Ateliers CREPP

Ensemble des supports lors des Ateliers CREPP

Nicolas Le Guerroué

5 décembre 2021
220 pages

TABLE DES MATIÈRES

SECTION 1

PRÉAMBULE

- Document réalisé en LaTeX par Nicolas Le Guerroué pour le Club de Robotique et d'Electronique Programmable de Ploemeur (CREPP)
- Permission vous est donnée de copier, distribuer et/ou modifier ce document sous quelque forme et de quelque manière que ce soit.
- Version du 5 décembre 2021
- Taille de police : 11pt

06.20.88.75.12

nicolasleguerroue@gmail.com

- **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.**

Première partie

Les amplificateurs opérationnels

Théorie sur les amplificateurs opérationnels

SECTION 2
INTRODUCTION

Généralités

Un AOP (Amplificateur Opérationnel) est un composant actif qui permet de réaliser des opérations mathématiques (addition, soustraction, intégration, dérivation, etc) et du fait de sa miniaturisation et de sa fiabilité, on le rencontre aujourd'hui dans de nombreuses applications comme l'audio, la radio, l'asservissement...

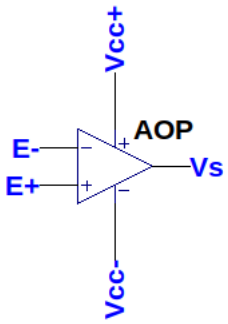


FIGURE 2.1 – Les entrées et sorties de l'AOP

Un AOP possède deux entrées notées appelées entrée non inverseuse et entrée inverseuse, une sortie et deux broches d'alimentation. L'AOP dispose souvent d'une alimentation symétrique ($Vcc+$ et $Vcc-$) avec comme référence de tension le point milieu (GND) des alimentations.

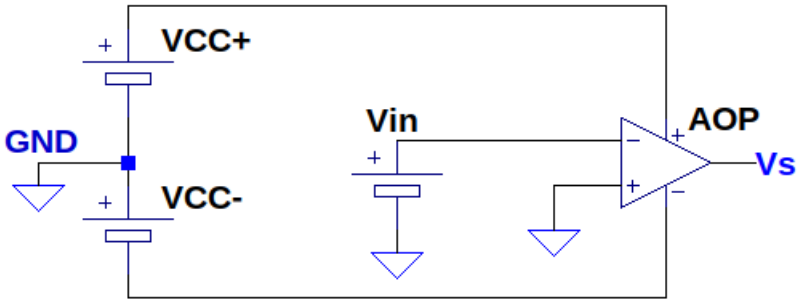


FIGURE 2.2 – L'alimentation d'un AOP

Conventions

Afin de simplifier les calculs sur les AOP, quelques conventions ont été adoptées :

- La tension de sortie de l'AOP est notée Vs
- La tension sur l'entrée inverseuse est appelée E_-
- La tension sur l'entrée non inverseuse est appelée E_+
- La tension différentielle $(E+-E-)$ est appelée ε
- Le gain d'amplification différentiel de l'AOP est appelé Ad Ce gain est variable entre différentes familles d'AOP mais reste constant dans le temps
- Le gain d'amplification du montage est appelé A_0 et varie en fonction des différents montages possibles

Remarque

L'alimentation des montages suivants ne sera pas représenté par souci de clarté.

SECTION 3

MODÉLISATION DE L'AOP

Modèle théorique

Dans un souci de simplification des calculs, un AOP peut être vu physiquement comme un composant ayant des caractéristiques parfaites. Ces caractéristiques sont les suivantes :

- impédance d'entrée : $Z \rightarrow +\infty\Omega$
- Impédance de sortie : $Z = 0\Omega$
- $Ad \rightarrow +\infty$
- Vsmax $= Vcc_+$
- Vsmin $= Vcc_-$
- Bande passante : $Fmax \rightarrow +\infty$

Cela se traduit par un modèle dont les deux entrées sont ouvertes (courant nul) et avec une tension de sortie qui ne serait pas affectée par le courant de sortie

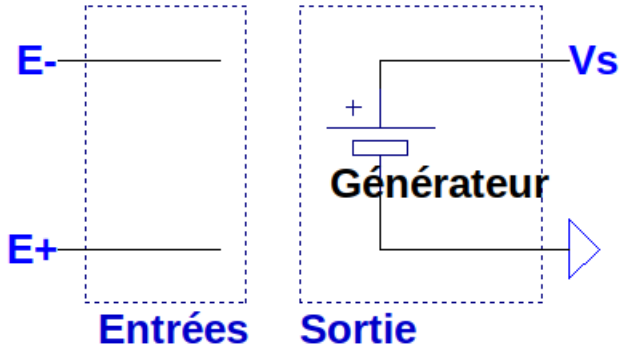


FIGURE 3.1 – Le modèle théorique de l'AOP

Modèle réel

Cependant, il convient de noter que ce modèle n'est que théorique.

Du fait de la nature des composants constituant les AOP (transistors, condensateurs), la tension de sortie ne peut pas être égale à la tension d'alimentation.
Cette tension de sortie max est appelée $Vsat_+$et $Vsat_-$

D'où le modèle suivant :

- impédance d'entrée : $Z > 10^5\Omega$
- Impédance de sortie : $Z > 0\Omega$ (courant de sortie max $20mA$)
- $Ad \gg 1000$
- $Vs_{max} = Vcc_{sat_+}$
- $Vs_{min} = Vcc_{sat_-} -$
- Bande passante : imposée par le constructeur (Ex : le LM324 tolère 1MHz)

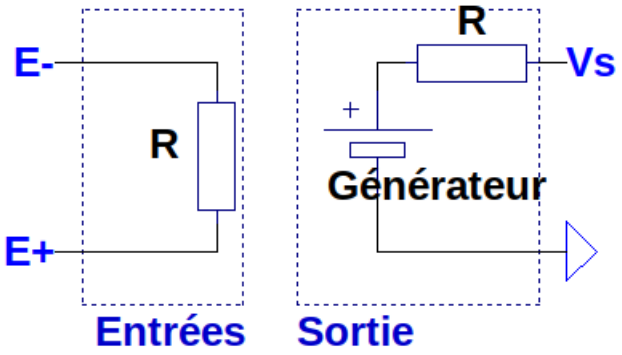


FIGURE 3.2 – Le modèle réel de l'AOP

Cependant, pour les calculs, l'hypothèse du courant d'entrée nul sera retenue, tout comme celle du gain Ad

Modes de fonctionnement

Montages linéaires

Le signal V_s est une fonction mathématique du signal d'entrée V_e.

Dans certains cas, le signal de sortie conserve la forme du signal d'entrée sous couvert que que l'AOP ne rentre pas en saturation.

Un montage linéaire impose un ε nul, sauf si l'AOP est saturé, c'est à dire si $V_s = V_{sat}$

Montages comparateurs

Le signal de sortie ne peut prendre que deux valeurs, $Vsat_+$ou $Vsat_-$ En l'absence de contre-réaction, $Vs = \varepsilon Ad$

Une réaction est un retour du signal sur une des deux entrées. Celle ci peut être positive ou négative en fonction de l'entrée choisie (entrée -,réaction négative et entrée +, réaction positive).

Type de réaction	*Positive*	*Négative*	*Aucune*
Mode de fonctionnement	***Comparateur***	***Linéaire***	***Comparateur***

FIGURE 3.3 – Les modes de fonctionnement de l'AOP

Résistance de charge

Il est possible de mettre une résistance de charge entre V_s et la masse.

Cette résistance symbolise un circuit relié directement à l'AOP.

Cependant, le courant de l'AOP étant limité à quelques dizaines de mA, il convient de prendre une résistance de charge R_c suffisamment grande ($Rc > 1000\omega$).

Si R_c est suffisamment élevée, cette dernière n'influence pas la tension de sortie V_s

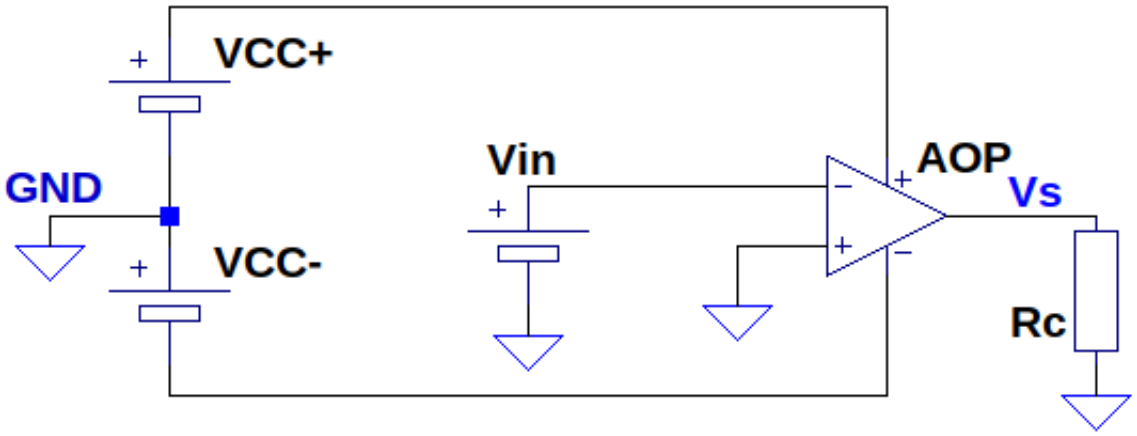


FIGURE 3.4 – La résistance de charge

SECTION 4

ÉTUDE EN MODE LINÉAIRE

Intérêt de l'étude

Les AOP en fonctionnement linéaire permettent de réaliser les opérations mathématiques :

- **amplification** : $Vs = A_0 \cdot Ve$ A_0 est le coefficient d'amplification du montage (A ne pas confondre avec A_d le coefficient d'amplification différentiel imposé par le constructeur) A_0 peut être positif ou négatif
- **addition algébrique** : $V_s = \sum_{k=0}^{n} V_k$
- **intégration et dérivation** (avec des condensateurs) à une constante près
- **logarithme et exponentielle**

Méthode de résolution

La réaction négative (liaison entre la sortie et l'entrée inverseuse) impose un fonctionnement stable et linéaire, d'où $\varepsilon = 0, E_+ = E_-$

L'hypothèse de la résistance d'entrée de l'AOP implique que $I_+ = I_- = 0$
Afin de déterminer Vs, il faut exprimer E_+**et** E_- **en fonction des éléments du montage** .

En égalisant les deux équations obtenues ($E_+ = k$ **et** $E_- = k'$**)** , on obtient une relation de type $V_s = f(V_e)$

Remarque

I_s est issu d'une source de tension, il n'y a donc pas de loi simple permettant de déterminer sa valeur algébrique. Il ne faut pas avoir d'à priori sur son sens

SECTION 5

MONTAGE SUIVEUR

Présentation

Ce montage permet de reproduire à l'identique une tension d'entrée.
L'intérêt de ce montage réside dans le fait que l'impédance d'entrée de l'AOP est considérée comme infinie et que son impédance de sortie est considérée comme nulle.

Ainsi, le comportement de la charge en entrée ne sera pas affecté par l'AOP, le signal d'entrée ne sera donc pas modifié.

Ce montage sert donc à faire une **adaptation d'impédance** .

Montage

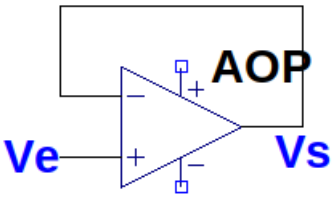


FIGURE 5.1 – Le montage suiveur

Démonstration

La réaction négative implique que $\varepsilon = 0$ (fonctionnement linéaire)

$$E_+ = Ve$$

$$E_- = Vs$$

$$\Rightarrow Ve = Vs$$

car $E_+ = E_-$

Exemple 1. *On souhaite mesurer une tension au borne d'un capteur avec un appareil de mesure.*

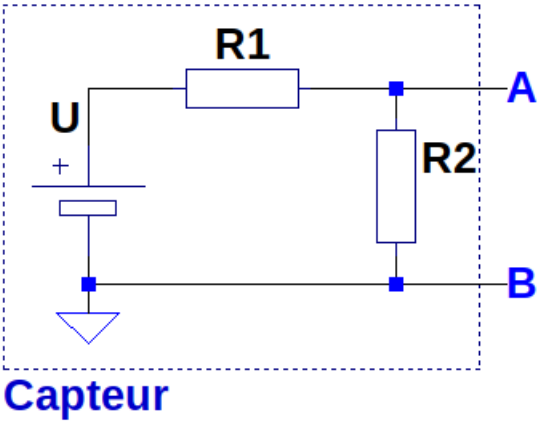


FIGURE 5.2 – Le capteur

On place ensuite une charge R_c au bornes de A et b. Cette résistance R_c représente l'appareil d'acquisition.

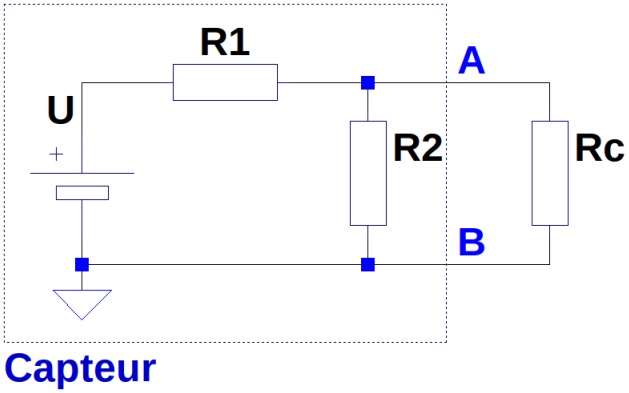


FIGURE 5.3 – Le modèle d'acquisition

Question 1. *Quelle est l'influence de Rc sur U_{AB} dans le montage suivant ?*

>>> **1.** *Sans la charge Rc :*

$$U_{AB} = \frac{U \cdot R_2}{R_1 + R_2}$$

Avec la charge Rc :

$$U_{AB} = \frac{U \cdot R_{equ}}{R_1 + R_{equ}}$$

Avec R_{equ} la résistance équivalente entre R_2 et R_c

Si $R_c \to +\infty$ *alors* $R_{equ} \to \frac{U_{AB} \cdot R_2}{R_1 + R_2}$

Si $R_c \to +0$ *alors* $R_{equ} \to 0 \Rightarrow U_{AB} \to 0$

D'où le montage suivant, avec $R_c \to +\infty$, le signal n'est pas déformé.

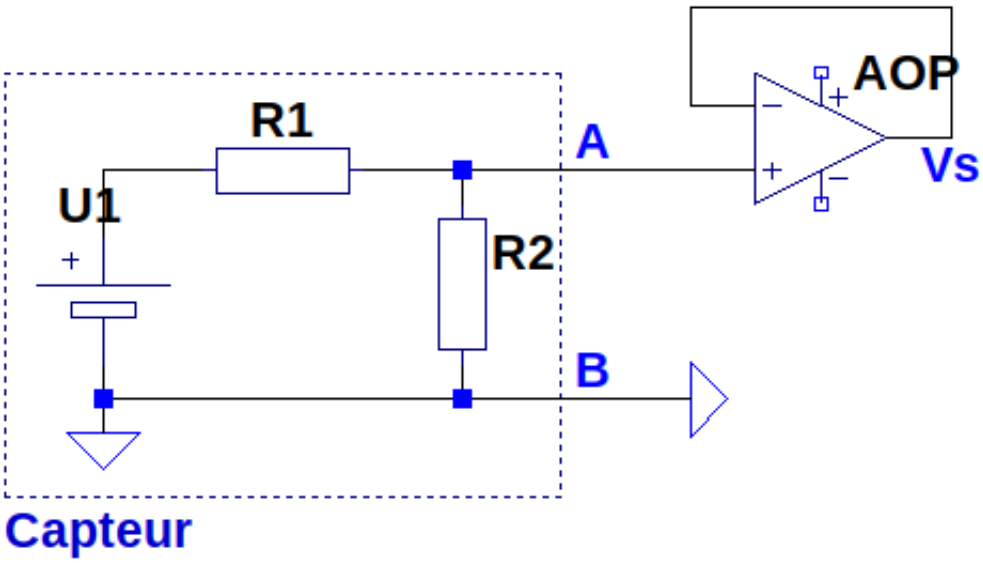


FIGURE 5.4 – L'adaptation d'impédance

SECTION 6

MONTAGE NON-INVERSEUR

Présentation

Ce montage amplifie la tension V_e par un **gain** A_0 **positif** .
L'amplificateur reste en mode linéaire si $Ve < Vcc_{sat} \cdot A_0$

Montage

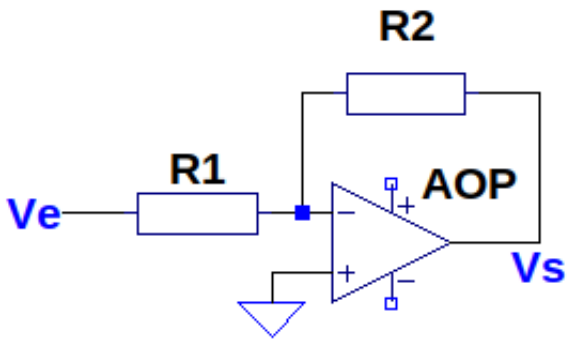


FIGURE 6.1 – Le montage amplificateur non inverseur

Démonstration

Un AOP en mode linéaire impose $\varepsilon = 0$ D'où $E_+ = E_-$

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

$$E_+ = E_- \Rightarrow V_e = V_s \cdot \frac{R_2}{R_1 + R_2} \tag{6.1}$$

$$\Rightarrow \frac{V_e}{V_s} = \frac{R_2}{R_1 + R_2} \tag{6.2}$$

$$\Rightarrow V_s = V_e \cdot \frac{R_1 + R_2}{R_2} \tag{6.3}$$

$$\tag{6.4}$$

Exemple

Exemple 2. *On souhaite amplifier un signal sinusoïdal par un coefficient $k = 5$.*
On peut donc utiliser le montage précédent.
On prendra $R_1 = 1k\Omega$ et $R_2 = 4k\Omega$ pour avoir $A_0 = 5$

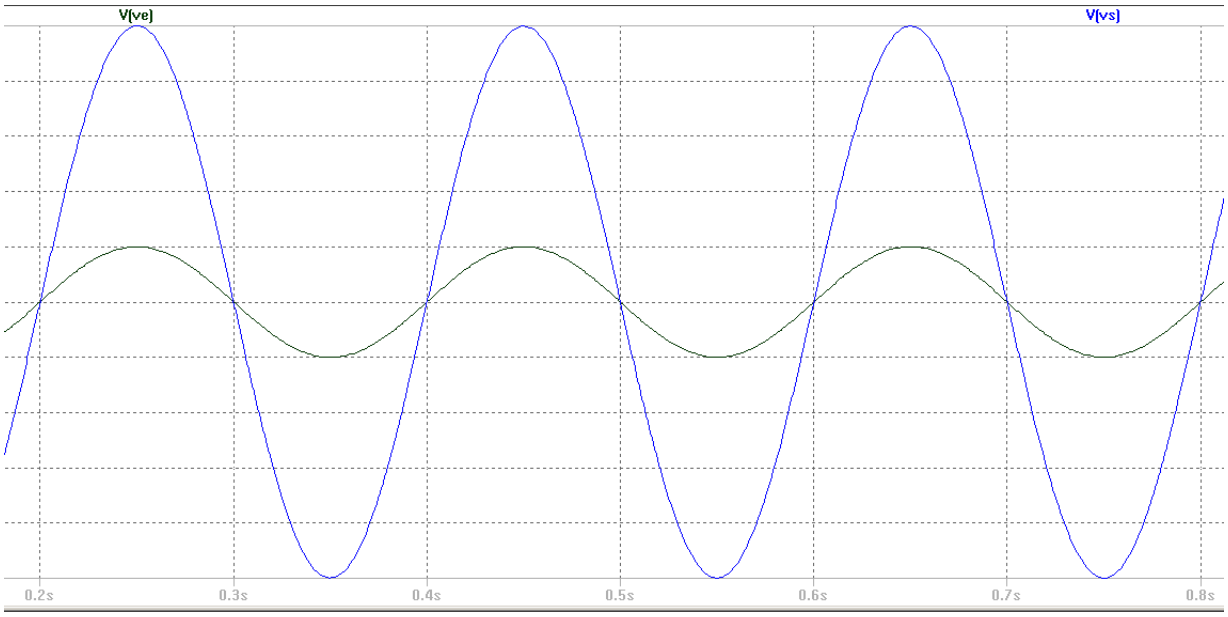


FIGURE 6.2 – Amplification du signal noir par 5

SECTION 7

MONTAGE INVERSEUR

Présentation

Ce montage amplifie la tension V_e par un **gain** A_0 **négatif** .

Montage

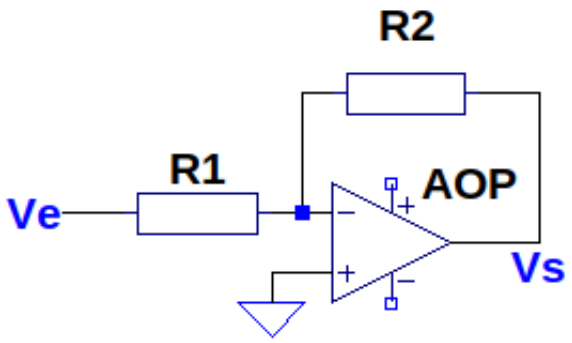


FIGURE 7.1 – Le montage amplificateur non inverseur

Démonstration

Mode linéaire : $\varepsilon = 0$

$$E_+ = 0 \tag{7.1}$$

$$E_- = \frac{\frac{V_e}{R_1} + \frac{V_s}{R_2}}{\frac{1}{R_1} + \frac{1}{R_2}} \tag{7.2}$$

$$E_- = \frac{V_e \cdot R_2 + V_s \cdot R_1}{R_1 + R_2} \tag{7.3}$$

$$\Rightarrow \frac{V_e}{V_s} = -\frac{R_1}{R_2} \tag{7.4}$$

$$\Rightarrow V_s = -V_e \cdot \frac{R_2}{R_1} \tag{7.5}$$

Avec $A0 = -\frac{R_2}{R1}$

Application

Exemple 3. *On souhaite amplifier un signal sinusoïdal par un coefficient $k = -5$.*
On peut donc utiliser le montage précédent.
On prendra $R_1 = 1k\Omega$ et $R_2 = 5k\Omega$ pour avoir $A_0 = -5$

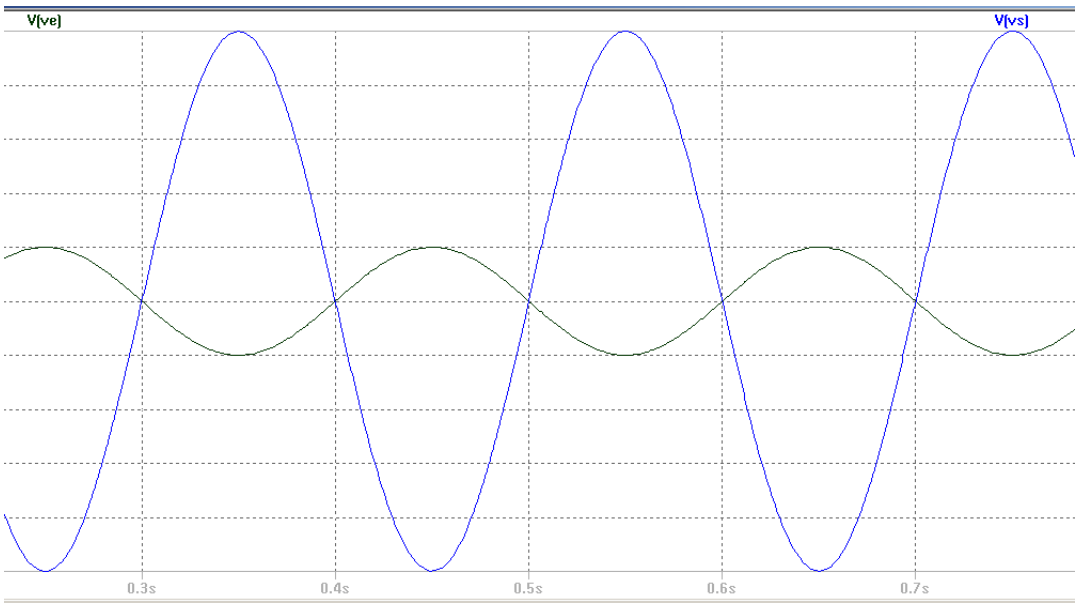


FIGURE 7.2 – Amplification du signal noir par -5

SECTION 8

MONTAGES COMPARATEURS

Présentation

Un montage comparateur se reconnaît par son branchement :

- Aucune contre réaction n'est présente
- Une contre réaction a lieu sur l'entrée non inverseuse via un dipôle passif

Le montage comparateur permet de comparer deux tensions entre elles.
Cependant, cette comparaison peut s'effectuer de plusieurs manières, avec un ou deux seuils, de manière inversée ou non...

Comparateur non inverseur simple seuil

Présentation

Ce montage permet de comparer simplement deux tensions entre elle.

Montage

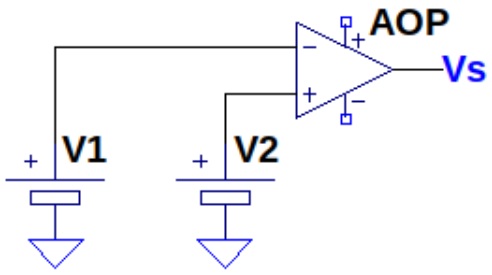


FIGURE 8.1 – Comparateur simple seuil

Ce mode est le plus simple et est régi de la manière suivante :

On sait que $\varepsilon = E_+ - E_-$ et que $V_s = \varepsilon \cdot A_d$ avec $Ad = +\infty$
(Circuit en boucle ouverte)

Si $E_+ > E_-$:

$$Vs = Vsat_+$$

si $E_+ < E_-$:

$$Vs = Vsat_-$$

Si $V_2 = 0V$, on obtient la caractéristique de transfert suivante

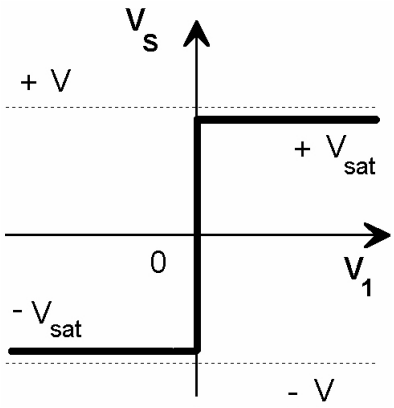


FIGURE 8.2 – Caractéristique de transfert, V_1 est la tension d'entrée de l'AOP

Comparateur inverseur simple seuil

Le raisonnement est le même sauf que les entrées sont inversées.
De ce fait, le seuil de basculement se fait dans l'autre sens.

Si $E_+ > E_-$:

$$Vs = Vsat-$$

Si $E_+ < E_-$:

$$V_s = Vsat_+$$

Comparateur non inverseur double seuil

Présentation

Ce type de montage permet d'éliminer les tensions "parasites", c'est à dire les tensions bruités et non indésirables.

Montage

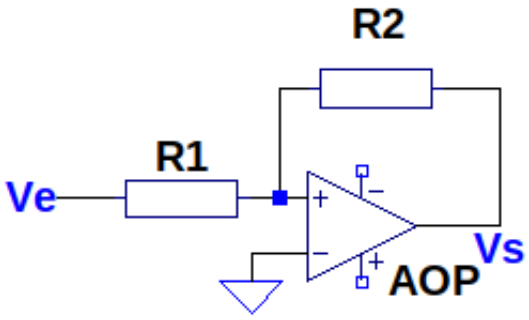


FIGURE 8.3 – Montage comparateur non inverseur double seuil

Application

Par exemple, un capteur de lumière résistif (photo-résistance) sera sensible aux variations de lumière (nuages...).

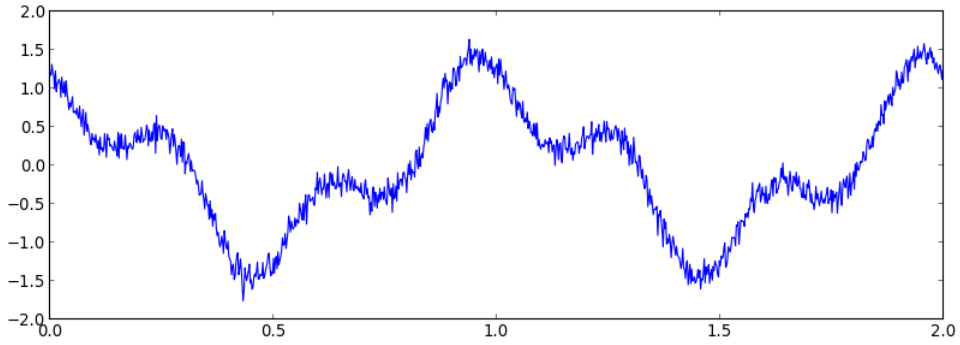


FIGURE 8.4 – Un signal bruité

Or, si on compare ce signal par rapport à une référence, on ne veut pas que le capteur déclenche plusieurs fois l'action.

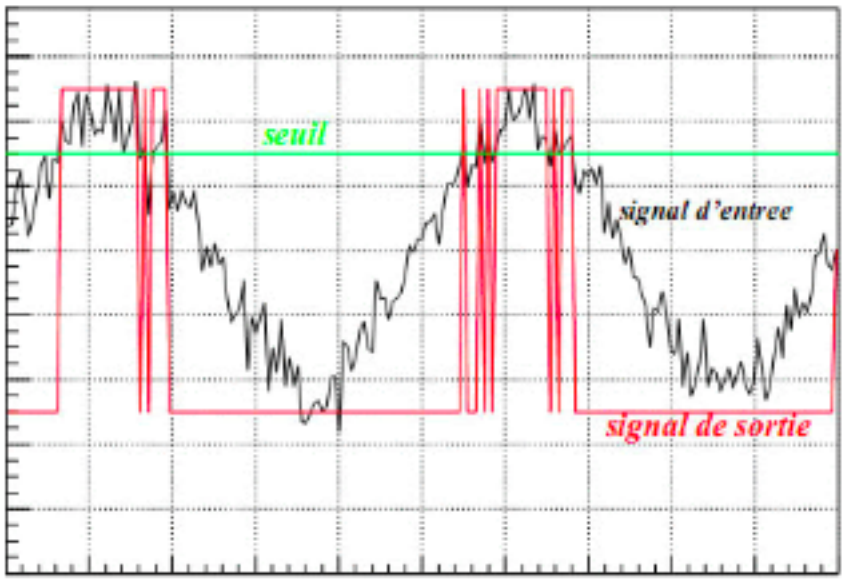


FIGURE 8.5 – Le cycle de déclenchement

Pour éviter ce problème, on utilise un comparateur à double seuil :
toute les tensions parasites ayant une amplitude inférieure à la tension de différence entre les deux seuils seront ignorées.

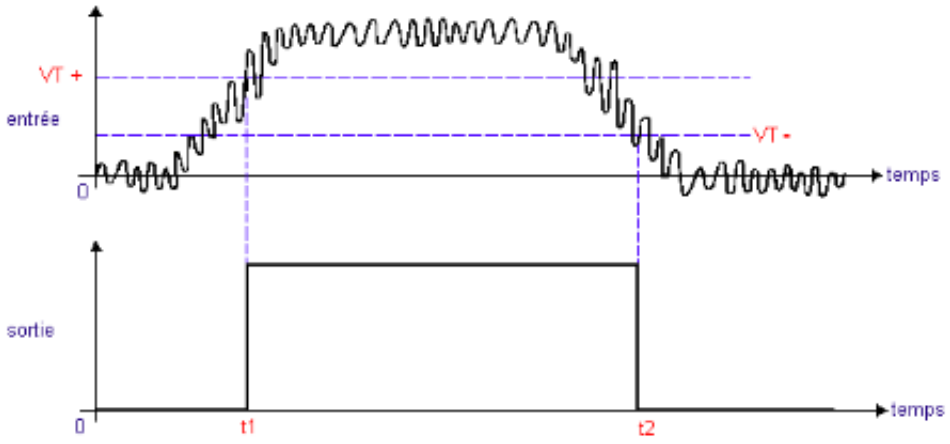


FIGURE 8.6 – Le principe

On va chercher les deux valeurs de basculement :

$$E_+ = \frac{V_e R_2 + V_s R_1}{R_1 + R_2}$$
$$E_- = U_0 = 0$$

Étudions le cas où $\varepsilon > 0$

$$\varepsilon > 0 \Leftrightarrow E_+ > E_- \tag{8.1}$$
$$\Leftrightarrow \frac{V_e R_2 + V_s R_1}{R_1 + R_2} > U_0 \tag{8.2}$$
$$\Leftrightarrow \frac{V_e R_2}{R_1 + R_2} > U_0 - \frac{V_s R_1}{R_1 + R_2} \tag{8.3}$$
$$\Leftrightarrow V_e R_2 > R_1 + R_2 \cdot U_0 - V_s R_1 \tag{8.4}$$
$$\Leftrightarrow V_e > \frac{R_1 + R_2 \cdot U_0 - V_s R_1}{R_2} \tag{8.5}$$

Ici, $U_0 = 0$ et $V_s = V_{sat_+}$ car $\varepsilon > 0$
D'où $V_e > \frac{-V_{sat} + R_1}{R_2}$

Remarque

U_0 peut être différent de $0V$ en mettant une source de tension sur E_-

Étudions le cas où $\varepsilon > 0$:

$$\varepsilon > 0 \Leftrightarrow E_+ < E_- \tag{8.6}$$
$$\Leftrightarrow \frac{V_e R_2 + V_s R_1}{R_1 + R_2} < U_0 \tag{8.7}$$
$$\Leftrightarrow \frac{V_e R_2}{R_1 + R_2} < U_0 - \frac{V_s R_1}{R_1 + R_2} \tag{8.8}$$
$$\Leftrightarrow V_e R_2 < R_1 + R_2 \cdot U_0 - V_s R_1 \tag{8.9}$$
$$\Leftrightarrow V_e < \frac{R_1 + R_2 \cdot U_0 - V_s R_1}{R_2} \tag{8.10}$$

Ici, $U_0 = 0$ et $V_s = V_{sat_-}$ car $\varepsilon < 0$

D'où $V_e < \frac{-V_{sat} + R_1}{R_2}$
On obtient deux seuils $S1$ et $S2$ de valeurs respectives :

$\frac{-V_{sat} + R1}{R2}$ et $\frac{Vsat - R_1}{R_2}$

Afin de basculer, la tension d'entrée doit dépasser $S1V$ et afin de basculer dans l'autre sens, la tension d'entrée doit être inférieure à $S2V$

On obtient le cycle d'hystérésis suivant :

$(VB = S2 et VH = S1)$

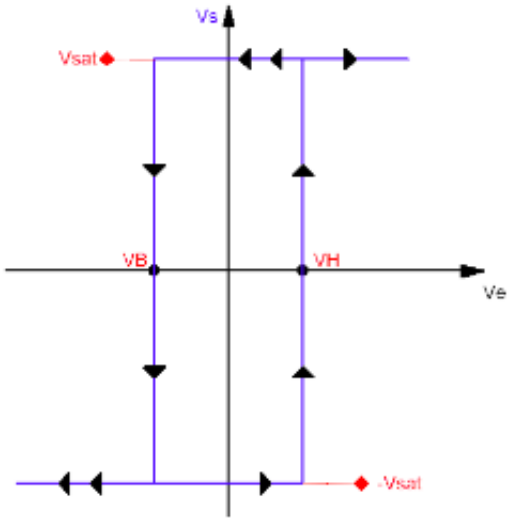


FIGURE 8.7 – Cycle d'hystérésis non inverseur

$$U_{milieu\_de\_cycle} = (V_{seuil1} + V_{seuil2}) \cdot 0.5$$
$$Largeur_{cycle} = V_{seuil1} - V_{seuil2}$$

Comparateur inverseur double seuil

Montage

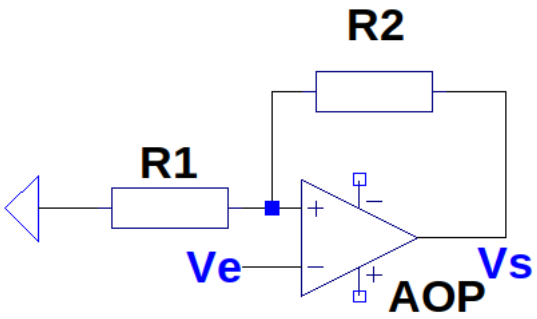


FIGURE 8.8 – Montage comparateur inverseur double seuil

Démonstration

La démarche est rigoureusement identique avec $\varepsilon > 0$, on a $V_s = Vsat_+$ et pour $\varepsilon < 0$ on a $V_s = Vsat_-$

On obtient le cycle d'hystérésis suivant :

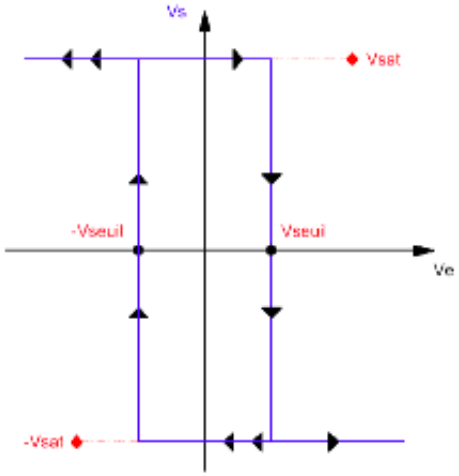


FIGURE 8.9 – Cycle d'hystérésis inverseur

SECTION 9

MONTAGE INTÉGRATEUR

Présentation

Ce montage intègre une tension d'entrée.
En sortie, on obtient une tension V_s valant $V_s = k \cdot \int V_e$

Montage

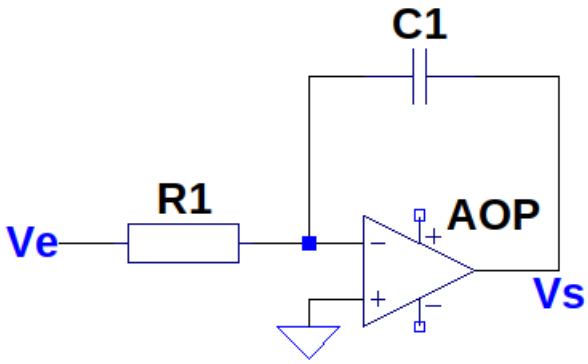


FIGURE 9.1 – Montage intégrateur

Démonstration

Montage en mode linéaire car contre-réaction négative.

$$I_{R1} + I_{C1} = 0 \qquad \Leftrightarrow I_{R1} = -I_{C1} \tag{9.1}$$

$$\Leftrightarrow \frac{E}{R} = -C \cdot \frac{dV_s}{dt} \tag{9.2}$$

$$\Leftrightarrow -\frac{1}{RC} \cdot E = \frac{dV_s}{dt} \tag{9.3}$$

$$\Leftrightarrow V_s = -\frac{1}{RC} \int V_e \tag{9.4}$$

$$k = -\frac{1}{RC} \tag{9.5}$$

Ce montage est notamment présent dans certains Convertisseurs Analogiques Numériques dit “simple” ou “double” rampe.

Application

Exemple 4. *On souhaite générer un signal triangulaire.*

En intégrant un signal rectangulaire, on obtient un signal triangulaire.

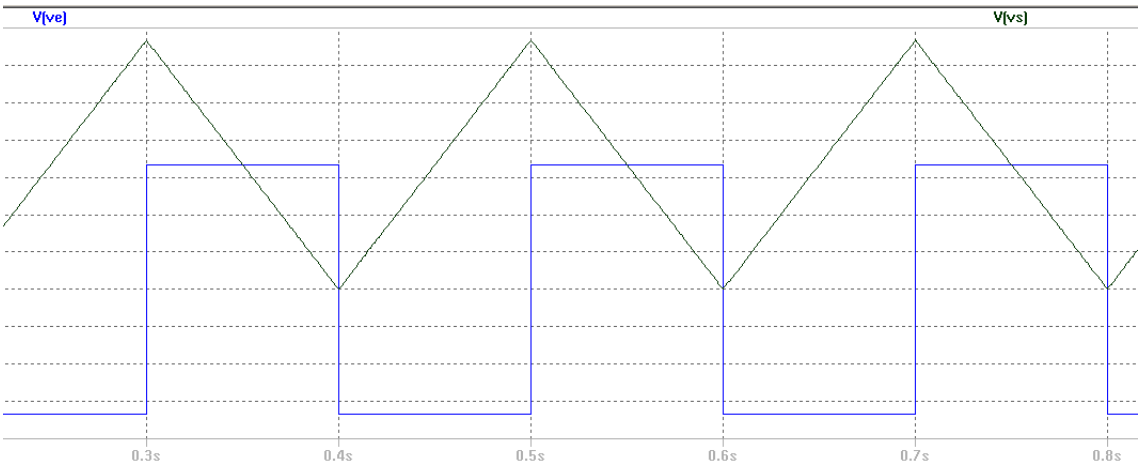


FIGURE 9.2 – Un signal triangulaire généré depuis un signal carré

Soit $V_e = 2V$ ou $V_e = -2V$
Si $V_e = 2V$

$$k \int V_e = kV_e \cdot t + x \tag{9.6}$$

$$= -\frac{2}{RC} t + c \tag{9.7}$$

avec $k = -\frac{1}{RC}$

$\Rightarrow$ droite d'équation $y = -\frac{2}{RC} + c$

Si $V_e = -2V$

$$k \int V_e = kV_e \cdot t + x \tag{9.8}$$

$$= \frac{2}{RC}t + c \tag{9.9}$$

avec $k = -\frac{1}{RC}$

$\Rightarrow$ droite d'équation $y = \frac{2}{RC} + c$

SECTION 10

MONTAGE SOUSTRACTEUR

Présentation

Ce montage permet de soustraire deux tensions d'entrée afin d'obtenir la différence en sortie.

Montage

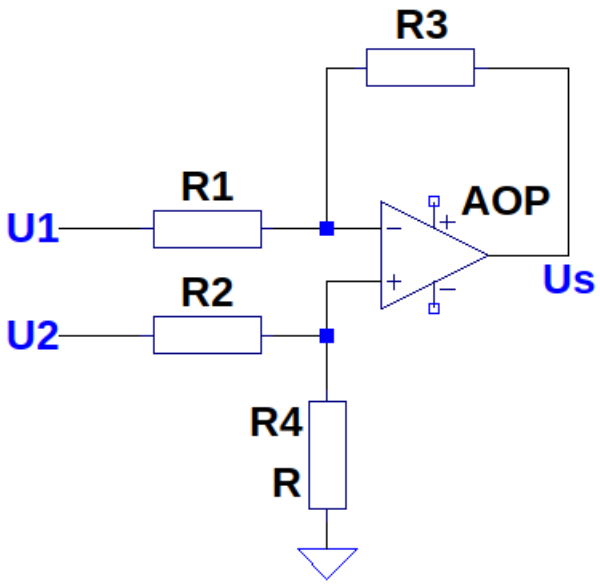


FIGURE 10.1 – Montage soustracteur

Démonstration

Contre-réaction négative donc mode linéaire.

$$E_+ = U_2 \frac{R_4}{R_2 + R_4}$$

$$E_- = \frac{\frac{U_1}{R_1} + \frac{U_5}{R3}}{\frac{1}{R_1} + \frac{1}{R_3}} \tag{10.1}$$

$$= \frac{U_1 R_3 + U_5 R_1}{R_1 + R_3} \tag{10.2}$$

Or $E_+ = E_-$

$$\Leftrightarrow U_2 \cdot \frac{R_4}{R_2 + R_4} = \frac{U_1 R_3 + U_5 R_1}{R_1 + R_3} \tag{10.3}$$

$$\Leftrightarrow \frac{U_5 R_1}{R_1 + R_3} = \frac{U_2 R_4}{R_2 + R_4} - \frac{U_1 R_3}{R_1 + R_3} \tag{10.4}$$

$$\Leftrightarrow U_5 R_1 = \frac{R_4(R_1 + R_3)}{R_2 + R_4} - \frac{U_1 R_3}{R_1} \tag{10.5}$$

Si $R_1 = R_2 = R_3 = R_4$, on obtient :

$U_s = U_2 - U_1$

Application

Exemple 5. *On souhaite mesurer une tension entre deux points A et B d'un circuit (tension différentielle)*

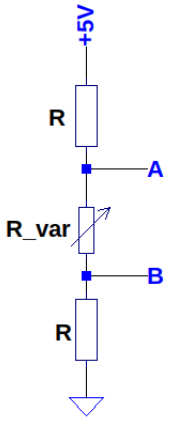


FIGURE 10.2 – Tension différentielle AB

Pour étudier la différence de potentiel entre les deux points du circuit, on peut utiliser un montage soustracteur afin qu'en sortie du montage avec l'AOP, on ait :

$$V_s = V_a - V_b$$

On peut réaliser le montage suivant.

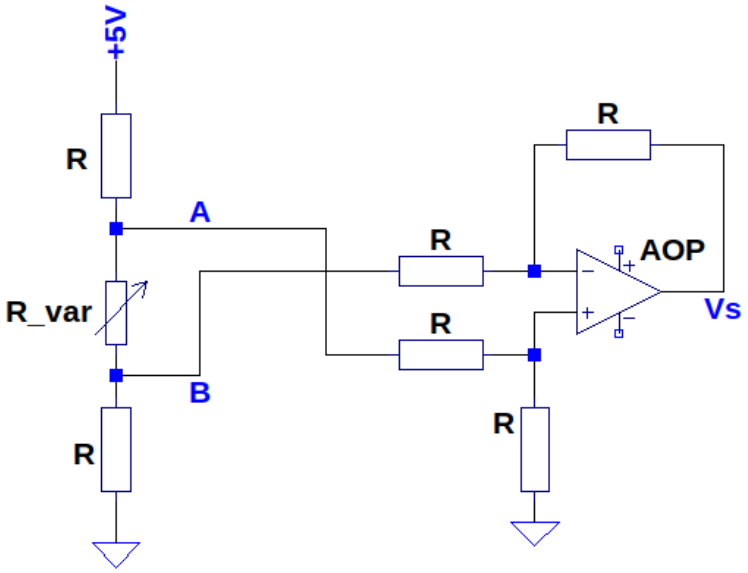


FIGURE 10.3 – Un montage pour lire une tension entre deux points

SECTION 11

MONTAGE SOMMATEUR

Présentation

Ce montage permet d'additionner en sortie plusieurs tensions d'entrée. Avec ce montage, la tension de sortie est multipliée par un coefficient -1

Montage

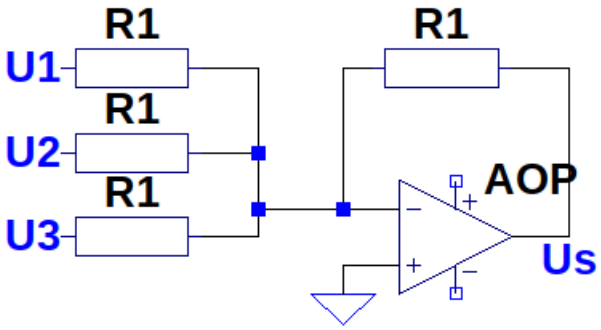


FIGURE 11.1 – Montage sommateur inverseur

Démonstration

Contre-réaction négative donc montage linéaire.
On applique le théorème de Millman [1]

$$E_+ = 0$$

$$E_- = \frac{\frac{U_1}{R_1} + \frac{U_2}{R_1} + \frac{U_s}{R_1}}{\frac{1}{R_1} + \frac{1}{R_1} + \frac{1}{R_1}} = 0$$

1. on fera abstraction de U3

$$\Leftrightarrow \frac{U_1 + U_2 + U_s}{R_1} = 0 \tag{11.1}$$

$$\Leftrightarrow \frac{U_s}{R_1} = \frac{-(U_1 + U_2)}{R_1} \tag{11.2}$$

$$\Leftrightarrow U_s = -(U_1 + U_2) \tag{11.3}$$

Deuxième partie

Les condensateurs

Théorie sur les condensateurs et applications

SECTION 12

HISTOIRE

C'est en 1745, que le physicien allemand Ewald Georg von Kleist invente le premier condensateur. Il a enroulé une feuille d'argent autour d'une bouteille en verre, et a chargé la feuille à l'aide d'un générateur à friction. Il était convaincu qu'une charge pourrait être accumulée lorsqu'il a reçu un choc électrique significatif (par un générateur par exemple).

Un an plus tard, Pieter van Musschenrboek poursuivra les recherches sur cette invention et lui donnera le nom de : "bouteille de Leyde". Pour la petite histoire, ce nom vient de l'université où travaillait ce dernier : l'université de Leyde. Le condensateur est une véritable révolution car il permet de contenir une importante charge électrique dans un très petit volume. La bouteille de Leyde est un condensateur formé de deux conducteurs séparés par le verre de la bouteille. Le premier conducteur est généralement constitué d'une électrode supérieure, reliée à des feuilles en étain mises dans la bouteille. Le second conducteur est formé par une feuille métallique autour la bouteille. Ces deux conducteurs permettent de créer deux charges égales mais de signes opposées.

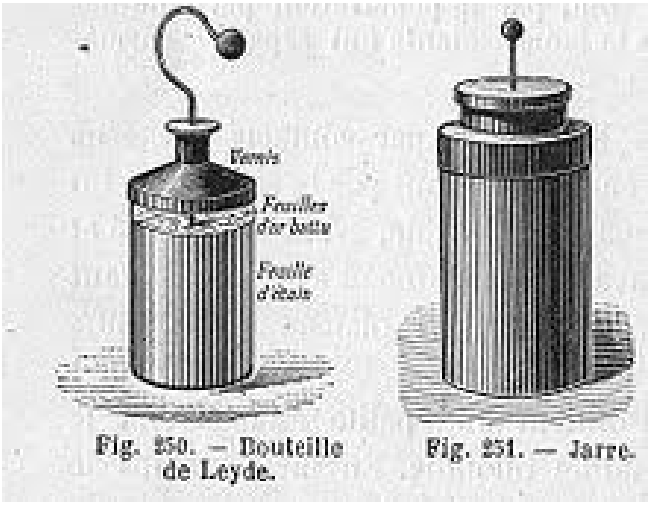


FIGURE 12.1 – Bouteille de Leyde

Puis avec le temps, d'autres condensateurs ont vu le jour :

FIGURE 12.2 – Différents types de condensateurs

SECTION 13

PRINCIPE DE FONCTIONNEMENT

Un condensateur est un dipôle électrique composé de deux armatures conductrices appelées électrodes et séparées par un matériel isolant ou diélectrique.

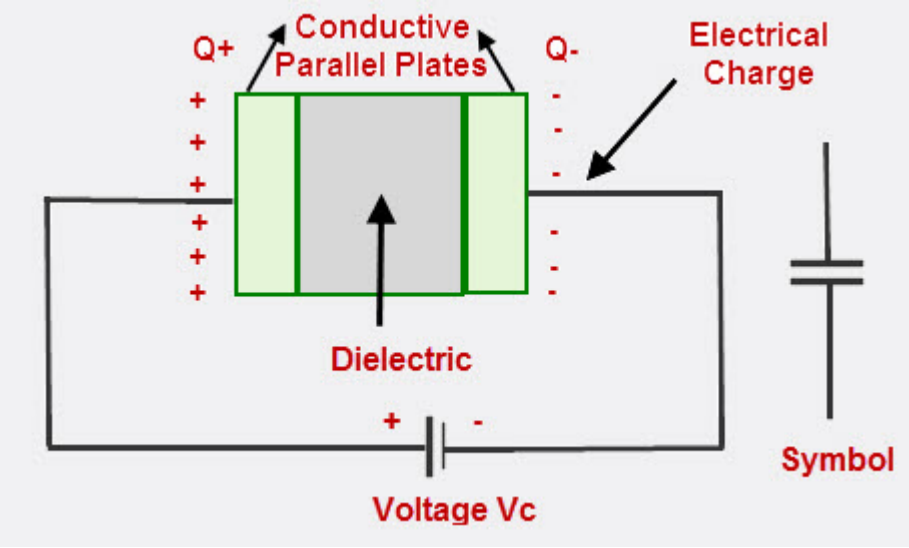


FIGURE 13.1 – schéma condensateur

Lorsque l'on exerce une tension sur un condensateur, une force électrique déplace des électrons vers la première armature pour s'y déposer. Cette augmentation du nombre d'électrons vient chargée négativement l'armature. Une force se créée entre les deux plaques et vient arracher des électrons à la seconde armature et donc charger positivement l'armature.

Malgré la présence d'un isolant entre les deux plaques, le courant dans le circuit n'est pas nul. En effet, tant que le condensateur n'est pas chargé ,un nombre d'électrons arrive vers le condensateur. Le nombre d'électrons arrivant sur la première plaque est égale au nombre d'électrons quittant la seconde plaque. Des charges sont arrachées à la seconde armature et continuent de se déplacer dans le circuit.

Le condensateur continue de se charger tant que la tension entre les deux armatures n'est pas égale à la tension exercée sur ses bornes. Si on exerce une tension sur le condensateur supérieur à sa tension admissible, le composant va casser ou exploser. Lorsque le condensateur est complètement chargé, les nouveaux électrons arrivant sont repoussés par ceux déjà présents sur la plaque, il n'y

a plus déplacement de charges , le courant devient nul.

Un condensateur est caractérisé par un coefficient de proportionnalité entre la charge et la tension à ses bornes. On note ce coefficient capacité électrique et il s'exprime en Farad.

$$Q = C(V_1 - V_2) \;\; ou \;\; i = C\frac{du}{dt}$$

La capacité d'un condensateur est déterminée par la géométrie du composant et la nature de l'isolant.

Le tableau suivant résume la valeur de capacité pour différente géométrie de condensateur. ε_r représente la permittivité relative de l'isolant.

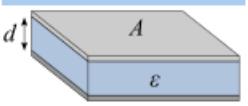
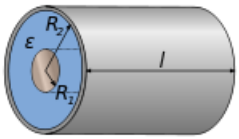
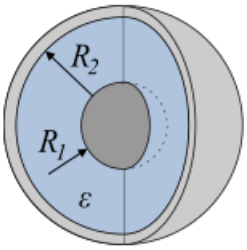
Désignation	Capacité	Champ électrique	Représentation
Condensateur plan	$C = \varepsilon_0\varepsilon_r \cdot \frac{A}{d}$	$E = \frac{Q}{\varepsilon_0\varepsilon_r A}$	
Condensateur cylindrique	$C = 2\pi\varepsilon_0\varepsilon_r \frac{l}{\ln\left(\frac{R_2}{R_1}\right)}$	$E(r) = \frac{Q}{2\pi r l \varepsilon_0\varepsilon_r}$	
Condensateur sphérique	$C = 4\pi\varepsilon_0\varepsilon_r\left(\frac{1}{R_1} - \frac{1}{R_2}\right)^{-1}$	$E(r) = \frac{Q}{4\pi r^2\varepsilon_0\varepsilon_r}$	
Sphère	$C = 4\pi\varepsilon_0\varepsilon_r R_1$		

FIGURE 13.2 – calcul capacité

SECTION 14

LES DIFFÉRENTES TECHNOLOGIES

Au cours de l'histoire plusieurs technologies ont été découvertes afin de fabriquer des condensateurs. Cela nous offre maintenant, un large choix de technologies suivant les utilisations et, ou les valeurs en Farad nécessaires. Afin de mieux comprendre les avantages et les inconvénients de chaque technologie, nous allons en faire une comparaison des plus couramment utilisées.

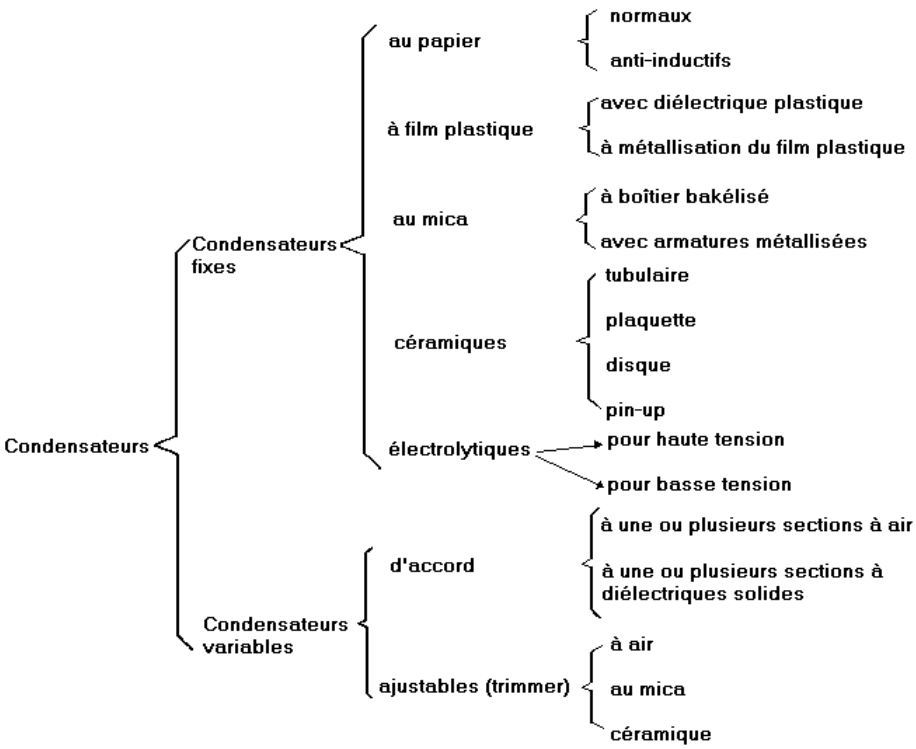


FIGURE 14.1 – Classification des différents types de condensateur

La figure 14.1 récapitule tous les types de condensateurs. Ils sont divisés en 2 catégories, les condensateurs à capacité fixes et ceux à capacité variable. Nous allons par la suite nous intéresser majoritairement au condensateur à capacité fixe car ce sont ceux le plus répandus.

Les condensateurs à film

Ces condensateurs utilisent un film plastique. Il en existe de 2 types, suivant l'utilisation du film plastique. Ceux de type un utilisent un film plastique comme diélectrique, c'est-à-dire comme moyen de limiter ou empêcher la conduction électrique mais laissant s'exercer les forces électrostatiques. Ceux de type deux utilisent quant à eux un film plastique métallisé.

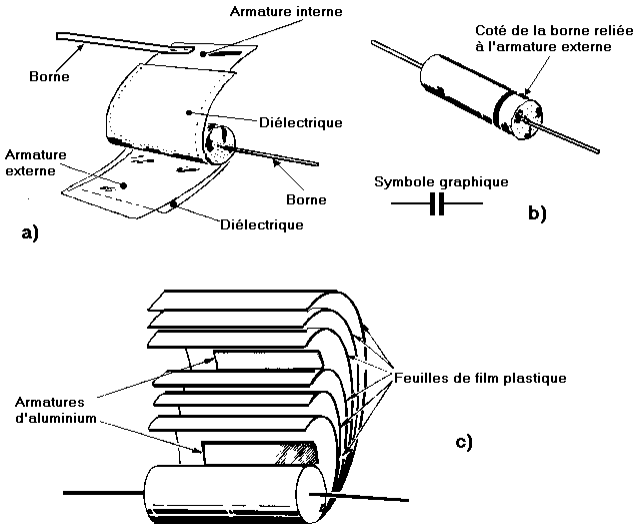


FIGURE 14.2 – Schéma interne d'un condensateur à film plastique

Ces condensateurs possèdent une faible capacité comprise entre 1nF et 30µF, mais à défaut permettent une très grande précision. Ils possèdent également une durée de vie supérieure à la plupart des autres types de condensateurs et ne sont pas polarisé. De plus, certains de ces condensateurs permettent une régénération après un claquage. Ils sont également conçus pour résister à de hautes tensions (de l'ordre du kilovolt) et permettent de fournir des impulsions de courant de surcharge très élevées.

Les condensateurs à céramique

Ce type de condensateur est celui majoritairement utilisé. Ils sont notamment très utilisés en électronique du fait de leur petite taille. Ils permettent des capacités comprises entre 1nF et 1µF. Ils ne sont pas polarisés comme les condensateurs à film ce qui permet leur utilisation dans des circuits à courant alternatif. Il en existe également 2 types :

- Ceux de classe 1 sont utilisés lorsque l'on nécessite une grande stabilité et de faible perte. Ils possèdent une valeur de capacité très stable.

- Ceux de classe 2 possèdent une capacité élevée. Ils perdent cependant en stabilité thermique et les tolérances de capacité sont plus élevée.

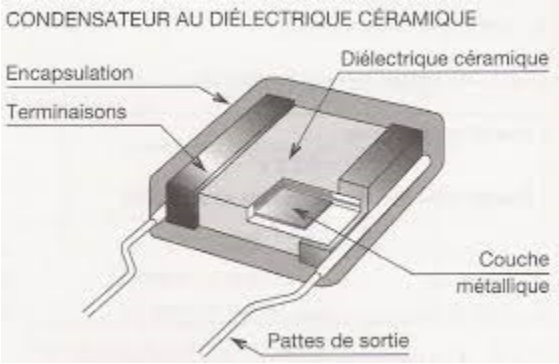


FIGURE 14.3 – Schéma interne d'un condensateur à céramique

Les condensateurs électrolytiques

Les condensateurs électrolytiques ou encore condensateurs chimiques, utilisent un électrolyte, c'est-à-dire une substance conductrice contenant des ions mobiles. Cela permet à ces condensateurs de pouvoir fournir une plus grande plage de valeur de capacité que les autres types de condensateur. Ils sont dans la très grande majorité polarisé ce qui oblige leur utilisation dans des circuits à courant continu. On en voit très souvent dans des alimentations notamment d'ordinateur etc... Ils permettent une capacité comprise entre 1µF et 47mF avec une tension de fonctionnement pouvant aller à l'ordre de quelques centaines de volts. Ils ne sont cependant pas très précis. En effet il possède une tolérance de 20%, une grande résistance en série et réagissent très mal aux hautes fréquences (surchauffe).

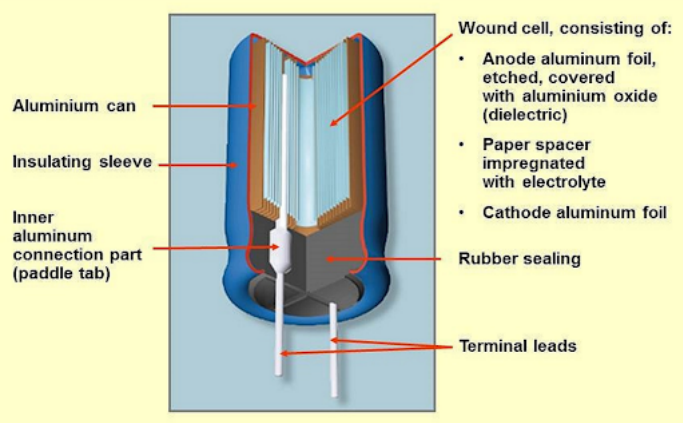


FIGURE 14.4 – Schéma interne d'un condensateur électrolytiques

Les condensateurs variables

Les condensateurs variables sont des condensateurs dont la valeur de la capacité est comme son nom l'indique variable. Ils sont constitués d'un rotor, d'un axe et d'un stator.

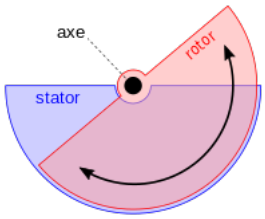


FIGURE 14.5 – Schéma interne d'un condensateur variable

Le rotor entraîné par l'axe tourne dans l'armature fixe du stator. La capacité de ces condensateurs varie en continu entre une valeur minimum appelé capacité résiduelle et une valeur maximum appelée capacité nominale.

Cette variation suit une fonction appelée loi de variation :

$$C = f(\theta) \tag{14.1}$$

f() dépend de multiples paramètres complexes. Il est donc difficile de pouvoir le définir clairement.
θ quant à lui correspond à l'angle de rotation de l'axe.

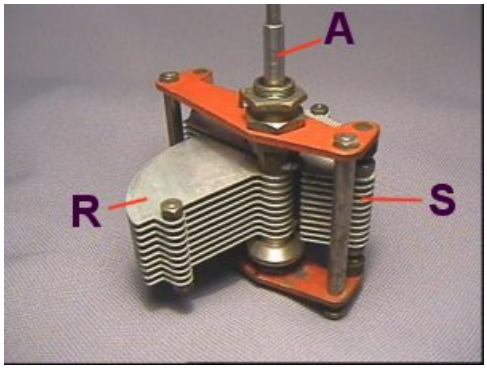


FIGURE 14.6 – Condensateur à capacité variable

Nous avons pu dans cette partie, nous intéresser à différents types de condensateur. Il en existe encore plein d'autres comme les condensateurs au mica, au papier... Mais nous avons vu ici, ceux les plus couramment utilisés.

SECTION 15

DOMAINES D'APPLICATION

Nous allons présenter trois domaines d'application des condensateurs. Cette liste n'est pas exhaustive. Les condensateurs permettent également de redresser le cosinus Φ ou bien de filtrer des signaux audio.

Les condensateurs de filtrage

Présentation

Les condensateurs permettent de lisser une tension afin de stabiliser cette dernière.

Objectif

On souhaite amortir les oscillations de tension en sortie d'un pont de Graëtz (14 V DC). On souhaite donc passer d'une tension alternative à une tension pseudo-continue.

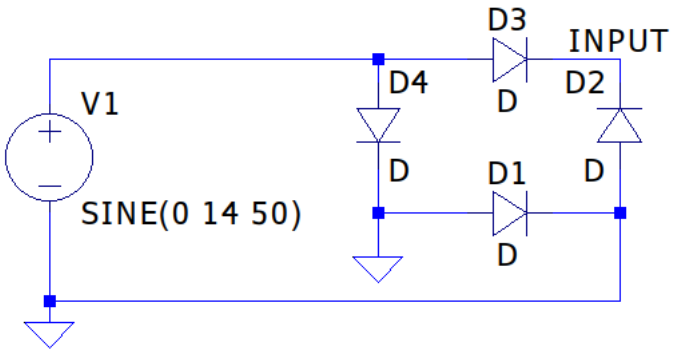


FIGURE 15.1 – Le pont de Graetz

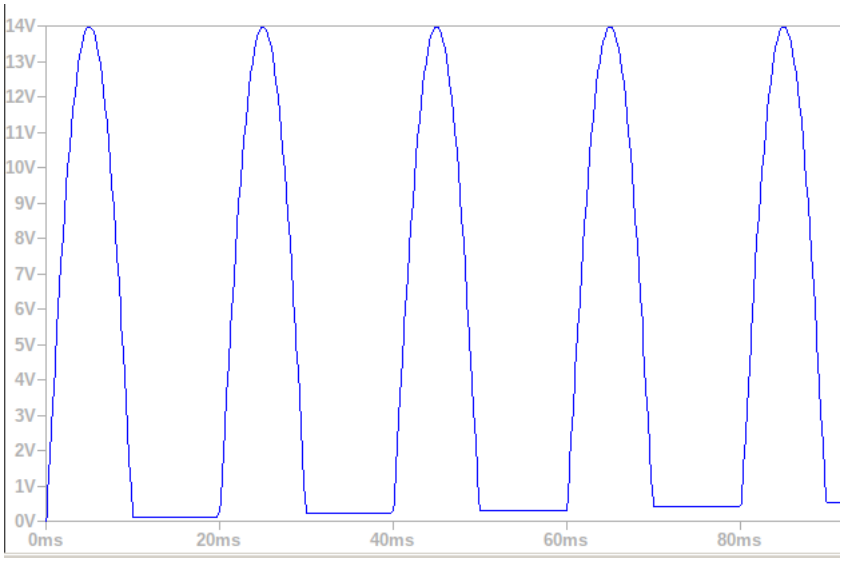


FIGURE 15.2 – La tension en sortie du pont

Mise en oeuvre

Afin de lisser la tension, il convient de mettre un condensateur en sortie du pont de Graëtz, avec une borne sur INPUT et une autre à la masse.
Une résistance de 50 Ω a été rajoutée pour simuler la présence d'une charge.

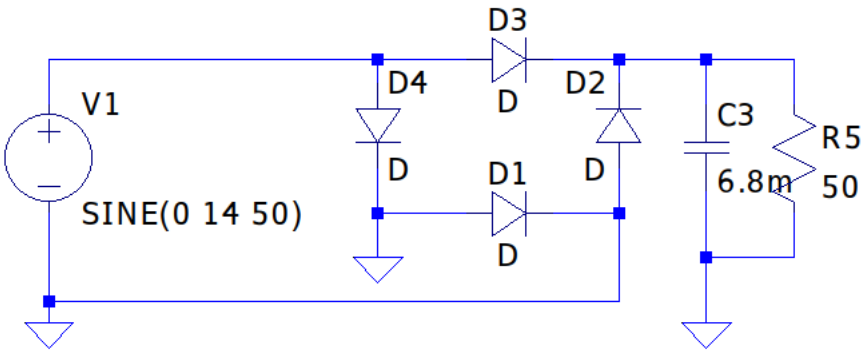


FIGURE 15.3 – Le circuit de filtrage

Observons le résultat.

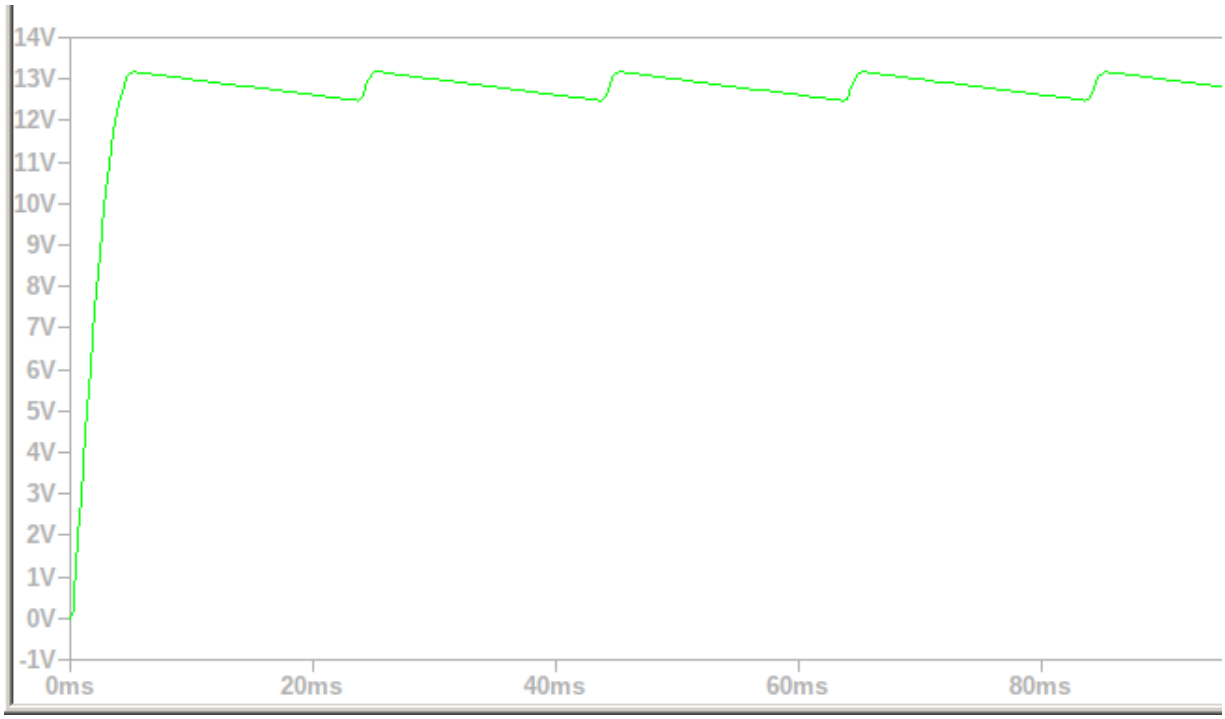


FIGURE 15.4 – La tension aux bornes du condensateur

Principe

Le condensateur joue le rôle d'accumulateur. Pendant les phases ou la tension est croissante, ce dernier se charge.
Lors des phases ou la tension décroît, le condensateur restitue une partie de son énergie, ce qui a pour effet de réduire l'amplitude de variation de tension.

On retrouve notamment ces condensateurs dans les alimentations linéaires.
Dans ce cas, la capacité du condensateur nécessaire croit avec le courant demandé par la charge.

Il est fréquent de voire des condensateurs chimiques de quelques mF dans ces alimentations. Une valeur inférieure risque de rendre instable l'alimentation.

Les condensateurs de découplage

Présentation

Certains circuits nécessitent une alimentation très stable (aucune fluctuation de la tension d'entrée). Cependant, lors de l'utilisation du circuit, des courants importants peuvent être demandés par le circuit.
Nous allons montrer un cas en exemple où, pour améliorer les performances du circuit, il conviendra de mettre de fameux condensateur de découplage.

Objectif

Nous souhaitons cascader deux portes logiques CMOS [1] de type inverseur, comme le montre la figure suivante.

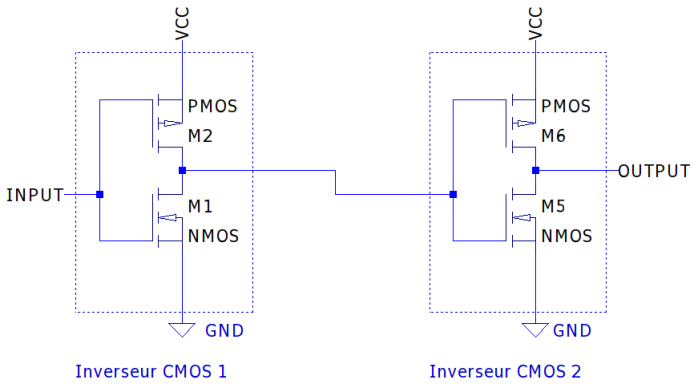


FIGURE 15.5 – Schéma d'exploitation

A première vue, ce modèle peut sembler satisfaisant. Cependant, nous souhaitons faire fonctionner ce circuit dans des fréquences assez élevée [2].

En prenant un modèle réel, on peut déjà prendre en compte les capacités parasites des transistors MOSFETs. Ces derniers possèdent une capacité entre la broche de commande (Gate) et la masse.

Le modèle réel des entrées de chaque porte logique est le suivant.

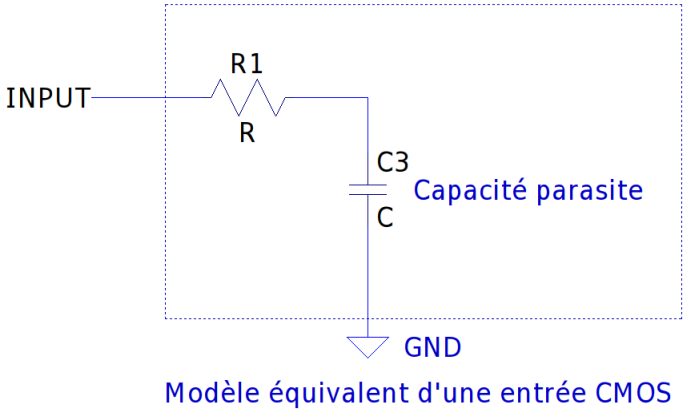


FIGURE 15.6 – Modèle équivalent des entrées CMOS

1. Complementary MOS : MOS complémentaires, mise en cascade d'un MOS canal P et d'un MOS canal N
2. Toutefois inférieures à la fréquence maximale du circuit

En régime transitoire

Intéressons nous au régime transitoire.
On constate que dans notre cas d'exemple, les grilles des MOSFETS consomment du courant pendant les phases de commutation, du fait qu'il faut charger ou décharger les condensateurs. Cet appel de courant peut faire réduire la tension d'alimentation si cette dernière n'est pas très puissante.

En régime stationnaire

Lorsque la tension d'entrée (INPUT) est constante et que le condensateur est chargé (ou déchargé), les grilles des MOSFETs ne sont parcourus par aucun courant

Le modèle réel

L'alimentation étant rarement en liaison directe avec le circuit logique (piste de circuit imprimé, fils d'alimentation), on constate l'apparition d'une inductance parasite entre la borne d'alimentation du circuit et l'alimentation en elle-même. Le modèle équivalent final du circuit peut être modélisé par la figure suivante.

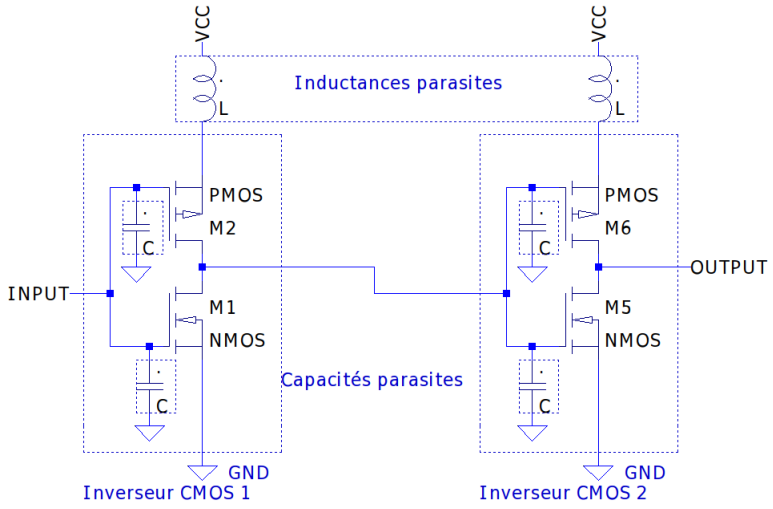


FIGURE 15.7 – Modèle équivalent du circuit

L'inductance parasite s'oppose aux variations de tensions.
Ainsi, lors des phases de commutation (régime transitoire), une partie de la tension d'alimentation va au bornes des inductances parasites. De ce fait, le circuit est de moins en moins efficace à mesure que l'on augmente la fréquence du signal d'entrée. Pour pallier à ce problème, nous allons ajouter un condensateur de découplage.

Mise en oeuvre du condensateur de découplage

Le condensateur de découplage doit être placé au plus près du circuit à alimenter. Une de ses bornes est reliée à la broche d'alimentation du circuit et la seconde à la masse.

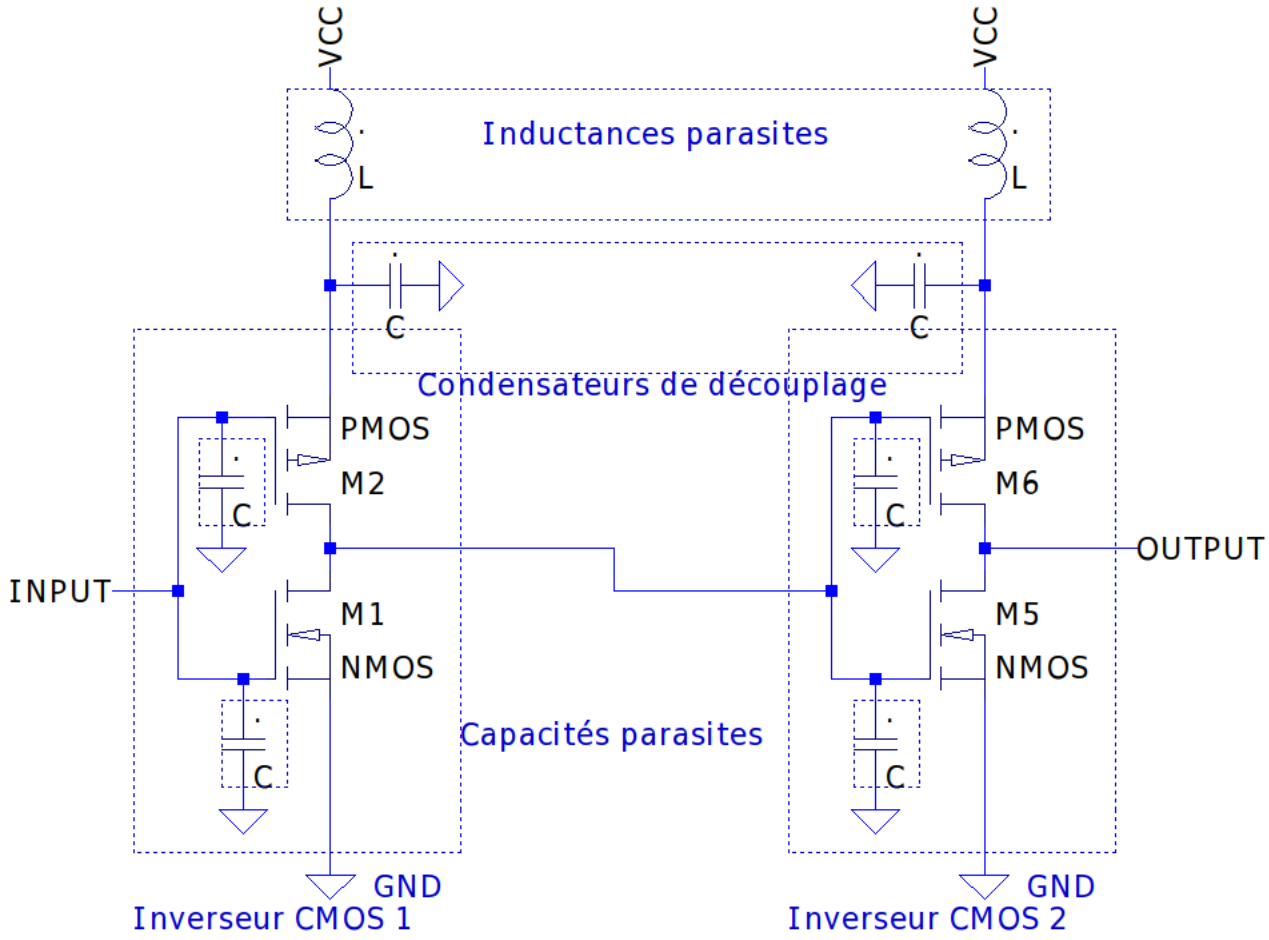


FIGURE 15.8 – Le placement du condensateur de découplage

Lorsque le circuit en aval va demander du courant lors des commutations, le condensateur, qui va se charger pendant le régime permanent du circuit, va pouvoir fournir un apport de courant qui évitera à la tension d'alimentation de s'écrouler. Le circuit sera plus efficace et les temps de communication seront plus faibles.

D'un point de vue des filtres, ces condensateurs peuvent être considérés comme des filtres passe-bas. En effet, la tension d'alimentation qui varie est vue comme un signal haute fréquence. Il faut donc éviter l'oscillation haute fréquence.

Les condensateurs de liaison

Présentation

Une des propriétés des condensateurs est de bloquer les composantes continues. Nous allons aborder un exemple ou le condensateur va permettre de se passer d'une alimentation symétrique.

Objectif

On souhaite amplifier la composante alternative d'un signal d'amplitude 1V, de tension moyenne 0.5V et de fréquence 1kHz avec un AOP alimenté en 0-12V (Alimentation asymétrique).

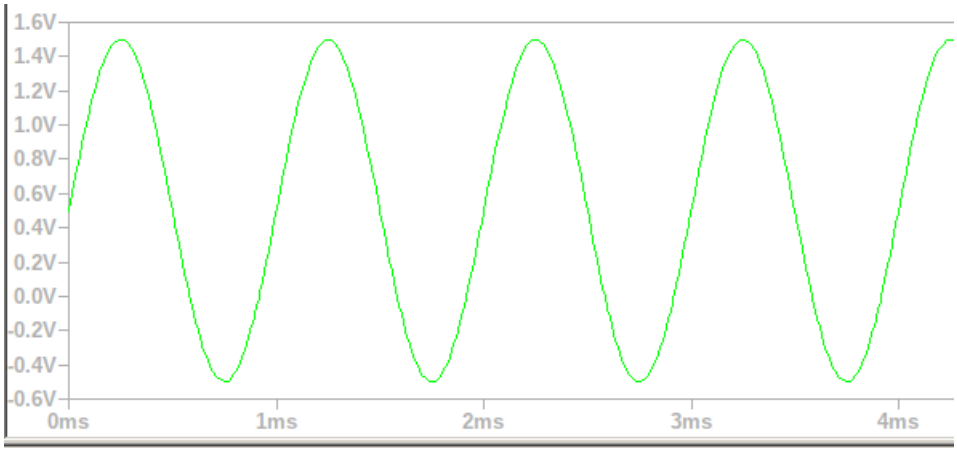


FIGURE 15.9 – Le signal à amplifier

Mise en oeuvre

Tout d'abord, on va chercher à recentrer le signal pour avoir une composante alternative toujours positive. Pour cela, on utilise le circuit suivant.

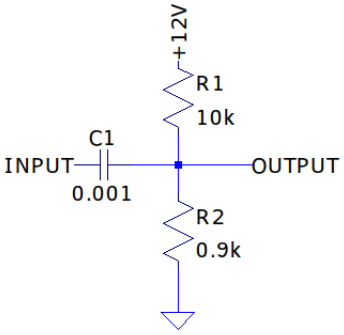


FIGURE 15.10 – Montage pour recentrer la tension

La fréquence de coupure valant $\frac{1}{2\pi RC}$ avec $R < 10k\Omega$, il faudra prendre un condensateur tel que la fréquence de coupure soit inférieure à la fréquence du signal d'entrée.

Un condensateur de 1 mF conviendra donc pour cette application.

En utilisant le théorème de superposition au point OUTPUT, on en déduit que une résistance de $10k\Omega$ et une résistance de 900Ω, on obtient la tension OUTPUT suivante.

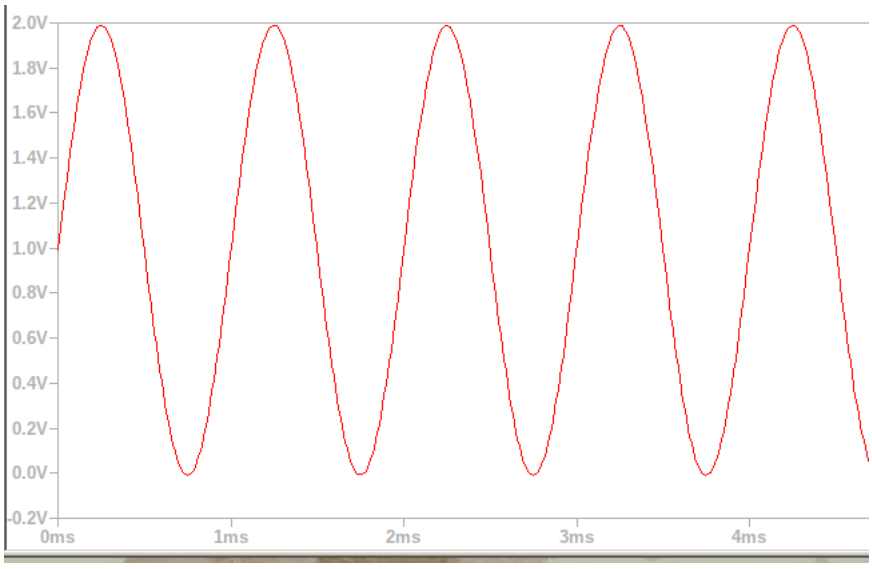


FIGURE 15.11 – Tension positive

Il nous reste à amplifier la tension OUTPUT (ici par 2). Nous obtenons donc une tension sinusoïdale d'amplitude 2V.

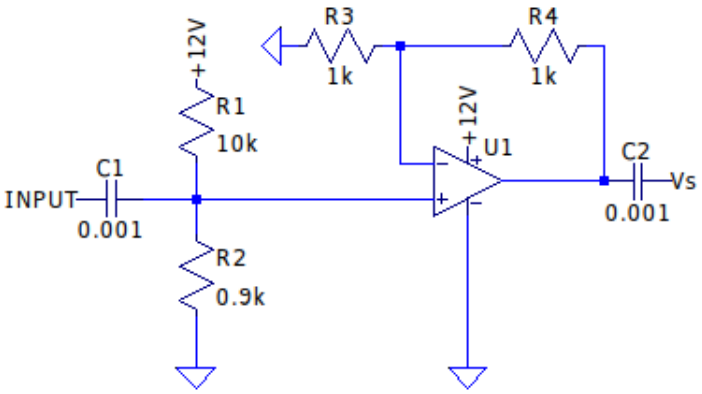


FIGURE 15.12 – Schéma avec les deux condensateurs de liaison

Pour extraire la composante alternative, il suffit de mettre en sortie un condensateur qui va recentrer le signal en 0V. Le condensateur sera de 1mF (même fréquence).
Nous obtenons finalement un signal de sortie avec un gain de 2 de la composante alternative d'entrée.

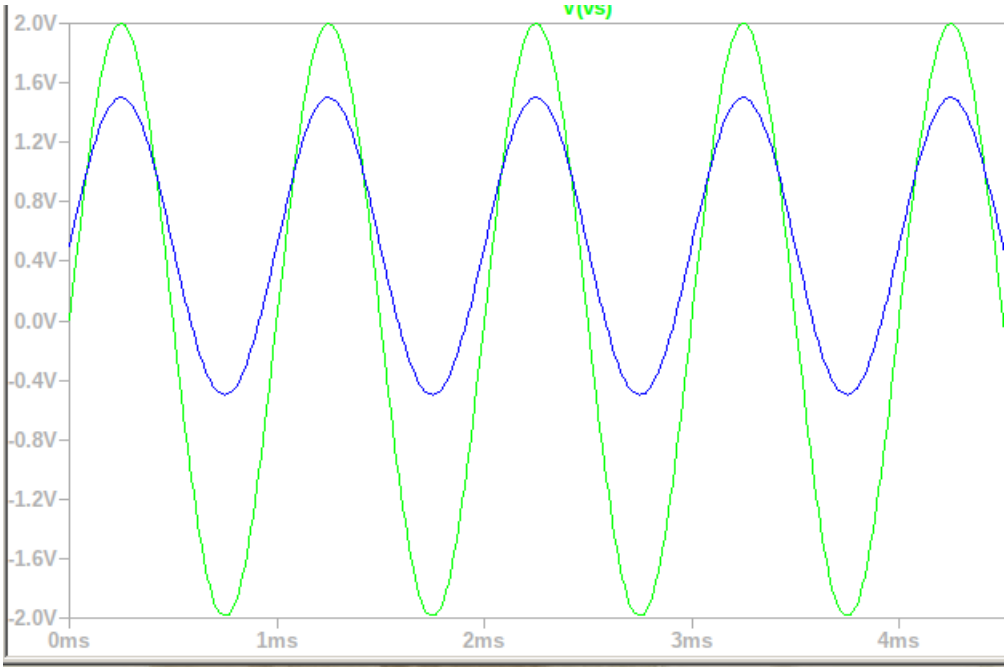


FIGURE 15.13 – La tension de sortie

Au final, les deux condensateurs permettent à des portions de circuits de communiquer entre elles avec des niveaux de tension différents. Ces condensateurs sont appelés **condensateurs de liaison ou de couplage**

SECTION 16

AVENIR DU CONDENSATEUR

Nous avons donc vu dans les parties précédentes que le condensateur est un élément essentielles à tout circuits électronique.
Nous nous sommes donc posé la question suivante : Le condensateur sera-t-il amené à disparaître dans le futur pour être remplacé par un autre composant ?

Il est évident que l'importance du condensateur est telle qu'il parait irremplaçable.
En effet, on ne peut se passer d'un telle composant dans un circuit électronique.
Cependant, depuis quelques années un composant est utilisé de plus en plus : le super-condensateur.

Les supercondensateurs

Les supercondensateurs sont une sous-catégorie des condensateurs électrolytiques.
Ils permettent de stocker une très grande quantité d'énergie grâce à une combinaison de 2 technologies de capacité.

La capacité double couche que l'on retrouve dans les condensateurs électrolytiques et la pseudo-capacité.
L'une est électrostatique et l'autre électrochimique.

Cela leur permet de combiner les caractéristiques des condensateurs ordinaires et celles des batteries.
En effet grâce à ces technologies ils peuvent atteindre des capacités allant jusqu'à 12 000F, tout en ayant des temps de charge et décharge très rapide comparable aux condensateurs ordinaires.
Toutes ces caractéristiques en feraient de bons candidats pour remplacer nos batteries au lithium.

Mais c'était sans compter leurs défauts. Ils sont en effet très chers à produire, possèdent une faible énergie spécifique (Rapport en Wh/kg entre l'énergie électrique fournie par unité de temps et la masse du convertisseur) et une tension de décharge linéaire.
Cela entraînerait de grandes chutes de tension d'alimentation très rapidement.

Le supercondensateur sera probablement l'avenir pour le stockage d'électricités, qui a terme remplacera sûrement les batteries des véhicules électriques.

Cependant, il est trop puissant pour le mettre dans la majorité des circuits électroniques. La science progresse énormément dans ce domaine.

Troisième partie

Les interfaces de puissance

Théorie sur les interfaces de puissance et applications pratiques avec Arduino

SECTION 17

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 prévu pour délivrer de **faibles courants** et **faibles tensions** . Nous allons donc créer un circuit où la puissance et la commande sont dissociés.

SECTION 18

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)

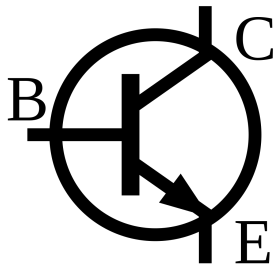


FIGURE 18.1 – La représentation du transistor bipolaire

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

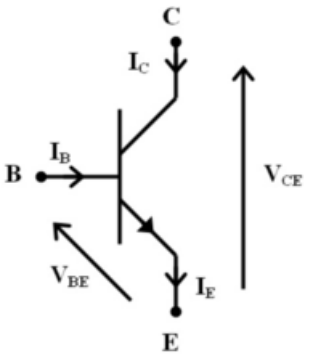


FIGURE 18.2 – Conventions du transistor bipolaire

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 NPN [1]
- Les transitors 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.

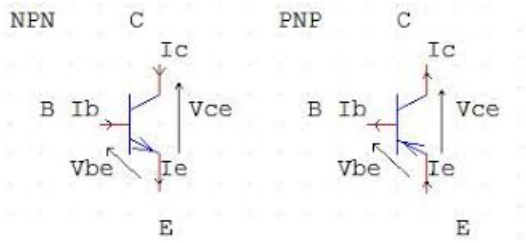


FIGURE 18.3 – Transistors NPN et PNP

Les paramètres de sélection du transistor

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

- La tension admissible sur V_{ce} [2] doit être supérieur 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.

1. Le nom de ces familles provient du type de jonction utilisé en interne. Pour plus de rensiegnement, consulter les diodes et semi-conducteurs
2. Cette tension est indiquée ddans les documentations techniques

- 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 1 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.

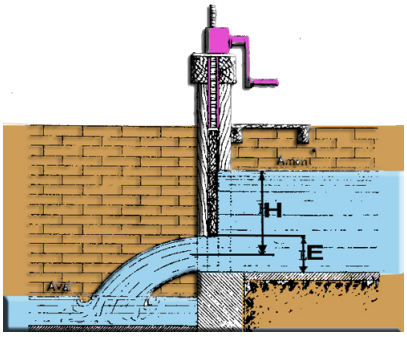


FIGURE 18.4 – Le rôle du transistor

.

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.

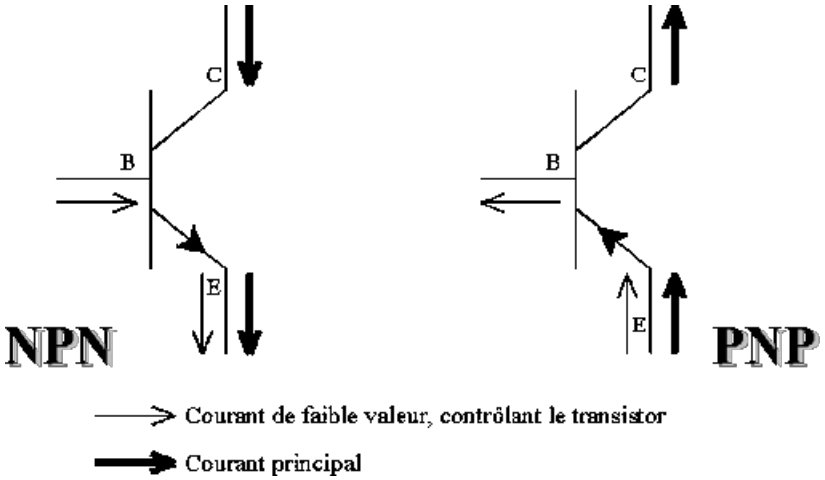


FIGURE 18.5 – Courant de commande et de puissance

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

$$\boxed{I_C = \beta \cdot I_B}$$

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 possède 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.

Remarque

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 :

$$\boxed{I_E = I_B + I_C}$$

Exemple

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

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 $\beta = 30$

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.

Remarque

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.

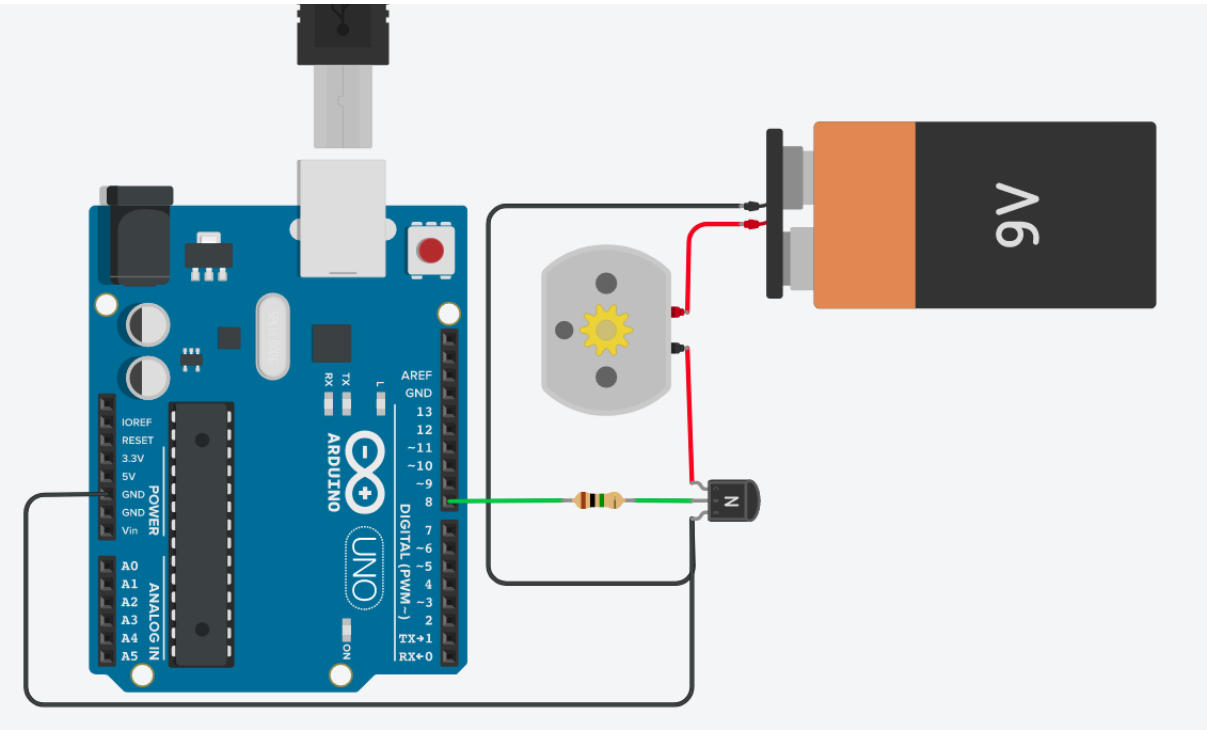


FIGURE 18.6 – Branchement du transistor bipolaire

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.

Nous somme 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.

Remarque

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

a. Le transistor laisse passer tout le courant nécessaire ou rien du tout

On refait donc les calculs.

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

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

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 :

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

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() {

  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 Arduino avec transistors NPN

SECTION 19

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 MOSFET [1] 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)

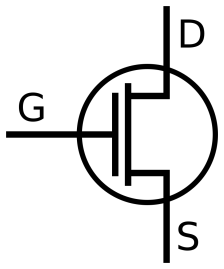


FIGURE 19.1 – La représentation du transistor MOSFET

Conventions

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

- Le courant entrant dans le Drain est appelé I_D

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 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 N [3]
- Les transitors 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.

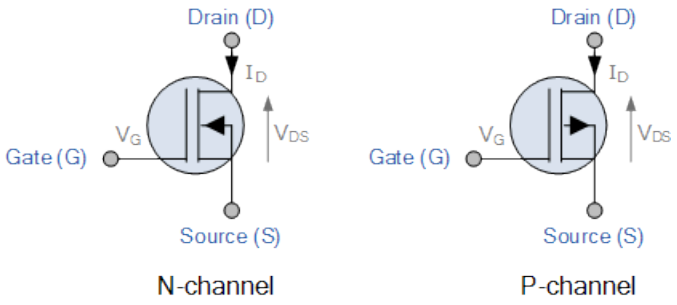


FIGURE 19.2 – Transistors à canal N et P

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 rensiegnement, 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 passant [4] 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. [5]

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

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

4. Le transistor laisse passer tout le courant autorisé.
5. Lorsque la tension V_{GS} est inférieure à $V_{GS_{th}}$, I_D vaut $K \cdot ((V_{GS} - V_{th}) \cdot V_{DS} - \frac{1}{2}V_{DS}^2)$
Cette relation montre que l'étude en amplification est plus complexe car non linéaire.

Remarque

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

Enfin, il ne nous reste plus qu'à relier une sortie numérique de l'Arduino vers notre porte **sans** résistance.

Nous obtenons donc le schéma suivant.

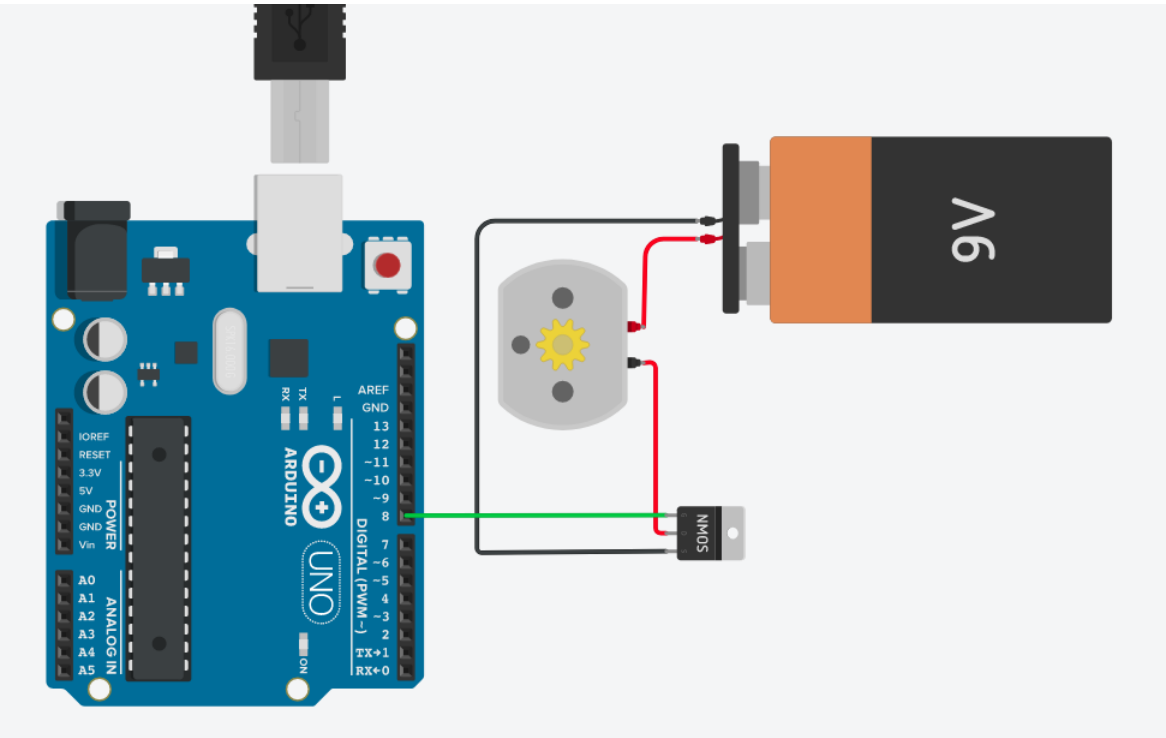


FIGURE 19.3 – Branchement du transistor MOSFET

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 11     //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 Arduino avec MOSFET

SECTION 20

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 transitors sont adaptés et prennent moins de place en encombrement.

Les fiches techniques

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

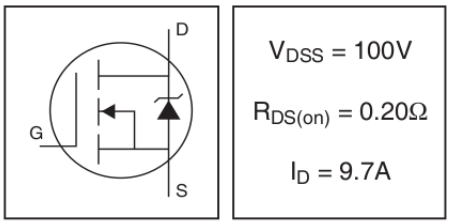
Ce document détaille les broches, les caractéristiques électriques, propose des schémas d'exemples....

Par exemple, voici quelques extraits de la documentation du transistors IRF520 [1] :

1. Transitor de puissance

IRF520NPbF

HEXFET® Power MOSFET



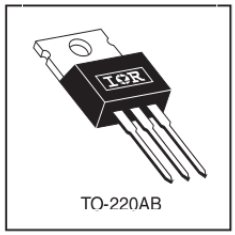


FIGURE 20.1 – Extrait n°1 du IRF520

Electrical Characteristics @ T_J = 25°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 = 250µA
$\Delta V_{(BR)DSS}/\Delta T_J$	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 = 250µA

FIGURE 20.2 – Extrait n°2 du IRF520

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

Quatrième partie

Les circuits logiques

Théorie sur les circuits logiques

SECTION 21

LES FAMILLES DES CIRCUITS LOGIQUES

Présentation

Les circuits logiques sont des éléments permettant de réaliser des opérations avec l'algèbre de Boole (uniquement des '0' et des '1'). Pour réaliser ces opérations (ET, NON, OU, NON-OU, NON-ET...), des circuits ont vu le jours dans les années 60 avec deux grandes familles de circuits :

- La famille TTL
- La famille CMOS

De nouvelles technologies arrivent à maturité mais nous ne les évoquerons pas ici.

Principe TTL

Les circuits TTL sont composées de transistors bipolaires NPN ou PNP.
Les transistors bipolaires sont commandés en courant.

Principe CMOS

La famille CMOS, quant à elle, repose sur l'utilisation en interne de transistors MOS complémentaires (C).
Les transistors MOS, du fait de leur structure, sont commandés en tension.

Comment les distinguer ?

La famille des CMOS est rapidement identifiable car le nom du composant contient un numéro commençant par 40 et se termine avec un nombre à 2 ou 3 chiffres (40XX ou 40XXX).

Par exemple, CD4001, CD4017 sont des composants CMOS.

La famille des TTL contient en général le chiffre 74 [1] encadré par des lettres et des chiffres.

Avantages et inconvénients

- Les TTL consomment plus de courant que les CMOS [2]
- Les CMOS ont des fréquences de fonctionnement plus faibles que les TTL.

1. La série militaire des TTL possède le numéro 54 et possède de meilleurs caractéristiques : plage de température de fonctionnement plus élevée, fréquence plus élevée...
2. Ces derniers consomment uniquement lors des phases de commutation

Cinquième partie

Les signaux

Théorie sur les signaux et leur traitement

SECTION 22

LES SIGNAUX

Introduction

Un signal est une évolution d'une grandeur physique.
Il permet donc de caractériser cette grandeur qui peut ensuite être traitée par un circuit adapté.

- Les signaux numériques
- Les signaux analogiques

Les grandeurs physiques

Afin de mieux comprendre comment un signal peut se former, nous allons étudier les notions de tensions, de courant et de résistances électriques. Pour cela, plongeons nous dans la matière, au niveau des électrons

Les atomes

Les électrons

Les électrons forment la couche extérieure de l'atome, élement insécable de notre univers.
Lorsqu'un électron se déplace dans l'espace, ce dernier engendre un courant.

La charge électrique représente quant à elle le nombre d'électron à un endroit donné et à un temps précis.

La tension électrique

Nous avons vu que le courant est une nombre de charges par unité de temps.
Comment pouvons nous engendrer ce courant ? Tout simplement en soumettant une différence de potentiel, autremeent dit une **tension**

Analogie

Afin de mieux comprendre ce principe, faisont une analogie.
Prenons un rivière, un barrage et une montagne.

Soit deux points sur une route dont nous pouvons régler la hauteur en chaque point. On considère une goutte d'eau entre ces points.
Notre objectif est de déplacer notre goutte d'un point à l'autre.

» Comment faire ?

En surelevant le point A et en abaisant notr point B , notre goutte va descendre du point le plus élevé au plus bas.

Le courant fonctionne de la même manière :

Il ira toujours du potentiel le plus haut au plus faible.
Pour la petite histoire, historiquement, le sens du courant étaient du potentiel le plus elevé au plus faible.
Bien des années plus tard, les scientifiques se sont rendu compte que le courant allait en réalité du potentiel le plus faible au plus elevé. Néanmoins, la convention a été gardée...

Exemples

Question 4. *Soient A et B nos deux points du circuits. Ces deux points sont soumis à une tension U_A et U_B*
Quel est le sens de parcours du courant (de A vers B , de B vers A ou bien autre chose) ?

U_A (V)	U_B (V)	Sens du courant ?
10	5	
5	10	
5	5	

FIGURE 22.1 – Question sur le sens du courant en fonction des tensions U_A et U_B

Sixième partie

Les alimentations

Théorie sur les alimentations linéaires

Introduction

Il existe deux grandes familles d'alimentations :

- Les alimentations linéaires
- Les alimentations à découpages

Les alimentations linaires sont réputées pour avoir une tension stable et propre (tension continue).
Regardons de plus près le principe de ces alimentations.

L'objectif

On souhaite transformer une tension sinusoïdale [1] en une tension continue, stable et régulée.

1. De valeur efficace 230V et de fréquence 50 Hz

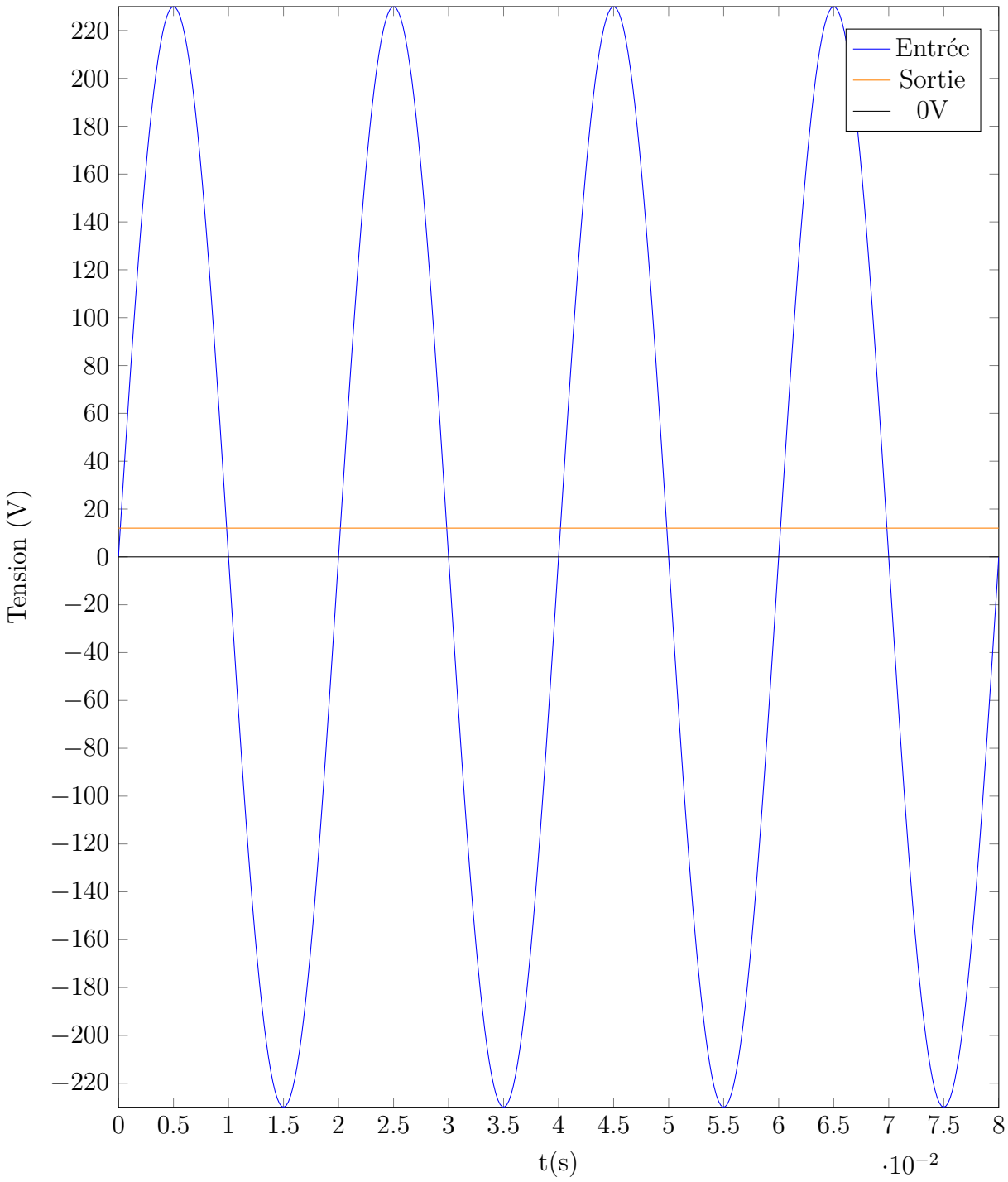


FIGURE 22.2 – Les tensions d'entrée et de sortie de notre alimentation

Le choix du transformateur

Pour notre circuit, nous devons connaître la tension finale en sortie de notre alimentation. Au vue des composants demandés (régulateurs, pont de diode), il faudra une tension efficace de $12V$ [2] Le rapport des bobines de notre transformateur sera donc au minimum de **19** .
Cela veut dire que pour 19 enroulements dans la bobine primaire, il en faudra une seule dans la bobine secondaire.

La puissance demandée pour le transformateur n'est pas critique, la puissance absorbée par le circuit en aval est négligeable.

Notre transformateur va donc abaisser la tension du secteur en une tension plus faible.

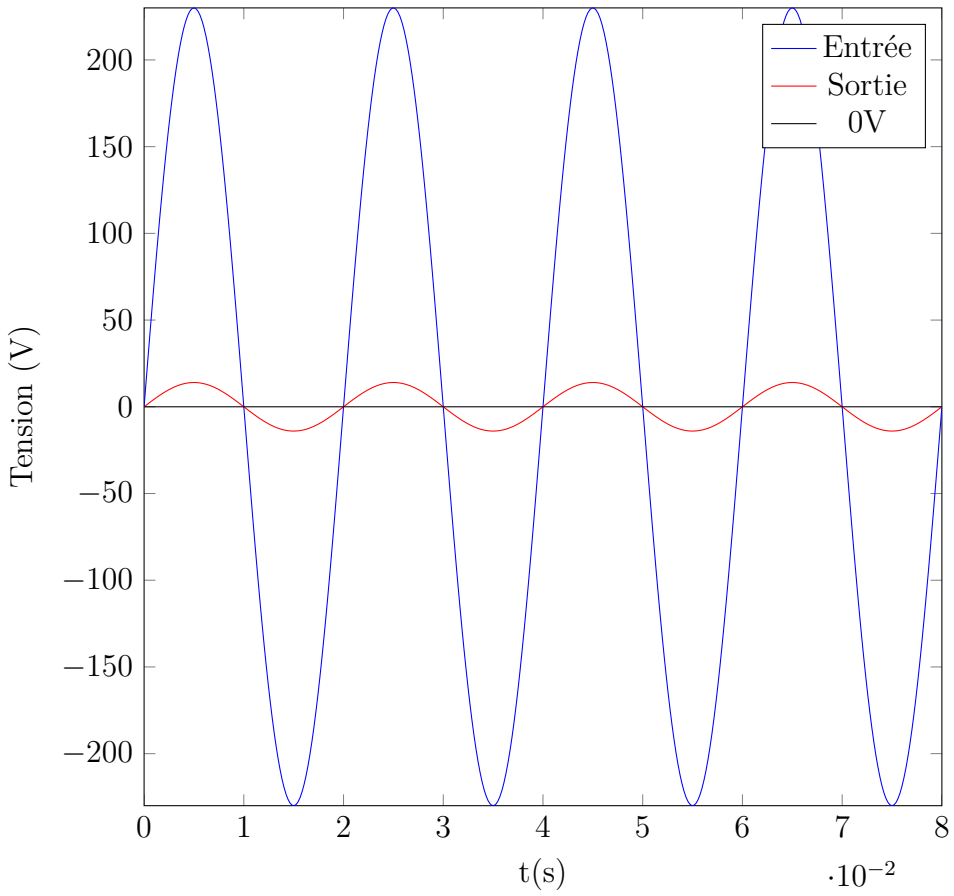


FIGURE 22.3 – Les tensions d'entrée et de sortie du transformateur

2. Nous verrons la justification de cette valeur par la suite.

Le pont redresseur de tension

Principe

Notre objectif est maintenant de supprimer la composante négative de notre signal. Pour cela, on va utiliser la propriété des diodes qui ont la faculté de laisser passer le courant dans un seul sens.
Un montage existant est le Pont de Graetz.

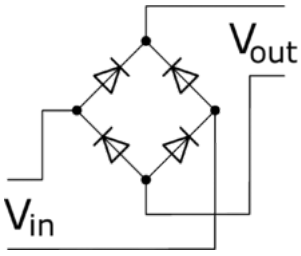


FIGURE 22.4 – Le pont de diode

Observons le résultat lorsque nous mettons notre tension de sortie du transformateur sur notre entrée de pont.

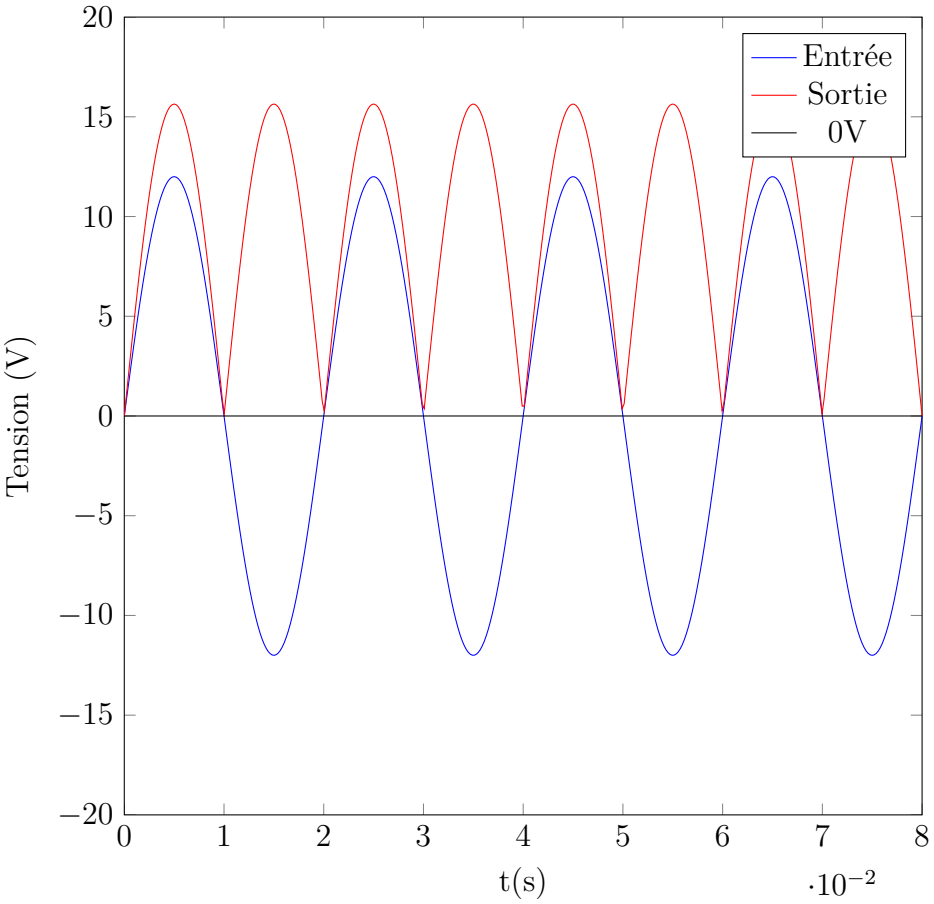


FIGURE 22.5 – Les tensions d'entrée et de sortie du pont de diode

On constate que la tension de sortie est plus élevée que notre tension d'entrée.
Cette valeur s'explique par les diodes.

En sortie, nous voulons une tension de $15V$. Or, les $12V$ en entrée de notre pont sont une valeur efficace, ce qui veut dire que la valeur réelle vaut $17.04V$ En effet :

Or, à chaque changement de polarité dans notre pont (signal alternatif [positif et négatif]), le courant passe à travers deux diodes.

Chaque diode engendre une chute de tension de $0.7V$, d'où une perte de $1.4V$ en sortie.
Nous avons comme tension finale en sortie de pont :

$$V_s = V_{efficace} \cdot \sqrt{2} - 0.7 \cdot 2 = 12 \cdot 1.41 - 1.4 = 15.64V$$

Un exemple de boîtier

La plupart des boîtiers possèdent 4 broches

- deux broches V pour la tension alternative
- Une broche V_- pour la masse du circuit en aval
- Une broche V_+ pour la tension positive du circuit en aval

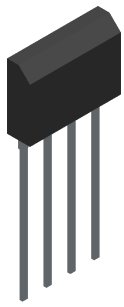


FIGURE 22.6 – Un exemple de boîtier

Les condensateurs

Principe

Comme vu à la section précédente, notre signal n'est pas continu et possède des oscillations élevées. Nous cherchons à produire une tension continue.

Pour cela, nous allons utiliser des condensateurs.

Leur rôle sera d'accumuler de l'énergie lorsque le signal monte en amplitude et de restituer cette énergie lorsque le signal sera sur sa phase descendante.

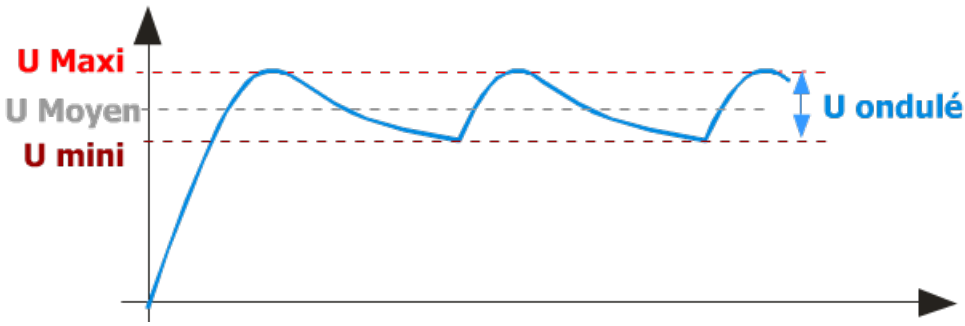


FIGURE 22.7 – Le rôle du condensateur

Choix de la valeur

Plus la valeur du condensateur est élevée, plus notre signal sera lisse en sortie, ce qui est l'effet recherché.
En effet, la capacité du condensateur exprime la quantité d'énergie emmagasinée dans le condensateur.

Un condensateur de $1mF$ et plus est une valeur normale.
Une fois notre tension assez propre (même si elle présente des oscillations), nous allons utiliser les régulateurs de tensions.

Les Régulateurs de tension

Principe et branchements

Ces composants à trois broches servent à stabiliser une tension pseudo-continue en une tension continue et stable, c'est à dire ne présentant pas d'oscillations.

Le branchement est le suivant :

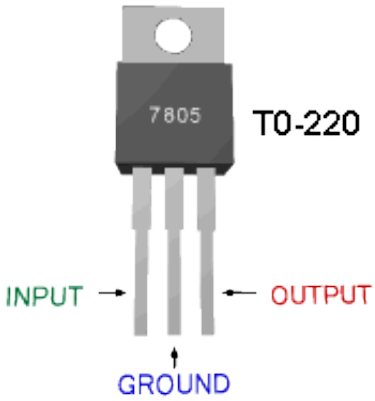


FIGURE 22.8 – Un régulateur de tension

- L'entrée INPUT est la tension à réguler
- La broche GROUND est la masse du circuit
- la sortie OUTPUT est la tension regulée (stable)

Les familles des régulateurs

Il existe différentes familles de régulateurs linaires, les plus connus étant les LM78XX [3] ou les LM317 qui sont des régulateurs de tension réglables.
Pour des tensions négatives, se référer à la famille des LM79XX.

Pour que ces composants fonctionnent correctement, il convient de leur mettre en entrée une tension supérieure à leur tension de régulation.

Concrètement, si on utilise un LM7812 (régulateur 12 V), il faudra mettre en entrée au moins 15 V pour que le composant fonctionne dans la plage idéale.

Cette tension de $15V$ justifie la valeur de tension du transformateur.
Ces composants sont linéaires, c'est à dire que la différence de tension entre l'entrée et la sortie $(V_e - V_s)$ engendre une perte de puissance dans le système.
Cette perte dépend directement du courant absorbé par le circuit en aval mais également de la différence de tension $(V_e - V_s)$.
Cette puissance vaut :

$$P_{perdue} = (V_e - V_s) \cdot I_{charge} \quad (W)$$

Un exemple

On souhaite réguler un circuit en 12V, ce dernier consommant 150 mA (I_{charge}).

3. XX prend la tension régulée de sortie (5,9,12,15,18V...)

La perte d'énergie vaut

$$P_{perdue} = (V_e - V_s) \cdot I_{charge} = (15 - 12) \cdot 0.150 = 0.45W$$

Quelques fournisseurs de composants et matériels

Voici ma liste des fournisseurs.

- **Reichelt**
  Cet site possède un choix très élevé de circuits intégrés (Amplificateurs opérationnels, transistors...) mais également de diodes, de leds, résistances.
  La documentation est bien fournie.
  Vente d'outils pour l'électronique.
  Le site est disponible à l'adresse https://www.reichelt.com/

- **Gotronic**
  Cet site possède un choix très élevé de capteurs.
  La documentation est bien fournie.
  Vente d'outils pour l'électronique.
  Délais de livraison rapides.
  Le site est disponible à l'adresse https://www.gotronic.fr/

- **Conrad**
  Similaire à Gotronic.
  La documentation est bien fournie.
  Vente d'outils pour l'électronique.
  Le site est disponible à l'adresse https://www.conrad.fr/

- **Semageek**
  Similaire à Gotronic.
  Vente d'outils pour l'électronique.
  Le site est disponible à l'adresse https://boutique.semageek.com/fr/

- **Dfrobot**
  Un large choix de capteur pour l'embarqué.
  La documentation est bien fournie, cependant certains prix sont parfois un peu excessifs.
  Le site est disponible à l'adresse https://www.dfrobot.com/

- Puis **Banggoods** pour les petites bricoles pas chères...
  Le site est disponible à l'adresse https://www.banggood.com/
  Délai de livraison au maximum de 3 semaines.

Septième partie

L'environnement Arduino

Présentation de l'environnement Arduino et de son langage

SECTION 23

INTRODUCTION

Ce document vise à présenter le projet Arduino et ses supports
Ce tutoriel a pour but également de présenter certaines possibilités d'Arduino en terme de langage et de ressources.
Bien évidemment, cette section n'est pas du tout exaustive.

Origines

Arduino est née en 2004 sous l'impulsion d'étudiants italiens souhaitant promouvoir l'accès à l'électronique. Ils se rencontraient fréquement dans un bar pour développer leur projet.
Aujourd'hui, Arduino c'est :

1. Un langage de programmation basé sur le C++
2. Une communauté
3. Un projet Open-Source

Supports

Arduino disposant d'une communauté assez vaste, de nombreux supports existent.
Nous avons notamment le site officiel d'Arduino à l'adresse suivante :
arduino.cc/Reference/en

Le langage Arduino est compatible avec les instructions du C++ dans la mesure ou le compilateur pour Arduino est g++. Ainsi, les types compososés comme les structures et classes sont supportés, tout comme le mot clé auto par exemple.

SECTION 24

PRÉSENTATION

Nous utilisons des cartes Arduino Uno, basées sur les microcontroleurs Atmega-328 du fabricant ATMEL.
Les microcontrôleurs sont des unités contenant dans un seul boitier une mémoire, un processeur et des interfaces entrées-sorties pour ne citer que ces élements.
Cela permet notamment de dialoguer avec des périphériques [1]

Le microcontroleur

Alimentation

Tension d'alimentation

Le microcontroleur doit être alimenté entre 1.8 V et 5.5 V.
Il existe deux façon d'alimenter la carte Arduino :

1. Via le port USB Le port USB délivre du 5V régulé avec un courant maximal de 500 mA (cas général)
2. Via la broche Vin (connectique Jack femelle) La carte Arduino possède un régulateur intégré de tension en 5 V, ce qui permet d'alimenter la carte entre 7V et 20V

Courants d'entrées-sortie

Fréquence d'horloge

La carte Arduino comporte un oscillateur de 16 MHz même si en interne du microcontroleur, un oscillateur de 8 Mhz est intégré. Cela donne une idée des performances maximales de l'Arduino.

Mémoire

1. Voire section Protocoles de communication

Ce microcontroleur dispose de clé **32 ko** de clé **mémoire flash**, c'est à dire la mémoire pour stocker le programme téléversé vers la carte.

Quand à la clé **mémoire vive (SRAM)**, elle est de **2 ko** et est utilisée pour les variables du programme en cours d'éxécution.
Cette mémoire peut être donc vite saturée lors de l'utilisation de grands tableaux par exemple.

Enfin, le possède une mémoire effacable electriquement, appelée **EEPROM** [2], lors de l'éxécution du programme.
Cette mémoire occupe **1 ko** et chaque registre de cette mémoire, pouvant stocker un nombre codé sur 8 bits (type byte ou char), peut être modifiée 100 000 fois avant son arrêt définitif.

Caractéristiques électriques

Le microcontrolleur dispose d'entrée sorties permettant d'intéragir avec des périphériques (Diodes electroluminescentes, capteurs, modules de communication. . .)
Les entrées sont deux types :

1. entrée **numérique** : la valeur lue sera perçue comme un niveau logique 0 ou 1 sur les broches allant de 1 à 13
2. entrée **analogique** : Un Convertisseur Analogique-Numérique 10 bits est intégrés sur les broches A0, A1, A2, A3, A4 et A5 De ce fait, le CAN est possède une résolution de 4.84 mV avec une référence de tension à 5V [3]

Astuce : Il est possible de configurer les broches analogique en broches digitales.

2. Référence : arduino.cc/Reference/EEPROM
3. Il est également possible de changer la tension de référence de la carte Arduino arduino.cc/Reference/en/language/function/analog-io/analogreference

SECTION 25

LE LANGAGE

Les types

Par défaut les types sont signés, c'est à dire que la plage de valeur pour un nombre codé sur n bits est compris entre $\frac{-2^n}{2}$ et $\frac{2^n+1}{2}$ Pour définir un type non signé, c'est à dire pour agrandir la plage positive, il suffit d'ajouter le mot clé **unsigned** avant les types concernés.

Les types supportés

1. **byte** *[1 octet]*
   Désigne la plus petite unité de mémoire allouable, permettant de stocker un nombre entier compris entre -127 et + 127.

2. **unsigned byte** *[1 octet]*
   Idem mais la plage de valeur strictement positive

3. **int** *[2 octets]*
   Permet de stocker un nombre entier compris entre -32536 et +32536

4. **unsigned int** *[2 octet]*
   Idem mais la plage de valeur strictement positive

5. **float** *[4 octets]*
   Permet de manipuler des réels

6. **double** *[4 octets]*
   Idem, mais ce type est plus précis que le type float et demande plus de ressources au micro-controleur.

7. **char** *[1 octets]*
   Ce type est utilisé pour stcoker et traiter des caractères de la table ASCII.

8. **String** *[4 octets]*
   String est un type élaboré qui permet de traiter des chaines de caractères.

Les types non supportés

Les types **vector**, **array** et **tuple** ne sont pas supportés par le langage Arduino.

Les fonctions

Les fonctions mathématiques

En ce qui concerne les fonctions mathématiques, les fonctions trogonométriques sont incluses.

SECTION 26

LES BROCHES D'INTERRUPTION

Dans certains cas, il est souhaitable de récuperer la valeur d'une broche à tout moment du programme, même quand celui ci est occupé dans une tâche et même dans une fonction de temporisation [1].
Pour remédier à ce problème, on peut utiliser les **broches d'interruption** qui permettent de récuperer la main sur l'ensemble du programme lorsqe'un évènement survient sur une broche.

Concrètement, lorsque un évènement e survient sur la broche b, la fonction f est appelée, quelque soit l'état du programme principal.

Prenons le cas d'un bouton qui doit changer l'état d'une LED à n'importe quel moment du programme.

```
int ledPin = 13;    //Led interne
int BOUTON = 2;  //Bouton relié à la broche 2 avec une résistance de charge

volatile int state = LOW;  //Etat courant de la LED

void setup() {

  Serial.begin(9600);//Vitesse de communication à 9600 bit/s

  pinMode(ledPin, OUTPUT);                //Led mise en sortie
  pinMode(BOUTON, INPUT_PULLUP);    //Bouton mis en entrée

  attachInterrupt(digitalPinToInterrupt(BOUTON), onEvent, CHANGE);  //Appel de
 la fonction onEvent à chaque changement de front du bouton
  Serial.println("Init");
}

void loop() {

 delay(5000); //Pause du programme principal
```

1. Voir delay(), delayMicroseconds()

```
}

void onEvent() {

  state = !state; //Inverse l'état de la LED

  if(state){
  Serial.println("ON");
  }else{
  Serial.println("OFF");
  }
   digitalWrite(ledPin, state); //Met à jour l'état de la LED
}
```

Code d'exemple

Ici, quelque soit l'action effectuée dans la fonction loop, dès qu'un front montant est détecté sur la broche BOUTON (2), la fonction onEvent() sera exécutée et changera l'état de la LED à chaque front

Mode d'interruption

Il existe différents modes pour les broches :

- RISING : front montant
- FALLING : Front descendant
- CHANGE : Front montant et descendant

Chronogrammes d'interruption

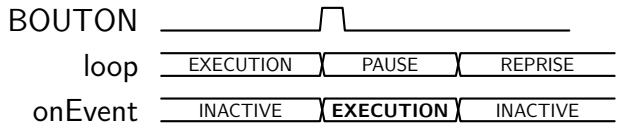


FIGURE 26.1 – Exemple avec mode RISING

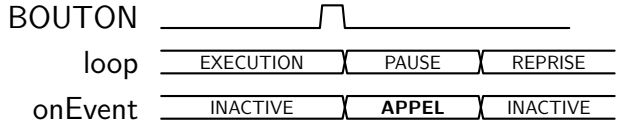


FIGURE 26.2 – Exemple avec mode FALLING

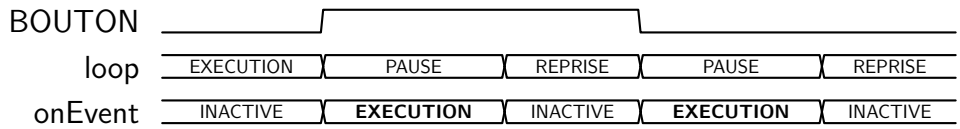


FIGURE 26.3 – Exemple avec mode CHANGE

SECTION 27

SYNTHÈSE ARDUINO

Caractéristiques

Matériel

1. Mémoire Flash : 32 ko
2. Mémoire Vive (SRAM) : 2 ko
3. Fréquence d'horloge : 16 MHz

Électriques

1. Impédance d'entrée : $> 1MOhm$
2. Courant de sortie par broche : 40 mA maximum
3. Courant de sortie pour toutes les broches entrée-sorties : 200 mA

Huitième partie

Les servo-moteurs

Théorie sur les servo-moteur et applications pratiques avec Arduino

SECTION 28

PRÉSENTATION

Les servo-moteurs sont utilisés lorsqu'on souhaite un asservissement en position d'un axe de rotation [1]

Asservissement

Un asservissement est un processus de correction pour maintenir une consigne.
Par exemple, un régulateur de vitesse dans une voiture est un système asservi car la vitesse doit être constante quelle que soit la pente.

Architecture

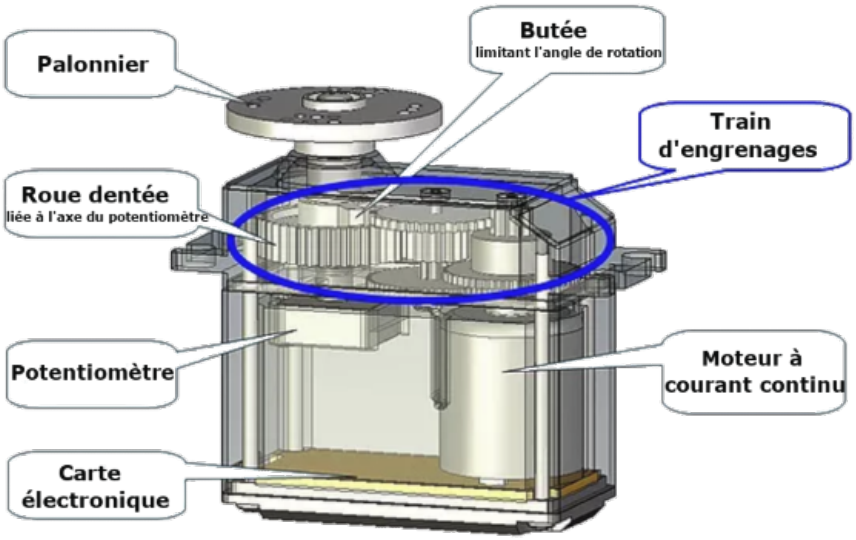


FIGURE 28.1 – Constitution d'un servo-moteur

Domaines d'application

1. Pulse Width Modulation : Modulation par Largeur d'Impulsion

- Modélisme
- Robotique

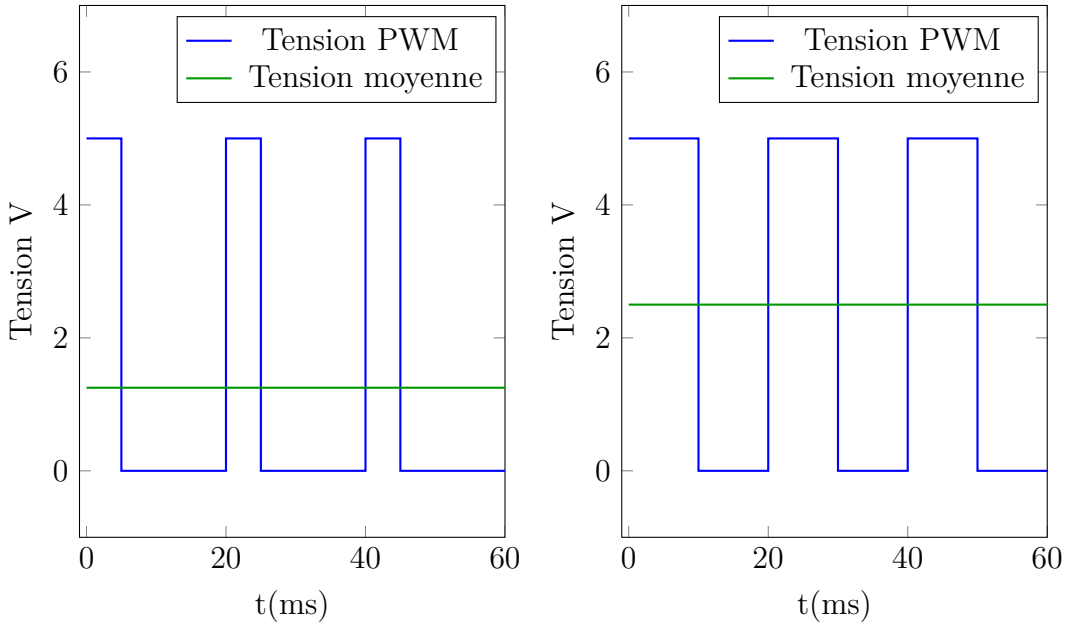
Commande des servo-moteurs

Les servo-moteurs ont besoins d'être controlés via un signal PWM.

Principe de la PWM

La PWM est la création d'un signal numérique dont le temps à l'état haut est variable. On fait varier le rapport cyclique (appelé r) qui est compris entre 0 et 1.

$$r = \frac{T_{on}}{T_{signal}}$$



(a) r=0.25 (b) r=0.5

FIGURE 28.2 – Différents rapports cycliques

Applications de la PWM

En faisant varier la tension de sortie dans le temps rapidement (¿50Hz), on peut simuler une tension analogique. Quelques applications :

- Controle de la luminosté d'une LED
- Controle de servo-moteurs

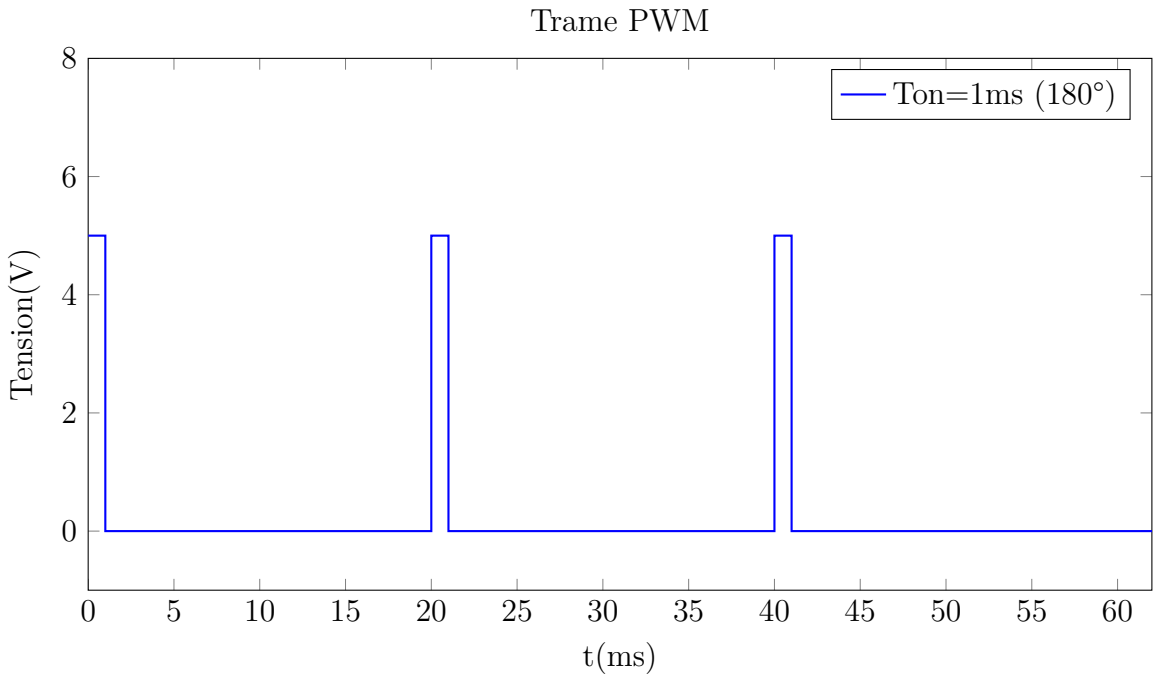
Code Arduino

Voici un code d'exemple pour faire varier la luminosité d'une LED.

```
const int pin_led = 11; //Selection d'une broche PWM

float duty_cyle[11] = {0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0};//
 Création d'un tableau avec les différents rapports cycliques

void setup() {

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

}//Fin setup

void loop() {

    for(int i=0;i<11;i++)
    {
        int value_r = duty_cyle[i]*255.0; //Conversion d'une valeur entre 0 et
 1 en une valeur entre 0 et 255
        analogWrite(pin_led, value_r); //Change le rapport cyclique pendant 3 s
        delay(600);        //Attend 0.6s
    }


}//Fin loop
```

Variation de la luminosité d'une LED

Trame de commande servomoteurs



Branchement d'un servo-moteur

- Câble noir ou marron : GND
- Câble rouge : +5V
- Câble blanc ou jaune : Signal Arduino (11)

Code Arduino

```
#include <Servo.h>      //Inclusion de la bibliothèque Servo
Servo myservo;  // Création d'un objet Servo
int pos = 0;    //Angle du servomoteur

void setup() {

  myservo.attach(11);  //Choix de la broche du servo moteur

}

void loop() {

  for (pos = 0; pos <= 180; pos += 1) { //Parcours la plage angulaire [0-180]
 degré par degré
```

```
    myservo.write(pos);              //Actualise la position
    delay(15);                       //Attend 15 ms avant l'actualisation

   }//Fin for

   for (pos = 180; pos >= 0; pos -= 1) {     //Parcours la plage angulaire
  [0-180] degré par degré

    myservo.write(pos);              //Actualise la position
    delay(15);                       //Attend 15 ms avant l'actualisation

   }//Fin for
}//Fin loop
```

Variation de la position d'un servo-moteurs

Caractéristiques

Electriques

- Tension de commande et d'alimentation : $\tilde{5}$V

Mécaniques

- Couple de sortie (Nm)
- Vitesse de rotation (degré/temps)

Neuvième partie

ESP8266

Configuration d'un ESP8266 pour une utilisation via REPL

SECTION 29

INTRODUCTION

Présentation

Cette partie a pour but de configurer un ESP8266-12E (NodeMCU) afin que ce dernier puisse être accessible en tant que réseau Wifi.

Ce tutoriel s'adresse également dans le cas où vous avez perdu vos mots de passe d'accès (réseau wifi ou WebRepl) ou bien que vous souhaitez partir sur des bases saines.

Information

Le temps estimé pour configurer l'ESP8266 est de 25 min

Conventions

Les commandes à saisir sont dans des encadrés similaires :

```
sudo apt-get update
```

Exemple de commande

Dans un souci de clarté :

- Les fichiers sont indiqués par le repère FILE nom du fichier
- Les dossiers sont indiqués par le repère DIR nom du dossier
- Les touches du clavier et le texte à saisir au clavier sont indiqués par le repère KEY Raccourci clavier ou texte à saisir
- Les bibliothèques, logiciels et utilitaires sont indiqués par le repère LIB nom de l'utilitaire

SECTION 30

PRÉ-REQUIS

Matériel

Pour réaliser ce tutoriel, vous aurez besoin de

- Un ordinateur (Linux, Apple ou Windows)
- Un ESP8266 (NodeMCU)
- Un câble USB Micro Type-B

FIGURE 30.1 – Un câble USB Micro type-B

Mise à jour des systèmes UNIX

Avant toute chose, il convient de mettre à jour la liste des paquets et de mettre à jour les logiciels déjà présents sur votre ordinateur si ce dernier est sous LINUX (UNIX).
Les commandes suivantes sont à saisir dans un terminal.

Mise à jour de la liste des paquets

```
sudo apt-get update
```

Mise à jour de la liste des paquets

Mise à jour des logiciels

```
sudo apt-get -y upgrade
```

Mise à jour des logiciels

Le -y sert à accepter automatiquement la mise à jour.

Mise à jour de Python

Il conviendra d'installer au minimum la version 3.6 de Python.
Pour vérifier votre version, ouvrez un terminal et saisissez la commande

```
python3
```

Vérification de la version de python

Si l'invité de commande Python suivant apparaît, la version est présente.
Pour quitter l'interpréteur python, il suffit de saisir KEY exit() dans l'interpréteur ou bien de faire KEY Ctrl +z

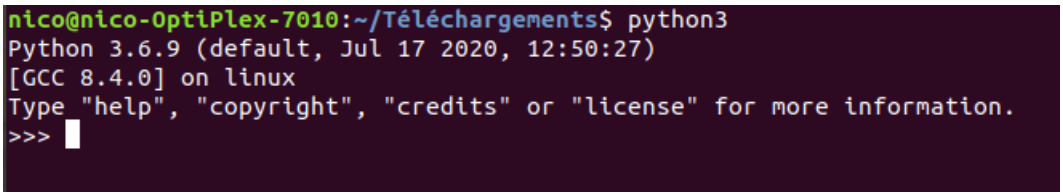


FIGURE 30.2 – Invité de commande Python

Le cas échéant, je vous invite à saisir la commande suivante :

```
sudo apt-get -y install python3.7
```

Installation de Python 3.7

Effacer la mémoire de l'ESP8266

Dans un premier temps, nous allons effacer le contenu de la puce ESP8266. Ceci nous permettra de partir sur des bases saines.

Point-clé

Maintenant, vous pouvez brancher votre ESP8266 sur un des ports USB de votre ordinateur.

Il convient ensuite d'installer les outils adéquats.

Installation de Pip3

LIB pip3 est un utilitaire Python qui va nous permettre d'installer le petit programme pour effacer l'ESP8266.
On l'installe de la manière suivante :

```
sudo apt-get -y install python3-pip
```

Installation de Pip3

Installation de Esptool

L'utilitaire qui va se charger d'exécuter cette opération s'appelle LIB esptool.
Pour l'installer, on effectue

```
pip3 install esptool
```

Installation de Esptool

Voici le résultat de la commande sur le terminal :

FIGURE 30.3 – Résultat de l'installation de pip3

Récupération du port USB

L'ESP8266 étant raccordé, l'ordinateur lui a affecté un nom de port de type */dev/ttyUSBx* avec x représentant le numéro du périphérique USB.

Pour récupérer la valeur de ce numéro, nous allons lancer la commande suivante :

```
ls /dev/ttyUSB*
```

Récupération du numéro du port série

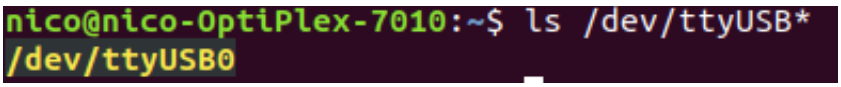


FIGURE 30.4 – Résultat de la commande

Dans le cas présent,le nom du port est */dev/ttyUSB0*

Effacer la mémoire

Lancer la commande suivante :

```
esptool.py --port /dev/ttyUSB0 erase_flash
```

Effacer la mémoire de l'ESP8266

Évidemment, si vous avez un autre numéro de port avec la commande **esptool.py flash_id** , vous mettez votre numéro.

Voici le résultat de la commande sur le terminal :

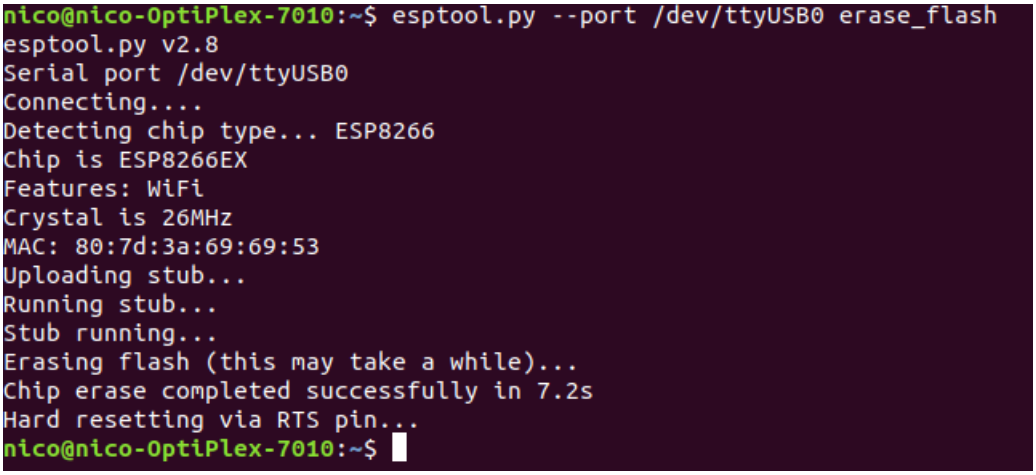


FIGURE 30.5 – Résultat de la commande pour effacer l'ESP8266

SECTION 31

INSTALLER LE FIRMWARE SUR L'ESP8266

Maintenant que l'ESP8266 est vide, il nous reste à installer son logiciel (firmware) fin qu'il puisse utiliser le Wifi selon deux modes :

- Point d'accès : l'ESP8266 créer son propre réseau Wifi
- Connexion : l'ESP8266 peut se connecter à un réseau Wifi pour dialoguer

Récupération du logiciel

Le logiciel se présente sous fichier binaire (.bin) et est disponible à l'adresse suivante :

http://micropython.org/download/esp8266/

Je vous invite à télécharger la dernière version stable (latest)

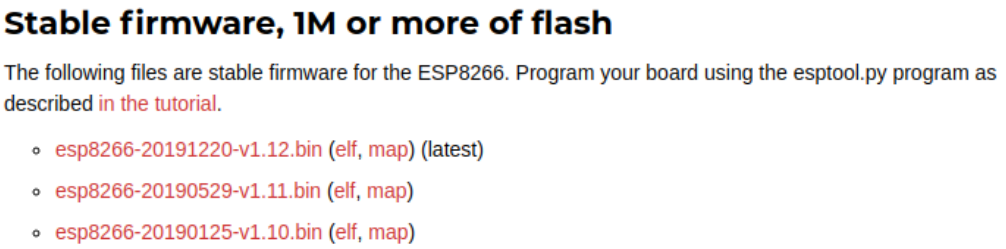


FIGURE 31.1 – Récupération du logiciel pour l'ESP8266

Installation du logiciel

Tout d'abord, placez vous dans le même répertoire que votre fichier binaire installé précédemment et ouvrez un terminal.

La commande KEY ls devrait confirmer votre contenu du répertoire.

```
ls
```

Vérification du répertoire

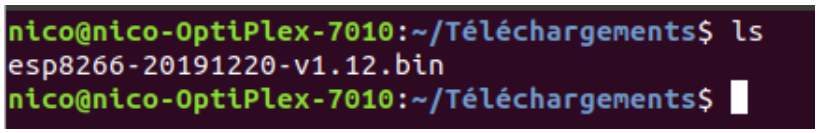


FIGURE 31.2 – Contenu du répertoire

Il ne vous reste plus qu'a saisir la commande pour installer le firmware.

```
esptool.py --port /dev/ttyUSB0 --baud 460800 write_flash --flash_size=detect 0
   esp8266-20190125-v1.10.bin
```

Installation du firmware

Comme précédemment, si vous avez un nom de fichier différent, je vous laisse le soin de changer de nom afin de coïncider avec le vôtre.

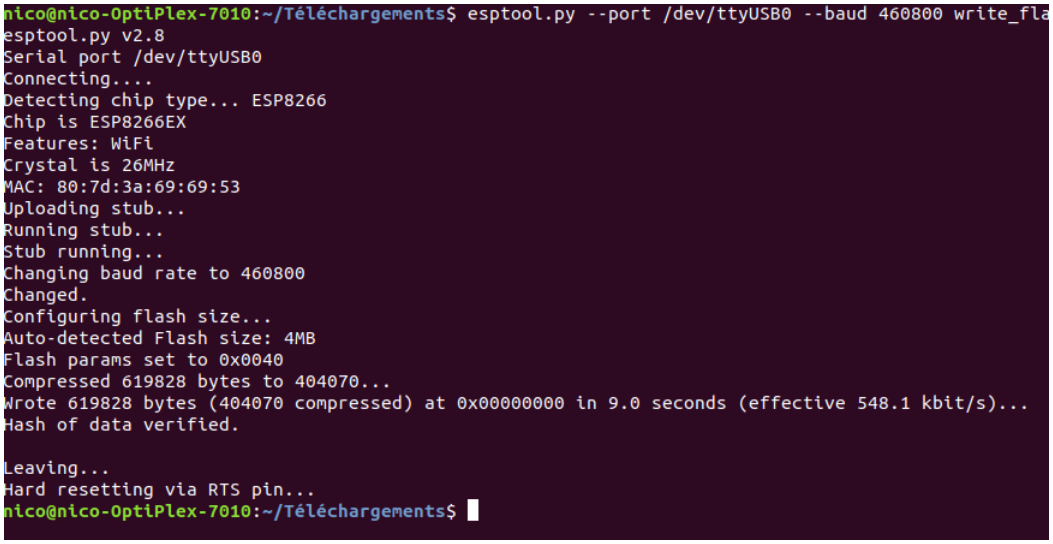


FIGURE 31.3 – Résultat de la commande pour installer le firmware

Après le déploiement du firmware, le module redémarre et il est configuré en point d'accès WiFi avec pour nom **MicroPython-6953** .
Les chiffres correspondent aux quatre derniers chiffres de l'adresse MAC du module.

SECTION 32

CONFIGURER LE MOT DE PASSE WEBREPL

Installation de WebRepl

Le logiciel LIB WebRepl va nous permettre de se connecter à l'ESP8266 afin de saisir des commandes Python.

Le logiciel est disponible à l'adresse suivante :

https://github.com/micropython/webrepl

Cliquez ensuite sur KEY Code puis KEY Download Zip

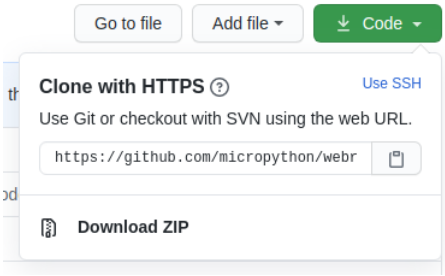


FIGURE 32.1 – Téléchargement de WebRepl sur Github

Veuillez commencer par extraire l'archive. Celle-ci contient les fichiers suivants :

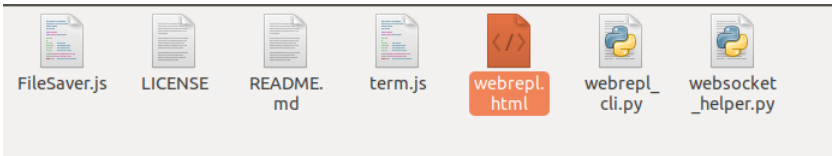


FIGURE 32.2 – Contenu du dossier WebRepl

Installation de screen

LIB screen est un utilitaire qui va nous permettre de se connecter à l'ESP8266 via le câble USB car actuellement, il nous est impossible d'utiliser WebRepl.
On installe l'utilitaire avec la commande suivante :

```
sudo apt-get install -y screen
```

Installation de screen

Création du mot de passe

Utilisation de screen

On peut accéder à l'interpréteur de commande Python REPL [1] via le port série en tapant la commande suivante dans un terminal :

```
screen /dev/ttyUSB0 115200
```

Exécution de screen

Point-clé

Il faut appuyer sur la touche Entrée pour afficher l'invité de commande MicroPython.

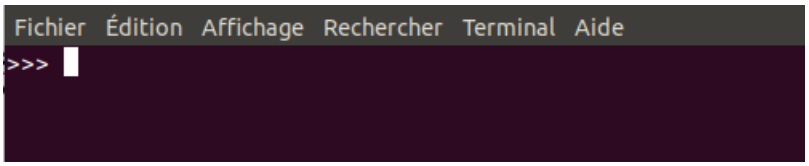


FIGURE 32.3 – Console screen

Pour quitter Screen, il faut appuyer sur les touches KEY CTRL + a puis écrire KEY :quit

Ensuite, entrez la commande suivante via le terminal Screen :

```
>>> import webrepl_setup
```

Commande pour créer un mot de passe

Le système vous demande tout d'abord d'activer l'accès par le réseau Wifi dès le démarrage en saisissant KEY E.

1. Read Evaluate Print Loop

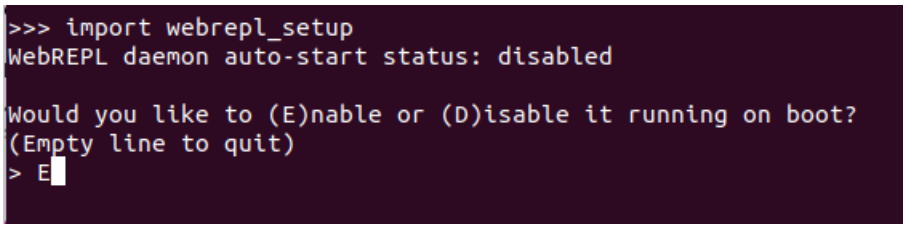


FIGURE 32.4 – Activation de l'ESP8266 screen

Il vous invite ensuite à saisir le mot de passe pour l'accès à WebRepl. Ici le mot de passe choisi est KEY crepp

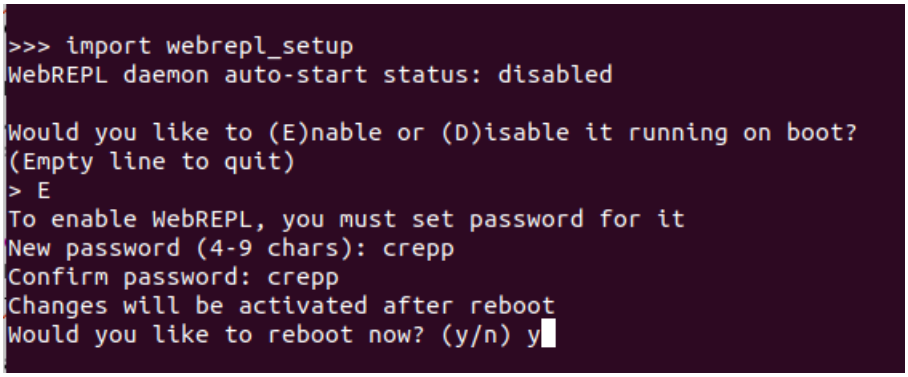


FIGURE 32.5 – Activation de l'ESP8266 screen

Enfin, saisissez KEY y pour redémarrer l'ESP8266.
A ce moment là, les lignes suivantes apparaissent :

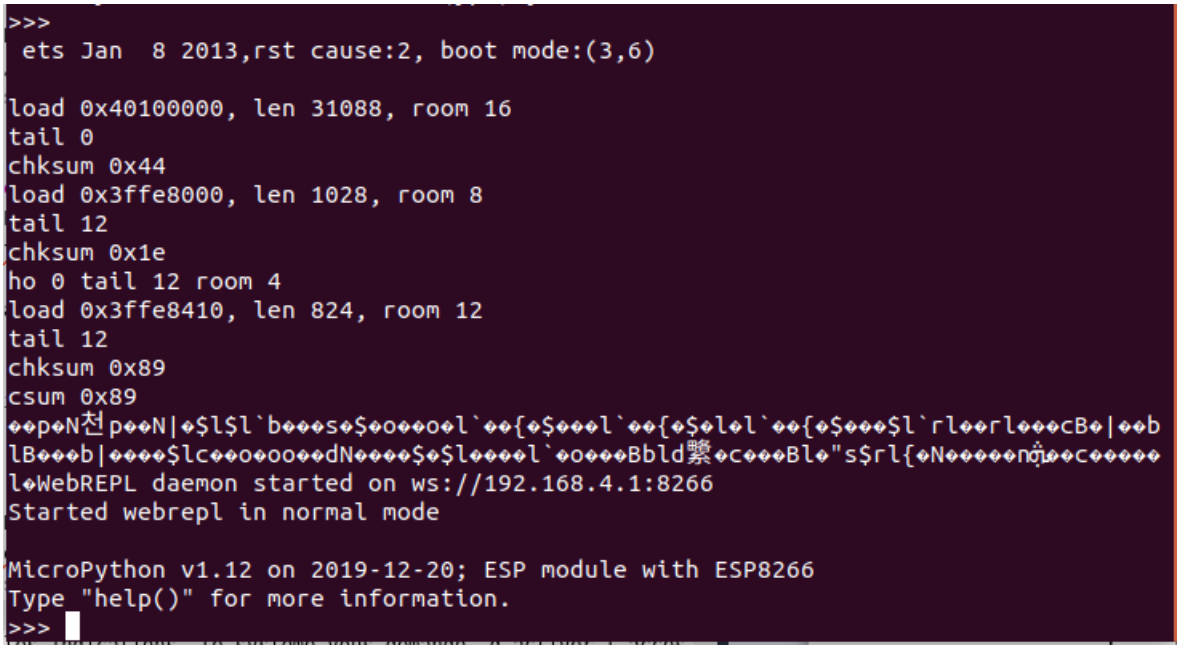


FIGURE 32.6 – Fin de la configuration

La configuration de l'ESP8266 est terminée

Connexion à WebRepl

Connexion au réseau de l'ESP8266

Tout d'abord, veuillez chercher parmi les réseaux Wifi le réseau de l'ESP8266 appelé MicroPython-XXXX avec XXX représentant les 4 derniers chiffres de l'adresse MAC de l'ESP8266.

Lors de la connexion, un mot de passe est demandé, saisir KEY micropythoN

Lancement de WebRepl

Veuillez lancer, à l'aide de votre navigateur Internet, le fichier FILE webrepl.html situé dans le dossier WebRepl précédemment installé.
Pour cela, veuillez faire un KEY Click droit + "Ouvrir avec le navigateur Web ...".
Vous tombez sur l'interface Web. Il suffit de cliquer sur KEY Connect et d'entrer le mot de passe que vous avez définie. (en l'occurrence **crepp**).

SECTION 33

CONNEXION À UN RÉSEAU WIFI

L'inconvénient avec la méthode présente est de jongler entre les 2 accès WiFi (Wifi classique ou réseau ESP8266).
Or, il est possible de configurer le module pour qu'il se connecte sur votre box WiFi en tant que client afin d'éviter les désagréments des connexions WiFi.
Pour cela il suffit de se connecter à l'ESP8266 via le port série et de taper les commandes suivantes :

```
import network
wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect('ssid', 'password')
```

Commandes de connexion

Vous pouvez vérifier la nouvelle adresse IP fournie par votre box en tapant la commande :

```
>>> wlan.ifconfig()
```

Vérification

Par contre il est nécessaire de passer les commandes suivantes après la connexion à votre box :

```
>>> import webrepl
>>> webrepl.start()
```

Initialisation

Dixième partie

Projet Cocci-Bot

Mise en place du Bluetooth dans le projet Cocci-Bot

SECTION 34
INTRODUCTION

Présentation

Ce tutoriel a pour objectif de créer une application pour diriger le robot en Bluetooth. L'application enverra des « commandes » au module Bluetooth de type Crius Cette application se fera à l'aide du logiciel en ligne «App Inventor 2», qui se divise en deux parties :

- une partie dédiée exclusivement à l'interface graphique (positionnement des boutons, etc).
- une seconde partie dédiée à la gestion des données et à leurs envois/ réceptions

Nous verrons donc comment faire une interface spécifique pour le Cocci-Bot.

Remarque importante

Il est impératif que le téléphone portable tactile soit sous le système d'exploitation Android et qu'il dispose de la fonctionnalité Bluetooth

Par exemple, si j'appuie sur un bouton «avancer», je veux que l'application envoie en Bluetooth la donnée qui permettra à l'Arduino de comprendre l'instruction à l'aide du module Bluetooth.

Liste du matériel

Pour cette première partie, nous aurons besoin de :

- Un téléphone portable, comme dit précédemment, étant tactile et fonctionnant sous Android. Les dimensions importent peu, pourvu que celui-ci dispose du mode Bluetooth
- Un module Bluetooth de type Crius (Alimentation en 5V)
- Une carte Arduino
- Des fils de connexion pour brancher le module à la carte
- Une connexion internet

Et c'est tout... pour le moment.

Cahier des Charges

Maintenant que nous avons la liste du matériel nécessaire, ce serait bien de mettre en place un cahier des charges afin de savoir ce que le robot sera capable de faire...

1) L'application devra se connecter au module Bluetooth du robot. Pour cela, on se connectera à un module en passant par la liste des modules Bluetooth disponibles.
Cette méthode permet de se connecter sans avoir l'adresse MAC. En revanche, l'étape de l'appairage est indispensable. (Chapitre 35)

2) Le robot devra être contrôlé de façon manuelle, il y aura donc 5 boutons de commande :

- 1 bouton «Avancer»
- 1 bouton «Reculer»
- 1 bouton «Droite»
- 1 bouton «Gauche»
- 1 bouton «Stop»

3) Le robot devra également se diriger de façon autonome grâce à ses capteurs (distance, lumière). Il y aura donc un bouton «automatique» pour déclencher ce mode.

Application facultative

Le robot pourra également indiquer l'état de la batterie (en %, sur l'écran LCD) par simple appui d'un bouton.
Cette section ne sera pas abordée içi.

SECTION 35

APPAIRAGE DU MODULE

Point-clé

Avant de créer l'application, il faut tout d'abord que le portable puisse reconnaître le module Bluetooth.
Pour cela, il faut «l'appairer», c'est à dire que l'on va définir que le module sera apte à recevoir les données envoyées par le portable.
Cette étape est très rapide et ne sera effectuée qu'à chaque changement de module Bluetooth.

1) Tout d'abord, branchez le module : la broche +5V du module est reliée à la broche 5V de la carte Arduino et la broche GND du module est reliée à la GND de la carte.

2) Démarrez le Bluetooth sur votre téléphone portable puis allez dans **paramètres ¿ Bluetooth**
Faites « **rechercher** » afin de trouver un périphérique. Vous devriez trouver un périphérique « BT Crius ». Cliquez dessus et faites « **associer** ».
A ce moment là, on vous demandera un mot de passe, qui par défaut, est **0000** (ou **1234**).

Lorsqu'il est entré, faites « **valider** » et au bout de quelques secondes, le clignotement du module devrait se stopper pour qu'il n'y ait qu'une lumière continue

Et voila, le module est appairé.

SECTION 36

CRÉATION DE L'INTERFACE

Préparation

Maintenant que nous nous sommes assurés que la communication s'effectuera, il est temps de faire l'application et de préparer notre outil (enfin, me direz-vous !)

Tout d'abord, saisissez dans un moteur de recherche l'adresse suivante : `http://appinventor.mit.edu/` (Site **App Inventor 2** ≫)

Une page comme ceci devrait apparaître :

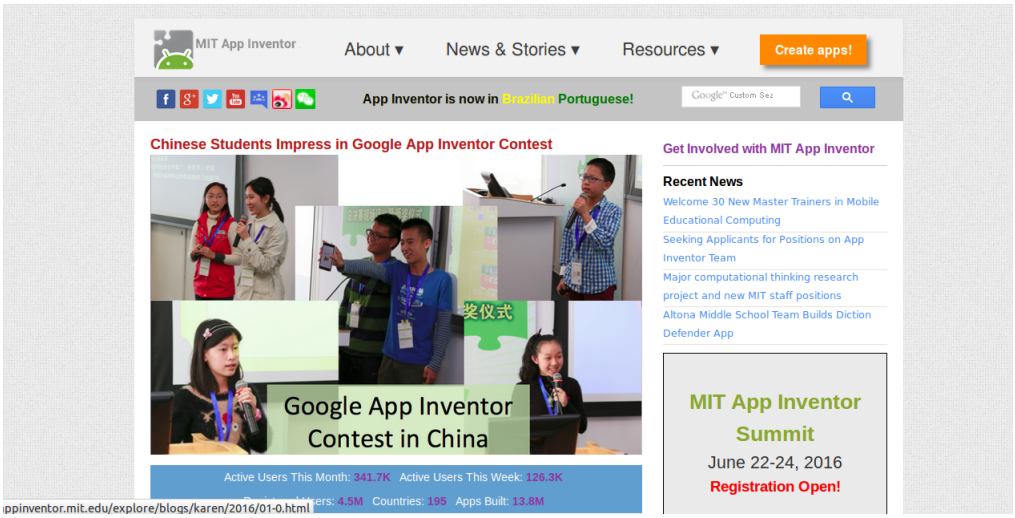


FIGURE 36.1 – Le portail App Inventor

Ensuite, cliquez en haut à gauche sur :

Create App

Cela devrait vous diriger vers ceci :

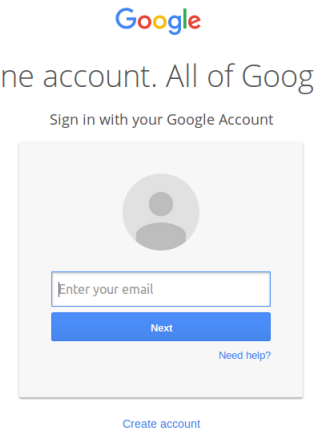


FIGURE 36.2 – La fenêtre de connexion

Puis connectez vous avec votre compte Google (ou Gmail). Si vous n'en avez pas, sa création est rapide... (create account).
Vous tombez ensuite sur une interface similaire :

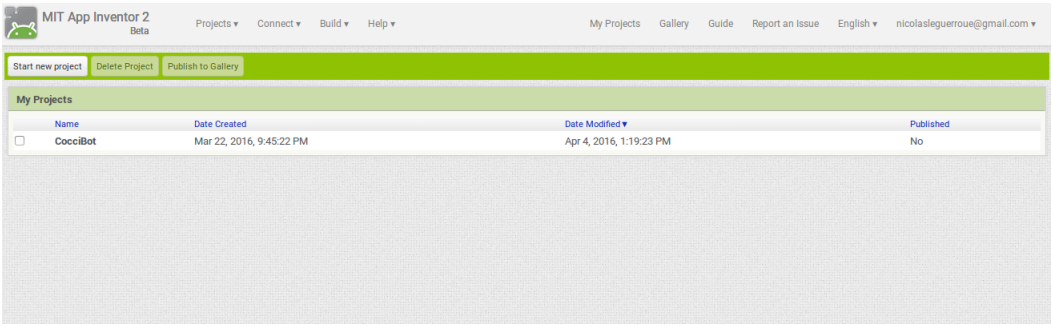


FIGURE 36.3 – La fenêtre des applications

Pour démarrer votre projet, cliquez sur « **Start new project** » (onglet « **Projects** »)et là, cette page apparaît :

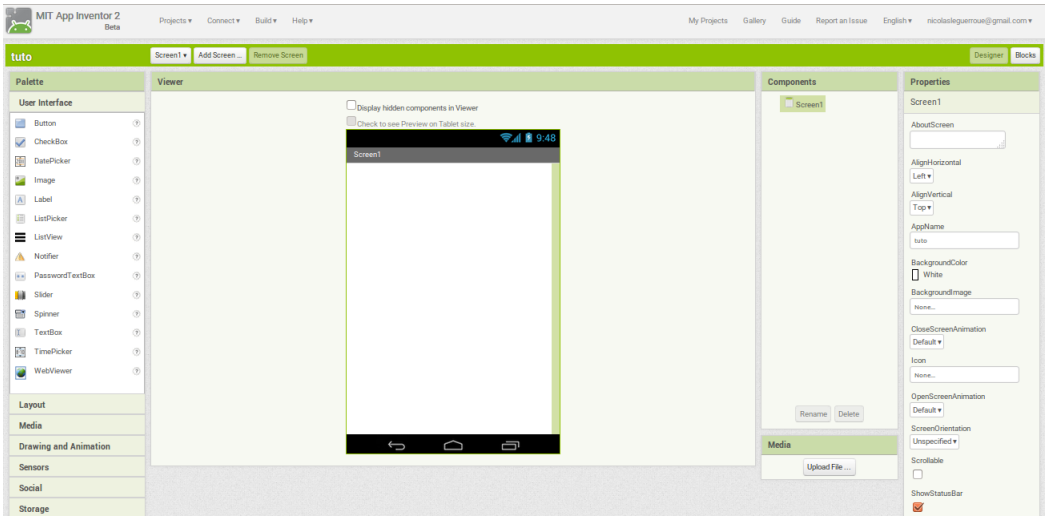


FIGURE 36.4 – La fenêtre de notre application

Avant toute chose, réglez la langue du site en français. Pour cela, sélectionnez l'onglet **English** et choisissez « **français** ».

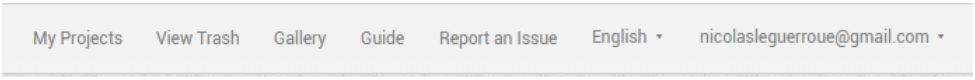


FIGURE 36.5 – Le choix de la langue

Sauvegarde du projet

Avant toute chose, voici une étape non négligeable : la sauvegarde du projet

Remarque importante

L'application App Inventor étant en ligne, un problème de réseau peut ruiner votre projet en cas de mauvaise sauvegarde. Il convient de sauvegarder **régulièrement** votre travail

Il faut se reporter au menu du haut et aller dans **Projets** puis **Enregistrer le projet**

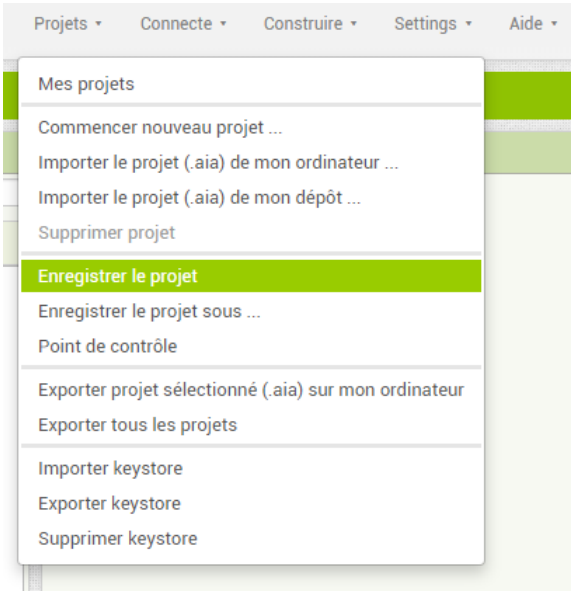


FIGURE 36.6 – Sauvegarde du projet

Quant à l'interface, un petit tour d'horizon s'impose.

Présentation des éléments graphiques

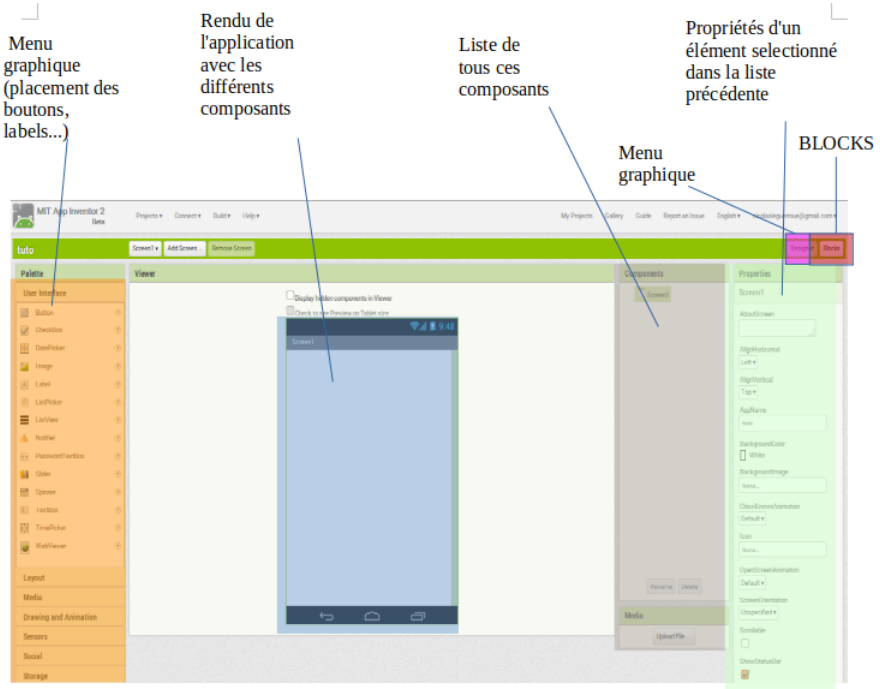


FIGURE 36.7 – Les outils de App Inventor

Par convention, dans ce tutoriel :

- la partie orange sera la partie menu
- la partie bleue sera l'interface
- la partie grise sera la partie composants
- la partie verte sera la partie propriétés
- la partie rouge garde son nom : blocks
- la partie magenta est la partie "graphique" (regroupement des quatre premières parties

Maintenant, prenons un sélectionneur de liste dans le menu, et faisons le glisser sur l'interface. Pour cela, maintenez enfoncé le« **sélectionneur de liste** » dans le menu et faites le glisser sur l'écran virtuel. Relâchez ensuite la souris. On obtient :

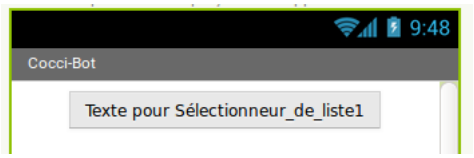


FIGURE 36.8 – Rendu de l'interface

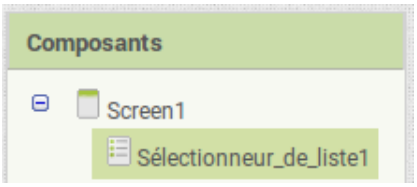


FIGURE 36.9 – Rendu des composants

Veuillez cliquer sur **Sélecteur_de_list1**

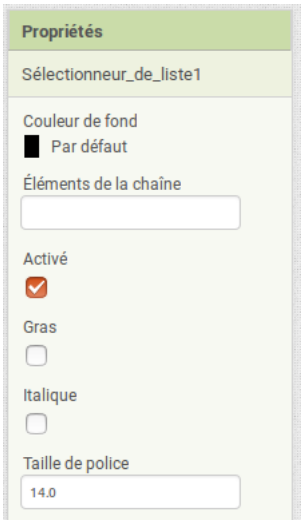


FIGURE 36.10 – Rendu des propriétés

Principaux élements

Maintenant, on peut voir que le menu offre plusieurs choix d'objets. Voici les plus importants :

Il vous est notamment possible de faire les actions suivantes :

- Créer un bouton
- Créer une case à cocher
- Sélectionner une date
- Afficher une image
- Afficher du texte
- Créer une liste de selection
- Créer une liste classique
- Envoyer une notification
- Afficher un curseur
- Créer une zone de saisie de texte (mot de passe...)
- Sélectionner l'heure

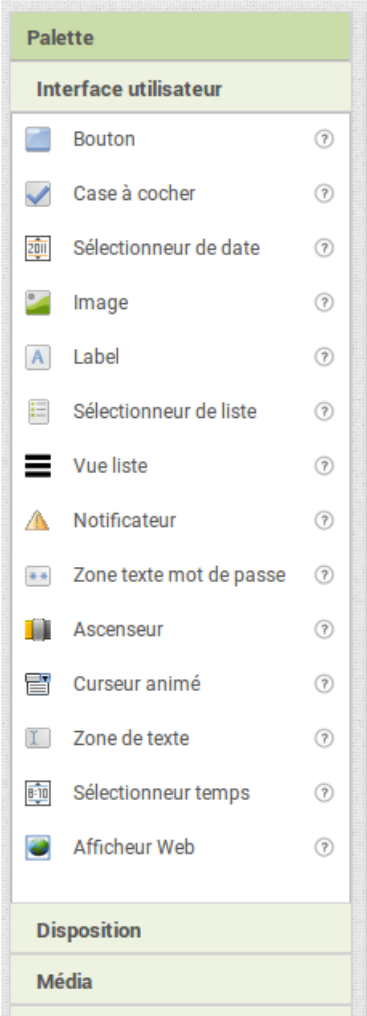


FIGURE 36.11 – La liste des composants

Présentation des propriétés

Nous allons présenter les propriétés principales des composants mis à notre disposition. Tout d'abord, gardez le sélectionneur de liste sur l'écran. Ensuite, pour centrer la liste, allez dans le menu composants, sélectionnez **screen 1** .

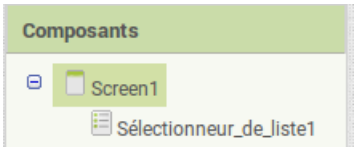


FIGURE 36.12 – Sélection de l'écran

Dans l'onglet propriété → alignement horizontal, sélectionner la liste sur **centrer** .

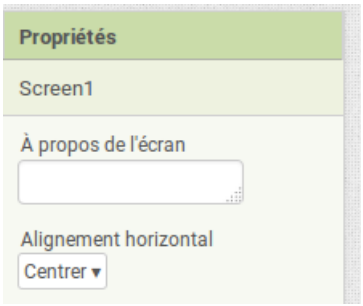


FIGURE 36.13 – Centrage de l'écran

Vous pouvez également changer la forme du bouton.
Allez dans Composants → sélectionneur de liste et dans Propriétés, trouvez l'onglet **forme** et sélectionnez celle voulue.

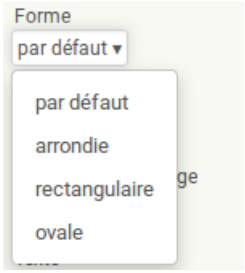


FIGURE 36.14 – Forme du bouton

Voici quelques paramètres de mise en forme :

- Vous pouvez modifier la couleur de la liste en sélectionnant **Couleur de fond**
- Pour modifier la taille de la liste, il suffit de rentrer la taille en pixels ou en % de l'écran (**Hauteur, Largeur**)
  Il vaut mieux préciser en % afin que l'application soit compatible au niveau du format sur tous les appareils.

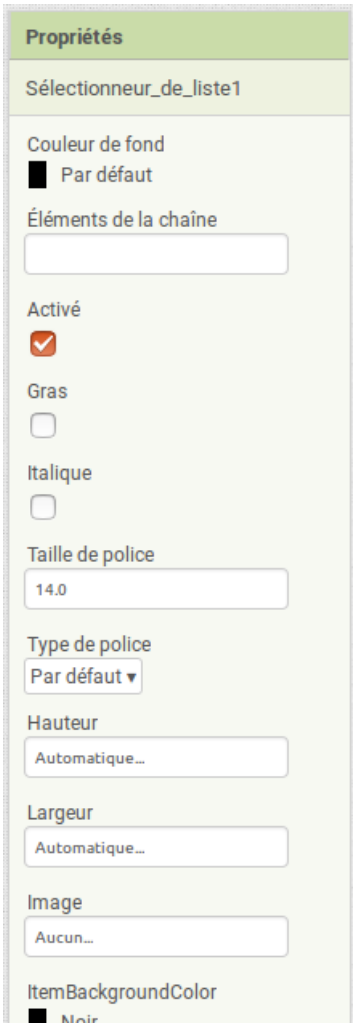


FIGURE 36.15 – La liste des propriétés

Enfin, pour décider du titre de la liste sur l'application, sélectionner la liste **Selectionneur_de_liste1** et dans ses propriétés, modifier l'onglet **texte** et écrivez, par exemple « **Connexion** »

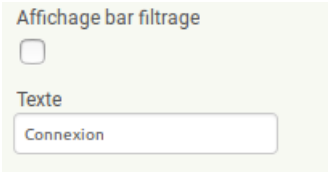


FIGURE 36.16 – Choisir le nom par défaut du bouton **Connexion**

Renommer les élements

Dans un souci de clarté, il convient de renommer les éléments que nous plaçons sur l'écran.

Pour renommer le sélectionneur de liste, il faut allez dans les composants de l'interface, et sélectionnez « Renommer » Je l'ai renommé en « connexion ».

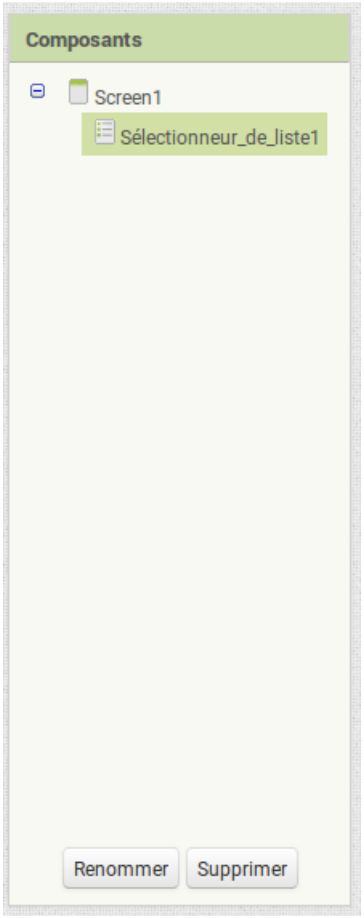


FIGURE 36.17 – Renommer un élement

Astuce

Pour changer la couleur d'arrière-plan de l'application, allez dans composants → screen1 et dans propriétés, sélectionnez couleur de fond

Ajouter un client Bluetooth

Par la suite, nous allons utiliser le client Bluetooth d'App Inventor.
Pour le trouver, il suffit d'explorer le menu latéral. Un ensemble d'éléments est disponible à la suite des principaux élements énumérés dans la section 36.3.1.

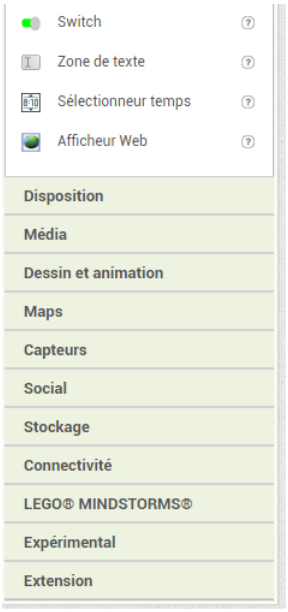


FIGURE 36.18 – Menu latéral

Le client Bluetooth se situe dans la section **Connectivité**

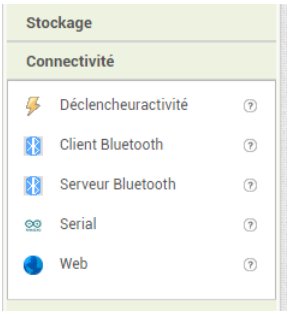


FIGURE 36.19 – Ajouter un client Bluetooth

Pour l'ajouter à noter application, il suffit de glisser le client Bluetooth sur l'interface. A ce moment là, sous l'écran virtuel, le client Bluetooth apparaît.

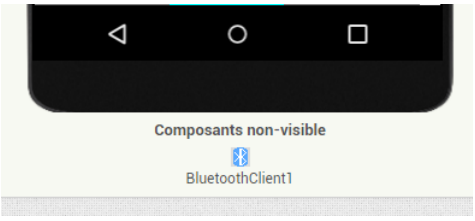


FIGURE 36.20 – Présence du client Bluetooth

Et voila, le client Bluetooth est mis en place.

Premier rendu et agencement des élements

En faisant toutes ces étapes, on obtient :

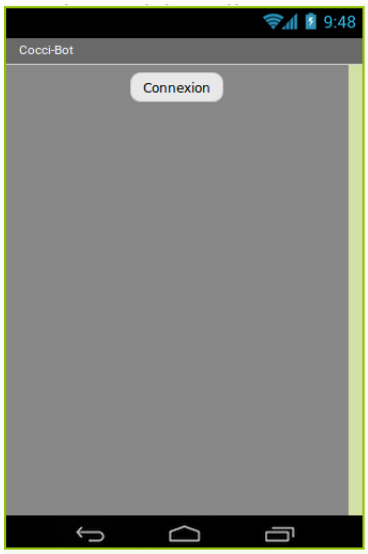


FIGURE 36.21 – Premier rendu de l'application

En ajoutant un bouton (glisser-déposer un bouton disponible dans le menu latéral) et en modifiant ses propriétés (forme, couleur...), on obtient ceci :

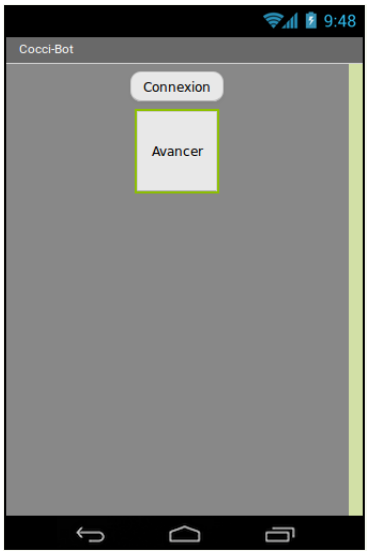


FIGURE 36.22 – Rendu avec un bouton "Avancer"

Générer un espace entre deux élements

On souhaite maintenant espacer le bouton "Connexion" et le bouton "Avancer"

Rien de plus simple ! Il suffit de placer un élément **label** (outil dans menu) entre les deux boutons puis de choisir ses dimensions, ce qui correspondra à l'espacement des boutons.

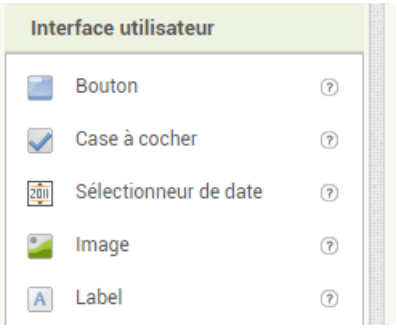


FIGURE 36.23 – Emplacement du label

Pour que le label ne soit pas visible, il suffira de mettre son texte vide.

FIGURE 36.24 – Texte du label

Et voilà le résultat :

FIGURE 36.25 – Espace entre deux élements

Alignement des élements

Maintenant, ce serait bien de pouvoir aligner des boutons (ou autre) horizontalement afin d'obtenir quelque chose comme ceci :

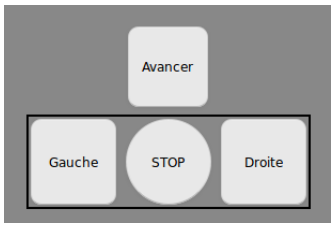


FIGURE 36.26 – Alignement d'éléments

Ne vous inquiétez pas, le cadre noir n'apparaîtra pas sur l'application ; c'est juste un moyen de distinguer les alignements.

Pour faire ceci, allez dans le menu et sélectionnez l'onglet **Disposition**

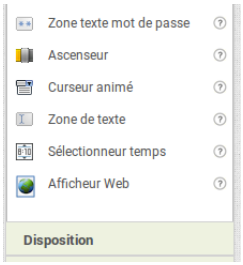


FIGURE 36.27 – Emplacement des éléments d'agencement

Une fenêtre comme ceci apparaît :

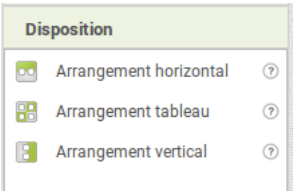


FIGURE 36.28 – Éléments d'agencement

Ensuite, faites glisser l'outil **Arrangement Horizontal** sous la liste « **Connexion** » et vous devriez avoir ceci :

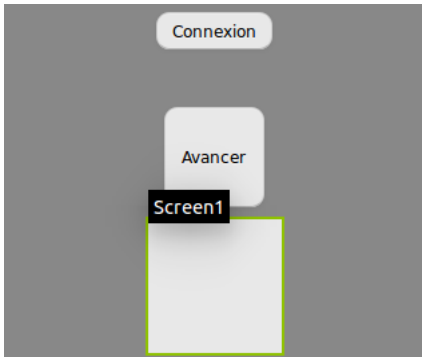


FIGURE 36.29 – Éléments d'agencement placé

Vous pouvez dorénavant faire glisser des boutons dans ce carré et ils se mettront automatiquement côte à côte. Il suffit de régler la taille des boutons sélectionnés dans propriétés.

Astuce

Il est possible de changer les dimensions de l'outil **Arrangement Horizontal**

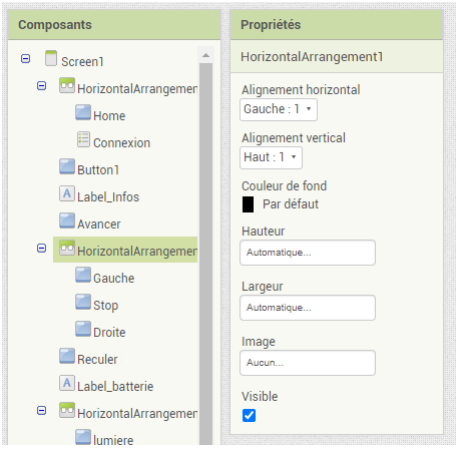


FIGURE 36.30 – Propriétés de l'objet **Arrangement Horizontal**

En revanche, il y a un fond blanc pour l'objet **Arrangement Horizontal** :

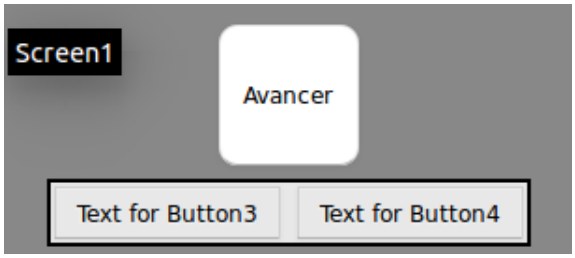


FIGURE 36.31 – Couleur d'arrière-plan de l'outil **Arrangement Horizontal**

En allant dans les propriétés de l'objet **Arrangement Horizontal** , mettez la couleur d'arrière-plan en « **aucun** »

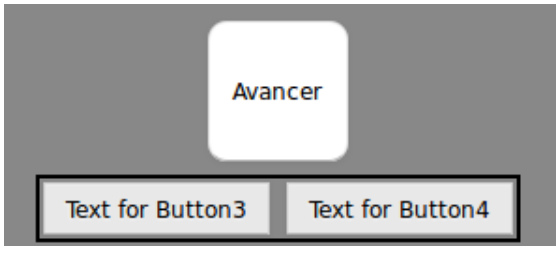


FIGURE 36.32 – Couleur d'arrière-plan transparent

Mise en place de l'ensemble des boutons

Après avoir mis les boutons "**Droite** ", "**Stop** ", "**Gauche** ", "**Reculer** ", "**Batterie** " et "**Auto** " et mis un peu de couleur, vous pouvez obtenir une interface de ce type :

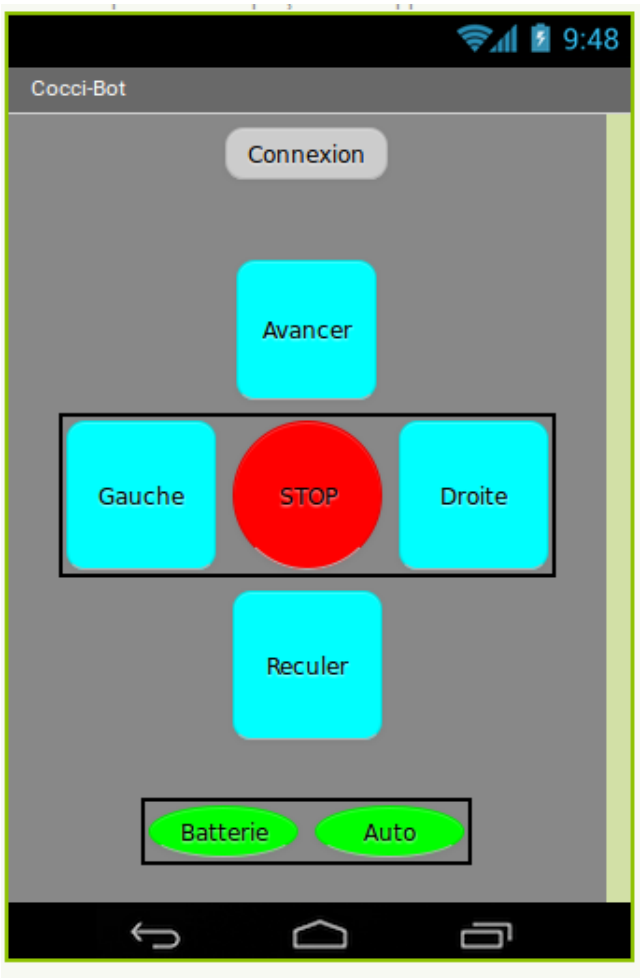


FIGURE 36.33 – Rendu de l'application

Remarque importante

Par la suite, veuillez mettre l'arrière-plan de « connexion » en rouge (non représenté ici)

SECTION 37

GESTION DE L'APPLICATION

Nous avons réalisé et mis en place tous nos élément graphiques. Il ne nous reste donc plus qu'a les faire interagir afin d'envoyer les données Bluetooth.
Pour cela, commencez par aller dans la section **Bloc** , disponible en haut à droite de la page

FIGURE 37.1 – Emplacement du menu "**Bloc** "

Présentation

A l'ouverture du menu **Bloc** , voici le rendu

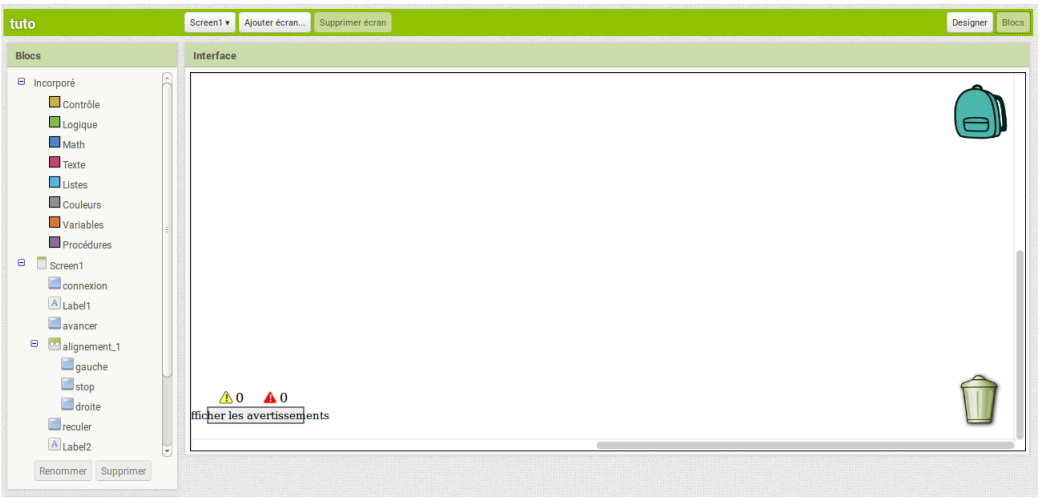


FIGURE 37.2 – Menu "**Bloc** "

Sur le menu de gauche, nous retrouvons une liste de blocs

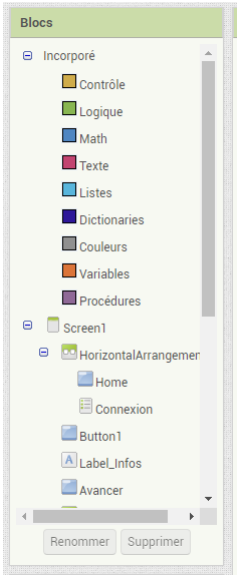


FIGURE 37.3 – Menu latéral

mais également nos élements graphiques agencés précédemment :

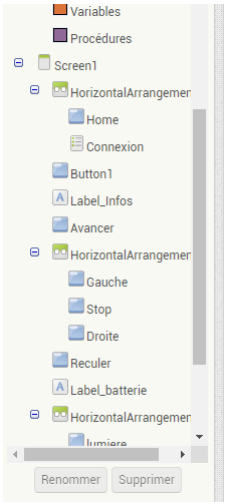


FIGURE 37.4 – Nos élements ajoutés

En cliquant sur chaque onglet (Contrôle, Logique, Math...), une liste de blocs apparaît : (ici, en cliquant sur Contrôle)

FIGURE 37.5 – Blocs **Controls**

Tous ces blocs sont faits pour pouvoir être glissés dans la partie blanche centrale...

Configuration de la liste des périphériques Bluetooth

Principe

Nous allons configurer notre liste **Connexion** afin qu'elle nous indique les modules Bluetooth disponibles.

Cette étape se fera en deux temps :

- Configuration de la liste **avant** le choix de l'utilisateur
- Mise à jour de la liste **après** le choix de l'utilisateur

Commencer par sélectionner la liste **Connexion** dans le menu latéral

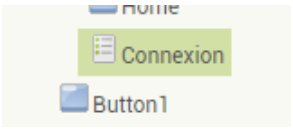


FIGURE 37.6 – Sélection de la liste

Le menu suivant apparaît :

FIGURE 37.7 – Éléments de l'objet **Connexion**

- Le "**Quand connexion.Avant prise** " représente le bloc qui contiendra les instructions de la liste, c'est à dire que dans ce bloc, nous annoncerons que la liste contiendra toutes les adresses Bluetooth disponibles.
- Le "**Quand connexion.Après prise** " représente le bloc d'instructions après avoir choisi le module Bluetooth dans la liste.
  Dans ce bloc, après avoir récupéré l'adresse du module, nous nous connecterons à ce dernier.

Il faut donc placer ces deux blocs dans la zone blanche (glisser-déposer).

Définir les clients Bluetooth disponibles

Le bloc pour définir les élements de la liste **Connexion** est le suivant :

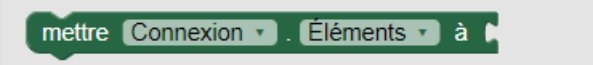


FIGURE 37.8 – Définition de la liste

Il attend en argument (zone de puzzle à droite) une liste.
Nous allons lui donner une liste des modules Bluetooth disponibles grâce à notre Client Bluetooth installé (Section 36.6).

Ce bloc est disponible dans le menu latéral, partie **BluetoothClient1**

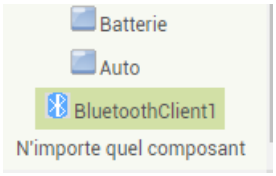


FIGURE 37.9 – Sélection du client Bluetooth

Puis

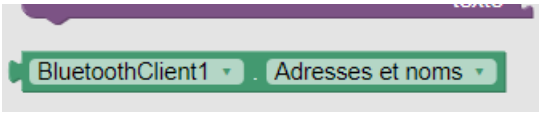


FIGURE 37.10 – Sélection du bloc des adresses

Au final, on obtient le bloc suivant :

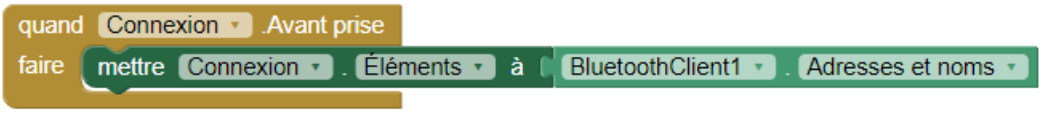


FIGURE 37.11 – Bloc de configuration de la liste

Connexion au module Bluetooth

Une fois que l'utilisateur a sélectionner le module auquel il souhaite se connecter, il faudra mettre à jour l'état de la liste et se connecter au module Bluetooth

Principe

Une fois que la personne a fait son choix dans "**Connexion** ", on vérifie la couleur de l'arrière plan de **connexion** :

- Si la couleur est rouge (par défaut), cela veut dire que nous ne sommes pas connectés. On remplace donc le mot « Connexion » par l'adresse MAC [1] sélectionnée et on définit la couleur d'arrière-plan en vert, pour obtenir ceci :

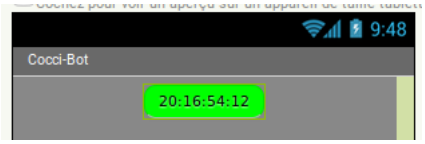


FIGURE 37.12 – Rendu de l'état de connexion

Visuellement, on sait que nous sommes connectés.
Ensuite, pour se connecter réellement, on établit une connexion Bluetooth sécurisée à l'adresse donnée par la sélection de la liste «connexion».
Quand c'est fait, on envoie en Bluetooth la lettre « c » que l'Arduino se chargera d'interpréter, et pourra, en conséquent, jouer une mélodie pour confirmer la connexion.

- Si la couleur est verte , cela veut dire que nous sommes déjà connectés. Il faut donc remettre la couleur et le texte d'origine (« connexion ») et se déconnecter

Emplacement des blocs

Dans le bloc **Quand connexion.Après prise** , nous allons avoir besoin d'une structure de contrôle, c'est à dire en l'occurrence un bloc conditionnel **Si-Alors-Sinon**

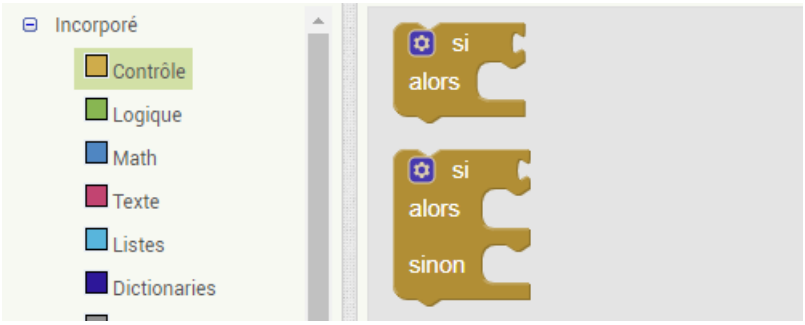


FIGURE 37.13 – Emplacement des blocs conditionnels

Nous aurons également besoin d'un bloc de comparaison :

1. L'adresse MAC est une adresse physique pour identifier de manière unique un composant.

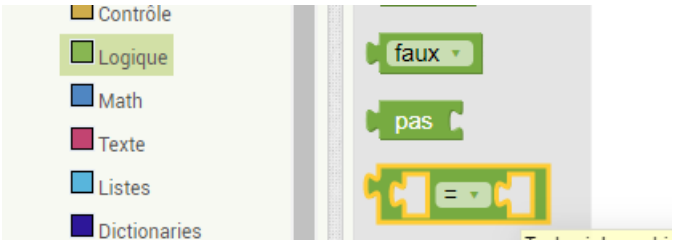


FIGURE 37.14 – Emplacement des blocs de comparaison

Les blocs de couleurs sont disponibles à l'emplacement suivant :

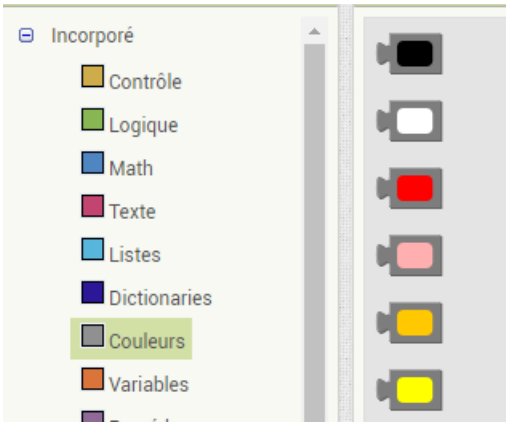


FIGURE 37.15 – Emplacement des blocs de couleurs

Enfin, les chaînes de caractères sont disponibles à l'emplacement suivant :

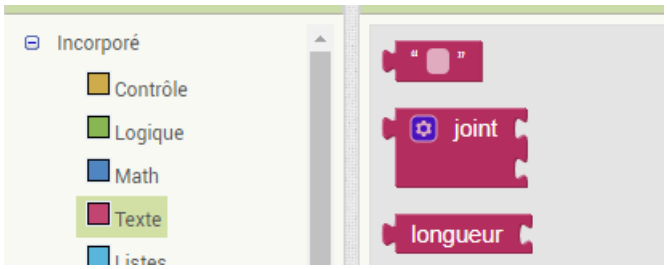


FIGURE 37.16 – Emplacement des chaînes de caractères

Astuce

Pour savoir où se situe une instruction, il faut regarder le nom de l'instruction. Elle sera de la forme :

nom_de_l'instruction.fonction

le "nom_de_l'instruction" sera affiché dans un des onglets de gauche (Figure 37.1)
Par exemple, pour "**mettre Connexion.elements** ", il faut aller dans l'onglet "**Connexion** "

Maintenant que vous savez ou trouver les blocs, je vous invite à recopier ce bloc d'instructions, qui n'est que la mise en pratique de la partie "Principe" de cette section

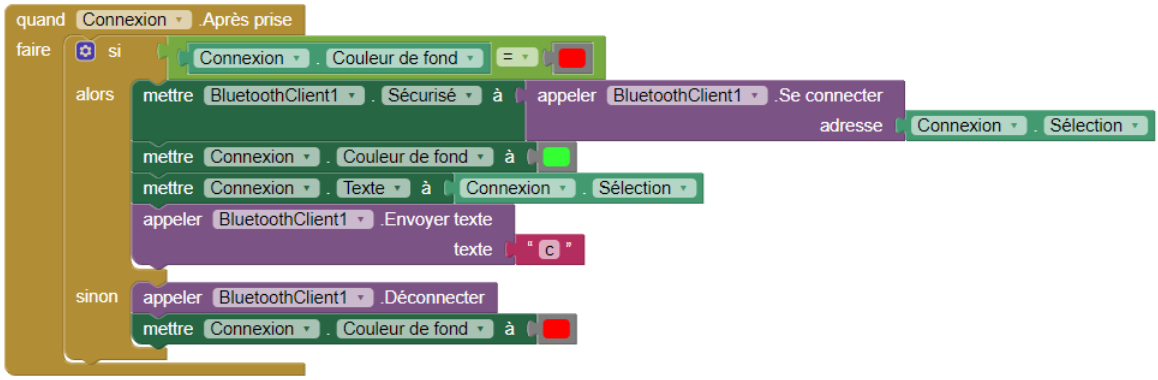


FIGURE 37.17 – Gestion de la liste après sélection

Le plus dur est fait, reste maintenant à gérer les boutons directionels.

Gestion des boutons de direction

Nous allons sélectionner le bouton **Avancer** dans le menu de gauche

FIGURE 37.18 – Menu du bouton Avancer

A l'intérieur de ce bloc, on mettra les instructions pour envoyer le mot « c » en Bluetooth
Pour le langage Arduino, c'est l'équivalent de :

```
if (bouton==appui) {allumer led ;}
```

Equivalent Arduino

Après, nous avons d'autres paramètres (appui long, après avoir retiré le doig, sur le bouton...)

Placez le "**Quand avancer. Click** " sur la page blanche.

Ensuite, nous allons vérifier si nous sommes connectés. Pour cela, nous mettrons un "**Si connexion.Couleur_de_fond = vert** » afin d'attester la connexion.
Enfin, sélectionnez l'onglet "**BluetoothClient1** " et insérez la fonction "**envoyer texte** " en mettant le mot "a" (comme avancer) en texte à envoyer. [2]
Pour obtenir ceci :

2. il faut privilégier de petites chaînes de caractère afin que le module puisse lire plus facilement. Il y a moins de perte de données

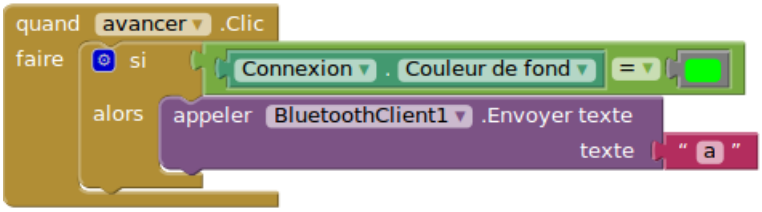


FIGURE 37.19 – Code du bouton Avancer

Il suffit de réaliser la même étape pour tous les boutons. Il faudra juste changer le "**Quand avancer.Click** " en "**Quand 'bouton'.Click** " et ne pas oublier de changer le texte à envoyer.

Par exemple, un deuxième bouton donnerait : ("g" pour gauche, "d" pour droite, "r" pour reculer, "s" pour stop, "b" pour batterie, "A" pour automatique)

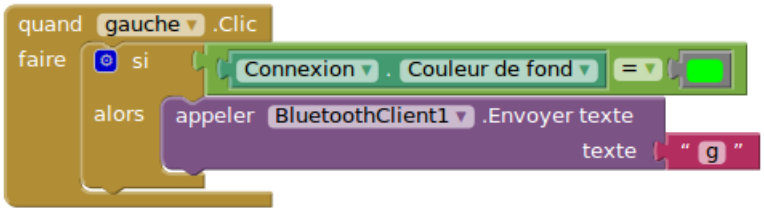


FIGURE 37.20 – Code du bouton Gauche

Après avoir fait ceci pour les autres boutons, l'application est terminée...
Nous pouvons donc passer à l'installation de l'application sur votre téléphone

SECTION 38

INSTALLATION DE L'APPLICATION

Installation de MIT App Inventor

Pour installer l'application que vous avez réalisé, il est préférable d'installer l'application MIT AI2 Companion disponible sur Play Store.
Cette application (relativement lègère (50 Mo) va faire le lien entre le site App Inventor et votre téléphone.
Une fois que votre application est terminée, voici la façon la plus simple pour l'installer :

- Installer MIT AI2 Companion sur votre téléphone

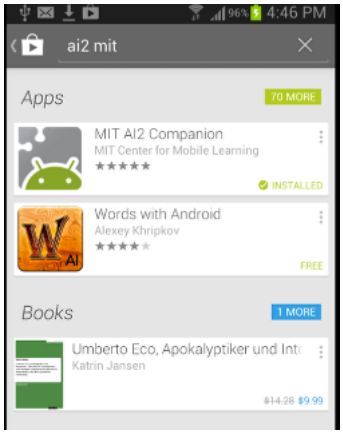


FIGURE 38.1 – Logo de l'application

- Lancer l'application MIT AI2 Companion lorsque'elle est installée **sur votre téléphone**

FIGURE 38.2 – Rendu de l'application MIT AI2 Companion

- Connecter vous au réseau Wifi de votre maison **depuis votre téléphone**
- Ouvrez votre application (Cocci-Bot) dans App Inventor (http://appinventor.mit.edu/) **depuis votre ordinateur**
- Rendez-vous dans le menu du haut, sélectionner **Construire** puis **App (Donnez le code QR pour fichier .apk)**

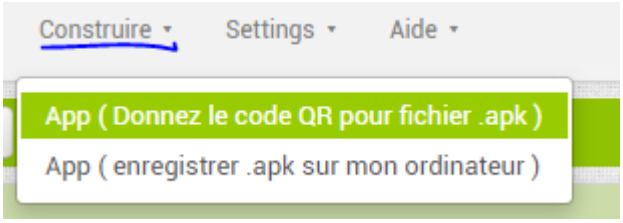


FIGURE 38.3 – Emplacement du menu "Construire"

Une barre de progression devrait apparaître.

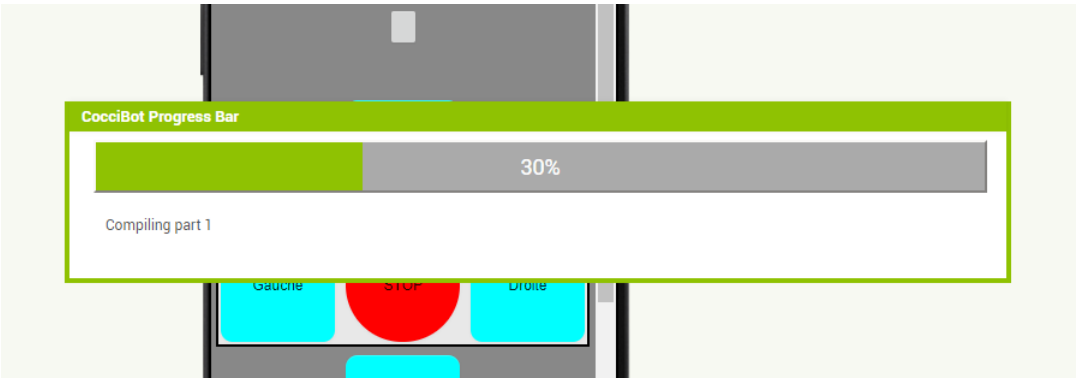


FIGURE 38.4 – Barre de progression

Une fois la barre de progression disparue, un QR Code apparaît.
Ne surtout pas cliquer sur **ok**

FIGURE 38.5 – QR code

- Il est temps de scanner le code avec l'application MIT AI2 Companion sur votre téléphone ("Scan with QRCode"). Une fois le processus terminé, vous pouvez cliquer sur **ok** , il faudra suivre les consignes sur votre téléphone portable (gestion des autorisations...)
- Et voila !

Nous pouvons passer au code Arduino et au traitement des données.

SECTION 39

TRAITEMENT DES DONNÉES

Branchements

Comme dit au chapitre 35, il faut relier le « +5V » du module au 5 Volts de la carte Arduino et la masse du module à celle de la carte.
Ensuite, nous allons relier la broche TX du module à la broche 12 de la carte et la broche RX à la broche 10.
Voilà, le branchement est terminé.

Code Arduino

Point-clé

Afin de lire les données du module, nous allons "émuler" une voie série, en l'occurrence les broches 10 et 12.
C'est-à-dire que nous allons déclarer que ces broches recevront et enverront des données.

Pour cela, il faut utiliser la bibliothèque **SoftwareSerial** . Dans le logiciel Arduino, allez dans **croquis** puis **inclure une bibliothèque** et sélectionnez **SotfwareSerial** .

Maintenant, nous allons définir les broches du module (toujours avant le « void setup ») :

```
const int RX = 10;
const int TX = 12
```

Définitions des broches RX et TX

Après ceci, il faut déclarer un objet **SoftwareSerial** qui prendra en argument respectivement les broches TX et Rx, un peu comme lors de la déclaration d'un écran LCD (« LiquidCrystal lcd (RS,E,D4,D5,D6,D7) ; »).

On obtient donc :

```
#include <SoftwareSerial.h>

const int RX = 10;
const int TX = 12;
```

```
SoftwareSerial crius(RX,TX);
```

Définition de l'objet SoftwareSerial

Bien entendu, "crius" peut être remplacé par ce que vous voulez.
Ensuite, on déclare que la communication carte-module peut débuter avec "crius.begin(115200) ;"
Où 115200 correspond à la vitesse de transmission en bauds (comme "Serial.begin(115200) ;") ;

```
SoftwareSerial crius(RX,TX);
void setup() {
    crius.begin(115200);
}
```

Vitesse de communication

Remarque importante

Par la suite, en cas d'erreurs de transmission Bluetooth (pas de données...), il conviendra de vérifier le branchement des broches RX et TX (essayer de les intervertir) et d'éventuellement changer la vitesse de communication car certains modules communiquent à 9600 bauds !

Pour lire les données du module, ce sont presque les mêmes fonctions que pour le port série : En effet :

Pour le port série :

- Serial,begin(115200) ;
- Serial.available() ;
- Serial.read() ;
- Serial.print() ;
- Serial.println() ;

Pour le module Bluetooth :

- crius,begin(115200) ;
- crius.available() ;
- crius.read() ;
- crius.print() ;
- crius.println() ;

Tant que des données (caractères) sont disponibles, nous allons les assembler en une chaîne de caractère (concaténation).
Ensuite, avec un **if** , nous allons voir si cette chaîne en question correspond par exemple à "a".
Si c'est la cas, nous allons appeler la fonction **robot.enAvant(100) ;**

Il faut donc définir un caractère x et une chaîne de caractère. Donc dans le programme Arduino, avant la fonction **setup** , on rajoute :

```
#include <SoftwareSerial.h>

const int RX = 10;
const int TX = 12;

char c;
String message;
```

Définition des structures du message

La boucle while va permettre de lire les données : Dans la boucle **loop** : on écrit :

```
void loop() {

   while() {

   }//Fin while

}//Fin void loop
```

Boucle de lecture partielle

Maintenant que la boucle va attendre des données, il suffit de les lire et de les transformer en chaîne de caractère.
Pour cela :

- on lit le premier caractère c
- on définit que la chaîne message = message + c

On obtient :

```
void loop() {

   while(crius.available()>0) {

       c = crius.read();
       message = message + c;
   }//Fin while

}//Fin void loop
```

Boucle de lecture complète

La structure conditionnelle est très simple :
Après la boucle « while », mettez :

```
void loop() {

   while(crius.available()>0) {

       c = crius.read();
       message = message + c;
   }//Fin while

   if(message=="c") {

       message="";
       Melodie(1);
   }//Fin if message=="c"

}//Fin void loop
```

Structure conditionnelle

« c » correspond au message envoyé par l'application lors de l'appui du bouton connexion.

Attention!!! il est important de vider la chaîne de caractère avec "message= " ";", sinon, lorsque le module recevra un autre message (par exemple "s"), la chaîne sera égale à "cs"

Après avoir fait ceci pour les boutons connexion, avancer, droite, gauche, stop, batterie et reculer, voici le résultat :

```
void loop() {

   while(crius.available()>0) {

       c = crius.read();
       message = message + c;
   }//Fin while

   if(message=="c") {

       message="";
       Melodie(1);
   }//Fin if message=="c"

 if(message=="c")  {message="";
                    Melodie(1);}
```

```
 else if(message=="a"){message="";
                       robot.enAvant(100);
                       etatMoteurs=true;}

 else if(message=="r")   {message="";
                          robot.enArriere(100);
                          etatMoteurs=false;}

 else if(message=="d")   {message="";
                          robot.tourneDroite(100);
                          etatMoteurs=true;}

 else if(message=="g")    {message="";
                           robot.tourneGauche(100);}

 else if(message=="s") {message="";
                             robot.arret();}

 else if(message=="b")  {message="";
                         TestBatterie();}

 else if(message=="A")  {gestion_mouvement();}

 }//Fin void loop
```

Code des if

Pour finir, nous allons nous occuper de la partie automatique, qui s'activera avec l'appui sur le bouton auto.
Le début est le même, avec le **if** , en revanche, cette fois-ci, nous ne viderons pas la chaîne de caractère.

```
void gestion_mouvement() {

 while(message=="A") {

 robot.enAvant(100);

   distance = moyenneMesure(30, A2);
   fin_automatique();
   if(distance>500) {

     robot.tourneGauche(100);
     delay(1500);
     distance = moyenneMesure(30, A2);
     fin_automatique();
     if(distance>500) {
```

```
        robot.tourneDroite(100);
        delay(3000);

        distance = moyenneMesure(30, A2);
        fin_automatique();
        if(distance>500) {

            robot.tourneGauche(100);
            delay(1500);
        }
        else{robot.enAvant(100);}

      }
      else  {robot.enAvant(100);}

    }
     else {robot.enAvant(100);}

 }//fin while
 }//fin gestion mouvement
```

Code gestion_mouvement()

On remarque que dans cette fonction, il y a une boucle "while". La boucle "while(message== "A") " s'exécutera tant que l'on n'appuiera pas sur un autre bouton ;
En effet, si j'appuie sur le bouton stop, la chaîne message ne sera plus égale à "A", la boucle "while" ne sera plus respectée et le robot s'arrêtera.

Afin de sortir de la boucle "while", il faut intercaler dans l'algorithme plusieurs fois une fonction **fin_automatique** » qui comprend :

```
 void fin_automatique() {

if(bluetooth.available()>0) {
  robot.arret();
  message="";
  etatMoteurs=false;

}//fin if

}//fin fin_automatique
```

Code fin_automatique()

Dès qu'une donnée est présente, cela arrête le robot, vide la chaîne de caractère. Le programme revient donc à la boucle "while" de départ, au tout début du **void loop** .

Si vous voulez ajouter un bouton sur l'application, vous savez le faire. Pour ajouter ce bouton dans le code Arduino, il suffit de rajouter une ligne :

```
if(message=="valeur envoyée depuis l'application")   {message="";
                              action();}
```

Code d'ajout de touche

Conclusion

Le programme complet du cocci-bot, l'application (que vous pourrez modifier) ainsi que les branchements se situent en annexe du dossier

Onzième partie

Projet MySensors

Documentation du projet MySensors

SECTION 40
INTRODUCTION

Présentation

Ce document a pour but d'expliquer la mise en place d'une passerelle et d'une sonde MySensors.

Organigramme

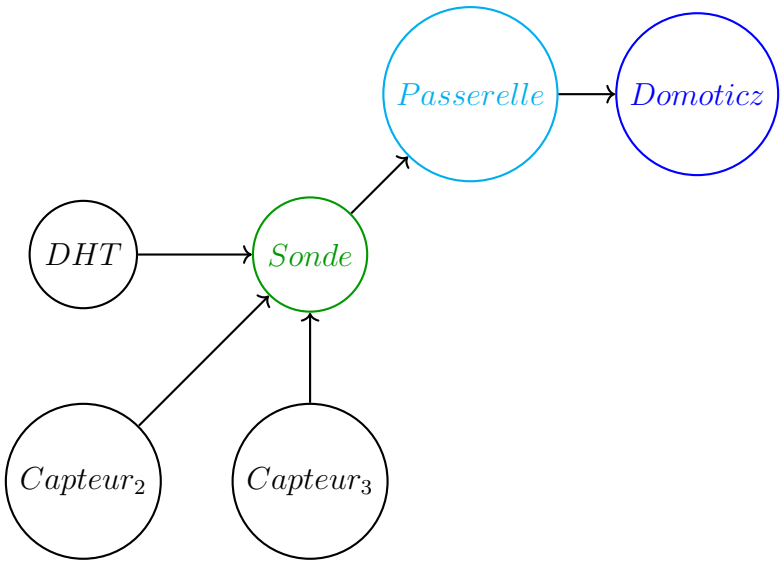


FIGURE 40.1 – Les différents composants du projet

Principe

Les capteurs vont être analysés par la sonde MySensors.
Cette dernière enverra à distance les informations vers la passerelle qui se chargera d'envoyer les informations au serveur Domoticz via une liaison USB.

Une sonde représente un endroit physique, un lieu de mesure.
Si vous souhaitez par la suite faire d'autres relevés dans un endroit différent, il suffira d'ajouter

une sonde et de garder la passerelle.
Chaque sonde est caractérisée par un identifiant de noeud (NODE_ID) et chaque capteur possède un identifiant enfant sur la sonde qui lui est rattachée (CHILD_ID)

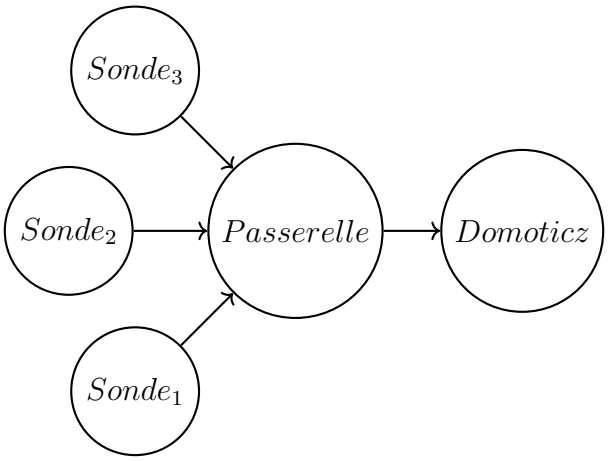


FIGURE 40.2 – Une extension possible

Structure du projet

Nous vous invitons à garder la structure suivante pour le projet :
Dans un dossier DIR Domoticz_Crepp, placez deux dossiers appelés DIR Sonde_MySensors et DIR Passerelle_MySensors
Ces deux derniers dossiers contiendront respectivement le programme de la sonde et de la passerelle.

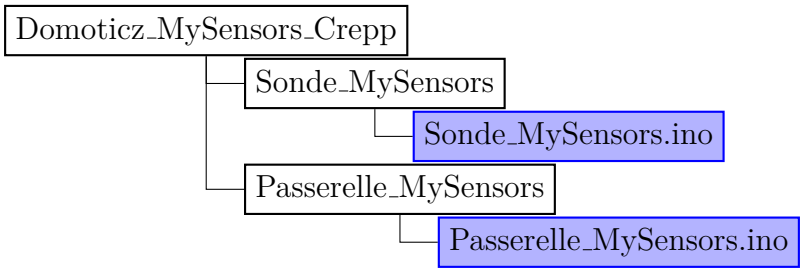


FIGURE 40.3 – Arborescence du projet

Occupons nous maintenant des bilbiothèques Arduino.

SECTION 41

BIBLIOTHÈQUES ARDUINO

Installation des bibliothèques

Lors de la première compilation d'un programme, il se peut que des bibliothèques soient manquantes. C'est ce que nous allon voir. Pour cela, ouvrez le programme de la sonde (**Sonde_MySensors.ino**) sans brancher de carte Arduino.

Ensuite, cliquez sur le bouton **Vérifier** (bouton de gauche) et patientez quelques secondes.

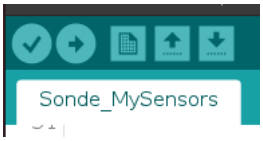


FIGURE 41.1 – Bouton de vérification

Si la bibliothèque MySensors est manquante, vous obtiendrez l'erreur suivante :

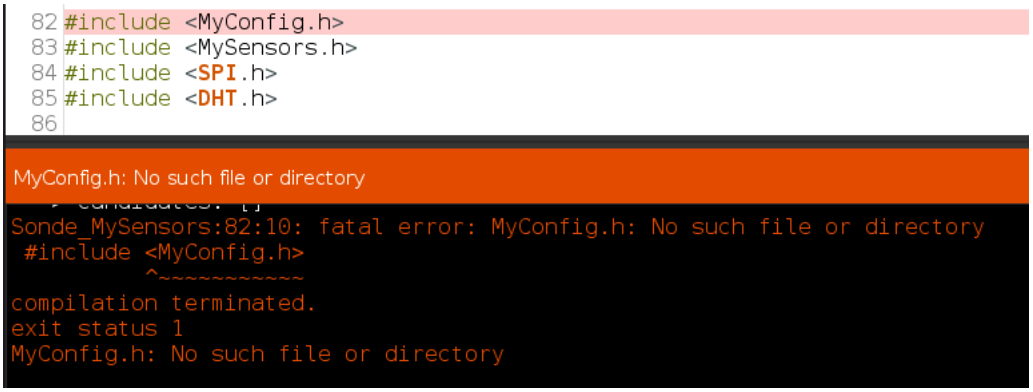


FIGURE 41.2 – La bibliothèque MySensors manquante

Pour installer la bibliothèque, il suffit d'aller dans **Croquis ¿ Inclure une bibliothèque ¿Ajouter la bibliothèque .ZIP**

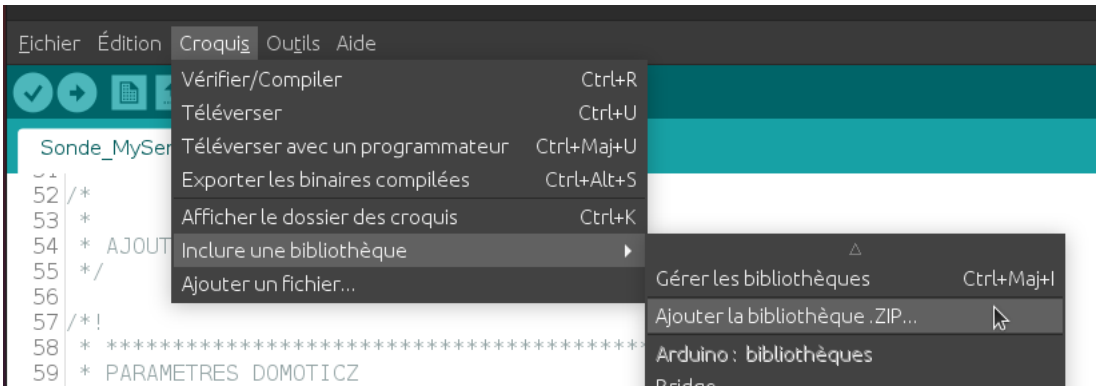


FIGURE 41.3 – Ajout d'une bibliothèque

Il ne reste qu'à trouver le fichier **Bibliothèque_MySensors.zip** et à faire **OK**

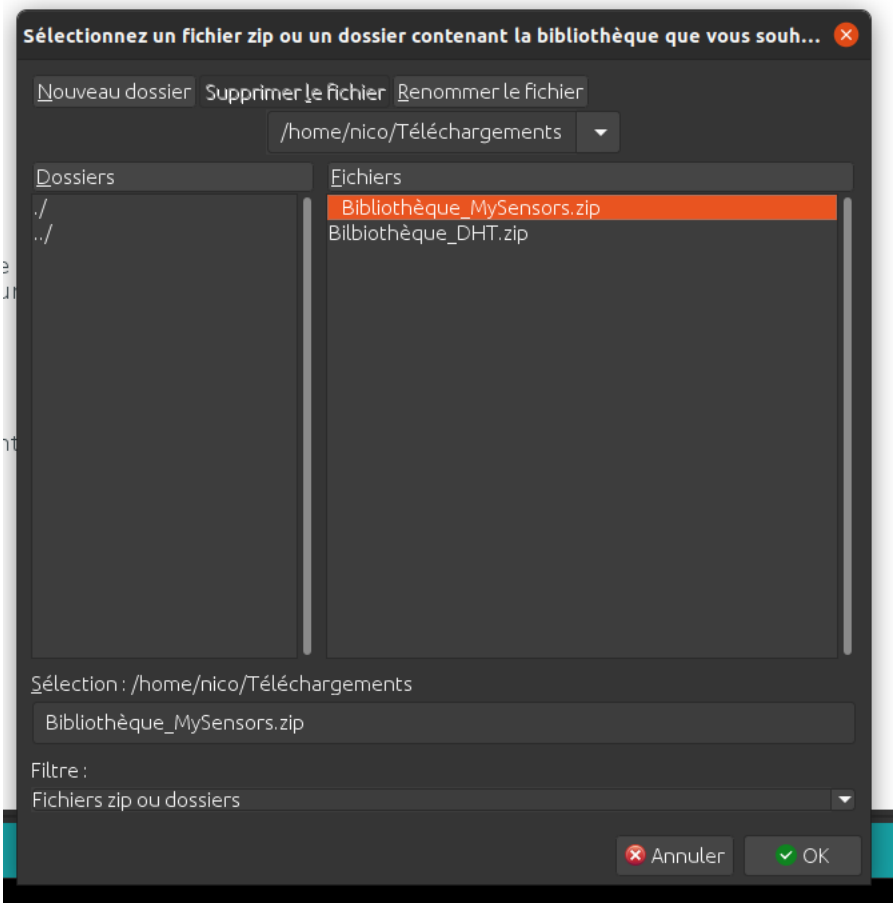


FIGURE 41.4 – Sélection du fichier ZIP

Un message de confirmation d'ajout est affichée en bas de la page du logiciel Arduino.

FIGURE 41.5 – La bibliothèque MySensors est ajoutée

On clique à nouveau sur le bouton **Vérifier** pour afficher les éventuelles erreurs.

Important

Dans certains cas, la bibliothèque **Adafruit_sensor** est manquante, il faut installer le fichier **Bibliothèque_Adafruit_sensor.zip** disponible en annexe.

On refait ensuite le bouton **Vérifier** et si la bibliothèque **DHT** n'est pas installé, on obtient de nouveau :

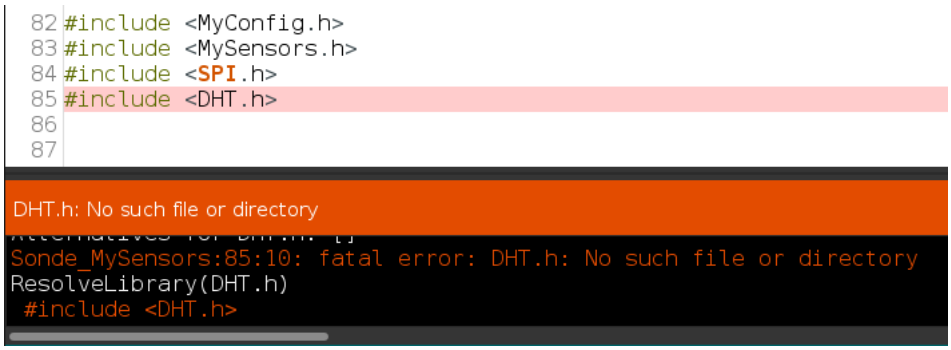


FIGURE 41.6 – La bibliothèque DHT manquante

On procède de la même façon, on va importer le fichier **Bibliothèque_DHT.zip** dans **Croquis ¿ Inclure une bibliothèque ¿Ajouter la bibliothèque .ZIP**

Une fois toutes les bibliothèques installées, le message suivant apparait :

FIGURE 41.7 – La compilation est terminée

On va ensuite s'occuper de la vérification de la communication entre les cartes Arduino et l'ordinateur.

SECTION 42

PROGRAMMATION PRO-MINI

Programmer une carte Arduino pro-mini avec une carte Arduino évite d'acheter un module FTDI.
De plus, la carte Arduino Uno pourra être réutilisée pour d'autres projets.

L'objectif est de programmer la carte Pro-mini sur la sonde MySensors.

Liste du matériel

- 5 câbles Dupont mâles-femelles [1]

FIGURE 42.1 – Les câbles de connexion

- Une carte Arduino Pro-Mini

1. Il est possible de faire des liaisons mâles-femelles avec des câbles mâles-mâles et femelles-femelles

FIGURE 42.2 – La carte Arduino Pro-mini

- Une carte Arduino Uno

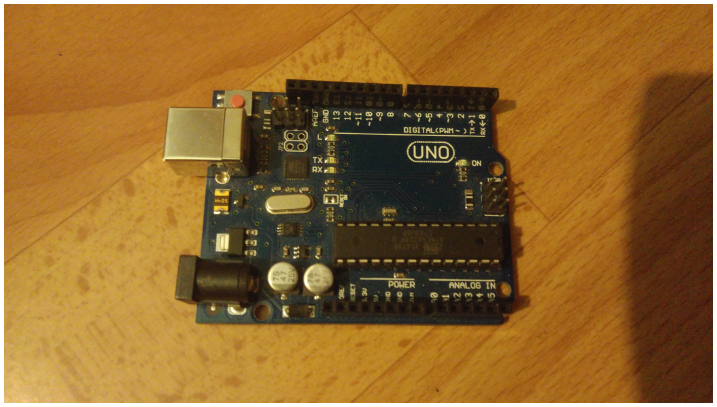


FIGURE 42.3 – La carte Arduino Uno

Important

Il faut retirer le microcontrôleur de la carte Arduino Uno pour pouvoir programmer la carte Pro-Mini
Pour le retirer, on prend un petit tournevis et on soulève délicatement la puce. On prendra le soin de repérer l'orientation de la puce sur la carte (méplat vers l'extérieur de la carte)

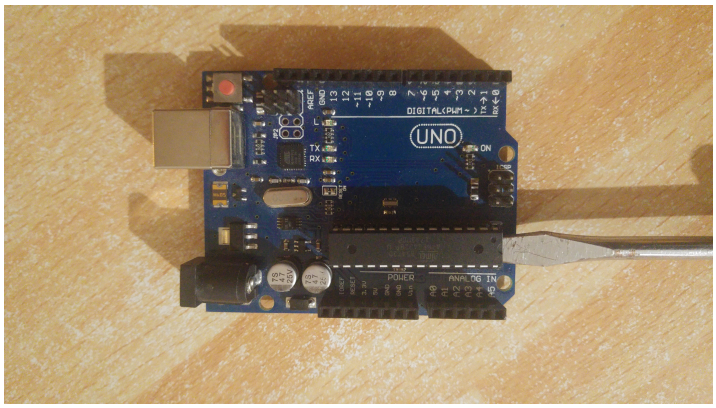


FIGURE 42.4 – On retire le microcontrôleur

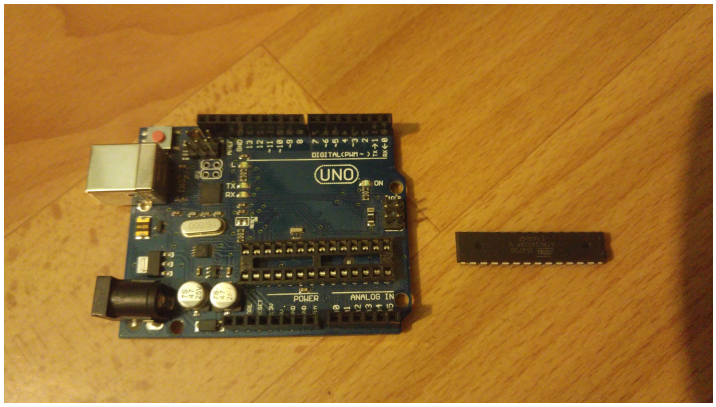


FIGURE 42.5 – La carte Arduino Uno sans son microcontrôleur

Branchements

Attention

La carte Arduino Pro-Mini doit être alimentée en **3.3V** et non en 5V !
La carte Arduino Pro-Mini ne doit pas être placée sur son support de sonde lorsque elle est en train d'être programmée !

Voici les connexions à faire pour programmer la Pro-Mini :
Le mot PIN XXXX_UNO représente une broche de la carte Arduino UNO et PIN XXXX_PRO-MINI représente une broche de la carte Arduino Pro-Mini.
XXXX est l'indication du nom de la broche.

- PIN RESET_UNO vers PIN RST_PRO-MINI
- PIN +3.3V_UNO vers PIN VCC_PRO-MINI
- PIN GND_UNO vers PIN GND_PRO-MINI
- PIN RX_UNO vers PIN RX_PRO-MINI
- PIN TX_UNO vers PIN TX_PRO-MINI

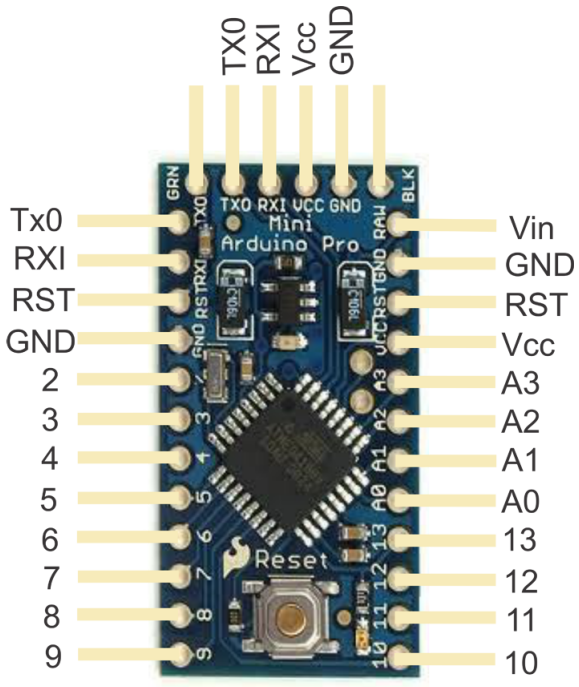


FIGURE 42.6 – Les broches du Pro-Mini

Remarque

Ici, la liaison série (**RX** et **TX**) n'est pas croisée, le **RX** de la carte Uno va sur le **RX** de la Pro-Mini, idem pour le TX

Vous pouvez ouvrir le programme Arduino que vous désirez charger [2] sur la carte Arduino Pro-Mini. Voici un programme minimal pour faire clignoter la LED du pro-mini.
Ce programme est disponible en allant, dans le logiciel Arduino, dans la section **Fichiers ¿**

2. Vous pouvez charger le programme de clignotement de la LED pour l'exemple

Exemples ¿ basics ¿ Blink

```
  void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage
   level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage
   LOW
  delay(1000);                       // wait for a second
}
```

Programme d'exemple Blink

Une fois le programme ouvert, voici les étapes pour compiler le programme.

Téléversement

- 1) Sélectionner la carte **Arduino Pro-mini** dans **Outils ¿ Types de carte**

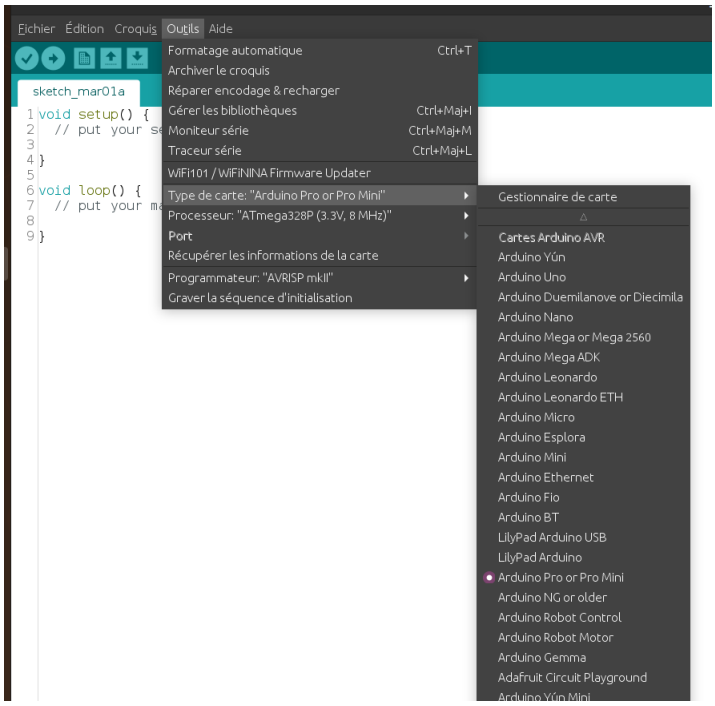


FIGURE 42.7 – Type de carte

▶ 2) Sélectionne le processeur **ATmega328P, 3.3V, 8Mhz** dans **Outils ¿ Processeur**

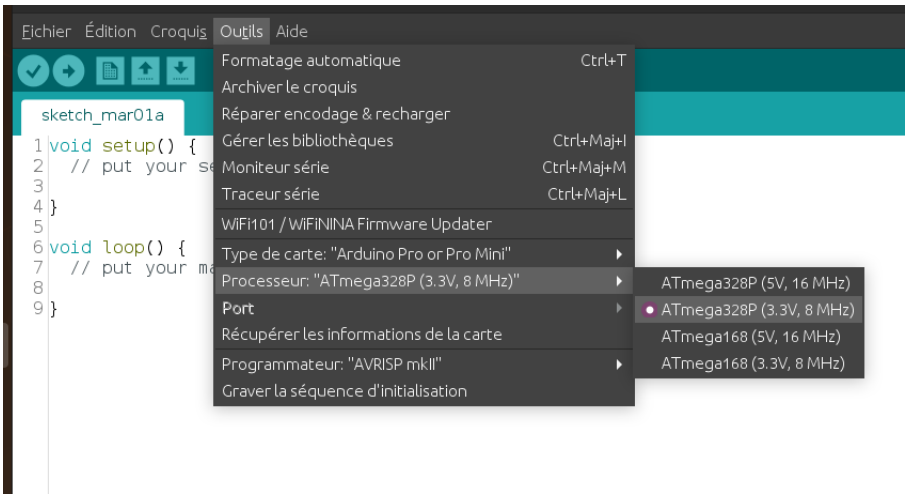


FIGURE 42.8 – Type de processeur

Avertissement

N'oubliez pas de sélectionner le port de communication de l'Arduino

Il ne vous reste plus qu'à cliquer sur le bouton de téléversement du programme. La LED de la carte Pro-Mini devrait clignoter.

Information

Ici, nous avons chargé un programme de test, par la suite, il conviendra de charger le programme **Sonde_MySensors.ino** .
Cette étape de chargement de programme sera nécessaire à chaque modification du code de la sonde.

Programmation de la Nano

La carte Nano étant reliée à l'ordinateur par un câble USb, sa programmation sera plus aisée. On alimente la carte via l'ordinateur, on sélectionne le type de carte (**Type de carte ¿ Arduino Nano**),
le type de processeur (**Outils ¿ Processeur ¿ Old bootloader**), le port puis on téléverse le programme désiré.

SECTION 43

MONTAGE DE LA PASSERELLE

Rappels

La passerelle reçoit sur sa barrette connecteurs femelles un **Arduino Nano** et est reliée par une nappe à 8 conducteurs à un **module transmetteur NRF24** .

L'Arduino Nano est relié au Raspberry Pi de votre plateforme Domoticz par un câble USB (transmission d'infos et alimentation 5v) et alimentera la passerelle en 5v.

Le module transmetteur NRF24 assure la liaison radio avec les sondes.

Liste du matériel de la passerelle

- 3 Led
- 3 résistances de 270 Ω
- 2 condensateurs électrolytiques (100 $\mu F, 16\ V$ – cylindriques noirs)
- 2 condensateurs céramiques monolithiques ($100nF, 50\ V$ – couleur jaune foncé)
- 1 régulateur 3.3v HT7533-1
- 1 module transmetteur NRF24
- 1 bouton poussoir
- 1 circuit imprimé
- 1 barrette connecteurs femelle (déjà montée sur le circuit imprimé)
- 1 nappe 8 conducteurs (dont l'un porte un liseré rouge),
- 2 jumpers

Placement des composants

Vue de dessus du circuit

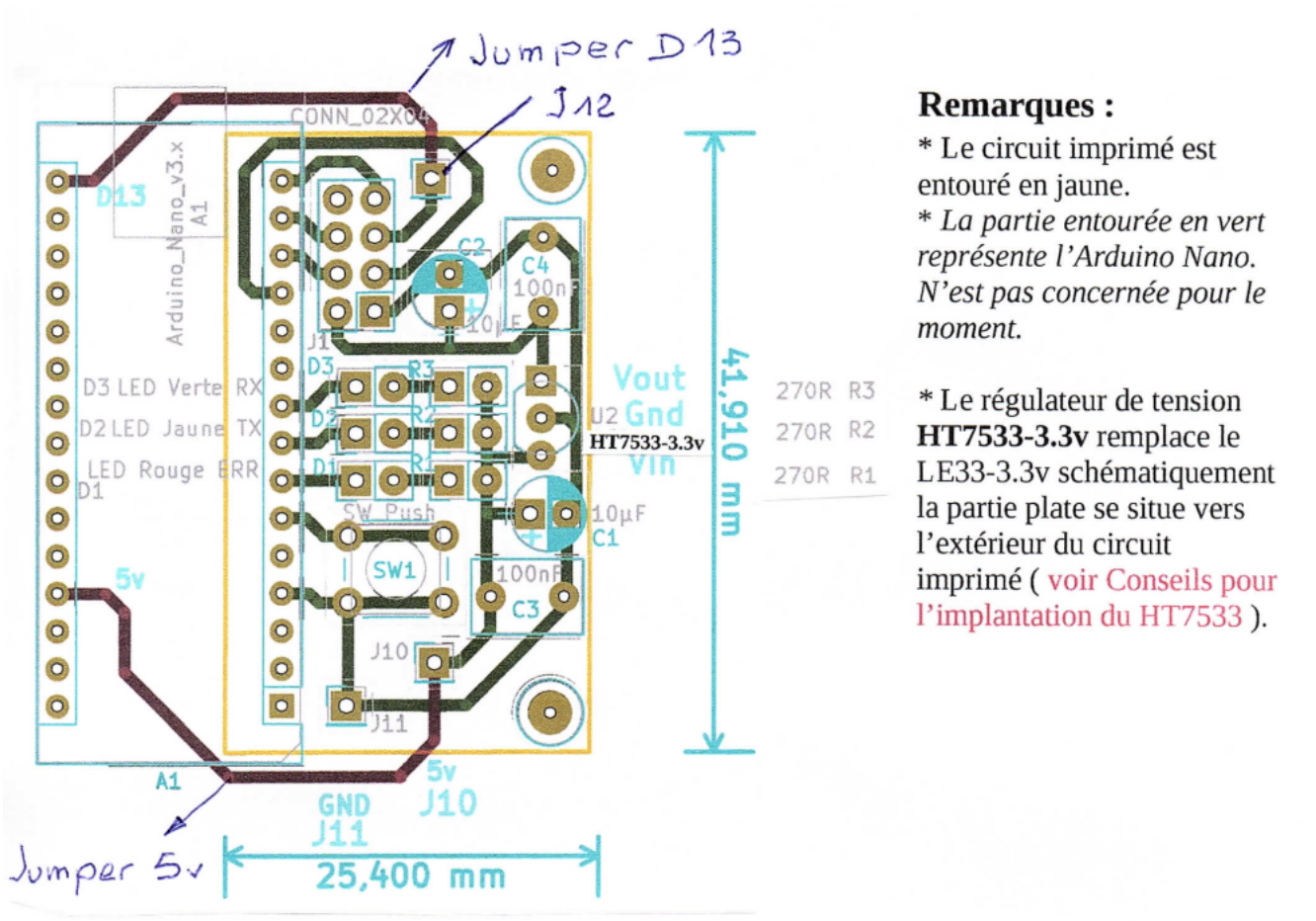


FIGURE 43.1 – Circuit imprimé vu de dessus, coté composants

Étapes

1. Souder la barrette déjà en place

Remarque

Il faudra faire très attention au placement de la carte Arduino nano par la suite : La broche D13 de la Nano doit être impérativement dans le trou le plus avancé (coté nappe de fils).
Un décalage de 1 trou lors du placement de la carte Nano dans sa barrette pourra endommager la carte Nano et le module RF24 !

2. Souder les 8 brins de la nappe en respectant les consignes suivantes :
   - Séparer les 8 conducteurs sur 2 cm environ.
   - Chaque conducteur étant multibrins, s'assurer qu'ils sont bien torsadés puis étamer.
   - Respecter l'ordre de soudage -¿ Fil au liseré rouge en 1 puis conducteurs suivants en 2,3 etc.

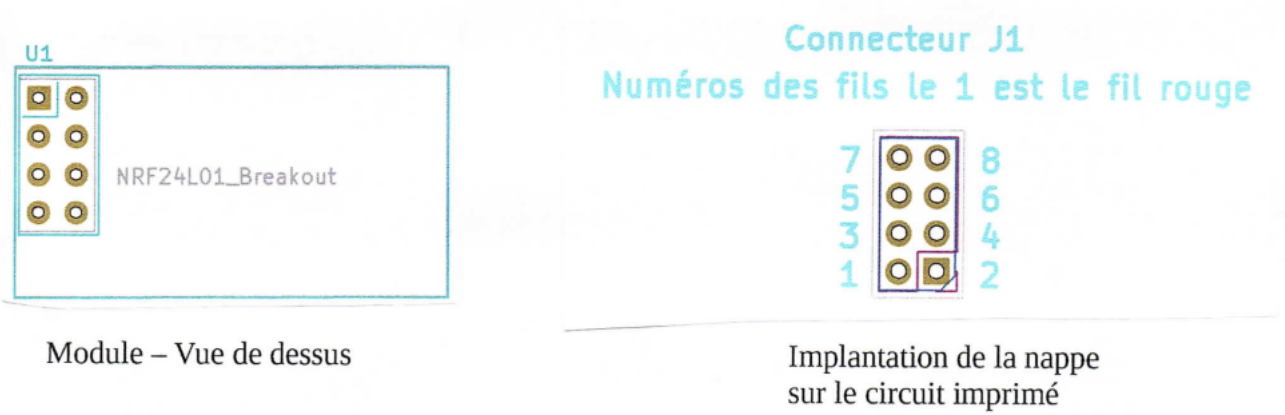


FIGURE 43.2 – Emplacement de la nappe

Remarque

Ne pas se tromper sur la soudure de la nappe
En gardant la vue de la Figure 3.1, le câble rouge de la nappe est en bas à gauche, le n°2 est en bas à droite, etc...

Puis dans l'ordre que vous souhaitez :

- Led rouge en D1 – Led jaune en D2 – Led verte en D3 (sont polarisées, patte longue au + , patte courte, méplat sur la led au -),
- Les 3 résistances sont à souder en R1, R2 et R3
- Les 2 condensateurs électrolytiques sont à souder en C1 et C2 (sont polarisées, le corps du condo est noir avec une bande grisée, la patte de ce côté est le -),
- Les 2 condensateurs céramiques sont à souder en C3 et C4
- Le bouton poussoir en SW1
- Le jumper D13 (Arduino Nano) en J12
- Le jumper 5v (Arduino Nano) en J10
- Le régulateur 3.3v HT7533 est à souder suivant les conseils de la section suivante

Mise en place du régulateur de tension

Comme vous l'avez sûrement remarqué, l'implantation précédente correspond à un régulateur LE33 et non à un HT7533

Si vous voulez utiliser un HT7533, il faut adapter le brochage du HT7533 au circuit.

- Vin : Entrée 5v
- GND : la masse
- Vout : sortie 3.3V

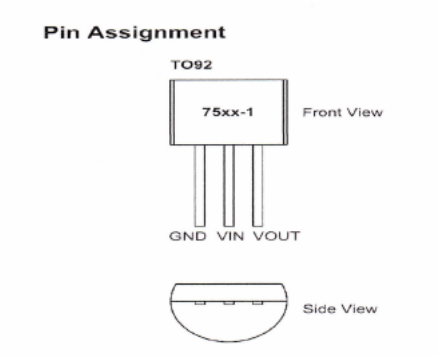


FIGURE 43.3 – Broches du HT7533

Adaptation des broches du HT7533 au schéma de la passerelle

Il faut que les broches **GND, Vin et Vout** rentrent dans les mêmes broches que celle du schéma de la passerelle. Même si les broches ne sont pas dans le même ordre, c'est assez simple à faire en tordant les broches du HT7533 avec une petite pince plate.

Sur le HT7533, sans que les broches se touchent, **on tord Vin vers l'avant, Vout vers l'arrière et on ramène GND au milieu.**

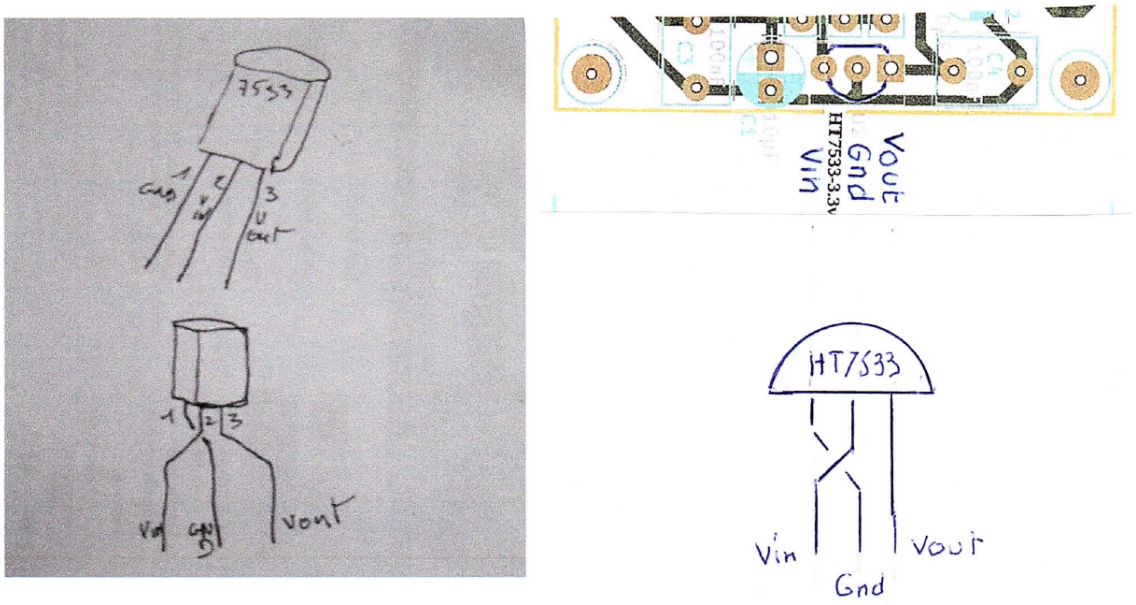


FIGURE 43.4 – Insertion du HT7533

Rendus

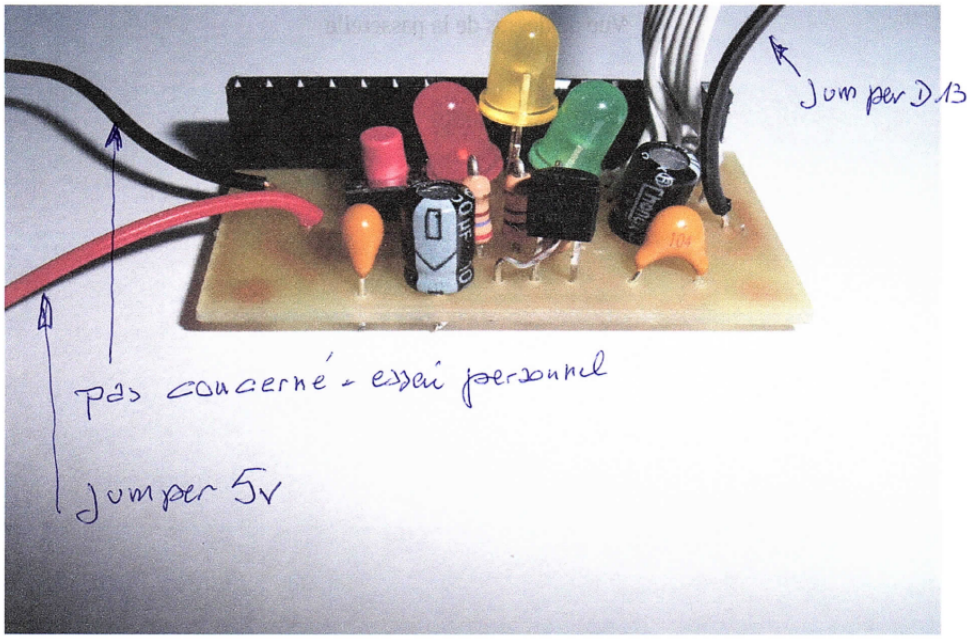


FIGURE 43.5 – Vue de coté

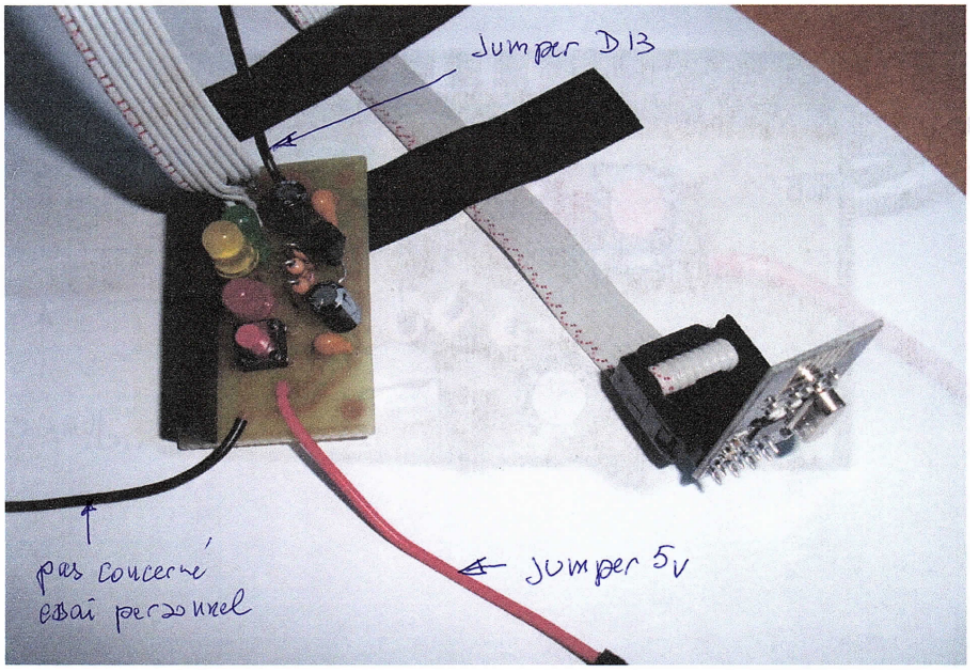


FIGURE 43.6 – Vue de la passerelle et de l'émetteur

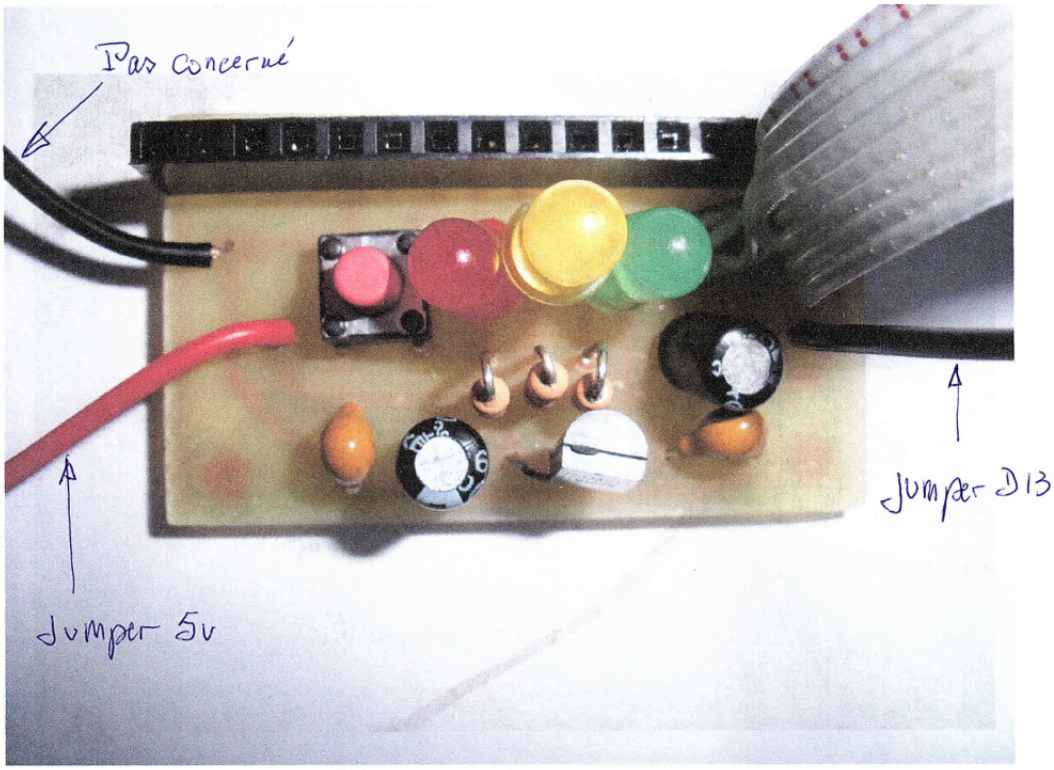


FIGURE 43.7 – Vue de dessus

SECTION 44

MONTAGE DE LA SONDE

Rappels

La sonde reçoit sur sa barrette connecteurs femelles un **Arduino Pro-Mini** et est reliée par une nappe à 8 conducteurs à un **module transmetteur NRF24** . Le module transmetteur NRF24 assure la liaison radio avec la passerelle.

Liste du matériel de la sonde

- 1 pcb
- 2 barrettes mâle-femelle 12 plots
- 1 barrette mâle-femelle 3 plots
- 1 barrette mâle-femelle 2 plots
- 3 barrettes mâle-mâle 5 plots
- 1 barrette mâle-mâle 4 plots
- 1 barrette mâle-mâle 3 plots
- 3 barrettes mâle-mâle 2 plots
- 1 barrette mâle-mâle 1 plot
- 2 condensateurs céramiques 100 nF
- 2 condensateurs chimiques 10 µF
- 1 résistance 1/4 w 330 KΩ
- 1 résistance 1/4 w 1MΩ
- 1 régulateur HT7533
- 1 longueur de fil d'acier pour shunts

- 1 longueur de fil isolé pour shunt
- 1 carte Pro mini
- 1 émetteur NRF24
- 1 nappe 8 conducteurs

Placement des composants

Vue de dessus du circuit

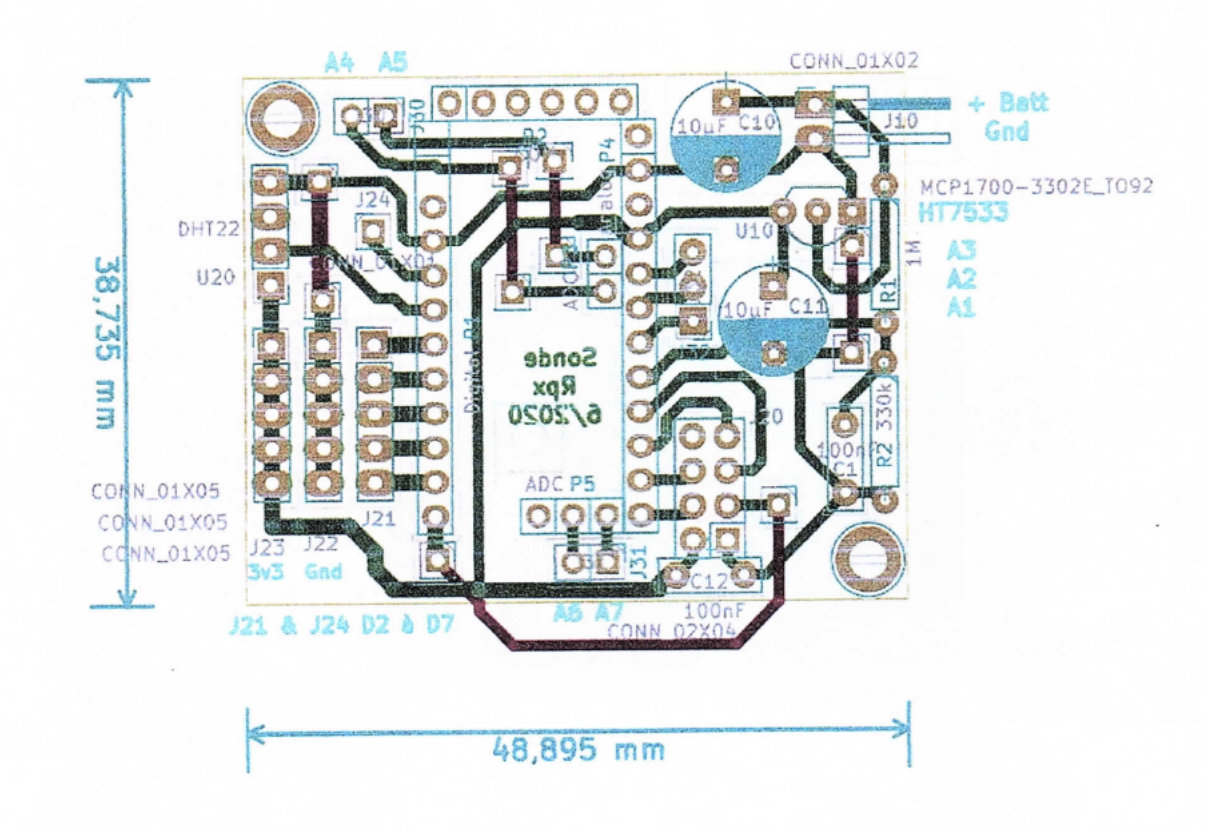


FIGURE 44.1 – Circuit imprimé vu de dessus, coté composants

Étapes

1. Souder les shunts. Voir schéma général et la photo platine sonde shunts
2. Souder les 8 conducteurs de la nappe en suivant les mêmes instructions que pour le montage de la passerelle

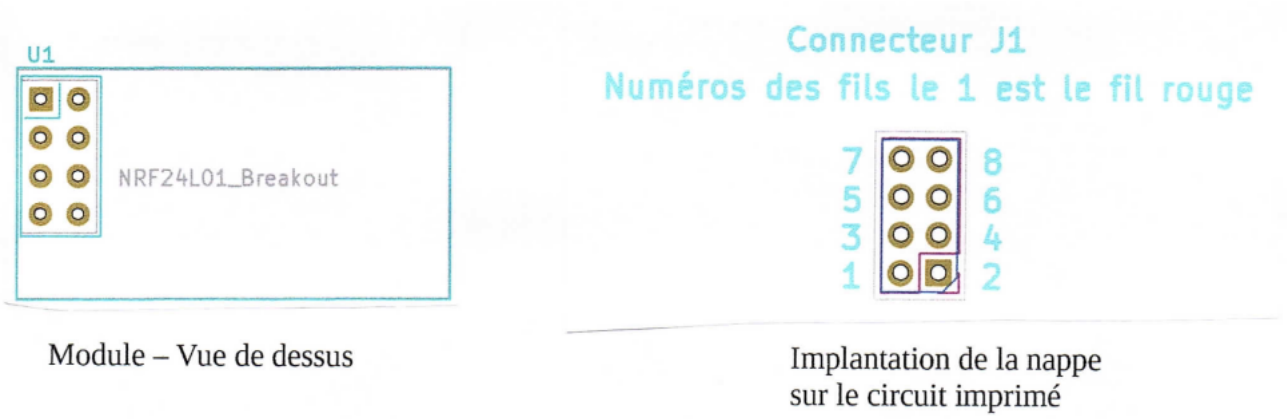


FIGURE 44.2 – Emplacement de la nappe

Puis dans l'ordre que vous souhaitez :

- Les 2 condensateurs électrolytiques sont à souder en C1 et C2 (sont polarisées, le corps du condensateur est noir avec une bande grisée, la patte de ce côté est le -),
- Les 2 condensateurs céramiques sont à souder en C3 et C4
- Le régulateur 3.3v HT7533
  A la différence de son implantation sur la passerelle MySensor, ici il ne faut pas croiser les pattes, implantez le module tel quel en respectant son positionnement sur le pcb grâce au méplat
- Les 2 condensateurs céramiques de 100 nF
- Les 2 résistances de 330 kΩ et de 1 MΩ (Elles ont le même aspect extérieur vérifier à l'ohmmètre avant la pose)
- les 2 condensateurs chimiques de 10 µF (Le moins est du côté grisé sur le corps du condensateur)
- Souder les connecteurs mâle-mâle sur la carte Pro mini, ils sont livrés dans la pochette. Attention à ne pas trop chauffer les points de soudure.
- Faire de même avec la carte Arduino Nano nécessaire à la passerelle.

Rendus

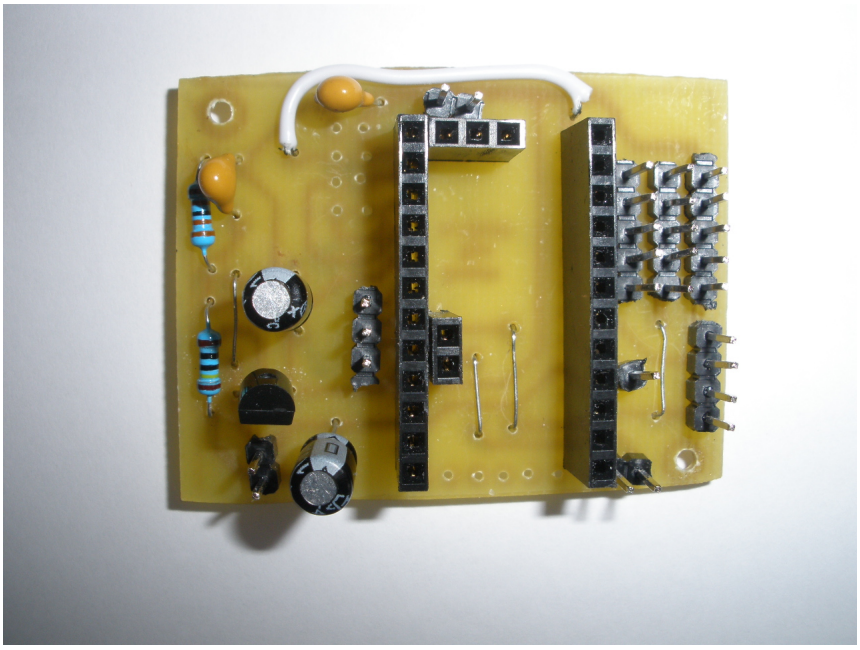


FIGURE 44.3 – Sonde vue de dessus sans la nappe

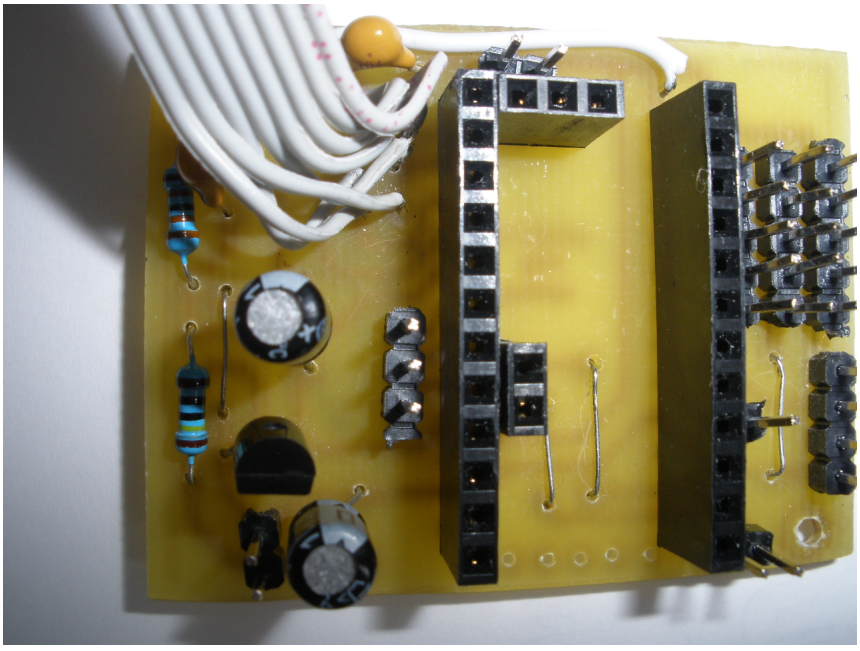


FIGURE 44.4 – Sonde vue de dessus avec la nappe

SECTION 45

VÉRIFICATION

Les court-circuit

Le multimètre

Le meilleur allié contre les courts circuit est le multimètre.
Sur les rangées de barrettes, nous allons regarder la résistance entre deux broches voisines.
Si la résistance est infinie (**Un 1 affiché sur l'écran**), il n'y a pas de court-circuit et si elle tend vers 0, il y a un risque.

Réglage

On règle le multimètre en mode **Ohmmètre** , c'est à dire avec le fil noir sur **COM** , le rouge sur Ω et le curseur réglé sur la résistance la plus élevée de l'appareil.

FIGURE 45.1 – un multimètre bien réglé

On regarde la résistance entre les broches 1 et 2 par exemple pour commencer puis ensuite entre la broche 2 et 3, etc...

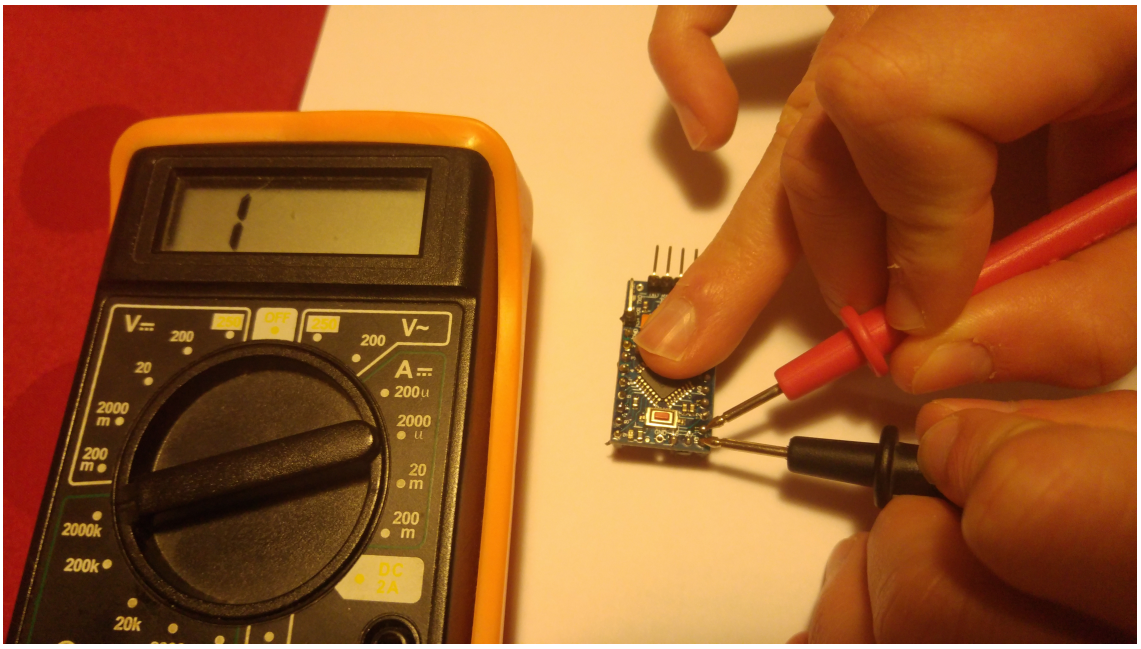


FIGURE 45.2 – Une vérification

L'alimentation

Le problème le plus grave peut survenir si un court circuit a lieu entre la broche +VCC [1] et la masse.
Il convient donc de trouver ces deux broches (VCC et GND) et de regarder la valeur de la résistance entre ces deux broches. Cette valeur doit être infinie (**1 sur l'afficheur**)

Les sondes NRF24

Les sondes NRF24 viennent s'insérer dans la nappe de fils (8 brins). Il ne faut pas se tromper de sens sous peine de détruire le module NRF24 lors de sa mise sous tension.

Pour cela, il faut que le coté avec les broches 1 et deux du NRF24 (repéré avec le carré blanc sur la broche 1) soit du même coté que le fils rouge de la nappe.

1. Alimentation positive, ici +3.3V pour la sonde et +5V pour la passerelle

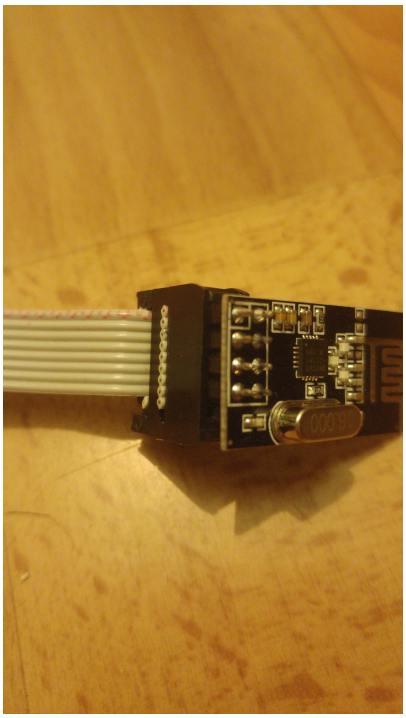


FIGURE 45.3 – Insertion du NRF24 dans son connecteur

Une fois que la connexion électrique est exacte, on peut alimenter le montage et vérifier la tension au bornes de la sonde NRF24. Si une tension inférieure à $3.2V$ ou supérieure à $3.4V$ apparaît, coupez l'alimentation et reprenez les vérifications.

Puis procédez de la même manière pour la sonde.

SECTION 46

PROGRAMMATION SONDE

Nous allons configurer le programme de la sonde (**Sonde_MySensors.ino**)pour l'envoi des données des capteurs.

Paramétrage du NRF24

Pour éviter les interférences entre les modules NRF24 dans une même salle, nous allons sélectionner un canal de communication pour chaque personne.
Un canal correspond à une fréquence précise d'émission et de réception pour le module NRF24.

Voici le tableaux des canaux attribué aux personnes :

Prénom	**CANAL_NRF24**
André P.	84
Florian M.	85
Guy D.	86
Marcel R.	87
Michel T.	88
Nicolas L.G.	89
Patrice G.	90
Patrick P.	91
Patrick Z.	92
Philippe C.	93
Pierre G.	94
Yvon	95

FIGURE 46.1 – Répartition des canaux pour les utilisateurs

Voici un extrait du code **Sonde_MySensors.ino** :

```
/*!
 * ************************************************************
 * PARAMETRES NRF24 (exemple canal 86)
 * Légende : (*) = A changer pour chaque personne
 * ************************************************************
 */
```

```
//Mode debug activé
#define MY_DEBUG

// Enable and select radio type attached
#define MY_RADIO_RF24

#ifndef MY_RF24_PA_LEVEL
#define MY_RF24_PA_LEVEL     RF24_PA_MAX
#endif

#ifndef MY_RF24_CHANNEL
#define MY_RF24_CHANNEL    86 //(*)  A CHANGER
#endif

#ifndef MY_RF24_DATARATE
#define MY_RF24_DATARATE RF24_250KBPS
#endif
```

Paramétrage du NRF24

Paramétrage de Domoticz

Comme vue dans le chapitre de présentation, chaque sonde possède un identifiant et chaque capteur rattaché à la sonde possède lui aussi un identifiant.

Pour éviter toute confusion, nous allons également attribuer un identifiant à notre sonde et à nos capteurs, ces identifiants sont définis pour chaque personne dans le tableau suivant :

Prénom	MY_NODE_ID	ID batterie	ID température	ID Humidité
André P.	10	11	12	13
Florian M.	20	21	22	23
Guy D.	30	31	32	33
Marcel R.	40	41	42	43
Michel T.	50	51	52	53
Nicolas L.G.	60	61	62	63
Patrice G.	70	71	72	73
Patrick P.	80	81	82	83
Patrick Z.	90	91	92	93
Philippe C.	100	101	102	103
Pierre G.	110	111	112	113
Yvon G.	120	121	122	123

FIGURE 46.2 – Répartition des identifiants pour les utilisateurs

```
/*!
 * ************************************************************
 * PARAMETRES DOMOTICZ (Exemple avec l'ID 30)
 * ************************************************************
 * Légende : (*) = A changer pour chaque personne
 */
#define MY_NODE_ID 30         //Noeud dans Domoticz (*)

//Voir le tableau
#define CHILD_ID_BATT 31      //(*) Identifiant Domoticz pour le niveau de
   batterie
#define CHILD_ID_TEMP 32      //(*) Identifiant Domoticz pour la température
#define CHILD_ID_HUM 33       //(*) Identifiant Domoticz pour l'humidité
```

Paramétrage de Domoticz

ici, il nous reste à définir le temps entre deux envois de données par la sonde.

```
//Temps entre deux envois de données
static const uint64_t UPDATE_INTERVAL = 10000;
```

Temps de mise à jour

Ensuite, on peut éventuellement changer le nombre de mesure au bout duquel la sonde envoie les données même si elles n'ont pas changés

```
    //Nombre au bout duquel la sonde envoie les données même si elles n'ont pas
   changés
    static const uint8_t FORCE_UPDATE_N_READS = 10;
```

Nombre de lectures forcée

Paramétrage du DHT

Il faut préciser si le capteur est un DHT11 ou DHT22 et indiquer la broche du capteur

```
    /*!
    * ************************************************************
    * DHT22/DHT11
    * ************************************************************
    */
   //Broche de données du DHT22 (ou DHT11)
   #define DHT_DATA_PIN 3

   #define DHT_TYPE DHT22 //use DHT11 or DHT22
```

```
//définir une valeur si le capteur à un offset permanent par rapport à la temp
érature réelle
#define SENSOR_TEMP_OFFSET 0

float lastTemp;   //Dernière valeur de température lue
float lastHum;    //dernière valeur d'humidité lue
```

Paramétrage du DHT

Paramétrage de la mesure de la batterie

Il suffit de renseigner les valeurs des résistances (en Ω) formant le pont diviseur de tension. Dans mon cas, les résistances ont une valeur de 330 $k\Omega$ et $1M\Omega$

Le calcul pour afficher la tension réel de la batterie est détaillé en annexe.

```
 /*!
 * ************************************************************
 * BATTERIE
 * ************************************************************
 */
//Partie pour mesure Batterie
int BATTERY_SENSE_PIN = A0;  //ne pas toucher

float oldBatteryV = 0;//Sauvegarde du niveau de batterie

const float r1_value = 1000000.0; //Valeur de la résistance la plus proche de
l'alimentation de la batterie
const float r2_value = 330000.0; //Valeur de la résistance la plus proche de
la masse
```

Paramétrage du multimètre

SECTION 47

BRANCHEMENTS DU DHT

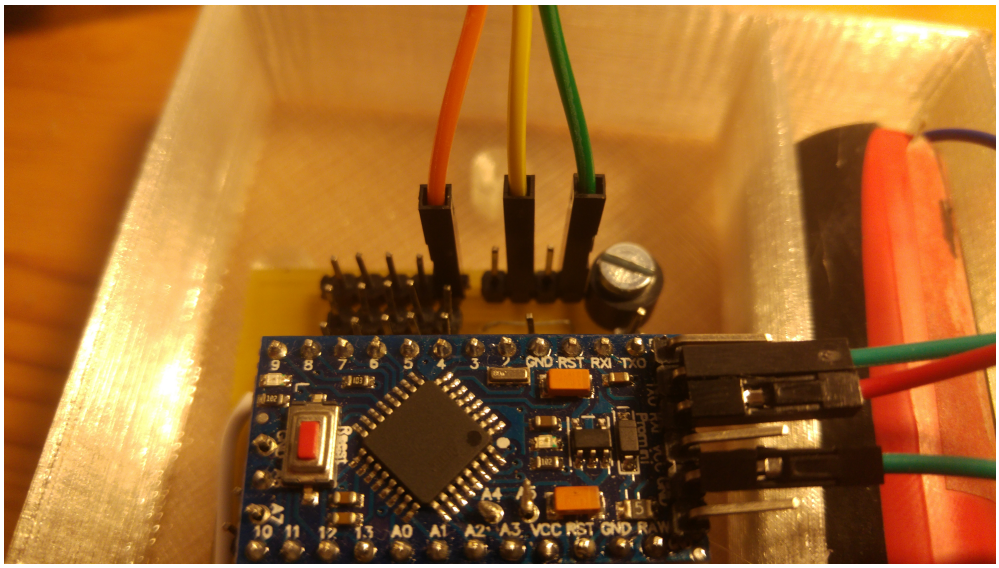


FIGURE 47.1 – Branchement du DHT sur la sonde

Voici les connexions pour brancher le DHT :

- PIN GND_DHT est représenté par le câble verts (droite)
- PIN VCC_DHT est représenté par le câble orange (gauche)
- PIN OUT_FHT est représenté par le câble jaune (D3) (centre)

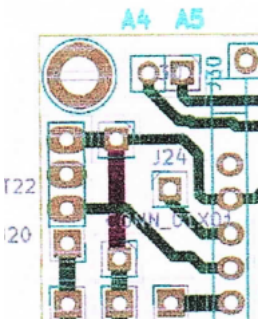


FIGURE 47.2 – Schéma du circuit imprimé

SECTION 48

PROGRAMMATION PASSERELLE

Nous allons configurer le programme de la passerelle(**Passerelle_MySensors.ino**)pour la réception des données.

Paramétrage du NRF24

Il faut mettre le même canal de communication que pour la sonde.
Un module recevant sur le canal 84 ne pourra pas recevoir des données en provenance d'un canal 83 ou 85.

Pour rappel, voici le tableau des canaux :

Prénom	**CANAL_NRF24**
André P.	84
Florian M.	85
Guy D.	86
Marcel R.	87
Michel T.	88
Nicolas L.G.	89
Patrice G.	90
Patrick P.	91
Patrick Z.	92
Philippe C.	93
Pierre G.	94
Yvon	95

FIGURE 48.1 – Répartition des canaux pour les utilisateurs

Voici un extrait du code **Passerelle_MySensors.ino** :

```
/*!
 * ************************************************************
 * PARAMETRES NRF24 (exemple canal 86)
 * Légende: (*) = A changer pour chaque personne
 * ************************************************************
 */
 #ifndef MY_RF24_CHANNEL
```

```
#define MY_RF24_CHANNEL    86          //Le canal doit être le même que celui de
  la sonde
#endif
```

Paramétrage du NRF24

Envoi des programmes

Une fois les deux programmes modifiés avec les bonnes valeurs, il ne vous reste plus qu'à envoyer le programme de la sonde sur la carte Pro-Mini et celui de la passerelle sur la carte Nano. Toutes les informations pour programmer les deux cartes sont disponibles au chapitre 2 (**Programmation**).

Lorsque les deux cartes sont programmées, occupons nous maintenant de Domoticz.

SECTION 49

CONFIGURATION DE DOMOTICZ

Une fois que la passerelle est fonctionnelle, nous allons configurer Domoticz pour que la plateforme reçoive les données en provenance de la passerelle.

Ajout de la passerelle

Tout d'abord, allez dans la section **Configuration ¿ Matériel**

FIGURE 49.1 – Emplacement du matériel

Ensuite, saisissez les informations suivantes :

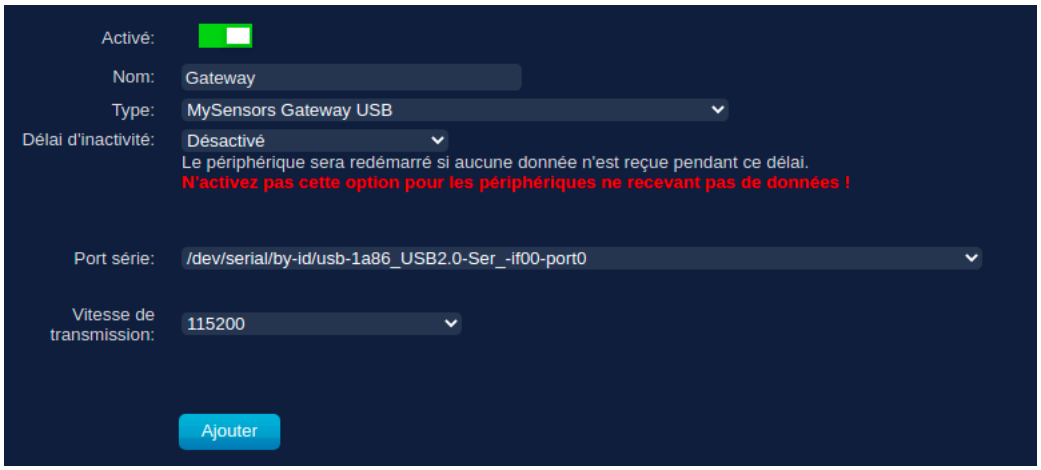


FIGURE 49.2 – Paramétrage de la passerelle

Le port série sélectionné sera celui où est raccordé la passerelle en liaison USB. Il ne faut pas prendre les noms simplifiés des ports USB (*COM_XXX*) mais le nom le plus complet.
Pour plus de simplicité, veuillez déconnectez tous les autres périphériques du Raspberry-Pi
.

Recherche des capteurs

Visualisons les données en provenance de la sonde en allant dans
Configuration ¿ Matériel
.

L'ensemble de vos dispositif apparaît. En cas de liste trop longue, saisissez **Gateway** dans la barre de recherche.

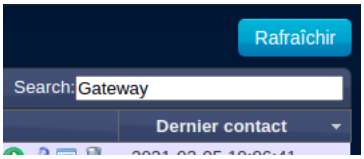


FIGURE 49.3 – Recherche de la passerelle

Remarque

Si le dispositif n'apparaît pas immédiatement, patientez quelques instants.

FIGURE 49.4 – La passerelle est détectée

Nous allons ensuite vérifier que les capteurs de la sonde envoient bien les données. Pour cela, cliquez sur **Configuration** . La page suivante apparaît :

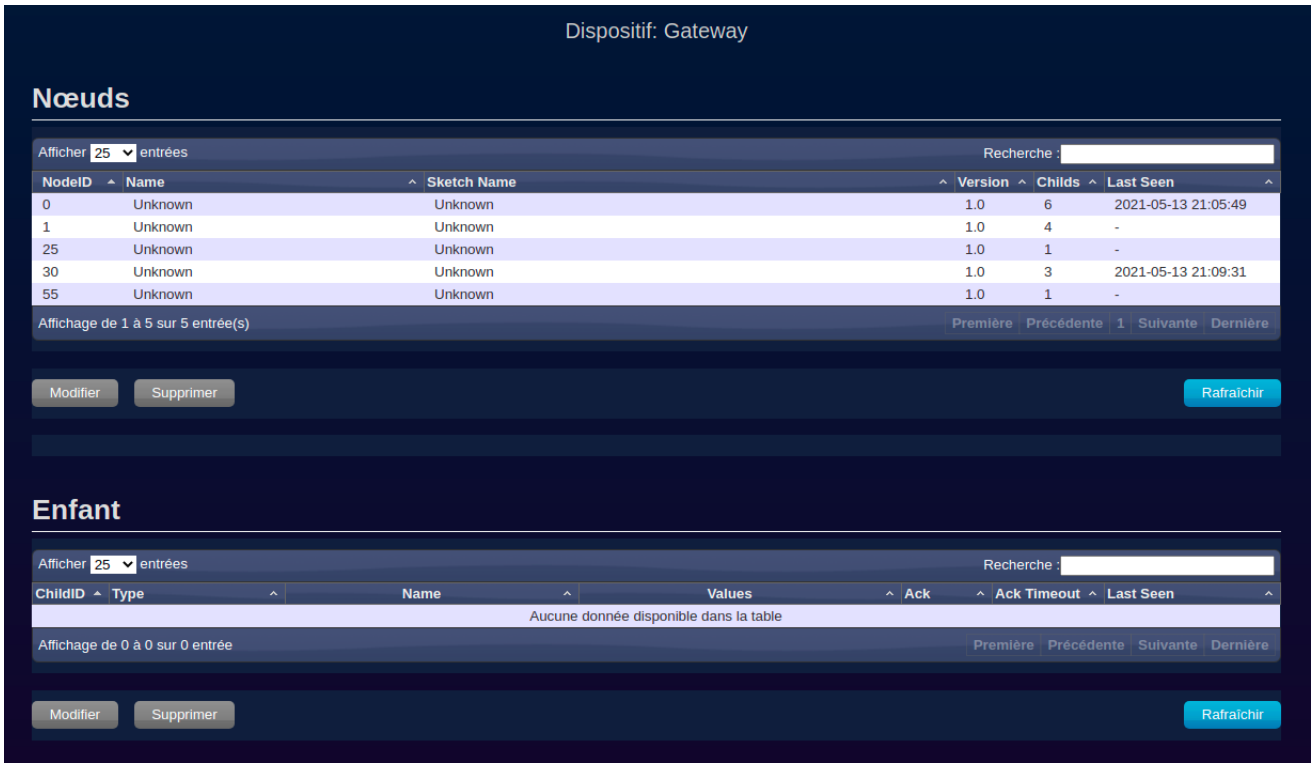


FIGURE 49.5 – Page de la Gateway

Pour visualisez les valeurs des capteurs, il faut sélectionner la passerelle avec l'ID de la sonde (ici, 30).

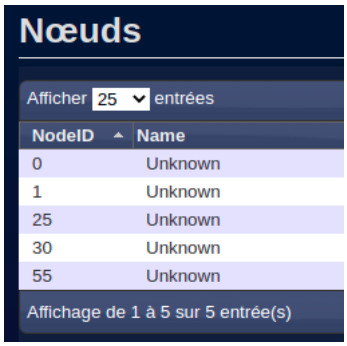


FIGURE 49.6 – Sélection de la passerelle

En cliquant dessus, on voit que la partie **Enfants** est mise à jour et contient les 3 capteurs avec les ID définis dans le programme de la sonde (31,32 et 33 en ce qui me concerne)

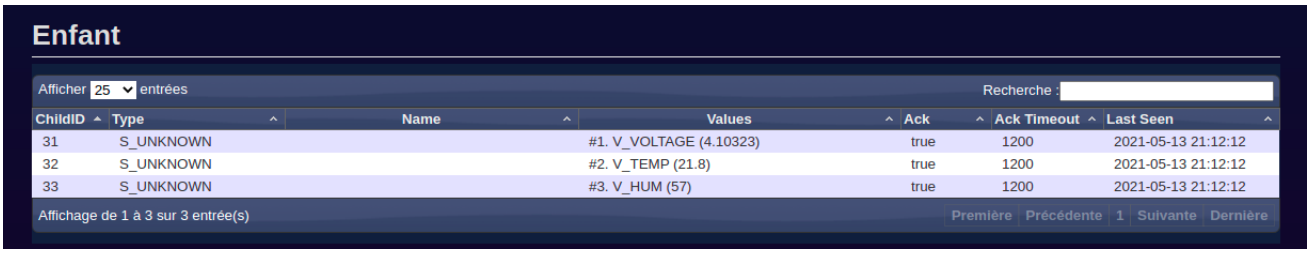


FIGURE 49.7 – Visualisation des enfants

Visualisation des données

Maintenant que nous savons que la sonde envoi les bonnes données, nous allons ajouter les capteurs dans les dispositifs. Pour cela, allez dans **Configuration ¿ Dispositifs** , les 3 capteurs de la sonde (Tension batterie, humidité et température) apparaissent dans la liste.
Si vous ne les trouvez pas, vous pouver nettoyer la page des capteurs en sélectionnant les capteurs non-utilisés et en le mettant à la poubelle.

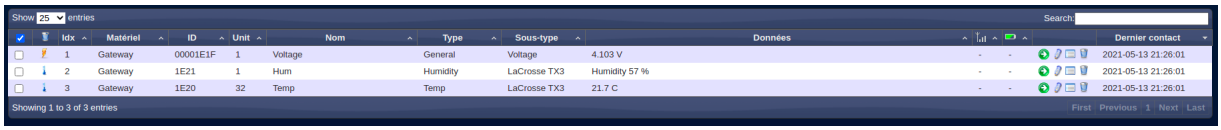


FIGURE 49.8 – Visualisation des capteurs

Les capteurs apparaissent sous les 3 noms suivants :

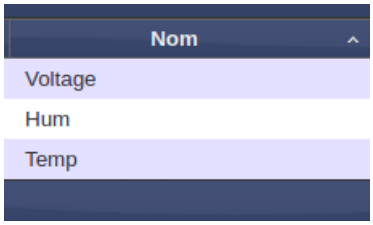


FIGURE 49.9 – Nom des capteurs

Pour ajouter un dispositif, il suffit de cliquer sur la flèche verte et de choisir le nom du dispositif.

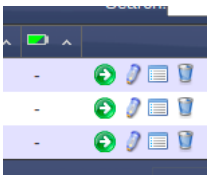


FIGURE 49.10 – Ajout des dispositifs

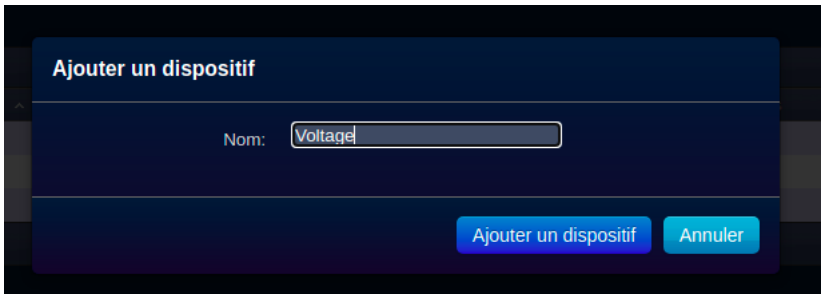


FIGURE 49.11 – Ajout des dispositifs - Sélection du nom

Il suffit de cliquer dans le menu **Mesures**

FIGURE 49.12 – Mesures

Et apparaît la tension de la batterie.

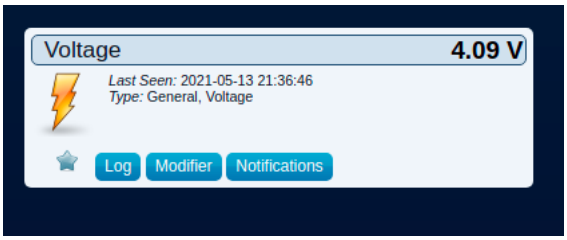


FIGURE 49.13 – tension de la batterie

On procède de même pour l'humidité et la température, les dispositifs seront mis dans l'onglet **Température** .

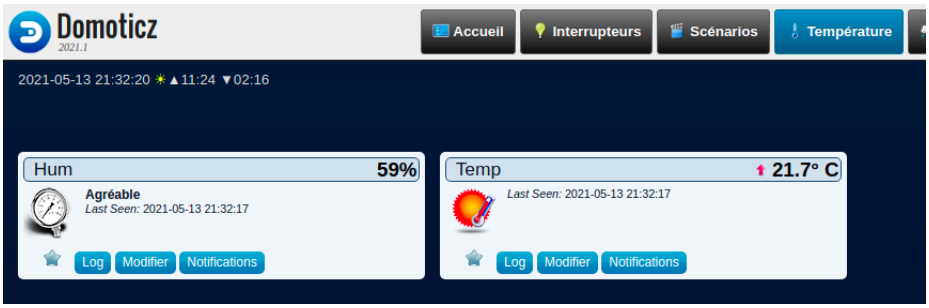


FIGURE 49.14 – Mesures de l'humidité et de la température

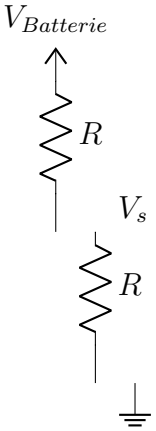
Pour visualiser les données, il suffit de cliquer sur le bouton **logs**

SECTION 50

PONT DIVISEUR

Démonstration

Soit le pont diviseur de tension formé par les deux résistances R_1 et R_2.
Le pont est alimenté avec la tension de la batterie.
La tension de sortie V_s va en entrée de la carte Arduino. (A0)



$R1R2$

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

$$V_s = V_{Batterie} \cdot \frac{R_2}{R_1 + R_2}$$

La résolution du Convertisseur Analogique Numérique de l'Arduino est de 10 bits, c'est à dire que la CAN va donner une valeur N_{CAN} comprise entre 0 et $2^{10} - 1$, c'est à dire entre 0 et 1023.

Ainsi, si le CAN affiche une valeur de 1023, cela veut dire que $V_s = 3.3V$ et si $N_{CAN} = 512$, la tension V_s vaut environ 1.65 V.

La tension V_s est obtenue par la relation suivante :

La tension de référence est généralement la tension de fonctionnement du microcontrôleur, c'est à dire ici 3.3V.

On obtient donc la formule liant $V_{Batterie}$ et la valeur N_{CAN} :

Cette formule sera utilisé pour déterminer la tension réelle de la batterie en fonction du Convertisseur Analogique Numérique.

SECTION 51

SCHÉMA PASSERELLE

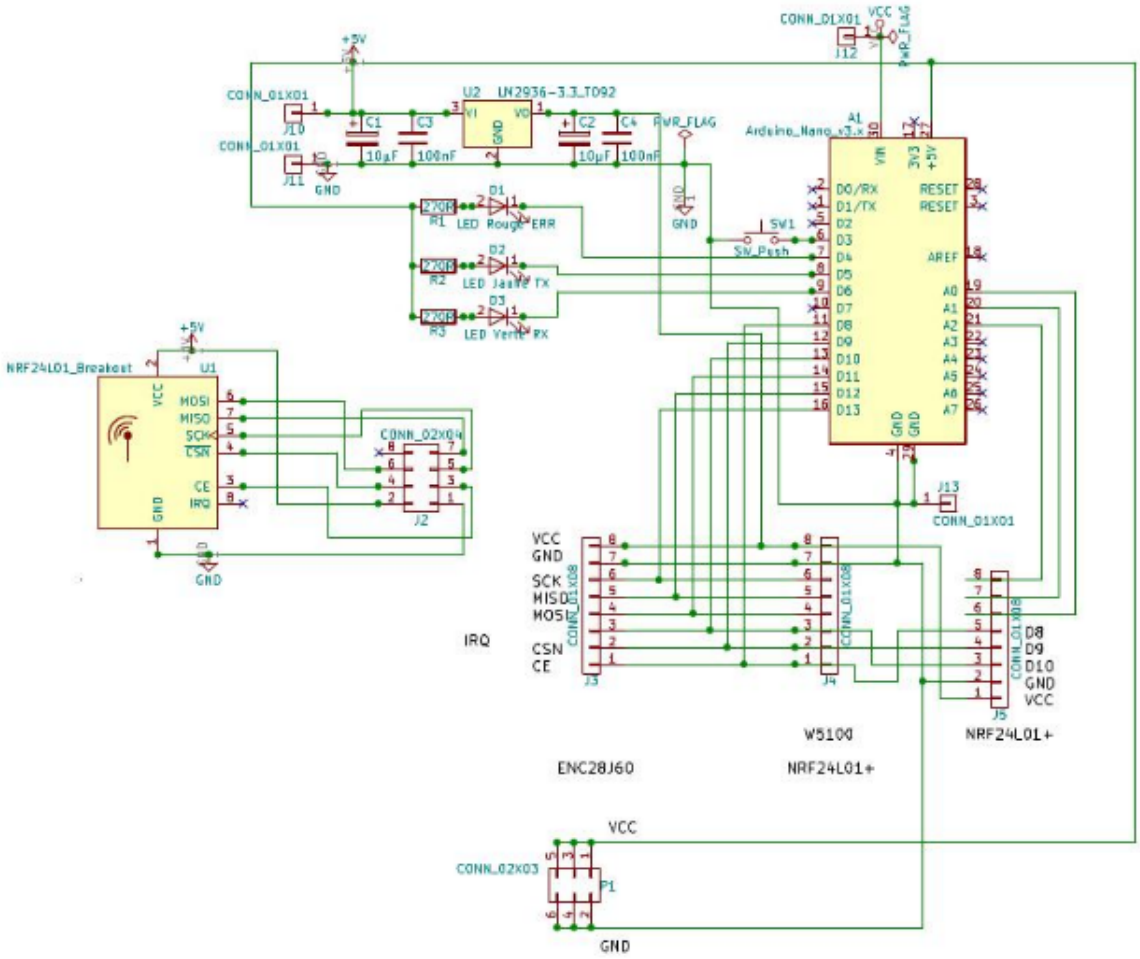


FIGURE 51.1 – Schéma de la passerelle

SECTION 52

SCHÉMA SONDE

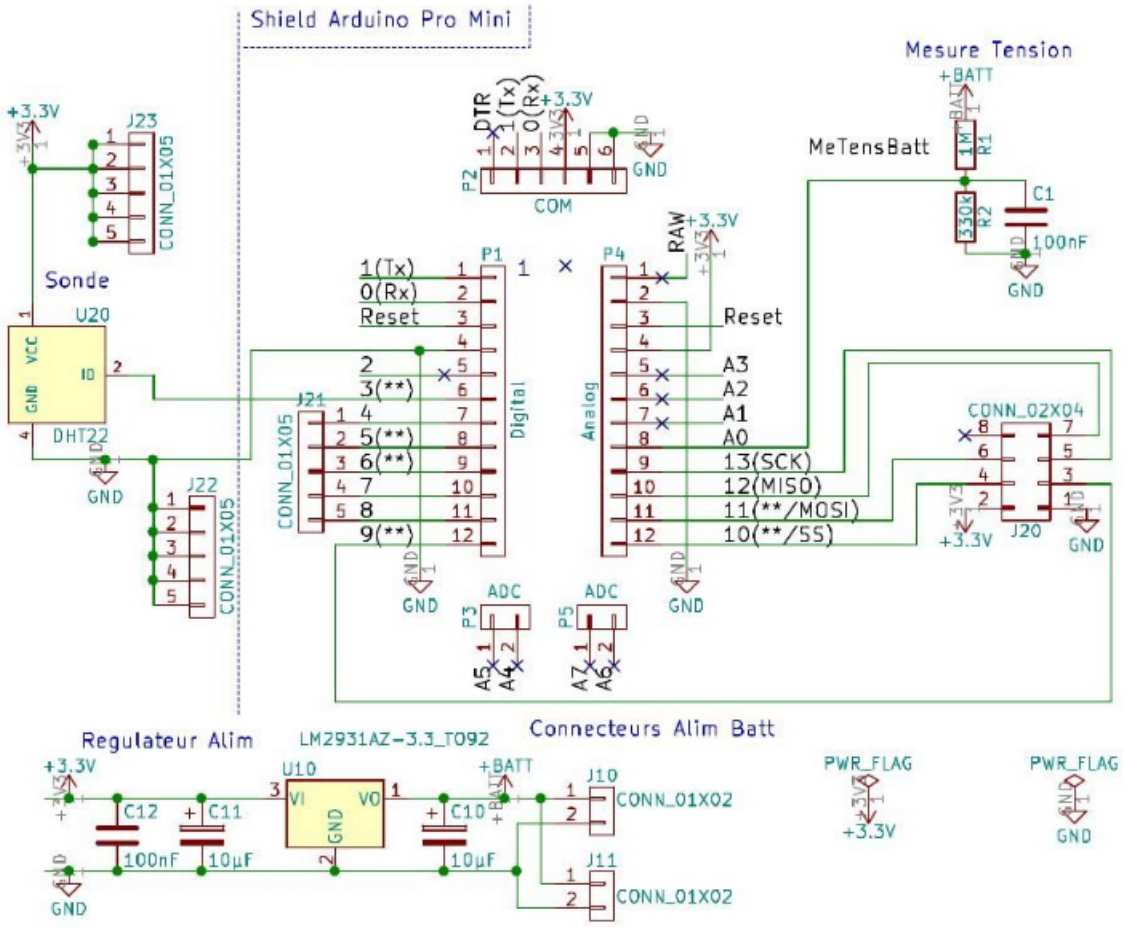


FIGURE 52.1 – Schéma de la sonde

SECTION 53

QUESTIONS

Question 5. *Pourquoi ma passerelle n'est pas détectée sur Domoticz ?*

>>> **4.** *Une erreur fréquence est de sélectionner le mauvais port lors de la configuration de la passerelle dans Domoticz.*

Question 6. *Pourquoi la passerelle allume sa LED rouge ?*

>>> **5.** *La LED rouge veut dire que des erreurs de communication sont survenues entre la passerelle et la sonde. Vérifier les branchements de la sonde.*

Douzième partie

Outils

Ensemble d'outils pour les Ateliers CREPP

ANNEXE A

INSTALLATION DE DISCORD

Connexion au serveur

Une fois le lien cliqué,vous tombez sur une interface similaire :

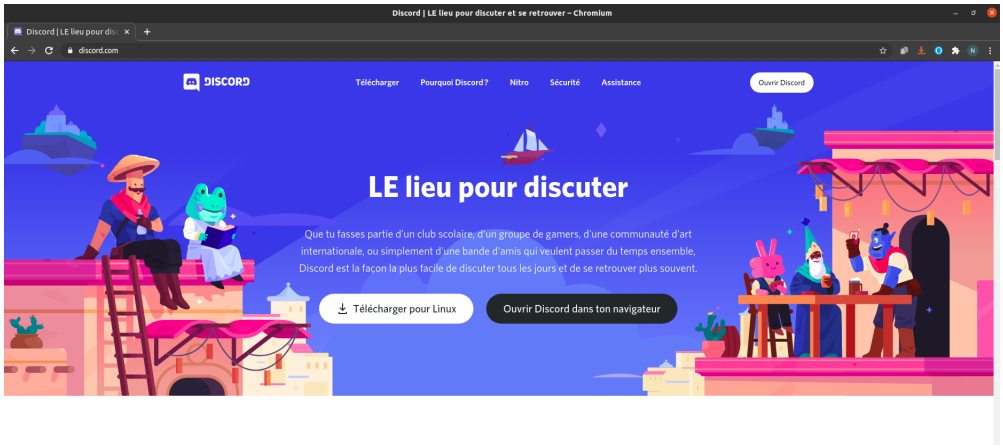


FIGURE A.1 – Accueil du site Discord

Ensuite, veuillez cliquer sur "**Ouvrir dans votre navigateur** "

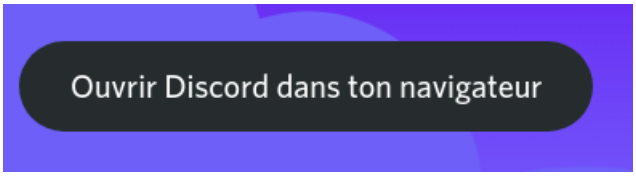


FIGURE A.2 – Lancement de Discord

Veuillez choisir un identifiant, qui sera votre nom visible par les membres de la session vocale, sans oublier de cocher la case "**J'ai lu et accepte les Conditions Générales d'Utilisation** "

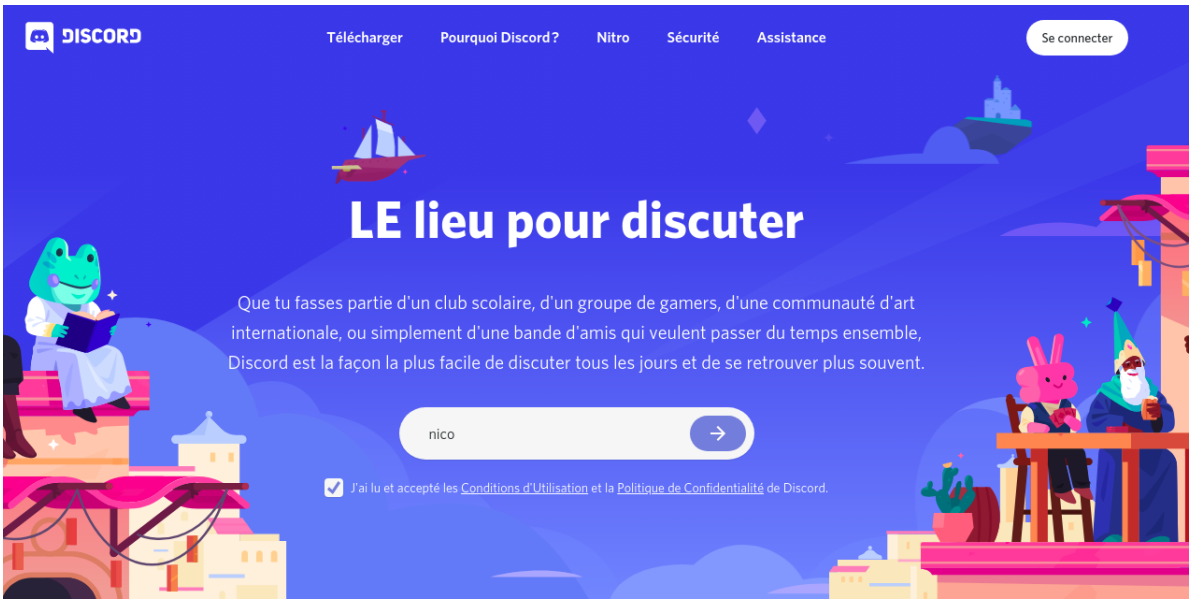


FIGURE A.3 – Choix de votre identifiant Discord

Ensuite, le site va vous demander si vous êtes un robot. Cochez la case "**Je ne suis pas un robot** "[1]

FIGURE A.4 – Vérification Captcha

Votre date de naissance va vous être demandée.

1. Parfois, une sélection d'images diverses va vous être imposée.

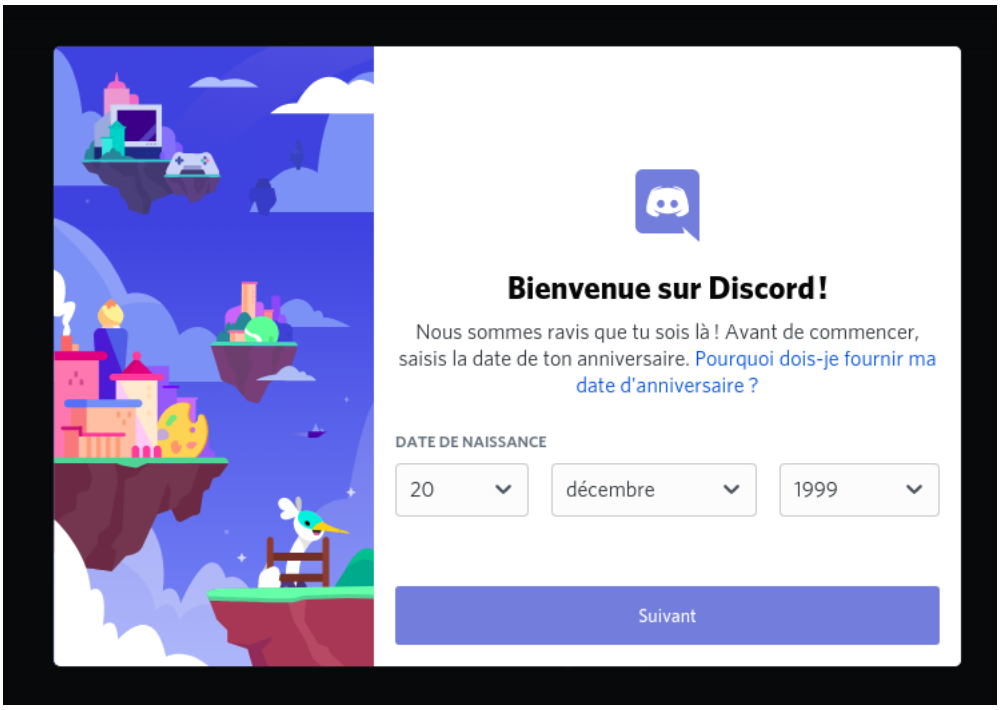


FIGURE A.5 – Date de naissance

On va vous demander le type de serveur que vous voulez créer. Dans notre cas, sur la fenêtre qui s'affiche, veuillez cliquer sur **"Rejoindre un serveur"**

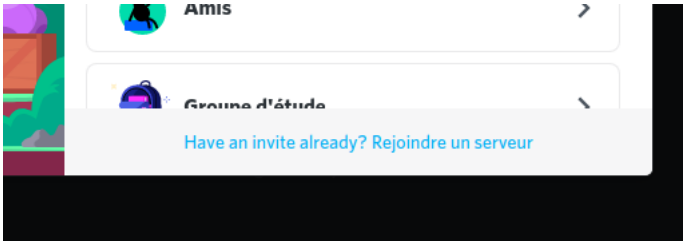


FIGURE A.6 – Rejoindre un serveur

Adresse du serveur

A ce moment, veuillez renseigner l'adresse du serveur :
https://discord.gg/Fm97K3Se

Attention, ce lien est valable 24 h et est est actif uniquement le 21 novembre 2020. Lorsque vous essayez ce tutoriel à une autre date, veuiller envoyer un SMS au 06.28.88.75.12 en demandant une nouvelle invitation sur Discord.
A partir du moment où vous recevrez le lien, vous aurez de nouveau 24 h pour vous connecter..

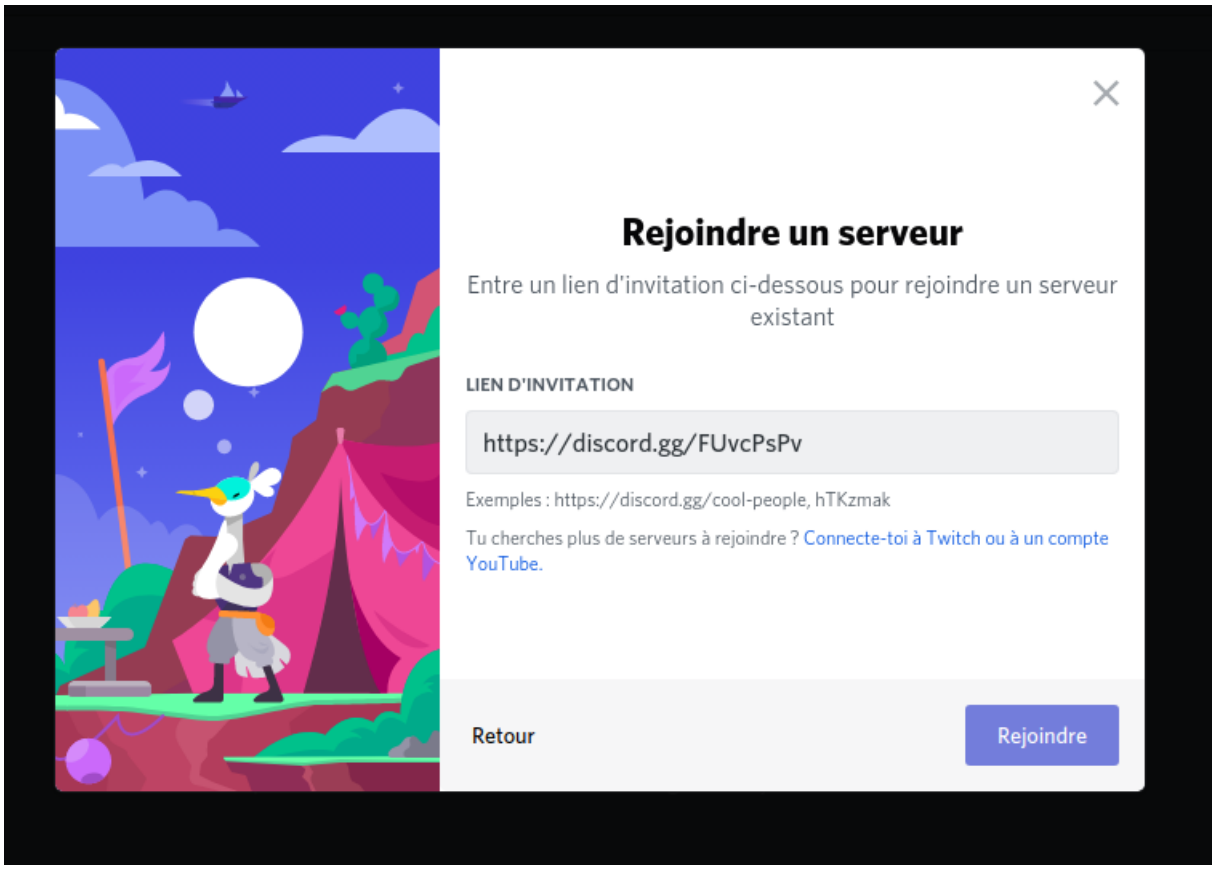


FIGURE A.7 – Saisir l'adresse du serveur, celle dans l'encadré orange

Cliquer ensuite sur **Rejoindre**

Une fenêtre avec un fond noir apparaît. Cette fenêtre demande à enregistrer votre compte.

Il faut renseigner une adresse mail et un mot de passe.

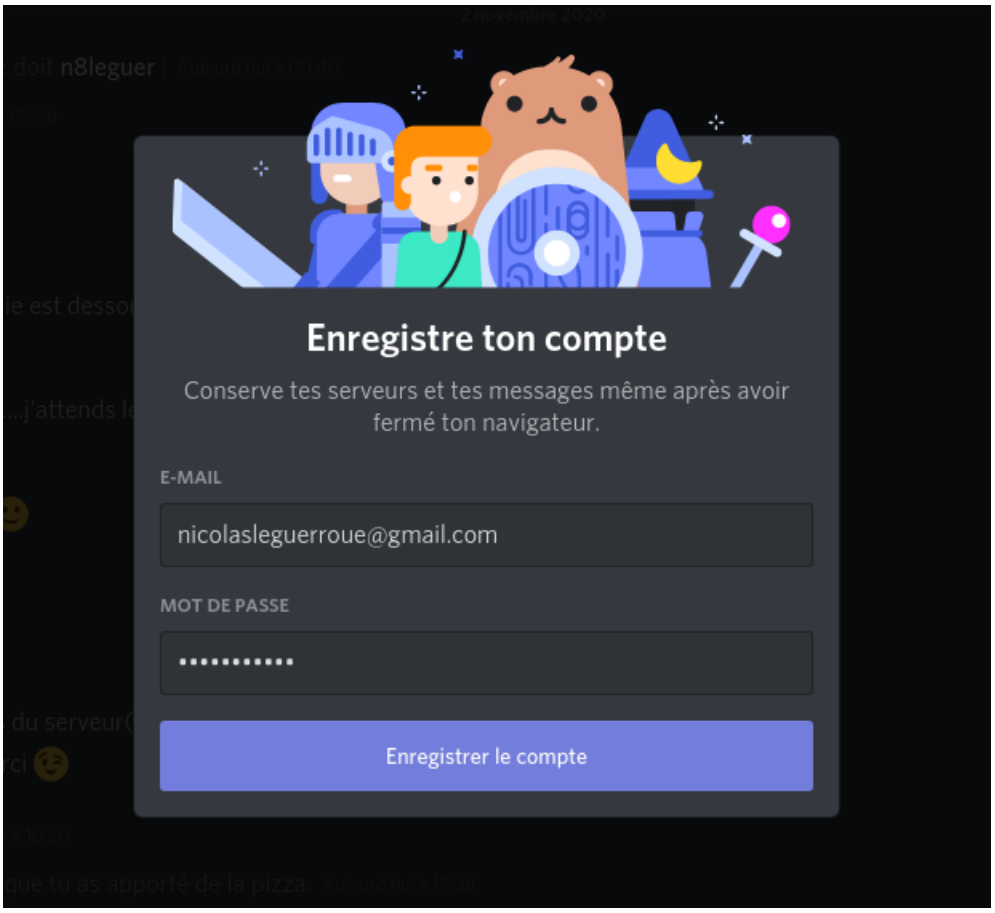


FIGURE A.8 – Saisir une adresse mail et un mot de passe

Veuillez noter précieusement l'adresse mail et le mot de passe utilisé. Cela vous permettra de vous connecter au serveur à tout moment.

Une fenetre de confirmation s'affiche. **Veuillez télécharger l'application pour utiliser le partage d'écran.**

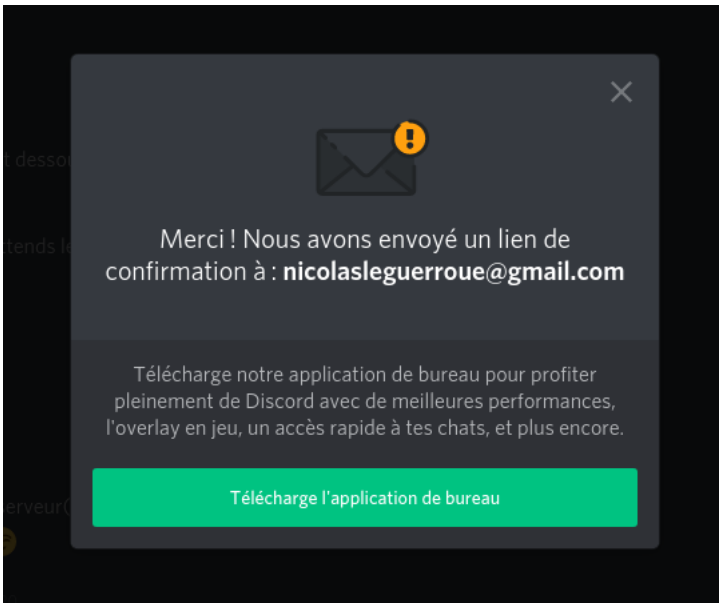


FIGURE A.9 – Confirmation de l'adresse mail

En cliquant sur le bouton **Télécharger l'application de bureau** , un fichier va se télécharger. Le format du fichier va dépendre du système d'éxploitation.

- Pour Windows, un fichier executable (.exe) va se télécharger.

  Il suffit de cliquer dessus pour lancer l'installation du logiciel.

- Pour Linux, un fichier compressé se télécharge. Il suffit de le décompresser, de se rendre dans le dossier décompressé puis dans le dossier Discord.

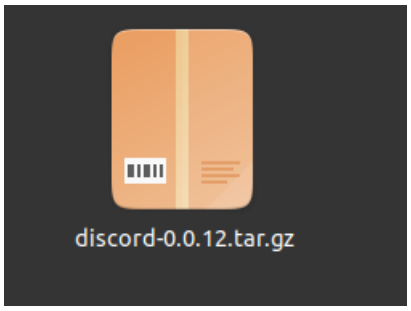


FIGURE A.10 – Dossier compressé

Vous trouvez normalement un fichier appelé Discord :

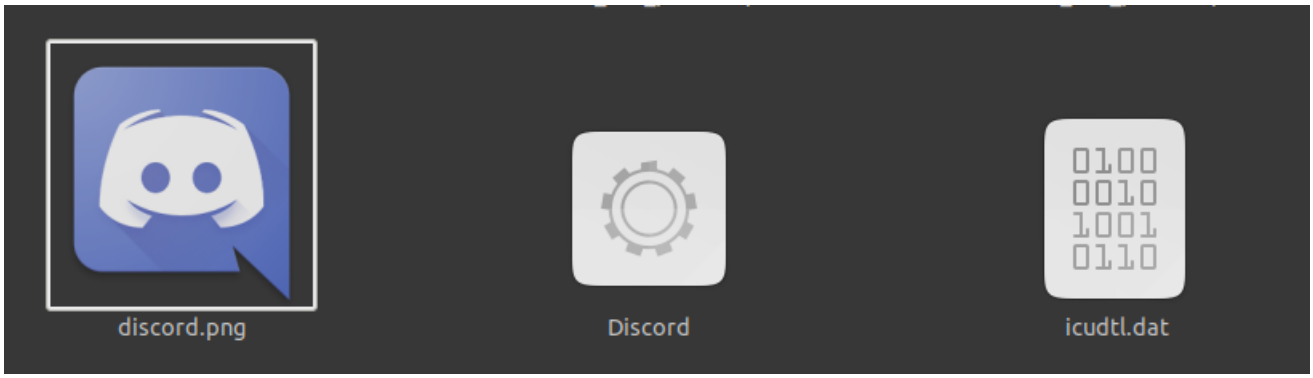


FIGURE A.11 – Emplacement de l'éxécutable

Puis click-droit sur le fichier **Discord ¿ Permissions** et cocher la case **Autoriser l'éxécution du fichier comme un programme**

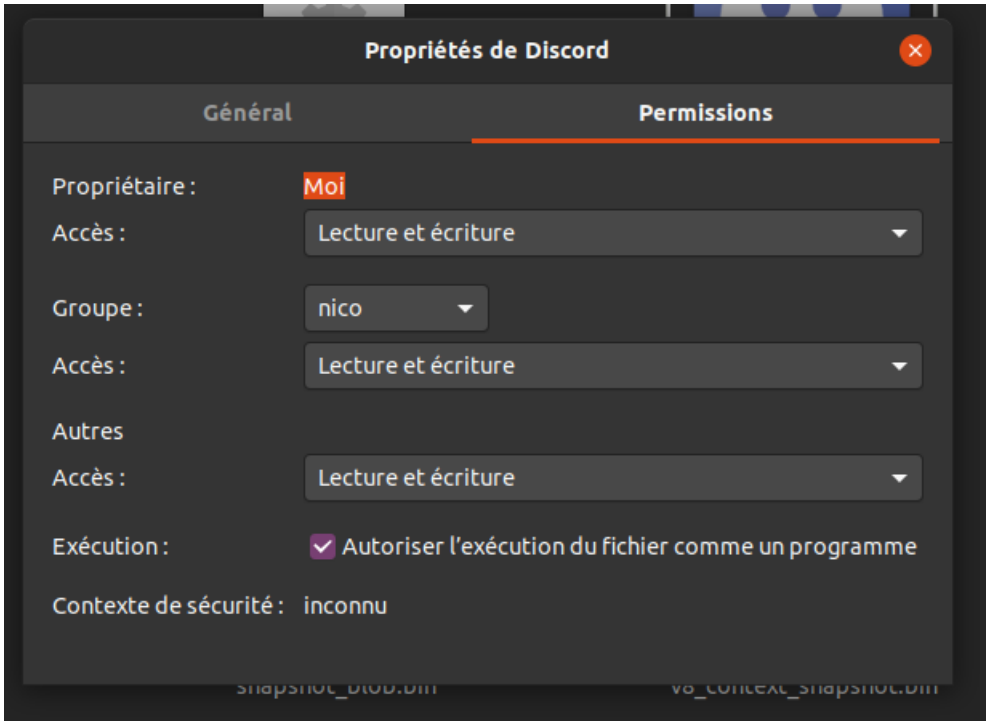


FIGURE A.12 – Droit d'éxécution du fichier

Puis click-gauche sur le fichier **Discord** pour le lancer.

Enregistrement du compte

je vous invite à consulter votre messagerie pour recevoir le mail de confirmation.

Une fois sur votre messagerie, vous recevrez un message similaire.

Veuillez cliquer sur **Vérifier l'adresse e-mail**

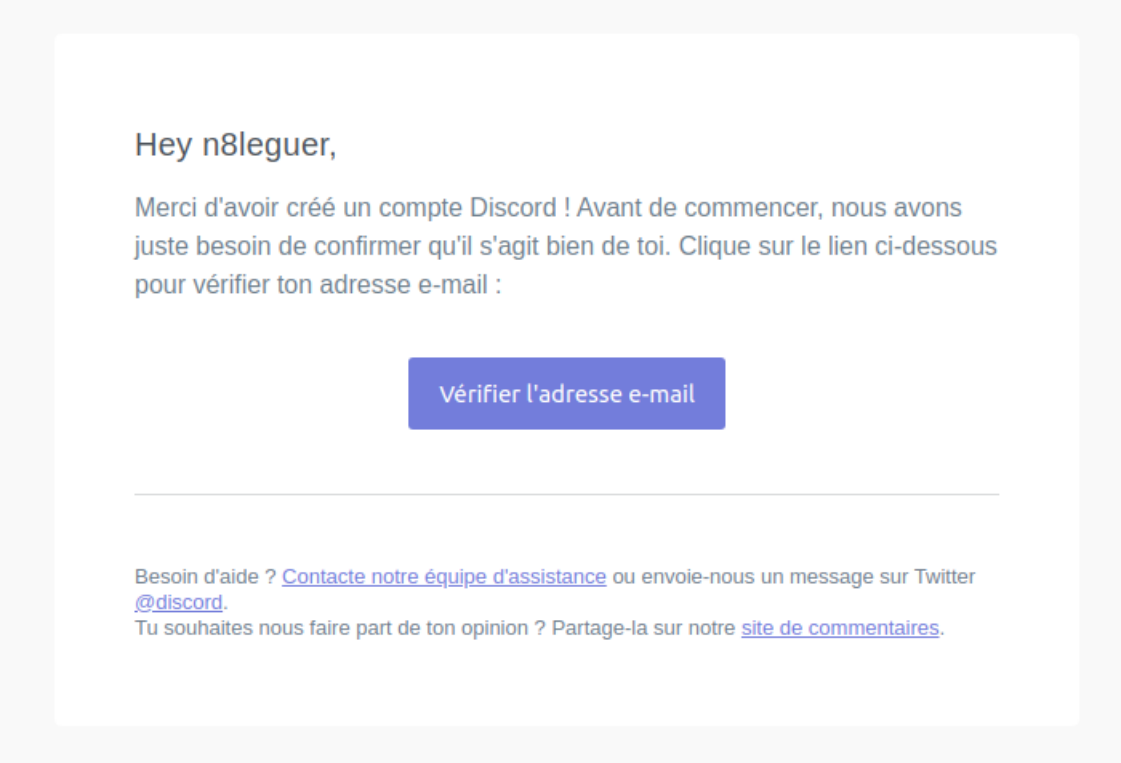


FIGURE A.13 – Vérification de l'adresse mail

Un message de confirmation devrait apparaître

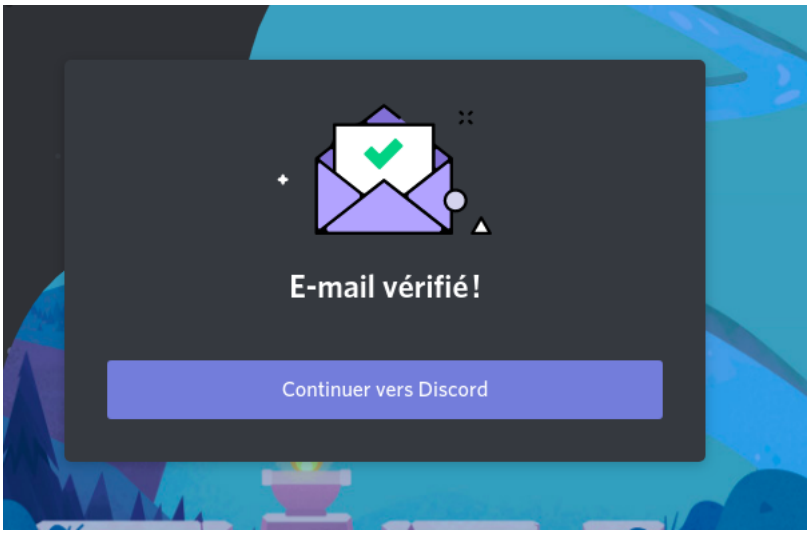


FIGURE A.14 – Confirmation de l'adresse mail

En cliquant sur **Continuer vers Discord** , la page suivante s'affiche. Il s'agit de la page d'accueil avec votre compte permanent.

Nous n'avons plus besoin du navigateur internet. Vous pouvez le fermer.

Dorénavant, pour se connecter, il suffira d'aller dans la barre de recherche de vos logiciels et de saisir "Discord" .

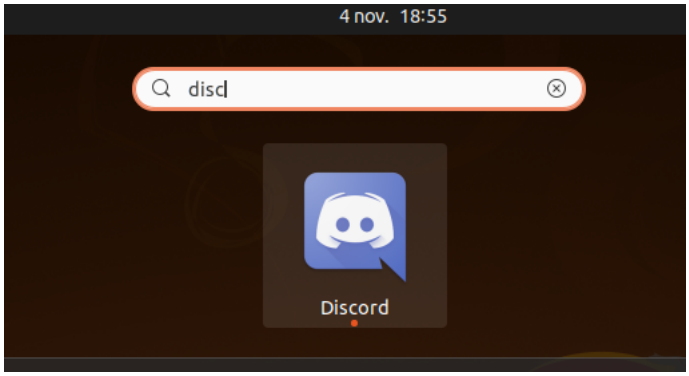


FIGURE A.15 – lancement du logiciel

La page suivante apparaît et vous permet de vous connecter avec vos identifiants.

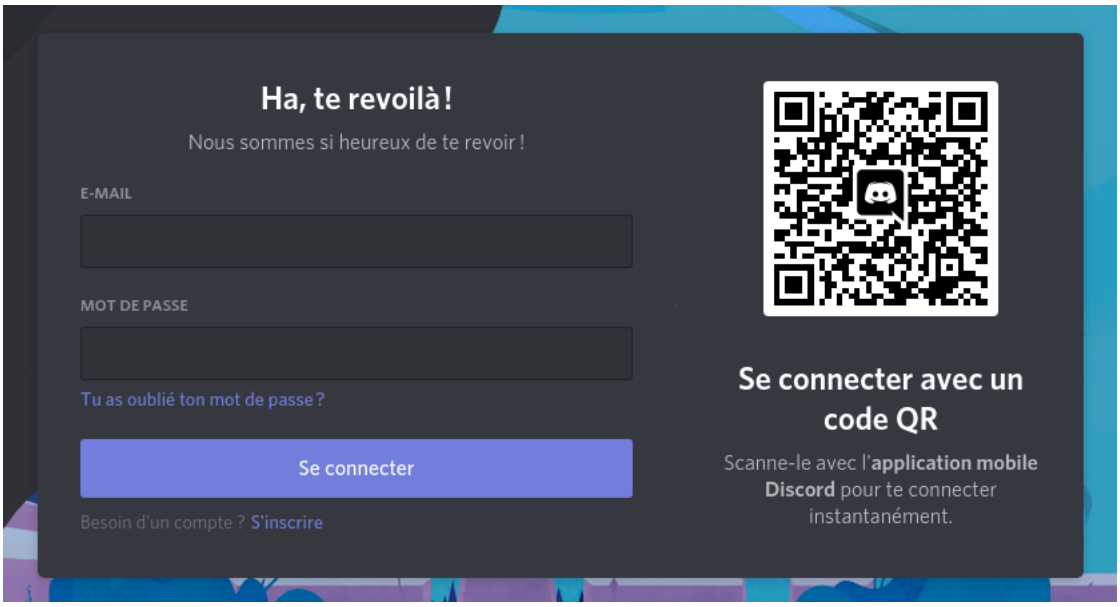


FIGURE A.16 – Connexion avec vos identifiants

Navigation et utilisation du serveur

Première vue

Une fois que vous êtes connecté sur le serveur, voici l'interface que vous devez avoir :

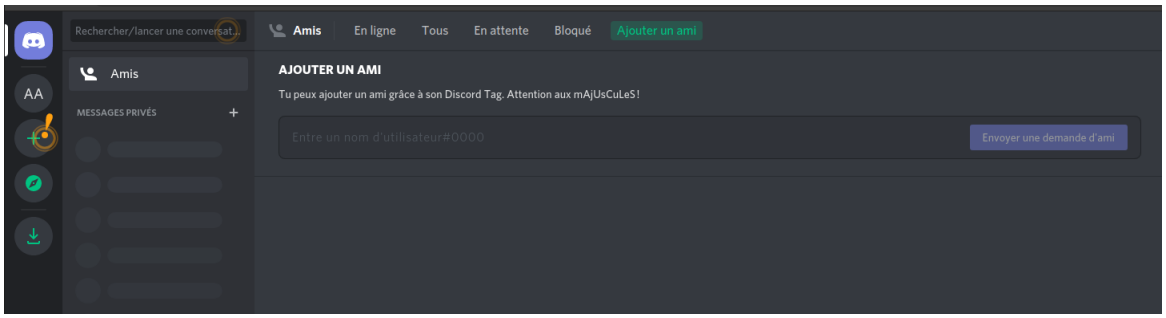


FIGURE A.17 – Accueil du serveur

Accéder à l'Atelier Arduino

Pour faire partie de l'Atelier Arduino, il faut cliquer sur le bouton **AA** dans le menu de gauche.

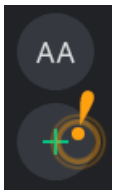


FIGURE A.18 – logo de l'Atelier Arduino non actif

Le logo "AA" en bleu vous indique que vous êtes bien dans le salon "Atelier Arduino"

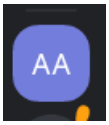


FIGURE A.19 – logo de l'Atelier Arduino actif

Présentation

Discord se décompose en trois parties.

- L'accès aux salons du serveur [menu de gauche]

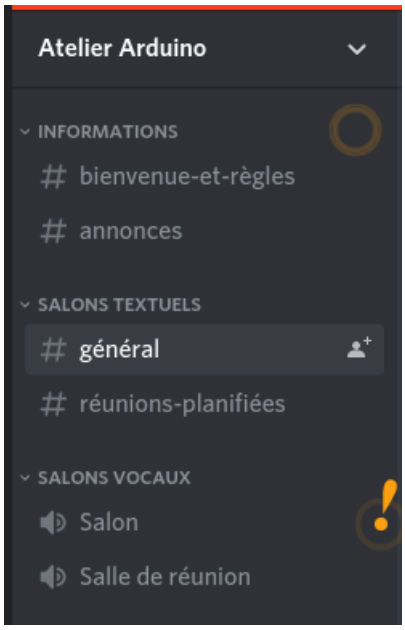


FIGURE A.20 – Les types de salons

Le serveur est composé de deux types de salons :

- Les salons textuels pour noter et faire partager des informations **écrites** .
- Les salons vocaux pour discuter de **vive voix**

- Le contenu du salon

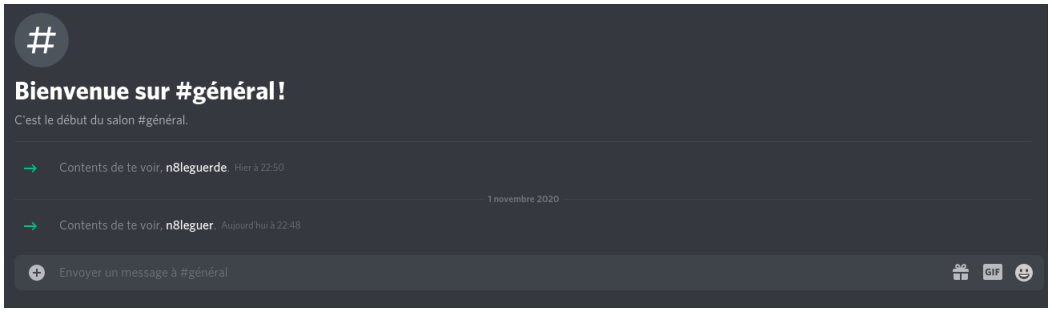


FIGURE A.21 – Le contenu du salon

- Les membres connectés

FIGURE A.22 – Les membres du salon

Entrer dans un salon

Pour pouvoir discuter avec les autres memebres, il suffit de cliquer sur le bouton **Salon** parmi les salons vocaux.

FIGURE A.23 – L'accès à un salon vocal

Une fois rentré dans ce salon, votre identifant apparait, signifant que vous pouvez discuter avec les autres membres si ces derniers sont dans le même salon.

Activer votre microphone

Par défaut, quand vous rentrez dans un salon vocal, le microphone est désactivé. Pour l'activer, il faut cliquer sur le petit microphone barré

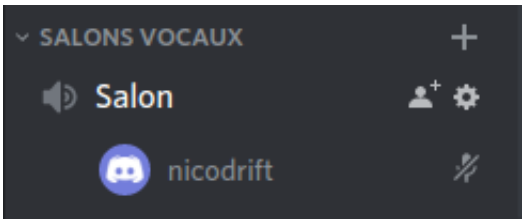


FIGURE A.24 – Activer le microphone

Partager son écran

.

La première condition pour partager son écran est de rejoindre un salon vocal.

Pour partager son écran, il faut cliquer sur le bouton **Ecran** en bas à gauche.

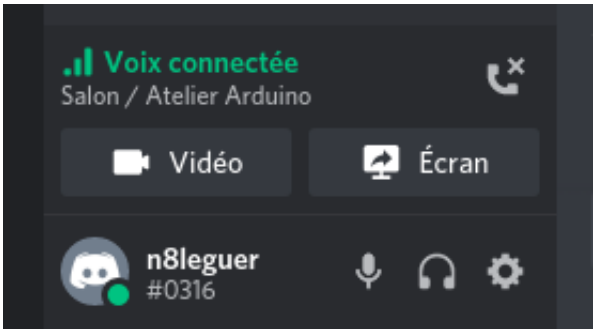


FIGURE A.25 – Le partage d'écran

A partir de cette fenetre, vous pouvez chosir ce que vous voulez partager :

- Votre écran courant
- Une application en particulier (et seulement cette application)

Vous cliquez sur l'image correspondante à votre choix puis **Partager**

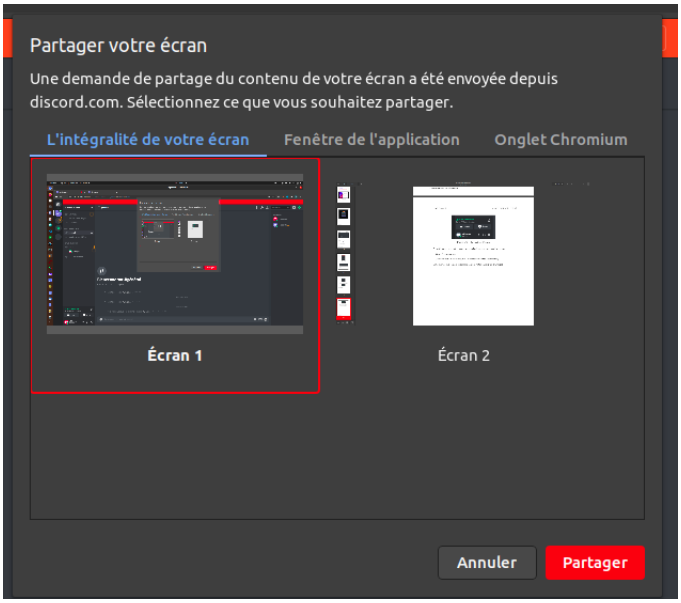


FIGURE A.26 – Le partage d'écran - validation

Accéder aux paramètres

Les paramètres de microphone, des écouteurs (ou casque) et de divers élements sont accessibles avec le bouton en forme d'engrenage en bas à gauche, à coté de votre identifant.

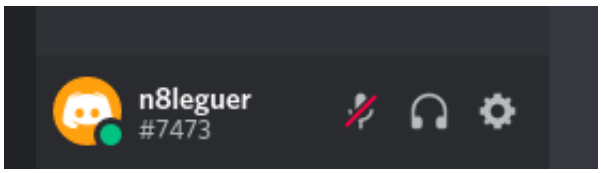


FIGURE A.27 – Accès aux paramètres

Les paramètres pour le matériel audio est disponible à la section suivante (page de paramètres)

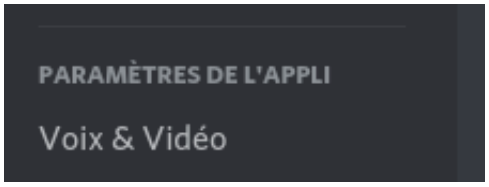


FIGURE A.28 – Accès aux paramètres sonores

Déconnexion du serveur

Pour quitter le serveur proprement, il suffit de se rendre sur la page **Paramètres** comme indiqué précédement et de se rendre en bas de la page pour cliquer sur **Déconnexion** .

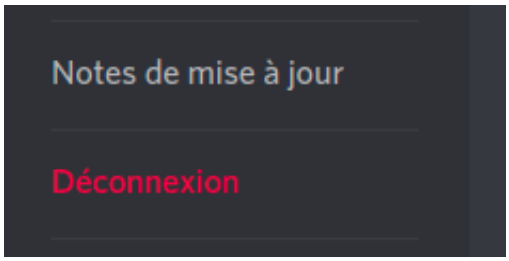


FIGURE A.29 – Déconnexion du serveur

ANNEXE B

INSTALLATION DE DOMOTICZ

Installation de Domoticz sur Linux

Veuillez ouvrir un terminal puis saisir les commandes suivantes :

```
sudo apt-get -y install cmake make gcc g++ libssl-dev git libcurl4-openssl-dev
   libusb-dev python3-dev curl zlib1g-dev zlib1g
```

Installation de domoticz

Puis lancez le script d'installation avec la commande suivante

```
  sudo curl -L https://install.domoticz.com | bash
```

Installation de domoticz

Le terminal devrait afficher un contenu similaire :

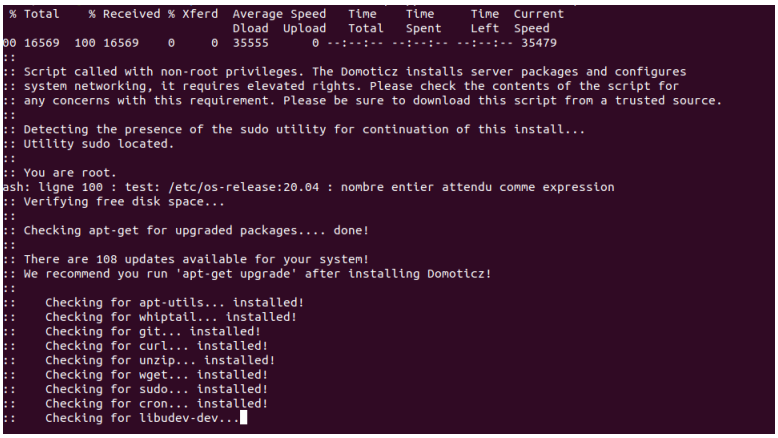


FIGURE B.1 – Vérification des bibliothèques

Ensuite, une interface utilisateur se lance dans le terminal :

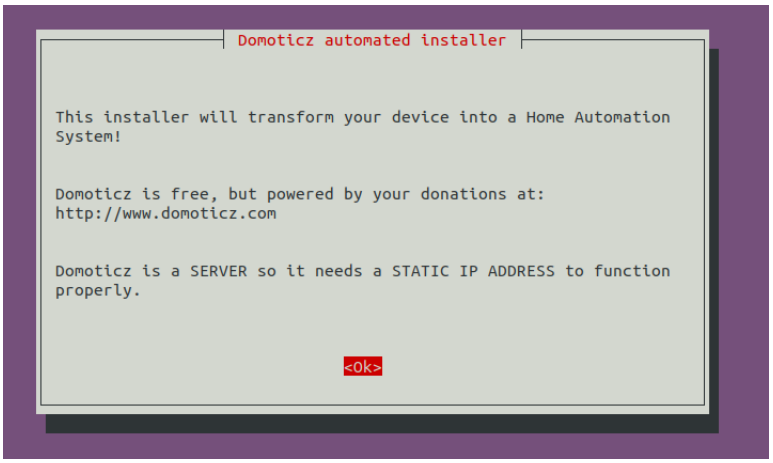


FIGURE B.2 – Présentation de Domoticz

Il faut saisir la touche KEY ENTREE pour afficher la fenêtre suivante. Une deuxième fenêtre apparaît. Veuillez sélectionner le service HTTP (par défaut) puis KEY ENTREE

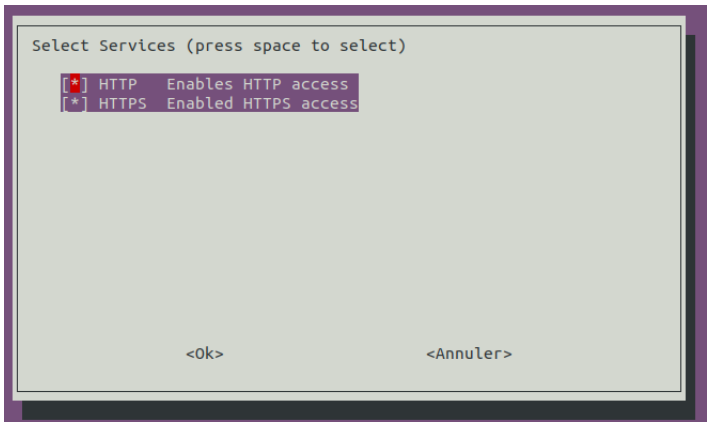


FIGURE B.3 – Choix du protocole par défaut

Nous allons ensuite choisir le port 8080 pour communiquer sur le réseau (par défaut : 8080)

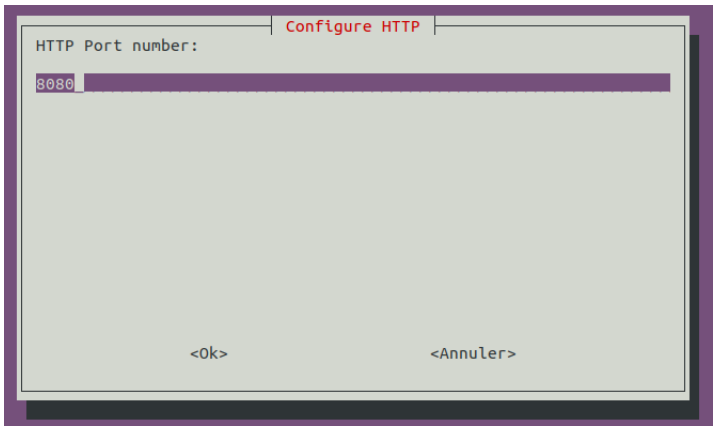


FIGURE B.4 – Choix du port

Nous utiliserons le port 443 (HTTPS) pour un protocole plus sécurisé.

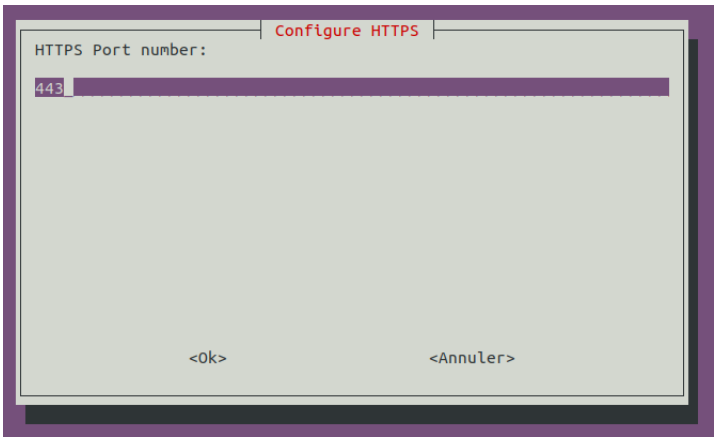


FIGURE B.5 – Protocole HTTPS

Il ne vous reste plus qu'à choisir l'emplacement du logiciel Domoticz.
Par défaut, Domoticz le place dans vos documents personnels (/home/nom_utilisateur)

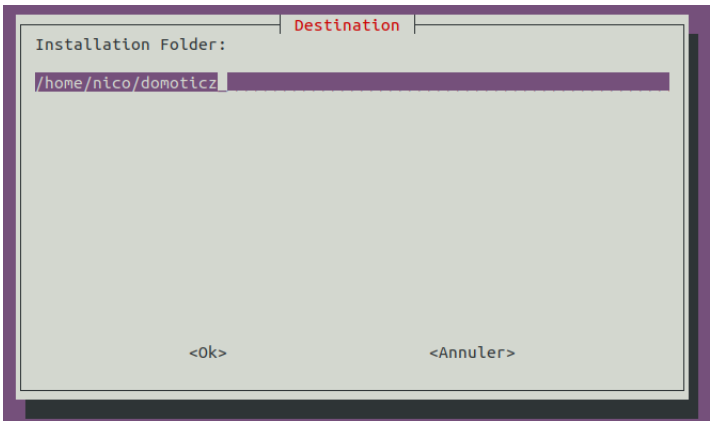


FIGURE B.6 – Emplacement des fichiers Domoticz

Il ne vous reste plus qu'à valider l'installation :

FIGURE B.7 – Validation de l'installation

Puis dans votre navigateur internet, saisir :

```
localhost:8080
```

Lancement de Domoticz