République Algérienne Démocratique et Populaire
Ministère de l'enseignement Supérieur et de la Recherche scientifique
Université Mouloud Mammeri Tizi-Ouzou
Faculté des Génie Electrique et Informatique
Département de Electronique
Option: Electronique industriel

Mémoire de Fin d'Etudes
En vue de l'obtention du diplôme:

MASTER PROFESSIONNEL

Thème
Réalisation d'un système de mesure de la vitesse d'un train miniature commandé par l'infrarouge

Présenté par:

BELAL Ziane

ABASSI Redouane

Encadreur:

Mr. LAZRI

Année universitaire : 2015 / 2016

Remerciement

Nous tenons à remercier notre dieu, le tout puissant de nous avoir donné la santé et la volonté pour compléter ce modeste travail.

Nous tenons à exprimer nous profonds gratitudes à nos vifs remerciements à notre promoteur pour son sérieux, sa patience et son aide et tous les enseignants qui nous a aider.

Enfin nous remercions tous ceux qui ont participé de près ou de loin à la réalisation de ce mémoire

Sommaire:

Index des tableaux

Chapitre 2 :

Index des figures

Chapitre 1:

Chapitre 2 :

Chapitre 3 :

Introduction

La connaissance de la vitesse d'un engin est un paramètre important dans la mécanique, l'industrie.... En particulier, en cinématique, la vitesse est une grandeur qui mesure pour un mouvement, le rapport de la distance parcourue au temps écoulé.

Pour mesurer la vitesse, plusieurs techniques sont développées. L'une les plus répondues est le déplacement d'un objet d'un point vers un autre point dont la distance est connue préalablement Il s'agit de déterminer le temps écoulé [6]. D'autres utilisent des capteurs de lacet pour véhicules automobiles baser sur un phénomène physique appelé piézo-électriques[4]; et d'autres basée sur un faisceau infrarouge[2]. C'est cette méthode que nous allons utiliser.

Dans notre projet, nous allons utiliser un système de rotation basé sur un émetteur (photodiode) et un récepteur (phototransistor). Le système permet de mesurer la vitesse d'un train miniature lequel fonctionne avec un moteur à courant continu. Pour faire une application complète, notre projet contiendra deux partie ; une partie commande qui consiste à contrôler la puissance et le sens de rotation, la deuxième partie est la partie mesure qui permet d'estimer la vitesse instantanée.

Pour permettre l'échange d'information. la télécommande émet des rayons infrarouges, qui sont des faisceaux de lumière invisible pour nos yeux. L'infrarouge est fabriqué par une diode, petit composant électronique qui transforme un signal électrique (d'où la nécessité de la pile) en une lumière ayant un spectre de longueur d'onde invisible à l'œil nu et se situant en dessous du rouge, dit infrarouge

En effet, grâce a un capteur de rotation qu'on va positionner en parallèle avec la roue de l'axe doté d'un trou au but de mesurer le temps d'une seul rotation. Ce temps sera tenu compte à l'aide de programme arduino qu'on a élaboré pour qu'on puisse mesurer et afficher la vitesse de notre application.

Notre mémoire est structuré en trois chapitres et une conclusion

Le chapitre 1 est consacré à l'explication détaillée de la carte arduino UNO qu'on va utiliser.

Le chapitre 2 illustre tous les composants que nous allons employer pour notre application ainsi que leur fonctionnement.

Le chapitre 3 montre le fonctionnement général de notre application.

Enfin, nous terminerons notre travail par une conclusion en mentionnant des perspectives.

Chapitre 1 :

Description de la carte Arduino

Préambule:

Dans ce chapitre, nous allons exposer la carte Arduino d'une façon générale et puis nous nous intéressons à la carte UNO que nous avons utilisée pour notre application. En effet, dans un premier temps, un aperçu général sur la carte est présenté, dans lequel nous avons décrit les lignes d' entrées/sorties numériques et analogiques, ainsi que leurs alimentations. Ensuite, nous avons montré les détails de l'architecture interne de cette carte et son microcontrôleur et ses mémoires.

1.1.Définition de la carte arduino:

Arduino est une plate-forme de prototypage d'objet interactifs a usage créatif constituée d'une carte électronique et d'un environnent de programmation, elle est composé de plusieurs composants électroniques et d'un microcontrôleur associée â des entrées et sorties qui permettent â l'utilisateur de brancher différents types d'éléments externes, permettant de recevoir, d'analyser et de produire des signaux électriques. Sa simplicité et son prix réduit le rend accessible à n'importe qui désireux de se lancer dans l'électronique ou voulant développer des composants électriques avancés. cet environnement matériel et logiciel permet â l'utilisateur de formuler ses projets par l'expérimentation directe. Il existe plusieurs variétés de carte Arduino. La figure(1) montre la carte qu'on a utilisé, une des cartes Arduino connu sous le nom«Uno»,[1].

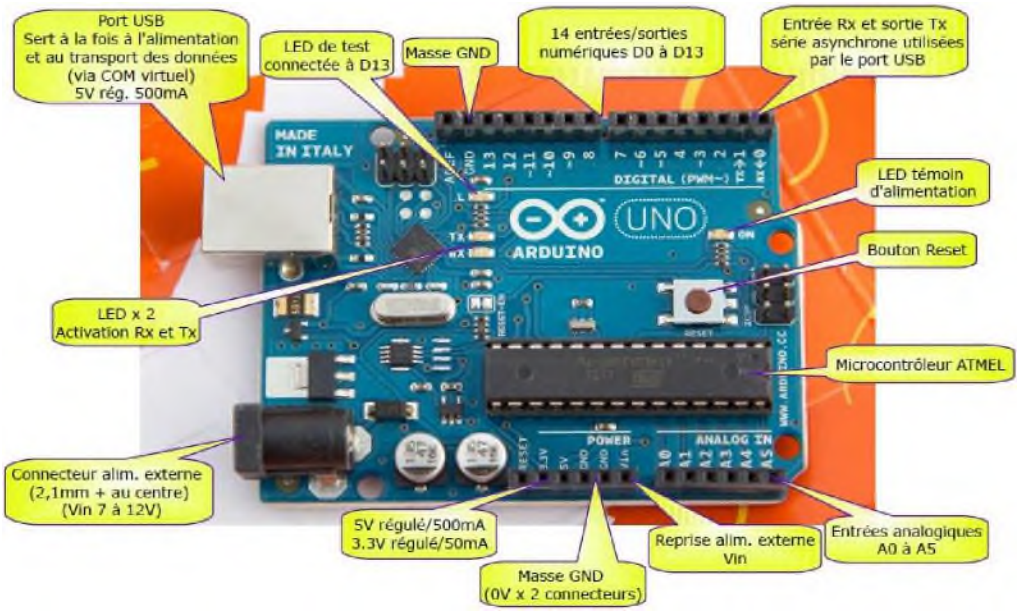


Figure (1) : forme d'une carte arduino UNO.

1.2.Historique d’arduino:

Le projet arduino est né en hiver 2005.Massino Banzi enseigne dans une école de Design à Ivrea en Italie, et souvent ses étudiants se plaignent de ne pas avoir accès a des solutions bas prix pour accomplir leur projets de robotique. Banzi en discute avec David Cuartielles, un ingénieur Espagnol spécialisé sur microcontrôleur.

Ils décident de créer leur propre carte en embarquant dans leur histoire un des étudiants de Banzi, David Mllis qui sera chargé de créer le langage de programmation allant avec la carte. Ils décidèrent de l’appeler Arduino.

Il devient un hit tout de suite auprès des étudiants. Tout le monde arrive à en faire quelque chose très rapidement sans même avoir de connaissances particulière ni en électronique ni en informatique : réponse à des capteurs, faire clignoter des leds, contrôler des moteurs, commander d’un robot, affichage des signaux, manipuler des systèmes ....Ils publient les schémas.

Les 50 premières cartes sont directement attribuées aux élèves de l’école. En 2006, on a enregistré la vente de 5000 cartes, puis plus de 30 000 en 2007, et en 2011 plus de 120 000, sans compter les clones. Ce qui lui a permet d’avoir une envergure mondiale vu ces performances et les avantages qu’elle offre aux usagers.

2. Protection:

La carte arduino Uno intègre un poly-fusible réinitialisable qui protège le port USB de votre ordinateur contre les surcharges en intensité (le port USB est généralement limite a 500 mA en intensité). Bien que la plupart des ordinateurs aient leur propre protection interne, le fusible de la carte fournit une couche supplémentaire de protection. Si plus de 500 mA sont appliquées au port USB, le fusible de la carte coupera automatiquement la connexion jusqu'à ce que le court-circuit ou la surcharge soit stoppé.

3. Alimentation:

Lorsque la carte est connectée a un ordinateur via **USB**, c'est le port USB de l'ordinateur qui fournit l'énergie (5V), et lorsque en branche une source d'énergie au connecteur de la carte (batterie, transformateur ou pile), le système peut fonctionner de manière autonome.

Ce circuit inclut un régulateur de tension a 5V mais il doit être alimenté entre 7 et 12 V pour garder une marge en basse tension et éviter que le circuit ne chauffe trop (car le régulateur de tension disperse toute surtension en chaleur).

4. Les entrées /sorties:

On va expliquer toute entrée et sortie de l'arduino [3] en commençant avec,

4.1. Les entrées /sorties numérique:

C'est par ces connections que le microcontrôleur est relie au monde extérieur, une carte Arduino standard est dotée de 14 entées sorties numériques dont 6 peuvent assurer une sortie PWM.

Chacune des 14 broches numérique de la carte UNO (numérotées des 0 a 13) peut être utilisée soit comme une entrée numérique, soit comme une sortie numérique, en utilisant les instructions pinMode (), digitalWrite (), et digitalRead () du language Arduino . Ces broches fonctionnent en 5V. Chaque broche peut fournir ou recevoir un maximum de 40mA d'intensité et dispose d'une résistance de rappel « (pull-up) (déconnectée par défaut) de 20-50 KOhms. Cette résistance interne s'active sur une broche en entrée a l'aide de l'instruction digitalWrite **(**broche, HIGH).

Il y a entre ces broches celles qui ont des fonctionnalités en plus:

4.1.1. Communication série:

Broches 0 (R X) et 1 (TX) .Utilisées pour recevoir (RX) et transmettre (TX) les données séries de niveau TTL. Ces broches sont connectées aux broches correspondantes du circuit intégré ATmega 8U2 programmé en convertisseur USB-vers-série de la carte (composant qui assure l'interface entre les niveaux TTL et le port USB de l'ordinateur). On fait appel a la transmission série a travers ces broches avec l'instruction Seril.print(), sinon va y avoir un chevauchement.

4.1.2. Interruptions externes:

Broches 2 et 3. Ces broches peuvent être configurées pour déclencher une interruption sur une valeur basse, sur un front montant ou descendant, ou sur un changement de valeur.

4.1.3. Interruption PWM (largeur d'impulsion modulée):

Broches 3, 5, 6, 9, 10, et 11. Fournissent une impulsion PWM 8-bits a l'aide de l'instruction analogWrite()

4.1.4. SPI (Interface Série Périphérique):

Broche 10 (SS) ,11(MOSI),12(MISO), 13(SCK).Ces broches supportent la communication SPI disponible avec la librairie . Les broches SPI sont également connecter à la broche 13.

Il y a une LED incluse dans la carte connectée a la broche 13. Lorsque la broche est au nivaux HAUT, la LED est allumée, lorsque la broche est au nivaux BAS, la LED est éteinte.

6.2. Les entrées analogiques:

Les entrées analogiques lui permettent de mesurer une tension variable (entre 0 et 5V) qui peut provenir de capteurs ou d'interfaces diverses (potentiomètres, etc.).

La carte UNO dispose de 6 entrées analogiques (numérotées de 0 à 5), chacune pouvant fournir une mesure d'une résolution de 10 bits (c.à.d. sur 1024 niveaux soit de 0 à 1023) à l'aide de la fonction analogRead () du langage Arduino. Par défaut, ces broche mesurent entre le 0V (valeur 0) et le 5V (valeur 1023), mais il est possible de modifier la référence supérieure de la plage de mesure en utilisant la broche AREF et l'instruction analogReference () du langage Arduino.

NB : les broches analogiques peuvent être utilisées en tant que broches numériques : elles sont numérotées en tant que broches numérique de 14 à 19, aux cas où le nombre de broches numériques n'est suffisant.

6.2.1. Autres broches:

Il y a deux autres broches disponibles sur la carte:

1.AREF :

Tension de référence pour les entrées analogiques (si différent du 5V), utilisée avec l'instruction analogReference (). Elle s'utilise pour réduire.

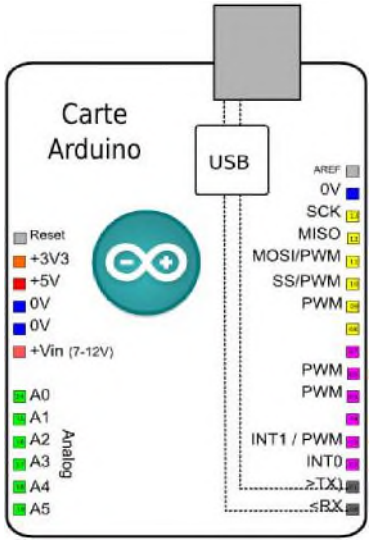
2.Reset:

Mettre cette broche au niveau BAS entraîne la réinitialisation (le redémarrage) du microcontrôleur. Typiquement, cette broche est utilisée pour ajouter un bouton de réinitialisation sur le circuit qui bloque celui présent sur la carte.

7. La communication avec l'extérieur:

La carte arduino Uno dispose de toute une série de facilités pour communiquer avec un ordinateur, une autre carte Arduino, ou avec d'autres microcontrôleurs.

L'ATmega 328 dispose d'une UART (Universal Asynchronous Receiver transmitter ou émetteur universel asynchrone en français) pour communication série de niveau TTL (5V) et qui est disponible sur les broche 0 (RX) et 1 (TX) comme le montre la figure suivante:



Figure(2) : communication série avec RX et TX

Un circuit intégré ATmega8U2 sur la carte assure la connexion entre cette communication série vers le port USB de l'ordinateur et apparait comme un port COM virtuel pour les logiciels de l'ordinateur. Le code utilisé pour programmer l'ATmega8U2 utilise le driver standard USB COM, et aucun autre externe n'est nécessaire.

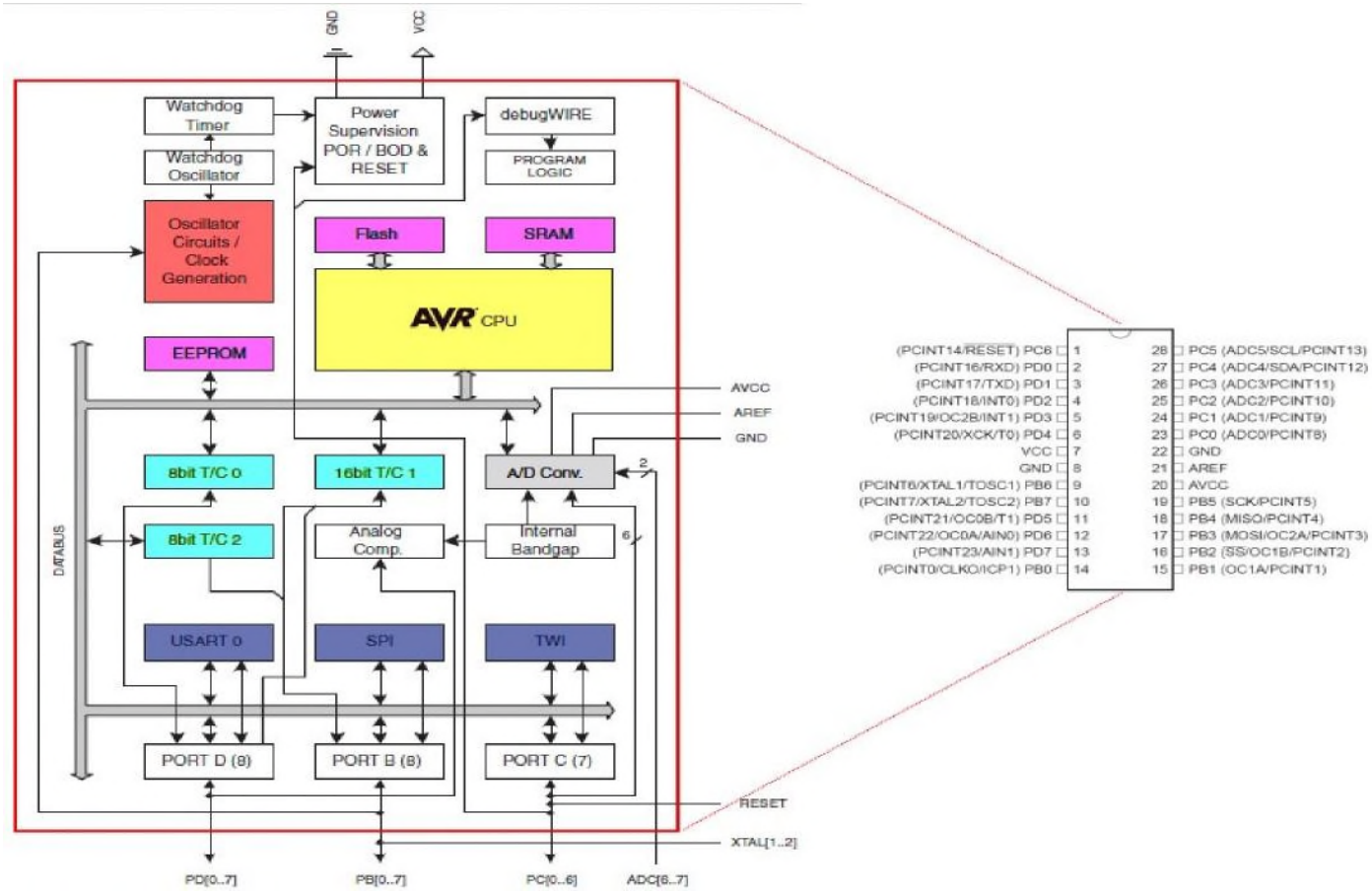
Le logiciel Arduino inclut une fenêtre terminal série (ou moniteur série) sur l'ordinateur et qui permet d'envoyer des textes simple depuis et vers la carte Arduino. Les LEDs RX et TX sur la carte clignote lorsque les données sont transmises via le circuit intégré USB-vers-série et la connexion USB vers l'ordinateur sur n'importe quelle broche numérique de la carte UNO.

L'ATmega 328 supporte également la communication par protocole I2C et SPI:

- Le logiciel Arduino inclut la librairie Wire qui simplifie l'utilisation du bus I2C.
- Pour utiliser la communication SPI (Interface Série Périphérique), la librairie pour communication SPI est disponible, il suffit de la faire inclure dans le programme au niveau du logiciel Arduino lors de la programmation.

8. Le microcontrôleur ATmega328:

Le microcontrôleur utilisé sur la carte Arduino UNO est un microcontrôleur ATmega328 (voir figure 3). C'est un microcontrôleur ATMEL de la famille AVR 8bits.



Figure(3) : architecteur interne et schéma de brochage de ATmega 328

6.1. Les principales caractéristiques de L'ATmega328:

Le microcontrôleur ATmega328 de ATMEL dispose:

- De 14 broches numériques d'entrées/sorties, dont 6 peuvent être utilisées en sorties PWM (largeur d'impulsion modulée), réparties selon l'ordre suivant : OC0A(PD6), OC0B5PD5), 0C1A(PB1), OC1B(PB3), OC2A(PB3), OC2B(PD3) et 2 (0 et 1) pour réception/émission série.
- De 6 entrées analogiques qui peuvent également être utilisées en broches entrées/sortie numériques). Ces entrées/sorties sont réparties sur trois ports : PortB, portC, portD (soit 23 broches E/S en tout).
- D'un courant max par broches E/S =40mA.
- D'un courant max sur sortie 3,3V=50mA.
- D'une mémoire Flash de 32 KB dont 512Bits utilisée par le bootloader.
- D'une mémoire SRAM de 2KB.

 D’une mémoire EEPROM de 1KB.

IL contient aussi trois compteurs(Timer0, Timer1, Timer2), le Timer0 et le Timer 2 sont comptage 8 bits, le Timer1 il est à comptage 16 bits. Chaque Timer peut être utilisé pour générer deux signaux PWM. Certaines broches peuvent avoir plusieurs fonctions différentes choisies par programmation:

PWM=pour l’utilisation de la PWM, le ATmega a 6 broches qui peuvent servir à cette fonction qui sont les broches OC0A(PD6), OC0B(PD5), OC1A(PB1), OC1B(PB3), OC2A(PB3), OC2B(PD3).

6.2.Convertisseur analogique/numérique:

le ATmega328 possède un convertisseur Analogique/Numérique d’une résolution de 10 bits, ce convertisseur peut être utilisé à travers 6 entrées multiplexées de ADC0(PCD) jusqu’à ADC5(PC5).

6.3.Gestion bus I2C :

ce bus est exploité via les deux broches SDA(PC5)/SCL(PC4).

6.4.Port série (USART):

émission /réception, série via les broches TXD(PD1)/RXD(PD0).

6.5.Comparateur analogique:

le comparateur analogique intégré dans le ATmega peut être utilisé à travers les deux broches AIN0(PD6) et AIN1 (PD7), ce comparateur peut déclencher une interruption.

6.6.Watchdog Timer programable:

L’ATmega possède un compteur dit de chien de garde programmable pour générer des interruptions à la fin de son comptage et il peut être utilisé comme étant un simple compteur.

Gestion d’interruptions (24 sources possibles) : en résumé

- Interruptions liées aux entrées INT0(PD2) et INT1 (PD3).
- Interruptions sur changement d’état des broches PCINT0 à PCINT23.
- Interruptions liées aux Timer 0, 1 et (plusieurs causes configurables).
- Interruption liéeau comparateur analogique.

- Interruption de fin de conversion ADC.
- Interruptions du port série USAR.
- Interruption du bus I2C.

 Remarque:

Les normes I2C(inter-integrated circuit) et SPI(Serial peripheral Interface) ont été crées pour fournir un moyen simple de transférer des informations numériques entre les capteurs et les microcontrôleurs.

La bibliothèque Arduino pour I2C et SPI facilitent l'utilisation de ces deux protocoles. Le choix entre I2C et SPI est en général déterminé par les périphériques que l'on souhaite connecter. Certains appareils offrent les deux standards, mais habituellement un périphérique ou une puce ne supporte qu'une seule des deux normes.I2C à l'avantage de n'avoir besoin que de deux connexions de signalisation (l'emploi de plusieurs périphériques sur les deux connexions est assez simple et on reçoit une confirmation que les signaux ont été correctement reçus).L'inconvénient est que la vitesse des données est inférieure à celle de SPI et qu'elles ne peuvent voyager que dans un seul sens à la fois (Simplex), ce qui abaisse encore plus le débit si des communications bidirectionnelles sont nécessaires .Il faut aussi connecter des résistances « pull-up » aux connexions pour s'assurer de la fiabilité de la transmission des signaux. L'avantage de SPI est qu'il a un meilleur débit et qu'il a des connexions d'entrée et de sorties séparées, si bien qu'il peut envoyer et recevoir en même temps (Full Duplex). Il utilise une ligne supplémentaire par appareil pour sélectionner le périphérique actif et il faut donc plus de connexions si on a de nombreux appareils à connecter.

7.Avantage et inconvénient l'arduino:

On cite ci-dessous les avantages et inconvénient de ces deux architecteurs

7.1. avantages:

- Diminution du l'encombrement du matériel et du circuit imprimer
- Simplification du tracé le circuit imprimé (plus besoin de tracé de bus)
- Augmentation de la fiabilité du système
- Intégration en technologie MOS ,CMOS ou HCMO (diminution de la consommation)
- Le microcontrôleur contribue a réduire les coûts a plusieurs niveaux.

- Moins cher que les composants qu'il remplace
- Diminution des coût de main d'ouvre (conception et montage)
- Environnement de programmation et de simulation évolués.

7.3. Inconvénients:

- Le microcontrôleur est souvent surdimensionné devant les besoins de l'application investissement dans les outils de développement
- Ecrire les programmes, les tester et tester leur mise en place sur le matériel qui entour le microcontrôleur
- Les microcontrôleur les plus intégrés et les moins coûteux sont ceux disposant de ROM programmable par masque
- Incompatibilité possible des outils de développement pour des microcontrôleurs de même marque
- Fabrication uniquement en grande sérié supérieur a 1000
- Défauts relatif car il existe maintenant systématique des versions OTPROM un peu plus chère (4).

8.Les Mémoires:

Trois types de mémoire sont utilisés dans la série ATMEGA, la mémoire programme FLASH, la mémoire de donnée SRAM et la mémoire morte de type EEPROM.

8.1. La mémoire programme:

La mémoire programme permet de stoker et de faire fonctionner le microcontrôleur, il contient de 4 à 256Ko de programme selon le modèle du microcontrôleur. Le nombre d'écriture sur cette mémoire est limité à 10.000, largement suffisant pour la majorité des applications.

8.2. La mémoire de donnée:

La mémoire de donnée contient les 32 registres de travail, les 64 registres de commande et la mémoire SRAM pour les variables du programme de 2048 octets pour le modèle ATMEGA 328 les relations entre espace physique et registre.

8.3. La mémoire morte:

La mémoire morte est de type EEPROM d'accès plus complexe contiendra la configuration du programme et les données importantes qui seront sauvé pendant l'absence de courant électrique. On peut écrire jusqu'à 100.000 fois dans l'EEPROM. La taille de l'EEPROM est fonction du modèle de microcontrôleur ATMEGA (de 256 bits à 8 Ko).

mémoire Flash (32 Ko) pour les programmes.

mémoire vive SRAM (2 Ko) pour les données.

mémoire EEPROM (2 Ko) pour les données de sauvegarde.

Discussion:

Nous avons présenté dans ce chapitre la carte que nous avons utilisée pour notre application. Cette carte présente des avantages d'être facile à implémenter et contient aussi la possibilité d'allier les performances de la programmation à celles de l'électronique. Plus précisément, dans notre application, nous allons réaliser et programmer un système électronique de mesure de vitesse que nous présenterons dans les prochains chapitres.

Chapitre 2 :

Principaux composants de notre application

Préambule:

L’objectif de ce chapitre est de présenter les composants que nous avons utilisés pour la mise au point de notre réalisation, à savoir des capteurs, des actionneurs et un afficheur de type LCD. Aussi, dans ce chapitre, nous avons mis en évidence la relation entre les capteurs et les actionneurs.

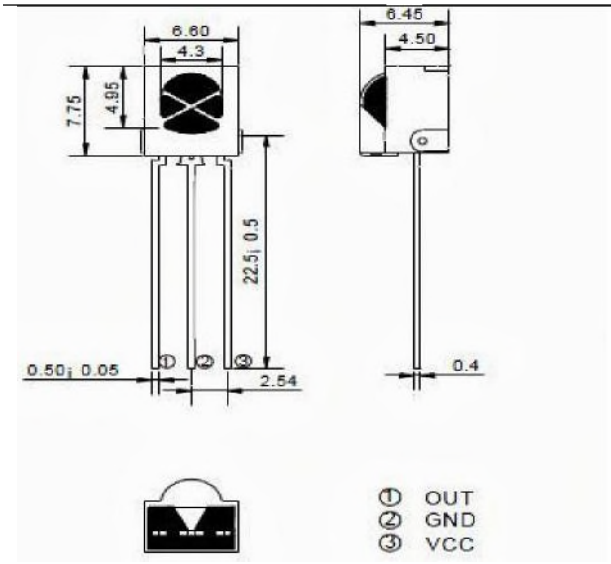
1.Capteur infrarouge tsop1838B et la télécommande:

On a besoin d’un capteur infrarouge commandé par une télécommande afin de contrôler notre application.

1.1. Capteur infrarouge :

Un capteur infrarouge modulé à 38 kHz qui est utile pour la plupart des télécommandes (dans certains cas, plus rares, on doit mettre un module à 56 kHz). Nous allons utiliser le TSOP1838B, mais il existe bien sûr d'autres possibilités.

Le circuit ci-dessous est sorti tout droit de la datasheet du TSOP1838B sur la figure (1et2); il sera peut-être un peu différent si vous utilisez un autre modèle. La sortie du capteur est normalement haute (5 V) mais elle devient basse (0 V) sur la réception d'un signal infrarouge qui oscille à 38 kHz. Lorsqu'on appuie sur un bouton de la télécommande en pointant cette dernière vers le capteur, la tension de sortie du capteur va changer de valeur très rapidement.



Figure(1) : Datasheettsop1838B

Figure(2): Tsop1838B

Mais pour décoder ce signal, l'Arduino est encore plus pratique qu'un oscilloscope.
Une fois la librairie IRRemote installée dans notre répertoire "Librairies", nous branchons notre capteur infrarouge à l'arduino ,la sortie du capteur va à l'entrée 2 de l'Arduino

Nous télé versons dans l'Arduino le sketch "IRrecvDump" disponible dans le menu des exemples sous l'onglet IRRemote, nous affichons le moniteur série ,et nous appuyez sur un bouton de la télécommande en visant le capteur.
La librairie a reconnu le protocole NEC à 32 bits (y'a aussi d'autre protocole : NEC, SONY, RC5 , RC6) , et la commande se traduit par la valeur hexadécimale. C'est tout ce qu'on a besoin de savoir, puisque le protocole a été reconnu, nous avons nullement besoin d'utiliser les données brutes qui apparaissent dans les lignes suivantes.

Tout d'abord on a décoder toutes les boutons de la commande par ses propres codes programmé a l'origine, pour les exploité et de commander ceux que nous voulons sur notre système.

1.2. Télécommande:

le telecommande sur la figure(3) suivante :

figure(3) : télécommande infrarouge

Les télécommandes infrarouge utilisent en général un signal constitué d'une suite de « rafales» d'oscillations de fréquence déterminée (38Khz le plus souvent), l'existence ou pas du signal et/ou les transitions de signal constituant des « zéro » et « un » du code émis.

2. Capteur de vitesse FC-33 :

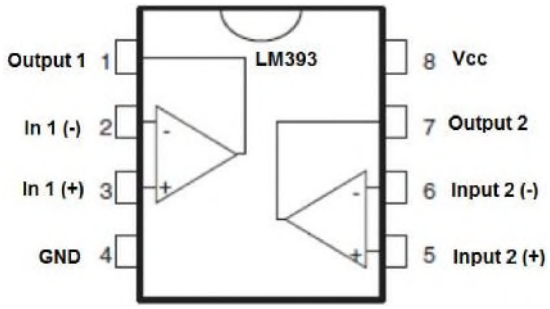
Module photo IR rupteur le quel qui est apparue sur la figure(4).

Figure(4) :Capteur de vitesse FC-33

Elle produit bas niveau qui correspond a 0V quand il n'y a pas de bloc dans la rainure, sinon il sortie de haut niveau correspondant a 5V.

2.1. Circuit intégré:

la puce LM393 qu'on montre sur la figure(5):

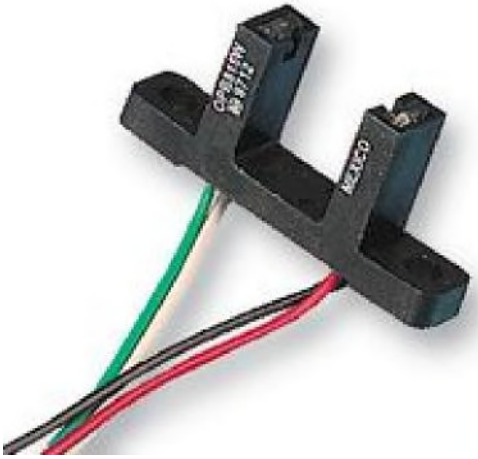


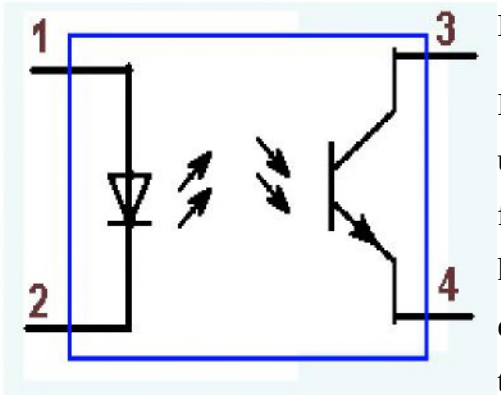
Figure(5) : Datasheet de circuit intégré LM393

2.2. Photocapteur H2010 :

Ce sont des disjoncteurs à infrarouge qui sont utilisés pour compter le nombre de tours du disque , détecter la présence d'un objet entre l'émetteur et le récepteur ,etc.

On montre son image et son schéma interne sur les figure(6 et 7).





Figure(6) :Image H2010

H2010 être utilisé pour fotoschetchika, capteur de vitesse, tourner à l'angle désiré, fin de course Contactless, etc. A bord il y a une lumière qui sort en présence d'un objet opaque dans l'alignement du photocapteur. Simultanément, le signal de sortie est généré. Le niveau de signal actif est faible. Mono-canal de signal de sortie.La sensibilité du capteur ne peut pas être ajustée. Tension de fonctionnement 5V. .Gabaritnye Dimensions: 32mm x 11mm x 20mm,[7].

3. Moteur à courant continu:

Prenons un moteur électrique simple sur le figure suivante:

Figure(8) : Image d’un moteur DC

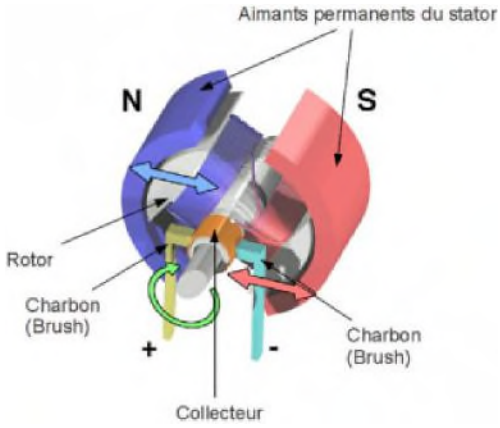
3.1. Définition de moteur:

Les moteurs à courant continu s’appelle aussi « Machine à Courant Continu »leur abréviation est MCC, son rôle est de transformer l’énergie électrique en énergie mécanique de rotation, pour être précis. Mais ils peuvent également servir de générateur d’électricité en convertissant une énergie mécanique de rotation en énergie électrique[5] .

Ce dernier point n’est pas à négliger, car même si dans la plupart des applications de notre moteur servira à générer un mouvement, il sera possible qu’il soit actionné en inverse et généré alors du courant. Il faudra donc protéger notre circuit pour ne pas l’abimer a cause de cette injection d’ énergie nos désiré.

3-2-Principe de fonctionnement du moteur à courant continu:

Cette figure suivante nous montre un schéma interne pour un moteur DC :



Figure(9) :schéma interne d'un MCC

Le MCC est composé de deux parties principales : le **rotor** (partie qui tourne) et le **stator** (partie statique, fixe).

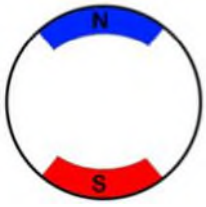
Dans notre projet, on va mettre la roue sur l'axe, bien souvent par l'intermédiaire d'un réducteur qui diminue la vitesse de rotation tout en augmentant le couple.

3.2.1. Le magnétisme:

Il possède un pôle Nord et un pôle Sud. L'aimant génère un champ magnétique permanent.

3.2.2. Le stator:

Le stator, est une partie immobile du moteur. Sur l'image, il se trouve sur les côtés contre le châssis. Il forme un aimant avec ses pôles Nord et Sud. Cet ensemble aimant+châssis constitue donc le stator qui se montre sur la figure suivante:

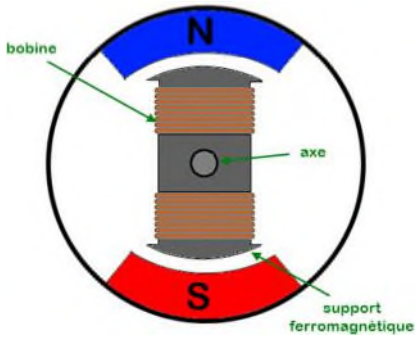


Figure(10) : Stator d'une MCC

3.2.3. Rotor et mise en mouvement:

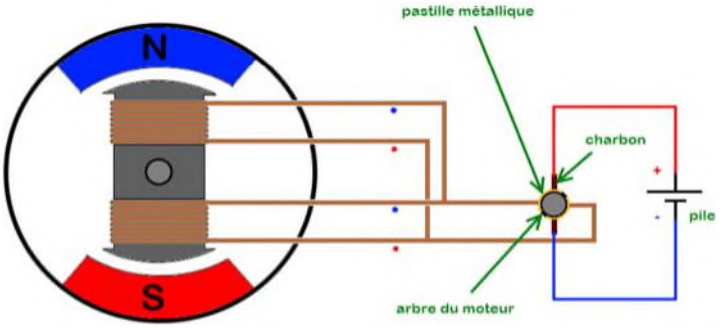
Le rotor qui se montre sur la figure(11), est situé au centre du stator, le rotor est la pièce maîtresse qui va recevoir un courant continu et va induire un champ magnétique variable pour mettre en rotation l'arbre du rotor.

Pour comprendre ce qu'il se passe, nous vous proposons de regarder comment est constitué un rotor de MCC:



Figure(11) : Rotor d'un MCC

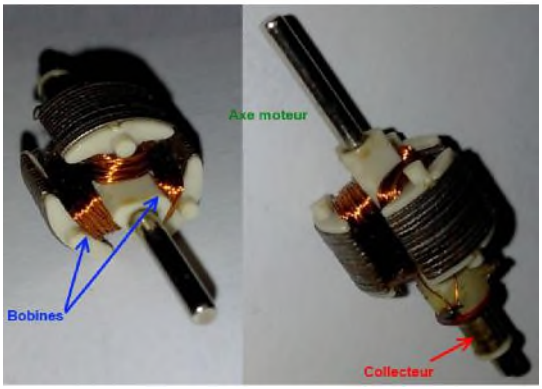
Le collecteur c'est un élément du moteur qui se situe sur l'arbre de rotation et qui a pour objectif de récupérer le courant afin de l'amener jusqu'aux bobines. Le Schéma complet du moteur avec les bobines et le collecteur sur la figure suivante :



Figure(12): Schéma complet du moteur avec les bobines et le collecteur

En réalité le MCC est constitué de trois bobines sur son rotor. Autrement on pourrait obtenir un équilibre qui empêcherait la rotation de l'arbre de moteur.

voilà une image réel sur la figure(13) du rotor d'un moteur à courant continu:



Figure(13) :rotor réel d'un MCC

3.3. Couple

L'unité du couple est le Newton-Mètre (Nm), Cette unité nous informe de deux choses : le couple est à la fois lié à une distance (le mètre) mais aussi à une force (le Newton). Maintenant on rajoute une information: le couple s'exprime par rapport à un axe. On peut conclure que le couple est la capacité du moteur à faire tourner quelque chose sur son axe. Plus le couple est élevé et plus le moteur sera capable de mettre en mouvement quelque chose de lourd. .[5]. Ce qui nous permet d'introduire la formule suivante:

$$C=F\times r$$

Avec:

- C : le couple, en Newton-mètre
- F : la force exercée, en Newton
- r : le rayon de l'action (la longueur du levier), en mètre

3.4. La vitesse de rotation

La vitesse de rotation est mesurée par rapport à l'axe de rotation du moteur. Imaginons que le moteur entraîne son axe, lorsqu'il est alimenté par un courant, ce dernier va avoir une vitesse de rotation. Il peut tourner lentement ou rapidement.

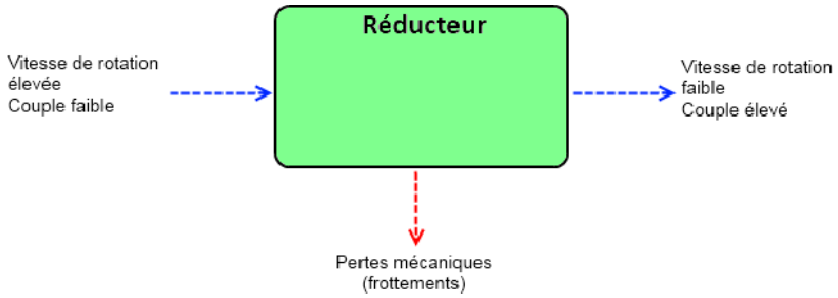
Et ce selon la relation suivante qui donne le **rapport de réduction**:

$$R=Wentree*Wsortie=Csortie*Centree$$

Avec:

- R : le rapport de réduction du réducteur
- Wentree: la vitesse de rotation de l'axe du moteur en entrée du réducteur
- Wsortie : la vitesse de rotation de l'axe du moteur en sortie du réducteur
- Csortie : couple exercé par l'axe de sortie du réducteur
- Centree: couple exercé par l'axe du moteur, en entrée du réducteur

Un réducteur s'apparente donc à un système qui modifie deux grandeurs qui sont liées : le couple et la vitesse. On peut schématiser le fonctionnement d'un réducteur de la manière suivante sur figure(14):



Figure(14) :schéma de le fonctionnement d'un réducteur

3.5. La puissance et le rendement

Dans un moteur, on trouve deux puissances distinctes:

- La première est la puissance électrique. Elle représente la quantité d'énergie électrique dépensée pour faire tourner l'axe du moteur, la formule suivante c'est pour calculer la puissance:

  Puissance = Tension x Courant (s'exprime habituellement en **Watts** (symbole **W**)).

- Selon les conventions, la tension est exprimée en Volt et le courant en Ampère. Quant à la puissance, elle est exprimée en **Watt** (**W**).
- La seconde est la puissance mécanique. Elle correspond au couple du moteur multiplié par sa vitesse angulaire :

  $P_{meca}=C\times W$

- Le couple doit être exprimé en Newton-Mètre (Nm) et la vitesse en radians par seconde (rad/s). Pour la puissance mécanique, il s'agit encore de Watt.

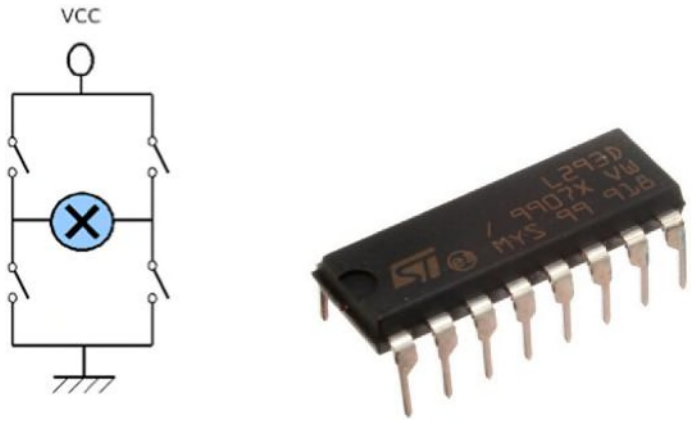
Lorsque le moteur est en fonctionnement, il génère des pertes. Ces pertes sont dues à différents phénomènes électriques ou thermiques (échauffement) ou tels que les frottements mécaniques (air, pièces en contact, magnétique).

3.6. Lien entre vitesse et tension:

Dans un moteur DC, quelque soit sa taille et sa puissance, il faut savoir que la tension à ses bornes et la vitesse de sortie sont liées. Plus la tension sera élevée et plus la vitesse sera grande. De préférence a rester dans les plages de tension d'alimentation de moteur et ne pas les dépasser. On peux dépasser de manière temporaire la tension maximale autorisée pour donner un coup de fouet à votre moteur, mais on reste jamais dans une plage trop élevée , la relation entre tension et vitesse n'est pas tout à fait linéaire pour les tensions faibles, elle est plutôt « écrasée ». C'est un peu comme si vous aviez une tension de seuil de démarrage. En dessous de cette tension, le moteur est à l'arrêt.

4. Circuit intégré L293D :

Ce module est basé sur le L293D qui est un double Pont-H



Figure(15) : le schéma de pont de H et un L293d Réel

4.1. Introduction

"Pont H à transistor pour contrôler un Moteur DC dans les deux sens" expliquait comment assembler un Pont H pour contrôler le sens de rotation d'un MCC [5].

Si le montage est un peu fastidieux, il permet de comprendre comment fonctionne un pont H. Pour l'utilisation de tous les jours, il existe des Ponts H pré-assemblés dans des circuits intégrés

Dans la gamme de circuit intégré "H Bridge Driver" disponibles, les plus courants sont le L293D (il possède les diodes de protection) et le L293E (il n'a pas des diodes de protection).

Contrôler un moteur DC avec Arduino et un circuit intégré L293D.
Un guide rapide avec quelques information complémentaire (configuration du circuit intégré etc) [5].

Ce guide présentant comment nous pouvons:

1. Utiliser une alimentation supplémentaire pour fournir la puissance nécessaire au moteur DC.
2. Utiliser le circuit intégré L293D pour piloter le moteur.
3. Utiliser un bouton poussoir pour changer la direction du moteur au bien un autre.

On va utilisé qu'un seul moteur ,mais il est possible de commander deux moteurs (dans les deux sens) avec un seul L293D chip. Dans ce cas, nous assurons de fournir assez de courant pour alimenter convenablement les deux moteurs pour les charges maximales.

“Le L293D un circuit intégré monolitique, à haut voltage, grand courant, 4 canal pilote.” Cela veut dire que ce circuit intégré peut être utilisé pour des moteurs DC et alimentation jusqu'a 36 Volts et que le circuit peut fournir un maximum de 600mA par canal.
Le L293D est aussi connu pour être une sorte de Pont-H. Typiquement, un pont H est un circuit électrique qui permet d'appliquer une tension en sortie sur une charge dans une direction ou l'autre[5].

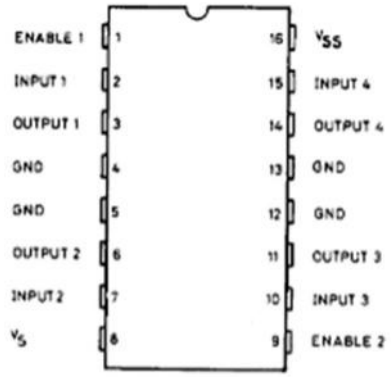
Cela signifie principalement que nous pouvons inverser la direction du courant et donc renverser le sens de rotation du moteur. Le principe de fonctionnement est basé sur 4 éléments du circuit communément nommés contacts.
En utilisant différentes combinaisons de fermetures, il est possible de démarrer, Stopper ou Inverser le courant.

Il est possible de faire ce montage à partir de relai mais il est plus simple d'utiliser un Circuit Intégré - Le L293D est double Pont-H convenable avec 1 Pont-H de chaque côté du circuit. La seule chose à laquelle il faut faire attention dans tout cela, ce sont les 2 broches d'entrée (Input Pins) qui commande la logique pour chaque moteur. Le plus important pour nos besoins, c'est que ces entrées sont commandables depuis l’Arduino.

Il n'est pas vraiment nécessaire de se préoccuper de la régulation de la tension parce que le L293d accepte deux sources de tensions.

- Une source d'alimentation directe (jusqu'a 36 Volts) pour alimenter les moteurs.
- Une autre source de tension (5 Volts) pour alimenter la logique du circuit intégré.

Il faut raccorder ensemble la masse (GND) d'Arduino et celle de l'alimentation externe. La masse doit être commune entre les alimentations (d'Arduino et du Pont-H). la configuration des broches du L293D et la table de la logique de commande sur la Figure(16)



Figure(16) : Datasheet du L293D

On va expliqué les déférents broche de L293D :

- VSS - Alimentation de la logique de commande (5V). A

raccorder à la borne +5V d'Arduino (donc sur le régulateur d'Arduino).

- VS - Alimentation de puissance des moteurs.

Par exemple, s'il s'agit d'une ancien véhicule téléguidé que vous avez canibalisé, il s'agira de la borne positive (+9.2v) de votre accumulateur.

- GND - Doit être raccorder à la masse (GND) de la source d'alimentation de puissance VS (donc le négatif de l'accumulateur) **et** à la masse de la source d'alimentation de VSS (donc GNDArduino).

Si vous n'avez qu'une source d'alimentation pour le tout, c'est forcement plus simple.

- OUTPUT1, OUTPUT2 - Broches à raccorder à la charge (le moteur). C'est via ces broches de
- INPUT1, INPUT2 - Broche de commande du Pont-H. Se raccorde a Arduino.
- ENABLE1 - permet d'envoyer (ou pas) la tension sur les sorties du moteur via OUTPUT1&OUTPUT2.

ENABLE1 commande l'activation du premier Pont-H.

Si ENABLE1 = GND, le pont-H est déconnecté et le moteur ne fonctionne pas.
Si ENABLE1 = VSS, le pont-H est connecté aux sorties et le moteur fonctionne dans un sens ou l'autre ou pas en fonction des tensions appliquée sur INPUT1&INPUT2.

4-2-Importance de la broche ENABLE1:

Si l'on ne désire pas faire de commande en vitesse et que le Pont-H sert uniquement en commande tout ou rien, alors dans ce cas, il suffit de raccorder ENABLE1 à VSS. Par contre, ENABLE1 offre l'avantage de pouvoir moduler la vitesse du moteur en y appliquant un train d'onde PWM.
Il est ainsi possible de commander assez finement une moteur DC avec 3 broches Arduino (dont une PWM).

Le tableau suivant explique le fonctionnement des entré sortie de L293d :

Enable 1	**Input 1**	**Input 2**	**Fonction**
High	Low	High	Tourne dans le sens**horlogique**
High	High	Low	tourne dans le sens **anti-horlogique**
High	Low	Low	Stop
High	High	High	Stop
Low	Non applicable	Non applicable	Stop

Tableau(1) : Fonctionnement des entrées sorties de L293

Dans ce cas pratique, le moteur que nous avons utiliser nécessite que 5 à 6 Volts mais pour bouger notre petite train il faut une puissance qui dépasse 12 volt.

Alimentation de notre Arduino avec une source de tension de 12 Volts. Mais cela la viderais assez vite si elle devait tout alimenter.

Donc, a la place, notre Arduino fonctionne avec une source d'alimentation séparée de 5 a 9 Volts.

4.3. Capacités de découplages:

On a besoin aussi de quelques capacités dans notre circuit pour réguler la puissance de charge aux moteurs autant que possible et ainsi éviter les pointes de courant et stabiliser le courant.

Nous pouvons aussi ajouter des capacités sur chaque moteur que nous utilisons - quelque-chose comme des capacités céramique 220nF multi-couche devrait être bon pour de petits moteurs.

4.4. Montage du circuit L293D pour Arduino:

On montre maintenant la schéma sur lab d'essaye comme la figure(17) le montre:

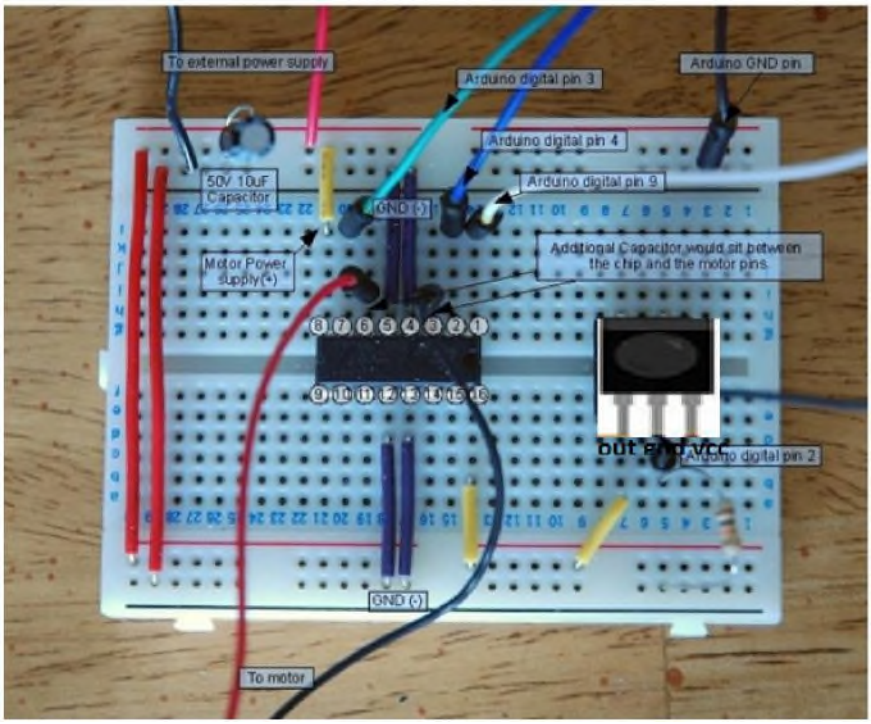


Figure (17) : schéma sur le lab d'essaie

Commençons par la pins 16 sur le L293D et tout ce qui doit y être raccordé. Comme nous pouvons le constater, il y a deux côté sur le L293D, 1 côté pour chaque moteur [5].

5.L’écrans LCD:

Avec les écrans LCD, nous allons pouvoir afficher du la vitesse instantané pour notre train miniature sur un écran qui n’est pas très coûteux.

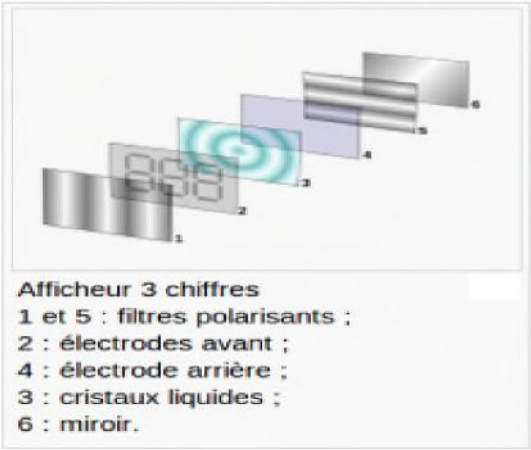
5.1. Définition d’un écran LCD:

LCD signifie « Liquid Crystal Display » et se traduit, en français, par « Écran à Cristaux Liquides » (mais on a pas d’acronymes classe en français donc on parlera toujours de LCD). Vous trouverez dans plein d’appareils électroniques disposant d’afficheur : les montres, le tableau de bord de votre voiture, les calculatrices, etc. Cette utilisation intensive est due à leur faible consommation et coût. Les écrans LCD sont aussi sous des formes plus complexes telles que la plupart des écrans d’ordinateur ainsi que les téléviseurs à écran plat [3].

5.2. Fonctionnement de l’écran:

Pour fonctionner il faut plusieurs choses. Si on regarde de très près notre écran, on vois une grille de carré. Ces carrés sont appelés des pixels.

Chaque pixel est un cristal liquide. Lorsque aucun courant ne le traverse, ses molécules sont orientées dans un sens (admettons, 0°). En revanche lorsqu’un courant le traverse, ses molécules vont se tourner dans la même direction (90°).On montrons une aperçue interne de LCD sur la figure suivante:



figure(18) : schéma interne d’un afficheur LCD

Parce que cette lumière est **polarisée**. Cela signifie que la lumière est orientée dans une direction . En effet, entre les cristaux liquides et la source lumineuse se trouve un filtre polariseur de lumière. Ce filtre va orienter la lumière dans une direction précise. Entre nos yeux et les cristaux se trouve un autre écran polariseur, qui est perpendiculaire au premier. Ainsi, il faut que les cristaux liquides soient dans la bonne direction pour que la lumière passe de bout en bout et revienne à nos yeux.

5.3. Commande du LCD:

Normalement, pour pouvoir afficher des caractères sur l'écran il nous faudrait activer individuellement chaque pixel de l'écran. Un caractère est représenté par un bloc de 7*5 pixels. Ce qui fait qu'un écran de 16 colonnes et 2 lignes représente un total de 16*2*7*5 = 1120 pixels .

5.4. Décodeur de caractères:

Tout comme il existe un driver vidéo pour notre carte graphique d'ordinateur, il existe un driver « LCD » pour notre afficheur. Ce composant va servir à décoder un ensemble de bits pour afficher un caractère à une position précise ou exécuter des commandes comme déplacer le curseur par exemple. Ce composant est fabriqué principalement par *Hitachi* et se nomme le **HC44780**. Il sert de **décodeur de caractères**. Ainsi, plutôt que de devoir multiplier les signaux pour commander les pixels un à un, il nous suffira d'envoyer des octets de commandes pour lui dire « écris moi 'zéros' à partir de la colonne 3 sur la ligne 1 ».

Ce composant possède 16 broches dans le tableau suivant:

N°	Nom	Rôle
1	VSS	Masse
2	Vdd	+5V
3	V0	Réglage du contraste
4	RS	Sélection du registre (commande ou donnée)
5	R/W	Lecture ou écriture
6	E	Entrée de validation
7 à 14	D0 à D7	Bits de données
15	A	Anode du rétroéclairage (+5V)
16	K	Cathode du rétroéclairage (masse)

Tableau(2) :Rôle des broches d'un afficheur LCD

Par la suite, les broches utiles qu'il faudra relier à l'Arduino sont les broches 4, 5 (facultatives), 6 et les données (7 à 14 pouvant être réduite à 8 à 14) en oubliant pas l'alimentation et la broche de réglage du contraste. Ce composant possède tout le système de traitement pour afficher les caractères. Il contient dans sa mémoire le schéma d'allumage des pixels pour afficher chacun d'entre eux.

Voici la table des caractères affichables résumé sur le tableau(3).

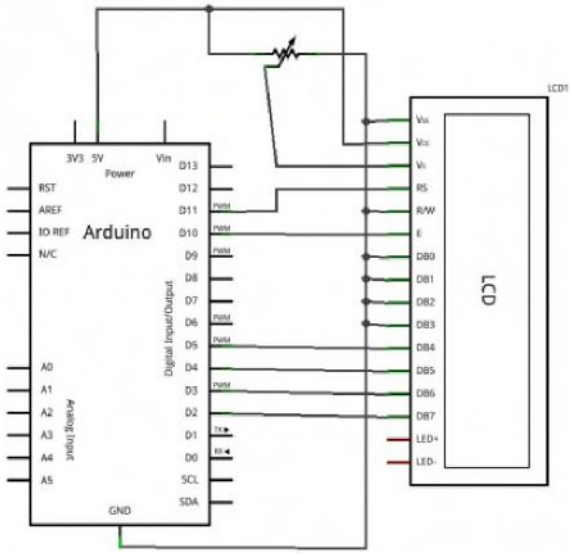
Tableau(3) : Tableau montre les caractère et leur code de LCD

5.5. Branchement:

L’afficheur LCD utilise 6 à 10 broches de données ((D0 à D7) ou (D4 à D7) + RS + E) et deux d’alimentations (+5V et masse). La plupart des écrans possèdent aussi une entrée analogique pour régler le contraste des caractères. Nous brancherons dessus un potentiomètre de 10 kOhms. Les 10 broches de données peuvent être placées sur n’importe quelles entrées/sorties numériques de l’Arduino. En effet, nous indiquerons ensuite à la librairie LiquidCrystal qui est branché où.[3].

5.6. Le montage à 4 broche de données:

La figure(19) montre les connections entre la carte UNO avec l’afficheur LCD avec 4 broche de données parce que on ’a pas besoin de des 4 autre broche de donnés.



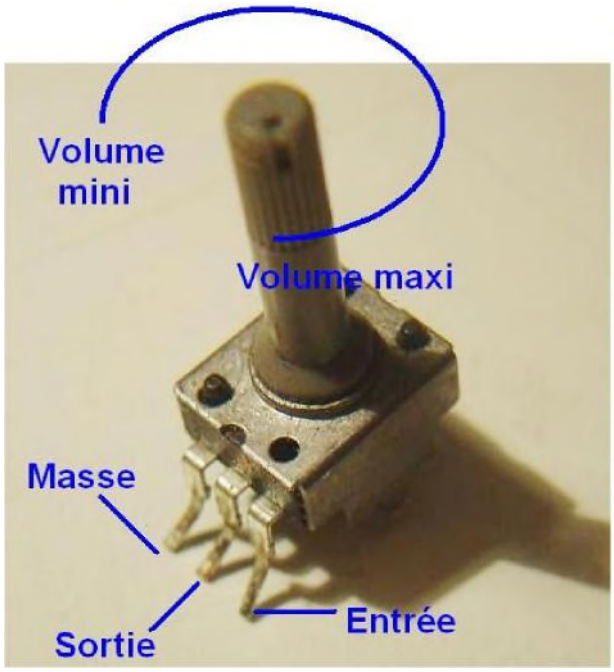
Figure(19) : Les connections entre l’UNO et LCD

5-7- Le démarrage de l’écran avec Arduino

Nous allons utiliser la librairie « LiquidCrystal ». Pour l'intégrer c'est très simple, il suffit de cliquer sur le menu « Import Library » et d'aller chercher la bonne. Une ligne #include « LiquidCrystal.h » doit apparaitre en haut de la page de code. Ensuite, il ne nous reste plus qu'à dire à notre carte Arduino sur quelles broches et quelle est la taille de ce dernier (nombre de lignes et de colonnes).

6. Potentiomètre:

On peut présenter le potentiomètre sur ce schéma sur la figure(20):



Figure(20) : caractéristique de potentiomètre

6-1- Mesures au multimètre (pour potentiomètre inconnu)

Le multimètre est utilisé en ohmmètre. On peut caractériser les bornes du potentiomètre de cette façon:

- entre les 2 extrémités, la résistance ne dépend pas de la position du curseur (on peut tourner l'axe du potentiomètre, la valeur au multimètre ne change pas). La résistance mesurée vaut

environ la valeur nominale (10kOhms, 50kOhms, etc). Si on mesure 94.3kOhms, il s'agira d'un potentiomètre 100kOhms.

- entre l'extrémité "masse" et la borne du milieu "sortie" : la résistance varie de zéro à la valeur nominale en fonction de la position du curseur. Du côté "volume mini", la valeur vaut 0. Du côté "volume maxi", la valeur vaut la valeur nominale.

6-2- Principe de fonctionnement du potentiomètre

Prenons un potentiomètre de 10kOhms. Il y a une résistance fixe (10kOhms ici) entre l'entrée et la masse. La borne de sortie est "entre" la masse et la sortie :

- entre sortie et masse, il y a **x** % de la résistance. Par exemple 2.5kOhms (25%) quand on est au quart de la course du potentiomètre.

- entre entrée et sortie, il y a alors **100-x** % de la résistance. Ici, il y a 7.5kOhms (100-25%).

Discussion:

Dans ce chapitre, nous avons présenté et expliqué les différents composants ainsi que leur principe de fonctionnement. Ces composants seront utilisés pour réaliser notre système qui es un calculateur et afficheur de vitesse placé sur un train miniature. Les détails de cette réalisation feront l’objet de troisième chapitre.

Liste des composants:

1. **Carte arduino UNO**
2. **Capteur infrarouge tsop1838B**
3. **Télécommande infrarouge**
4. **Capteur de vitesse FC-33**
5. **Moteur DC 12Volts**
6. **Circuit intégré L293D**
7. **Afficheur LCD**
8. **Potentiomètre (20kohm – 50 kohm)**
9. **Labd'essaie**
10. **Deux résistance (50 kohm)**
11. **Batterie de 12Volts**
12. **Les files de liaison**
13. **Un ordinateur**

Chapitre 3 :

Réalisation pratique et fonctionnement

Préambule:

Cette partie consiste à présenter notre application. En effet, notre application est un système permettant de calculer et d'afficher une vitesse instantanée commandée à l'aide d'une télécommande à infrarouge. Le système est posé sur un train miniature que nous avons aussi réalisé. Donc, le système mesure la vitesse du train. Cette réalisation se présente comme une ouverture pour mesurer des vitesses pour des systèmes réels dont le principe de mesure est rotationnel.

Pour bien mener ce travail, nous allons dans ce chapitre détailler le fonctionnement de toutes les parties constituant notre réalisation.

1. Fonctionnement de notre application:

La réalisation contient six blocs principaux, qui sont la télécommande à infrarouge, le circuit L293D (Pont de H), le moteur de rotation, le capteur FC-33, la carte Arduino (UNO) et l'afficheur LCD. Les étapes de fonctionnement peuvent être divisées en deux parties ; partie commande et partie mesure.

Dans la partie commande, nous avons trois étapes:

- Début de signal par la télécommande
- La réception du signal par arduino
- Fonctionnement de L293D et commande de moteur

La partie mesure contient trois étapes qui sont:

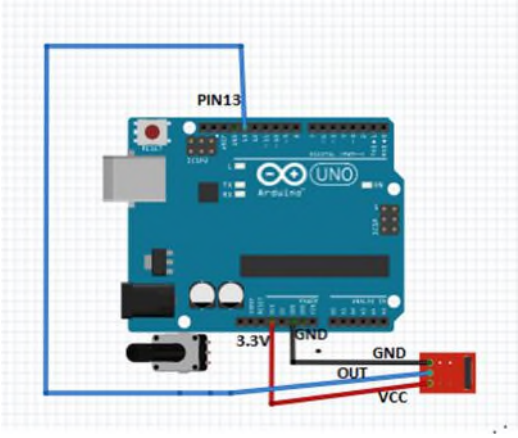
- Calcul de nombre de rotation
- Traitement de signal
- Affichage de la vitesse

2. Partie commande:

Cette partie consiste à contrôler en utilisant la télécommande la puissance et le sens de rotation de moteur sur lequel on veut calculer la vitesse. Pour ce faire, on opère comme suit:

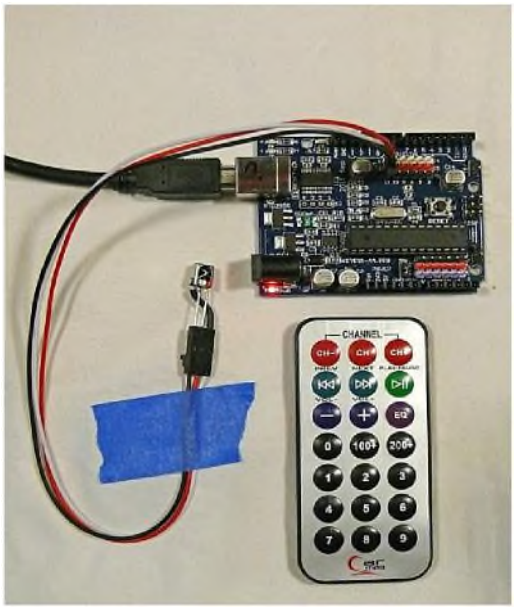
2.1. Début de signal par la télécommande:

Pour faire marcher le moteur, un signal infrarouge sera émis par la télécommande qui sera reçu par le capteur infrarouge. Ce dernier est connecté à la carte Arduino comme le montre la figure(1).



Figure(1): Schéma avec fritzing de montage.

la figure(2) montre l'émetteur et le récepteur infrarouge.



Figure(2) : Emetteur et récepteur infrarouge sur arduino

Ensuite, l'ordre est donné comme un code "morse", sauf qu'il ne s'agit pas de tiret-point mais de 1 (la diode s'allume) et de 0 (la diode s'éteint). Cela permet de produire des signaux de structure différente selon la commande sur laquelle on désire agir.

Le clavier constitue l'intermédiaire entre l'utilisateur et le système électronique de l'émetteur. Par exemple, en appuyant sur "pause", la télécommande envoie "10110101", sous forme d'une série de signaux électriques, à la diode. Celle-ci se met alors à clignoter, et c'est ce clignotement qui est capté par un récepteur.

Ce dernier il comporte un élément sensible non seulement à la lumière infrarouge émise par la diode, mais en plus capable de comprendre son langage codé.

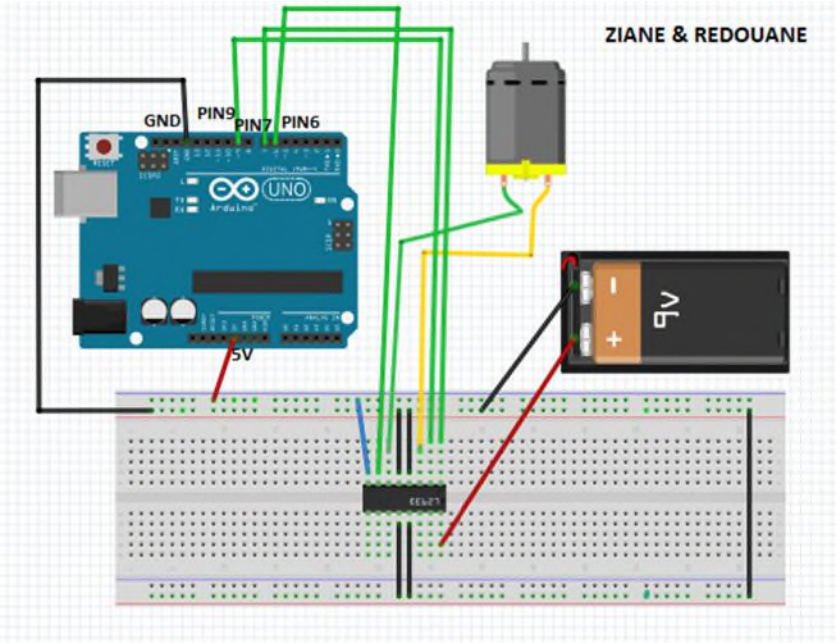
Les signaux de commande émis par l'émetteur sont captés par le récepteur monté sur le capteur , puis sont amplifiés et décodés par la librairie IRremot . A la sortie de ce décodeur, les impulsions permettant de commandé notre train.

2.2. La réception du signal par l’arduino:

Le signal est reçu à l’entrée 13 de l’arduino. En fonction du signal reçu qui sera traité et conditionné, un code obtenu à l’aide du programme que nous avons élaboré est généré sur les pins (6 ;7 ;9) dans notre cas , est envoyé vers le circuit L293D. Le programme inséré dans la carte Arduino permettant cette opération est indiqué ci-dessous.

2.3. Fonctionnement de L293D et commande de moteur:

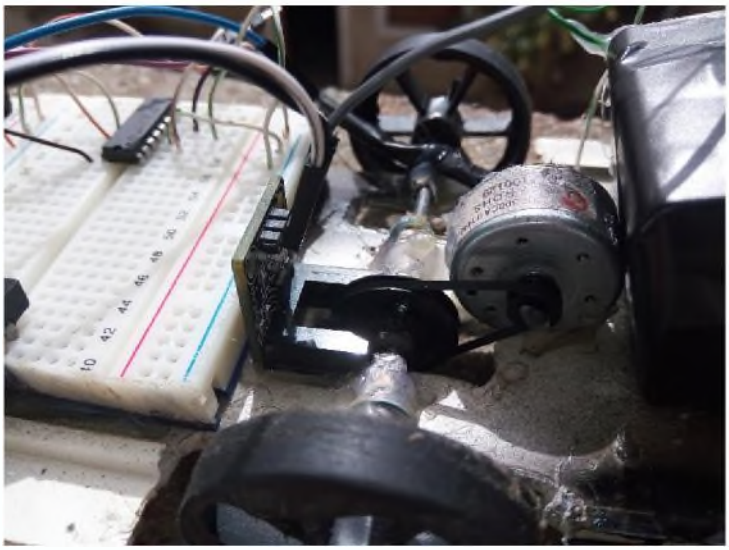
Le signal envoyé par Arduino est reçu dans le L293D qui permet déclencher la rotation de moteur tout en contrôlant le sens de rotation et la puissance comme la Figure(3) le montre.



Figure(3) : Schéma de connexion de L293d et le moteur vers l'arduino

Lorsque le moteur tourne, on pourra également se souvenir que plus la force exercée sur l'axe de rotation d'un moteur est grande, plus il faudra un couple élevé. Et plus le couple du moteur sera élevé, moins notre train aura de difficultés à supporter de lourdes charges. Cela dit, tout n'est pas parfait car plus la charge est lourde, plus la consommation électrique du moteur va augmenter.

Grâce à une croix raccordant le moteur à la roue (qui a un trou pour comptage de vitesse) sur l'axe de rotation (voir figure(4)).



Figure(4) : raccordement entre le moteur et l'axe de rotation

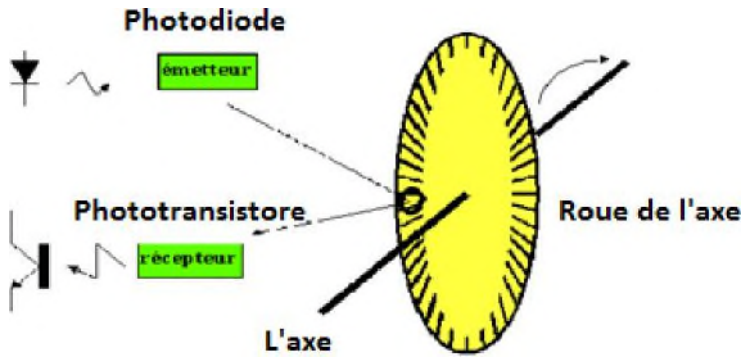
3. Partie mesure

Une fois le moteur est en marche, la partie suivante est la partie mesure qui consiste à mesurer la vitesse du train. A cet effet, trois étapes sont nécessaires :

3.1. Calcul de nombre de rotation:

Pour calculer le nombre de rotation, nous utilisons un capteur dont le rôle est de détecter une rotation complète. Le principe de cette détection est le suivant:

Indiqué par la figure(5), la roue contient un trous à travers lequel le rayonnement peut traverser. Le capteur utilisé est constitué d'un émetteur(photodiode) et d'un récepteur (phototransistor) qui sont placés parallèlement à la roue. Pour pouvoir détecter une rotation complète, la roue est placée entre l'émetteur et le récepteur. Une rotation est comptée lorsque le signal émis par l'émetteur est réceptionné au niveau de récepteur. En fonction de temps qui met le récepteur pour détecter la présence de rayonnement, nous pouvons estimer la vitesse. Cette vitesse est inversement proportionnelle à ce temps [2].



Figure(5): Schéma de fonctionnement d'un capteur de rotation.

3.2. Traitement de signal:

La carte Arduino reçoit d'une façon continue un état (niveau bas ou niveau haut) à partir du capteur par la broche 12. L'état haut est généré lorsqu'aucune présence de rayonnement n'est détectée. En revanche, l'état bas reçu en présence de rayonnement de niveau de récepteur.

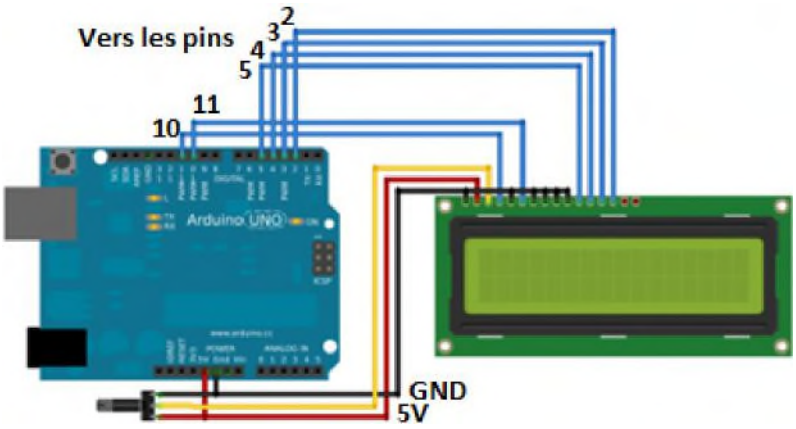
A l'aide de programme que nous avons élaboré, nous avons pu calculer la vitesse de train. La relation permettant de calculer la vitesse est donnée par l'équation suivante :

$$V(mm/ms) = P(mm) / T(ms)$$

Avec V est la vitesse de notre train miniature, P représente le périmètre de la roue et T c'est le temps que peut prendre une rotation complète.

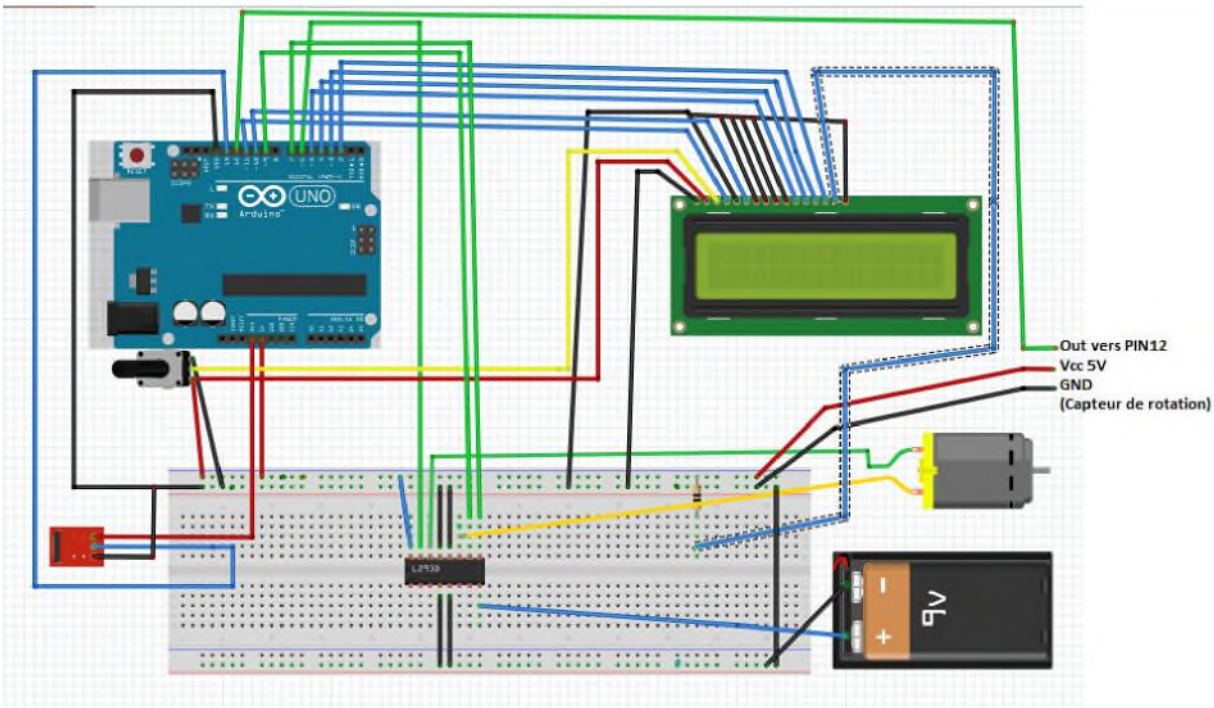
2.3. Affichage de la vitesse instantanée:

Pour afficher la vitesse, nous avons utilisé un afficheur LCD. Lorsque la vitesse est calculée, un code binaire qui dépend de la vitesse calculée sera envoyé vers l'afficheur LCD.
L'afficheur est connecté à la carte Arduino selon la figure(6) ,[3].



Figure(6): Schéma sur porteuse branchement de l'afficheur avec arduino.

Le schéma global de notre réalisation et le programme correspondant sont donnés respectivement par la figure(7) et la section 3. Le schéma est obtenu en utilisant le logiciel Fritzing.



Figure(7): Schéma général de notre application.

3. Programme arduino:

```
#include<IRremote.h> //ajout de la librairieIRremote.h

#include "LiquidCrystal.h" //ajout de la librairieliquidCrystal.h

//Vérifier les broches!

LiquidCrystallcd(11,10,5,4,3,1); //liaison 8 bits de données

int led=12;

intRECV_PIN =2;
```

```
IRrecvirrecv(RECV_PIN);// crée une instance
decode_resultsresults; // stockage données reçues
//declaration des variables

intmotor1Pin1 = 6; // pin 2 de L293D
intmotor1Pin2 = 7; // pin 7 deL293D
intenablePin = 9; // pin 1 deL293D
float v = 0.0;            // Variable de Vitesse
float s = 0.0;          // Variable de temps convertie en secondes
float t= 0.0;          // Variable temps en secondes
//fonction d'initialisation de la carte
void setup() {
irrecv.enableIRIn(); // démarre la réception
pinMode(enablePin, OUTPUT);// met la "enablePin" comme sortie
pinMode(led, INPUT);/ / met la "led" comme entré
lcd.begin(16, 2);}
//fonction principale, elle se répète (s'exécute) à l'infini
voidloop() {
int val= digitalRead(12); // lire la valeur entière numérique sur la pine 12
while(val==1){ // la boucle pendant val égale a 1 on va exécuter ce qui suit
int val= digitalRead(12); // reconnaitre la valeur de val
s=s+val;// on a initialisé la valeur de s à zéro et cette instruction va incrémenter s en lui
ajoutant la valeur de val qui est égale à 1 pour compter le temps de rotation
t =((float)s); //mettre la valeur de s dans t en second
```

```
if(val==0){ // la boucle si val égale à 0, c’est une condition

v = (12000 / (float)t); //calcul de la vitesse


lcd.clear();                    // on efface l'écran

lcd.print(v);                  // affichage de de la vitesse sur LCD

lcd.println(" mm/ms"); // affichage l’unité de la vitesse instantané

delay(12); // pause pendant 12 Millie second

s=0; //remettre s a zéro juste âpre affiché la vitesse chaque fois

}

while(irrecv.decode(&results)) { // pedant la reception de l’infrarouge par la command

irrecv.resume(); // reçoit le prochain code

// on condition maintenant les fonctionnalité des bouton de la commande pour marché notre train, soit avant au arrière, soit la vitesse de notre moteur , soit l’arrêté

if(results.value==1086279855){ // la boucle si le résultat de reception infrarouge est 1086279855


digitalWrite(motor1Pin1, LOW); // set pin 6 on L293D low

digitalWrite(motor1Pin2, HIGH); // set pin 7 on L293D high

analogWrite(enablePin, 245); // démarrage de moteur avec une vitesse max

}

if(results.value==1086296175){// la boucle si le résultat de réception infrarouge est 1086296175


 digitalWrite(motor1Pin1, HIGH); // set pin 6 on L293D high
```

digitalWrite(motor1Pin2, LOW); // set pin 7 on L293D low

analogWrite(enablePin, 245); // démarrage de moteur avec une vitesse max

}

if(results.value==1086259455){// la boucle si le résultat de réception infrarouge est 1086296175

digitalWrite(motor1Pin1, LOW); // set pin 6 on L293D low

digitalWrite(motor1Pin2, LOW); // set pin 7 on L293D low //

ce qui veux dire le moteur éteins alors le train va s'arrêté }

if(results.value==1086271695){// la boucle si le résultat de réception infrarouge est 1086271695

analogWrite(enablePin, 115);}

// la sortie enablePin qui control la puissance est de :255 égale 100%

//Alors 155 égale aune puissance de 45%

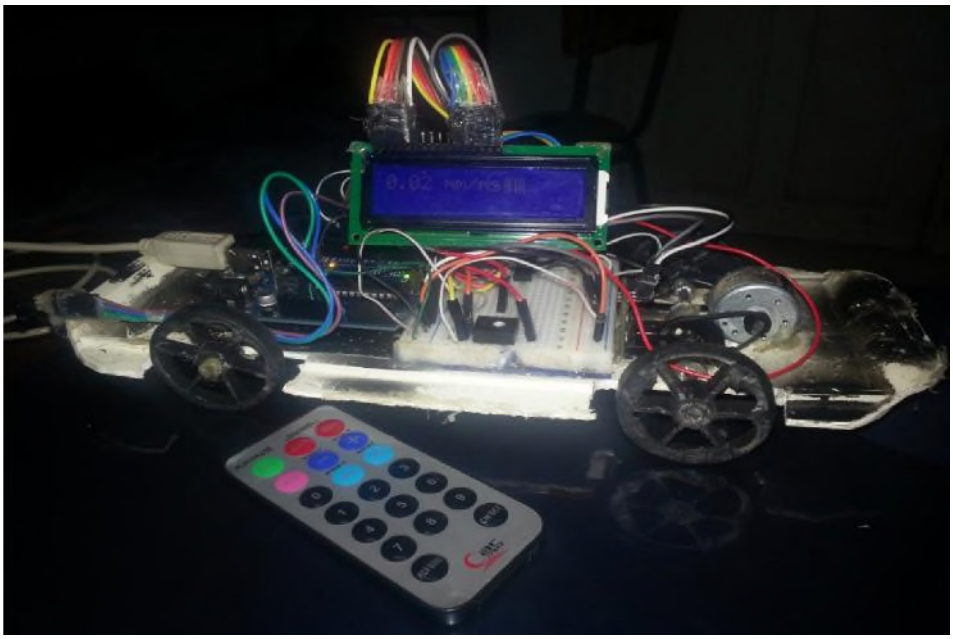
if(results.value==1086304335){// la boucle si le résultat de réception infrarouge est 1086304335

analogWrite(enablePin, 170);} // Alors 155 égale aune puissance de 66%

if(results.value==1086288015){// la boucle si le résultat de réception infrarouge est 1086288015

analogWrite(enablePin, 255);} }}}

A la aide ce programme on a arrivé à réaliser notre application qui apparait a la figure suivante:



Figure(8) : Aperçu globale de notre projet.

Discussions :

Dans ce chapitre, nous avons d'abord expliqué pratiquement et théoriquement le fonctionnement de notre projet. Ensuite, nous avons mis au point notre application qui consistait à mesurer la vitesse d'un train miniature. Elle est basée sur la carte Arduino et des capteurs infrarouges et de rotation.

Conclusion

La conception d’un système qui va mesurer la vitesse sont multiples. Dans ce travail, nous avons opté pour une méthode qui identifie une rotation et estime le temps de cette rotation. Le temps estimé permet de mesurer la vitesse de chaque rotation complète de la roue.

Ce projet peux être contrôlé par infrarouge ou Bluetooth ou wifi etc. L’avantage dans ce cas c’est d’éviter les boutons poussoirs ou toute commande filaires afin d’avoir une commande à distance. Cette réalisation est très bénéfique pour une application sur un système réel. Il s’agit de garder le même principe mais en changeant le capteur de rotation par un émetteur et un récepteur qui seront implémentés avec la carte Arduino ou un microcontrôleur.

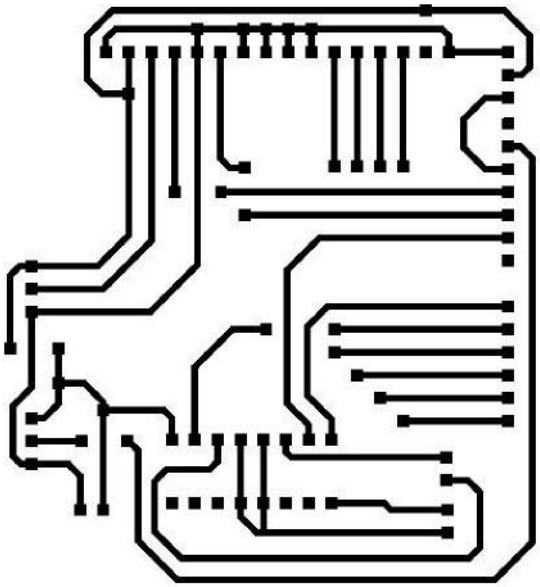
Rappelons notre application qui contient un afficheur LCD permettant d’afficher la vitesse peut être remplacé par un ordinateur. L’acquisition de la vitesse instantanée permet à l’utilisateur de la traiter, de la stocker et de l’envoyer. Avec cette procédure, il est possible de synchroniser tous les systèmes embarqués dans les différents trains.

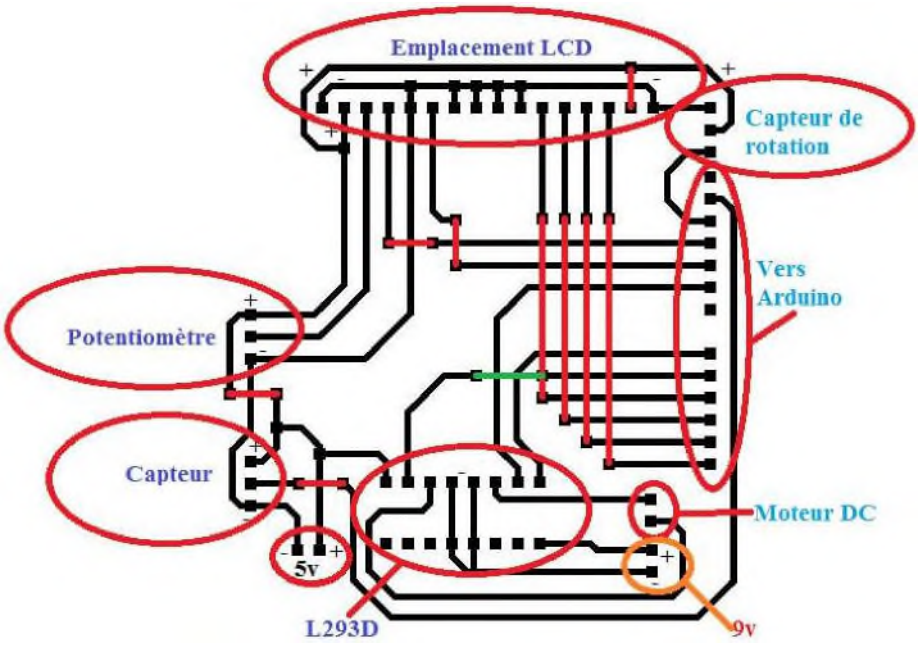
Après avoir réalisé notre projet, il est évidant que la carte Arduino est très utile pour assembler beaucoup de fonctions en un seul système, qui sera multifonction.

L’amélioration de ce système de mesure de la vitesse consiste à utiliser une mémoire de stockage dans le but de constater la puissance et la fiabilité des moteurs et les système de rotation. Aussi, il est possible de rendre le système manuel en système

semi-automatique ou complétement automatique en utilisant des capteurs permettant notamment d’indiquer la proximité devant des objets. La mise en marche d’un tel système nécessite le perfectionnement du programme.

Le typon de notre projet est réalisé comme suit:





Attention aux fils conducteurs en rouge et un en vert une fois que on réalise notre typon et lors de soudeur composants on dois souder ces fils en rouge et celui en vers du coté composant.

N'oublie pas que si on veux réalise ce typon on dois minimiser sa taille avec paint tout d'abord puisque les espacement entre les pins sont standards, il faut juste minimiser sa taille parce que en réalité ce typon est d dimension ne dépassant pas 6cm carré

Bibliographie:

[1] : X. HINAULT Ateliers Arduino www.mon-club-elec.fr – 2012

[2] : Patrick ABATI 07/12/2000 « Capteur de vitesse et de position» http://sitelec.org/cours/abati/captvit.htm

[3] : **Hippolyte Weisslinger (olyte) et simon landrault(Eskimon) «** Arduino: Premiers pas en informatique embarquée » Le blog d'Eskimon Édition du 01juin 2004

[4] : Michel MARTIN, Lycée Claveille, Périgueux « **Les principes de fonctionnement des capteurs de lacet sur véhicules automobiles »** 12-11-2002.

[5] : [Arduino] Contrôler des moteurs DC avec le composant L293D

January 13, 2014 , http://www.zem.fr/

[6]: 2014 Y. Reiser [*Position. Vitesse. Accélération* « *Mécanique: Cinématique du point»]*

www.lnw.lu/Departements/Physique/personnel/reiyv/documents/2bc/cours B01.pdf