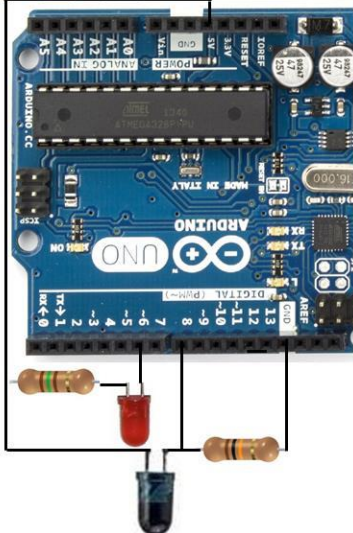
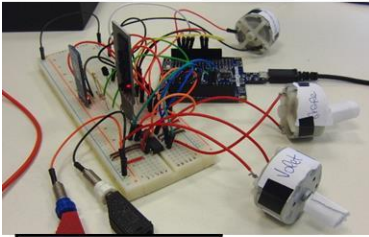


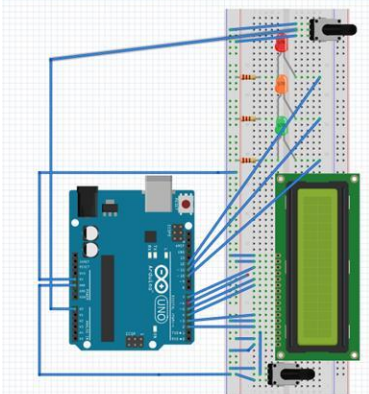
ECOLE POLYTECHNIQUE UNIVERSITAIRE DE NICE SOPHIA

*Cycle Initial Polytechnique
Deuxième année*



Travaux Dirigés d'électronique avec Arduino

Pascal MASSON



SOMMAIRE

TD No. 0 : LED et bouton poussoir	5
TD No. 1 : Entrées analogiques et LED	6
TD No. 2 : Ecran LCD	7
TD No. 3 : PWM	8
TD No. 4 : Transmission IR	10
TD No. 5 : Accéléromètre	12
TD No. 6 : Bluetooth HC-06, arduino et smartphone	14
TD No. 7 : Moteur et voiture arduino	15
TD No. 8 : Servomoteur, distance et voiture arduino	16
TD No. 9 : Ligne et voiture arduino	17

TD No. 0 : LED et bouton poussoir

Exercice 0.1 : LED qui clignote

0.1.1. Réaliser le montage du cours avec une LED (couleur de votre choix) et une résistance. Vérifier que la LED s'allume lorsque le montage n'est pas relié à une I/O mais directement à la masse.

0.1.2. Réaliser le programme qui permet de faire clignoter la LED avec une période d'1s par exemple.

0.1.3. Bien que l'I/O soit configurée comme une sortie (générateur de tension), il est possible de lire la valeur appliquée avec la fonction « `digitalRead` ». Il est aussi possible de prendre le complément d'un bit avec la « `!` » placée avant le bit. Modifier votre programme pour utiliser la commande : `digitalWrite(2, !digitalRead(led_rouge));`

Exercice 0.2 : LED et bouton poussoir

0.2.1. Ajouter la fonction « bouton poussoir » au montage de l'exercice no 0.1. La lecture se fera sur l'I/O n°7 par exemple.

0.2.2. Est-ce que l'I/O n° 7 doit être configurée comme une entrée ou comme une sortie et pourquoi ?

0.2.3. Réaliser un programme qui permet de lire la valeur binaire sur l'entrée n°7 et d'allumer la LED si la valeur est égale à « 0 » (donc quand on appuie sur le bouton poussoir)

0.2.4. En vous aidant de la question 0.1.3. modifier votre programme pour que la LED clignote avec une période de 1s quand vous appuyez sur le bouton.

TD No. 1 : Entrées analogiques et LED

Exercice 1.1 : Réalisation d'un voltmètre

A l'aide du moniteur série du logiciel Arduino, afficher la tension obtenue avec un potentiomètre de 10 k Ω .

Exercice 1.2 : Réalisation d'un voltmètre à LED

I.2.1. Réaliser un voltmètre qui donne une indication sur la valeur de la tension avec des LED comme illustré à la figure (1.2). La tension variable à mesurer sera obtenue avec un potentiomètre de 10 k Ω . Vous dessinerez l'algorithme avant d'écrire le programme.

I.2.2. A l'aide du moniteur série, relever la tension maximale mesurée par l'entrée analogique lorsque les 3 LED sont allumées. Déterminer pourquoi vous n'obtenez pas 5 V mais un peu moins.

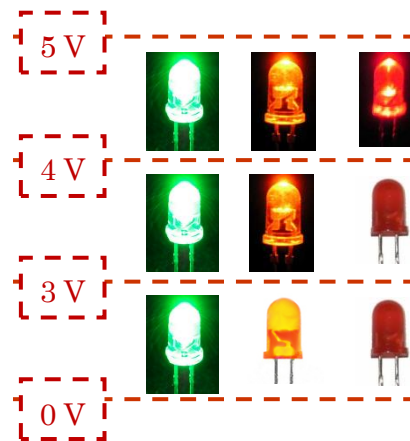


Figure 1.2. Seuils d'allumage des LED en fonction de la tension mesurée

Exercice 1.3 : Indicateur de tension à fenêtre

Dans cet exercice on cherche à régler une tension entre 2.4 V et 2.6 V comme illustré à la figure (1.2). Vous dessinerez l'algorithme avant d'écrire le programme.

La tension variable à mesurer sera obtenue avec un potentiomètre de 10 k Ω

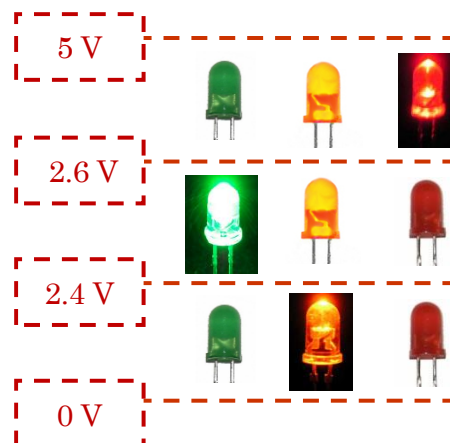


Figure 1.3. Seuils d'allumage des LED en fonction de la tension mesurée

Exercice 1.4 : Seuil de basculement d'une I/O

Une I/O de l'arduino configurée en entrée détecte les tensions 0 et 5 V, or les circuits que nous utilisons ne délivre pas exactement ces tensions. A l'aide du moniteur série et d'un potentiomètre de 10 k Ω , déterminer la tension de basculement entre l'état LOW et l'état HIGH.

TD No. 2 : Ecran LCD

Exercice 2.1 : Réalisation d'un voltmètre

Faire les branchements de l'écran LCD et afficher la valeur de la tension obtenue sur une des entrées. On utilisera un potentiomètre de 10 k Ω

Exercice 2.2 : Smiley

Réaliser un symbole « smiley » et le faire circuler sur l'écran

TD No. 3 : PWM

Exercice 3.1 : Modulation de la luminosité d'une LED

3.1.1. Réaliser un montage et un programme qui allume progressivement une LED puis l'éteint progressivement. Vous dessinerez l'algorithme avant d'écrire le programme.

3.1.2. A l'aide de l'oscilloscope, mesurer les fréquences des signaux PWM des I/O n°3 et 5

Exercice 3.2 : Modulation de la luminosité d'une LED avec bouton poussoir

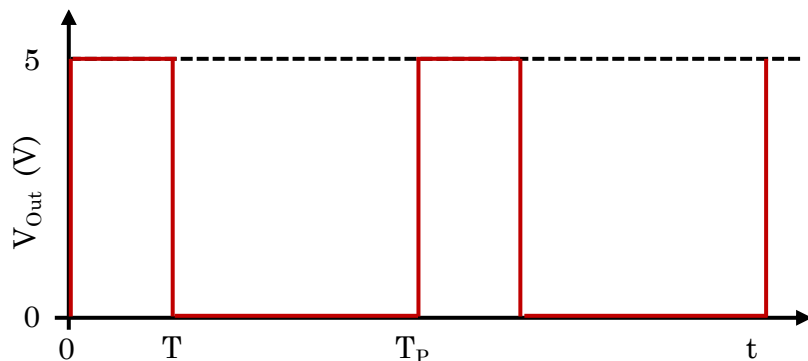
Réaliser un montage et programme qui allume progressivement une LED en fonction du temps durant lequel on appuie sur un bouton. Quand la LED est au maximum de son éclairement, elle s'éteint avant de recommencer le cycle. Vous dessinerez l'algorithme avant d'écrire le programme.

Exercice 3.3 : Obtention d'une tension analogique

Un signal rectangulaire se compose d'une valeur moyenne et d'une somme de sinusôides d'amplitudes et de fréquences différentes. On obtient ces paramètres en effectuant une décomposition en série de fourrier du signal rectangulaire. La fréquence minimale (appelée le fondamental) des sinusôides est égale à celle de la fréquence du signal rectangulaire.

3.3.1. Déterminer la valeur moyenne, A_0 , du signal rectangulaire de la figure (3.3.1)

Figure 3.1. Signal PWM en sortie de l'arduino



3.3.2. On souhaite filtrer ce signal pour ne conserver que la valeur moyenne

3.3.2.1. Quel type de filtre faut-il utiliser ?

3.3.2.2. Représenter la courbe de ce filtre sur la figure (3.2)

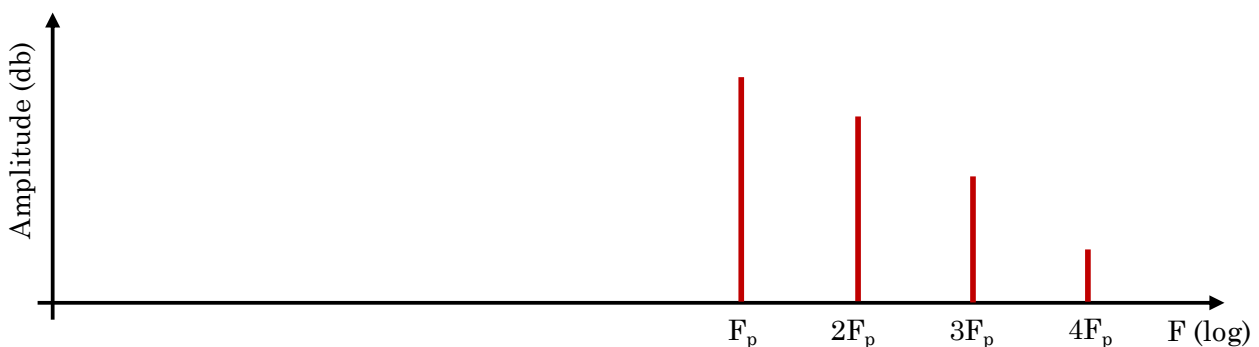


Figure 3.2. Représentation des quatre premières fréquences qui constituent un signal rectangulaire

3.3.2.3. Comment faut-il choisir la fréquence de coupure, F_c , du filtre ?

3.3.2.4. La fréquence du signal PWM de l'I/O n°11 est de 490 Hz. Vous avez à disposition un condensateur de 470 nF, que faut-il choisir comme valeur de résistance ?

3.3.3. Filtrage d'un signal PWM

La tension A_0 est obtenue à l'aide d'un potentiomètre branché sur l'entrée analogique n°0. L'arduino génère un signal PWM dont la valeur moyenne est A_0 , ce signal est filtré pour en extraire la valeur moyenne A_0' ($= A_0$). La sortie du filtre est branchée sur l'entrée analogique n°1.

On rappelle qu'une entrée analogique varie de 0 à 5 volts ce qui est traduit par l'arduino en un chiffre (entier) qui va de 0 à 1023. Une sortie PWM est ajustable avec une variable (entière) qui va de 0 à 255.

3.3.3.1. Réaliser le montage sur la figure (3.3) et le programme qui permet de réaliser cet exercice. Il faudra afficher sur votre moniteur série les tensions appliquées aux entrées analogiques n° 0 et 1.

3.3.3.2. Visualiser les signaux PWM et PWM filtrés avec l'oscilloscope en fonction des valeurs de résistances à votre disposition.



Figure 3.3. Schéma du montage

TD No. 4 : Transmission IR

Exercice 1 : Réalisation du montage et vérification

1.1. Afin de prendre en main la diode IR et le phototransistor (Ph_TR) IR, réaliser le circuit de la figure (1.1) qui permet de faire une transmission/réception IR. On prendra $R_2 = 10\text{ k}\Omega$ pour commencer. On remarquera que lorsque $I/O\ n^{\circ}2 = 0\text{ V}$, la LED IR est allumée et que le phototransistor est passant (court-circuit) donc que $I/O\ A0 = 0\text{ V}$. Dans ce cas, il faudra allumer la LED rouge avec votre programme.

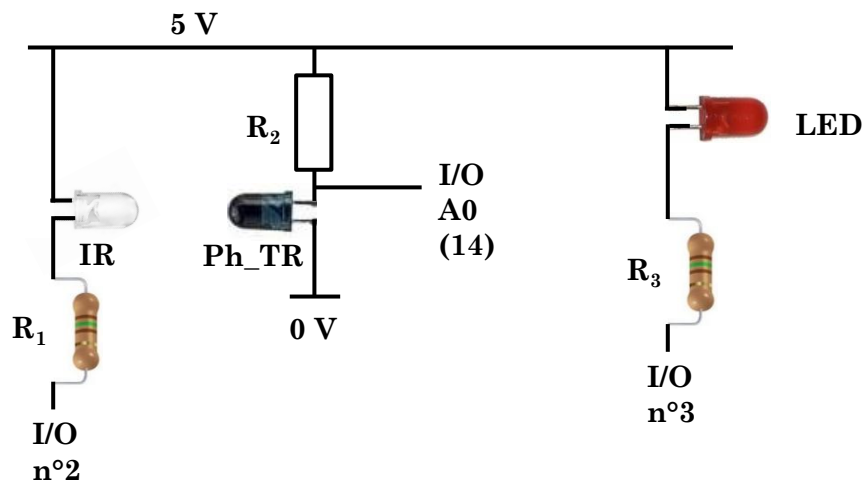


Figure 1.1. Montage d'une transmission/réception IR

1.2. Après avoir remplacé l'I/O n°2 par un fil connecté à 5 V, mesurer la tension entre le phototransistor et R_2 avec l'entrée analogique A0. Est-ce que cette tension est bien 5 V ?

1.3. Connecter ce fil à 0 V, quelle est la valeur de la tension mesurée par A0 ? En déduire le courant qui traverse R_2 .

1.4. En supposant ce courant constant quel que soit R_2 , en déduire la résistance minimale de R_2 pour obtenir $A0 = 0V$.

1.5. En fonction de ce que vous avez à disposition, remplacer la résistance de $10\text{ k}\Omega$ par une de valeur plus adéquate et vérifier que vous obtenez bien $A0 \approx 0V$. Si ce n'est pas le cas, est-ce qu'une entrée numérique considèrera cette tension comme un « 0 » logique ?

1.6. Reconnecter l'I/O n°2 et réaliser un programme qui allume/éteint la LED IR et allume/éteint la LED rouge quand le transistor reçoit/ne reçoit pas de photons IR. On rappelle que l'I/O analogique A0 est aussi l'I/O numérique 14.

En variante, vous pouvez laisser la LED IR allumée et couper (ou non) le faisceau lumineux vers le phototransistor avec votre doigt.

Exercice 2 : Manipulation d'un mot binaire et d'un caractère

« Byte permet de définir un mot binaire, par exemple Byte B = B10010101

« Char » permet de définir un caractère (codé sur 8 bytes). Par exemple Char X = a ;

bitRead(X, N) permet de lire un bit en particulier du mot binaire X. N est le n° du bit avec 0 pour le LSB

La fonction `Serial.print(X, BIN)` permet d'afficher le caractère `X` sous sa forme binaire

La fonction `Serial.available()` permet de savoir si un message est arrivé en provenance du moniteur série. Par exemple `if Serial.available() { }` correspond à : si un message a été reçu, alors faire ...

`!LOW = HIGH` et `!HIGH = LOW`

2.1. Appliquer un mot binaire sur l'IO n°2, par exemple 10010101, avec un débit suffisamment lent pour voir la diode rouge s'allumer et s'éteindre

2.2. Modifier le programme pour qu'un caractère soit envoyé par le moniteur série. Une fois, le caractère reçu, le faire transiter bit par bit entre la diode IR et le transistor et allumer/éteindre la LED rouge en conséquence.

Exercice 3 : Communication entre 2 arduino

Ce travail se fait en binôme avec 2 diodes IR et 2 phototransistors IR.

La communication série de la carte arduino s'appelle `Serial` et elle est déjà utilisée par le port USB. Pour contourner ce problème et envoyer des messages à la LED IR avec votre moniteur série (et lui envoyer des messages via le phototransistor), il faut utiliser la librairie `SoftwareSerial` qui permet d'émuler une communication série.

Avant le `SETUP` :

- `#include <SoftwareSerial.h>` permet d'utiliser la librairie
- `SoftwareSerial NOM(rxPin, txPin)` : permet de définir un jeu de fonctions de communication qui porte le nom « `NOM` ». `rxPin` correspond à l'I/O de réception et `txPin` à l'I/O de transmission

Dans le `SETUP` :

- `NOM.begin(X)` : définition de la vitesse de communication
- Le TX doit être configuré comme une sortie et le RX comme une entrée.

Dans la `LOOP` :

- `NOM.available()`, `NOM.print()` ...

Par exemple `if NOM.available() { }` correspond à ; donc si un message a été reçu, alors faire ...

En pratique :

- Le moniteur série envoie un message à la carte arduino qui le reçoit avec `Serial`. La carte arduino envoie ce message à la LED IR avec la librairie `SoftwareSerial`.
- La carte arduino reçoit un message via le phototransistor avec la librairie `SoftwareSerial`. Elle envoie ce message au moniteur série avec `Serial`.

3.1. Ecrire un programme qui permet d'envoyer/recevoir des messages avec `SoftwareSerial` et les moniteurs séries

3.2. Relier directement RX et TX entre les 2 cartes sans passer par la partie IR. Il faudra connecter le GND des deux cartes arduino. Vérifier que vous pouvez bien vous envoyer des messages.

3.2. Enlever les 2 fils et réaliser la communication avec les composants IR

TD No. 5 : Accéléromètre

Exercice 1 : Mesure d'angles avec le GY-61

La figure (1.1) illustre un montage très simple de l'accéléromètre analogique sur une carte arduino. L'alimentation est fournie par les IO n° A0 et A4 (respectivement IO n° 14 et 18) qu'il faut déclarer comme des OUTPUT :

```
pinMode(A0, OUTPUT);
```

```
digitalWrite(A0, HIGH);
```

Les mesures des accélérations suivant les axes x, y et z sont obtenues sur les entrées analogiques A1, A2 et A3. Il faudra utiliser un rapporteur pour incliner correctement l'accéléromètre.

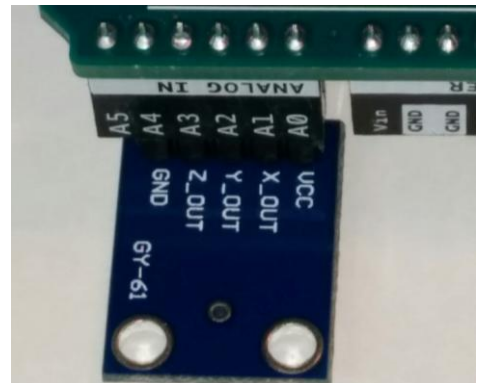


Figure 1.1. Montage du GY-61 sur la carte arduino

1.1. Réaliser le montage et écrire le programme qui permet de visualiser les 3 accélérations (valeurs entre 0 et 1023) sur le moniteur série.

1.2. Visualiser (dans votre tête) la position des 3 MEMS et observer la modification des accélérations en la corrélant à la position de l'accéléromètre et à la déformation des MEMS.

1.3. L'accéléromètre est placé à l'horizontal et on s'intéresse à l'axe y. Relever les valeurs de l'accélération (entre 0 et 1023) `accy_min` et `accy_max` qui correspondent aux angles $\pm 90^\circ$. Afficher alors l'accélération en g. Vous pouvez utiliser la fonction « constrain » pour que l'accélération ne sorte pas de la gamme $[-1, 1]$ en cas de bruit, cela évitera des problèmes de calcul d'angle par la suite. Par exemple : `y = constrain((analogRead(yPin)-322.0)/65.0,-1.0,1.0);`

1.4. Déterminer la sensibilité du capteur donnée en V/g.

1.5. Modifier votre programme pour obtenir directement l'inclinaison (angle) de l'accéléromètre en utilisant uniquement l'axe y. On rappelle que la fonction « asin » (i.e arc sin) donne un angle en radian, il faut le convertir avec $\pi = 3.14159$ (qui correspond à 180°) pour l'obtenir en degrés.

1.6. L'accéléromètre est placé à l'horizontal et on s'intéresse à l'axe z. Relever les valeurs de l'accélération `accZ_min` et `accZ_max` qui correspondent aux angles $\pm 90^\circ$. Afficher alors l'accélération en g.

1.7. Modifier votre programme pour obtenir directement l'inclinaison en $^\circ$ de l'accéléromètre en utilisant les axes y et z. Vérifier le bon fonctionnement en utilisant un rapporteur.

1.8. Avec le traceur série, observer ensemble les 2 angles obtenus avec les questions (1.4) et (1.6). Dans quelle gamme, ces 2 angles sont identiques ?

1.9. A présent vous pouvez réduire le bruit de mesure en utilisant un filtre passe bas numérique disponible ici : playground.arduino.cc/Code/Filters . Il faut visualiser une mesure (par exemple suivant y) avec le traceur série avant et après filtrage en modifiant la fréquence de coupure. Vous pouvez commencer avec $F_c = 5$ Hz. Remarque, il est aussi possible d'ajouter un condensateur entre l'entrée analogique et la masse. La fréquence de coupure est calculée avec la résistance de sortie de l'accéléromètre qui a pour valeur 32 k Ω .

Exercice 2 : Détection de chocs avec le GY-61

2.1. Sismographe

Le circuit est posé sur un cahier à l'horizontal.

2.1.1. Visualiser les 3 accélérations tout en simulant un tremblement de terre caractérisé par 2 ondes (définition Wikipédia) :

- les ondes P ou ondes primaires appelées aussi ondes de compression ou ondes longitudinales (ondes P car ondes de Pression). Le déplacement du sol qui accompagne leur passage se fait par des dilatations et des compressions successives. Ces déplacements du sol sont parallèles à la direction de propagation de l'onde
- les ondes S ou ondes secondaires appelées aussi ondes de cisaillement (shear waves d'où ondes S) ou ondes transversales. À leur passage, les mouvements du sol s'effectuent perpendiculairement au sens de propagation de l'onde.

2.1.2. Réaliser un programme qui envoie la mesure de l'accéléromètre vers excel lorsqu'un séisme est détecté. Pour cela vous vous aiderez du site :

electroniqueamateur.blogspot.com/2014/10/transmettre-les-donnees-darduino-vers.html

2.2. Pedomètre

2.2.1. Avec l'aide de votre binôme (qui devra porter le PC portable), regarder s'il est possible de détecter vos pas.

2.2.2. Réalisez alors un programme qui permet de compter les pas

TD No. 6 : Bluetooth HC-06, arduino et smartphone

Avant de démarrer ce TD, il faut installer l'application « Bluetooth Electronics » sur votre smartphone. Cette appli fonctionne sous environnement Androïde. On se réfèrera aussi au site : www.keuwl.com/apps/bluetoothelectronics/

Exercice 1 : Appariement du module bluetooth et du smartphone

En suivant la procédure expliquée dans le cours, donner un nom à votre module bluetooth, un mot de passe et connecter votre téléphone portable au module HC-06. Il ne faudra pas modifier la vitesse de transmission entre le module HC-06 et votre carte arduino qui devra être de 9600.

Exercice 2 : Allumer/éteindre une LED

Connecter une LED (avec résistance !) sur une des sorties PWM de votre carte arduino

2.1. Sur votre téléphone, sélectionner une page vide de l'application et ajouter un bouton. Choisir la lettre à envoyer quand on appuie sur le bouton et quand on le relâche. Ecrire le programme qui permet d'allumer la LED en conséquence

2.2. Sur votre téléphone, sélectionner une page vide de l'application et ajouter un « Slider ». Choisir la lettre à envoyer ainsi que le chiffre compris entre 0 et 255 pour le PWM. Ecrire le programme qui permet d'allumer plus ou moins la LED en conséquence.

2.3. Modifier la luminosité de la LED en utilisant cette fois l'accéléromètre de votre téléphone

Exercice 3 : Envoyer des données au téléphone

Connecter un potentiomètre sur une des entrées analogiques de l'arduino

3.1. Sur votre téléphone, sélectionner une jauge et régler ses paramètres pour correspondre à la variation de l'entrée analogique (chiffre entre 0 et 1023). Modifier alors la jauge en jouant sur le potentiomètre

3.2. Ajouter un « buzzer indicator » qui sonnera quand la tension analogique en entrée de l'arduino dépasse 4 V.

TD No. 7 : Moteur et voiture arduino

Ce TD se base sur la voiture arduino de la figure (0). Elle comprend, un bloc piles $4 \times 1.5\text{ V}$, une carte arduino UNO qui nécessite l'installation du driver CH340 (www.wch.cn/products/CH340.html), un driver de moteur L298 et 2 moteurs avec roues ($3 - 6\text{ V}$, 100 RPM, 0.44 N.m)

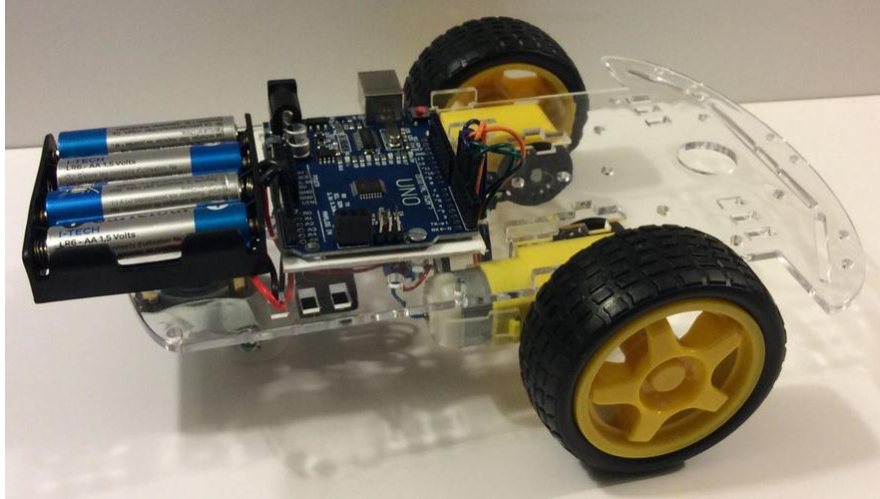


Figure 0. Voiture arduino

Exercice 1 : Mise en œuvre des moteurs

- 1.1. Connecter la carte arduino avec le driver de moteur L298
- 1.2. Ecrire un programme qui permet de mettre le moteur gauche au maximum de puissance 1s alternativement en marche avant et arrière. Testez la voiture sans la poser au sol
- 1.3. Ecrire un programme qui permet de mettre les deux moteurs au maximum de puissance 1s alternativement en marches avant et arrière. Testez la voiture sans la poser au sol. Que constatez-vous et comment est-ce que vous l'interprétez ?
- 1.4. Placer un condensateur de 2.2 mF entre la masse et le 5 V. Reprenez la question (1.3)
- 1.5. Reprenez la question (1.3) mais en posant la voiture au sol. Que constatez-vous et comment est-ce que vous l'interprétez ?
- 1.6. Ecrire un programme qui permet de mettre les deux moteurs au maximum de puissance 1s alternativement en marche avant et arrière mais cette fois en montant et diminuant graduellement en puissance. Que constatez-vous ? Si le comportement est en partie incorrect, comment pourrait-on y remédier.

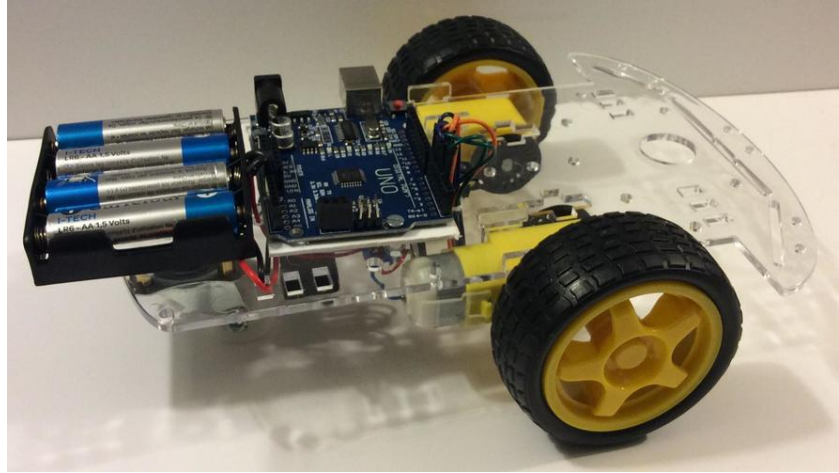
Exercice 2 : Télécommande Bluetooth

- 2.1. Réaliser la chaîne arduino – HC-06 – Smartphone
- 2.2. Avec le logiciel « Bluetooth Electronics », réaliser une page qui permet de contrôler la vitesse (et le sens) des deux moteurs.
- 2.3. Ecrire un programme arduino qui reçoit les données de la question (2.2) et qui les exécute. Télécommander alors le déplacement de votre voiture.

TD No. 8 : Servomoteur, distance et voiture arduino

Ce TD se base sur la voiture arduino de la figure (0). Elle comprend, un bloc piles 4×1.5 V, une carte arduino UNO qui nécessite l'installation du driver CH340 (www.wch.cn/products/CH340.html), un driver de moteur L298, 1 servo moteur qui permet une rotation du capteur de distance et 2 moteurs avec roues ($3 - 6$ V, 100 RPM, 0.44 N.m)

Figure 0. Voiture arduino



Exercice 1 : Mise en œuvre du servomoteur

1.1. Connecter la carte arduino avec le servomoteur

1.2. On suppose que l'angle 90° correspond au servomoteur se trouvant dans l'axe de la voiture. A l'aide de la librairie « Servo.h », du moniteur série, de la fonction « Serial.parseInt() » et de la fonction « .writeMicroseconds(angles en microsecondes) », déterminer la valeur de « l'angle en microsecondes » qui correspond à un angle de 90°

1.3. Déterminer les angles qui correspondent aux « angles en microsecondes » 1000 μ s et 2000 μ s. Donner alors l'expression : angles en μ s = f(angle) en remarquant que c'est une droite. Déterminer alors les « angles en microsecondes » qui correspondent aux angles 0° et 180°

1.4. Prendre en compte les 3 arguments de la fonction « .servo(IO, min, max) » et déterminer si le servomoteur s'oriente correctement pour des angles allant de 20° à 160° .

1.5. Faire passer alternativement le servomoteur de l'angle min à l'angle max en réglant le temps de basculement (en millisecondes) avec le moniteur série. Vous commencerez avec un temps de 1000 s. A partir de quel temps de basculement, le servomoteur n'arrive plus à suivre. Est-ce que le servomoteur se comporte comme un filtre :

Passe-haut

Passe-bande

Passe-bas

Passe-tout

Exercice 2 : Réalisation d'un radar

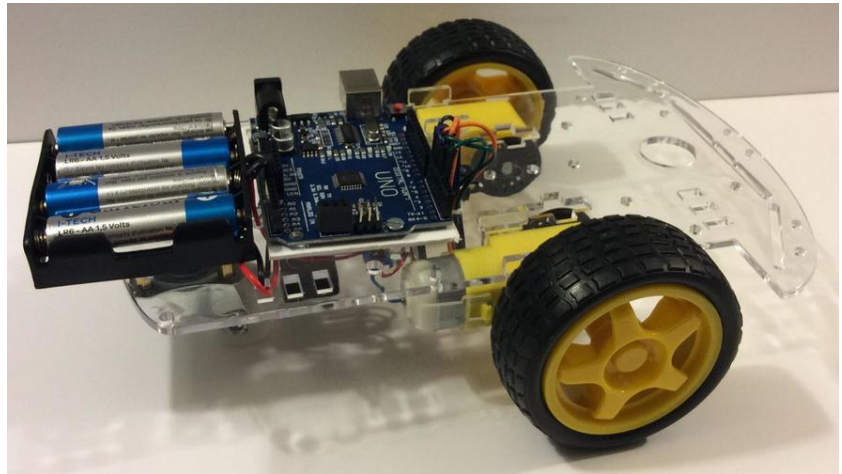
2.1. Avec l'application « Bluetooth Electronic », réaliser le radar avec deux indicateurs, un pour l'angle et un autre pour la distance (il n'est malheureusement pas possible d'utiliser les graphes XY). Réaliser le programme arduino qui envoie les données et vérifier le bon fonctionnement de l'ensemble

2.2. Ajouter alors la commande des roues de votre voiture (c.f. TD n°7) et guider votre voiture en regardant seulement votre téléphone

TD No. 9 : Ligne et voiture arduino

Ce TD se base sur la voiture arduino de la figure (0). Elle comprend, un bloc piles 4×1.5 V, une carte arduino UNO qui nécessite l'installation du driver CH340 (www.wch.cn/products/CH340.html), un driver de moteur L298, 2 moteurs avec roues (3 – 6 V, 100 RPM, 0.44 N.m), 1 servo moteur qui permet la rotation du capteur de distance et 2 capteurs de proximité Keyes IR-01

Figure 0. Voiture arduino



Exercice 1 : Mise en œuvre du capteur

- 1.1. Connecter la carte arduino avec un des capteurs de proximité
- 1.2. Réaliser un programme qui permet de visualiser l'état de la sortie sur le moniteur série
- 1.3. Quel est la valeur logique de la sortie quand l'obstacle est loin ou proche du capteur ?
- 1.4. Etudier la réponse du capteur en fonction de la nature et de la distance de l'obstacle et du réglage du potentiomètre

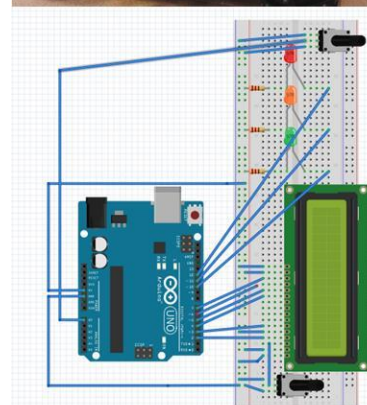
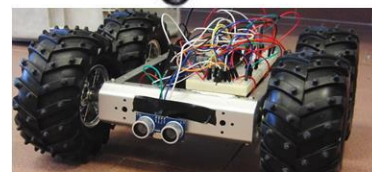
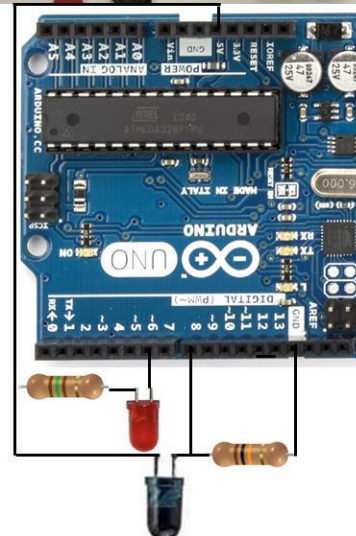
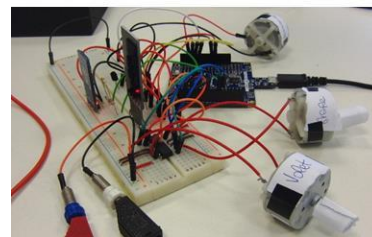
Exercice 2 : Suivi d'une ligne

Vous avez à votre disposition un tableau blanc positionné à l'horizontal sur lequel un parcours a été dessiné avec du scotch noir.

- 2.1. Régler le potentiomètre pour détecter la ligne noire
- 2.2. Réaliser un programme qui permet aux 2 capteurs de détecter la ligne noire et de corriger la trajectoire de la voiture pour rester à cheval sur cette ligne.

ECOLE POLYTECHNIQUE UNIVERSITAIRE DE NICE SOPHIA-ANTIPOLIS

*Cycle Initial Polytechnique
Deuxième année*



Pascal MASSON
Polytech'Nice Sophia, Département électronique
930 route des Colles
06410 BIOT

Tél : 04 89 15 41 67
Email : pascal.masson@unive-cotedazur.fr



Année scolaire 2025/2026