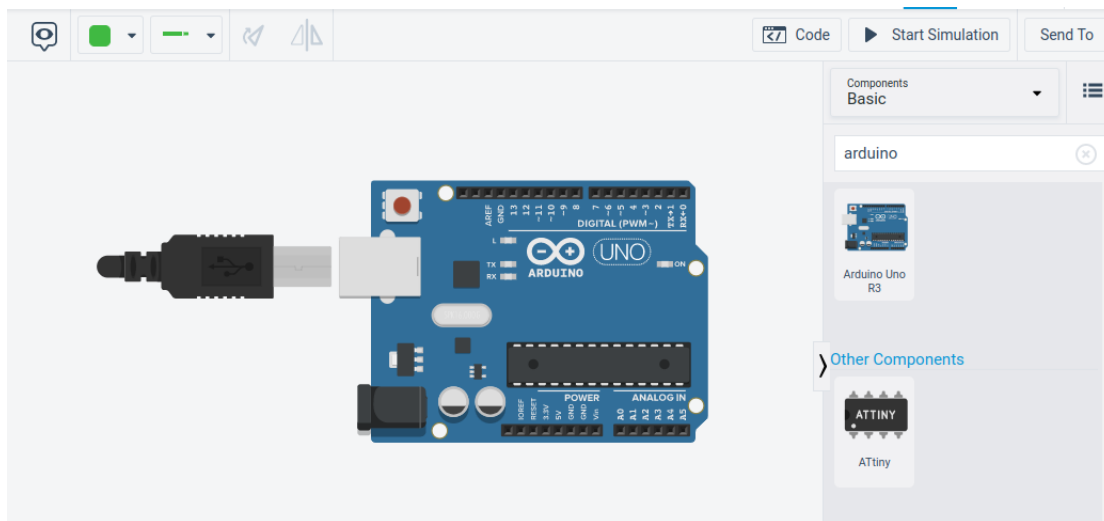


Figure 6.10 – Une carte Arduino

Nous allons ajouter un MOSFET de la même manière :

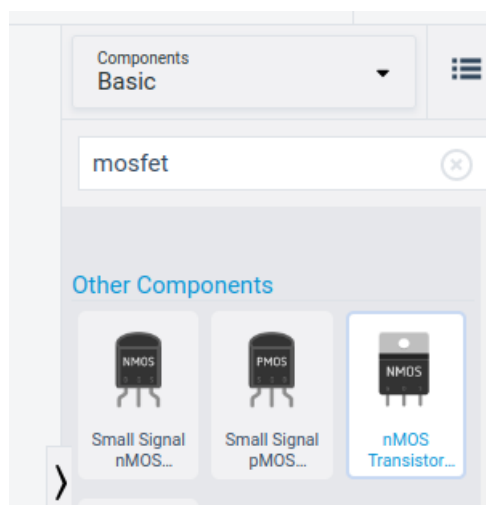


Figure 6.11 – Un MOSFET

Pour ajouter des câbles, il suffit de cliquer sur une des bornes des différents composants.

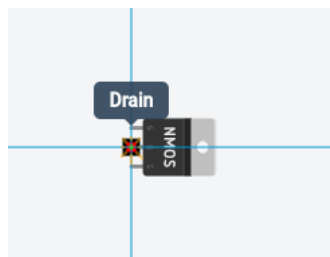


Figure 6.12 – Ajouter un câble

Et pour modifier la couleur des câbles, il faut cliquer sur le bouton coloré dans la barre supérieure :

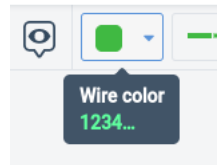


Figure 6.13 – Ajouter un câble

A vous de réaliser le montage suivant :

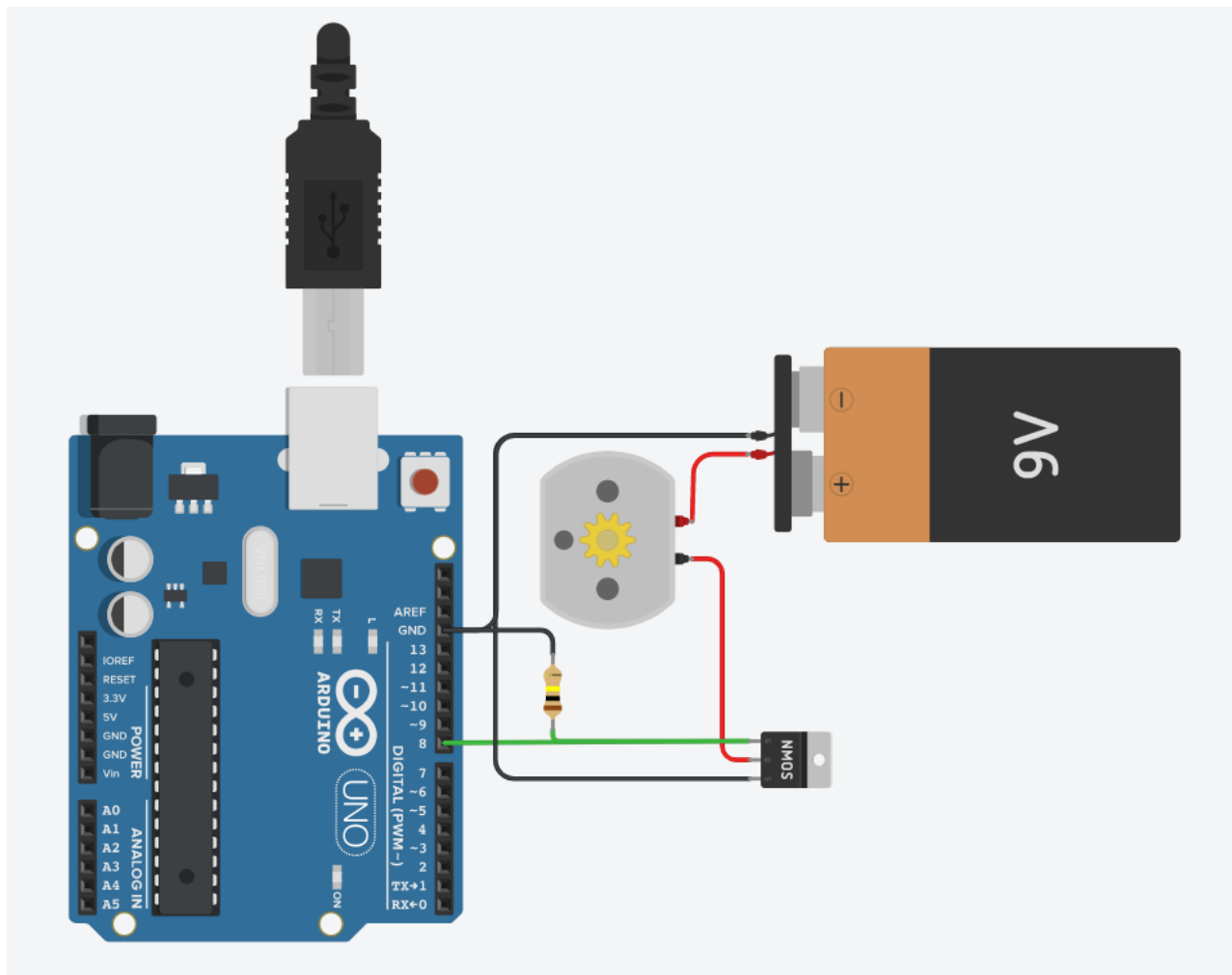



Figure 6.14 – Notre montage d'exemple

Voici donc les mot-clés à saisir pour la recherche des composants :

- ▶ Arduino Uno r3
- ▶ resistor
- ▶ nmos
- ▶ dc motor

► 9v battery

Une fois le schéma édité, nous allons pouvoir programmer la carte Arduino. Pour cela, il faut cliquer sur  **Code** en haut à droite dans le menu :

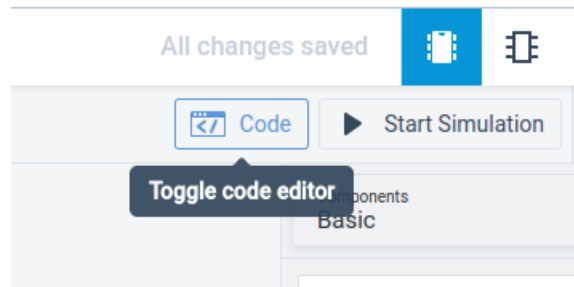


Figure 6.15 – Bouton du code

Puis mettre le code en mode Texte :

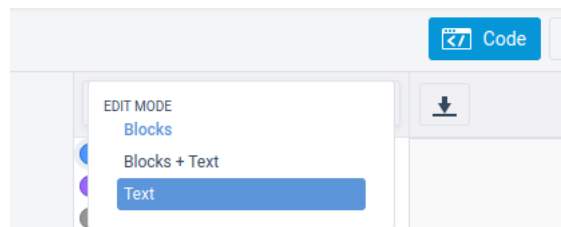


Figure 6.16 – Code en texte

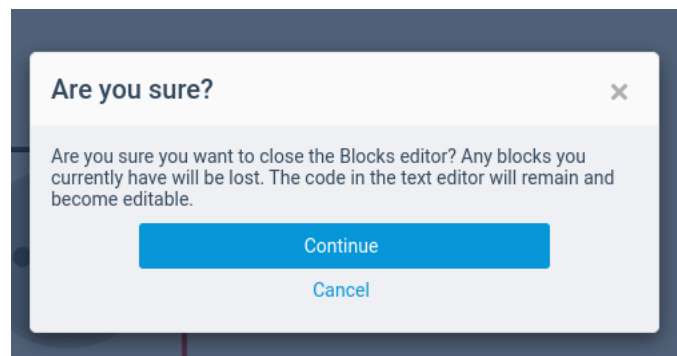


Figure 6.17 – Confirmation du passage en mode Texte

Le code par défaut est le code pour faire clignoter la LED interne de la carte.

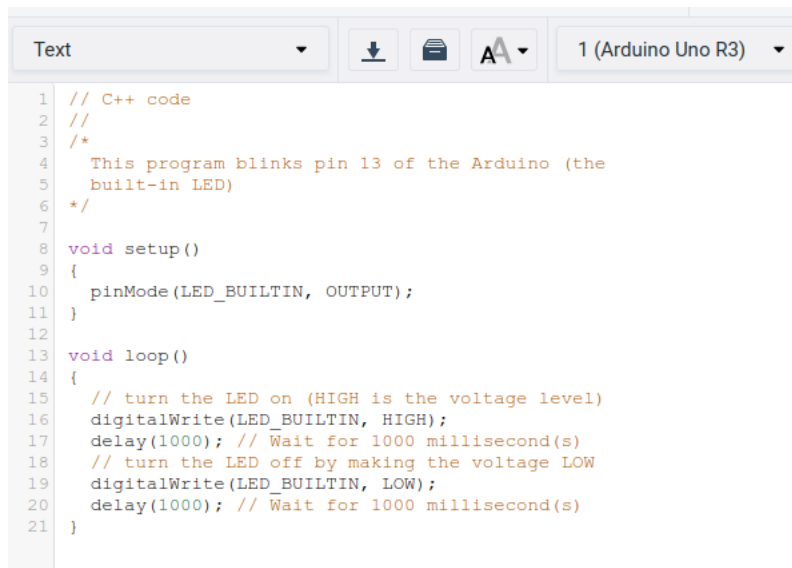


Figure 6.18 – Code par défaut

Veuillez le remplacer par le code suivant :

```

#define PIN 8      //GATE du transistor

void setup() {

    pinMode(PIN, OUTPUT); //Mise en sortie de la broche

} //Fin setup

void loop() {

    digitalWrite(PIN, HIGH);    //Mise en route du transistor
    delay(5000);                //Délai de 5s
    digitalWrite(PIN, LOW);     //Arrêt du transistor
    delay(5000);                //Délai de 5s

} //Fin loop

```

Code 4 - Code MOSFET

Une fois le code copié, nous allons pouvoir lancer la simulation.

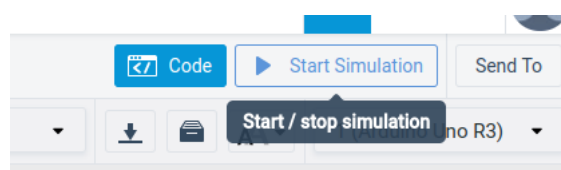


Figure 6.19 – Lancement de la simulation

Le bouton devient vert lorsque la simulation est en cours.

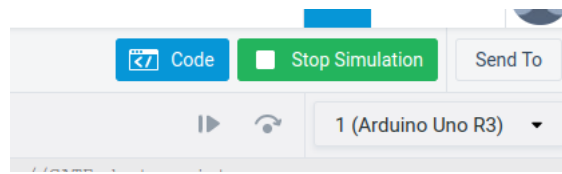


Figure 6.20 – Simulation en cours

Pour preuve, pendant 5 secondes, le moteur affiche une vitesse en tour/min :

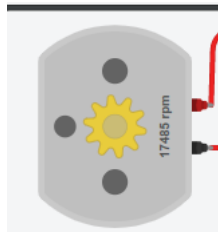


Figure 6.21 – Simulation du moteur

Puis pendant 5 autres secondes, la vitesse affichée est nulle. La simulation est donc fonctionnelle.

Bibliographie

- [1] Art Pini. Les multiplexeurs analogiques. <https://www.digikey.fr/fr/articles/save-space-cost-power-using-analog-multiplexers-switches>.
- [2] Wikipédia. Théorème d'échantillonnage. https://fr.wikipedia.org/wiki/Th%C3%A9or%C3%A8me_d%27%C3%A9chantillonnage.