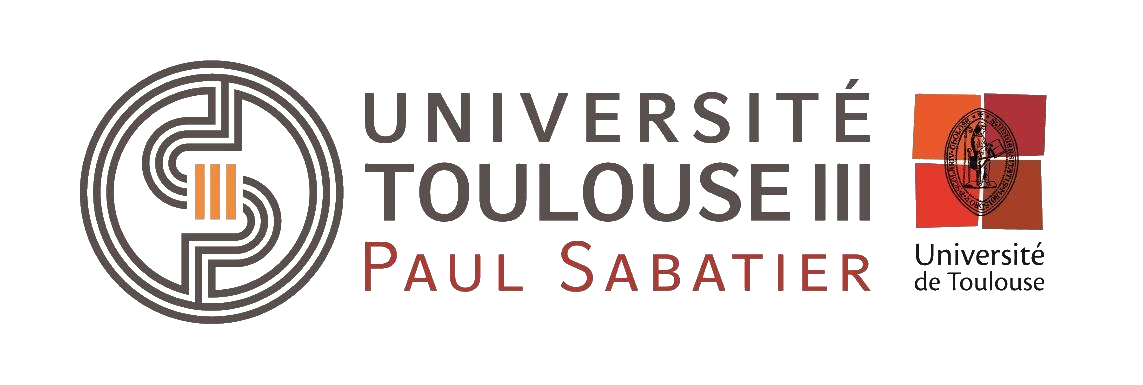
**Bureau d’étude**

**L3 EEA REL**



Tables des Matières

[Introduction 3](#_Toc3614)

[Matériels 3](#_Toc12045)

[Affichage de l’écran LCD 4](#_Toc20893)

[Méthodes 4](#_Toc21577)

[Résultats 6](#_Toc24032)

[Conclusion 6](#_Toc23098)

[La tension du panneaux solaire 6](#_Toc28669)

[Méthodes 6](#_Toc19378)

[Résultats 7](#_Toc26952)

[Conclusion 8](#_Toc10071)

[Le courant du panneaux solaire 8](#_Toc28139)

[Méthodes 8](#_Toc2139)

[Résultats 9](#_Toc23862)

[Conclusion 9](#_Toc26127)

[Commande du hacheur 9](#_Toc6514)

[Méthodes 9](#_Toc10479)

[Résultats 9](#_Toc824)

[Conclusion 9](#_Toc2321)

[Annexe 10](#_Toc27804)

[Sources 10](#_Toc2342)

# Introduction

Notre projet de bureau d’étude porte sur le rechargement d’une batterie grâce à un panneau solaire. Le système disposera d’un afficheur LCD qui affichera la tension ainsi que le courant instantané au bornes du panneaux solaire.

Pour realiser le projet, nous avons décidés de la découper en plusieurs étapes :

- Afficher du texte et des variables sur l’écran LCD

- Récupérer la tension max du panneaux solaire

- Récupérer le courant max du panneaux solaire

- Commander le hacheur

# Matériels

Pour la réalisation du projet nous disposons de plusieurs panneaux solaire se situant sur le toit du bâtiment mais pour notre système nous en utiliserons qu’un seul.

Pour la gestion du programme de gestion de rechargement nous disposons d’une carte Arduino Uno

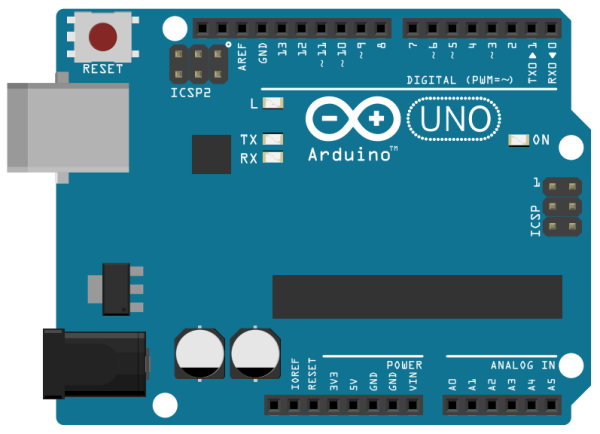


Figure X : représentation graphique de la carte Arduino

Pour simplifier le câblage nous disposons d’une extension qui se superpose à la carte Arduino, il s’agit d’une carte shield

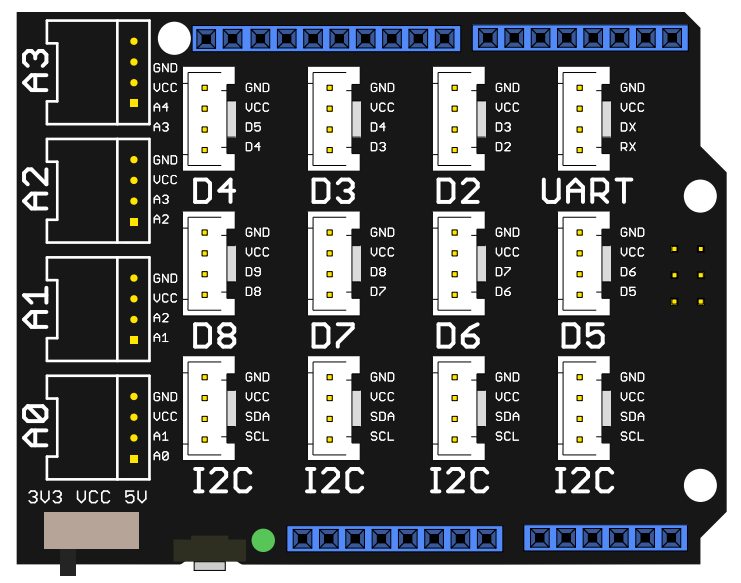


Figure X : Schéma de la carte Shield

Le raccordement entre les deux composants est très simple il suffit seulement de branche le shield sur la carte Arduino car il possède exactement les mêmes connecteurs.

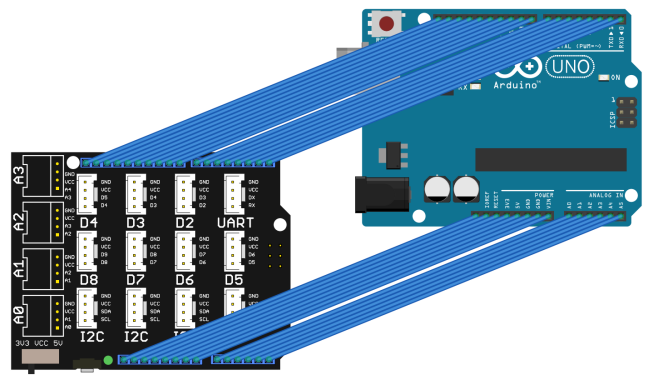


Figure X : Schéma de raccordement de la carte Arduino et le Shield

# Affichage de l’écran LCD

Dans notre système, nous allons afficher des informations en continu du panneaux solaire tel que la tension et le courant maximale qu’il peut fournir . Pour ce faire nous allons utiliser un afficheur LCD qui peut afficher 16 caractère sur 2 lignes, il affiche du texte de couleur noir sur fond jaune. L’afficheur se connecte en I2C ce qui simplifie grandement le câblage mais qui nécessite d’utiliser une libraire

## Méthodes

Pour afficher du texte sur l’écran il nous faut raccorder l’afficheur à la carte Arduino, étant donné que nous avons la carte d extension Shield, nous avons simplement représenté le raccordement au niveau de l’extension. Puisque que l’afficheur se raccorde en I2C, nous utilisons que 4 broches comme indiqué dans le tableau de la figure X.

|  |  |  |
| --- | --- | --- |
| Port Arduino | Broche | Couleur |
| GND (I2C) | GND | Noir |
| Vcc (I2C) | Vcc | Rouge |
| SDA (I2C) | SDA | Blanc |
| SCL(I2C) | SCL | Jaune |

Figure X : Tableau de raccordement entre l’afficheur et le shield

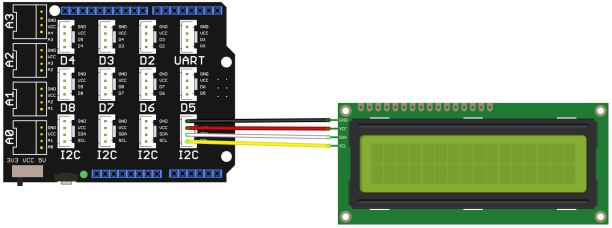


Figure X : Schéma de raccordement entre l’afficheur et le shield

Pour ce qui en est de la programmation le principe de la communication I2C est de réduire le nombre de fils et d’utiliser un bus de communication. Nous devons donc ajouter une librairie qui a déjà était créer pour ce système d’affichage tel que la librairie «rgb\_lcd» visible à la ligne 2 de la figure X. De plus il faut ajouter le libraire Arduino «Wire.h» qui est faite pour l’utilisation des communications en bus I2C.

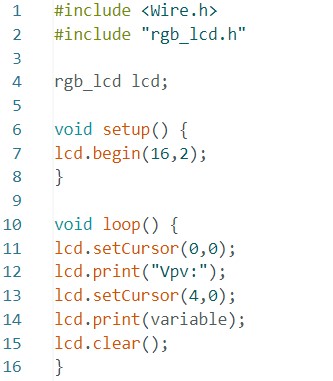


Figure X: programme de l’affichage sur l’écran LCD

Dans le programme nous pouvons voir la fonction «lcd.» qui est la simplification de «rgb\_lcd» qui est plus longue à écrire, ce paramétrage est visible ligne 4.

Pour pourvoir afficher du contenu sur l’afficheur LCD il faut définir les caractéristiques de l’afficheur par la fonction «lcd.begin» en définissant le nombre de caractères possible est le nombre de linges. Il faut également définir la position du curseur, ligne 11, on définit le curseur sur première ligne au premier caractère. Puis avec la fonction «print», on affiche du texte en le mettant entre guillemet, si on veut afficher la valeur d’une variable, il faut seulement mettre le nom de la variable. Si on veut changer le texte sur l’afficheur il faut l’effacer puis réaffirmer le texte ou la variable souhaité.

## Résultats

Au premier test, l’afficheur n’a afficher que la moitier du texte souhaité, ce problème provenais du fait que la fonction loop afficher puis effacer le texte à la même fréquence que horloge de la carte Arduino, mais l’afficheur n’a pas le temps d’écrire tout les caractère. Pour résoudre ce soucis il à fallu rajouter une ligne pour mettre en pause l’affichage de quelques secondes.

Mais une fois ce problème résolut, l’affichage est clair et le texte est suffisamment lisible, les lettres qui sont en majuscules sont facilement differentiable de celles en minuscules. Le premier programme réaliser pour afficher du texte et des variables étant long et peu pratique pour afficher en continu les différents textes et variables à la suite, nous avons décider de créé deux fonctions externe à la fonction void loop afin de simplifier et alléger la fonction principale visible sur la figure X.

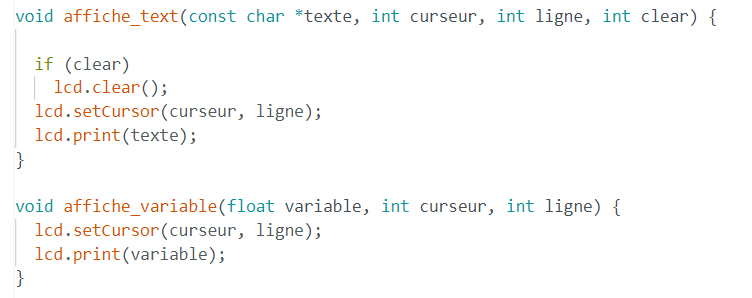


Figure X : fonction permettant d’afficher du texte et des variables

## Conclusion

L’affichage nous permet pour la suite de l’évolution du projet, voir les valeurs reçu par les différents capteurs tel que la valeur de la tension ou le courant. Grâce au deux fonctions d’affichage nous pouvons facilement voir le programme que nous somme modifiant sans chercher parmi les lignes de codes ce qui permet l’affichage et ce qui nous intéresse.

# La tension du panneaux solaire

## Méthodes

Pour lire la tension au borne du panneaux solaire nous allons utiliser les ports d’entrée analogique de la carte Arduino. Dans un premier temps nous avons fait un programme de test avec une résistance variable pour verifier si l’on pouvais réceptionner une tension. Pour ce faire nous avons alimentée un potentiomètre qui a pour sortie l’entré A0, le schéma de cable est visible sur la figure X.

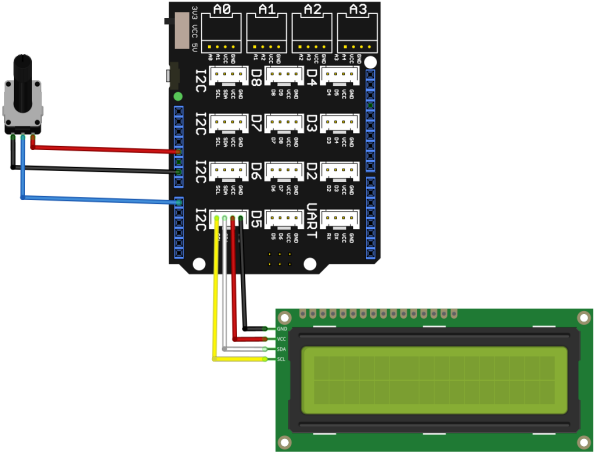


Figure X : Shéma de mesure de tension en A0

Le panneaux solaire fournit en permanence une tension que va allumé la batterie, pour calculer la tension maximale qui peut être produite par le panneaux solaire nous avons mit en place un pont diviseur de tension.

Le panneaux solaire peut fournir une tension maximale de 22V mais la carte Arduino ne peut réceptionné maximum 5V au borne de ses ports Analogique, nous devons donc abaisser la tension pour que la tension maximale du panneaux solaire soit perçu comme 5V pour la carte Arduino. D’où le système de pont diviseur de tension.

## Résultats

Pour mettre en place ce système, il faut utiliser la formule est Vs=Vcc\*(R2/(R1+R2)) où R1 est la résistance en amont du système alimenté par la tension Vcc et R2 celle reliée à la masse. Entre les deux résistance, nous tirons un fil jusqu’à l’entrée A0 de la carte Arduino. Dans notre cas, nous avons une tension Vcc de 22V et une tension Vs qui doit être de 5V. Nous avons choisit la valeur d’une des deux résistance car nous avons deux inconnues mais qu’une seule equation. En choisissant 10kΩ pour R1 nous trouvons une valeur de 34kΩ pour R2, mais n’ayant pas de résistance de cette valeur, nous avons choisit deux résistance en série de 12kΩ et 22kΩ. En figure X, nous avons le schéma électrique du pont diviseur de tension raccorder au Shield de la carte Arduino, la valeur de le tension est afficher sur l’écran LCD.

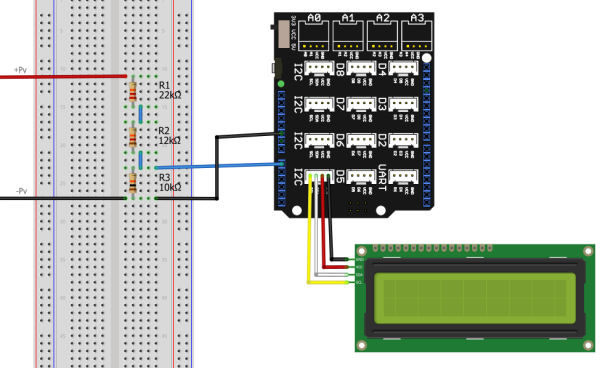
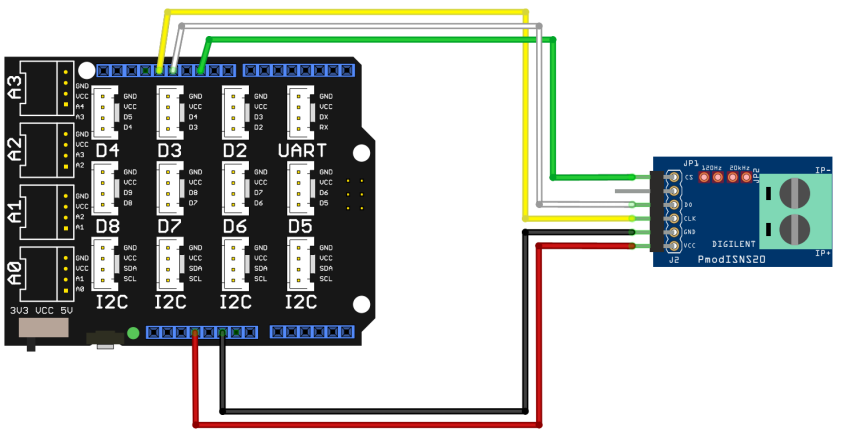


Figure X : Shéma du pont diviseur de tension du panneaux solaire

## Conclusion

# Le courant du panneaux solaire

## Méthodes



|  |  |  |
| --- | --- | --- |
| Port Arduino | Broche | Couleur |
| +3.3V | Vcc | Rouge |
| GND | GND | Noir |
| 13 | Clock | Jaune |
| 12 | Miso | Blanc |
| 10 | Chip Select | Vert |

Figure X : Tableau de câblage du capteur de courant

## Résultats

## Conclusion

# Commande du hacheur

## Méthodes

Pour la commande du hacheur nous allons la commander grace à la commande du transistor qui se trouve dans le hacheur. On le commandera à une fréquence supérieur à 20kHz pour ne pas entendre de bruit, de ce fait nous avons décider de l’utiliser à 50kHz minimum selon ce que peut fournir la carte Arduino.

Au niveau de la carte Arduino nous allons utiliser la fonction PWM (Pulse Width Modulation) ce qui signifie Modulation de largeur d’impulsion, elle nous permettra de créer le signal de commande du hacheur. En fonctionnement normale la fréquence du PWM est de 500Hz, nous avons donc du effectuer des modifications pour augmenter sa fréquence au maximum soit 62 500Hz.

Pour cela il faut augmenter la fréquence de l’horloge des sorties PWM et cette modifications ne peut s’applique que pour certaine sorties appelés Timers. La carte Arduino en possèdes 3, Timer 0 est pour les broches 5et 6, Timer 1 pour les broches 9 et 10 et le Timer 2 pour les broches 3 et 11.

## Résultats

## Conclusion

# Annexe

# Sources

1. <https://arduino-france.site/pwm-arduino/>
2. <https://discordapp.com/channels/@me/1037786304852787281/1084463527621828669>
3. <https://www.youtube.com/watch?v=uBVdJnP87ho&t=75s>
4. <https://www.youtube.com/watch?v=YBNEXWp-gf0>
5. <https://www.locoduino.org/spip.php?article89>