REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE Dr YAHIAFARES de MEDEA

Faculté des Sciences et technologie

Département du Génie Electrique et de l'Informatique

Spécialité : Electrotechnique

Option : Energie solaire et éolienne

Mémoire en vue de l'obtention du diplôme de master

THEME :

CONCEPTION ET REALISTION D'UN SUIVEUR DU SOLEIL A BASE DE LA CARTE ARDUINO UNO

Présenté par :

BENKORTBI Abdelhakim

FERROUKH Mohamed Amine

Dirigée par:

Mr. Noureddine Ould cherchali

Soutenu : 10 septembre 2015

REMERCIEMENT

Avant tout, je remercie ALLAH, le tout puissant, de m'avoir donné le courage et la volonté pour accomplir ce travail de recherche.

J'aimerais exprimer tout particulièrement ma gratitude à mon encadreur, Mr. OULD CHERCHALI, pour la confiance qui m'a accordé, pour le dévouement, sa compétence, dont il a fait preuve en m'offrant la possibilité de mener à bien ce travail de recherche et pour ses encouragements constants tout en long de celui-ci, ses critiques constructives, pour sa droiture et sa franchise.

Mes plus profonds remerciements vont à mes parents. Tout au long de mon cursus, ils m'ont toujours soutenu, encouragé et aidé. Ils ont su me donner toutes les chances pour réussir. Qu'ils trouvent, dans la réalisation de ce travail, l'aboutissement de leurs efforts ainsi que l'expression de ma plus affectueuse gratitude.

BENKORTBI ABDELHAKIM

Remerciement

Nous tenons tout d’abord à remercier Allah le tout puissant et miséricordieux, qui nous a donné la force et la patience d’accomplir ce Modeste travail.

Je remercier notre encadreur Mr : N. Ould cherchali, qui a eu la gentillesse de lire et corriger ce travail, son précieux conseil son patience et son aide durant toute la période du travail.

mes vifs remerciements vont également aux membres du jury pour l’intérêt qu’ils ont porté à notre recherche en acceptant d’examiner notre travail Et de l’enrichir par leurs propositions.

Mes remerciements s'adressent également à mes chers parents et ma famille pour me soutenir à tous les stades de l'étude.

Sans oublier tous mes amis et mes collègues qui m'ont aidé durant l'étude et tous ceux qui m'ont encouragé à réaliser ce travail.

FERROUKH Mohamed Amine

ملخص

هذا العمل يهتم بدراسة و انجاز متتبع شمسي ثنائي المحور لتتبع مسار الشمس باستعمال المعادلات الرياضية لزاوية السمت و زاوية الإرتفاع التي تحدد موقع الشمس و متحكم أردوينو بإستخدام لغة البرمجة C+ الذي يسمح لنا بتشغيل محركات الخطوة لتحريك اللوح الشمسي بمواجهة الشمس.

كلمات مفتاحية: مسار الشمس, لغة البرمجة C, متحكم, أردوينو, محرك الخطوة, متتبع شمسي.

Résumé

Ce travail concernant l'étude, conception et la réalisation d'un suiveur solaire a deux axes suivant la trajectoire du soleil et selon les équations mathématiques d'azimut et l'angle d'élévation pour orienter un panneau solaire. En utilisant la carte Arduino Uno pour le control de ce suiveur solaire et la programmation de langage C+ de ce microcontrôleur pour actionner les moteurs pas à pas

Mots clé : Arduino, langage C, microcontrôleur, moteurs pas à pas, suiveur de soleil, trajectoire de soleil.

Abstract

this work concern the study, design and realization of a solar tracker with two axes following the sun's path and the mathematics equations of azimuth and elevation angle to move a solar panel using an Arduino Uno board to control the solar tracker and the microcontroller programing language C+ which allow to move the stepper motors.

Keywords: Arduino, language C, microcontroller, solar tracker, stepper motors, sun's path.

SOMMAIRE

Pages

Introduction générale 1

Chapitre I : Le gisement solaire

I. 1. Introduction……..…… 3
I. 2. Géométrie terre-soleil……..…. 3
I. 2.1. Position du soleil par rapport à la terre……... 4
a) La déclinaison solaire δ…….. 5
b) Hauteur h et Azimut *a* de soleil……..…..... 5
c) La longitude λ……..……..... 6
d) La latitude φ……..... 6
e) L'altitude Z ……..... 7
f) L'angle horaire ω……..... 7
I. 2.2. Les temps solaires ……...…. 8
I. 2.3. L'irradiation solaire ……... 9
a) Diffusion par l'atmosphère terrestre……..... 9
b) Nombre d'air mass……... 9
I.3. l'énergie solaire photovoltaïque…….. 10
I. 3. 1. Principes de fonctionnement d'une cellule photovoltaïque……..... 10
I. 3. 2. Avantages et inconvénients de l'énergie solaire photovoltaïque……..... 12
I.4. Le gisement solaire en Algérie……..... 13
I.4. Conclusion……..... 15

Chapitre II : Les moteurs pas à pas et les suiveurs solaires

II. 1. Introduction……..... 16
II. 2. Les moteurs pas à pas ……..……..... 16
II. 2. 1. Caractéristiques générales d'un moteur pas à pas……..... 17
II. 2. 2. Les différents types de moteur pas à pas……..……..... 18
II. 2.2.1 Les moteurs à aimant permanent……..…..... 18
II. 4.2. Les moteurs à reluctance variable……..... 20
II. 4.3. Les moteurs pas à pas hybride……..... 21
II. 3. La commande d'un moteur pas à pas par un Arduino…… 23
II. 3. 1. La connexion d'un moteur unipolaire…… 23
II. 3. 2. La connexion d'un moteur pas à pas bipolaire…… 27
II. 4. Etude des différents systèmes de poursuite……..... 33
II.4.1. Mécanisme d'orientation……..... 33
II.4.2. Degré de liberté de mouvement ……..... 36
II.4.3. Type de commande……..... 37
II.4.4. L'intérêt des panneaux mobiles par rapport aux panneaux fixes……..... 37
II. 5. Conclusion ……..... 38

Chapitre III : Le circuit Arduino

III. 1. Introduction…… 39
III. 2. Le circuit Arduino ……. 39
III. 2. 1. Le microcontrôleur…….. 40
III. 2. 1. 1. La structure de microcontrôleur……... 41
III. 2. 3. Les différentes familles d'Arduino…….. 45
III. 2. 3. 1. La carte Arduino UNO …… 45
III. 2. 3. 4. Les autres cartes Arduino …… 51
III. 3. Les cartes d'interface ou shields……..... 51
III. 3.1. La carte d'interface RTC (Real Time Clock)……..... 52
III. 4. Environnement de développement intégré (EDI)……... 54
III. 4. 1. Transfert des programmes vers la carte Arduino…… 55
III. 5. Le langage de programmation……. 60
III. 6. Conclusion…… 66

Chapitre VI : Réalisation d'un suiveur solaire

IV. 1. Introduction…… 67
IV. 2. Principe de fonctionnement d'un suiveur solaire ……. 67
IV. 3. Câblage de fonctionnement d'un suiveur solaire contrôlé par un Arduino…….. 69
IV. 3. Etude mécanique et la réalisation d'un suiveur…….. 73
IV. 3. 1. Croquer de montage…….. 74
IV. 4. Organigramme de suiveur solaire a deux axes ……. 76
IV. 5. Programmation…… 79
IV. 5. 1. Programmation de carte RTC DS1302…….. 79
IV. 5. 1. Programmation d'un suiveur solaire a deux axes avec Arduino et le module RTC……. 75
IV. 6. Conclusion……..

Conclusion Générale…… 87

ANNEXE …….. 88

Bibliographie…… 96

Pages

Figures du chapitre I

Figure. I. 1 : La structure interne du soleil..... 3
Figure. I. 2 : L'orbite terrestre et les saisons ……..... 4
Figure. I. 3 : Repérage du soleil……..... 5
Figure I. 4 : Hauteur angulaire h et l'azimut *a*.……..... 6
Figure. I. 5 : les coordonnées angulaires d'un point sur la terre..... 7
Figure. I. 6 : L'angle horaire ω. ……..... 8
Figure. I. 7: Convention photovoltaïque ……..…… 10
Figure. I. 8 : L'effet photovoltaïque…….. 11
Figure. I. 9 : Composant d'une cellule photovoltaïque …… 11
Figure. I. 10: L'émission de CO2 dans chaque source d'énergie ……..…… 12
Figure I. 11 : Irradiation globale journalier reçu sur plan horizontal au mois de juillet et décembre.……..... 13

Figures du chapitre II

Figure II. 1 : Différents tailles des moteurs pas à pas….....
Figure II. 2 : Moteur à aimants permanents.....
Figure II. 3 : Moteur à réluctance variable (MRV) a circuit magnétique simple.....
Figure II.4 : le moteur pas à pas hybride.....
Figure II.5 : le moteur pas à pas hybride utilise dans le montage de suiveur.....
Figure II. 6 : Schéma d'un moteur unipolaire.....
Figure II. 7 : Schéma de câblage d'un moteur unipolaire….....
Figure II. 8 : Schéma de câblage d'un moteur unipolaire avec Arduino et ULN2003....
Figure II. 9 : les séquences d'Arduino pour piloter les moteurs unipolaires.....
Figure II. 10 : Schéma d'un pont H.....
Figure II. 11 : Schéma de pont H L298 avec Arduino.....
Figure II. 12 : Schéma de pont H L298 avec Arduino.…….....
Figure II. 13 : les séquences d'Arduino pour piloter les moteurs bipolaires avec un pas
Figure II. 14 : les séquences d'Arduino pour piloter les moteurs bipolaires avec un couple max.....
Figure II. 15 : les séquences d'Arduino pour piloter les moteurs bipolaires à demi-pas.....
Figure II. 16 : Chronogramme du pilotage à demi-pas…….....
Figure II. 17 : suiveur solaire passif…….....
Figure II. 18 : suiveur de soleil a une cellule bifaciale.....
Figure II. 19 : Principe d'équilibrage d'ombre….....
Figure I.20 : comparaison entre système fixe incliné 35° et système suiveur a deux axes toute l'année au site de Médéa …….....

Figures du chapitre III

Figure III.1 : Une carte Arduino……..... 39
Figure III.2 : Atmel Microcontrôleur……..... 39
Figure III.3 : un microcontrôleur Microchip……..... 39
Figure III.4 : Structure de microcontrôleur……..... 42
Figure III.5 : La carte Arduino Uno……..... 46
Figure III.6 : Schéma bloc de description du microcontrôleur AVR…..... 47
Figure III.7 : 32 Ko disponibles pour vos programmes en Arduino UNO..... 48
Figure III.8 : structure d'une carte Arduino Uno ……..... 49
Figure III.9 : Entrées/sorties numériques D0 à D13..... 50
Figure III.10 : La carte Arduino Méga ……..... 51
Figure III.11 : La carte Arduino Bluetooth …… 51
Figure III.12 : La carte Arduino mini.……..... 51
Figure III.13 : La carte Arduino Nano…… 51
Figure III.14 : La carte Arduino LilyPad…… 51
Figure III.15: Un shield prototypage Sparkfun ProtoShield.…… …… 52
Figure III.16: Une carte RTC DS1302…… 53
Figure III.17: Le circuit de la carte RTC ……. 53
Figure III.18 : Espace de développement intégré (EDI) dédié au langage Arduino.…… 55
Figure III.19 : Un terminal série…….. 55
Figure III.20 : Compilation terminée…….. 56
Figure III.21 : Sélectionner la carte Arduino……..... 57
Figure III.22 : Transférer le programme a la carte Arduino……..... 58
Figure III.23 : réglage de débit de communication…….. 59
Figure III.24 : Du code source a la carte ……..... 59
Figure III.25 : Structure d'un programme……..... 60

Figures du chapitre IV

Figure IV.1 : Les coordonnées géographiques de site MEDEA…… 67
Figure IV.2 : Trajectoire du soleil de site MEDEA……..... 67
Figure IV.3: Darlington ULN2803……. 68
Figure IV.4 : Carte de commande de moteur pas à pas unipolaire……. 69
Figure IV.5 : prototypage de la carte de commande avec Fritzing…… 69
Figure IV.6.a : Soudage des supports pour ULN2803…….. 70
Figure IV.6.b : Carte de commande moteur pas à pas……... 70
Figure IV.7 : schéma électrique de commande d'un suiveur solaire a deux axes par Arduino……... 71
Figure IV.8 : Croquer de suiveur solaire dessiné sur Solidworks…….. 73
Figure IV.9 : Support de rotation pour l'angle d'hauteur de soleil…… 74

Figure IV.10 : la réalisation du suiveur solaire qui nous fait..... 75
Figure IV.11 : les différents composants du suiveur solaire qui nous réalisons..... 76
Figure IV.12 : Organigramme d'un suiveur solaire a deux axes avec l'horloge DS1302..... 77
Figure IV.13 : Organigramme des fins de course..... 78
Figure IV.14 : Moniteur série résultant de la carte DS1302..... 81

Figures de l'annexe

Figure II.A.1. Block Diagramme de DS1302..... 87
Figure II.A.2. Résumé de transfert de données dans DS1302..... 87
Figure III.A.1. Schéma de branchage de pont H avec Arduino..... 88
Figure IV.A.1. Schéma des entrées et sorties de ULN2803..... 88
Figure IV.A.2. Schéma d'un pair entrées et sorties de l'ULN2803..... 89
Figure IV.A.5 : Vue éclaté de support d'un suiveur solaire sur Solidworks. 91
Figure IV.A.6 : Les dimensions de support d'un suiveur solaire dessiné sur Solidworks. 92
Figure IV.A.7 : Les cotations de support de rotation pour l'angle d'hauteur de soleil. 93
Figure IV.A.8 : L'angle max de rotation pour l'angle d'hauteur de soleil. 94

Liste des tableaux

Tableau. I. 1 : Potentiel solaire en Algérie..... 15
Tableau II. 1 : Étude comparative de différents types de moteur pas à pas..... 22
Tableau III.1 : Synthèse des caractéristiques d'Arduino Uno..... 46
Tableau III.2 : Fonction d'entrées et sorties numériques et analogiques..... 65

Introduction générale

INTRODUCTION GENERALE

La production d'énergie est un défi de grande importance pour les années à venir, en effet les besoins énergétiques des pays industrialisés ne cessent d'augmenter, par ailleurs les pays en voie de développement auront besoin de plus en plus d'énergie pour mener à bien leur développement.

De nos jours, une grande partie de la production mondiale d'énergie est assurée à partir des énergies fossiles comme le pétrole, le charbon, le gaz naturel ou encore l'énergie nucléaire, la consommation de ces sources donne lieu à des émissions de gaz à effet de serre et donc une augmentation de la pollution. Le danger supplémentaire est qu'une consommation excessive du stock de ressources naturelles réduit les réserves de ce type d'énergie de façon dangereuse pour les générations futures, Il est estimé que les réserves mondiales seront épuisées vers 2030 si la consommation n'est pas radicalement modifiée, et au maximum vers 2100 si des efforts sont produits sur la production et la consommation.[1]

Aussi face aux multiples crises économiques et pétrolières comme la crise pétrolière de 1973, par l'augmentation brutale du prix du pétrole, conduit l'homme à s'intéresser à d'autres sources d'énergie renouvelable, au premier rang desquelles figure l'énergie solaire, qui constituent un secteur stratégique et occupent une place privilégiée dans les domaines de recherche et développement.

Pour améliorer le rendement des GPV, il y a beaucoup de technologies l'un est d'utiliser des system de poursuite (avec des moteurs pas à pas par exemple) pour qu'ils puissent tourner et suivre le soleil toute la journée pour maximiser la production de l'énergie solaire électrique à partir d'un panneau photovoltaïque, dans notre réalisation on utilise une carte Arduino pour commander les moteurs pas à pas de ce suiveur.

Dans ce travail, nous nous somme intéresses à l'étude, conception et la réalisation d'un suiveur de soleil, Ce mémoire est partagé en quatre chapitre :

Dans le premier chapitre nous présentons les coordonnées relatives au mouvement du soleil et une généralité sur le gisement solaire.

Dans le deuxième chapitre nous présentons les différents types de moteur pas à pas que nous pouvons utiliser pour la réalisation de notre suiveur de soleil et les différentes modes de

commandes de moteur pas à pas. Puis les différents systèmes de poursuite d'un suiveur solaire.

Le troisième chapitre dédié à l'étude de microcontrôleur Arduino Uno, sa structure et ses différents types, puis l'environnement de développement intégré (EDI), le langage et les fonctions de programmation d'Arduino.

Dans le quatrième et dernier chapitre, nous présentons toutes les étapes de la réalisation du suiveur solaire.

Et à la fin on termine par une conclusion générale.

Chapitre I

Le gisement solaire

I. 1. Introduction

Le soleil l'origine de l'énergie photovoltaïque, le principe du panneau solaire a été découvert par E. Becquerel en 1839, qui consiste en une conversion de la lumière en énergie électrique, n'aura été exploité qu'un siècle plus tard comme une solution adéquate pour les populations isolées, leur permettant une certaine autonomie énergétique [2].

L'importance de gisement solaire c'est calculer la présence de rayonnement solaire qui nous définit la quantité d'énergie fournie est aussi le rôle des rayonnements en changement du climat et des phénomènes métrologiques.

Dans ce chapitre, nous allons avoir comment calculer les données fondamentales comme les cordonner géographiques, les rayonnements globaux reçu par le panneau photovoltaïque.

I. 2. Géométrie terre-soleil

Le Soleil (Sol en latin, Helios) est l'étoile du Système solaire. Dans la classification astronomique, le soleil est une étoile de type naine jaune son diamètre est de 1391000 km (100 fois celui de la terre), qui se compose d'hélium (24 % de la masse ou 7,8 % du volume), et d'hydrogène (74 % de la masse ou 92,1 % du volume). Autour du soleil gravite la Terre, avec sept autres planètes, de très nombreux astéroïdes et comètes et une bande de poussière. Le Soleil représente à lui seul 99,86 % de la masse du système solaire ainsi constitué, la terre étant à une distance de 150.106 km du soleil, elle reçoit approximativement une énergie de $1{,}8.10^{14}$ kW, une énorme énergie renouvelable et inépuisable [2] [3].

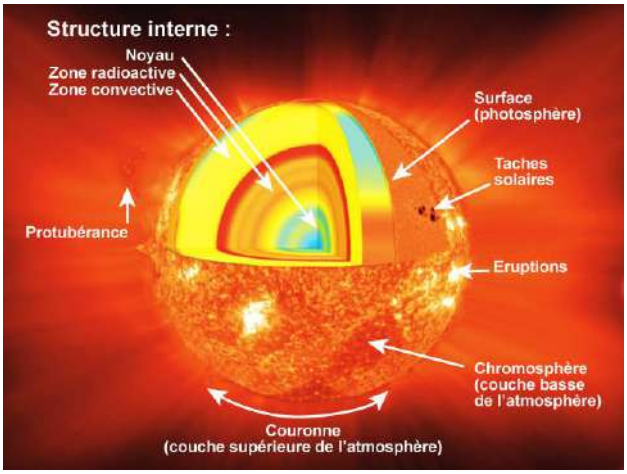


Figure I. 1 : La structure interne du soleil.

I. 2. 1. Position du soleil par rapport à la terre

La terre déplace autour du soleil sur une trajectoire elliptique on l'appelé l'elliptique céleste dont le soleil occupe un foyer au cours d'une année avec une distance de 150 millions kilomètres de la terre, La terre tourne par rapport à cet écliptique céleste avec une déclinaison qui varie avec les saisons, En vingt-quatre heures, la terre effectue une rotation complète d'Ouest en Est autour de son axe ; c'est la base de notre système horaire. De cette rotation découle la succession des jours et des nuits [4].

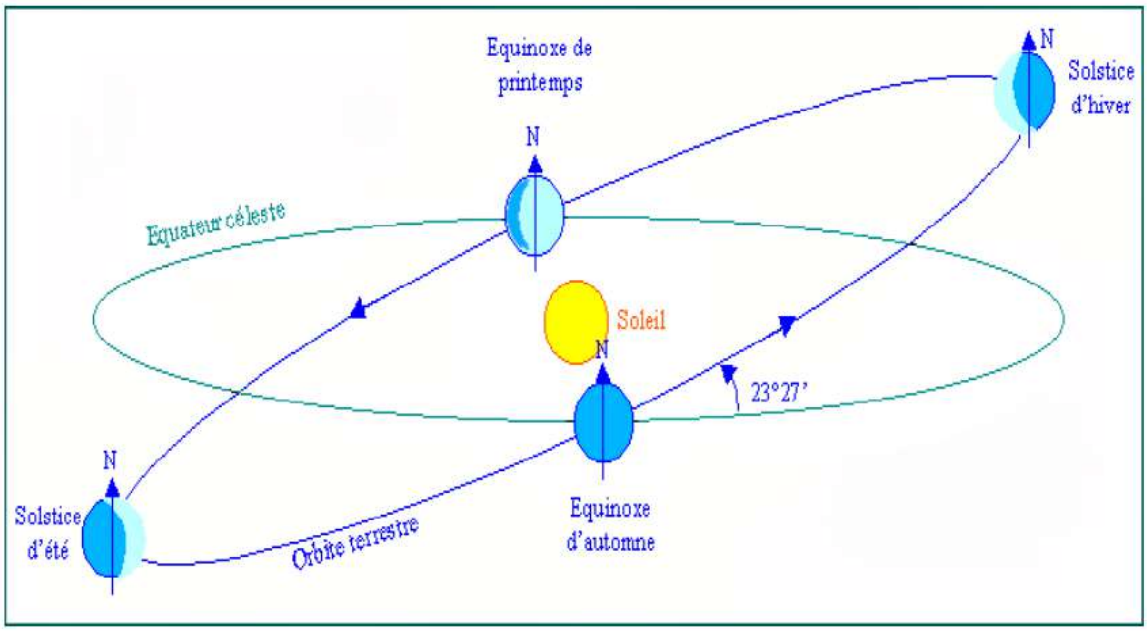


Figure I. 2 : L'orbite terrestre et les saisons.

Sur la figure I.3 on a représenté le mouvement apparent du soleil, On peut donc repérer le soleil grâce à 2 angles :

1/ l'azimut « a » ou angle que fait la projection de la direction du soleil avec la direction du sud. Cet angle est positif vers l'ouest.

2/ La hauteur « h » ou angle de la direction du soleil avec l'horizontale.

Ces deux angles sont évidemment fonction de la latitude du lieu, de la date et de l'heure. La date sert à déterminer la déclinaison f du soleil

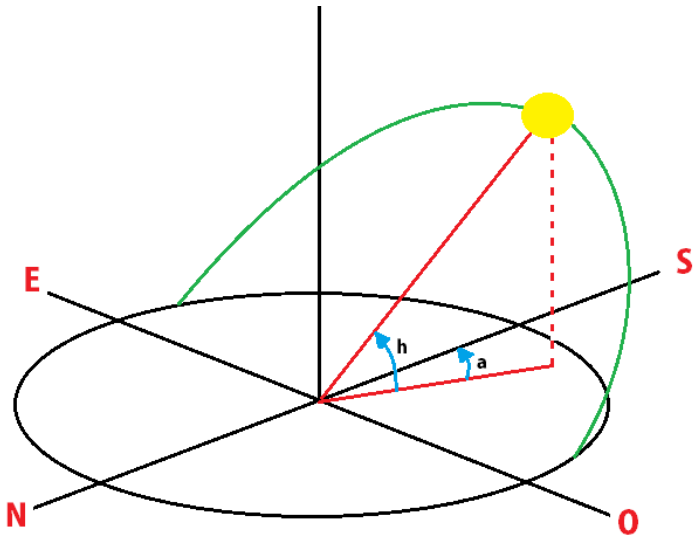


Figure I. 3 : Repérage du soleil.

a) La déclinaison solaire δ

La déclinaison solaire est l'angle formé par la direction du soleil et le plan équatorial terrestre, cet angle varie au cours de saisons de -23.45° le 21 Décembre à +23.45° le 21 Juin. Pour calculer cette déclinaison par deux méthodes. L'équation la plus simple a été développée par *Cooper* [5] :

$$\delta = 23.45 \sin\left(360 \times \frac{284+n_j}{365}\right) \qquad (I.1)$$

n_j *le nombre de jours ecoulés depuis le* 1^{er} *Janvier*

Et pour une meilleure précision on utilise l'équation de *Spencer* avec un écart de 0.0006 radian :

$$\delta = \frac{180}{\pi}\ (0.006918 - 0.399912\cos\tau + 0.070257\sin\tau - 0.006758\cos 2\tau + 0.000907\sin 2\tau - 0.00269\cos 3\tau - 0.00148\sin 3\tau) \qquad (I.2)$$

Avec :

$$\tau = \frac{360(n_j - 1)}{365} \qquad (I.3)$$

b) Hauteur h et Azimut a de soleil

Le midi solaire est le moment ou le soleil est à sa hauteur maximale, la hauteur angulaire du soleil sur le plan horizontal s'annule au lever ou au coucher du soleil et prend sa valeur maximale au midi. Pour calculer cette *hauteur angulaire* on a la relation suivante [5], [6] :

$$\sin(h) = \sin\varphi \sin\delta + \cos\varphi \cos\delta \cos\omega \quad (I.4)$$

$\sin(h)$hauteur angulaire , φ la latitude du lieu, δ declinaison du soleil, ω l'angle horaire.

Au lever de soleil s'obtient:$\sin(h) = 0$:

$$\cos\omega_0 = -\tan\varphi \tan\delta \quad (I.5)$$

Et on peut calculer la durée astronomique du jour en faisant :

$$D_0 = \frac{2}{15} arc\ \cos(-\tan\varphi \tan\delta) \quad (I.6)$$

L'azimut c'est l'angle a compris entre le meridien du lieu et le plan vertical passant par le soleil, L'azimut est relie a l'angle horaire, la hauter et la declinaison par la relation :

$$\sin a = \frac{\cos\delta . \sin\omega}{\cos h} \quad (I.7)$$

Ou la relation indépendante de la hauteur est la suivante :

$$\tan a = \frac{\sin\omega}{\sin\varphi \cos\omega - \cos\varphi \tan\delta} \quad (I.8)$$

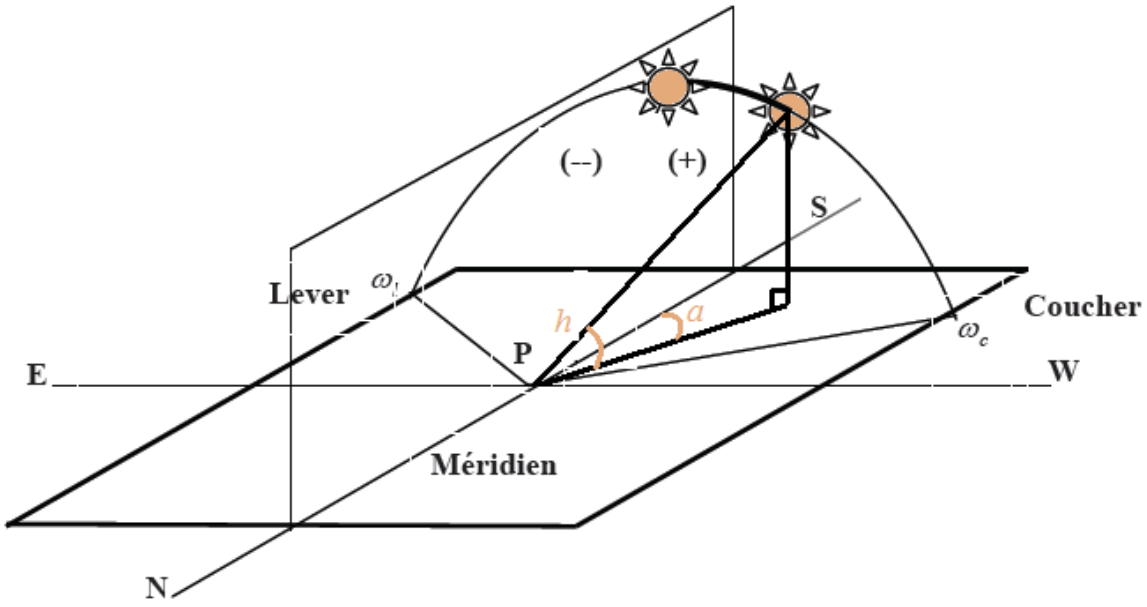


Figure I. 4 : Hauteur angulaire *h* et l'azimut *a*.

c) La latitude φ

Une des coordonnées terrestres d'un point de notre planète. C'est l'angle que fait le plan de l'équateur avec la direction reliant le centre de la terre au point considéré. Sa valeur est positive dans l'hémisphère nord, et est négative dans l'hémisphère sud.

d) La longitude λ

Une des coordonnées terrestres d'un point de notre planète. C'est l'angle que fait le méridien local passant par le point considéré avec le méridien d'origine passant par la ville de Greenwich (en Angleterre). Sa valeur est positive à l'ouest et négative à l'est de la méridienne origine [5].

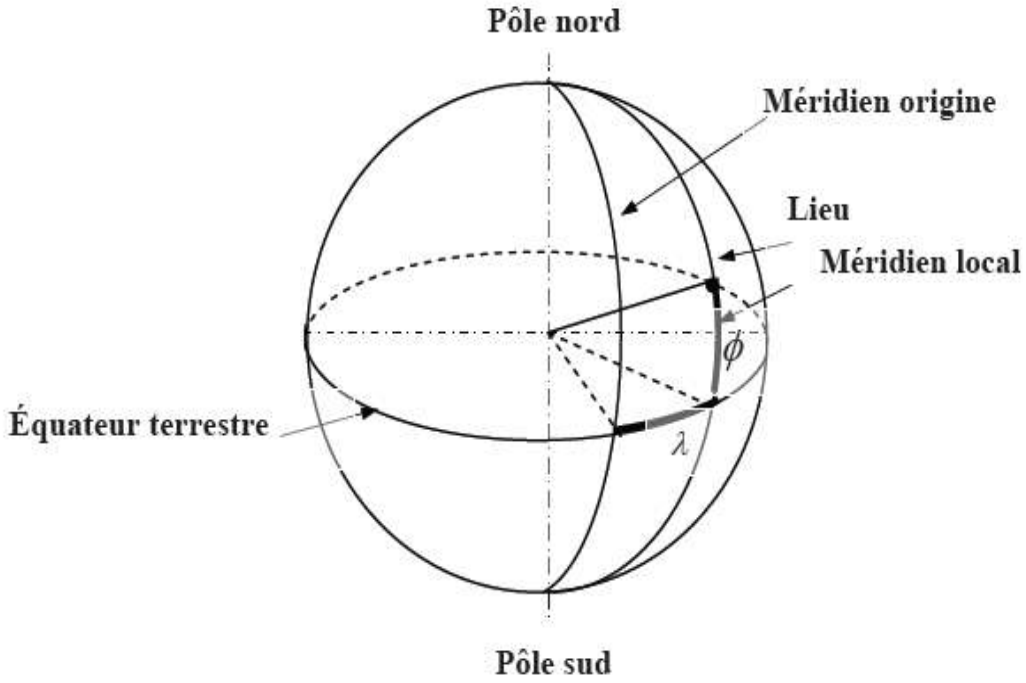


Figure I. 5 : les coordonnées angulaires d'un point sur la terre.

- ***Méridien***

Grand cercle de la terre passant par les pôles. Tous les points d'un même méridien ont évidemment la même longitude ; le méridien pris pour origine (0°) des longitudes est celui de Greenwich. Le plan méridien en un lieu est déterminé par ce lieu et par l'axe des pôles ; sa trace au sol est parfois dénommée : méridienne. Le temps solaire vrai est identique, à un instant donné, pour tous les points d'un même méridien.

e) L'altitude Z

C'est la distance verticale entre ce point et une surface de référence théoriquement est le niveau de mer qui égale à *0* mètre.

f) L'angle horaire ω

L'angle horaire c'est l'angle de rotation régulière de la terre autour de son axe qui se varie positivement de 0° à 360°, cet angle augmente de 15° par heure et la valeur 0° est atteinte au passage du soleil au vertical zénith du plan méridien.

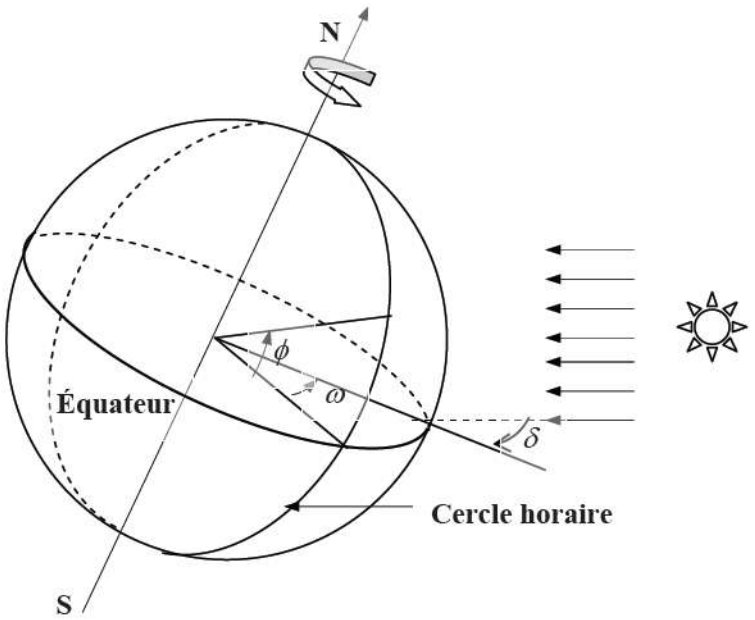


Figure I. 6 : L'angle horaire ω.

L'angle horaire est mesuré à partir du midi solaire, positif s'il est avant-midi et négatif après-midi.

$$\omega = 15(12 - TSV) \qquad (I.9)$$

TSV est le temps solaire vrai exprimé en heures.

I. 2. 2. Les temps solaires

I. 2. 2. 1. Le Temps Universel

Le temps universel (TU) est défini par l'heure de passage du soleil à la méridienne origine de Greenwich (GMT).

I. 2. 2. 2. Le Temps Solaire Local

En temps solaire locale (TSL) on fait la correction de longitude avec le temps universel (TU) avec la formule :

$$TSL = TU + \frac{\lambda}{15} \qquad (I.10)$$

I. 2. 2. 3. Le Temps Solaire Vrai

C'est le temps solaire local corrigé du l'équation du temps, donnée astronomique universel liée à l'excentricité de l'orbite de la terre autour du soleil :

$$TSV = TSL + \Delta t \tag{I. 11}$$

I. 2. 2. 4. Le temps solaire moyen

Donné par la relation $TSV - TSM = Et$

Et : Equation du temps pour corriger TSV par rapport TSM, elle est donnée par l'équation :

$$Et = 9.87 \sin 2n - 7.53 \cos n - 1.5 \sin n \tag{I. 12}$$

$$n = \frac{360}{365}(n_j - 81) \tag{I. 13}$$

n_j: $le\ numero\ du\ jour\ dans\ l'année.$

I. 2. 3. L'irradiation solaire

I. 2. 3. 1. Diffusion par l'atmosphère terrestre

La diffusion de la lumière est une redistribution spatiale du rayonnement par des particules matérielles, c'est un phénomène complexe puisqu'il intègre à la fois la diffraction, la réfraction et les réflexions par les particules, mais aussi et surtout leurs dimensions par rapport à la longueur d'onde de la lumière modifient sensiblement la répartition spectrale et l'intensité lumineuse.

Les nuages sont constitués de gouttelletes d'eau dont la taille est nettement superieure à la longuere d'onde, la diffusion est alors neutre et provient de l'ensemble des phenomenes de réflexion, refraction et diffraction. La nubulosité est le parametre meteorologique qui permet d'estimer l'importance prise part les nuages dans les phenomenes de diffusion. Elle s'exprime en huitiemes ou en fractions décimales de ciel couvert, cette notion n'etant que descriptive on la relie à la fraction d'insolation σ qui est une grandeur mesurable dés qu'on connait les durées d'isolation engistrées par l'héliographe.

$$\sigma = \frac{durée\ d'insolation}{durée\ du\ jour} \tag{I. 14}$$

I. 2. 3. 2. Nombre d'air mass

Prenant pour referance unité, l'epaisseur verticale de l'atmosphere moyenne reduite à 7.8 km dans des conditions dites normales, supposant cette couche plane et stratifiée hirizontalement donc la longure de trajet rectiligne d'un rayon lumineux incliné d'un angle h par rapport l'horizontale sera donnée par la formule [5]:

$$OM = \frac{OA}{\sin h} \qquad \text{(I. 15)}$$

A une pression P differente de 1013mbar et à une altitude z en kilometre, on désignera par masse atmosphérique ou nombre d'air mass le nombre :

$$m = \frac{p_m}{1013} \times \frac{1}{\sin h} exp\left(-\frac{z}{7.8}\right) \qquad \text{(I. 16)}$$

Ou p_m est la pression atmospherique au niveau de la mere, on trouvera par exemple :

- ✓ Soleil au zénith,niveau de la mere : AM 1 ;
- ✓ Soleil à 42° sur l'horizon : AM 1.5 ;
- ✓ Soleil à 30° sur l'horizon : AM 2.

I. 3. L'énergie solaire photovoltaïque

L'énergie solaire photovoltaïque est l'électricité produite par transformation d'une partie du rayonnement solaire dans une cellule photovoltaïque. Les cellules photovoltaïques sont fabriquées à partir de matériaux semi-conducteurs, comme le silicium, produits à partir d'une matière première de très grande pureté [2].

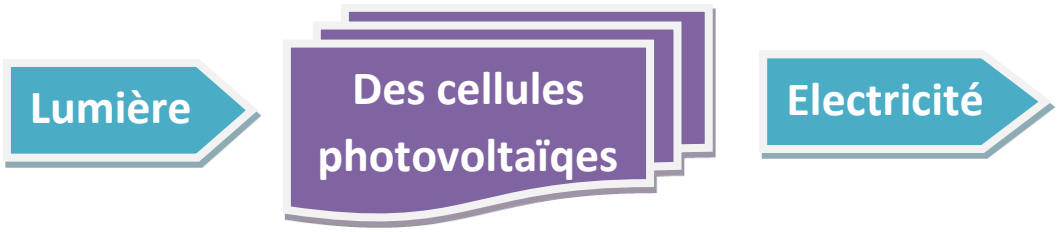


Figure I. 7 : Convention photovoltaïque.

I. 3. 1. Principes de fonctionnement d'une cellule photovoltaïque

Une cellule photovoltaïque est principalement constituée à partir de silicium dopé (jonction PN). Le phénomène photovoltaïque mis en œuvre est celui de l'interaction de la lumière composée des Photons avec un matériau semi-conducteur. Sous l'effet de la lumière, une tension électrique apparaît entre les deux côtés N et P de la photopile,

les électrons du silicium se mettent en mouvement uniquement quand ils entrent en contact avec les photons, cette énergie permet aux électrons de libérer par les atomes, générant ainsi des électrons des charges N et des trous dans la charges P. Ces charges sont alors maintenues séparées par un champ électrique qui constitue une barrière de potentiel [4].

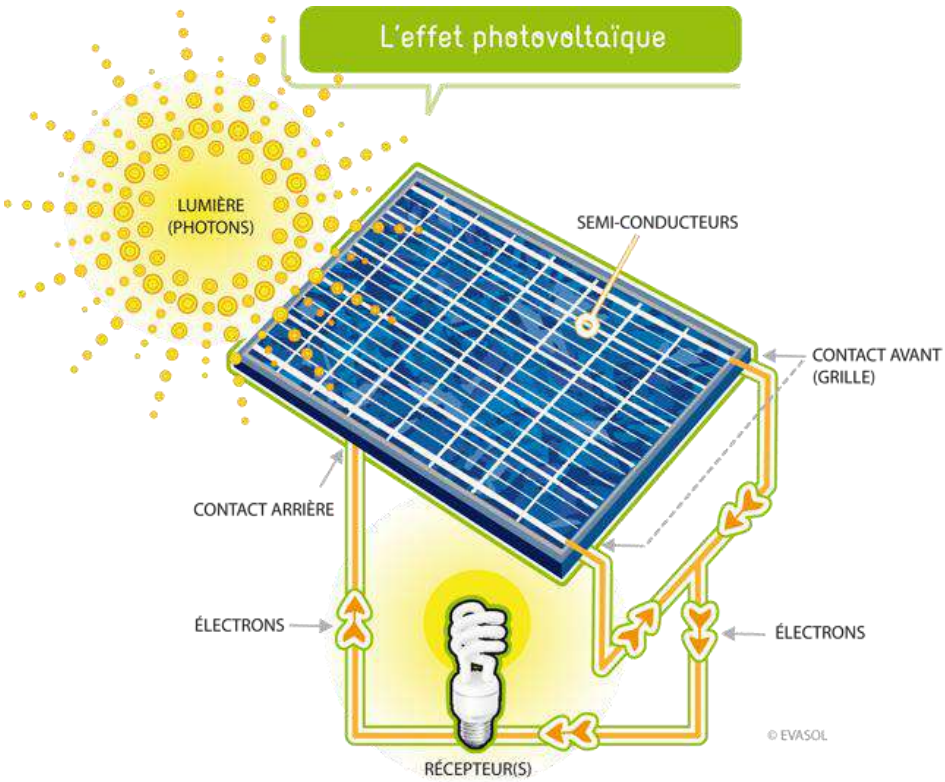


Figure I. 8 : L'effet photovoltaïque.

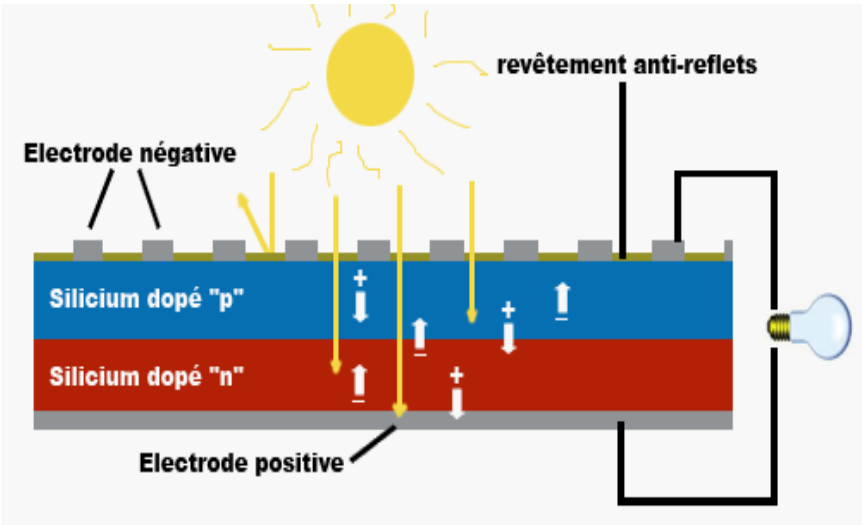


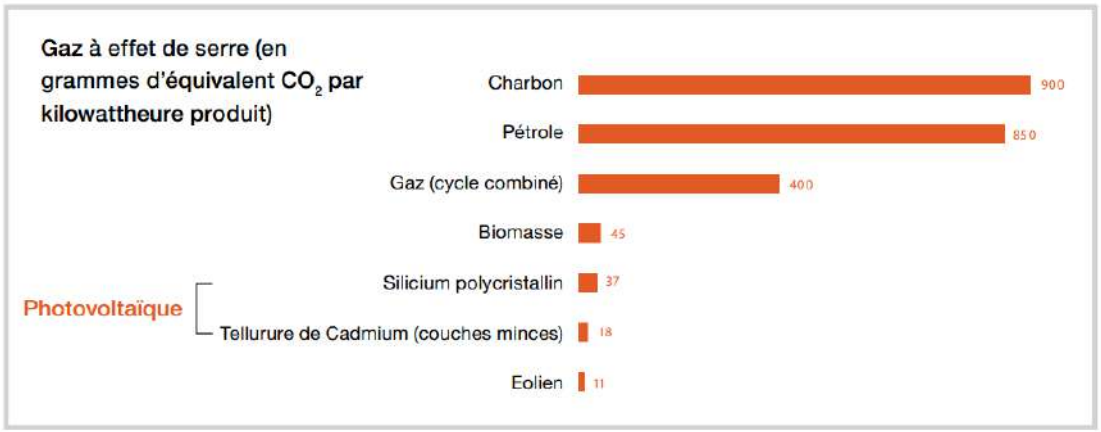
Figure I. 9 : Composant d'une cellule photovoltaïque.

L'impact des nombreux photons des rayons du soleil, produisent ainsi un courant continu. Le dispositif devient générateur électrique. L'effet photovoltaïque réalisé permet de convertir l'énergie du rayonnement solaire (photons) en énergie électrique à condition de collecter les électrons produits par un récepteur.

I. 3. 2. Avantages et inconvénients de l'énergie solaire photovoltaïque

I. 3. 2. 1. Les avantages

- ✓ Son carburant est gratuit : Le soleil est une source d'énergie propre et renouvelable, qui ne produit ni le gaz ni le déchet toxique par son utilisation.
- ✓ L'énergie solaire photovoltaïque ne génère ni bruit ni émissions nocives, ni gaz polluants.



Source : Européenne photovoltaïque industrie 2010

Figure I. 10 : L'émission de CO_2 dans chaque source d'énergie.

- ✓ Les systèmes photovoltaïques sont très sûrs et d'une grande fiabilité : l'espérance de vie d'un module solaire est d'environ 30 ans. La performance des cellules photovoltaïques est généralement garantie par les fabricants pour une durée de 20 à 25 ans.
- ✓ Les modules photovoltaïques sont recyclables et les matériaux utilisés pour leur production (silicium, verre, aluminium, etc.) peuvent être réutilisés
- ✓ Les modules solaires ne nécessitent pratiquement aucune maintenance et sont faciles à installer.

- ✓ L'énergie solaire photovoltaïque fournit de l'électricité aux zones rurales les plus isolées, Les systèmes de télécommunications en zones isolées utilisent également des panneaux solaires
- ✓ L'énergie solaire photovoltaïque peut être intégrée de manière esthétique dans les bâtiments

I. 3. 2. 2. Les inconvénients

- ✓ Les frais de production sont trop élevés : par rapport au rendement obtenu, en grande partie en raison du coût élevé des matières semi-conductrices employées dans le bâtiment et à l'installation nécessaire pour assurer un bon niveau d'efficacité.
- ✓ Lorsque le stockage de l'énergie électrique sous forme chimique (batterie) est nécessaire, le coût du générateur photovoltaïque est accru. La fiabilité et les performances du système restent cependant équivalentes si la batterie et les composantes de régulations associées sont bien choisies.
- ✓ Le rendement électrique diminue avec le temps (20% de moins au bout de 20 ans).
- ✓ Les générateurs photovoltaïques ne sont compétitifs par rapport aux générateurs Diesel que pour des faibles demandes d'énergie en région isolée.
- ✓ La taille des installations : il faut en effet de grandes superficies de panneaux solaires pour produire de l'énergie.

I. 4. Le gisement solaire en Algérie

Le gisement solaire est un ensemble de données décrivant l'évolution du rayonnement solaire disponible au cours d'une période donnée. Son évolution peut se faire à partir des données de l'irradiation solaire globale. Elle est utilisée pour simuler le fonctionnement probable d'un système énergétique solaire et donc faire le dimensionnement le plus exact possible pour satisfaire nos besoins.

La situation géographique est très importante pour développer, exploiter et mettre le rayonnement solaire à notre service pour une énergie renouvelable propre.

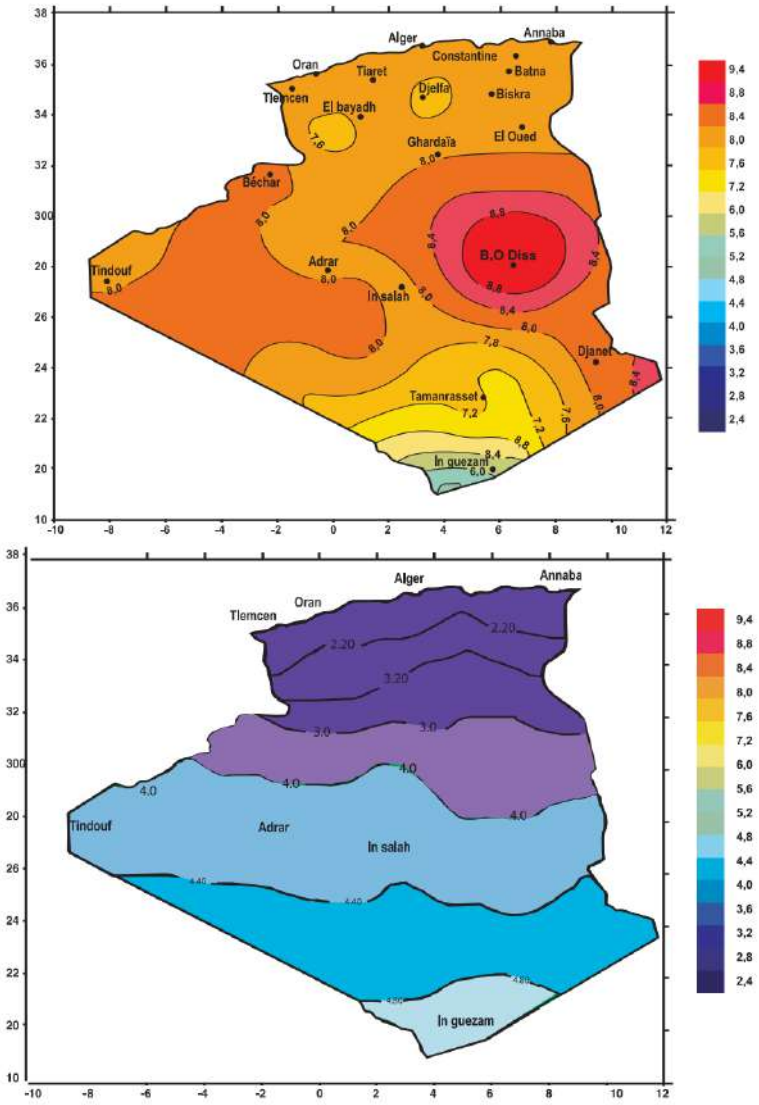


Figure I. 11 : Irradiation globale journalier reçu sur plan horizontal au mois de juillet et décembre.

L'Algérie parmi les pays riches du pétrole et gaz, elle porte un potentiel solaire très important de tout le bassin méditerrané. Avec 3500 heures d'ensoleillement par année elle présente le meilleur endroit au développement en énergie renouvelable.

Un programme national de développement des énergies renouvelables a été tracé pour la période 2011 - 2030, ambitionnant, à terme, de produire 40% de la consommation nationale d'électricité à partir des filières solaire et éolien, Et le bon exemple dans le cadre de ce développement la station hybride à Hassi R'mel avec une puissance de 150 MW qui permet d'économiser 7 millions de mètres cubes de gaz pour d'autre application. Et la puissance d'Algérie en énergie solaire avec son potentiel équivaut à huit fois les réserves du gaz. [2]

Tableau. I. 1. Potentiel solaire en Algérie. [2]

Régions	Cote	H. plateaux	Sahara
Superficies (%)	4	10	86
Durée moyenne d'ensoleillement (heure/an)	2650	3000	3500
Energie moyenne reçu (KWh/m2/an)	1700	1900	2650

La durée d'insolation sur la quasi-totalité du territoire national dépasse les 2000 heures annuellement et atteint les 3900 heures (hauts plateaux et Sahara). L'énergie reçue quotidiennement sur une surface horizontale de $1m^2$ est de l'ordre de 5 KWh sur la majeure partie du territoire national, soit près de 1700 KWh/m^2/an au Nord et 2263 KWh/m^2/an au sud du pays.

I. 6. Conclusion

L'énergie solaire est une énergie relativement chère et qui, de ce fait, reste encore peu développée, elle a l'avantage de permettre la réalisation d'importantes économies d'énergie, mais c'est aussi une énergie propre qui ne produit pas de déchets toxiques et qui ne dégage pas de gaz à effet de serre. Par ailleurs, les technologies de l'énergie solaire domestique tendent à se répandre et sont particulièrement efficaces et prouvées. Car l'énergie solaire est toutefois une solution d'avenir qui vaut le coup d'être plus exploitée, d'autant plus que de nombreuses améliorations sont encore possibles. Le Sahara algérien a des bonnes conditions pour l'investissement en domaine d'énergie solaire.

Chapitre II

Les moteurs pas à pas

et les suiveurs solaire

II. 1. Introduction

Les moteurs pas à pas est un convertisseur électromécanique ayant pour fonction la transformation d'une énergie électrique en une action mécanique pouvant être un déplacement linéaire ou angulaire. [7]

Ils sont très courants dans tous les dispositifs où l'on souhaite faire du contrôle de vitesse ou de position en boucle ouverte. [8]

L'usage le plus connu du grand public est dans les imprimantes et les scanner, mais ils sont présents dans de nombreuses applications telles : les photocopieurs, robotique, instrumentation, système de positionnement sur machine industrielle

Dans ce chapitre, nous présentons le principe de fonctionnement d'un moteur pas à pas, leur comportement et leurs différents types, puis nous comprenons la partie commande qui nous permettent de réaliser une carte de commande pour faire fonctionner les moteurs pas à pas que nous utilisons dans ce projet.

On va voir aussi les différents types de suiveurs solaires et leurs commandes.

II. 2. Les moteurs pas à pas

Les moteurs pas à pas (stepping motor en anglais) constituent un convertisseur électromécanique destiné à transformer le signal électrique (impulsion ou train d'impulsions de pilotage) en déplacement (angulaire ou linéaire) mécanique, et qui assurent simultanément deux fonctions :

- la conversion d'énergie électrique en énergie mécanique,

- la conversion d'information électrique digitale en information mécanique de nature incrémentale.

Au point de vue électrotechnique, le moteur classique ressemble à la machine synchrone, dont le stator (le plus souvent à pôles saillants) porte les enroulements de pilotage et le rotor (presque toujours à pôles saillants) est soit muni d'aimants permanents (structure dite polarisée ou active), soit constitué par une pièce ferromagnétique dentée (structure dite réluctance ou passive).



Figure II. 1 : Différents tailles des moteurs pas à pas.

Suivant leur construction, le moteur pas à pas avance de 90°, 45°et 18° ou d'une fraction de degré par impulsion. En faisant varier la fréquence des impulsions, on peut faire tourner le moteur lentement ou rapidement à des vitesses aussi élevées que 4000 rad /min.

A cause de sa grande précision de positionnement, le moteur pas à pas est utilisé dans les machine outils, les impriment et les traceurs suiveurs de soleil …etc.

II. 2. 1. Caractéristiques générales d'un moteur pas à pas

Malgré la différence existent entre les moteurs pas à pas, Ce qui est nécessaire est l'avance d'un seul pas, c'est-à-dire la rotation de leur axe suivant un angle déterminé à chaque impulsion que l'une ou l'autre de leurs différentes bobines recevra, cet angle, qui varie selon la constitution interne du moteur, est en général compris entre 0.9° et 90°, les moteurs sont plus couramment rencontrés sont :

- 0.9° soit 400 pas par tour
- 1.8° soit 200 pas par tour
- 3.6° soit 100 pas par tour
- 7.5° soit 48 pas par tour
- 15° soit 24 pas par tour

Il est évident que les moteurs pas à pas, de leur technologie, présentent une très grande précision et une durée de vie quasi illimitée. [9]

Les moteurs pas à pas existent en différents tailles qui varient environ 1 cm et plus d'une dizaine de centimètres, tout dépendra des applications dans lesquelles ils seront utilisés, le plus petit moteur sera destiné au déplacement des têtes de lecture de disquette, ou un couple très faible est requis, par contre, le déplacement du bras d'un robot demandera un couple nettement plus important, donc un moteur de diamètre élevé.

Signalons que le couple est exprimé le plus souvent en kilogrammes par centimètre ($kg.cm^{-1}$), ce qui définit le poids en kilogrammes que pourra soulever l'axe d'un moteur pourvu d'un bras de longueur exprimée en centimètres. [9]

La valeur de leur tension d'alimentation varie dans de grandes proportions, elle peut être comprise entre 3V et plusieurs dizaines de volts, de même, selon la résistance ohmique de leurs bobinages, le courant consommé s'étendra dans une gamme allant de quelques dizaines de milliampères à plusieurs ampères, on dit que plus le courant sera élevé, plus le couple sera important.

Les moteurs de type standard et de consommation moyenne sont possédant un couple de puissance et possédant un couple de puissance relativement bas, ce faible couple pourra être compensé par l'adjonction des pignons, la vitesse de rotation diminuera mais le couple augmentera dans de fortes proportions. [9]

II. 2. 2. Les différents types de moteur pas à pas

Les moteurs pas à pas sont identiques dans leurs fonctionnements, ils diffèrent cependant dans leur constitution et leur domaine de travail, ils peuvent être rangés en trois catégories constructives.

• les moteurs à aimant permanent
• les moteurs à reluctance variable
• les moteurs hybrides

II. 2. 2. 1. Les moteurs à aimant permanent

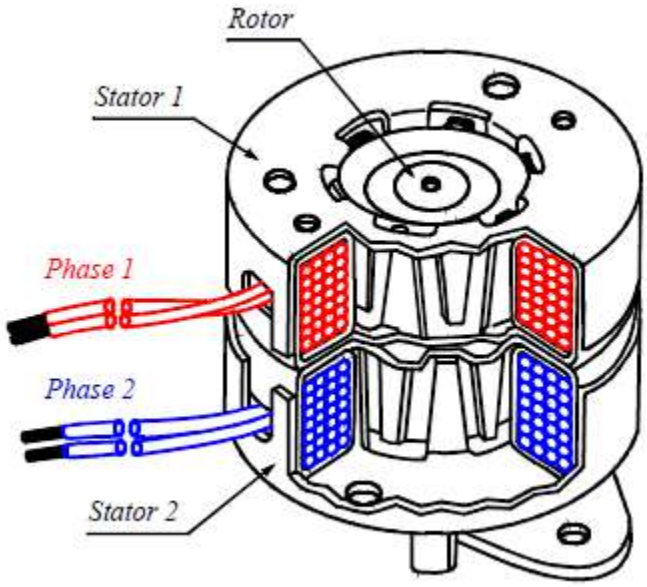


Figure II.2 : Moteur à aimants permanents.

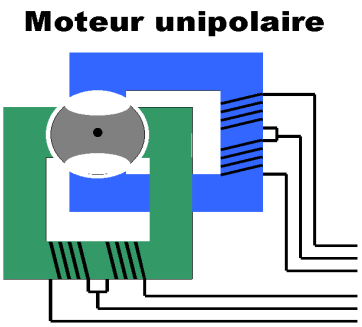
Le rotor comporte des aimants permanents radiaux et le stator est constitué de bobinages. Lorsque chaque phase statorique est alimentée séquentiellement, le flux statorique réagit avec le flux rotorique des aimants ce qui tend à aligner les deux flux le long d'un axe de réluctance minimum (Loi d'attraction des pôles de noms contraires).

Le rotor se met en mouvement rotatif pas à pas pour suivre l'alimentation successive des phases statoriques.

Ces moteurs sont assez efficaces, souples et délivrent un couple relativement important même lorsque le moteur n'est plus alimenté, mais le nombre des pas par tour est faible, cette catégorie de moteur se subdivise en deux types :

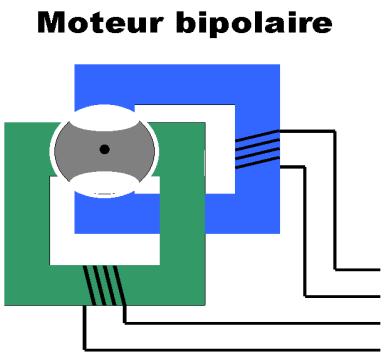
A) Moteur unipolaire

Le rotor est constitué par un aimant permanent (en ferrite par exemple) comportant une paire de pôles. Le stator, comprend deux circuits magnétiques décalés de 90°. Les enroulements à point milieu sont alimentés avec une polarité toujours du même signe, d'où le nom d'unipolaire.



Pour inverser le sens de rotation il suffit d'inverser les séquences de commutation

b) Moteur bipolaire



Les enroulements au stator n'ont pas de point milieu, chaque borne de chaque enroulement est alimentée successivement par une polarité positive puis négative (d'où le terme bipolaire).

En inversant les polarités des enroulements statoriques, on inverse les pôles. Le nombre des phases est égale au nombre d'enroulements.

II. 2. 2. 2. Les moteurs à reluctance variable

Ils sont considérés comme les moteurs de base des moteurs pas à pas. Le couple moteur est dû à la variation de la réluctance du circuit magnétique de la machine.

Le moteur à réluctance variable a été mis pour permettre des pas plus petits. On obtiendra avec ces moteurs des fréquences de travail élevées. Ces moteurs existent sous forme de plusieurs structures ; moteurs à simple circuit magnétique et moteur à plusieurs circuits magnétiques.

A) Moteur reluctance a simple circuit

Ce moteur est plus compact. Sa structure est simple. Il a un seul groupe de dents sur le rotor et le stator. Le stator est construit en tôles feuilletées, alors que le rotor est massif (pour réduire l'inertie). Son pas angulaire peut prendre pour chaque moteur une valeur entre 1.8° et 30°. Dans ces moteurs le nombre des dents dans les pôles statoriques et rotoriques sont différents [9].

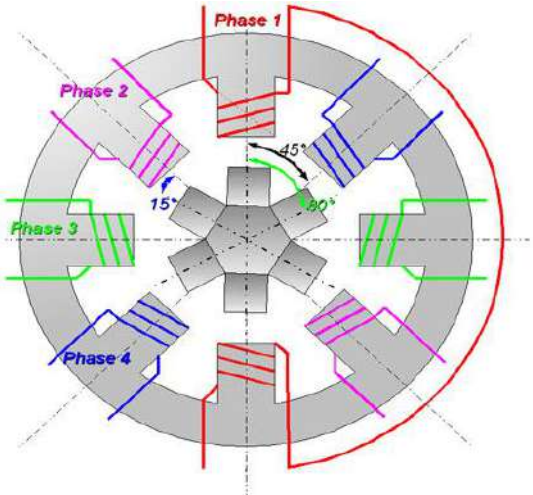


Figure II.3 : Moteur à réluctance variable (MRV) a circuit magnétique simple.

B) Moteur reluctance multi-circuit

Ces moteurs sont les plus anciens, ils ont une construction multi-circuit. Ils sont moins utilisés aujourd'hui à cause de leur complexité de construction. L'avantage de cette structure permet d'éliminer toute inductance mutuelle entre phases statiques. La disposition du rotor peut se faire de deux façons différentes :

- Les stators sont en phase et les rotors sont régulièrement décalés.
- Les rotors sont positionnés de manière identique et les stators sont décalés.

II. 2. 2. 3. Les moteurs pas à pas hybride

Le moteur pas à pas hybride ou moteur réluctant polarisé possède l'avantage d'associer le phénomène de la réluctance variable à une excitation par aimant permanent,

On superposant le principe du moteur à réluctance variable et à aimant permanent on obtient un moteur hybride qui combine les avantages des deux moteurs, d'où le nom hybride.

Le rotor est constitué de deux disques dentés décalés mécaniquement. Entre ces deux disques est inséré un aimant permanent. Le nombre de dents du rotor est différent de celui du stator. Lorsqu'on alimente une bobine, le rotor place les dents Nord et Sud de telle façon à ce que le flux traversant le rotor soit maximal.

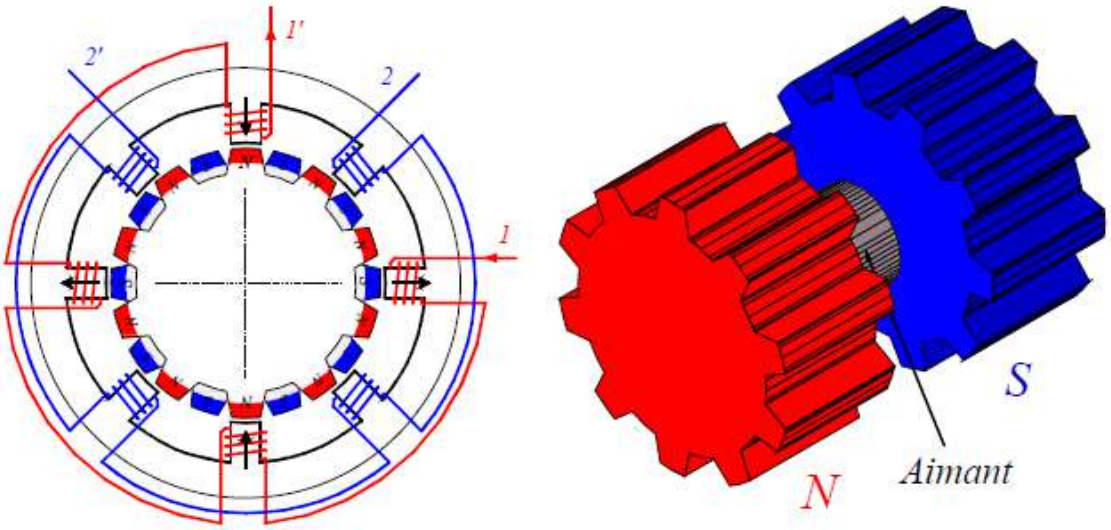


Figure II.4 : le moteur pas à pas hybride.

On peut citer quelques caractéristiques de moteur pas à pas hybride:

- un couple moteur élevé et proportionnel au courant.

- un couple de détente.
- un nombre de pas élevé ce qui permet d'éviter l'utilisation du réducteur.
- un couple massique élevé.
- une précision de positionnement.

A cause de ces avantages, le moteur pas à pas hybride domine le marché des applications de l'actionneur pas à pas. Pour cette raison nous l'avons choisi dans notre application.

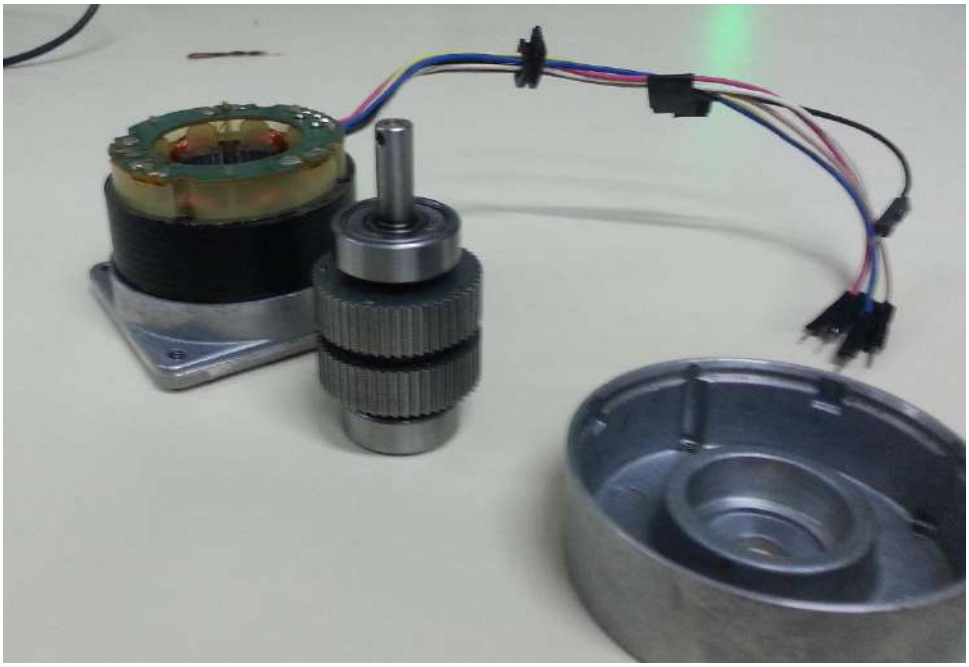


Figure II.5 : le moteur pas à pas hybride utilise dans le montage de suiveur.

Tableau II. 1 : Étude comparative de différents types de moteur pas à pas

Type de moteur pas à pas	***Moteur à réluctance variable***	***Moteur à aimants permanents***	***Moteur hybride***
résolution (nb de pas par tour)	Bonne	Moyenne	Elevée
Couple moteur	Faible	Elevée	Elevée
Sens de rotation	Dépend : *- de l'ordre d'alimentation des phases*	Dépend : *- de l'ordre d'alimentation des phases* *- du sens du courant dans les bobines*	Dépend : *- de l'ordre d'alimentation des phases* *- du sens du courant dans les bobines*
Fréquence de travail	Grande	Faible	Grande
Puissance	Quelques Watts	Quelques dizaines de Watts	quelques Kilo Watts
Inconvénients	Pas de mémoire de position		

II. 3. La commande d'un moteur pas à pas par un Arduino

II. 3. 1. La connexion d'un moteur unipolaire

Pour rappel, voici la structure du moteur unipolaire :

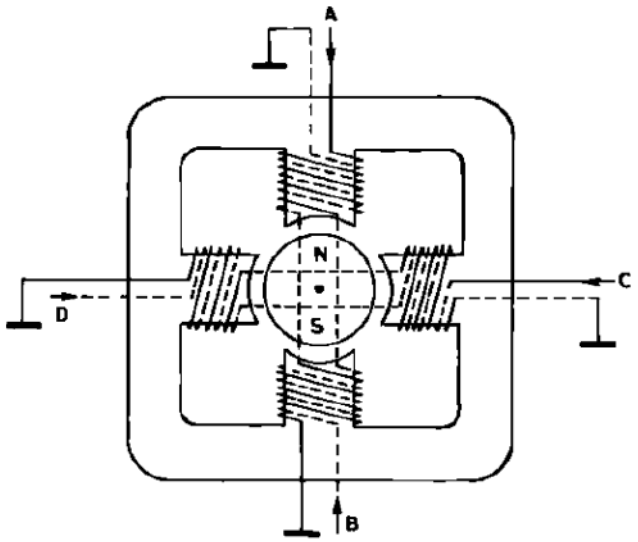


Figure II. 6 : Schéma d'un moteur unipolaire.

On a vu dans la partie précédente que ce moteur est assez simple à piloter. En effet, il suffit d'alimenter une à une les bobines pour que le moteur tourne. Il nous faut juste utiliser le bon composant pour alimenter les bobines. A priori, le bon composant serait un bon transistor qui pourrait supporter 50V et 500mA pour débuter. Pour cela il faudrait ensuite rajouter une diode de roue libre pour ne pas l'abîmer lors des phases de roue libre (tout cela multiplié par 4 puisqu'on a 4 bobines). Plutôt que de s'embêter à câbler tout ça, Le ULN2003A. Ce dernier regroupe les transistors pour faire passer la puissance et aussi la diode. Il est très simple à câbler puisqu'il faut simplement amener l'alimentation et les broches de commandes. Chacune de ces dernières possède respectivement une sortie où la tension sera celle de l'alimentation. Une illustration de ce câblage dans figure III.7.

Les schémas qui vont suivre ont pour source d'énergie une pile +9V. Cependant il est déconseillé de les faire avec car la consommation des moteurs est assez importante et la pile risque de se fatiguer très vite. Utilisez plutôt une source d'alimentation prévue à cet effet (une alimentation de laboratoire).

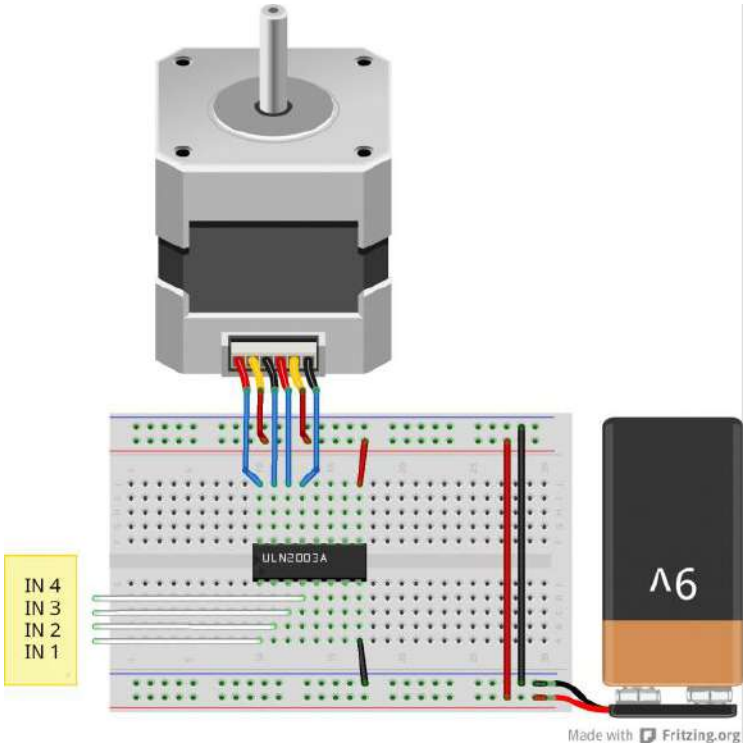


Figure II. 7 : Schéma de câblage d'un moteur unipolaire. [***Fritzing***]

A. Principe de commander un moteur pas à pas unipolaire par Arduino

Pour ce moteur il suffit de piloter chaque bobine une à une, chacune son tour. On va donc résumer tout cela de manière plus schématique. A la fin, si vous avez bien compris vous devriez être capable de le faire tourner tout doucement en plaçant alternativement les fils In1 à 4 en Arduino.

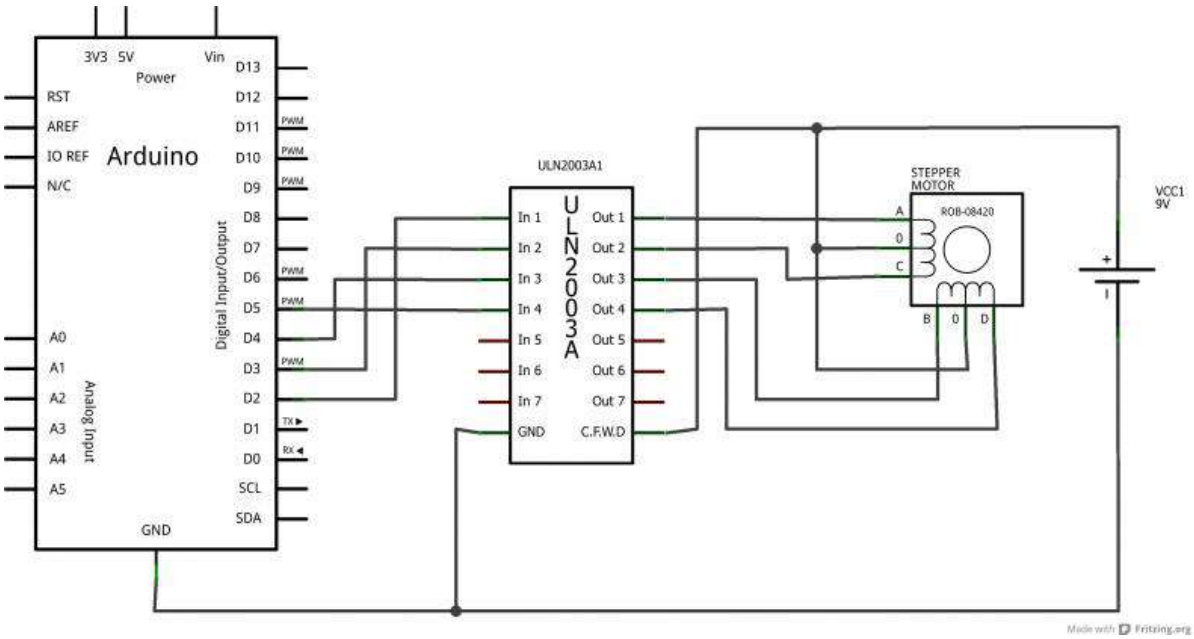
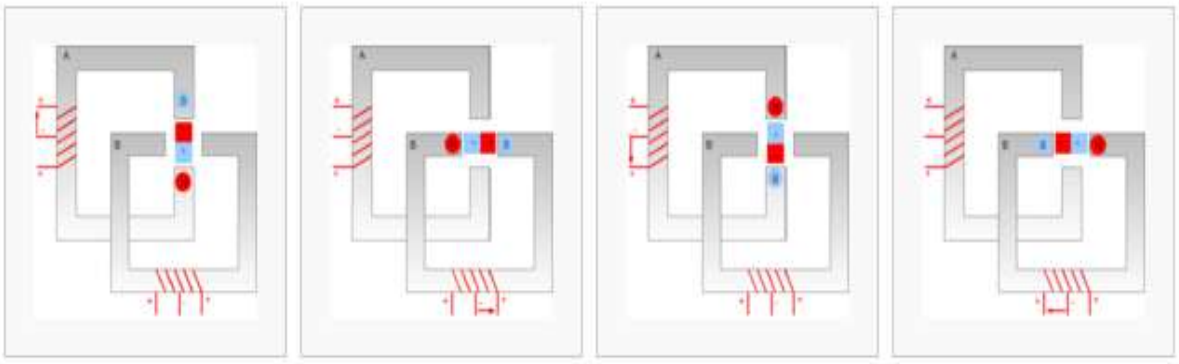


Figure II. 8 : Schéma de câblage d'un moteur unipolaire avec Arduino et ULN2003.

L'idée est toute simple, il suffit de générer la bonne séquence pour piloter les moteurs à la bonne vitesse, vous vous en doutez surement. La principale difficulté réside dans la génération des signaux dans le bon ordre afin que le moteur se déplace correctement. Bien entendu, c'est plus facile à dire qu'à faire. En effet, pour que le mouvement soit fluide, il faut que les changements dans la séquence soient fait de manière régulière et pour cela il faut une gestion du temps correcte. Ça peut sembler simple au premier abord, mais quand il s'agira de mixer le comportement du moteur avec celui du programme principal (qui devra peut-être faire des traitements assez lourd) cela deviendra probablement beaucoup moins trivial. Une bonne méthode consisterait à utiliser un **timer** sur lequel on réglera la période à avoir qui refléteraient ainsi la vitesse à obtenir. Mais avec Arduino vous allez voir que tout devient plus simple.



Etape	Ln1	Ln2	Ln3	Ln4
Pas n° 1	HIGH	LOW	LOW	LOW
Pas n° 2	LOW	HIGH	LOW	LOW
Pas n° 3	LOW	LOW	HIGH	LOW
Pas n° 4	LOW	LOW	LOW	HIGH

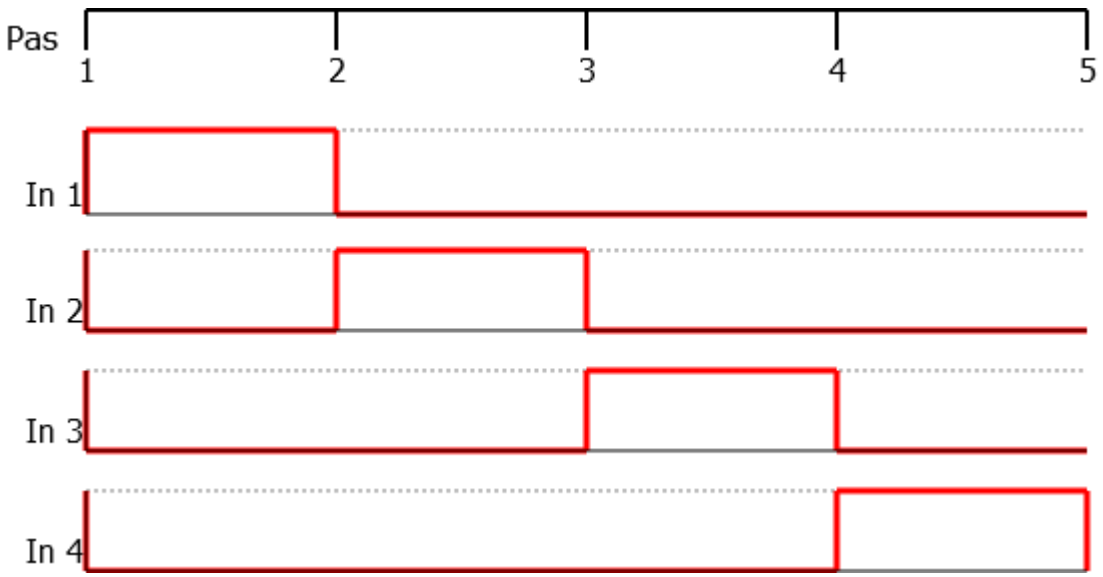


Figure II. 9 : les séquences d'Arduino pour piloter les moteurs unipolaires.

Sur Arduino les choses sont bien faites pour rester simples et accessibles, un objet a déjà été créé pour nous aider à piloter un moteur pas à pas. Attention cependant, il ne fonctionne que pour les moteurs unipolaire et bipolaire. Il tire parti du fait que ces deux types de moteur peuvent fonctionner avec une séquence commune. Ainsi, tout est généralisé et utilisable le plus simplement possible. Ce nouveau composant s'appelle *"Stepper"* en Arduino. A sa création il prend en argument le nombre de pas total que fait le moteur pour faire un tour (information trouvable dans la documentation constructrice ou empiriquement). Cette information sert à déterminer la vitesse de rotation par minute que vous pourrez ensuite régler à loisir pour faire des déplacements lents ou rapides. Il prend aussi en arguments les quatre broches servant à contrôler l'engin. Son constructeur est donc : *Stepper (steps, pin1, pin2, pin3, pin4)*. Pour initialiser le moteur, nous pouvons donc écrire la ligne suivante :

```
1   //pour un moteur de 200 pas par tour et brancher sur les broches 2, 3, 4, 5
2   Stepper moteur (200, 2, 3, 4, 5);
```

Pour l'utiliser, deux fonctions sont utilisables. La première sert à définir la vitesse de rotation, exprimée en tours par minute (trs/min). Pour cela, on utilise la fonction *step(steps)* qui prend en paramètre le nombre de pas à effectuer. Si ce nombre est négatif, le moteur tournera en sens inverse du nombre de pas spécifié. Voici un petit exemple qui va faire faire un aller-retour de 200 pas toute les 2 secondes à votre moteur :

```
2    Stepper moteur (200, 2, 3, 4, 5);
3    void setup()
4    {
5        moteur.setSpeed(30); //30 tours par minute
6    //(rappel : ici le moteur fait 200 pas par tour, on fera donc 6000 pas
7    par minute)
8    }

9    void loop()
10   {
11       moteur.step(1000);
12       delay(100);
13       moteur.step(-1000);
14       delay(2000);
15   }
```

La fonction *step(x)* est bloquante. Cela signifie qu'elle agit comme un délai. Tant que le moteur n'a pas fait les x pas demandés, le reste du programme est en attente.

II. 3. 2. La connecter d'un moteur pas à pas bipolaire

Nous avons vu dans le chapitre précédent que le moteur bipolaire a besoin, à un moment, d'une inversion de courant si l'on veut pouvoir utiliser les bobines à bon escient. Vous vous souvenez probablement que nous avons vu précédemment un composant capable de faire cela : le pont en H. L'idée va donc être de câbler correctement les bobines pour pouvoir faire passer le courant dans un sens, ou dans l'autre. Je vous rappelle la structure du pont en H :

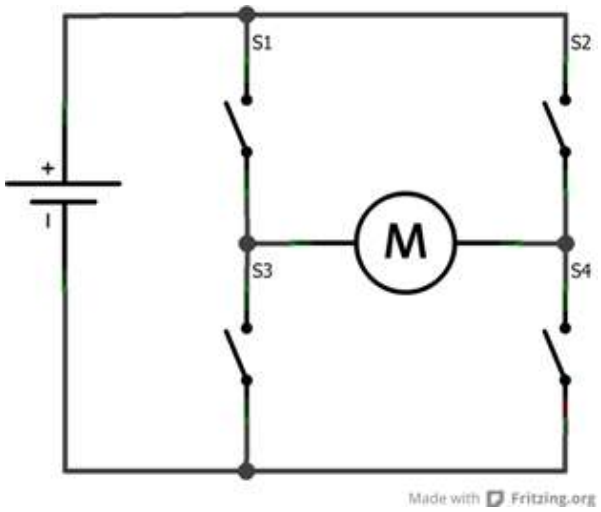


Figure II. 10 : Schéma d'un pont H.

Un pont en H ne peut piloter qu'une "paire de bobine" (celle qui sont dans le même axe et relié entre elles), hors nous en avons deux à piloter. Heureusement le L298 est un composant très bien pensé et ne possède non pas un mais bien deux ponts en H.

Dans le figure II.10, nous avons justement vu un composant qui possède deux ponts en H : le L298. Le reste du travail semble presque trop simple. Pour la partie "sortie/puissance", On doit relier les sorties *Out1 (broche 2)* et *Out2 (broche 3)* à la première bobine et ensuite *Out3* et *Out4* (13 et 14) à la seconde. N'oubliez pas les diodes de roue libre (8 au total et qui supporte la puissance). Enfin, il ne reste qu'à connecter les entrées In1 à ln4 entrée/sortie numérique (pas besoin de PWM). Voici un petit schéma récapitulatif :

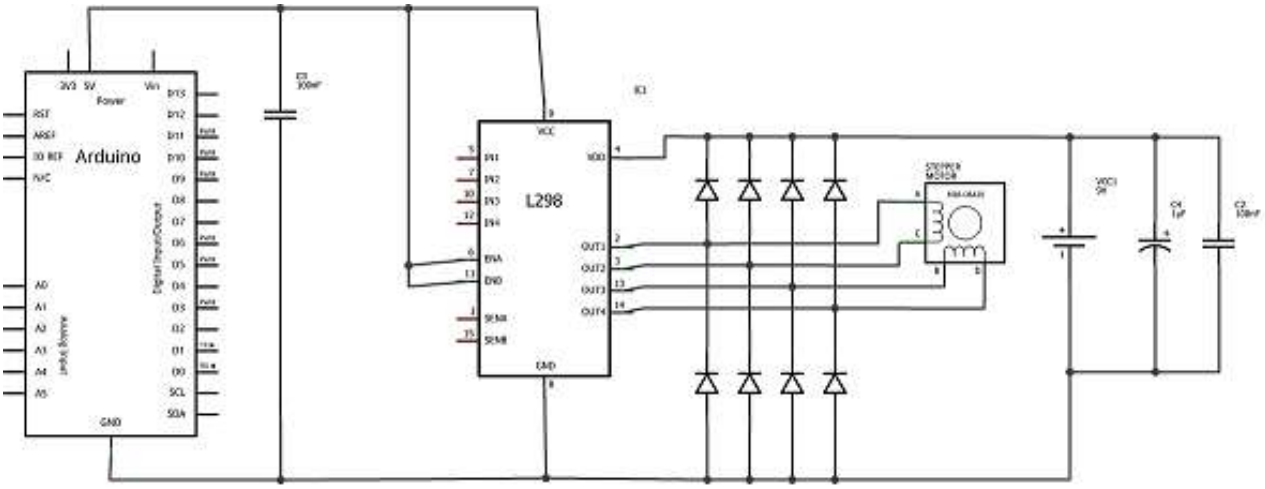


Figure II. 11 : Schéma de pont H L298 avec Arduino.

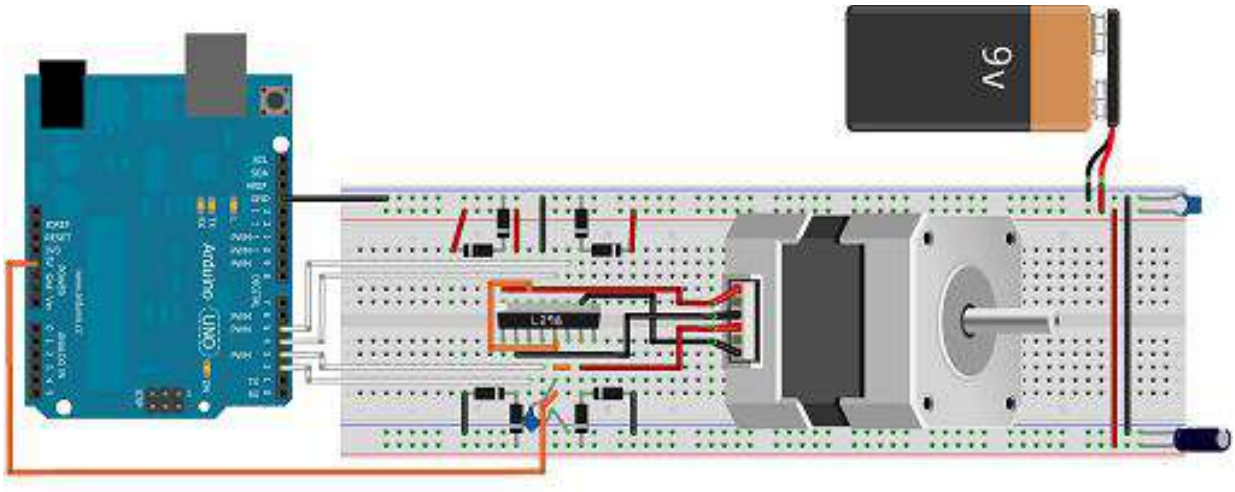


Figure II. 12 : Schéma de pont H L298 avec Arduino.

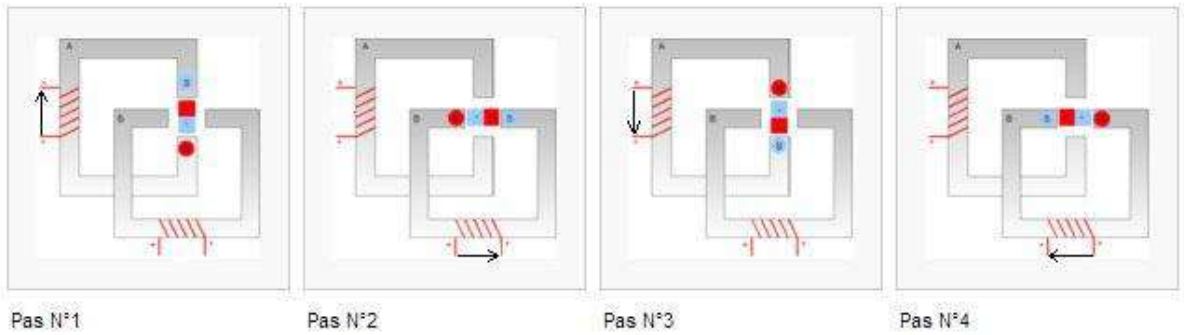
Le schéma de montage avec quelques condensateurs de filtrage qui viennent aider l'alimentation en cas de forte demande de courant et des condensateurs qui filtre les parasites engendrés par les bobines du moteur.

Une fois que le moteur est branché, nous allons pouvoir le faire tourner. Comme son nom l'indique, il s'appelle pas à pas et donc nous allons le faire pivoter étape par étape et non de manière continue comme le moteur à courant continu. Cependant, en répétant les étapes de rotation rapidement et successivement, le moteur donnera l'impression de tourner sans s'arrêter entre chaque étape. Il existe trois méthodes distinctes pour piloter les moteurs bipolaires. Nous allons maintenant les voir une par une. Dans les cas qui vont suivre, on va considérer un moteur de 4 pas par tour).

A). Rotation par pas complet

Ce mode de fonctionnement est le plus simple, c'est pourquoi nous allons commencer par lui. Grâce à ce dernier, vous allez pouvoir faire des rotations "pas par pas". Pour cela, nous allons alternativement alimenter les bobines de droites et de gauche et inverser le sens du courant pour pouvoir faire une rotation complète du champ magnétique.

Le sens du courant symbolisé par une flèche et en noir la bobine qui n'est pas alimentée, En considérant que la bobine A est connectée aux entrées IN1 et IN2 et que la bobine B est connectée aux commandes IN3 et IN4, on peut donc écrire la séquence de commande suivante :



Etape	Ln1	Ln2	Ln3	Ln4
Pas n° 1	HIGH	LOW	-	-
Pas n° 2	-	-	HIGH	LOW
Pas n° 3	LOW	HIGH	-	-
Pas n° 4	-	-	LOW	HIGH

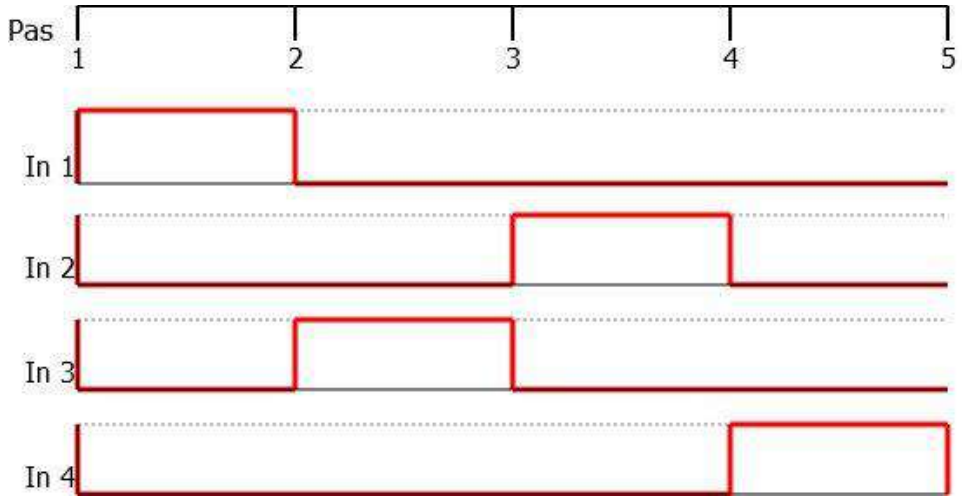
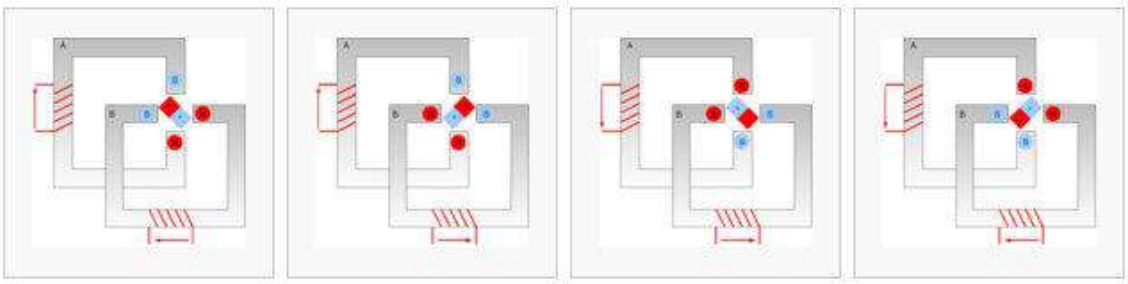


Figure II. 13 : les séquences d'Arduino pour piloter les moteurs bipolaires avec un pas complet.

Comme vous pouvez le voir à travers ces différents moyens d'explication, c'est somme toute assez simple. On va chercher à déplacer l'aimant central en le faisant tourner petit à petit. Pour cela on cherchera à l'attirer dans différentes positions.

B). Rotation à couple maximal

Un autre mode de fonctionnement est celui dit à couple maximal. Cette méthode de pilotage utilise toutes les bobines à la fois pour pouvoir immobiliser au maximum l'aimant central. En effet, en utilisant plus de champs magnétiques on obtient une force supplémentaire. Par contre, on consomme évidemment d'avantage de courant. Pour comprendre ce fonctionnement, voyons les différentes étapes par un dessin puis par un chronogramme. Vous verrez, ce n'est pas très compliqué, le fonctionnement est très similaire, seules les activations de bobines changent un peu :



Etape	Ln1	Ln2	Ln3	Ln4
Pas n° 1	HIGH	LOW	HIGH	LOW
Pas n° 2	HIGH	LOW	LOW	HIGH

Pas n° 3	LOW	HIGH	LOW	HIGH
Pas n° 4	LOW	HIGH	HIGH	LOW

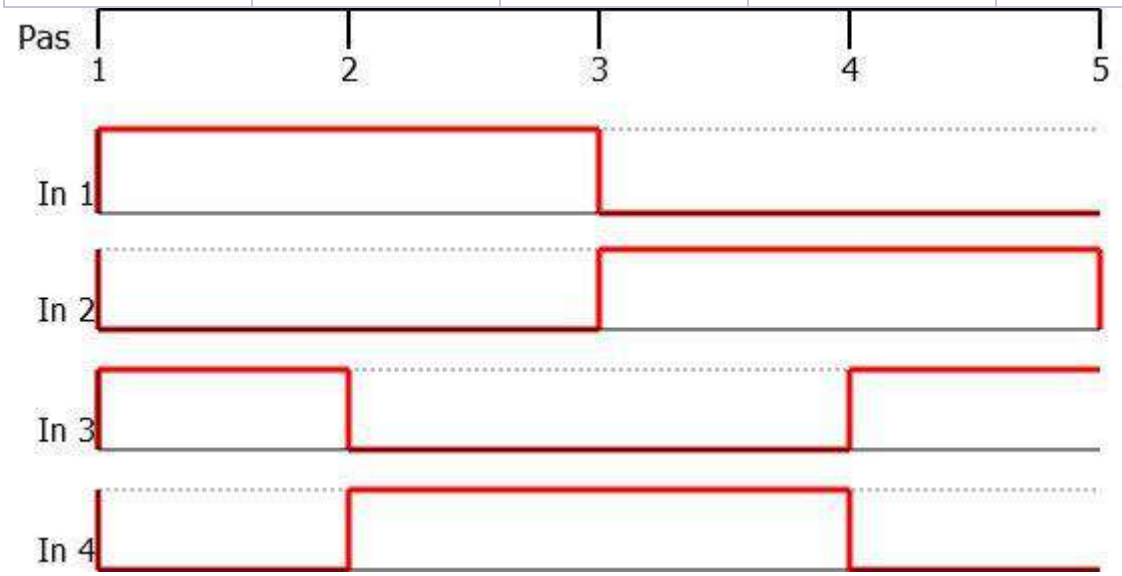


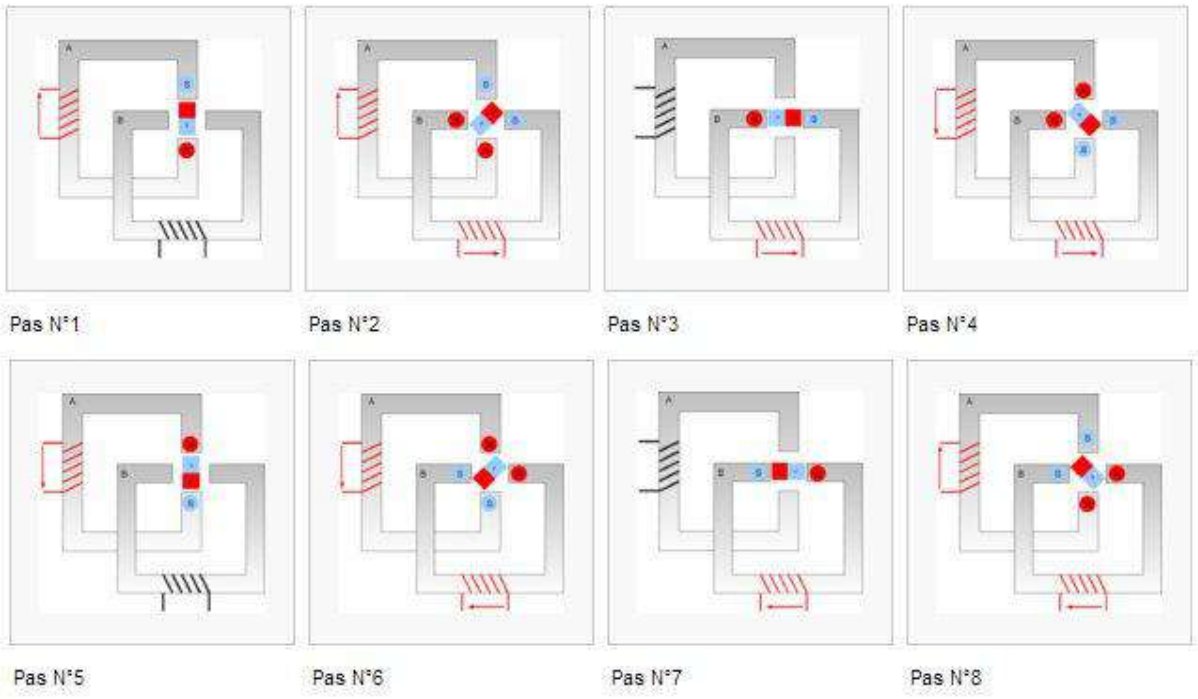
Figure II. 14 : les séquences d'Arduino pour piloter les moteurs bipolaires avec un couple max.

Avez-vous remarqué quelque chose de particulier ? Dans cette utilisation, l'aimant ne fait plus face aux bobines mais se place entre les deux. Par contre, il effectue toujours des pas entiers, ces derniers ont juste un décalage constant par rapport à avant.

C). Rotation par demi-pas

Le dernier mode de fonctionnement est celui dit à demi-pas. Ce mode mélange les deux précédents puisqu'on va alterner les étapes du mode à pas complet et les étapes du mode à couple maximal. En effet, comme nous avons pu le voir dans les explications précédentes, les deux modes placent l'aimant central de manière différente. L'un est "en face des bobines" alors qu'avec l'autre est plutôt "entre les bobines".

Ainsi, en se mettant alternativement "en face" puis "entre" les bobines on va effectuer deux fois plus de pas que précédemment puisqu'on intercalera des étapes supplémentaires. Attention, lorsque je dis "deux fois plus de pas" je veux surtout dire que l'on aura des étapes intermédiaires qui augmentent la précision du déplacement. Ce mode de pilotage est un peu plus compliqué que les précédents puisqu'il est "plus long" (8 étapes au lieu de 4) mais rien d'insurmontable.



Etape	Ln1	Ln2	Ln3	Ln4
Pas n° 1	HIGH	LOW	-	-
Pas n° 1 $^1/_2$	HIGH	LOW	HIGH	LOW
Pas n° 2	-	-	HIGH	LOW
Pas n° 2 $^1/_2$	LOW	HIGH	HIGH	LOW
Pas n° 3	LOW	HIGH	-	-
Pas n° 3 $^1/_2$	LOW	HIGH	LOW	HIGH
Pas n° 4	-	-	LOW	HIGH
Pas n° 4 $^1/_2$	HIGH	LOW	LOW	HIGH

Figure II. 15 : les séquences d'Arduino pour piloter les moteurs bipolaires à demi-pas.

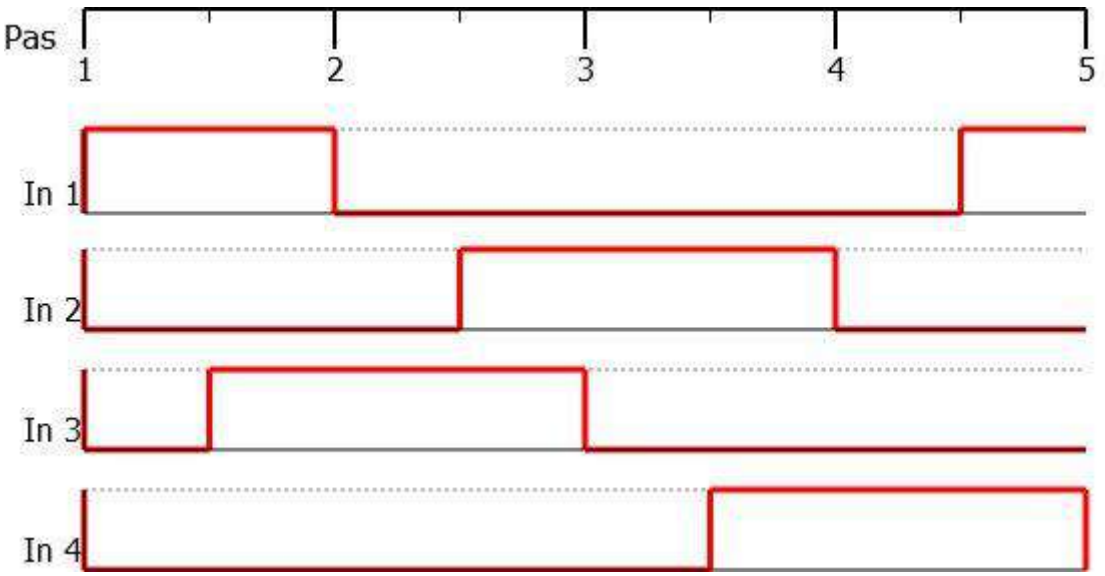


Figure II. 16 : Chronogramme du pilotage à demi-pas.

II. 4. Etude des différents systèmes de poursuite d'un suiveur solaire

La puissance électrique produite par un système solaire photovoltaïque dépend de l'intensité du flux lumineux selon la position du soleil qui varie le long de la journée. Afin de collecter le maximum d'énergie, on utilise des dispositifs dits systèmes de poursuite solaire *'Sun Tracking System'*. Le système de poursuite solaire doit être ajusté de sorte qu'il soit toujours visé avec précision face au soleil. Les systèmes de poursuite solaire à un seul axe sont moins coûteux et leur commande est facile à mettre en œuvre, par contre leur efficacité est inférieure à celle des systèmes de poursuite solaire à deux axes [10].

Il est possible d'identifier trois facteurs principaux permettant de classer les différents types de suiveurs solaires :

- Mécanisme d'orientation
- Degré de liberté de mouvement
- Type de commande

II.4.1. Mécanisme d'orientation

a) les capteurs passifs

Une option possible pour le suivi est un système chimique/mécanique, Ce système utilise l'idée de dilatation thermique des matériaux comme une méthode de poursuite comme le chlorofluorocarbone (CFC), habituellement les suiveurs passifs sont composés de deux actuateurs qui fonctionnent d'une manière opposée. Lorsque les deux actuateurs sont soumis à la même illumination le suiveur se trouve en repos perpendiculaire avec le soleil. Une fois que le soleil se déplace l'illumination se change entre les deux actuateurs le suiveur se déplace vers un nouveau point d'équilibre ou l'illumination devient égales pour les deux actuateurs. L'une des inconvénients de ce type qu'ils ne fonctionnent pas dans les basses températures et une efficacité de 23% et ils ne sont pas disponible commercialement [11].

La figure I.15 montre un suiveur de soleil passif présenté par Clifford et al. Ce suiveur consiste à utiliser deux tubes cylindriques identiques contenants un liquide sous une pression partielle. Ces tubes sont positionnés sur un cadre en bois, symétriquement par rapport à l'axe horizontale. Les deux tubes sont ombrés tels que le tube le plus éloigné du soleil absorbe la radiation solaire alors que le deuxième reste ombré ce qui

provoque le mouvement du système, pour éviter une réponse excessive et les oscillations on équipe le suiveur par système d'amortissement [12].

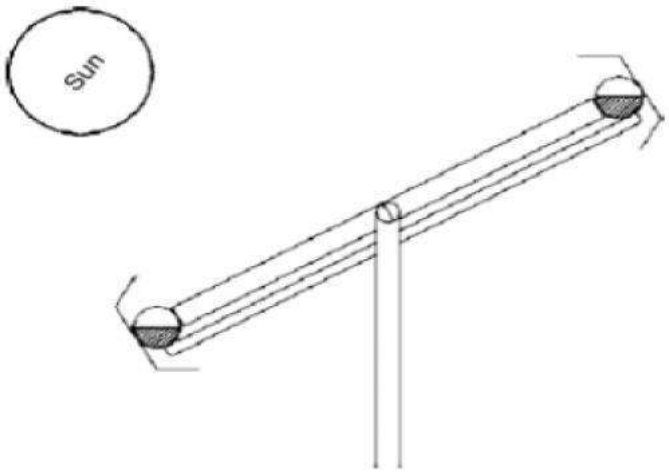


Figure II. 17 : suiveur solaire passif.

b) les capteurs actifs

Il y'a trois types principales de suiveurs actifs : suiveur a une cellule solaire bifaciales auxiliaire, suiveur électro-optiques, et suiveurs à base de microprocesseur.

Suiveur a une cellule solaire bifaciales auxiliaire est le plus simple, la cellule solaire bifaciales auxiliaire qui agit comme un capteur de lumière est fixé sur l'axe de rotation, elle est placée perpendiculaire sur le panneau solaire, la cellule est connectée directement à un moteur à courant continue, lorsque le soleil déplace, l'angle d'incidence augmente sur la cellule, ce qui produit une énergie capable à faire tourner le moteur qui tourne par son tour le panneau solaire. La figure II.18 montre le schéma de ce type de suiveur [11].



Figure II. 18 : suiveur de soleil a une cellule bifaciale.

Les suiveurs électro-optiques sont aussi des systèmes simples qui consistent à utiliser deux photorésistances comme capteurs pour les suiveurs mono-axe, Ces capteurs sont positionnés très proche l'un par rapport à l'autre et ont un diviseur, ou montés sur des plans inclinés par un angle bien déterminé pour créer un courant ou différence de potentiel entre les deux capteurs comme il est montré dans la figure II.19.

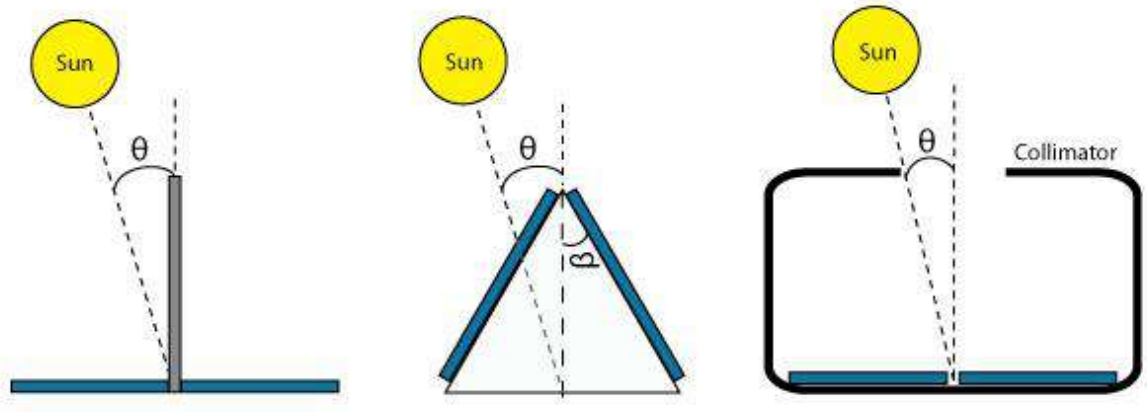


Figure II. 19 : Principe d'équilibrage d'ombre.

Une combinaison de résistances, capacitances, portes logiques, et transistors sont utilisés pour former un circuit de comparaison et un circuit de commande, la sortie de circuit de comparaison alimente le circuit de commande qui alimente le moteur et change son sens de rotation selon le capteur qui récrive le maximum d'illumination.

Lorsque les capteurs de la lumière sont employés, on obtient une commande du suiveur en boucle fermé, parce que l'image du soleil est envoyée à l'organe de commande pour voir si la position du soleil est changée ou non. L'avantage de commande en boucle fermé est la précision de suivi qui atteinte jusqu'à 0.1°, par contre les désavantages sont la nécessité à une structure mécanique très compliqué pour le suiveur, en plus si le ciel devient nuageux il n y'a plus d'asservissement de la position de soleil et par conséquent pas de suivi, le dernier désavantage est la limitation de champs visuel des capteurs [8, 11].

Les suiveurs à base de microprocesseur utilisent des algorithmes pour déterminer la position de soleil au lieu des capteurs de lumière. Ce type de suiveur est utilisé dans la commande en boucle ouvert. L'avantage de cette commande est que le suiveur suit le soleil quel que soit l'état du ciel (nuageux / claire), par contre la désavantage est la précision qui est dégradé par rapport à la commande en boucle fermé [8].

II.4.2. Degré de liberté de mouvement

a) suiveurs de soleil mono-axe

Cette catégorie se compose de différents types de suiveurs :

• **Les suiveurs a axe horizontale** : Ils ont un axe de rotation est parallèle à la surface de terre, il est dirigé soit de l'Est vers l'Ouest ou bien de nord vers le sud.

• **Les suiveurs a axe vertical :** Ils se caractérisent par un axe de rotation verticale à la surface de terre, ils tournent de l'Est vers l'Ouest, dans les hautes altitudes ils sont plus efficaces que ceux a axe horizontale. Pendant l'installation il faut prendre en considération l'ombre, ce qui provoque une diminution de l'énergie collecté.

• **Les suiveurs a axe incliné :** Tous les suiveurs possédant un axe de rotation entre l'horizontale et la verticale sont considérés comme suiveurs a axe incliné, les panneaux sont montés directement sur l'axe de rotation

b) suiveurs de soleil biaxe

Ils disposent de deux degrés de liberté grâce auxquels l'axe perpendiculaire aux panneaux photovoltaïques est aligné parfaitement et en temps réel avec les rayons du soleil, au détriment en revanche d'une plus grande complexité de construction Cette catégorie se compose de différents types de suiveurs :

• **Suiveurs de soleil azimute-altitude** : Dans cette configuration l'axe primaire ou bien l'axe d'azimute doit être parallèle à l'axe de zénith, et l'axe d'élévation est perpendiculaire à l'axe primaire. L'angle de suivi autour l'axe d'azimute est l'angle d'azimute solaire et l'angle de suivi autour l'axe de l'élévation est l'angle de d'élévation solaire.

• **suiveurs de soleil polaire :** Ces suiveurs ont un axe primaire parallèle à l'axe polaire de la terre qui est orienté vers l'étoile polaire, l'axe primaire fait avec l'horizon un angle d'inclinaison égale à l'angle de latitude locale, le deuxième axe de rotation est perpendiculaire au l'axe polaire, l'angle de suivi autour l'axe polaire est égale à l'angle heure de soleil et l'angle de suivi autour le deuxième axe dépend de l'angle déclination.

II.4.3. Type de commande

- Analogiques : la commande est générée en fonction des informations fournies par un capteur qui identifie la position du point le plus lumineux dans le ciel.
- Numériques : la commande provient d'un microprocesseur qui, à l'aide de tableaux mémorisés, connaît à tout moment la position du soleil dans le ciel.

II.4.4. L'intérêt des panneaux mobiles par rapport aux panneaux fixes

Lorsqu'un panneau photovoltaïque est fixe par rapport au sol et orienté vers le sud (implantation en hémisphère nord), son rendement énergétique n'est pas constant au cours de la journée : en début et en fin de journée, le mauvais angle d'éclairement du panneau diminue le rendement de la production électrique.

Lorsqu'un système suiveur oriente le panneau solaire en le faisant pivoter en direction du soleil et en maintenant en permanence un angle d'incidence des rayons idéal, voisin de 90°, la production électrique par m^2 de panneau atteint alors son maximum.

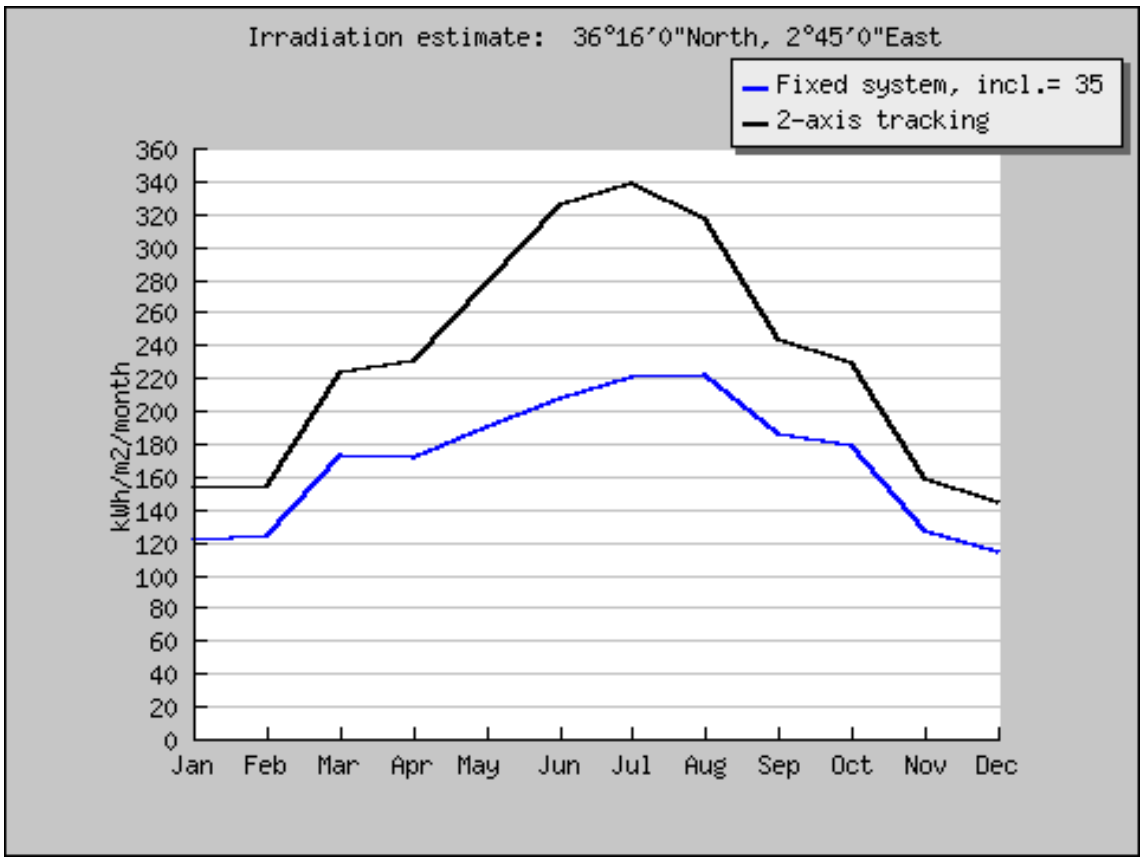


Figure I.20 : comparaison entre système fixe incliné 35° et système suiveur a deux axes toute l'année au site de médéa, algerie.

Le diagramme comparatif de production électrique (figure II.20) montre pour une journée ensoleillée d'été, le gain de production électrique obtenu en passant d'une installation fixe à une installation équipée d'un suiveur de trajectoire solaire.

D'une part à titre énergétique, la technologie avec suiveur ne se justifie qu'à la condition où la commande des mouvements de suivi ne gaspille pas une partie importante du surplus de l'énergie photovoltaïque produite

II. 5. Conclusion :

Cette partie a été consacrée à l'étude de différents types de moteurs pas à pas et les différents modes de commande. L'étude a montré que pour générer un mouvement incrémental de translation. Les modes de commande présentés dans cette partie pour des moteurs pas à pas rotatifs peuvent être adoptés et adaptés au cas des structures linéaires.

Selon les moyens dont nous disposons, nous avons choisi le moteur pas à pas hybride à 1,8 degré/pas dans l'axe de rotation azimut et un autre moteur pas à pas de 7.5degré/pas de type unipolaire pour l'axe d'élévation solaire et la commande mode B (Rotation de couple maximale) en utilisant l'amplificateur ULN2803 dans notre projet de suiveur solaire.

Chapitre III

La carte Arduino Uno

III. 1. Introduction

La carte de microcontrôleur Arduino a excité l'imagination de nombreux passionnés d'électronique. Il séduit quiconque désire fabriquer des montages électroniques grâce à sa facilité d'emploi et à son approche libre.

Dans ce chapitre, nous présentons brièvement :

- Le circuit Arduino, ses origines et ses constituants principaux. Et les différents types de cartes Arduino disponibles.

- Nous montrons le langage de programmation qui permet de programmer une carte Arduino.

III. 2. Le circuit Arduino

À l'origine, Arduino avait été conçu pour aider les étudiants dans leurs expérimentations. En 2005, il a été proposé à la vente par Massimo BANZI et David CUARTIELLES. En quelques années, il est devenu très populaire auprès des concepteurs, des étudiants et des artistes, par sa facilité d'emploi et sa robustesse [13].

Un point clé dans le succès de ce circuit est que tous les schémas de conception Arduino ont été rendus librement accessibles dans le cadre d'une licence Créative Commons. C'est ce qui a permis de voir apparaître des circuits compatibles encore moins chers. Seul le nom Arduino est protégé. Les clones portent souvent des noms contenant une mention du type duino : Boarduino, Seeeduino ou Freeduino. Cela dit, les cartes officielles qui sont fabriquées en Italie se vendent toujours très bien. Les revendeurs les plus importants ne vendent que des cartes officielles dans un joli emballage et avec une garantie de qualité [13].



Figure III.1 : Une carte Arduino.

Une autre raison du succès d'Arduino est que ce circuit n'est pas limité au microcontrôleur. Une grande quantité de cartes d'extension compatibles avec Arduino est apparue. Ce genre de carte, appelée bouclier (shield), s'enfiche directement sur la carte Arduino, dans une relation carte-mère/carte-fille. On trouve des cartes d'extension pour quasiment tous les besoins, ce qui peut souvent on évite de devoir sortir le fer à souder, car il est bien plus pratique de connecter plusieurs cartes d'extension en les empilant les unes sur les autres. Voici quelques-unes des cartes boucliers les plus répandues :

• Ethernet, pour permettre à une carte Arduino de devenir un serveur web ;

• moteur, pour piloter un moteur électrique ;

• hôte USB, pour contrôler des périphériques USB ;

• relais, pour contrôler des relais depuis la carte Arduino.

III. 2. 1. Le microcontrôleur

Un microcontrôleur est un petit processeur informatique relié à des entrées et des sorties numériques (0 ou 1) ou analogiques (tension variable). Il est capable de mémoriser et d'exécuter un programme visant à interpréter les entrées pour agir sur les sorties. Il se programme en général à l'aide d'un ordinateur mais peut fonctionner de manière autonome. En milieu industriel, les automates programmables qui servent à gérer et piloter des machines en sont une illustration. Ce type de machine intervient sur la commande d'un système mais ne peut délivrer beaucoup de puissance. Pour cela, on relaye ses commandes avec des transistors ou des relais [14].

Il existe de nombreux types et tailles de microcontrôleurs avec plus ou moins d'entrées /sorties et d'interfaces. À petite échelle, les PIC (produits par la société Microchip) sont des circuits intégrés bien connus des électroniciens depuis de nombreuses années. De nombreuses autres marques comme ATMEL, STMicroelectronics, Parallax ou Motorola produisent également des machines de ce type. Ils sont en général programmés en C/C++ et nécessitent des connaissances approfondies en électronique. La plate-forme Arduino a apporté une avancée majeure dans l'accessibilité de cette technologie au plus grand nombre par sa simplicité d'utilisation et son coût abordable [14].



Figure III.2 : Atmel Microcontrôleur série SAM D20
Source : ATMEL

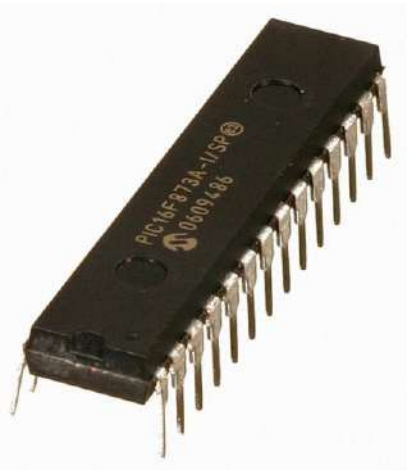


Figure III.3 : un microcontrôleur Microchip série PIC16F628
Source : MICROCHIP

III. 2. 1. 1. La structure de microcontrôleur

Un microcontrôleur est littéralement un petit ordinateur contenu dans une seule puce. Il embarque même plus de fonctions que les premiers ordinateurs personnels, puisqu'il réunit un processeur central, un ou deux kilo-octets de mémoire vive (RAM) pour stocker les données, quelques kilo-octets de mémoire en lecture seule effaçable (EEPROM) ou en mémoire flash, dans lesquels on peut stocker les programmes, ainsi que des broches d'entrée et de sortie. Ces broches permettent de relier le microcontrôleur à vos composants électroniques [13] [15].

Le microcontrôleur se devise grossièrement en trois parties [15] :

- Unité centrale (CPU).
- Mémoire (ROM et RAM)
- Portes d'entrés et de sortie.

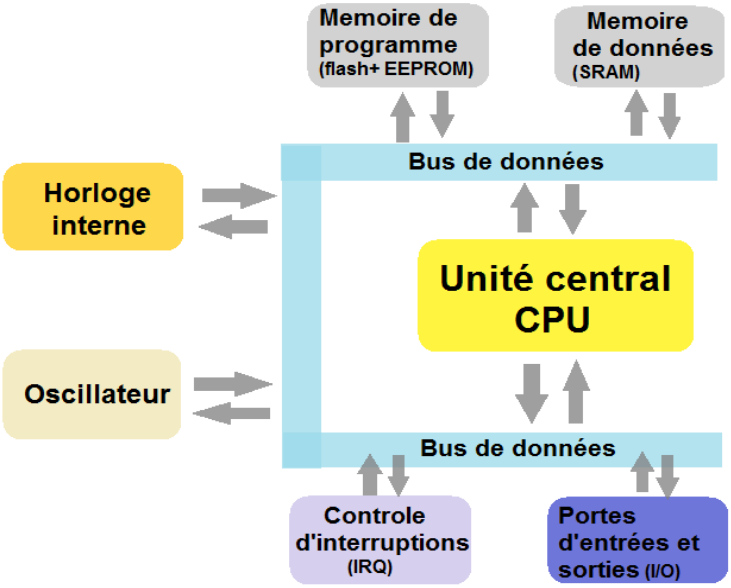


Figure III.4 : Structure de microcontrôleur [4].

A) L'unité central :

L'élément le plus important dans le microcontrôleur est l'unité central appeler également CPU *(Central Processing Unite)*, sa fonction principale consiste à décoder et à exécuter les commandes, elle peut adresser des mémoires, gerer des entrées ou des sorties et réagir a des interruptions, Une interruption *(IRQ ou Interrupt request)* est un signal qui demande au CPU d'interrompre un cycle de calcule en cours pour pouvoir réagir a certain événement.

B) Le bus de données :

Le bus de données sert à transporter les données d'un bloc a un autre , par exemple le CPU demande des données provenant de la mémoire qui sont prise en charge par le bus et immédiatement mise à disposition pour traitement, lorsque le résultats de calcule est disponible il est à nouveau transféré sur le bus et transmis a un port de sorti qui par exemple pilote un moteur pour atteindre un bus précis, cette structure de bus est une autoroute de données utilisable en commun par tous ce qui sont desservis.

C) Les zones de mémoire :

En principe il existe deux types de mémoires d'un microcontrôleur :

- la mémoire de programme.
- la mémoire de données.

La première accueille le programme que le CPU doit exploiter alors que la seconde est utilisée pour gérer les résultats de calcul du moment.

La mémoire de programme d'un microcontrôleur est particulière, le microcontrôleur n'a bien sure pas de disque dur, mais il garde son programme en mémoire en absence de tension d'alimentation externe, un type de mémoire particulier est utilisé à cet effet qu'on l'appelle un mémoire flash, comme son nom l'indique c'est une mémoire non volatile c'est-à-dire que bits et octets ne s'envolent pas en cas de coupure d'alimentation et restons disponible [15].

On a déjà utilisé cette forme de mémoire des milliers de fois sur notre ordinateur, Le BIOS est héberger dans une mémoire flash de type EEPROM et ses données peuvent être écrasées au besoin par l'insertion d'une nouvelle version, on dit alors que le BIOS est reflaché, le contraire se produit dans les mémoires de données dites SRAM, Ces dispositifs sont volatiles et les données mémorisée sont perdues dès qu'il y a coupure d'alimentation mais rien de grave puisque ces derniers ne sont nécessaires que si le programme est exécuter, quand le microcontrôleur est sans courant il n'a rien besoin de calculer mais la mémoire SRAM a sur la mémoire flash un avantages très importants : elle offre un accès plus rapide [15].

D) Les ports entrées et sorties :

Les ports entrées-sorties sont les connexions qui relient le microcontrôleur au monde extérieur, ils constituant une interface a laquelle la périphérique peut être connecté, On entend par périphérique tt ce qui peut être avantageusement raccordé a l'interface il peut s'agir par exemple des composants électriques ou électroniques comme [15], [13] :

- LED (diode électroluminescente)
- Bouton poussoir.
- Commutateur.
- LDR (*Light Dependant Resistor* ou photoresistance)
- Transistor.
- Resistance.
- haut parleur ou élément piézoélectrique….. etc.

Nous reviendrons sur ces éléments et aussi sur leur raccordement aux différents ports.

E) L'horloge :

L'élément essentiel en fonctionnement de microcontrôleur est l'horloge, Un microcontrôleur est en effet un circuit logique séquentiel, c'est-à-dire qu'il fonctionne au rythme d'un signal rectangulaire, appelé horloge, qui cadence toute sa circuiterie interne. La fréquence de cette horloge dépend du type de microcontrôleur utilisé et peut varier de quelques kilohertz (kHz) à plusieurs dizaines de Megahertz (MHz) (dans un PC ou un Mac on dépasse même le Gigahertz (GHz) Plus la fréquence d'horloge est élevée, plus le microcontrôleur est rapide mais, en contrepartie, plus il consomme d'énergie.

Même s'il n'est pas indispensable que la fréquence de cette horloge soit parfaitement stable, c'est cependant généralement le cas car rares sont les applications qui n'ont pas besoin de mesurer du temps ou de générer des signaux aux chronogrammes précis, et cela n'est possible que si l'horloge qui pilote le microcontrôleur est elle-même stable. De ce fait, elle est en général pilotée par un quartz ou au moyen d'un résonateur céramique, seuls composants électroniques capables de générer des signaux à une fréquence stable et précise [14].

F) Le circuit de reset :

Tout comme votre PC, un microcontrôleur exécute en permanence un programme et si, lorsque ce programme est au point et ne comporte plus de ' bug' (oui, je sais, ce n'est jamais le cas sur votre PC…) il se déroule en continu tant que l'application est alimentée ; en phase de développement il est fréquent qu'il se plante.

Pour sortir d'une telle situation, il est bien sûr possible de couper l'alimentation puis de la rétablir, ce qui fait redémarrer le programme au début. En effet, le microcontrôleur dispose d'un circuit de reset automatique à la mise sous tension qui se charge donc d'une telle opération. Mais il est nettement plus confortable de pouvoir réaliser cette opération par simple appui sur un poussoir, et c'est là le rôle joué par le circuit de reset manuel.

III. 2. 3. Les différentes familles d'Arduino

Maintenant que on sait ce qu'est un microcontrôleur et ce qu'il faut pour pouvoir le faire fonctionner, nous pouvons passer aux différentes versions d'Arduino. Les concepteurs de l'Arduino ont fixé leur choix sur les microcontrôleurs fabriqués par la société Atmel, et plus précisément sur la famille AVR de ce fabricant. Même si l'on peut discuter pendant des heures de ce choix et des mérites comparés de ces circuits par rapport à ceux de fabricants concurrents ; les microcontrôleurs AVR représentent aujourd'hui un excellent compromis prix/puissance et, même s'il ne s'agit que de microcontrôleurs à unité centrale 8 bits, ils permettent de développer des applications très intéressantes.

Il existe plusieurs versions de cartes Arduino. La version courante de base, la Duemilanove (à 2009 en italien - les créateurs de l'Arduino sont des italiens...), utilise le microcontrôleur Atmel ATmega328. La version précédente Diecimila, et les premières versions de la Duemilanove utilisent le microcontrôleur ATmega168, tandis que les cartes plus anciennes utilisent le microcontrôleur ATmega8, La carte Arduino Mega est basée sur l'ATmega1280.

III. 2. 3. 1. La carte Arduino UNO

La carte Arduino Uno est une carte à microcontrôleur basée sur l'ATmega328 Elle dispose :

- De 14 broches numériques d'entrées/sorties (dont 6 peuvent être utilisées en sorties PWM (largeur d'impulsion modulée)),
- De 6 entrées analogiques (qui peuvent également être utilisées en broches entrées/sorties numériques),
- D'un quartz 16Mhz,
- D'une connexion USB,
- D'un connecteur d'alimentation jack,
- D'un connecteur ICSP (programmation "in-circuit"), et d'un bouton de réinitialisation (reset).

Elle contient tout ce qui est nécessaire pour le fonctionnement du microcontrôleur, Pour pouvoir l'utiliser et se lancer, il suffit simplement de la connecter à un ordinateur à l'aide d'un câble USB (ou de l'alimenter avec un adaptateur secteur ou une pile, mais ceci n'est pas indispensable, l'alimentation étant fournie par le port USB).

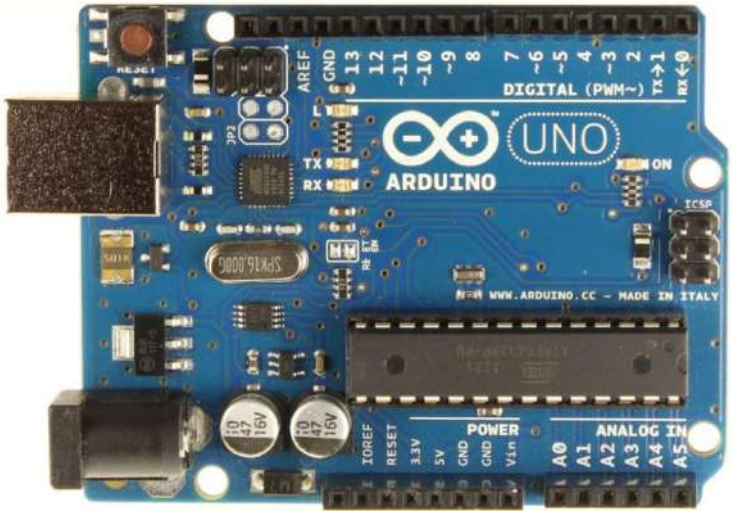


Figure III.5 : La carte Arduino Uno.

Source : ARDUINO.CC

Tableau III.1 : Synthèse des caractéristiques d'Arduino Uno.

Microcontrôleur	**ATmega328**
Tension de fonctionnement	5V
Tension d'alimentation (recommandée)	7 – 12V
Tension d'alimentation (limites)	6 – 20V
Broches E/S numériques	14 (dont 6 disposent d'une sortie PWM)
Broches d'entrées analogiques	6 (utilisables en broches E/S numériques)
Intensité maxi disponible par broche E/S (5V)	40 mA (ATTENTION : 200mA cumulé pour l'ensemble des broches E/S)
Intensité maxi disponible pour la sortie 3.3V	50 mA
Intensité maxi disponible pour la sortie 5V	Fonction de l'alimentation utilisée - 500 mA max si port USB utilisé seul
Mémoire Programme Flash	32 KB (ATmega328) dont 0.5 KB sont utilisés par le bootloader
Mémoire SRAM (mémoire volatile)	2 KB (ATmega328)
Mémoire EEPROM (mémoire non volatile)	1 KB (ATmega328)
Vitesse d'horloge	16 MHz

Au moment de la rédaction de ces lignes, le cœur de la carte Arduino Uno est un microcontrôleur de la famille AVR, un Atmel Atmega 328P. Ce microcontrôleur renferme dans un seul composant :

- Un processeur 8 bits à architecture RISC ;

- De la mémoire avec des espaces physiques séparés : mémoire Flash (32 Ko) pour les programmes, mémoire vive SRAM (2 Ko) pour les données, mémoire EEPROM (2 Ko) pour les données de sauvegarde

- Toute la logique d'horloge (16 MHz) ;

- Des circuits d'interface et des périphériques d'entrée-sortie permettant au processeur d'accéder au monde extérieur : des Timers/Counters (T/C) 8 et 16 bits, Génération des signaux PWM, des interfaces de communication série (UART, SPI, TWI….), Un convertisseur Analogique-Numérique (A/D Conv.), etc.

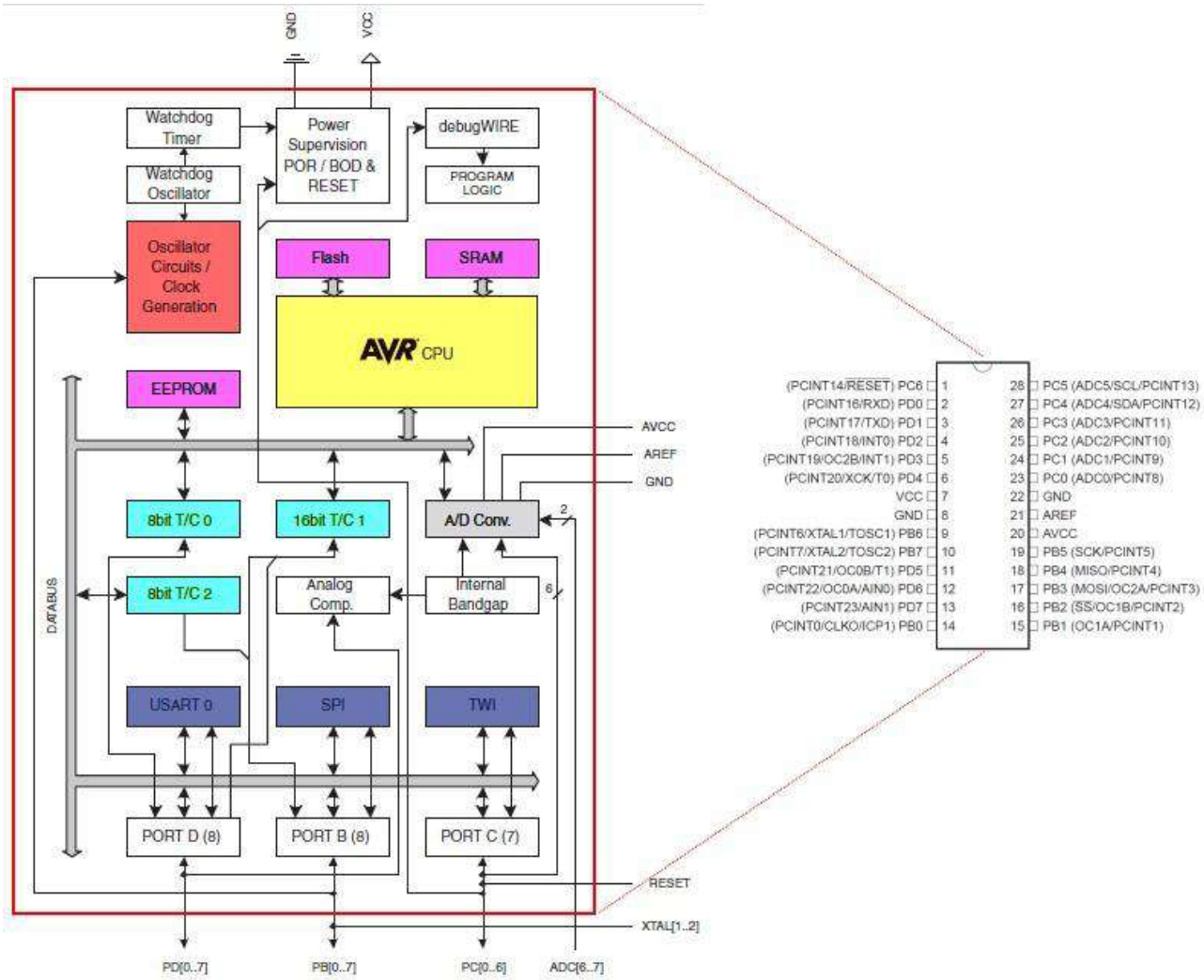


Figure III.6 : Schéma bloc de description du microcontrôleur AVR.

Un processeur à 16 MHz et 32 Ko de mémoire Flash pour stocker vos programmes, ces chiffres peuvent prêter à sourire en comparaison des GHz et des Go de votre ordinateur personnel. Mais avec son format carte de crédit et une consommation inférieure au Watt, la carte Arduino satisfera pourtant vos premières exigences en termes d'embarqué.

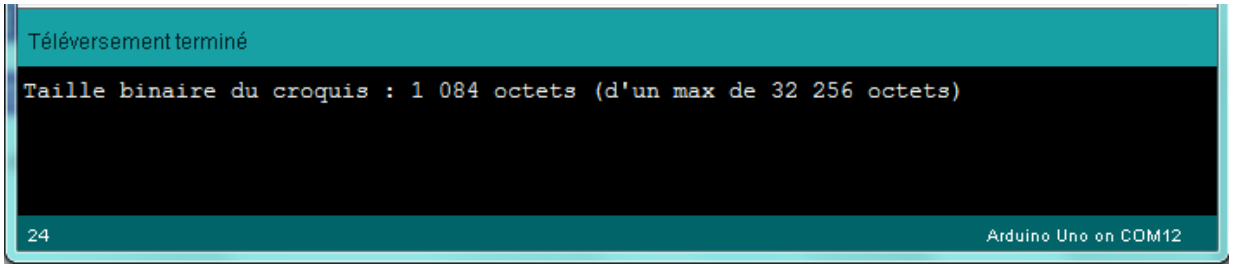


Figure III.7 : 32 Ko disponibles pour vos programmes en Arduino UNO.

A) Alimentation d'une carte Arduino

Sur la figure II.8 ; nous pouvons repérer juste sous le connecteur USB un composant qui est le régulateur de tension 5 V. Ce composant se charge de stabiliser et de réguler à 5 V la tension d'entrée qui peut varier entre 7 V et 12 V.

En comparaison des autres composants montés en surface (CMS), ce régulateur est assez volumineux, ce qui lui permet de mieux dissiper la chaleur qu'il dégage pour faire baisser la tension en délivrant un ampérage suffisant. Cela permettra au circuit de piloter correctement des composants externes.

B) Entrées analogiques

Le second connecteur du bas comporte les légendes A0 à A6. Ces broches d'entrée permettent de mesurer une tension pour pouvoir en utiliser les valeurs dans les programmes. Précisons que les broches servent à mesurer une tension et non un courant.

Le circuit d'entrée auquel chacune des broches donne accès possède une résistance interne (impédance) très importante, ce qui fait que le courant passant par la broche est toujours très faible. Ces entrées sont désignées comme étant de type analogique, et elles le sont par défaut, mais on peut également les exploiter en tant qu'entrées numériques ou même comme sorties numériques.

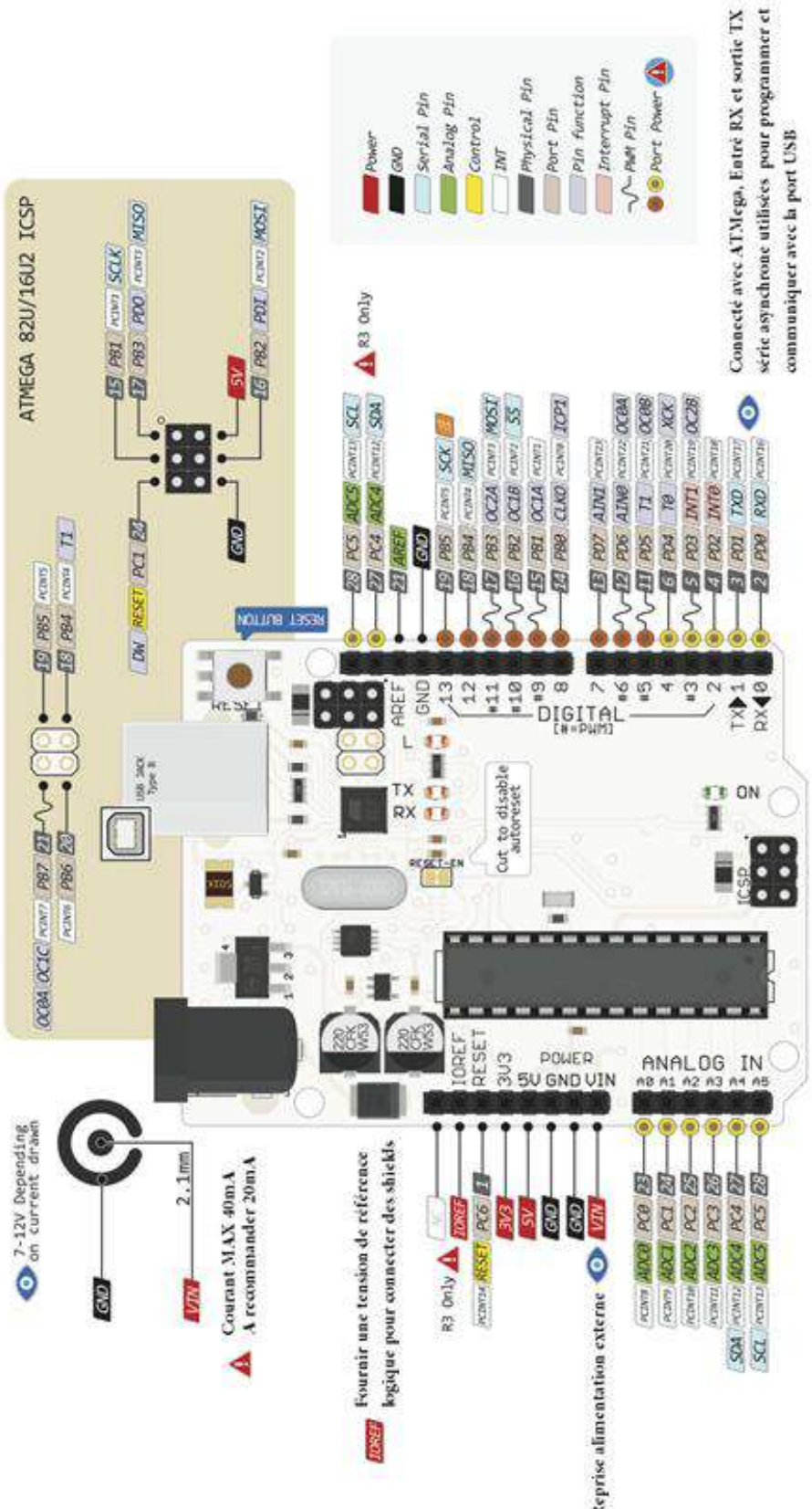


Figure II.8: Structure d'une carte Arduino UNO

C) Connexions numériques

Passons maintenant aux deux connecteurs du haut de la carte, et commençons par celui de droite. Les légendes désignent des entrées numériques de 0 à 13 en partant de la droite (Digital). Ces broches peuvent servir d'entrées ou de sorties.

Lorsqu'on les utilisé en tant que sorties, elles se comportent comme les broches de source de tension présentées plus haut, sauf qu'elles fonctionnent toutes en 5 V et qu'elles peuvent être activées et désactivées depuis un programme. Si on active une de ces broches, la tension présente sera à 5 V. Si on désactive la broche, la tension sera à 0 V. Comme pour les broches de tension, il faut veiller à ne jamais dépasser le courant maximal qu'elles peuvent supporter. Les deux premières broches à droite (0 et 1) portent une légende supplémentaire (RX et TX) pour la transmission et la réception. Ces deux broches sont réservées à la communication ; elles sont, d'une façon indirecte, les connexions en émission et en réception de la connexion USB vers votre ordinateur.

Toutes les broches numériques peuvent supporter jusqu'à 40 mA (milliAmpères) à 5 V. C'est largement suffisant pour alimenter une LED standard, mais assurément pas pour alimenter directement un moteur électrique.

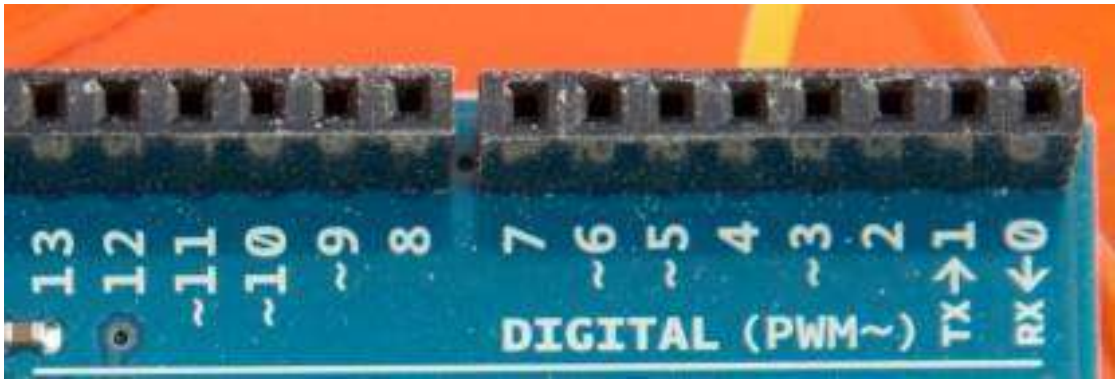


Figure III.9 : Entrées/sorties numériques D0 à D13

III. 2. 3. 2. Les autres cartes Arduino :

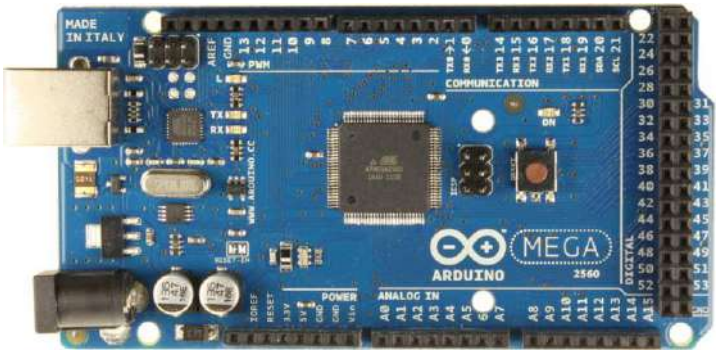


Figure III.10 : La carte Arduino Méga.

Source : ARDUINO.CC



Figure III.11 : La carte Arduino Bluetooth.

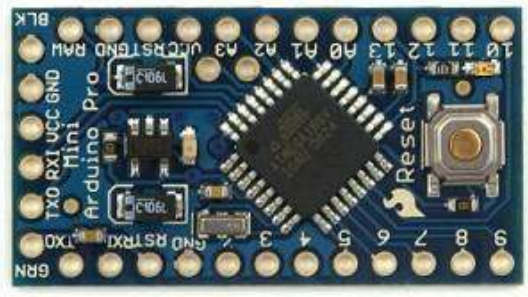


Figure III.12 : La carte Arduino mini.

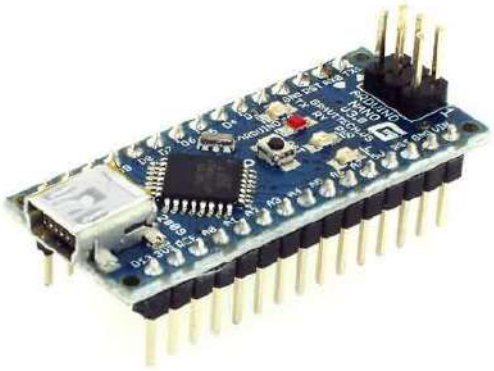


Figure III.13 : La carte Arduino Nano.

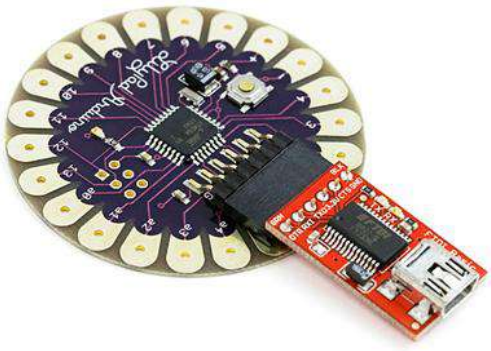
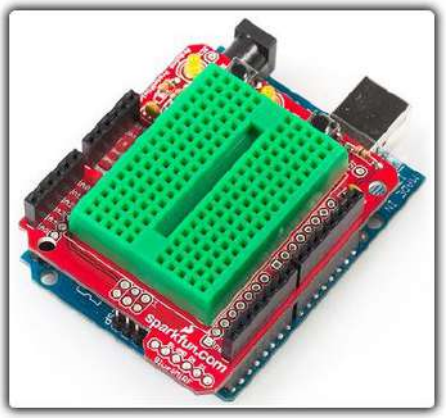


Figure III.14 : La carte Arduino LilyPad.

III. 3. Les cartes d'interface ou shields

Une carte Arduino seule, même si on dispose de quelques LED, résistances, boutons poussoir, etc. n'est pas d'une grande utilité et reste dans ces cas limités à des fins d'apprentissage. Si on veut aller plus loin et piloter les moteurs de votre robot, on devra passer par une interface dédiée (des composants sur une carte électronique).

Un shield est une carte d'interface spécialement dédiée à l'Arduino. Ces cartes ont des dimensions sensiblement voisines de la carte Arduino et peuvent s'enficher directement sur les connecteurs de celle-ci. Il est bien sûr impossible de recenser entièrement ici les centaines de shields existant sur le marché (dont quelques-uns référencés sur le site officiel). on trouve une liste de fabricants et/ou distributeurs de shields pour Arduino dans le site en fin de tutoriel, On peut tout de même en signaler quelques-uns bien pratiques et donner une idée.



Source : Sparkfun
Figure III.15: Un shield prototypage Sparkfun ProtoShield.

- Les shields d'affichage : Ce shield, en plus de son écran LCD (deux lignes de 16 caractères), comporte cinq boutons poussoirs programmables.

- Les shields pour commander des moteurs : Ce shield permet de piloter jusqu'à deux moteurs électriques simultanément (idéal pour un petit robot roulant à deux roues motorisées indépendantes). Avec deux connecteurs numériques utilisés par moteur seulement, on contrôle le sens et la vitesse de rotation du moteur.

- Les shields pour communiquer : Si on souhaite rendre disponible l'état d'un capteur de température sur un réseau local Ethernet ou sur Internet, ce shield évolué avec une bibliothèque très complète et de nombreux exemples de programmes saura transformer votre Arduino en véritable serveur web connecté sur l'extérieur.

III. 3.1. La carte d'interface RTC (Real Time Clock)

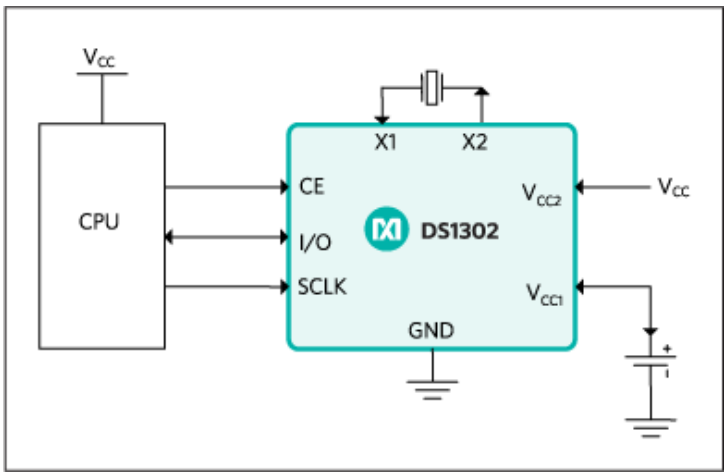
A la recherche d'une horloge RTC pour Arduino, On a fait l'inventaire de ce qui peut se trouver aujourd'hui sur le marché. Le besoin est simple : Lorsque l'Arduino est éteint, ou s'il redémarre, il ne connait ni le jour ni l'heure.

La carte de Hobby Electronics embarque un DS1302 de Maxim. Ce circuit utilise une entrée /sortie série et intègre 31 octets de RAM sauvegardés par la pile.



Source : Hobby Electronics
Figure III.16: Une carte RTC DS1302

La carte est munie de cinq contacts qui vont permettre de la relier à l'Arduino, Elle comporte un circuit intégré DS1302+, un quartz horloger et une pile CR2032, un grand classique des piles de sauvegarde sur PC. Ce circuit intègre une horloge/calendrier et 31 octets de RAM statique, sauvegardés par la pile. La communication avec le microprocesseur se fait en communication série. Seuls 3 fils sont nécessaires pour faire communiquer le DS1302 : CE (RST), I/O (DATA) et SCLK (Serial Clock).



Source : MAXIM Intégrée
Figure III.17: Le circuit de la carte RTC

Le DS1302 peut être alimenté entre 2 volts et 5,5 volts. Il se comportera donc parfaitement sous 3,3 volts, et il n'y aura aucun problème d'adaptation entre sa sortie et l'entrée avec l'Arduino auquel il sera relié. On relie une résistance de tirage entre la borne DATA et le + 3,3 volts pour la protection.

Le DS1302 conserve les données et maintient l'heure avec une puissance inférieure à 1µW. La notice technique indique un courant de maintien sur la pile entre 0,3 et 1µA selon la tension. Le quartz est un quartz horloger de fréquence 32,768 kHz. La précision de l'horloge dépend de sa précision, Les cristaux ordinaires ont une précision de +/- 50ppm environ, ce qui représente 0,005%… Pour obtenir une meilleure précision, il faut mettre en œuvre des moyens disproportionnés avec le prix de la carte RTC.

III. 4. Environnement de développement intégré (EDI)

Le logiciel Arduino a pour fonctions principales :

- De pouvoir écrire et compiler des programmes pour la carte Arduino
- De se connecter avec la carte Arduino pour y transférer les programmes
- De communiquer avec la carte Arduino

Cet espace de développement intégré (EDI) dédié au langage Arduino et à la programmation des cartes Arduino comporte :

- une BARRE DE MENUS comme pour tout logiciel une interface graphique (GUI).
- une BARRE DE BOUTONS qui donne un accès direct aux fonctions essentielles du logiciel et fait toute sa simplicité d'utilisation.
- un EDITEUR (à coloration syntaxique) pour écrire le code de votre programme, avec onglets de navigation.
- une ZONE DE MESSAGES qui affiche indique l'état des actions en cours.
- une CONSOLE TEXTE qui affiche les messages concernant le résultat de la compilation du programme.
- un TERMINAL SERIE (fenêtre séparée) qui permet d'afficher des messages textes reçus de la carte Arduino et d'envoyer des caractères vers la carte Arduino. Cette fonctionnalité permet une mise au point facilitée des programmes, permettant d'afficher sur l'ordinateur l'état de variables, de résultats de calculs ou de conversions analogique-numérique : un élément essentiel pour améliorer, tester et corriger ses programmes.

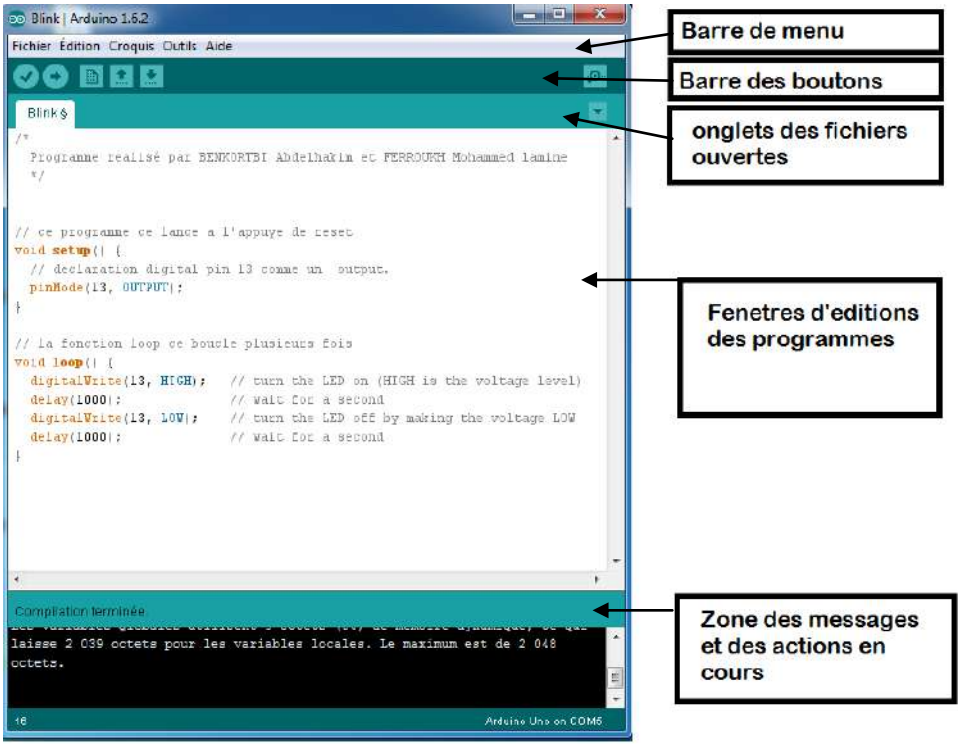


Figure III.18 : Espace de développement intégré (EDI) dédié au langage Arduino.

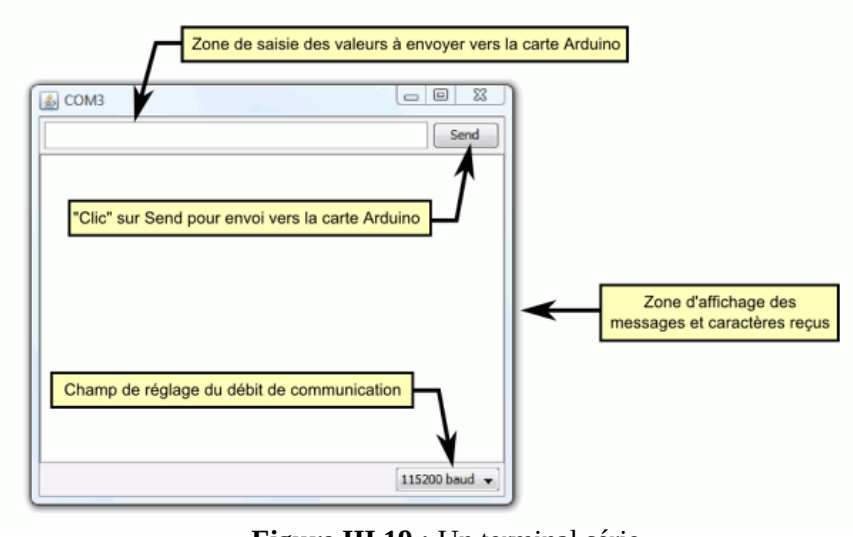


Figure III.19 : Un terminal série.

III. 4. 1. Transfert des programmes vers la carte Arduino

Le code écrit avec le logiciel Arduino est appelé un programme (ou une séquence - sketch en anglais), Ces programmes sont écrits dans l'éditeur de texte. Celui-ci a les fonctionnalités usuelles de copier/coller et de rechercher/remplacer le texte. La zone de messages donne l'état de l'opération en cours lors des sauvegardes, des exportations et affiche également les erreurs. La console texte affiche les messages produits par le logiciel Arduino incluant des messages d'erreur détaillés et autres informations utiles. La barre de

boutons permet de vérifier la syntaxe et de transférer les programmes, créer, ouvrir et sauver votre code, et ouvrir le moniteur série. La barre des menus permet d'accéder à toutes les fonctionnalités du logiciel Arduino.

A. Saisir votre programme et vérifier le code

On suppose ici qu'un programme correctement écrit se trouve dans la fenêtre éditeur. Pour votre première programmation de la carte, on choisit dans le menu *Fichier* > *Exemples* > *Digitale* > *Blink* : un programme s'ouvre avec du code dans la fenêtre éditeur. Appuyez alors sur le bouton *Vérifier* de la barre d'outils pour lancer la vérification du code:

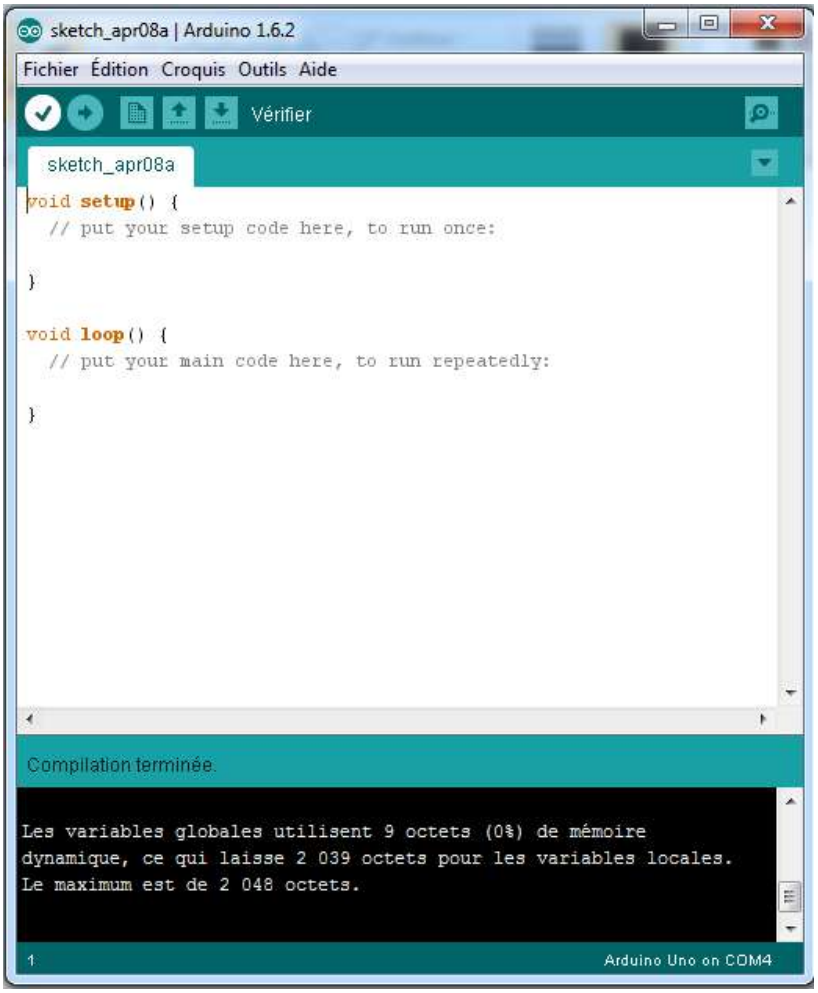


Figure III.20 : Compilation terminée.

B. Sélectionner le port série et la carte Arduino

Avant de transférer votre programme vers la carte Arduino, on doit vérifier que bien sélectionné la bonne carte Arduino depuis le menu *Outils* > *Carte*. Les cartes sont décrites ci-dessous. Votre carte doit évidemment être connectée à l'ordinateur via un câble USB.

On doit également sélectionner le bon port série depuis le menu *Outils* > *Port Série*, Sous Windows, c'est probablement COM1 ou COM2 (pour une carte série) ou COM4, COM5, COM7 ou supérieur (pour une carte USB) - pour trouver le bon, on peut chercher dans la rubrique des ports série USB dans la section des ports du panneau de configuration ou du gestionnaire de périphériques.

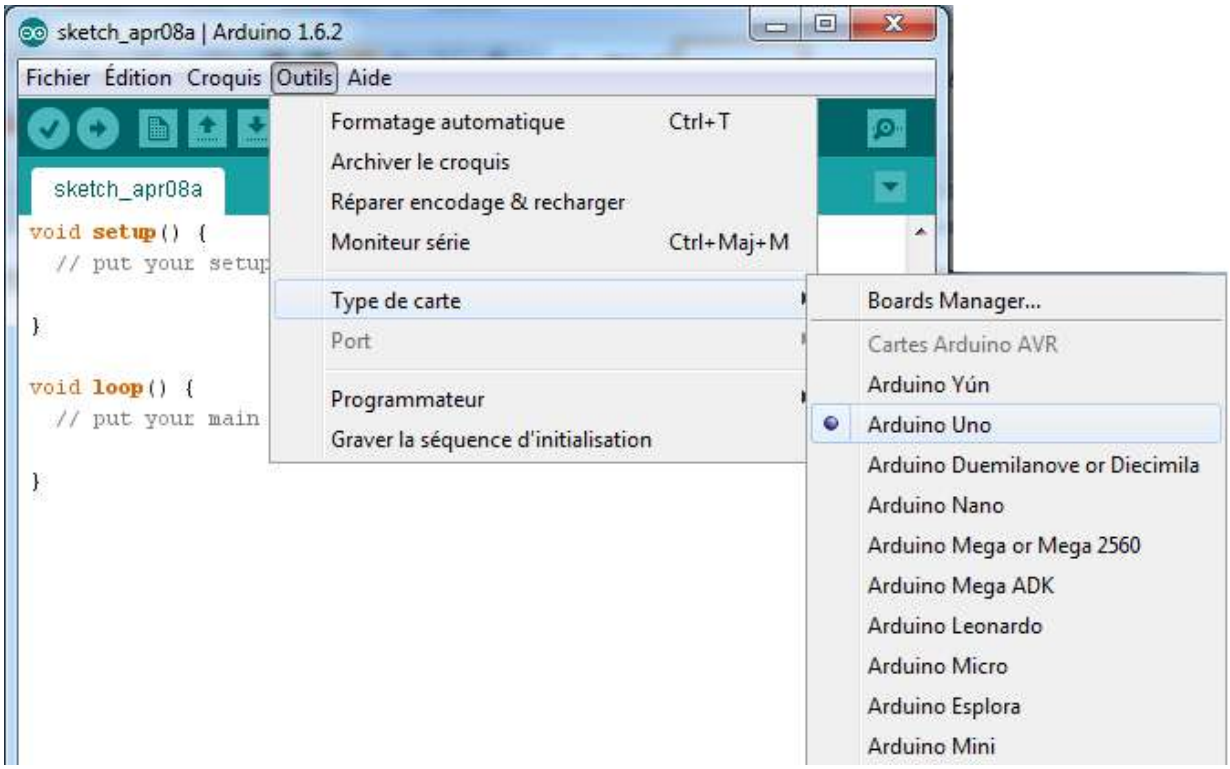


Figure III.21 : Sélectionner la carte Arduino.

C. Transférer le programme

Une fois que l'on a sélectionné le bon port série et la bonne carte Arduino, cliquez sur le bouton *Transfert*, dans la barre d'outils, ou bien sélectionner le menu Fichier > Transférer vers la carte. Avec les versions récentes (Duemilanove notamment), la carte Arduino va alors automatiquement se réinitialiser et démarrer le transfert. Avec les versions précédentes qui ne sont pas équipées de l'auto-réinitialisation, on peut appuyer sur le bouton "reset" de la carte juste avant de démarrer le transfert.

Une fois le transfert terminé, le logiciel Arduino doit afficher un message indiquant que le transfert est bien réalisé, ou montrer des messages d'erreurs



Figure III.22 : Transférer le programme a la carte Arduino.

D. Lancement de programme

Quand on a transféré un programme, en utilisant le bootloader Arduino, un petit programme (code binaire) a été chargé dans le microcontrôleur sur votre carte Arduino. Cette technique permet comme l'on pouvait le voir de transférer votre programme sans aucun matériel externe. Une fois le transfert terminé, le bootloader est actif une petite seconde une fois que la carte est réinitialisée à la fin du transfert; puis le dernier programme programmé dans la carte s'exécute. Le bootloader fait clignoter la LED de la carte (broche 13) quand il démarre c'est à dire quand la carte est réinitialisée.

E. Le moniteur série

Le logiciel Arduino intègre également un TERMINAL SERIE (fenêtre séparée) qui permet d'afficher des messages textes reçus de la carte Arduino et d'envoyer des caractères vers la carte Arduino. Cette fonctionnalité permet une mise au point facilitée des programmes, permettant d'afficher sur l'ordinateur l'état de variables, de résultats de calculs ou de conversions analogique-numérique : un élément essentiel pour améliorer, tester et corriger ses programmes.

Il faut alors régler le débit de communication dans le TERMINAL SERIE sur la même valeur que celle utilisée par le programme avec lequel nous allons programmer la carte Arduino

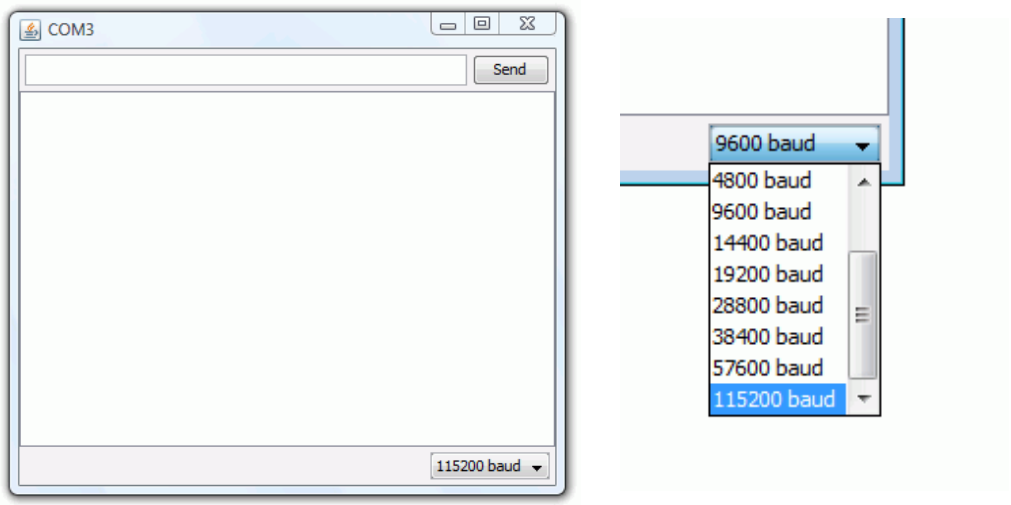


Figure III.23 : réglage de débit de communication.

III. 5. Le langage de programmation

Il n'est pas rare de nos jours que les gens parlent plus d'une langue. D'ailleurs, plus l'on en apprend, plus il devient simple d'en apprendre d'autres, parce que on commence à repérer des invariants de grammaire et de vocabulaire qui sont partagés entre plusieurs langues. Il en va de même avec les langages de programmation. Autrement dit, si l'on ont déjà pratiqué un autre langage, on sera vite à niveau en langage C, Le langage qui permet de programmer une carte Arduino [13].

Nous devons maintenant aller voir plus précisément comment le code source du langage, c'est-à-dire un ensemble de mots dans un programme d'édition, finit par se transformer en quelque chose qui agit sur le monde réel, comme allumer ou éteindre une LED. La Figure III.23 montre les grandes étapes de cette chaîne de production qui va de la saisie du code source dans l'atelier IDE Arduino jusqu'à l'exécution du programme sur la carte.

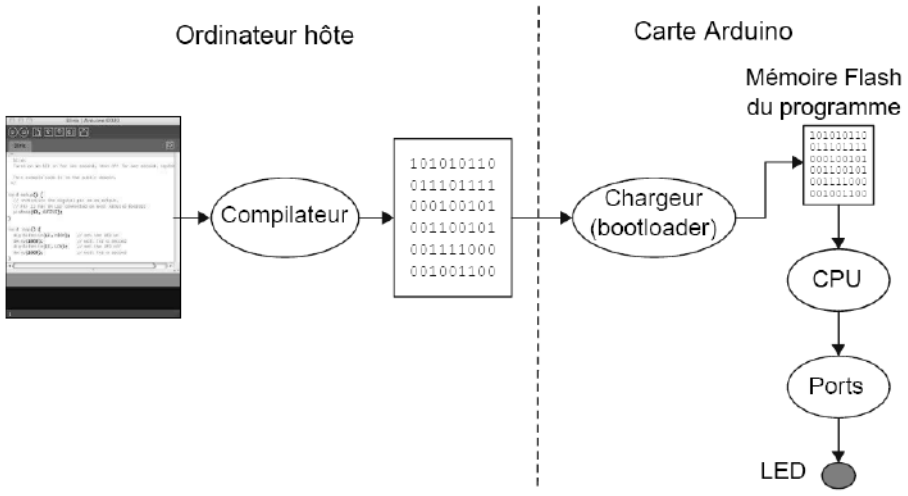


Figure III.24 : Du code source a la carte.

Lorsqu'on utilise le bouton de téléversement dans notre logiciel Arduino, On déclenche une série d'événements dont les deux dernières étapes correspondent à l'installation du programme exécutable sur la carte Arduino et à son démarrage. Noté qu'il ne s'agit pas de simplement transférer les instructions de code source qu'on saisit vers la carte Arduino.

La première étape de cette chaîne se nomme la compilation. Elle consiste à analyser les instructions de code source que on a écrite pour les convertir en séquences de code machine, c'est-à-dire le langage binaire que peut comprendre le processeur de la carte Arduino. Si on utilise le bouton triangulaire de vérification dans l'atelier Arduino, on demande de faire une compilation du programme C sans faire suivre cette action du transfert du code résultant vers la carte. La compilation du code permet en effet de vérifier que la totalité du contenu est conforme aux règles de syntaxe du langage C.

III. 5. 1. Structure d'un programme

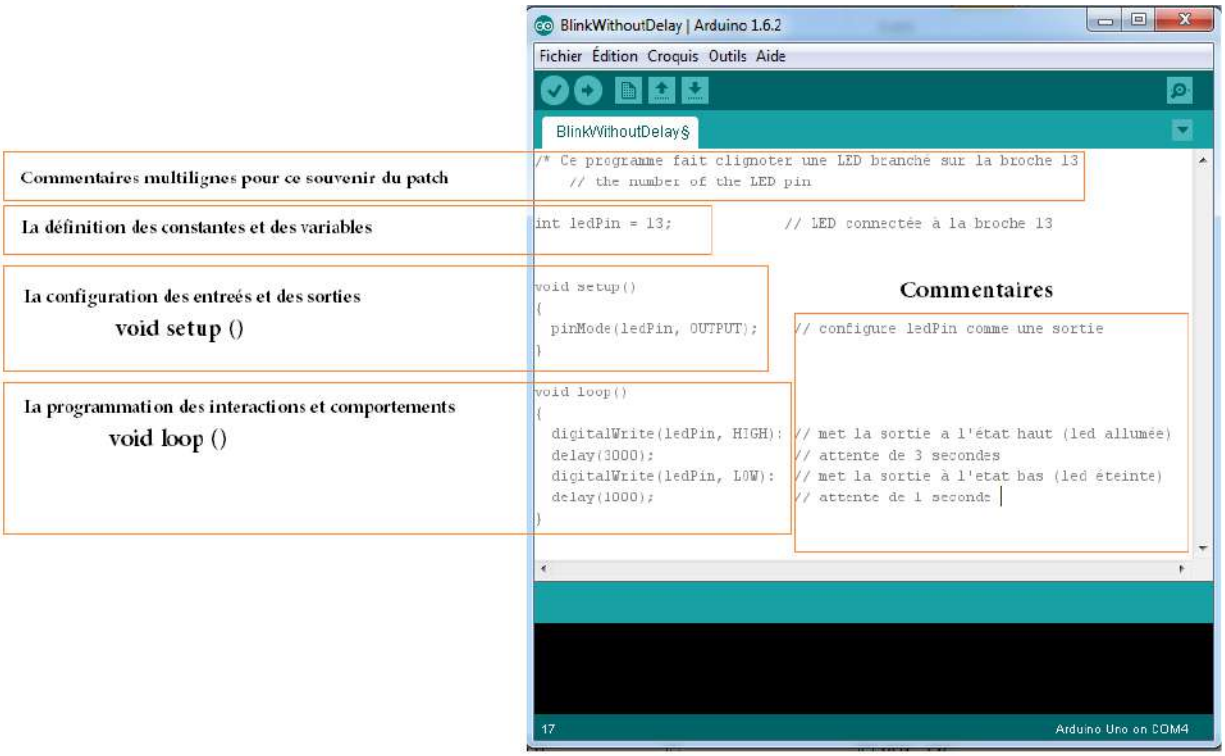


Figure III.25 : Structure d'un programme.

Examinons de plus près dans la figure III.24 dans la configuration d'entrées et des sorties les lignes de code initiales qui doivent être présentes dans tous vos croquis. Nous y trouvons de nouveaux éléments, et notamment le mot *void* ainsi que des paires

d'accolades. Voyons d'abord à quoi sert le mot *void*. *void setup()* signifie que on définissait une fonction portant le nom *setup()*. Dans Arduino, certaines fonctions sont préprogrammées, et c'est le cas de *digitalWrite()* et de *delay()*, mais on pouvait définir vos propres fonctions. Les deux fonctions *setup()* et *loop()* sont particulières : leur nom est immuable, mais il est à l'un de définir leur contenu dans chacun de vos croquis.

Le mot-clé *void* est suivi d'un espace, puis du nom de la fonction, puis d'un couple de parenthèses délimitant la liste des paramètres. Dans les deux cas de notre exemple, il n'y a aucun paramètre, mais les parenthèses doivent cependant être présentes.

A. Fonctions *setup* et *loop*

La raison pour laquelle tout programme Arduino rend obligatoire la présence des deux fonctions nommées *setup()* et *loop()* est que cela permet de séparer les actions qui ne doivent être réalisées qu'une fois au démarrage du croquis de celles qui doivent pouvoir se dérouler plusieurs fois. La fonction *setup()* est utilisée pour initialiser les variables, le sens des broches, les librairies utilisées. La fonction setup n'est exécutée qu'une seule fois, après chaque mise sous tension ou reset (réinitialisation) de la carte Arduino.

Après avoir créé une fonction *setup(),* qui initialise et fixe les valeurs de démarrage du programme, la fonction *loop()* (boucle en anglais) fait exactement ce que son nom suggère et s'exécute en boucle sans fin, permettant à votre programme de s'exécuter et de répondre. Utiliser cette fonction pour contrôler activement la carte Arduino.

B. Venus des mathématiques : les opérateurs arithmétiques

Le signe égal = dans le langage de programmation C est appelé opérateur d'assignement. Il indique au microcontrôleur d'évaluer la valeur ou l'expression qui se trouve à droite du signe égal et de la stocker dans la variable à gauche du signe égal.

Addition +, Soustraction -, Multiplication *, et Division / Ces opérateurs renvoient respectivement la somme, la différence, le produit ou le quotient entre deux opérandes. Cette opération est réalisée en utilisant le type des données des opérandes. Ainsi par exemple, 9 / 4 donne 2 dès lors que 9 et 4 sont de type int. Cela signifie cependant que l'opération peut déborder si le résultat est plus grand que ce que le type de donnée utilisé peut contenir.

C. Syntaxe de base

Les accolades {} sont un élément majeur de la programmation en langage C. Elles sont utilisées dans plusieurs constructions différentes, Toute ouverture d'une accolade d'ouverture "{" doit obligatoirement être accompagnée dans le code d'une accolade de fermeture "}" correspondante. On dit souvent qu'il faut que les accolades soient équilibrées (c.à.d. autant de"{" que de "}"). L'EDI (Environnement de Développement Intégré = l'éditeur de programmes) Arduino inclut une fonctionnalité pratique pour vérifier la correspondance des accolades entre elles. Il suffit de sélectionner une accolade, ou même de cliquer juste après l'accolade, et son compagnon logique sera mis en surbrillance. Actuellement cette fonctionnalité bogue légèrement à savoir que l'IDE prend souvent en compte de façon incorrecte les accolades incluses dans les commentaires.

Point-virgule **;** Utilisé (et obligatoire) à la fin d'une instruction. Oublier le point-virgule en fin de ligne donnera une erreur de compilation. Par exemple :

```
int a = 13;         // le point-virgule indique la fin de l'instruction
```

D. Instruction de contrôle

L'instruction *if* ("si" en français), utilisée avec un opérateur logique de comparaison, permet de tester si une condition est vraie, L'instruction *if/else* (si/sinon en français) permet un meilleur contrôle du déroulement du programme que la simple instruction if, en permettant de grouper plusieurs tests ensemble.

L'instruction *for* est utilisée pour répéter l'exécution d'un bloc d'instructions regroupées entre des accolades. Un compteur incrémental est habituellement utilisé pour incrémenter et finir la boucle. L'instruction for est très utile pour toutes les opérations répétitives et est souvent utilisées en association avec des tableaux de variables pour agir sur un ensemble de données ou broches.

Il y a 3 parties dans l'entête d'une boucle for :

```
for (initialisation; condition; incrémentation) {
//instruction(s)à exécuté;
}
```

E. Fonctions mathématiques et trigonométriques

degrees(rad) Converti un angle en radians en degrés. Pour mémoire, 180° vaut pi (=3.14159) radians.

sin(rad) Calcule le sinus d'un angle (en radians). Le résultat sera entre -1 et 1.

cos(rad) Calcule le cosinus d'un angle (en radians). Le résultat sera entre -1 et 1.

abs(x) Calcul la valeur absolue d'un nombre.

sqrt(x) Calcule la racine carrée d'un nombre.

F. Gestions de temps

delay(ms) Réalise une pause dans l'exécution du programme pour la durée (en millisecondes) indiquée en paramètre.

delayMicroseconds(us) Stoppe le programme pendant la durée (en microsecondes) spécifiées en paramètres. Il y a 1000 microsecondes dans une milliseconde, et un million de microsecondes dans un seconde, Actuellement, la valeur la plus grande qui peut produire une pause précise est 16383. Ceci pourra changer dans les versions futures d'Arduino. Pour des délais plus longs que quelques milliers de microsecondes, on devrait plutôt utiliser l'instruction *delay().*

G. Par nature invariables : les constantes

Dans le langage Arduino, les constantes sont des variables prédéfinies. Elles sont utilisées pour rendre les programmes plus faciles à lire. Nous les avons classées ici en groupes.

True (vrai) et *False* (faux) deux constantes utilisées pour représenter le vrai et le faux dans le langage Arduino. *false (= faux)* la constante false est la plus facile des deux à définir. false est définie comme le 0 (zéro).

La constante *true* est souvent défini comme étant le 1, ce qui est correct, mais la constante *true* a une définition plus large. Tout entier qui n'est pas zéro est vrai dans un sens booléen. Ainsi -1, 2 et -200 sont tous définis comme vrai, toujours dans un sens booléen. Noter que les constantes *true* et *false* sont écrits en minuscule à la différence des constantes HIGH, LOW, INPUT et OUTPUT.

Lorsqu'on lit ou on écrit sur une broche numérique, seuls deux états distincts sont possible, la broche ne peut prendre/être qu'à deux valeurs : HIGH (HAUT) ou LOW (BAS). La signification de la constante HIGH (en référence à une broche) est quelque chose de différent selon que la broche est définie comme une ENTREE ou comme une SORTIE. Si la broche est configurée en ENTREE avec l'instruction *pinMode*, et lue avec l'instruction *digitalRead*, le microcontrôleur renverra HIGH (=HAUT) si une tension de 3V ou + est présente sur la broche.

Dans le cas d'une broche qui est configurée en ENTREE avec l'instruction *pinMode*, et secondairement mise au niveau HIGH avec l'instruction *digitalWrite*, cela active une résistance interne de 20 Kohms, dite de "rappel au plus" (ou pullup en anglais), qui maintiendra la broche en entrée au niveau HIGH (=HAUT) tant qu'un circuit externe (un bouton poussoir par exemple) ne la mettra au niveau LOW (=BAS).

Quand une broche est configurée en SORTIE avec l'instruction *pinMode*, et mise au niveau HAUT avec l'instruction *digitalWrite*, la broche est mise à 5V. Dans cet état, la broche peut fournir une certaine intensité (jusqu'à 40 mA par broche, sans dépasser 200 mA pour l'ensemble des broches), par exemple pour allumer une LED qui est connectée sur cette broche en série avec une résistance à la masse (0V), ou à une autre broche configurée en sortie et maintenue au niveau LOW (=BAS).

LOW Cela signifie que la constante LOW a également une signification différente selon que la broche est configurée en ENTREE ou en SORTIE. Quand une broche est configurée en ENTREE avec l'instruction *pinMode*, et lue avec l'instruction *digitalRead*, le microcontrôleur renverra un niveau BAS si une tension de 2V ou moins est présente sur la broche.

Quand la broche est configurée en SORTIE avec l'instruction *pinMode*, et est mise au niveau LOW avec l'instruction *digitalWrite*, la broche est à 0 volts. Dans cet état, elle peut fournir un courant, par exemple pour allumer une LED qui est connectée en série avec une résistance au +5 volts, ou sur une autre broche configurée en sortie et mise au niveau HIGH.

Les broches numériques peuvent être utilisées soit en mode INPUT (en entrée), soit en mode OUTPUT (en sortie). Modifier le mode de fonctionnement d'une broche du mode

INPUT (entrée) en mode OUTPUT (sortie) avec l'instruction *pinMode()* change complètement le comportement électrique de la broche.

Nous insistons tout de même sur les fonctions implantées pour la gestion des entrées/sorties de la carte :

Tableau III.2 : Fonction d'entrées et sorties numériques et analogiques.

Fonction	***Description (partielle)***
pinMode()	Configuration des broches numériques en entrée ou en sortie.
digitalWrite()	Pour une broche numérique configurée en sortie, cette instruction permet de mettre son niveau logique à HAUT ou BAS.
digitalRead()	Lecture du niveau logique (HAUT ou BAS) d'une broche numérique configurée en entrée.
analogWrite()	La carte ne possède pas de Convertisseur Numérique Analogique. Il n'est donc théoriquement pas possible d'obtenir une tension continue qui soit l'image d'une valeur numérique (pas de sortie analogique). Il est par contre possible d'en reproduire le comportement en générant un signal rectangulaire (PWM ou MLI) de rapport cyclique réglable. Beaucoup de composants (LED, moteur électrique à courant continu) agissent comme des filtres passe-bas et fonctionnent comme s'ils étaient alimentés sous une tension continue égale à la valeur moyenne du signal.
analogRead()	Lecture de la valeur de la tension présente sur une entrée analogique (A0 à A5 sur la Uno). La fonction retourne la valeur issue de la Conversion Analogique Numérique, soit une valeur comprise entre 0 et 1023 (convertisseur 10 bits).

III. 6. Conclusion

Par sa simplicité d'utilisation, Arduino est utilisé dans beaucoup d'applications comme l'électronique industrielle et embarquée, le modélisme, la domotique mais aussi dans des domaines différents, On constate sur la toile, que par le biais d'Arduino, de nombreux curieux redécouvrent l'électronique

Dans notre réalisation on utilise un suiveur de commande numérique à l'aide de la carte Arduino qui connaît à tout moment la position du soleil dans le ciel avec une horloge et les équations de trajectoire de soleil.

Chapitre IV

Réalisation de suiveur solaire

IV. 1. Introduction

Un suiveur solaire est une installation solaire qui se base sur le principe du tournesol. Il suit le soleil dans sa course afin d'augmenter l'exposition des panneaux solaires et donc leur rendement.

Dans ce chapitre, nous présentons pas à pas les étapes de conception et réalisation d'un suiveur solaire commander par un Arduino Uno.

Cahier de charge :

On va réaliser un suiveur solaire de deux axes contrôlé par un Arduino Uno et nous allons utiliser la méthode de suivi avec les équations qui utilisent le rapport entre Latitude d'un lieu, date et heure du jour.

et on va utiliser un shield horloge DS1302, la rotation en deux axes ce fait par deux moteur pas à pas, un moteur unipolaire PM55L-048 pour actionner l'axe de l'élévation avec 48 pas/tour et moteur pas à pas hybride SPH-54AB pour actionner l'axe de l'azimut avec 200 pas/tour avec des roues dentées de différents diamètres pour la réduction des pas de moteurs et avoir plus de précision.

IV. 2. Principe de fonctionnement d'un suiveur solaire

Son principe de fonctionnement est de s'orienter vers le Soleil tout au long de la journée, ce qui a pour effet d'augmenter la production d'énergie de manière substantielle, En effet, la position du soleil varie constamment, à la fois pendant la journée, mais aussi pendant les différentes périodes de l'année. Le suiveur solaire permet ainsi de placer au mieux le panneau par rapport au positionnement perpendiculaire au rayonnement du Soleil.

L'idée derrière est que le surplus d'énergie produite grâce à la meilleure orientation du panneau sera plus important que l'énergie perdue pour orienter le panneau de manière optimale. En effet, un tracker peut augmenter le rendement des panneaux solaires jusqu'à 40% par rapport à une installation fixe orientée plein Sud.

Suivre le soleil peut se faire sur deux axes : en azimut d'est en ouest, à mesure de l'avancée de la journée et en hauteur selon la saison et l'avancée de la journée aussi. L'idéal est d'utiliser un suiveur solaire à deux axes, mais il en existe aussi avec un seul typiquement avec un suivi seulement en azimut, l'angle par rapport au sol étant fixé selon l'optimum local, qui dépend de la latitude.

Dans notre étude, nous présentons un système de poursuite solaire à deux axes où la partie mécanique a été minutieusement calculée et vérifiée pour assurer une transmission d'une faible puissance motrice via des mécanismes de réduction appropriés. Le premier axe celui qui fait le mouvement Nord Sud est le deuxième l'Est Ouest.

Le système que nous avions étudié et réalisé, tenait compte de la zone géographique où l'on était pour choisir son angle d'inclinaison, On prend notre ville MEDEA comme un lieu pour placer notre tracker ou suiveur solaire, le soleil entame une course d'Est en Ouest avec variation de sa hauteur par rapport à un point fixe dans ce lieu.

Site géographique	**Médéa**		Pays	**Algeria**
Situation	Latitude	36.3°N	Longitude	2.8°E
Temps défini comme	Temps solaire		Altitude	907 m
Horizon	Hauteur moyenne	0.0°	Facteur sur diffus	1.00
	Facteur sur albédo	100 %	Fraction d'albédo	1.00

Hauteur [°]	0.0	0.0	0.0	0.0
Azimut [°]	-120	-40	40	120

Source : PVsys

Figure IV.1 : Les coordonnées géographiques de site MEDEA.

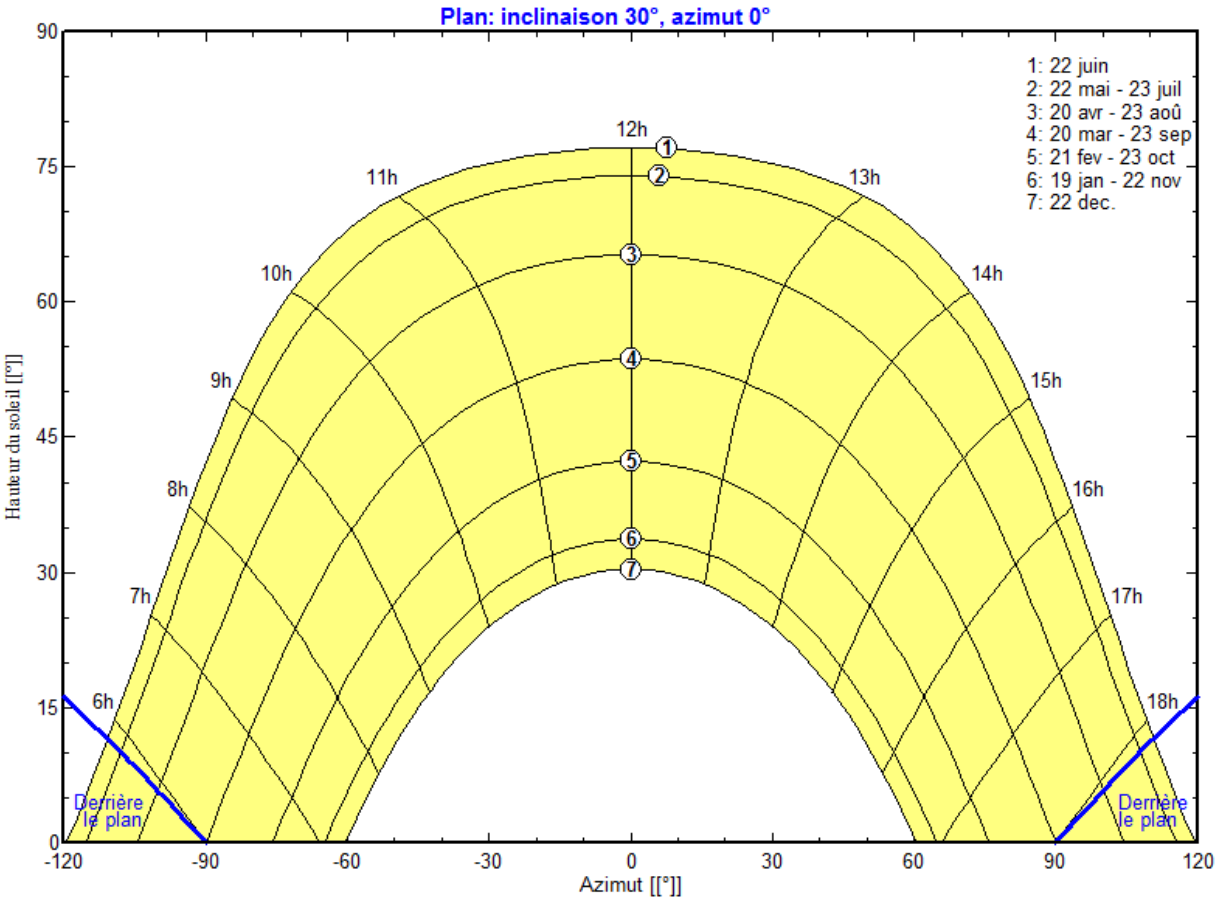


Figure IV.2 : Trajectoire du soleil de site MEDEA. [*PVsys*]

IV. 3. Câblage de fonctionnement d'un suiveur solaire contrôlé par un Arduino

Chacune des broches E / S numériques sur l'Arduino peut être une entrée Sink absorbe le courant provenant du capteur ou une entrée Source débite le courant dans le capteur de 40mA. Et en poussant l'Arduino a dépassé ces limites peut l'endommager, Si on a besoin d'obtenir un peu plus de puissance dans nos sorties le ULN2803 est un autre composant utile.

Ce circuit intégré s'appelle Darlington ULN2803, capable de faire circuler 500mA depuis une alimentation 50V et intègre des diodes anti-retour aussi appelée diode en roue libre, ce qui permet de piloter des bobinages. Cela permet au petit microcontrôleur (ou micro-ordinateur) de contrôler facilement des solénoïdes, relais, moteur continu (dans un seul sens) et moteurs pas-à-pas unipolaire. Les entrées peuvent être pilotées en logique 3.3V ou 5V.



Source: Eight Darlington array
Figure IV.3: Darlington ULN2803.

On va commander deux moteurs pas à pas de 1000mA donc on a besoin de 2 circuit intégrer ULN2803 pour avoir ce courant. On utilisant aussi les LED à la sortie de chaque bobine de moteurs pas a pas pour voir les séquences d'excitation des bobines, le schéma suivant explique l'emplacement de chaque composant :

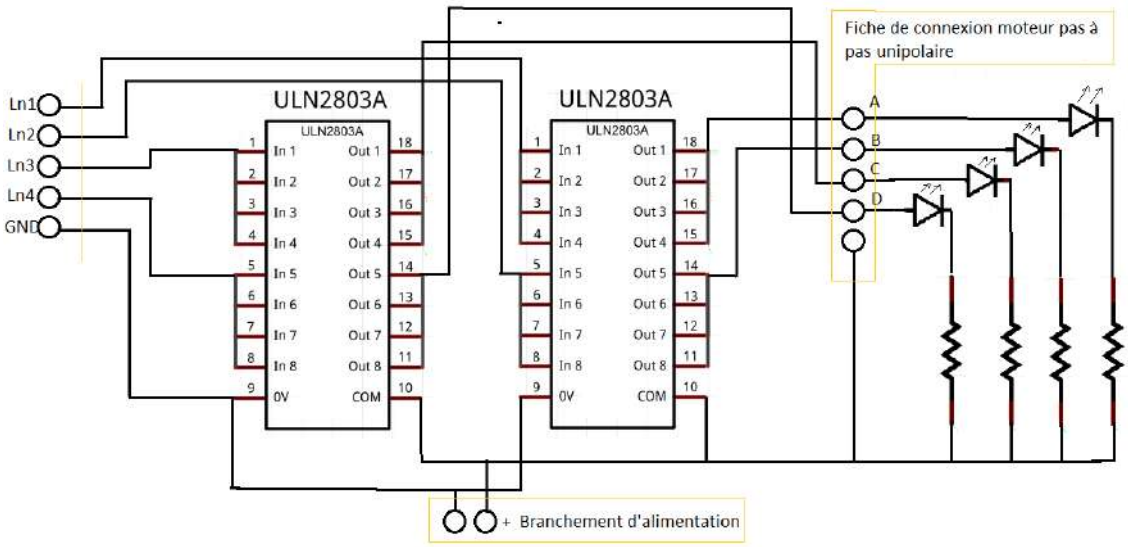


Figure IV.4 : Carte de commande de moteur pas à pas unipolaire.

Il existe différentes sortes et tailles de plaques d'expérimentation, mais le principe d'utilisation reste le même dans tous les cas, On a utilisé une plaque prototype pour circuit imprimé 5cm/7cm déjà avec des trous, on place des composants électroniques dans certains des trous présents, on les soude, puis on relie entre elles les pattes des composants qui doivent l'être.

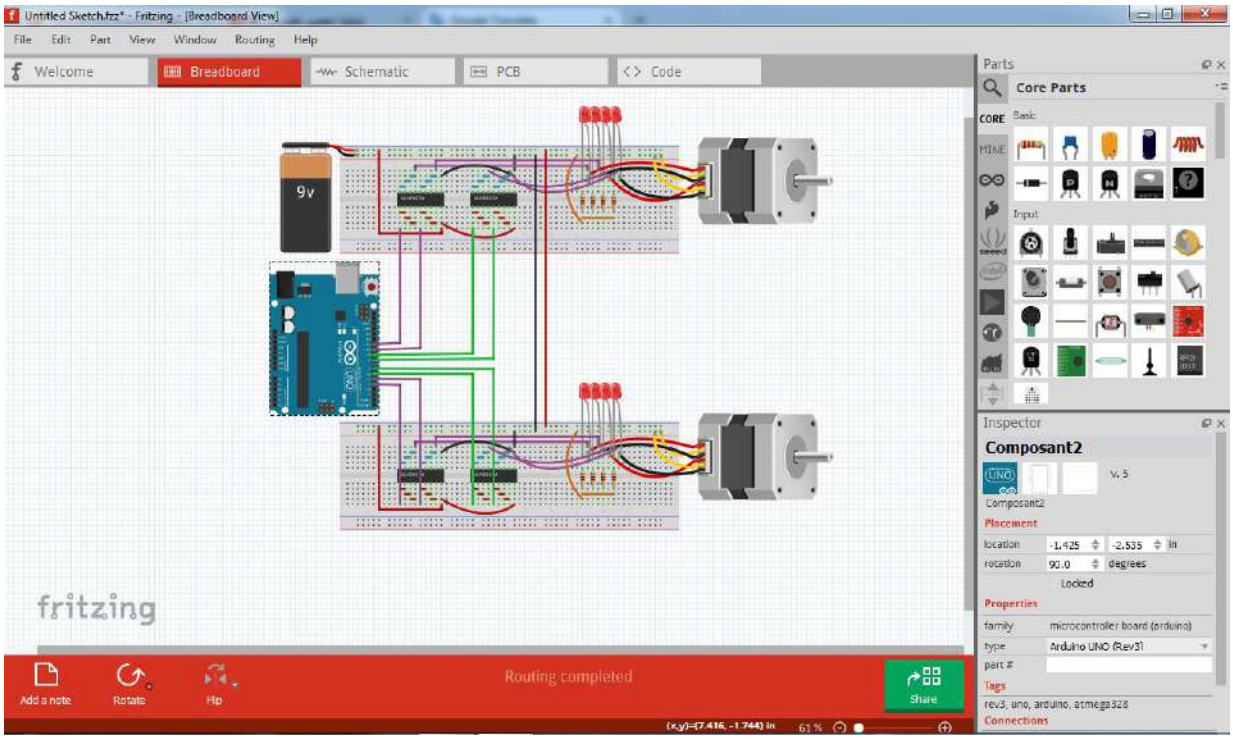


Figure IV.5 : prototypage de la carte de commande avec Fritzing.

On utilisant aussi des supports pour les circuits intégré pour faciliter le montage ou démontage, en cas de changement.

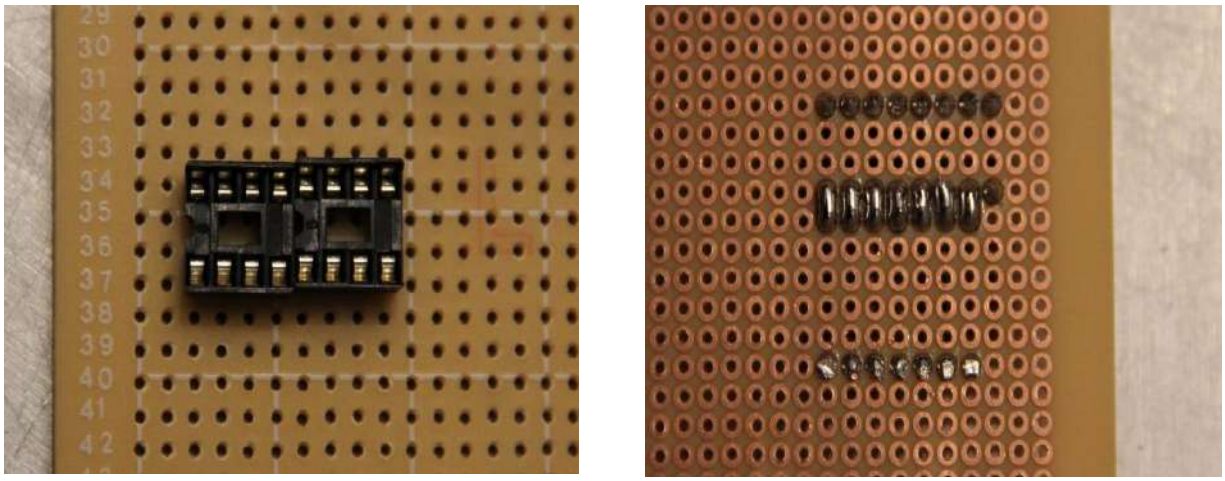


Figure IV.6.a : Soudage des supports pour ULN2803.

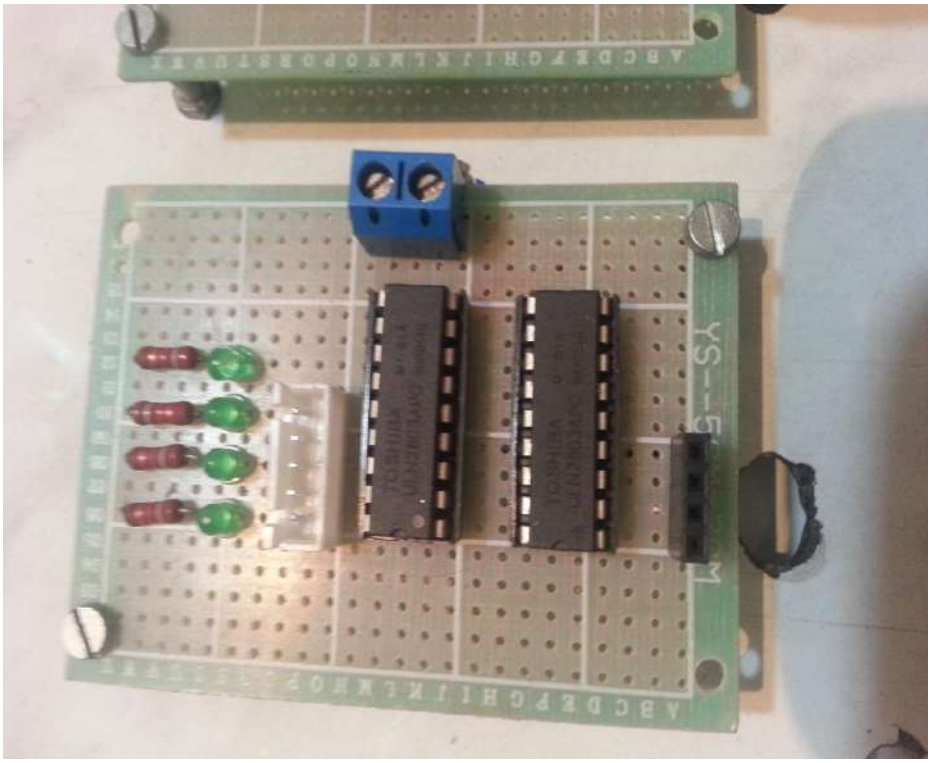


Figure IV.6.b : Carte de commande moteur pas à pas.

On a monté deux fins de courses pour les deux moteurs d'azimut et d'hauteur solaire, la fin de course 'A' pour le moteur d'azimut est branchée sur pin 6 et l'autre en pin 7 d'Arduino.

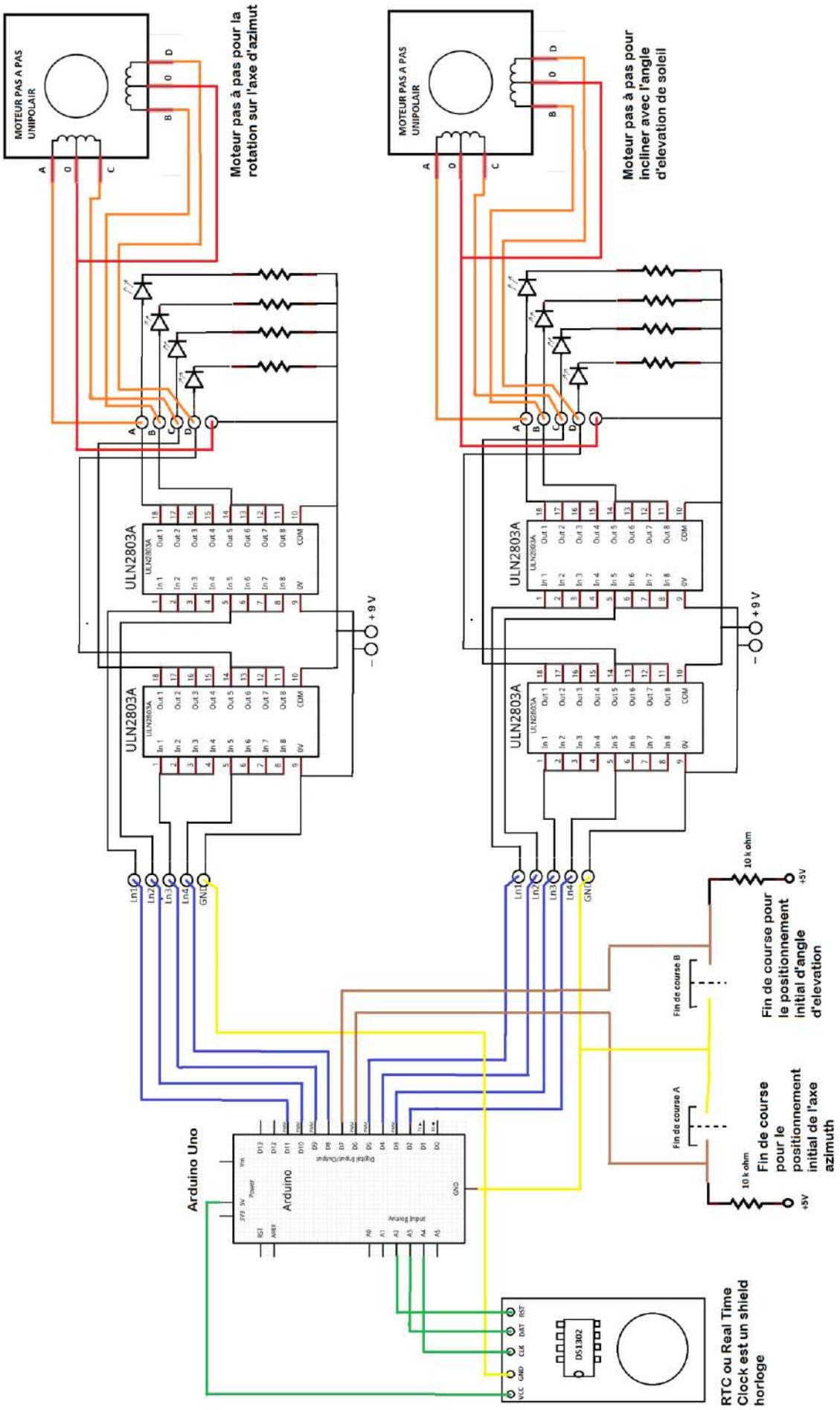


Figure IV.7. Schema electrique de commande d'un suiveur solaire a deux axes de rotation par un Arduino UNO

IV. 3. Etude mécanique et la réalisation d'un suiveur

La base de cette étude porte sur le fait que la course du soleil est connue depuis des lustres et qu'à tout moment de la journée on connaît sa position pour un lieu donné, contrairement à ce que l'on trouve dans beaucoup de publications, on basera la poursuite du soleil non pas sur des capteurs de 'lumière', mais sur le 'rapport' entre Latitude d'un lieu, date et heure du jour.

La taille du panneau étant relativement faible, 250x250x8mm, 0.5Kg, l'asservissement mécanique se fera par deux moteurs pas à pas suffisamment dimensionnés pour, d'une part, vaincre les inerties et le poids du panneau, et d'autre part, pouvoir les débrayer électriquement pour qu'ils ne consomment pas trop de courant en poursuite du soleil.

L'électronique sera constituée par microcontrôleur Arduino UNO, assisté par un module RTC DS1302 gérant la date et l'heure, et une carte de commande des moteurs pas à pas. Pour le positionnement de panneau sera géré de la manière suivante :

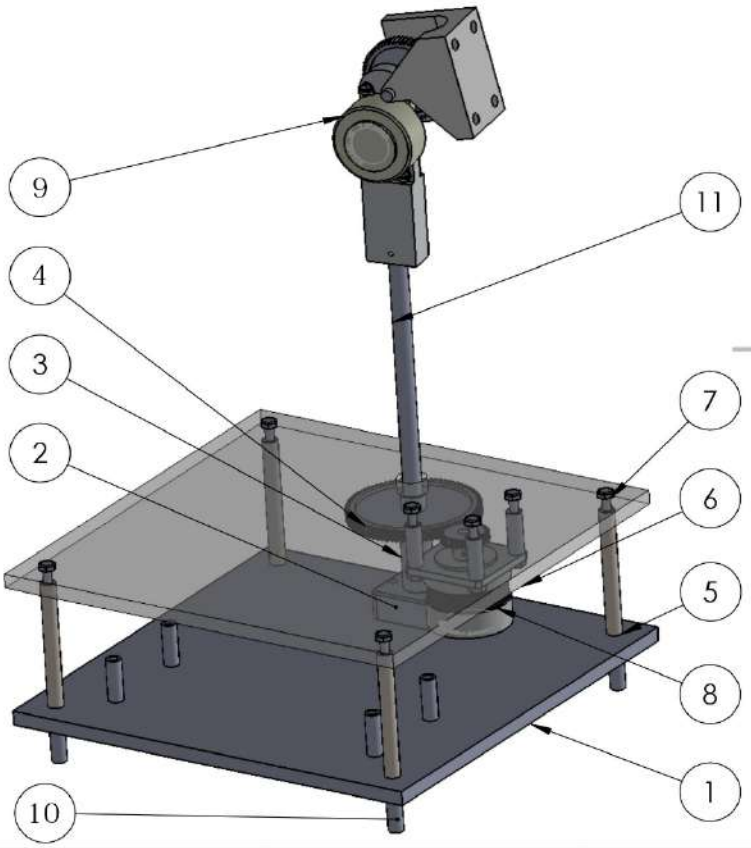
- Initialement, le panneau est ramené en butée ou fin de course 'A' Est comme un repère pour démarrer sur l'axe d'azimut.
- le panneau est positionner vertical au sol par la fin de course 'B' comme un repère pour démarrer sur l'axe d'hauteur solaire.
- Toutes les 10 minutes, le panneau effectue une rotation de l'Est vers l'Ouest et un degré d'inclinaison entre Sud et Nord calculé par la partie commande.

Le system mécanique comprend :

- 1 moteur pas à pas unipolaire PM55L-048 avec 48 pas/tour
- 1 moteur pas à pas hybride SPH-54AB avec 200 pas/tour
- 2 fins de course.
- Un jeu de roues dentées pour l'axe d'azimut de 14 dents et 60 dents
- Un jeu de roues dentées de 30 dents et 82 dents permet en outre une réduction
- Une table d'aluminium pour fixer le support d'axe d'azimut et les composants électrique.
- Une table de verre pour la protection de composants électrique et un emplacement de moteur pas à pas.

IV. 3. 1. Croquer de montage.

NO. ARTICLE	NUMERO DE PIECE	QTE
1	table d'aluminuim	1
2	Support d'axe de rotation azimut	1
3	Bague de centrage d'axe	1
4	Roue dentée ∅88 82 dents	1
5	boulon de fixation long	4
6	table de verre	1
7	boulon M6	7
8	moteur pas a pas 200pas/tour pour retation ESt-ouest	1
9	assemblage de tete	1
10	bague de fixation	11
11	axe de rotation Est - Ouest	1



SAUF INDICATION CONTRAIRE:
LES COTES SONT EN MILLIMETRES
ETAT DE SURFACE:
TOLERANCES:
LINEAIRES:
ANGULAIRES:

FINITION:

CASSER LES ANGLES VIFS

NE PAS CHANGER L'ECHELLE

REVISION

	NOM		DATE
AUTEUR	BENKORTBI	Abdelhakim	10/06/2015
	FERROUKH	Mohamed lamine	
APPR.			
FAB.			
QUAL.			

TITRE: Montage d'un suiveur solair a deux axe

MATERIAU:

No. DE PLAN: Assemblage tt

A4

MASSE:

ECHELLE:1:10

FEUILLE 1 SUR 1

Figure IV.8 : Croquer de suiveur solaire dessiné sur Solidworks.

La course de moteur pas à pas chargé de l'Azimut sera limitée à l'Ouest et à l'Est. En effet, le panneau solaire peut tourner 360° avec une fin de course au-dessus de table d'aluminium pour repérer le point zéro de démarrage du Nord.

La course du moteur pas à pas chargé de la Hauteur sera limitée en bas à 0° et en haut à 120°, le zéro étant le plan horizontal de sol.

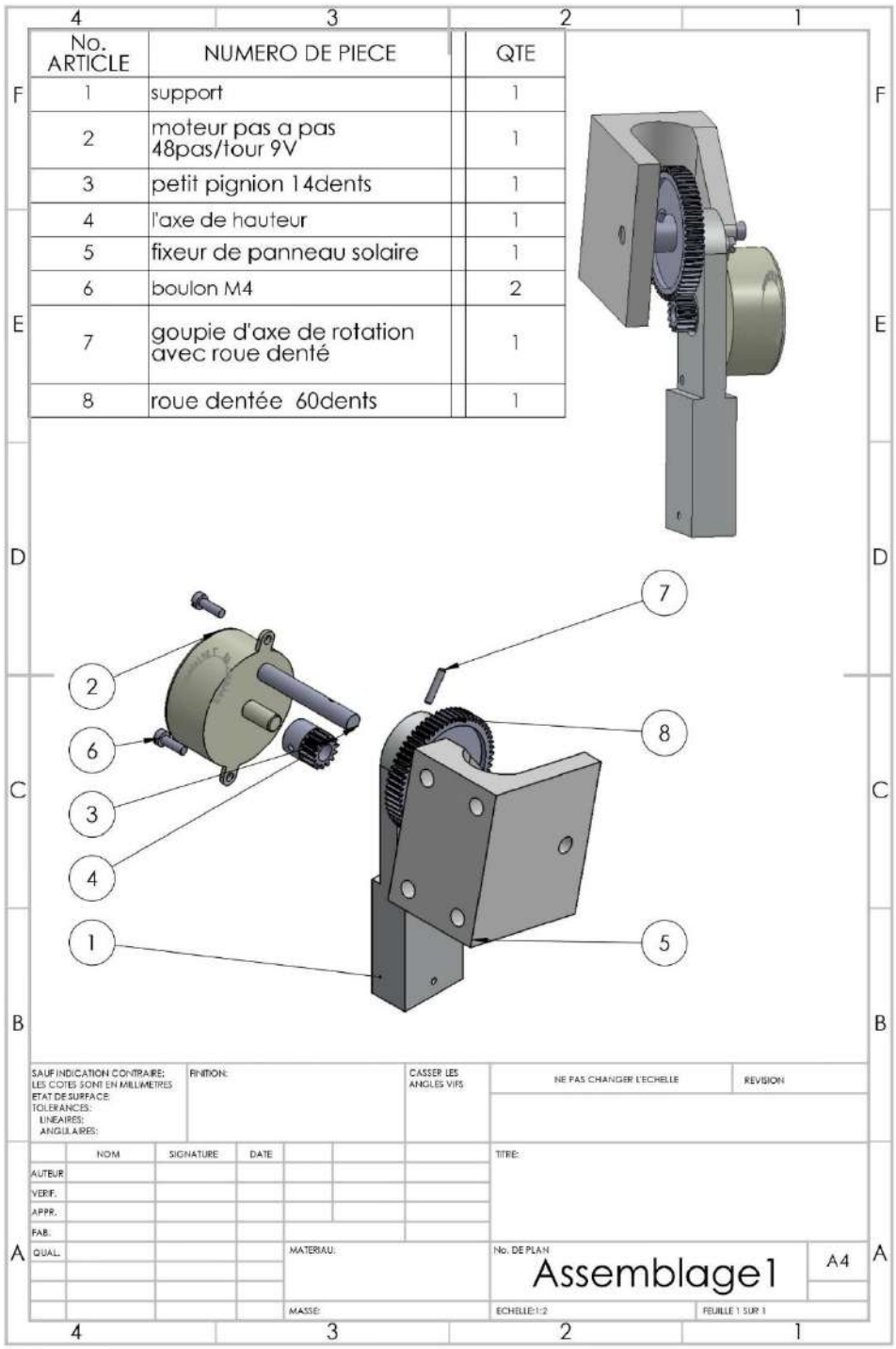


Figure IV.9 : Support de rotation pour l'angle d'hauteur de soleil.

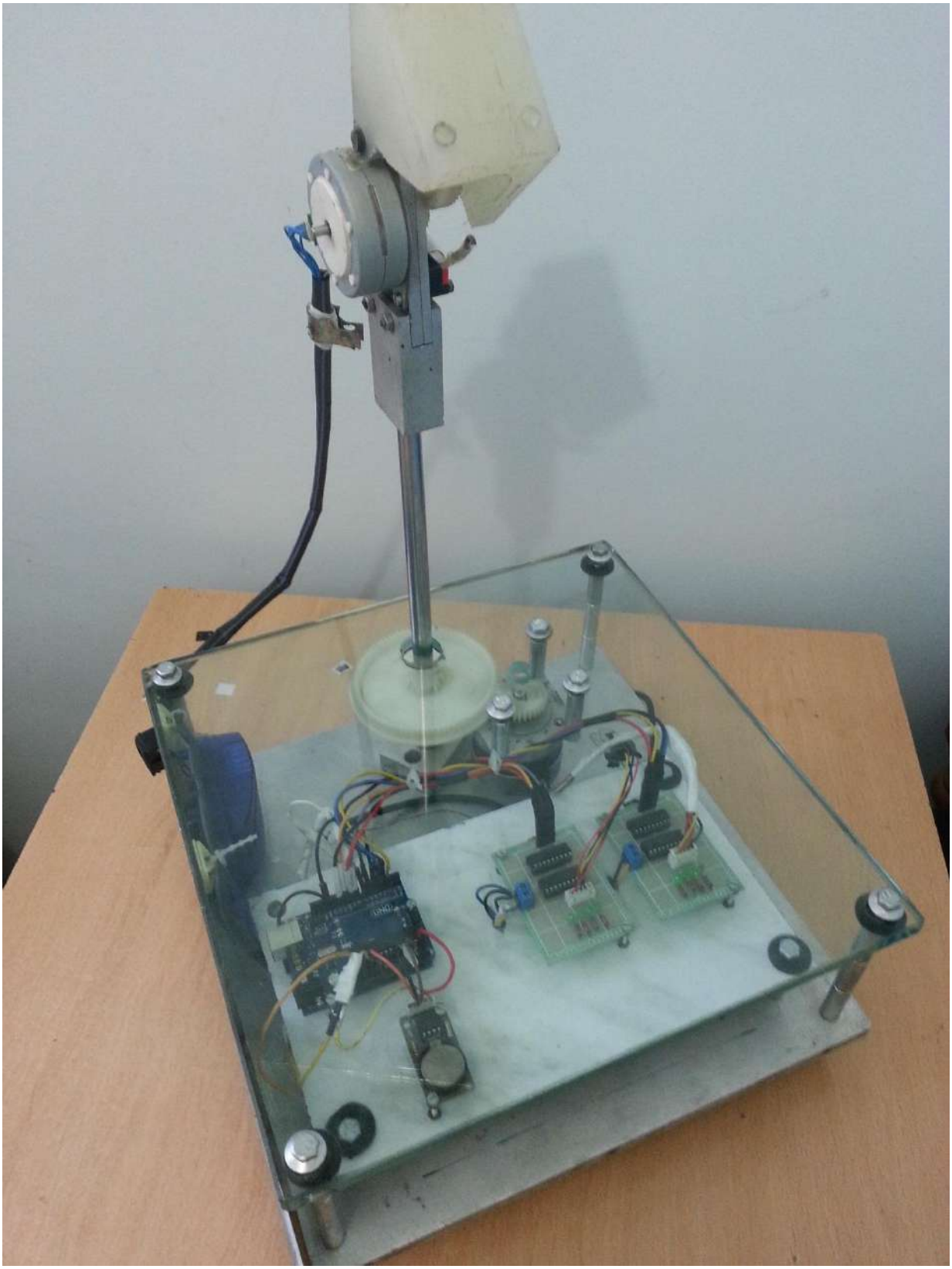


Figure IV.10 : la réalisation du suiveur solaire qui nous fait.

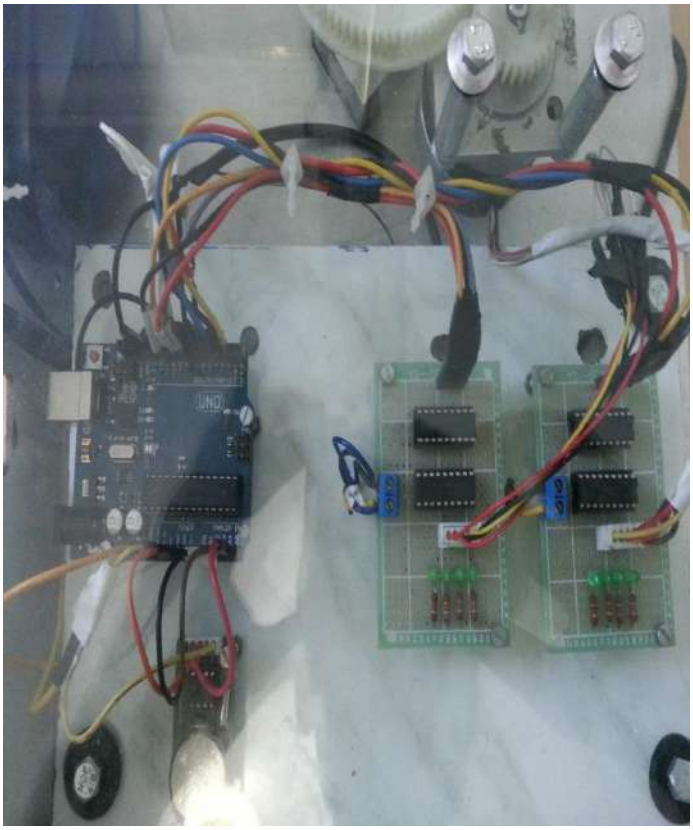





Figure IV.11 : les différents composants du suiveur solaire qui nous réalisons.

IV. 4. Organigramme de suiveur solaire a deux axes

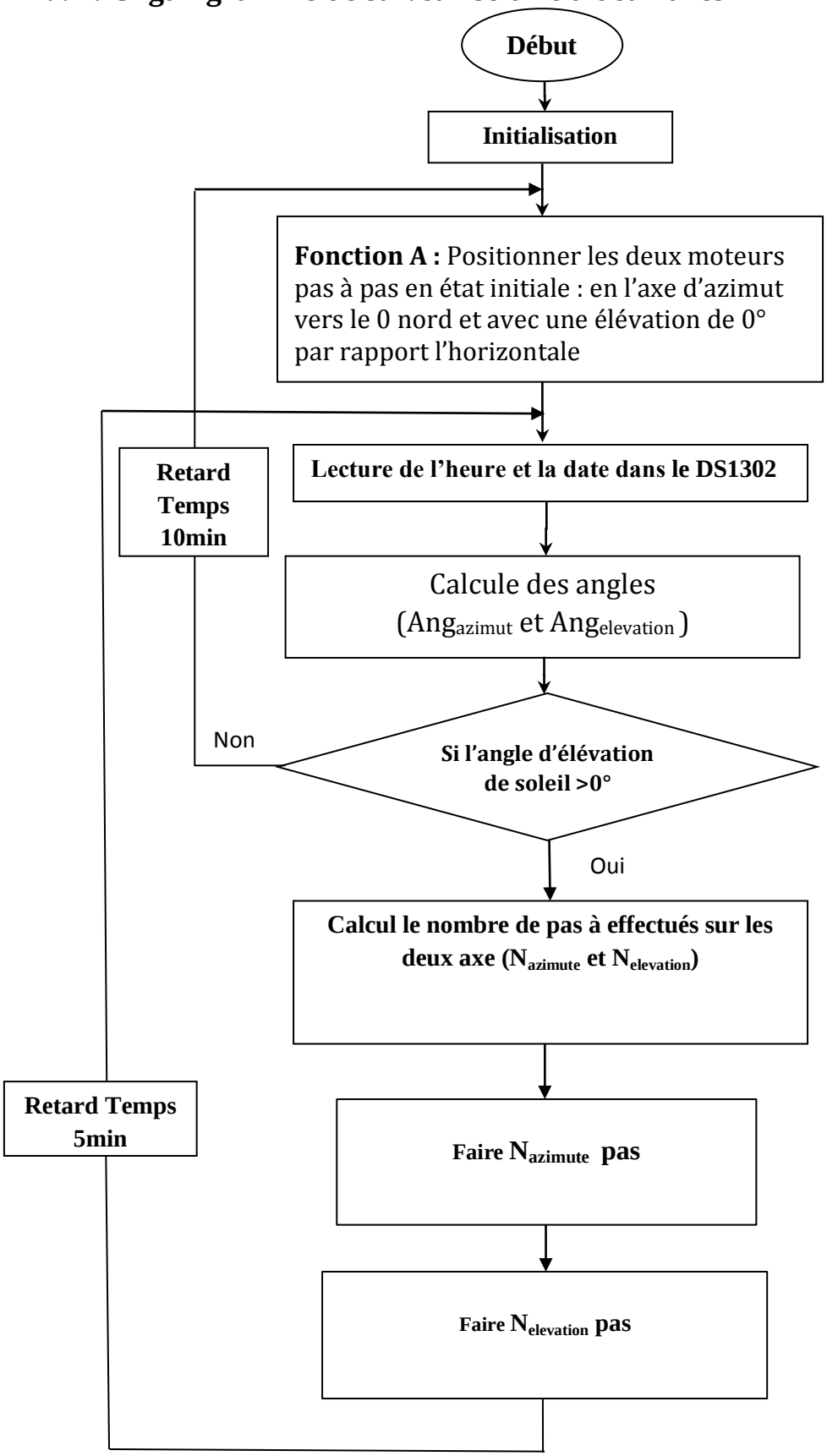


Figure IV.12 : Organigramme d'un suiveur solaire a deux axes avec l'horloge DS1302.

Organigramme de fonction A :

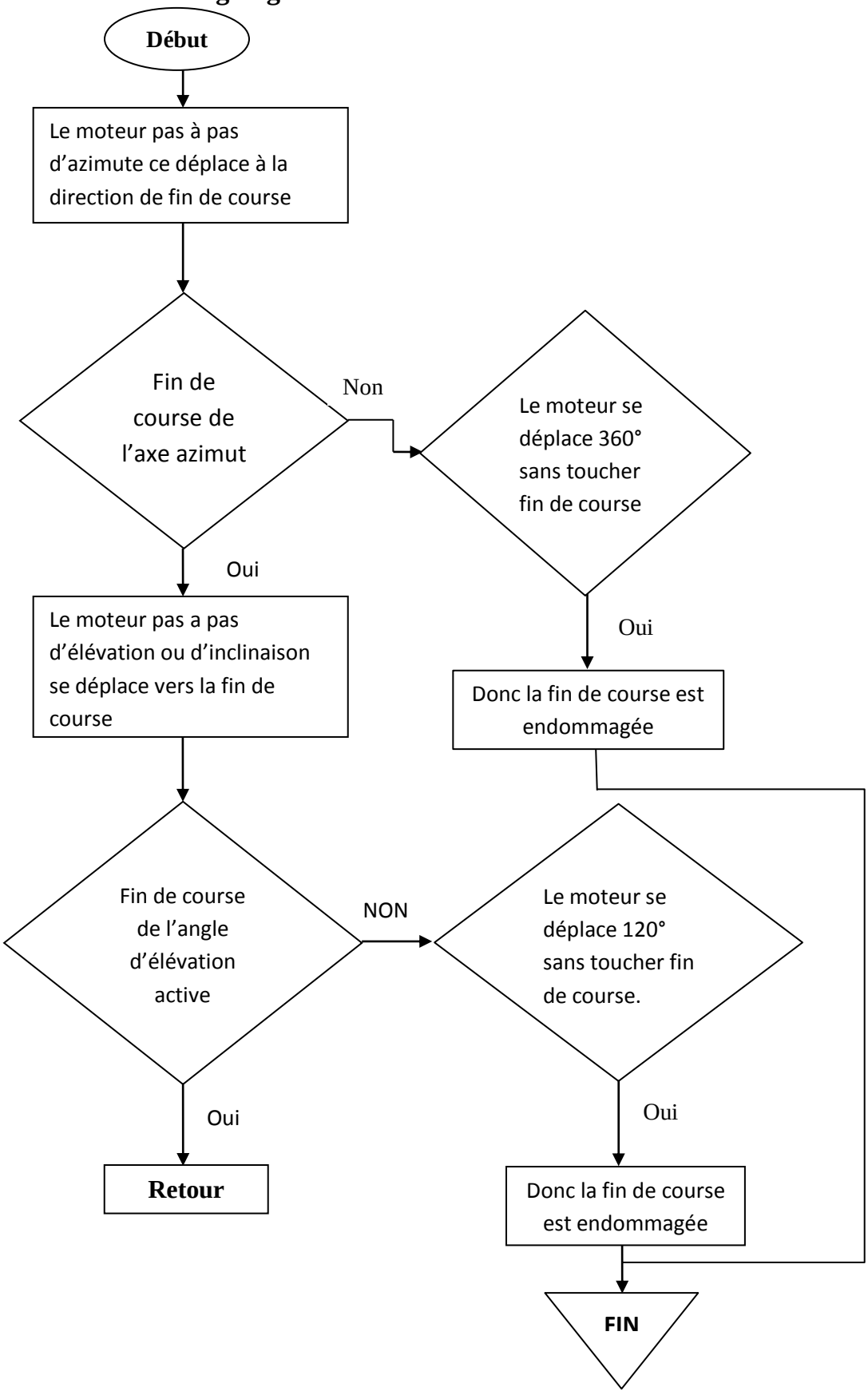


Figure IV.13 : Organigramme des fins de course.

IV. 5. Programmation

IV. 5. 1. Programmation de carte RTC DS1302

La programmation de la carte s'effectue via une interface logicielle d'Arduino en langage C. Une fois le programme écrit, il est envoyé sur la carte qui peut ensuite fonctionner en autonomie (sans connexion avec l'ordinateur).

Premièrement on va déclarer la date, et le temps et les traverser vers la carte Arduino pour l'enregistrer circuit intégrer DS1302.

```
// On Install le module DS1302 dans la librairie de le  logiciel de programmation
// On envoi la date et le temps  par le port serial USB
// la connexion avec DS1302 et Arduino
//         CE pin    -> Arduino Digital A2
//         I/O pin   -> Arduino Digital A3
//         SCLK pin  -> Arduino Digital A4
#include <DS1302.h>
// Init the DS1302
DS1302 rtc(A2, A3, A4);
void setup()
{
  // On pose l'horloge en mode mise en fonctionnement, et on désactive la protection en écriture
  rtc.halt(false);
  rtc.writeProtect(false);
    // la connexion avec une porte série USB
  Serial.begin(9600);
  // Les données saisi  à nouveau vont supprimer les derniers enregistré dans le module DS1302
  rtc.setDOW(FRIDAY);        // On pose le jour de semaine
  rtc.setTime(12, 0, 0);     // on pose le temps  12:00:00 (24hr format)
  rtc.setDate(12, 6, 2015);   // on pose la date a 12 juin 2015
}
void loop()
{
  // envoyer le jour de la semaine
  Serial.print(rtc.getDOWStr());
  Serial.print(" ");
    // envoyer la date
```

```
  Serial.print(rtc.getDateStr());
  Serial.print(" -- ");
  // envoyer le temps
  Serial.println(rtc.getTimeStr());
    // attendre une seconde pour l'envoi prochain
  delay (1000);
}
```

La mise en route de module DS1302

```
// On active la protection d'écrire et en enlève la partie de déclaration de date et temps
// la connexion avec DS1302 et Arduino
//         CE pin    -> Arduino Digital A2
//         I/O pin   -> Arduino Digital A3
//         SCLK pin  -> Arduino Digital A4
#include <DS1302.h>
// Init the DS1302
DS1302 rtc(A2, A3, A4);
void setup()
{
   // la connexion avec une porte série USB
  Serial.begin(9600);
  }
void loop()
{
  // envoyer le jour de la semaine
  Serial.print(rtc.getDOWStr());
  Serial.print(" ");
    // envoyer la date
  Serial.print(rtc.getDateStr());
  Serial.print(" -- ");
  // envoyer le temps
  Serial.println(rtc.getTimeStr());
    // attendre une seconde pour l'envoi prochain
  delay (1000);
}
```

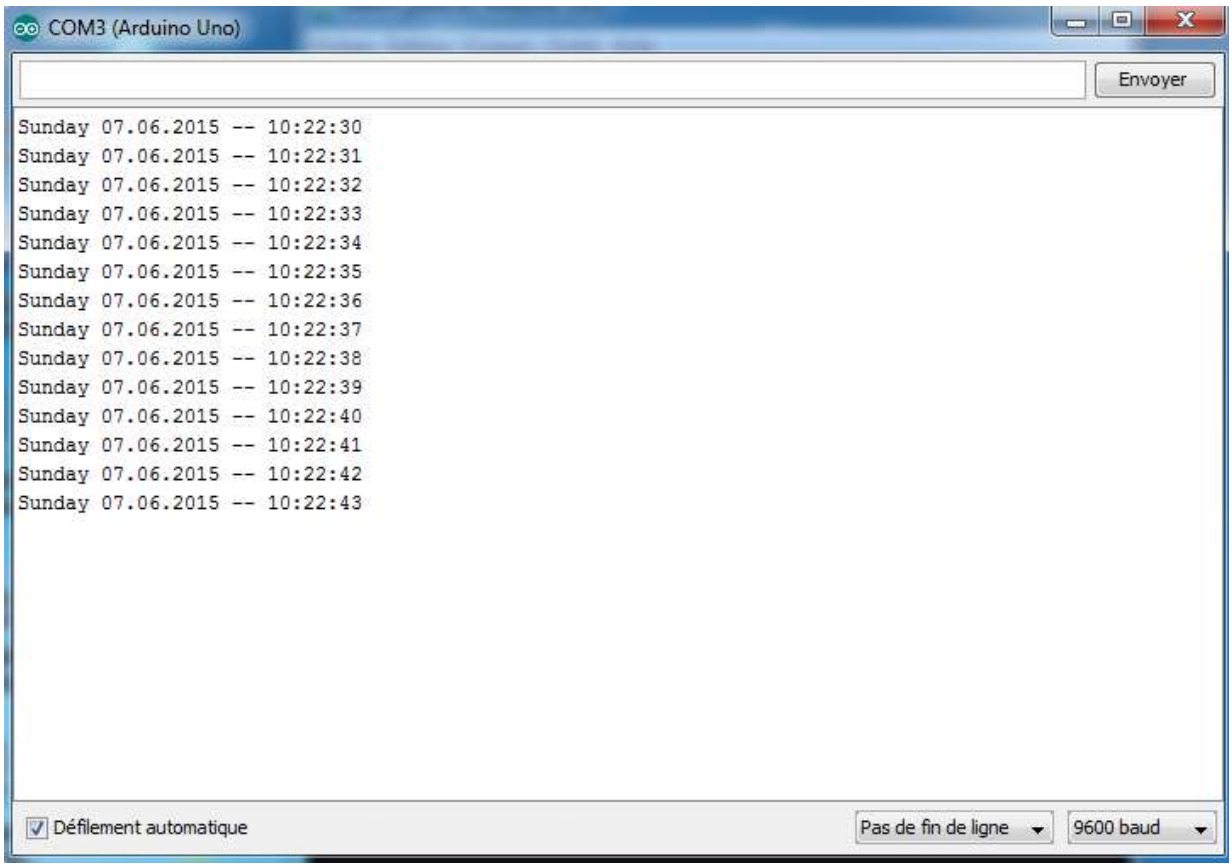


Figure IV.14 : Moniteur série résultant de la carte DS1302.

IV. 5. 1. Programmation de carte Arduino pour le control d'un suiveur solaire a 2 axes

```
// On peut avec ce programme savoir la position du soleil en cours de l'année avec les équations géométriques terre-soleil
// le module RTC DS1302 qui conserve la date et le temps sans alimenter l'Arduino
// DS1302: CE pin    -> Arduino Digital A2
// I/O pin   -> Arduino Digital A3
// SCLK pin  -> Arduino Digital A4
#include <math.h>
#include <DS1302.h>
#include <Stepper.h>
#define pi    3.14159265358979323846
#define twopi (2*pi)
#define rad   (pi/180)
#define EarthMeanRadius     6371.01      // en kilometre
#define AstronomicalUnit    149597890      // en kilometre
#define AZIMUTH_SPEED 4
#define ELEVATION_SPEED 4
// les pins des moteurs
#define motorPin1  2     // IN1 en carte de commande1
#define motorPin2  3     // IN2 en carte de commande1
```

```
#define motorPin3  4     // IN3 en carte de commande1
#define motorPin4  5     // IN4 en carte de commande1
#define motorPin5  8     // IN1 en carte de commande2
#define motorPin6  9     // IN2 en carte de commande2
#define motorPin7  10    // IN3 en carte de commande2
#define motorPin8  11    // IN4 en carte de commande2
// Initialisation de DS1302
DS1302 rtc(A2, A3, A4);
Time t;
const int stepsPerRevolutionAzimuth = 576; // le nombre de pas par tour sur l'axe d'azimut avec la réduction utilisé
const int stepsPerRevolutionElevationAngle = 205; // le nombre de pas sur le moteur pas a pas d'angle d'élévation avec la réduction utilisé
// les pins pour contrôler le moteur pas a pas pour la rotation azimut :
Stepper azimuthmoteur(stepsPerRevolutionAzimuth, motorPin1, motorPin2, motorPin3, motorPin4);
// les pins pour contrôler le moteur pas a pas pour l'angle d'élévation solaire :
Stepper elevationmoteur(stepsPerRevolutionElevationAngle, motorPin5, motorPin6, motorPin7, motorPin8);
// le fin de course de rotation sur l'azimut branché en pin 6  le panneau en zéro nord
// le fin de course d'angle d'élévation en pin 7 le panneau en état perpendiculaire avec le sol
const int azimuthPin = 6;
const int elevationPin = 7;
//Position des moteurs pas à pas
long azimuthCurrentPosition = 0;     // zero azimut
long elevationCurrentPosition = 0;     // zero elevation
float Year; //année
float Month; //mois
float Day; //jours
float Hours;//heures
float Minutes; //minutes
float Longitude = 2.75 ; //longitude de MEDEA
float Latitude = 36.26; //Latitude de MEDEA
//Program Variables
float ZenithAngle;
float Azimuth;
float RightAscension;
float Declination;
float Parallax;
float ElevationAngle;
float ElapsedJulianDays;
float DecimalHours;
float EclipticLongitude;
float EclipticObliquity;
//=====================================
void setup() {
  Serial.begin(9600);
  pinMode(azimuthPin, INPUT_PULLUP);
  pinMode(elevationPin, INPUT_PULLUP);
  azimuthmoteur.setSpeed(AZIMUTH_SPEED);
```

```
  elevationmoteur.setSpeed(ELEVATION_SPEED);
  //positionner le moteur d'azimute en point initial
  long tmpSteps=0;
  do
  {
          azimuthmoteur.step(-1);
          if (++tmpSteps > stepsPerRevolutionAzimuth) //plus d'une tour veut dire que le fin
de course ne marche pas
          {
                  Serial.println("Le fin de course de moteur Azimut ne fonctionne pas!suiveur
arreter");
                  do{}while (true); //cycle interminer le suiveur s'arretera
          }
  } while (digitalRead(azimuthPin)== HIGH);
  //maintenant on es dans la position initial
  azimuthCurrentPosition=0;
   //positionner le moteur d'élévation en point initial
  tmpSteps=0;
  do
  {
          elevationmoteur.step(-1); //marcher a l'inverse jusqu'à point zéro
          if (++tmpSteps > stepsPerRevolutionElevationAngle) //plus d'une tour veut dire que
le fin de course ne marche pas
          {
                  Serial.println("Le fin de course de moteur d'elevation ne fonctionne
pas!suiveur arreter");
                  do{}while (true); //cycle interminer le suiveur s'arrêtera
          }
  } while (digitalRead(azimuthPin)== HIGH);
  //maintenant on es en position initial
  elevationCurrentPosition=0;
}
//=====================================
void sunPos() {
// prendre les données de  DS1302
  t = rtc.getTime();
Year = (t.year, DEC); //année
Month = (t.mon, DEC); //mois
Day = (t.dow, DEC); //jours
Hours = (t.hour, DEC)-3;//heures
Minutes = (t.min, DEC); //minutes
// Les axes variables
  float dY;
  float dX;
 // On calcule le nombre des jours prenent le premier janvier comme un repere on l'appelle
'JulianDate'
  // donc JD 2451545.0, est le 1 Janvier 2015 temps universal
  float JulianDate;
  long int liAux1;
  long int liAux2;
```

```
// Calculer le temps dans la journée en universel temps dicimal heures
DecimalHours = Hours + (Minutes / 60.0);
// calculer le nombre de jour
liAux1 = (Month - 14) / 12;
liAux2 = (1461 * (Year + 4800 + liAux1)) / 4 + (367 * (Month
      - 2 - 12 * liAux1)) / 12 - (3 * ((Year + 4900
                        + liAux1) / 100)) / 4 + Day - 32075;
JulianDate = (float)(liAux2) - 0.5 + DecimalHours / 24.0;
// Calculate difference between current Julian Day and JD 2451545.0
ElapsedJulianDays = JulianDate - 2451545.0;
// Calculez les coordonnées écliptique
// écliptique en radians sans limiter le degré de retation moins de 2*pi
float MeanLongitude;
float MeanAnomaly;
float Omega;
Omega = 2.1429 - 0.0010394594 * ElapsedJulianDays;
MeanLongitude = 4.8950630 + 0.017202791698 * ElapsedJulianDays; // en radians
MeanAnomaly = 6.2400600 + 0.0172019699 * ElapsedJulianDays;
EclipticLongitude = MeanLongitude + 0.03341607 * sin( MeanAnomaly )
            + 0.00034894 * sin( 2 * MeanAnomaly ) - 0.0001134
            - 0.0000203 * sin(Omega);
EclipticObliquity = 0.4090928 - 6.2140e-9 * ElapsedJulianDays
            + 0.0000396 * cos(Omega);
// Calculez les coordonnées célestes ( ascension droite et declinaison ) en radians
 float Sin_EclipticLongitude;
Sin_EclipticLongitude = sin( EclipticLongitude );
dY = cos( EclipticObliquity ) * Sin_EclipticLongitude;
dX = cos( EclipticLongitude );
RightAscension = atan2( dY, dX );
if ( RightAscension < 0.0 ) RightAscension = RightAscension + twopi;
Declination = asin( sin( EclipticObliquity ) * Sin_EclipticLongitude );
// Calculer Les coordonnées locales ( azimuth et l'angle zenith ) en degrés
float GreenwichMeanSiderealTime;
float LocalMeanSiderealTime;
float LatitudeInRadians;
float HourAngle;
float Cos_Latitude;
float Sin_Latitude;
float Cos_HourAngle;
GreenwichMeanSiderealTime = 6.6974243242 +
                  0.0657098283 * ElapsedJulianDays
                  + DecimalHours;
LocalMeanSiderealTime = (GreenwichMeanSiderealTime * 15
                + Longitude) * rad;
HourAngle = LocalMeanSiderealTime - RightAscension;
LatitudeInRadians = Latitude * rad;
Cos_Latitude = cos( LatitudeInRadians );
Sin_Latitude = sin( LatitudeInRadians );
Cos_HourAngle = cos( HourAngle );
ZenithAngle = (acos( Cos_Latitude * Cos_HourAngle
```

```
                    * cos(Declination) + sin( Declination ) * Sin_Latitude));
  dY = -sin( HourAngle );
  dX = tan( Declination ) * Cos_Latitude - Sin_Latitude * Cos_HourAngle;
  Azimuth = atan2( dY, dX );
  if ( Azimuth < 0.0 )
    Azimuth = Azimuth + twopi;
  Azimuth = Azimuth / rad;
  // correction parallele
  Parallax = (EarthMeanRadius / AstronomicalUnit)
           * sin(ZenithAngle);
  ZenithAngle = (ZenithAngle
  + Parallax) / rad;
  ElevationAngle = (90 - ZenithAngle); // On fait sortir l'angle d'élévation de l'angle zénith
}
//======================================
void loop() {
  sunPos(); //mise en route de programmepour calculer la position de soleil
  Serial.print("Elevation Angle:  ");
  Serial.println(ElevationAngle, 0); //Print Elevation (Vertical)
  Serial.print("Azimuth:  ");
  Serial.println(Azimuth, 0); //Print Azimuth (Horizontal)
   // envoyer le jour
  Serial.print(rtc.getDOWStr());
  Serial.print(" ");
  // envoyer la date
  Serial.print(rtc.getDateStr());
  Serial.print(" -- ");
  // envoyer le temps
  Serial.println(rtc.getTimeStr());
    long azimuthDelta=(Azimuth*stepsPerRevolutionAzimuth/360)- azimuthCurrentPosition;
  long elevationDelta=(ElevationAngle*stepsPerRevolutionElevationAngle/360)-
elevationCurrentPosition;
  //if (ElevationAngle < 0)
  // moteur pas a pas de l'azimut
  // On calcule l'angle pour rectifier la position de panneau par rapport le soleil on fait
  // l'angle a ajouter = ( l'angle totale de deplacement de soleil par rapport azimut - l'angle
deja parcouri)
  // nombre de step a ajouter = (Azimuth*576/360)- stepCount
  if ( azimuthDelta!= 0)
  {
          azimuthmoteur.step(azimuthDelta);
          azimuthCurrentPosition+=azimuthDelta;
          Serial.print("move azimuth:");
          Serial.println(azimuthDelta);

  }
   // moteur pas a pas de l'angle d'elevation
  // On calcule l'angle pour rectifier la position de panneau par rapport le soleil on fait
  // l'angle a ajouter = ( l'angle totale de deplacement de soleil par rapport l'horizontale -
l'angle deja parcouri)
```

```
  // nombre de step a ajouter = (ElevationAngle*205/360)- stepCount
    if ( elevationDelta!= 0)
  {
            elevationmoteur.step(elevationDelta);
            elevationCurrentPosition+=elevationDelta;
            Serial.print("move elevation:");
            Serial.println(elevationDelta);
  }
   delay (300000);
}
```

IV. 6. Conclusion

Dans ce chapitre nous avons présenté tous les étapes de la réalisation de notre suiveur solaire et tous les composants utilisé et le programme qui suivi la trajectoire de soleil avec précision, nous avons fait le travail de trois domaine différent (mécanique, électronique et programmation), qui n'a pas été facile et malgré les difficultés qui nous avons intercepté et la limitation de notre moyens pour faire des essaies de comparaison entre l'énergie produit par un panneau fixe et un panneau avec suiveur solaire, nous avons réalisé ce modeste projet.

Conclusion général

Le but principal de ce travail était la conception et réalisation d'un suiveur solaire de deux axes qui fonctionne et qui permettrait d'améliorer la puissance de sortie d'un module solaire par rapport à un module solaire fixe.

Tout d'abord, nous avons étudié le gisement solaire et spécialement en Algérie qui ont un bon ensoleillement et une durée considérable.

Aussi, le chemin du soleil toute la journée du lever jusqu'au coucher et de comprendre les équations et les différentes variables qui nous permet de suivre la trajectoire du soleil avec une haute précision.

Pour utiliser les équations que nous avons vues dans le premier chapitre, nous avons besoin de comprendre la façon d'intégrer ces équations dans le microcontrôleur Arduino pour déplacer le suiveur solaire en fonction du mouvement du soleil, nous avons choisi le microcontrôleur Arduino parce qu'il est un open source plate-forme électronique basée sur matériel et logiciel facile à utiliser, pour cela, nous expliquent comment utiliser Arduino et nous étudiés le langage de programmation qui permet Arduino de communiquer avec son environnement, à travers le programme Environnement de développement intégré « IDE » où nous pouvons mettre les instructions de commande pour exécuter les moteurs attachés à chaque axe de suiveur.

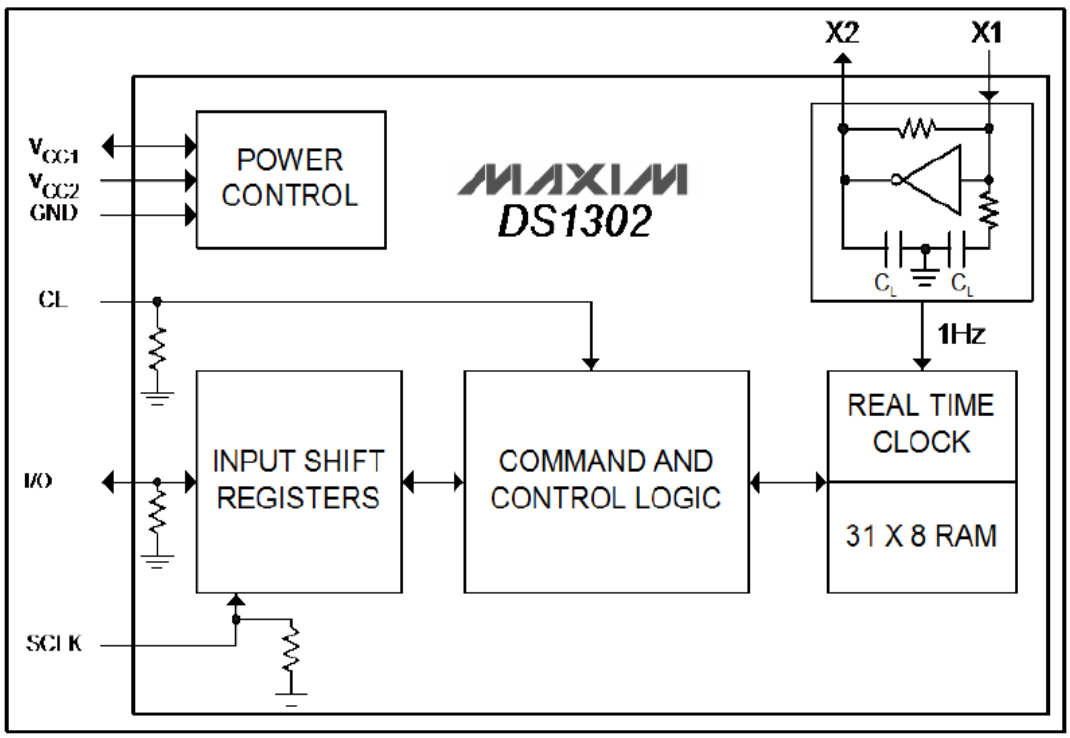
Comme nous le mentionnons précédemment, nous utilisons des moteurs pour actionner le suiveur, nous avons choisi les moteurs pas à pas, de sorte que nous expliquons comment les moteurs pas à pas fonctionner et leurs différents types, qui nous aident à réaliser une carte de commande simple pour commander les moteurs pas à pas.

Après toutes ces études, nous commençons la conception de chaque pièce qui composé la structure de notre suiveur solaire et le construire pour le tester.

Enfin, nous testons le programme que nous avons implanté dans l'Arduino sur le traquer solaire, il fonctionne bien et de suivre la trajectoire du soleil précisément que nous avons programmé.

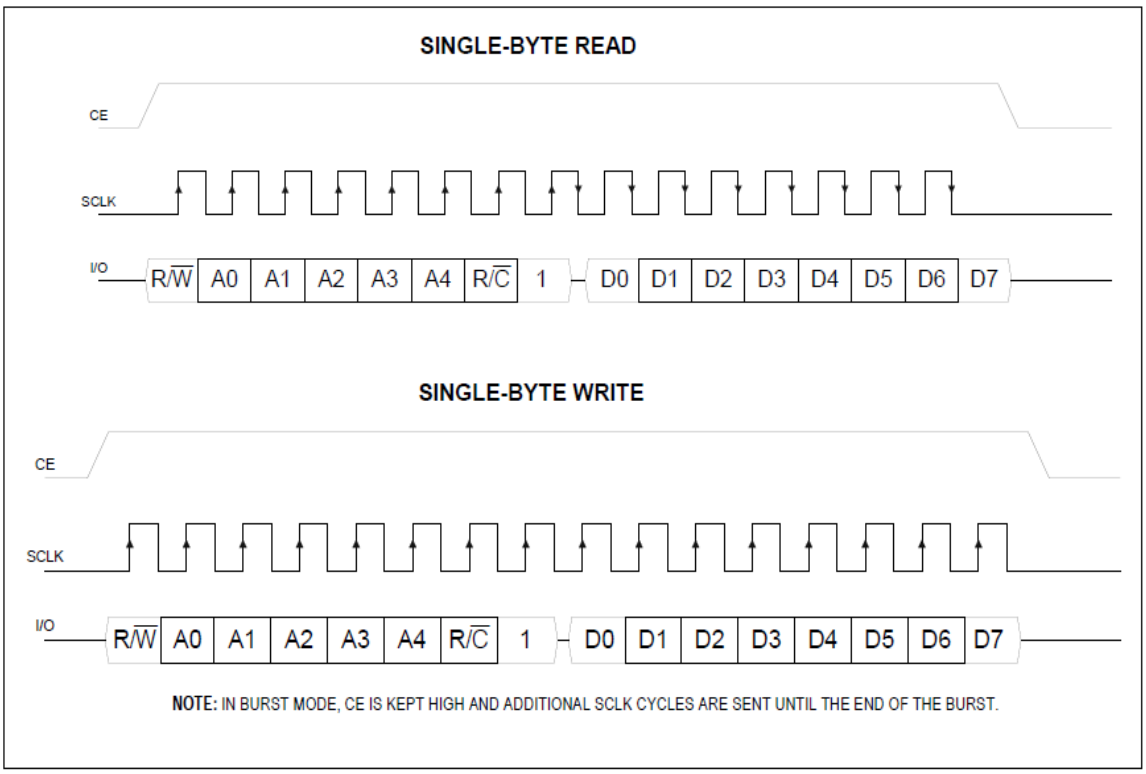
ANNEXES

Figure II.A.1. Block Diagramme de DS1302



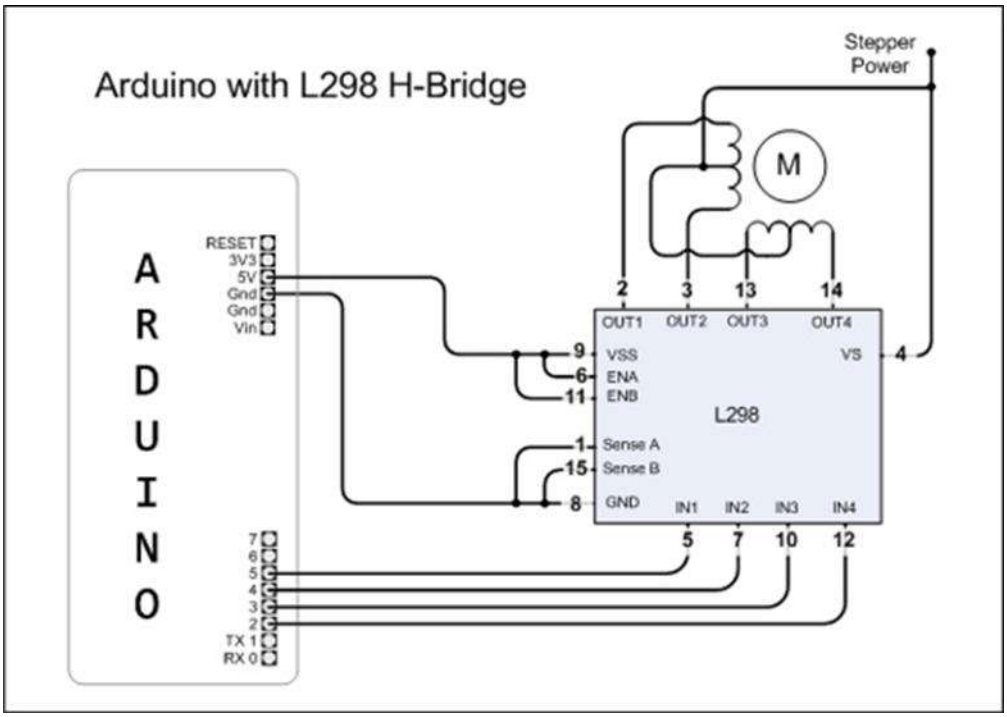
Source : MAXIM Intégrée

Figure II.A.2. Résumé de transfert de données dans DS1302



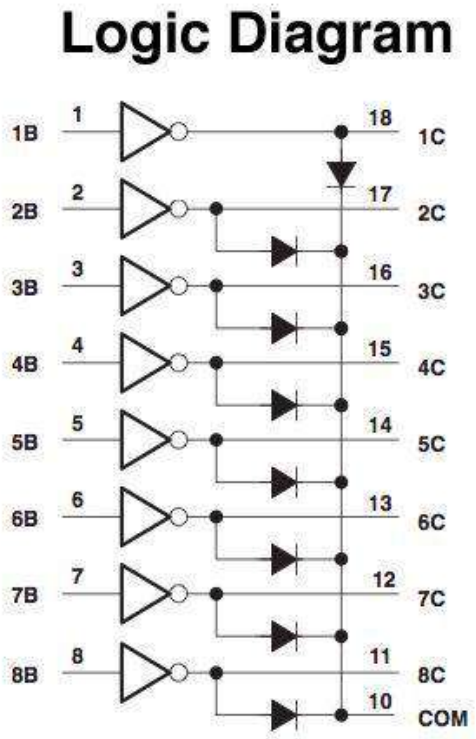
Source : MAXIM Intégrée

Figure III.A.1. Schéma de branchage de pont H avec Arduino



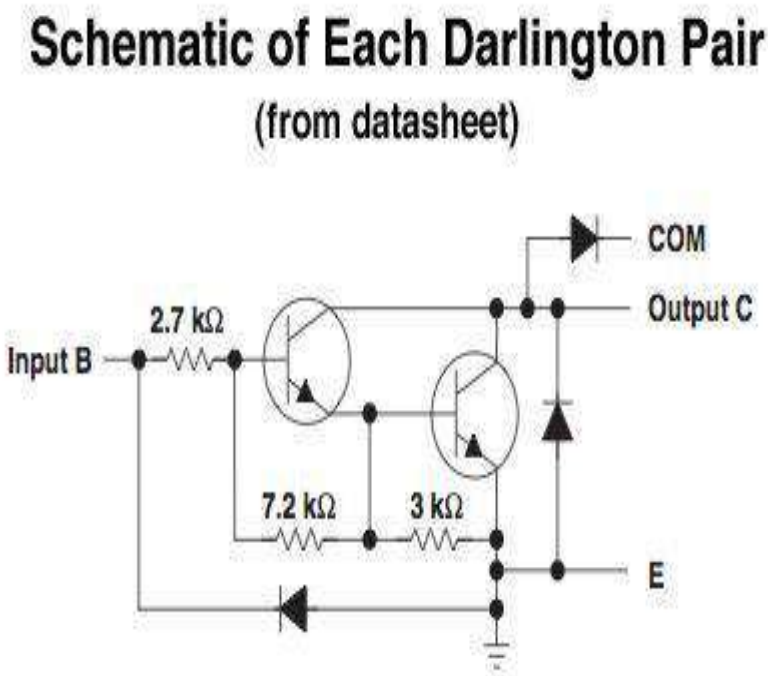
Réalisé avec logiciel ***Fritzing***

Figure IV.A.1. Schéma des entrées et sorties de ULN2803



Source : fiche technique de Darlington ULN2803
Eight Darlington array

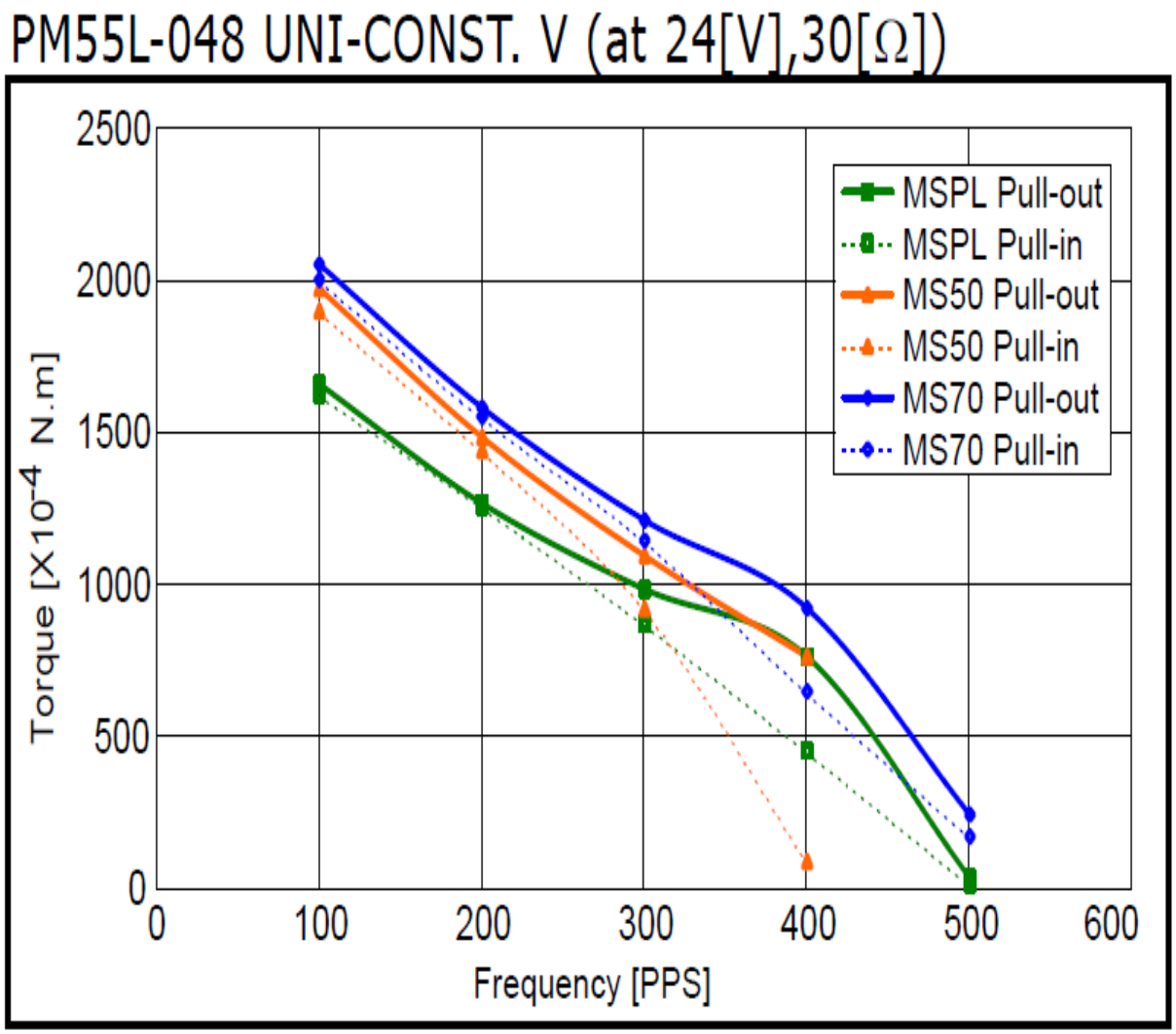
Figure IV.A.2. Schéma d'un pair entrées et sorties de l'ULN2803



Source : fiche technique de Darlington ULN2803
Eight Darlington array

Figure IV.A.3. Caractéristiques de moteur pas à pas PM55L-048

Moteur pas à pas PM55L-048	
Nombre de pas en tour	48 (7.5°/pas)
Circuit de commande	Moteur unipolaire
Voltage de commande	24V
Résistance dans chaque phase	30Ω
Température de fonctionnement	-10° jusqu'à 50°



Source : Minebea Motor Manufacturing Corporation

Figure IV.A.4. Caractéristiques de moteur pas à pas SPH-54AB

Moteur pas à pas SPH-54AB	
Nombre de pas en tour	200 (1.8°/pas)
Circuit de commande	Moteur hybride
Voltage de commande	9 V
Résistance dans chaque phase	3.5 Ω
Température de fonctionnement	-10° jusqu'à 50°

Figure IV.A.5 : Vue éclaté de support d'un suiveur solaire sur Solidworks.

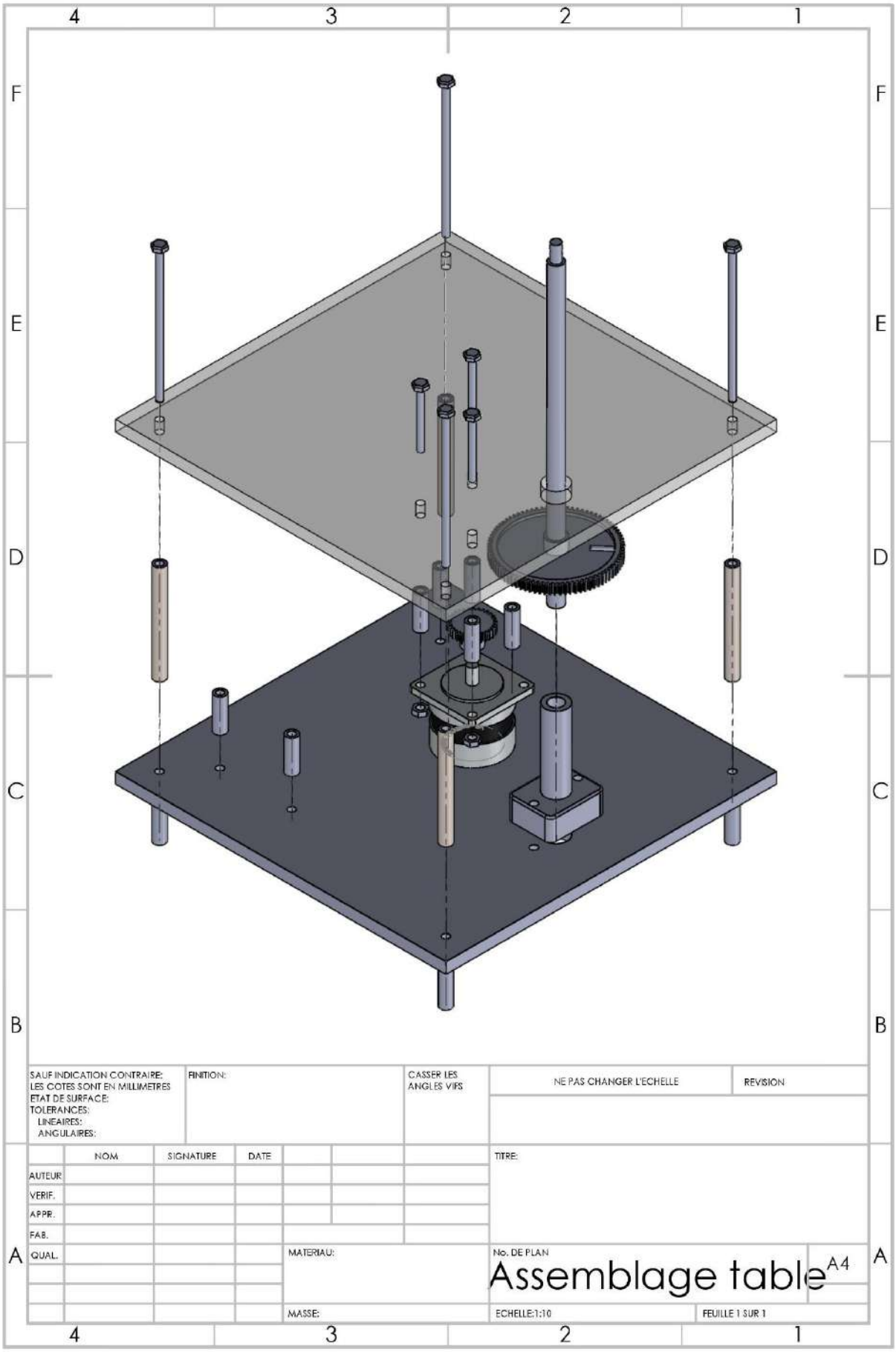


Figure IV.A.6 : Les dimensions de support d'un suiveur solaire dessiné sur Solidworks.

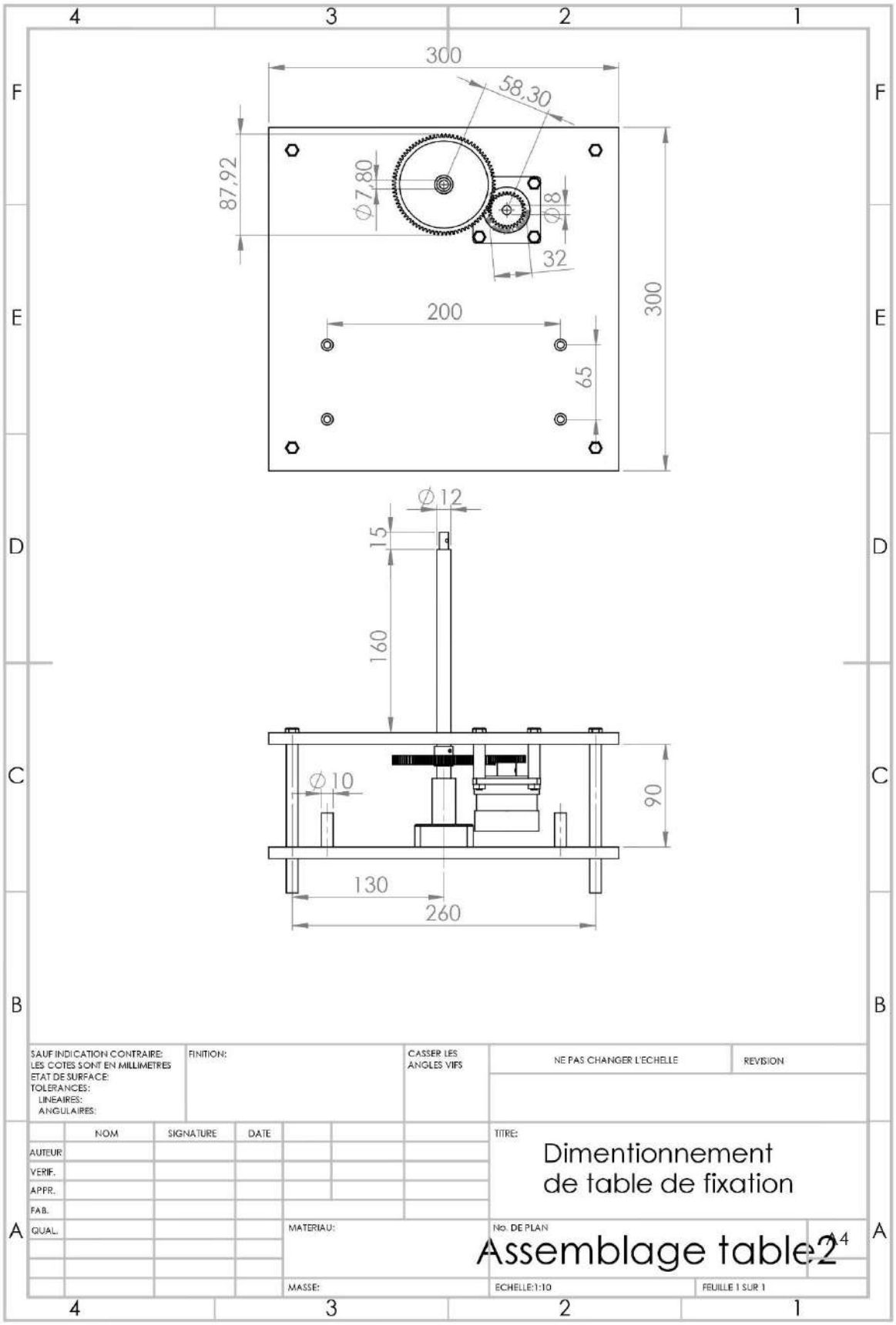


Figure IV.A.7 : Les cotations de support de rotation pour l'angle d'hauteur de soleil.

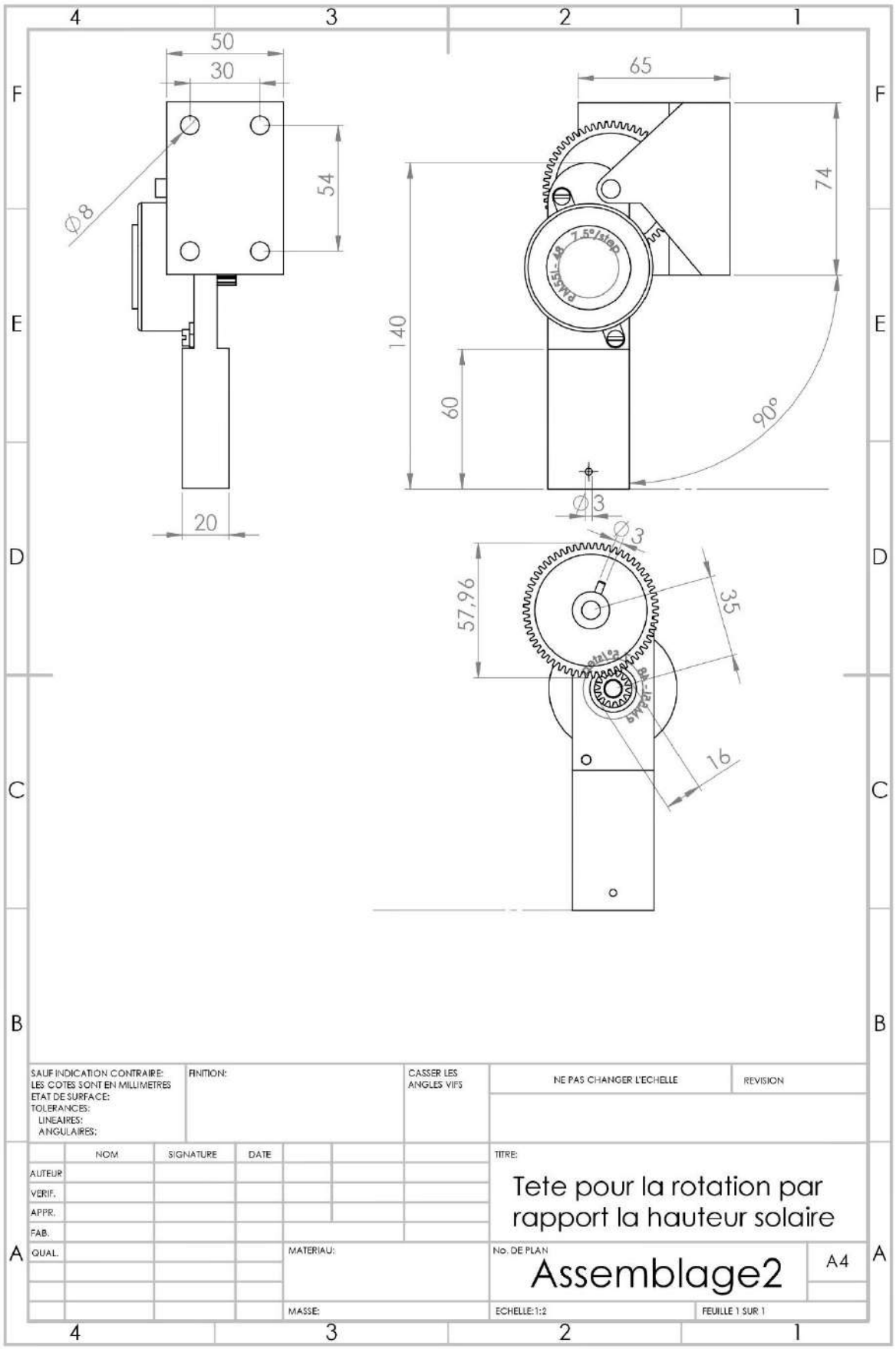
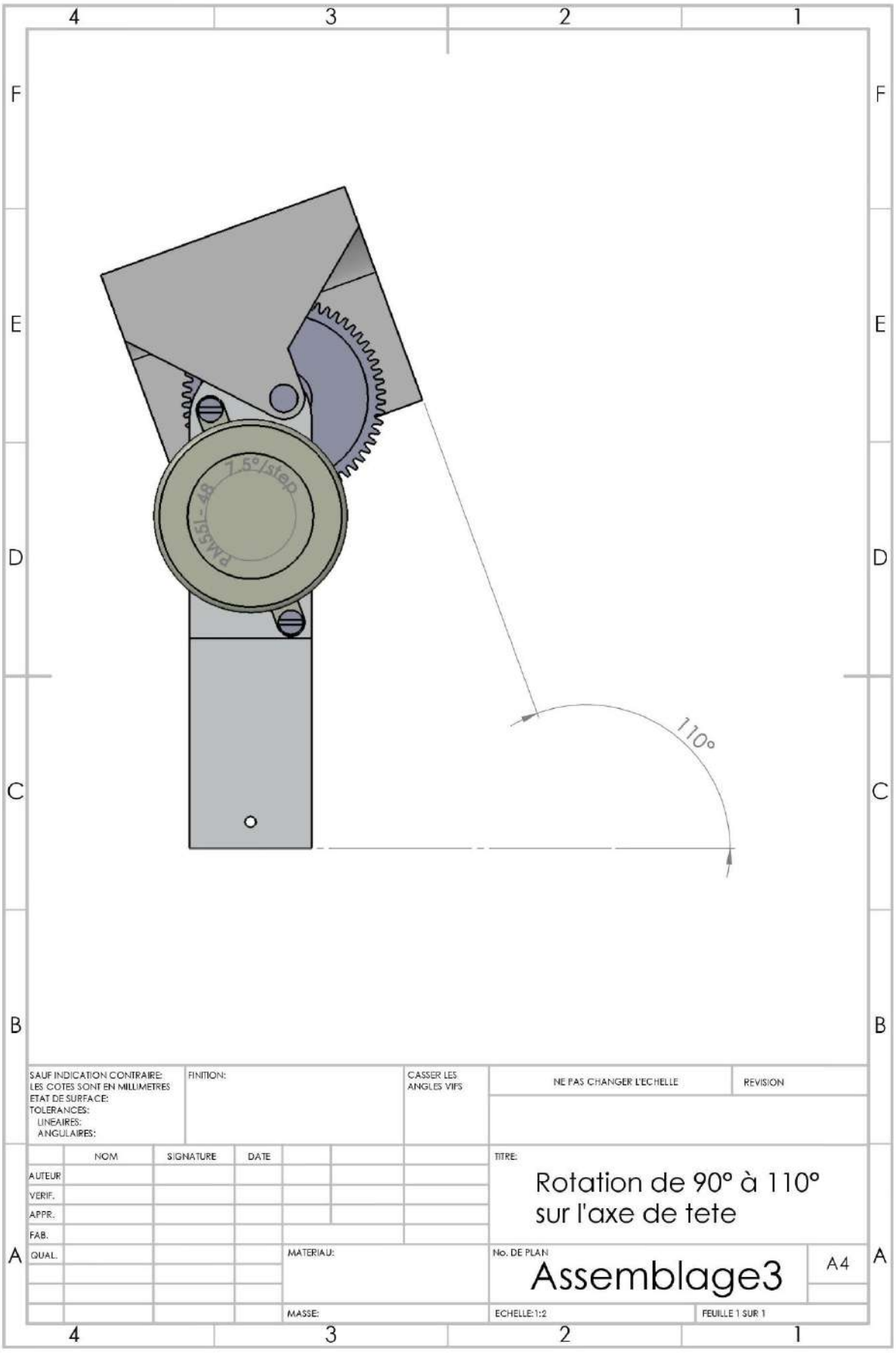


Figure IV.A.8 : L'angle max de rotation pour l'angle d'hauteur de soleil.



BIBLIOGRAPHIE

[1] : K. Kassmi1, M. Hamdaoui1 et F. Olivié, Conception et modélisation d'un système photovoltaïque adapté par une commande MPPT analogique, Revue des Energies Renouvelables Vol. 10 N°4 (2007) 451 – 462, 2007.

[2] BENKHELIFA Abdelaziz, Modélisation et expérimentation d'un capteur solaire plan à eau Influence de l'intensité de l'éclairement solaire et de l'inclinaison du capteur solaire sur le rendement du capteur, Université Kasdi Merbah Ouargla, mémoire master 2012.

[3] Salaheddine MANSOUR, Etude de Dimensionnement optimal d'une station solaire autonome en site isolé, Université de Tlemcen, mémoire master 2014.

[4] MILOUDI Leila, Etude et simulation de la poursuite de la trajectoire de soleil par un panneau solaire photovoltaïque, UNIVERSITE M'HAMED BOUGARA – BOUMERDES, mémoire master.

[5] Alain Ricard, Gisement solaire 2009.

[6] Rojer ZOGBI, Etude et réalisation d'un suiveur solaire, Université de DAKAR, rapport de stage, mémoire DEA, Energie Solaire, 1983.

[7] Lilia el Amraoui, Conception électromécanique d'une gamme d'actionneurs linéaires tubulaires à réluctance variable, Université des sciences et technologies de Lille, thèse de doctorat 2002.

[8] Luc Yin, Étude d'une commande moteur pas à pas et Réalisation de la carte de commande et programmation de la carte PCI4400, rapport de stage, Supélec France , juin 2008.

[9] PATRICE OGUICE Moteurs pas à pas et PC deuxième édition, 2012.

[10] T. Bendib, B. Barkat, F. Djeffal, N. Hamia et A. Nidhal, Commande automatique d'un système de poursuite solaire à deux axes à base d'un microcontroleur PIC16F84A par Revue des Energies Renouvelables Vol. 11 N°4, 2008.

[11] Adrian Catarius, Mario Christiner, Azimuth-Altitude Dual Axis Solar Tracker, Worcester polytechnic institute, memoire master 2010.

[12] BENZINA Yagoub et EULDJI, Etudes et la mise en œuvre de la commande d'un moteur pas à pas pour la position de panneau solaire, Université Dr YAHIA FARES de MEDEA, mémoire master.

[13] Simon Monk, Arduino, année 2013 Pearson France.

[14] Christian TAVERNIER, Arduino – maitriser sa programmation et ses cartes d'interface (shields), DONUD, 2013.

[15] Patrick CHANTEREAU, Erik BARTMANN, Le grand livre d'Arduino, serial markers 2013.

[16] OUKACI Mourad et BEZIOU Mohamed, Etude et réalisation d'une machine à commande numérique CNC à 3 axes, Université de MEDEA, mémoire master.

[17] IDDIR Hayet, Etude et réalisation d'un kit électronique du type système embarqué à base de microcontrôleur, Université M'Hamed BOGUERRA, mémoire master, Boumerdes.

[18] André PITTET, Contribution à l'étude des moteurs pas à pas monophasés, Thèse, Lausanne EPFL, 1980.

[19] BENTCHIKOU Mohamed Amine et STOUTAH Hamid Commande machine CNC a trois axes études et réalisation, Université Dr YAHIA FARES de MEDEA, mémoire master, 2013.