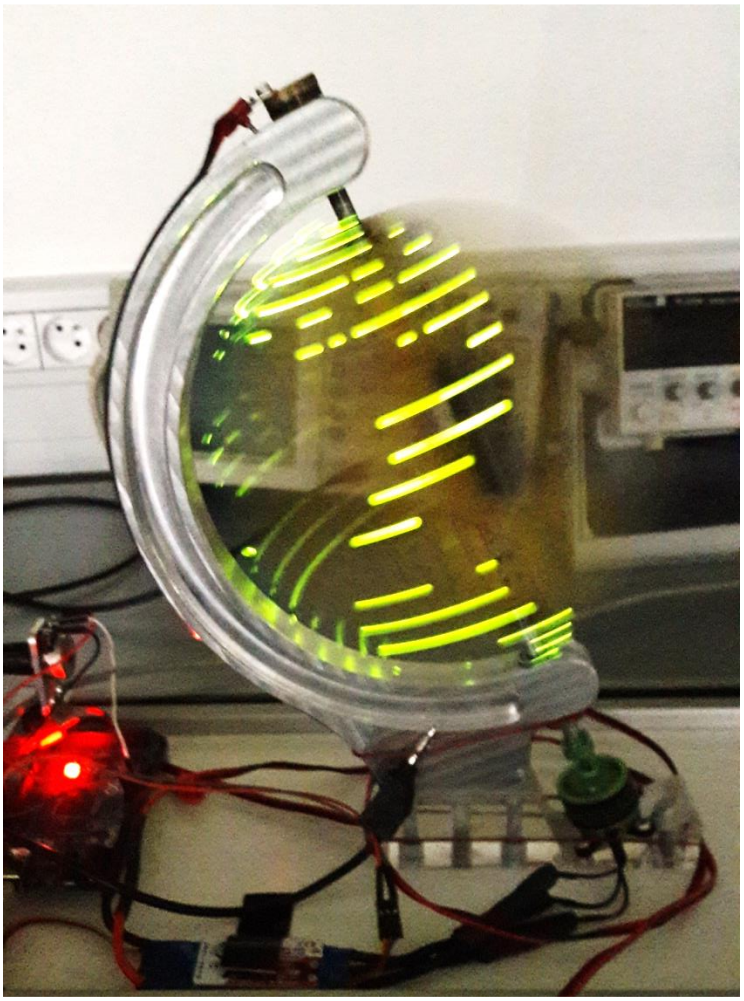
Projet tutoré - globe terrestre à LED



Jean-Baptiste SEVESTRE
IUT – GEII 2017
Tuteur : Monsieur MANGEARD

Sommaire

I – PRESENTATION DU PROJET..... 1

II – DEROULEMENT GANTT 2

II a) Déroulement prévu..... 2
II b) Déroulement réel 3

III – RESSOURCES HUMAINES ET MATERIEL..... 4

IV – PROGRAMMATION ET MISE EN MARCHE DU MOTEUR..... 5

IV a) Explication technique 5
IV b) Programmation 6

V – CARTE ELECTRONIQUE (CAO)..... 7

V a) Schématique et routage 7
V b) Explications des différentes caractéristiques 8
V c) Fonctionnement de l'ATMEGA 32 9

VI – PROGRAMMATION DU GLOBE..... 10

VI a) Logique de base..... 10
VI b) Explication programme 11

CONCLUSION..... 12

ANNEXES 13-16

I Présentation du projet

Le globe terrestre à LED est un projet tuteuré réalisé par des étudiants de première année de 2014-2015. Celui qui s'en est occupé l'année dernière était Jordan FRAUD-GELDOF. A ma connaissance ce dernier a réussi à faire fonctionner le globe terrestre mais j'ai eu toutes les difficultés à reprendre son travail pour réaliser une deuxième fois sa prouesse.

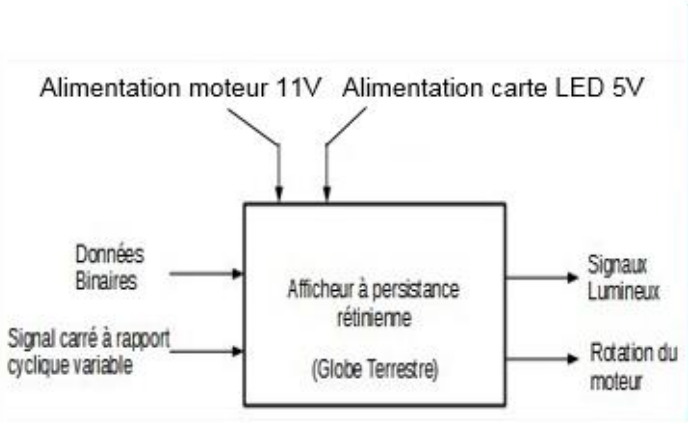
Ce projet a pour objectif d'afficher la terre au moyen d'une série de 24 LED commandés par un code en langage C et entraînés en rotation sur une carte électronique grâce à un moteur 5V codé par une carte Arduino. La rotation du moteur liée en théorie à la séquence d'allumage des LED utilise la persistance rétinienne comme moyen de visualisation par les êtres humains de la terre (continents) pour donner l'impression que l'image est complète.

Ce principe de persistance rétinienne a été découvert en 1912. C'est un phénomène physique pendant lequel des images qui se ressemblent sont attribuées à notre rétine toutes les 1/25^{ième} de seconde, ce qui donne à notre cerveau une impression d'image en mouvement ou d'image fixe. Le mieux est d'être légèrement plus rapide que la limite des 25 images par seconde pour un meilleur confort visuel. La même technique est utilisée au cinéma, à la télévision ou sur nos moniteurs d'ordinateur et fonctionne particulièrement bien. Tout est une question de fréquence de défilement.

Pour ce faire, la carte du globe où se trouvent les LED devra entièrement être refaite. Les deux schémas en bas de la page constituent le fonctionnement général et souhaité du globe à LED.

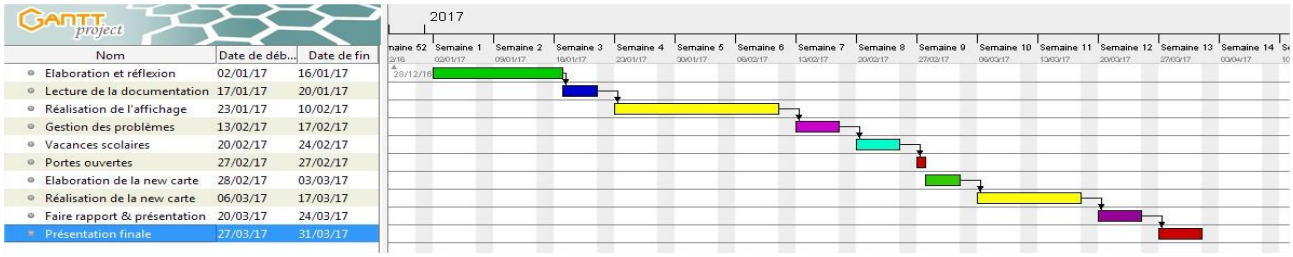
Avant de commencer ce projet j'ai défini quelque nouvelles compétences que je vais acquérir ou approfondir :

- La recherche documentaire (pour faire fonctionner le moteur Brushless, avec une documentation plutôt conséquente) ;
- Du codage Arduino et Atmel (langage de programmation en C).
- L'utilisation des tableaux (d'une taille importante) et leur adaptation à mes besoins (allumage pour la séquence des LED)
- Dessouder et ressouder des composantes CMS (test).
- La modification de l'ordre électrique comme pour placer des isolants plastiques évitant ainsi des courts-circuits entre les trois pôles du moteur.
- Perçage



II Déroulement GANTT

a) Déroulement prévu



Élaboration et réflexion

Cette partie prend en compte la recherche de stage et la réflexion personnelle du projet global. Il englobe le matériel que j'aurais besoin et l'ajustement du déroulement des séances à venir pour coller le plus possible à la réalité.

Lecture de la documentation

S'imprégner de ce qui a déjà été fait, comprendre et poser des questions au professeur référent si besoin. Comprendre les défis à relever pour la réalisation de l'affichage du globe.

Réalisation de l'affichage

Codage du globe pour se rapprocher du résultat obtenu.

Gestion des problèmes

Finalisation de l'affichage globe.

Vacances scolaires

Du 19 au 25 Février.

Porte ouverte

Le 4 mars.

Réflexion de la carte électronique

Réfléchir à la taille de la carte, forme, composants disponibles, calcul, plan de la carte, alimentation, harmonie générale...

Réalisation de la carte électronique

Semaine de soudage, intégration et installation de la carte sur son socle. Prendre en compte la résolution des différents problèmes qui pourraient subvenir.

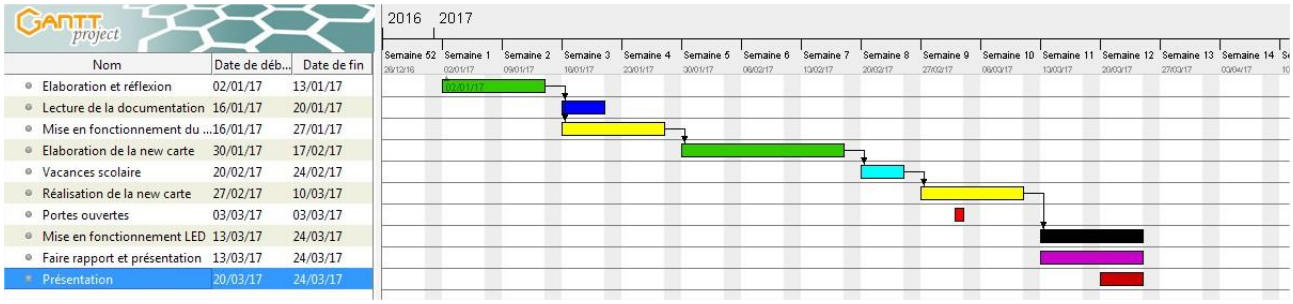
Faire rapport & présentation

Présentation avec un diaporama, préparer la présentation orale, soigner et clair. Se préparer à d'éventuelles questions.

Présentation finale

Du 27 au 31 Mars.

b) Déroulement réel



Le déroulement prévu au début du projet en décembre 2016 n'a été que partiellement respecté. En effet, on observe qu'à partir du milieu du mois de mars 2017, beaucoup de fonctions sont agglutinées sur le Gantt réel. Il était prévu que ce travail soit terminé pour le 31 mars. Or les cours se terminent le 24 mars afin de permettre aux étudiants de rechercher un logement (déménagement) pour leur stage qui débute le 4 avril et de se reposer. Ceci n'a pas empêché l'aboutissement de mon projet et le peaufinage mécanique des câbles et même du potentiomètre a ainsi pu être réalisé.

Malgré tout la vitesse du moteur laisse à désirer. En effet la vitesse est manuellement choisie par l'utilisateur, s'il n'est pas expérimenté il peut avoir du mal à synchroniser la vitesse avec la séquence afin que le globe affiche la terre sans un mouvement de rotation par la droite ou la gauche trop rapide. Il ne me restait donc plus qu'à synchroniser la vitesse du moteur par rapport à la vitesse de séquence des LED pour qu'on puisse voir le globe terrestre sans mouvement.

De plus, il était prévu que le globe à LED soit opérationnel pour la porte ouverte qui a eu lieu le 04 mars dernier. Malheureusement la carte était encore en court de réalisation plus particulièrement dans la partie test. Elle n'était donc pas installée à la place de l'ancienne sur son socle. Il m'était donc impossible de le présenter comme terminé. Néanmoins le moteur étant opérationnel, une version dinarique (rotation de la carte) a pu être présentée. Je remercie Monsieur MANGEARD pour l'avoir exposé et présenté lors de cette porte ouverte.

Dans l'ensemble, les grandes lignes ont été suivies, l'ordre des taches a été respecté. Seuls leurs délais ont été parfois modifiés.

III Ressources humaines et matériel

Objectifs :

Mon travail doit pouvoir être repris et facilement compris par une autre personne qui souhaite travailler sur le globe ou le faire fonctionner. L'objectif principal de ce projet consiste qu'à la fin de ce module, l'étudiant soit capable de faire fonctionner le globe et afficher, à l'aide des LED, les continents de la terre.

Le matériel :

Un ordinateur,

Les travaux réalisés les années précédentes sur ce sujet (intérieur et extérieur à l'IUT),

Documentation technique,

Matériels de création de carte électronique,

Une alimentation,

Le globe avec ses composants déjà en place,

Les ressources humaines :

Les professeurs,

Contact avec les anciens élève en charge de ce projet,

Budget :

Tout ce que j'ai besoin est présent à l'IUT (magasin d'électronique, salle informatique et techniciens),

Délais :

3 Mois (Mi-Janvier à fin Mars) avec 110 heures de travail effectif,

Contrainte sur le projet :

Présentation pour les portes ouvertes le 04 Mars. Pour cela il est prévu de :

- Faire fonctionner le moteur et le commander par un potentiomètre à l'aide d'un Arduino MEGA,
- Réaliser une nouvelle carte électronique dans le but de remplacer l'ancienne aux caractéristiques désuètes,
- Faire fonctionner la carte du globe avec ses 24 LED qui sera directement relié aux pattes spécifique de l'ATMega32. Le codage devra, en théorie, se faire sans problème de matériel et sera commandé avec un programme informatique pour acquérir la séquence LED désirée,
- Synchroniser la vitesse du moteur avec la séquence que les LED affichent.

IV Programmation et mise en fonctionnement du moteur

a) Explication technique

Nous possédons un servomoteur utilisé en modélisme aussi appelé « servo ». Les servomoteurs sont commandés par l'intermédiaire d'un câble électrique à trois fils qui permet d'alimenter le moteur et de lui transmettre des consignes de position sous forme d'un signal codé en largeur d'impulsion plus communément appelé PWM, soit la modulation de largeur d'impulsions**.** Cela signifie que c'est la durée des impulsions qui détermine l'angle absolu de l'axe de sortie et donc la position du bras de commande du servomoteur. Le signal est répété périodiquement, en général toutes les 20 millisecondes, ce qui permet à l'électronique de contrôler et de corriger continuellement la position angulaire de l'axe de sortie, cette dernière étant mesurée par le potentiomètre.

Lorsque le moteur tourne, l'axe du servomoteur change de position, ce qui modifie la résistance du potentiomètre. Le rôle de l'électronique est de commander le moteur pour que la position de l'axe de sortie soit conforme à la consigne reçue, c'est à dire un asservissement.

Dans un premier temps, la lecture de la documentation technique a été un facteur clé du début de mon projet. Je me suis ensuite penché sur la réalisation physique de la maquette. J'ai introduit un ARDUINO MEGA pour avoir une liaison physique entre le moteur BRUSHLESS et l'ordinateur d'où le programme provient. Internet m'a été d'une grande aide durant tout ce projet mais plus particulièrement pour cette partie. J'y ai appris comment fonctionnait ce moteur, son codage avec un potentiomètre pour régler à notre guise la vitesse du moteur.

DM 2210 / Kv 1100

Soyez le premier à commenter ce produit

2210 : Kv 1100 / 43 g / 110 W
HT : 24,42 €
TTC : 29,31 €
Réf. : 72210

Ajouter au panier Qté : 0

Ajouter à la liste d'envies Ajouter au

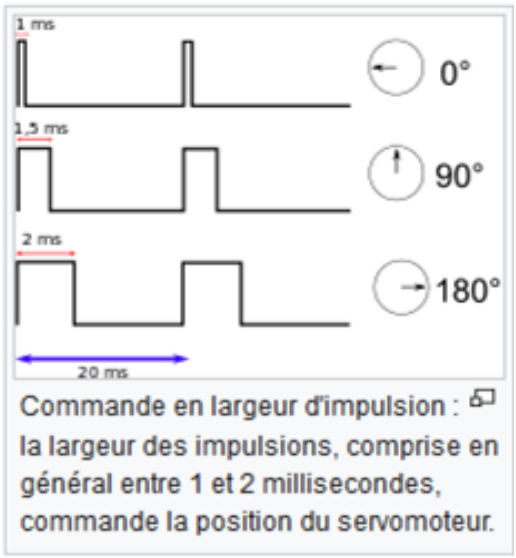
Envoyer à un ami

Description du produit | Informations complémentaires | Commentaires

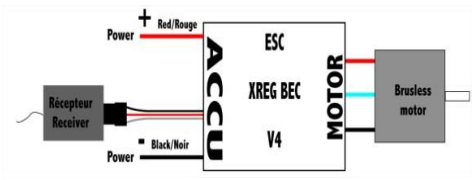
2210 : Kv 1100 / 43 g / 110 W

PROTRONIK 2210-1100 (72210)

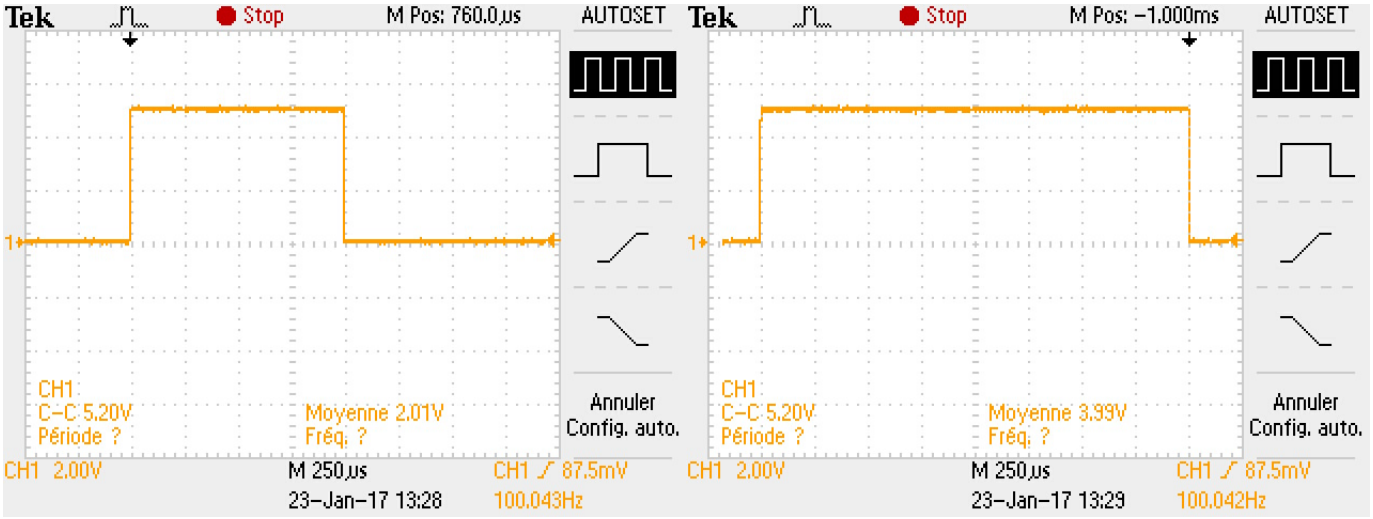
Diamètre 27,7 mm	Longueur 27,4 mm	Poids 42,4 g	Arbre 3 mm + Sauve hélice APC et GWS / Fixation : 4 vis M3		14 pôles	APPLICATIONS		
Courant Io 0,463 A	Résistance 0,171 Ω	Kv 1100 Tr/V	P nominale 110 W	Courant Max 14 A	Rendement 82,4 %	14 spires	ParkFlyers/modèles jusqu'à 1000 g	
HELICE	GWS 8 x 4,3			APC 9 x 4,7			APC 10 x 4,7	
TENSION	Courant	Puissance	Régime	Courant	Puissance	Régime	Courant	Puissance	Régime
6 V	3,3 A	19,8 W	5515 Tr/Min	4,9 A	29,4 W	5070 Tr/Min	6,5 A	39,0 W	4575 Tr/Min
7 V	4,1 A	28,7 W	6284 Tr/Min	6,0 A	42,0 W	5705 Tr/Min	8,4 A	58,8 W	5127 Tr/Min
8 V	5,0 A	40,0 W	7079 Tr/Min	7,4 A	59,2 W	6346 Tr/Min	10,4 A	83,2 W	5601 Tr/Min
9 V	6,0 A	54,0 W	7844 Tr/Min	8,8 A	79,2 W	7011 Tr/Min	12,6 A	113,4 W	6078 Tr/Min
10 V	6,3 A	63,0 W	8210 Tr/Min	10,2 A	102,0 W	7600 Tr/Min			
11 V	7,1 A	78,1 W	8831 Tr/Min						
12 V	7,8 A	93,6 W	9318 Tr/Min						



Commande en largeur d'impulsion : la largeur des impulsions, comprise en général entre 1 et 2 millisecondes, commande la position du servomoteur.



Mes mesures sur les impulsions envoyées au moteur sont les suivantes :



La première photo représente une impulsion courte, la vitesse du moteur ne sera pas très importante. Il faut savoir que si l'impulsion est inférieure à 1 milliseconde le moteur ne démarrera pas. Plus l'impulsion sera longue et plus le moteur tournera vite. Ici la photo de droite représente l'impulsion où la vitesse du moteur est au maximum (2 millisecondes). Nous avons ici un moteur deux pôles donc en théorie le moteur peut tourner à 3500 tr/sec.

b) Programmation

PROG_ARDUINO

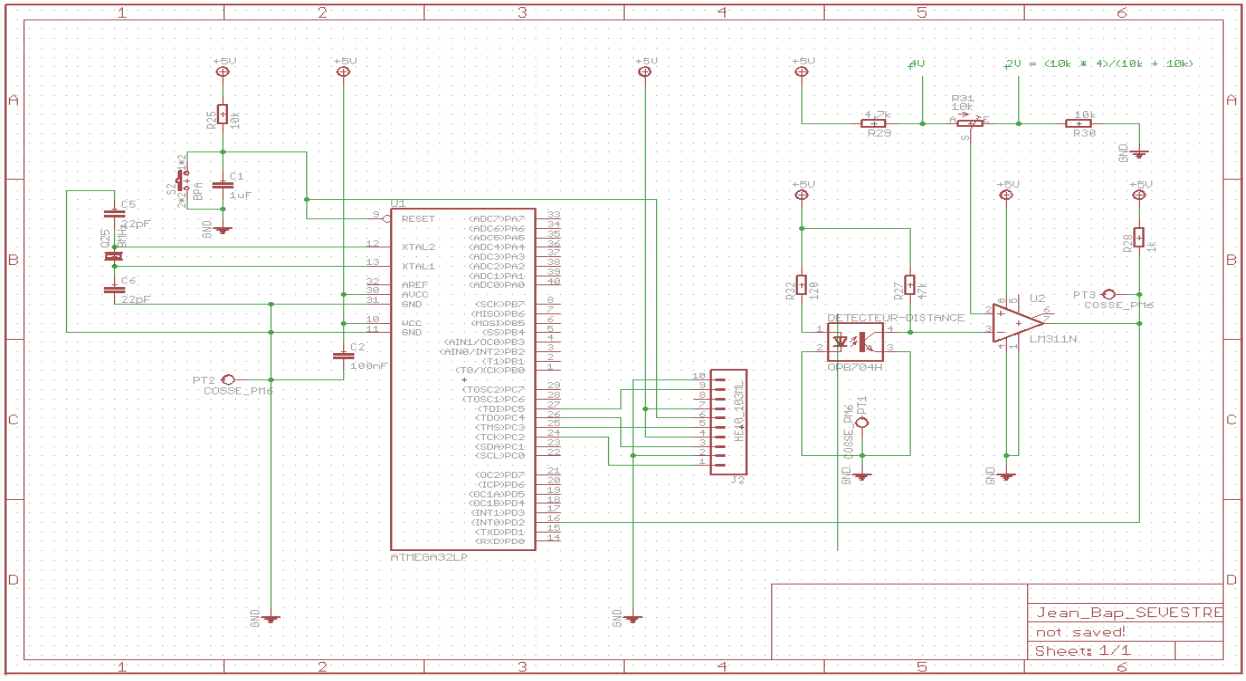
```
/*
Programme gestion du moteur Brushless DM2210
Jean-Baptiste SEVESTRE
GEII - G231
*/

#include <Servo.h>

Servo Moteur;                    // Créer un servo-objet pour contrôler un servo

int potpin = 0;                  // Analogique utilisée pour connecter le potentiomètre.
int val;                         // Variable pour lire la valeur de la broche analogique

void setup() {
  Moteur.attach(11);             // Fixe le servo sur la broche 11 à l'objet d'asservissement
}

void loop() {
  val = analogRead(potpin);          // Lit la valeur du potentiomètre (valeur comprise entre 0 et 1023)
  val = map(val, 0, 1023, 1000, 2000);// L'échelle pour l'utiliser avec le servo (valeur entre 0 et 180)
  Moteur.writeMicroseconds(val);     // Définit la position de servo en fonction de la valeur à l'échelle
  delay(15);                         // Attend le servo pour y arriver
}
```

Contrairement à ce que je m'attendais, le rapport laissé par mon prédécesseur ne permettait pas de faire fonctionner le moteur. J'ai dû par conséquent tout reprendre du début. Nouvelle programmation, nouvelle mesure pour faire tourner le moteur à différentes vitesses à l'aide d'un potentiomètre, celui-ci branché intelligemment sur l'ARDUINO MEGA.

V Carte électronique (CAO)

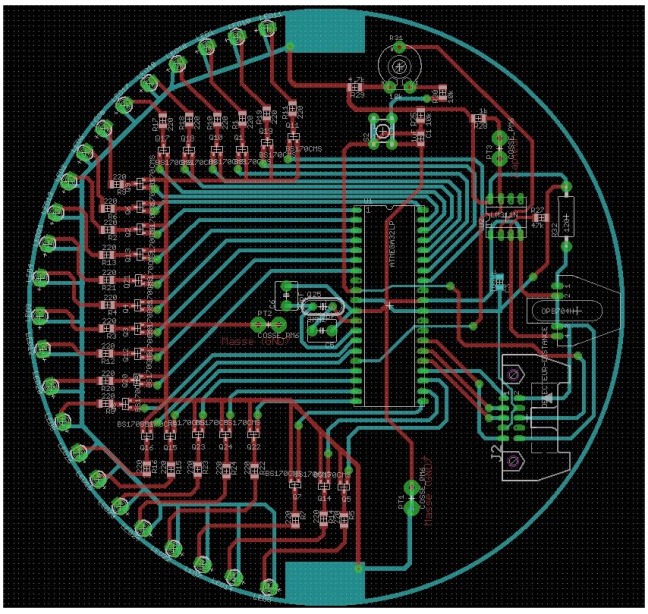
a) Schématique et routage

Dans un premier temps, la conception sur papier puis aussitôt la retranscription sur le schématique EAGLE a été un gain de temps et d'énergie. La version finale est donnée ci-dessous avec les LED et sans les LED pour une meilleure appréhension.



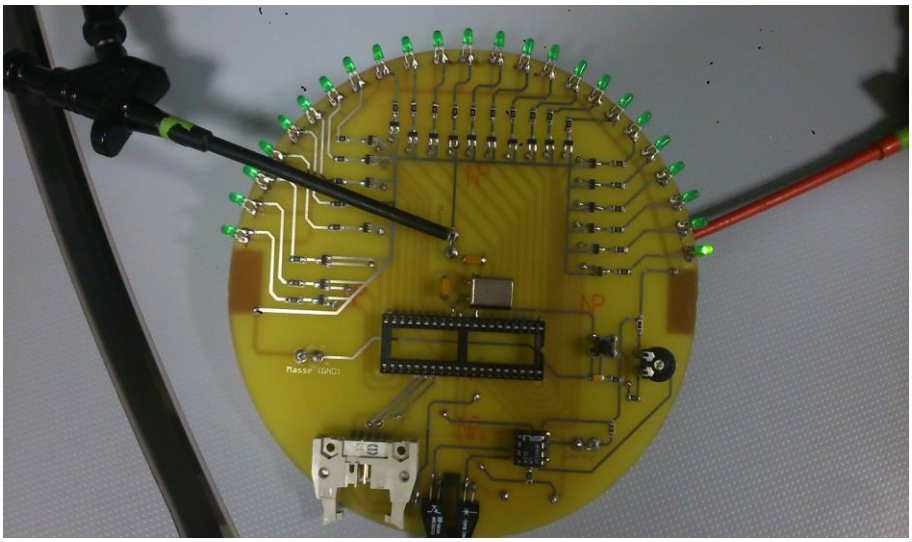
La schématique a été fastidieuse mais intéressante et je n'ai pas compté mes heures pour y parvenir à bien. La cosse de test pour le détecteur de distance qui devait être une LED (test visuel) a finalement été modifiée en test physique avec l'utilisation d'un voltmètre pour la mesure des variations entre « un obstacle » et « pas d'obstacle ». Le détecteur de présence a été introduit pour avoir la possibilité de détecter lorsqu'un tour est fait par le globe. Ainsi on peut connaitre sa vitesse de rotation. En théorie il faut que le globe fasse 1500tr/min soit un tour en 40 millisecondes.

La synchronisation entre le schéma et le routage est très importante. Mon erreur a été de faire le routage indépendamment du schématique. Ce qui a eu pour conséquence qu'au moment où j'ai voulu insérer les 24 LED au routage, la synchronisation ne se faisant plus, il m'était impossible de les relier à l'ATMEGA32. J'ai donc dû la recommencer entièrement. De plus une très courte nappe a été oubliée lors du routage. J'ai donc très simplement résolu le problème en reliant des deux extrémités de la piste nom imprimés par un strap.



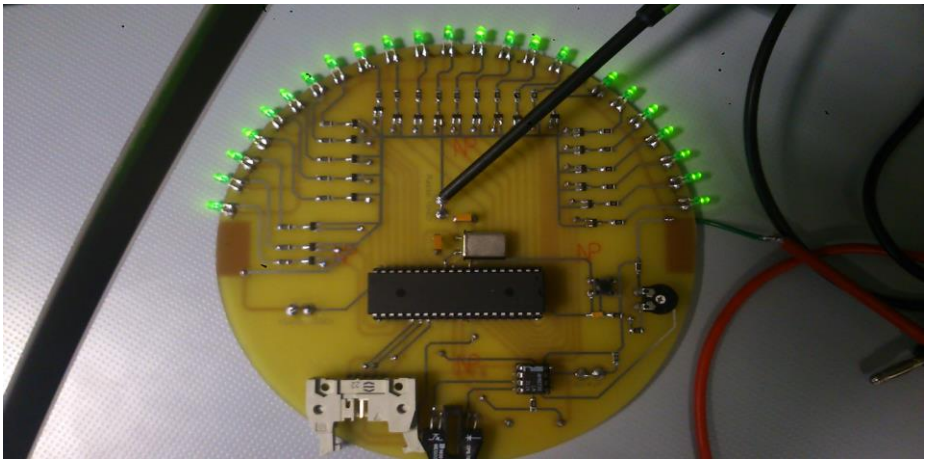
b) Explications des différentes caractéristiques

- La phase du soudage et de test



Une fois que tous les composants ont été soudés, vérifiés avec un ohmmètre et un voltmètre, il a fallu vérifier que toutes les LED éclairaient et fonctionnaient normalement avec les caractéristiques mécaniques et électriques prévus. Dans un premier temps le sens des LED était inversé j'ai donc dû les dessouder pour corriger l'erreur. Ensuite certains transistors étaient défectueux, j'ai donc dû les dessouder puis en remettre d'autres qui étaient en bon état de fonctionnement. Ces transistors sont de type CMS. Cette étape était l'une des plus difficiles à réaliser. Une piste a aussi malencontreusement été oubliée durant le routage, j'ai donc dû procéder à mettre une nappe pour le résoudre. Dernière chose au niveau du connecteur HE10-10, une des piste était beaucoup trop fine et était insensible au test de continuité ; j'ai donc dû installer une nappe pour relier les deux extrémités.

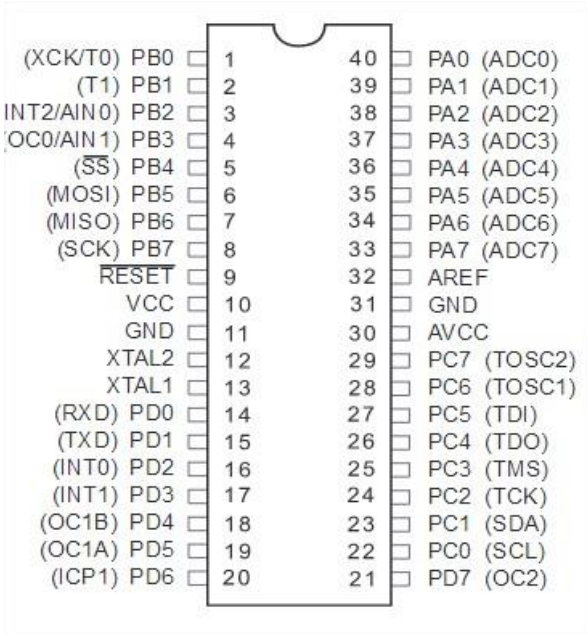
La phase de test avec code simple :



Le résultat avec le code, bien que fastidieux parfois, pour savoir si le problème venait du code ou de la carte, a finalement été obtenu avec succès. Le logiciel Atmel-Sudio dysfonctionne, il se ferme souvent sans prévenir. Cela est dû à plusieurs problèmes à mon sens :

- un espace dans le nom du fichier ou le chemin.
- Le débogage peut aussi le faire planter, il faut alors utiliser le triangle vert clair qui démarre le programme sans déboguer.

c) Fonctionnement de l'ATMEGA32

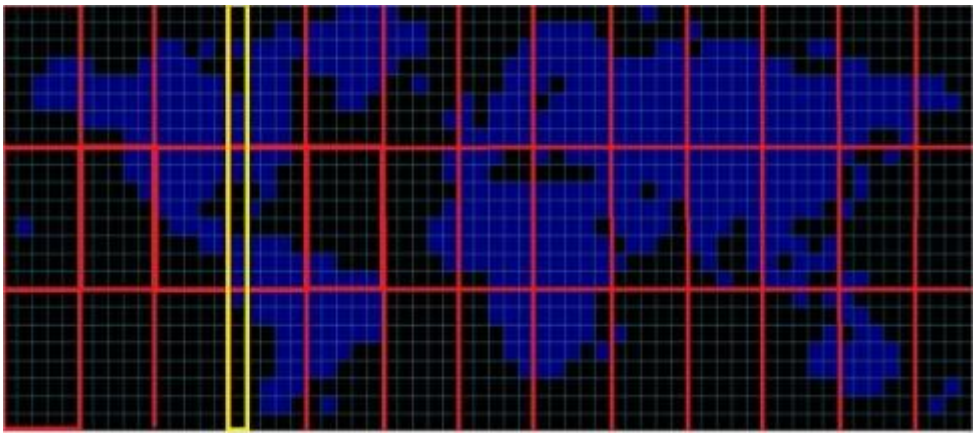


L'ATMEGA32 est un microcontrôleur CMOS 8 bits de faible puissance basé sur l'architecture RISC amélioré AVR. AVR est le terme utilisé par Atmel pour désigner le cœur du processeur et la famille de microcontrôleurs qui le mettent œuvre. Il est composé de 32 registres et un jeu de 90 à 135 instructions en fonction des composants. Le cœur AVR a été optimisé pour exécuter du code produit par un compilateur C. En exécutant des instructions puissantes dans un cycle d'horloge unique, l'ATmega32 atteint des débits de 1 MIPS par MHz, permettant au concepteur du système d'optimiser la consommation électrique en fonction de la vitesse de traitement. Chaque LED y est reliée, cela permet de les contrôler directement sans l'aide de relais. ~~C~~eux-ci auraient été incontournables s'il y avait eu davantage de LED ou de couleurs que de pattes au microcontrôleur. Ce sont des LED simples unicolores qui ont été installés, de couleurs vertes pour une question d'esthétique. De ce fait, il y a suffisamment de pattes de l'ATMEGA32 libres pour toutes les brancher. Si on avait eu des LED bicolores il nous aurait fallu deux fois plus de ports (24x2) donc nous aurions eu besoin de relais comme sur l'ancienne carte électronique de Jordan FRAUD-GELDOF.

VI Programmation du globe

a) Logique de base

Pour programmer le globe, il est nécessaire de définir ce qui doit être affiché par les LED et avec la bonne séquence. L'image de référence est la suivante, elle représente l'hémisphère terrestre :



Grâce au 24 LED disposées verticalement sur un coté de la carte à espace régulier nous pouvons afficher les 64 colonnes de ce tableau les unes après les autres.

Pour représenter cela, il est nécessaire de convertir chaque colonne de 8 lignes en une donnée de 8 bits, ce qui nous donne 3 valeurs pour une colonne entière.

Un carreau noir est codé par un 0.

Un carreau bleu est codé par un 1.

La colonne 16 en jaune prend donc 3 valeurs soit $\begin{cases} 00100111 \\ 11000111 \\ 10000000 \end{cases}$

A présent il est aisé de comprendre comment créer la bonne séquence pour que les LED affichent correctement le globe terrestre. Les 64 colonnes du motif avec 3 tableaux pour les trois parties d'hémisphère représentées sur l'image.

```
unsigned char
earth0[64]={0b00000000,0b00000000,0b00001100,0b00011100,0b00011100,0b00011110,0b00001110,0b00001110,0b00001111,0b00011111,0b00111111,0b00111111,0b00011111,0b00101111,0b00001111,0b00100111,0b00111111,0b00111111,0b01111111,0b00100000,0b01110000,0b01110000,0b11111100,0b11111100,0b11111000,0b11110000,0b01100000,0b01101000,0b00000000,0b00000010,0b00000111,0b00000001,0b00000111,0b00011111,0b00111111,0b01111011,0b01110111,0b01111111,0b01111111,0b00111111,0b00011111,0b00011111,0b10011111,0b00111111,0b00111111,0b01011111,0b00111111,0b00111111,0b00111111,0b00111111,0b00011111,0b00011111,0b00001111,0b00101111,0b00101111,0b00011111,0b00011111,0b00001110,0b00001110,0b00001111,0b00001100,0b00001100,0b00000100,0b00000000};

unsigned char
earth1[64]={0b00000000,0b00001000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b11000000,0b11100000,0b11110000,0b11111000,0b11111000,0b11101100,0b11001100,0b11000111,0b11010111,0b11100111,0b10000111,0b00000011,0b00000011,0b00000011,0b00000001,0b00000001,0b00000001,0b00000000,0b00000000,0b00000000,0b00001100,0b11011110,0b11111110,0b11111110,0b10111111,0b00111111,0b00111111,0b10111111,0b00111111,0b10111111,0b10111111,0b11110111,0b11111010,0b11111100,0b11011100,0b11111000,0b11100000,0b11111100,0b11111110,0b11111001,0b11110000,0b11111110,0b11111111,0b11111101,0b11110111,0b11110010,0b11100110,0b10010001,0b01100001,0b00000000,0b10000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000};

unsigned char
earth2[64]={0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b10000000,0b11000000,0b11000110,0b11111110,0b11111100,0b11111000,0b11111010,0b11110000,0b11100000,0b11100000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b11000000,0b11110000,0b11111000,0b11111000,0b11111000,0b11110000,0b11100000,0b00010000,0b00100000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b10000000,0b00011000,0b10111000,0b11011100,0b01111100,0b00111100,0b00011000,0b00000000,0b00000000,0b00000010,0b00000100,0b00000000};
```

Ce programme demeure incomplet il faut ajouter par exemple un quadrillage toutes les 5 colonnes et 8 lignes afin d'éviter au maximum les erreurs de lecture et donc de conversion. De plus ce code ne prend pas en compte les PORTS, il faut donc adapter entièrement le code aux quatre PORTS de la carte ATMEGA32 ingénieusement.

b) Explication du programme

Contrairement à ce que l'on pourrait penser, le programme déjà existant m'a presque été complètement inutile. Seuls les tableaux ont pu être réutilisés. Cela est dû au fait que la carte de Jordan FRAUD-GELDOF utilisait des LED bicolores avec des relais ce qui n'est pas mon cas. De plus, il utilisait des boucles for qui me paraissaient obsolètes. J'ai donc recréé un programme de A à Z pour que les LED affichent la bonne séquence.

```
int main(void)
{
	initports();                                                    //Initialisation des PORTS
	while(1)
	{
		for (unsigned char j=0; j<=63;j++)  //Parcourir les 64 colonnes des tableaux
		{
			PORTA=nord[j];                          //PORTA 1 à 7
			PORTB=equa[j];                          //PORTB 1 à 7
			PORTC=sudC[j];                          //PORTC 22 23 &

		PORTD=sudD[j];                          //PORTD 21 20 19 18 17 15 (soit 8 PORTS pour 8
LEDS avec le PORTC)


			_delay_us(625);          //La vitesse du moteur est de 1500tr/min
                                                //Soit un tour fait 40 ms
                                                //Donc delay= 40ms/64 colonnes = 625µs
		}
	}
	return (0);
}
```

Le PORT A va chercher ses instructions dans le tableau « nord » et fait toutes les valeurs de la variable « j » qu'il trouve jusqu'à la fin. Une fois terminée, la boucle recommence.

La même chose est réalisée pour le PORT B. La petite modification physique de l'arrière de la carte avec une nappe ajoutée a été réalisée dans cet objectif. De plus j'ai dû inverser toute la séquence du code car les PORTS de la carte ont été réalisés à l'envers, mais ceci n'a pas été une grande difficulté pour résoudre ce problème.

Pour les PORT C et D le travail a été grandement différent. Je n'utilise que deux sorties du PORT D, j'ai donc dû faire un tableau avec deux bits dans chaque séquence du tableau.

Pour le PORT C, j'avais donc 8 – 2 sorties utilisées. Encore une fois l'inversion des 6 bits a dû être effectuée.

Conclusion

Le projet a été relativement bien terminé. La nouvelle carte a été installée. Les LED s'allument et le moteur tourne correctement. Seul bémol, la vitesse de celui-ci n'est pas adaptée à la séquence des LED. Par conséquent on ne voit pas l'hémisphère lorsque la carte électronique tourne sur son axe. Durant ce projet tuteuré, j'ai eu le souci des choses bien faites et belles. Je trouve ma carte particulièrement réussie, les masses des différents composants la font légèrement basculer d'un côté. Ceci a été résolu avec l'ajout de vis du coté trop léger.

J'ai eu un réel plaisir à présenter mon projet aux classes de première année et j'espère leur avoir donné l'envie de prendre l'option électronique l'an prochain. J'ai pu la semaine avant mon départ en stage de fin d'études, montré à un petit groupe d'élèves, intéressés par mon projet, le globe terrestre fonctionnel. J'ai pu voir dans leurs yeux un réel intérêt, un vrai plaisir et une certaine admiration pour mon travail.

Si c'était à refaire, au sujet du moteur je demanderais de l'aide à mes camarades probablement un peu plus tôt. Autre fait marquant, j'avais oublié que même en codage Arduino il y a besoin de Librairie. Au sujet de la carte électronique je placerais une cosse au niveau du +5 Volts car j'ai dû, ingénieusement, mettre un fil pour pouvoir mettre un grippe-fil pour alimenter la carte. Ce fil ajouté au niveau du +5 Volts est uniquement utilisé durant les tests. Il est tout sauf esthétique et n'apparait pas sur la carte sur son socle final. Au sujet de la masse j'en avais installé deux ce qui m'a énormément simplifié la tâche.

Pour la réalisation de la carte électronique, je n'ai pas été perdu. Cela a même été un plaisir de la réaliser sur tous ses aspects. Le travail ressemblait beaucoup au travail d'Etude et Réalisation de l'année dernière. Malgré tout, nous nous trouvions à un niveau au-dessus du socle de compétences avec une forme de carte différente et un travail plus conséquent pour le schéma, mais aussi pour le routage avec le placement judicieux des LED. Pour ce qui a été du codage des LED, j'ai sollicité une fois l'aide du groupe de GEII option informatique pour une boucle « for » que j'avais mal écrite dans le programme, ce qui me posait d'innombrable problème car celui-ci ne fonctionnait pas.

Je remercie Monsieur MANGEARD pour le suivi de mon projet. C'est grâce à ses nombreux conseils et réflexion, mais aussi son refus de m'aider, que j'ai réalisé que j'étais capable de tout faire moi-même. J'aurais aimé pouvoir en faire d'avantage comme programmer une nouvelle séquence de LED pour afficher autre chose sur le globe, mais le projet touche à sa fin et je laisse donc le soin à mon successeur de réaliser ce rêve.

ANNEXE

Programme de la séquence des LEDS :

```
#define F_CPU 8000000UL                         //Quartz à 8 Méga-Hertz
#include <avr/io.h>                             //Bibliothèque
#include <util/delay.h>                         //Bibliothèque
#include <stdio.h>                              //Bibliothèque

#define bitSet(Port,Bit)    Port|=(1<<Bit)      //Met a 1 le bit "Bit" du port "Port"
#define bitClear(Port,Bit)  Port&=~(1<<Bit)     //Met a 0 le bit "Bit" du port "Port"


#define ST_CP_low PORTB &=~(1<<PB5)
#define ST_CP_high PORTB|=(1<<PB5)
#define SH_CP_low PORTB &=~(1<<PB7)
#define SH_CP_high PORTB|=(1<<PB7)

unsigned char
nord[64]={0b00000000,0b00000000,0b00110000,0b00111000,0b00111000,0b01111000,0b01110000,0b011
10000,
      0b11110000,0b11110000,0b11111100,0b11111100,0b11111000,0b11110100,0b11110000,0b111001
00,
      0b11111100,0b11111100,0b11111110,0b00000100,0b00001110,0b00001110,0b00111111,0b001111
11,
      0b00011111,0b00001111,0b00000110,0b00010110,0b00000000,0b01000000,0b11100000,0b100000
00,
      0b11100000,0b11111000,0b11111100,0b11011110,0b11101110,0b11111110,0b11111110,0b111111
00,
      0b11111000,0b11111000,0b11111001,0b11111100,0b11111100,0b11111010,0b11111100,0b111111
00,
      0b11111100,0b11111100,0b11111000,0b11111000,0b11110000,0b11110100,0b11110100,0b111100
00,
      0b11111000,0b01110000,0b01110000,0b11110000,0b00110000,0b00110000,0b00100000,0b000000
00}; //tableau de LEDS coté NORD

unsigned char
equa[64]={0b00000000,0b00001000,0b00000000,0b00000000,0b00000000,0b00000000,0b00000000,0b000
00000,
      0b11000000,0b11100000,0b11110000,0b11111000,0b11111000,0b11101100,0b11001100,0b110001
11,
      0b11010111,0b11100111,0b10000111,0b00000011,0b00000011,0b00000011,0b00000001,0b000000
01,
      0b00000001,0b00000000,0b00000000,0b00000000,0b00001100,0b11011110,0b11111110,0b111111
10,
      0b10111111,0b00111111,0b00111111,0b10111111,0b00111111,0b10111111,0b10111111,0b111101
11,
      0b11111010,0b11111100,0b11011100,0b11111000,0b11100000,0b11111100,0b11111110,0b111110
01,
      0b11110000,0b11111110,0b11111111,0b11111101,0b11110111,0b11110010,0b11100110,0b100100
01,
      0b01100001,0b00000000,0b10000000,0b00000000,0b00000000,0b00000000,0b00000000,0b000000
00};



unsigned char
sudC[64]={0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b
00,0b01,0b01,0b00,0b00,0b01,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00
,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,
```

```
0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b00,0b01,0b00,0b00};
unsigned char
sudD[64]={0b000000,0b000000,0b000000,0b000000,0b000000,0b000000,0b000000,0b000000,
       0b000000,0b000000,0b000000,0b000000,0b000000,0b000000,0b000000,0b000001,
       0b000011,0b100011,0b111111,0b111111,0b011111,0b011111,0b001111,0b000111,
       0b000111,0b000000,0b000000,0b000000,0b000000,0b000000,0b000000,0b000000,
       0b000011,0b001111,0b011111,0b011111,0b011111,0b001111,0b000111,0b001000,
       0b001000,0b000000,0b000000,0b000000,0b000000,0b000000,0b000000,0b000000,
       0b000000,0b000000,0b000000,0b000000,0b000000,0b000000,0b011101,0b111011,
       0b111110,0b111100,0b011000,0b000000,0b000000,0b000000,0b100000,0b000000};
void initports(void)
{
              DDRA =0b11111111;                 // PA0 à PA7  -> Sorties en série
              DDRB =0b11111111;                 // PB0 à PB6  -> Sorties en série
              DDRC =0b00000011;                 // PC0 à PC1  -> Sorties en série
              DDRD =0b11111010;                 // PD1 et PD3 & PD4 à PD7-> Sorties en série

              PORTA |=0xFF; //Forcer certain PORT à 1 (Ceux ou il y a des broches LED)
              _delay_us(1000);
              PORTB |=0xFF; //Forcer certain PORT à 1 (Ceux ou il y a des broches LED)
              _delay_us(1000);
              PORTC |=0x03; //Forcer certain PORT à 1 (Ceux ou il y a des broches LED)
              _delay_us(1000);
              PORTD |=0xFA; //Forcer certain PORT à 1 (Ceux ou il y a des broches LED)
              _delay_us(1000);

       PORTA =0x00;                       //Forcer le PORT à 0 (Initialisation des PORTS)
       PORTB =0x00;                       //Forcer le PORT à 0
       PORTC =0x00;                       //Forcer le PORT à 0
       PORTD =0x00;                       //Forcer le PORT à 0
}

void pulseST_CP(void)
{
       ST_CP_high;
       _delay_us(1);
       ST_CP_low;
}

int main(void)
{
       initports();                                          //Initialisation des PORTS
       while(1)
       {

              for (unsigned char j=0; j<=63;j++)  //Parcourir les 64 colonnes des tableaux
              {
                     PORTA=nord[j];                          //PORTA 1 à 7
                     PORTB=equa[j];                          //PORTB 1 à 7
                     PORTC=sudC[j];                          //PORTC 22 23 &
                     PORTD=sudD[j];                          //PORTD 21 20 19 18 17 15
(soit 8 PORTS pour 8 LEDS avec le PORTC)

                     _delay_us(625);             //La vitesse du moteur est de 1500tr/min
                                                 //Soit un tour fait 40 ms
                                                 //Donc delay= 40ms/64 colonnes = 625µs
              }
       }
       return (0);
}
```

Programme pour le bon fonctionnement du moteur avec potentiomètre :

PROG_ARDUINO §

```
/*
Programme gestion du moteur Brushless DM2210
Jean-Baptiste SEVESTRE
GEII - G231
*/

// Fils du potentiomètre sur l'Arduino UNO:
// Noir  -> GND,
// Rouge -> 3.3V,
// Blanc -> A0,

//Fils sortants du moteur Brocheless :
//Jaune  -> Broche 11,
//Marron -> GND,

//Le progamme ne fonctionne pas si l'Arduino UNO n'est pas alimenté,
//Le moteur doit être alimenté en 11V.

#include <Servo.h>

Servo Moteur;                    // Créer un servo-objet pour contrôler un servo

int potpin = 0;                  // Analogique utilisée pour connecter le potentiomètre.
int val;                         // Variable pour lire la valeur de la broche analogique

void setup() {
  Moteur.attach(11);             // Fixe le servo sur la broche 11 à l'objet d'asservissement
}

void loop() {
  val = analogRead(potpin);          // Lit la valeur du potentiomètre (valeur comprise entre 0 et 1023)
  val = map(val, 0, 1023, 1000, 2000);// L'échelle pour l'utiliser avec le servo (valeur entre 0 et 180)
  Moteur.writeMicroseconds(val);     // Définit la position de servo en fonction de la valeur à l'échelle
  delay(15);                         // Attend le servo pour y arriver
}
```

Programme du test pour les LED de la carte électronique :

```
#ifndef F_CPU
#define F_CPU 8000000UL
#endif

#include <avr/io.h>
#include <util/delay.h>
#include <avr/iom32.h>
#include <avr/sfr_defs.h>

int main(void)
{
    DDRA = 0xFF; //Nakes PORTC as Output
    DDRB = 0xFF;
    DDRC = 0xFF;
    DDRD = 0xFF;
    while(1) //infinite loop
    {
        PORTA = 0xFF; //Turns ON All LEDs
        PORTB = 0xFF; //Turns ON All LEDs
        PORTC = 0xFF; //Turns ON All LEDs
        PORTD = 0xFF; //Turns ON All LEDs
    }
}
```

Carte Eagle complète de la carte électronique :