UNIVERSITE MOULOUD MAMMERI TIZI-OUZOU

Faculté du Génie Electrique et de l'informatique

Département d'électronique

Mémoire de fin d'études

En vue de l'obtention du diplôme MASTER en Electronique

Option : Réseaux et Télécommunication

Thème :

Conception, étude et réalisation d'un banc de mesure pour la surveillance d'un moteur à base de la carte de développement ARDUINO UNO.

Proposé et dirigé par :

Pr. M.LAGHROUCHE
et ***Mr.Nachef Mhenna.***

Réalisé par :

Boudjenah Djamel .

Promotion 2013-2014

Remerciements

Je remercie le Bon DIEU de m'avoir donné la force et le courage d'accomplir ce travail.

Je remercie mon promoteur Pr.M.LAGHROUCHE pour son aide, son orientation et ses conseils durant l'accomplissement du projet.

Je tiens à remercier également Mr. Nachef Mhenna pour son aide très précieuse et son soutient tout au long de mon travail.

Je remercie également les membres du jury d'avoir accepté de juger ce modeste travail.

Dédicace

Je dédie ce travail à mes chers parents qui m'ont soutenu tout au long de mon travail et mes études. Je dédie aussi ce travail à mes deux sœurs Zahra et Lila de m'avoir aidé et encouragé et à leurs enfants Amine, Dehbia et les deux petites Binouche et Nina.

A tous mes amis en particulier Nachef Mhenna, Temmam Rabah, Hichem, Bounaceur Mounir, Nechiche Sofiane, Khelil Kamel, Ketami Idris,Ahnouche Djamel et Gouri Brahim.

A Fari qui m'a aidé et soutenu et encouragé jusqu'au bout.

A toute personne qui m'a aidé et soutenu de près ou de loin pour la réalisation de mon travail.

Sommaire :

Introduction générale.

Chapitre I : Défaillances dans les moteurs électrique.

Introduction 1

I .L'influence des différents paramètres sur les moteurs électriques 2

I.1.L'influence de la température 2

I.2.Influence de la vitesse de rotation 2

I.3.Influence des vibrations 3

I..3.1.Les sources de vibration 3

I.3.2.L'analyse vibratoire 5

I.3.3.Le but de l'analyse vibratoire 5

I.3.4.Structure des signaux vibratoires 5

Chapitre II : Description de la carte et des capteurs utilisés.

II.1. Description de la carte ARDUINO UNO 7
II.1.1. Avantages de la carte ARDUINO UNO 8
II.1.2.Alimentation de la carte ARDUINO 9
II.1.3.Protection du port USB contre la surcharge en intensité 10
II.1.4. Gestion de la mémoire 10
II.1.5.Les entrées/ sorties numériques 10
II.1.6. Communication avec l'extérieur 12
II.1.7. Le microcontrôleur ATMega328 13
II.1.8.Les principales caractéristiques de l'ATMega328..... 14
II.2. Description des capteurs utilisés 17
II.2.1. Le capteur de température DS18B20 17
II.2.1.1. Brochage du DS18B20 18
II.2.1.2. Schéma bloc du DS18B20 18
II.2.1.3.Mesure de la température par le DS18B20 19
II.2.1.4. Structure de la mémoire du DS18B20 20
II.2.1.5. Le bus One-Wire..... 21
II.2.1.6.Emission d'un bit du maître vers l'esclave 23

II.2.1.7.Réception d'un bit par le maître 23
II.2.1.8.Organigramme du fonctionnement du DS18B20 24
II.2.1.9. Schémas du branchement du DS18B20 25
II.2.2. Le capteur d'accélération BMA180 25
II.2.2.1. Bloc diagramme du BMA180 26
II.2.2.3. Le bus I2C 27
II.2.2.4.Le protocole I2C 27
II.2.2.5.Transmission d'un octet 28
II.2.2.6.Transmission d'une adresse 28
II.2.2.7.Brochage du BMA180 30
II.2.3.Mesure de la vitesse de rotation du moteur 31
II.2.3.1.L'effet Hall 31
II.2.3.2.Le capteur effet hall UGN3503 31
II.2.3.3.Quelques applications 32
II.2.3.4.Principe de la mesure de la vitesse 33
II.2.3.5.Schéma électrique 34
II.2.4.Détecteur de bruit 35
II.2.4.1.Le TDA2822 35
II.3. Le module de transmission 38
II.3.1.Historique 38
II.3.2 Le module bluetooth de Sparkfun 39
II.3.3.Les caractéristiques du module BlueSMiRF 39
II.3.4. Brochage du module BlueSMiRF 39

Chapitre III : Programmation et conception logicielle.

Introduction 43
III.1.L'environnement de programmation Arduino 43
III.1.1.Description du logiciel Arduino 44
III.1.2.Description de la barre des boutons 46
III.1.3.Description de la barre des menus 47
III.1.4.Les étapes de téléversement d'un programme vers la carte Arduino 48
III.2. Le logiciel BlueSoleil 49
III.2.1.Description de BlueSoleil 50

III.3. Le logiciel de programmation graphique LabVIEW 51

III.3.1.Les instrument virtuels (VI) 52

III.3.2.Les données de bases dans LabVIEW 54

III.3.3.Description de l'interface réalisé 55

III.3.4.Configuration du port 56

III.3.5.Configuration de la taille du tampon d'entrée/sortie 57

Conclusion

Bibliographie

Introduction générale :

Les machines électriques tournantes sont utilisées dans divers domaines comme le transport (les trains et les véhicules motorisés), la production de l'électricité (les alternateurs), l'industrie de production, ou encore l'électroménager. De plus,l'emploi des systèmes électroniques est en constante progression ce qui leur donne une large application.

Les défaillances principales dans les moteurs électriques sont dûes aux vibrations ou aux températures élevées qui dépassent le seuil autorisé.A titre d'exemple, le fonctionnement d' un moteur sous une température dépassant le seuil permis réduit de moitié sa durée de vie.

Les coséquenses de ces défaillances se voient au démarrage, une tempéraure élevevée du moteur et des vibrations palpables.Tout cela se répercute sur le rendement de la machine.

Ainsi, un système de surveillance du moteur est assez important pour réduire les risques de panne, préserver le bon rendement et ses performaces et d'augmenter sa durée de vie.

Le projet développé ici consiste à surveiller les paramètres essentiels qui provoquent la détérioration et l'usure du moteur électrique.Il est basé sur l'acquiqition de quatre paramètres déterministes qui sont : la température à l'intérieur du moteur, sa vitesse de rotation, les vibrations causées par son fonctionnement et le bruit qui est émis.Ce travail est réalisé avec la carte ***ARDUINO UNO*** qui sera présentée dans les chapitres à venir.

Le banc de mesure comporte :

- L' acquisition des ses quatre paramètres.
- Le transfert via la liaison BLUETOOTH et affichage dans un interface graphique sous LabVIEW.

Le travail est réparti comme suit :

- Introduction générale.
- Chapitre I : Les des défaillances dans les moteurs électriques.
- Chapitre II : Description de la carte et des capteurs utilisés.
- Chapitre III : programmation et conception logicielle.
- Conclusion générale.

Chapitre I

Les défaillances dans les moteurs électriques.

Introduction :

La majorité des pannes dans les moteurs électriques sont classées en deux principales familles qui sont les défaillances mécaniques et les défaillances électriques.

L'origine de la défaillance peut être soit :

- Ø Défauts de fabrication du moteur.
- Ø Le résultat d'une utilisation inadéquate.
- Ø Phénomènes apparaissant lors du fonctionnement comme l'échauffement et les vibrations.

L'origine d'un défaut dans un composant peut être l'usure, la mauvaise fabrication, un mauvais montage, une utilisation inadéquate ou l'ensemble de ces causes. Le mieux c'est que la défaillance soit détectée le plus tôt possible.

Les résultats des études menées sur les principales défaillances dans les moteurs électriques sont montrés dans la figure suivante :

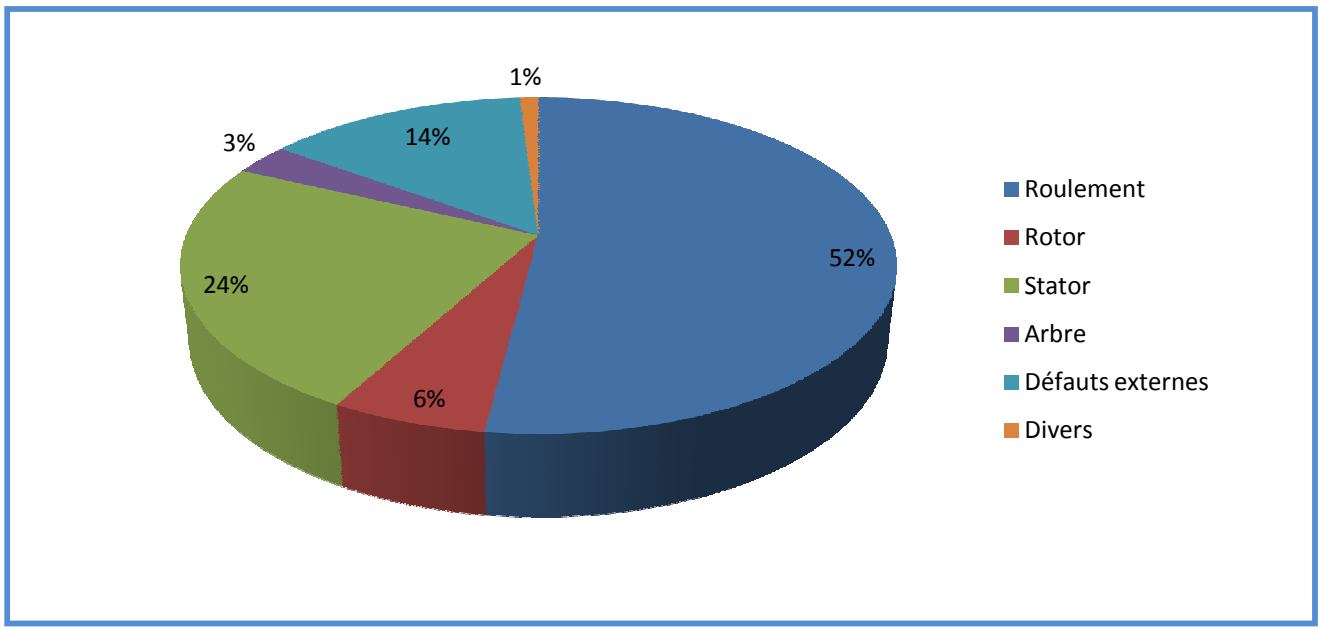


Figure 1.1.Les défaillances dans un moteur électrique (TD99).

Les défaillances dans les moteurs électriques souvent sont liées aux phénomènes physiques apparaissant lors du fonctionnement. Une partie de l'énergie consommée par le moteur est dissipée par effet joule sous forme de chaleur pouvant causer des dommages aux structures les plus fragiles du moteur comme les isolations électriques.

La rotation du rotor d'un moteur électrique engendre des vibrations transmissibles jusqu'aux structures avoisinantes via les points de fixation qui peuvent causer des anomalies de fonctionnement pouvant même provoquer l'arrêt complet de la machine. L'enroulement statorique du moteur est soumis à de rudes conditions comme une surcharge thermique et les vibrations mécaniques.

I .L'influence des différents paramètres sur les moteurs électriques :

I.1.L'influence de la température :

Pendant le fonctionnement du moteur, il faut prendre en considération le paramètre de température. En effet, lors la fabrication du moteur, il faut tenir compte des matériaux utilisés pour la fabrication des composants, il faut qu'ils soient adaptés aux seuils de températures admises dans le moteur. Cependant, la température peut dépassée les limites admissibles causant la panne. L'utilisateur doit être en mesure de prévenir les signes précurseurs de cette anomalie et prendre les mesures nécessaires pour éviter d'endommager le moteur.

Le maintien d'une température élevée peut endommager le système d'isolation des enroulements. Le point chaud est généralement situé dans le bobinage augmentant ainsi la défaillance des isolants électriques.

I.2.Influence de la vitesse de rotation :

Chaque moteur a une plaque signalétique sur laquelle est mentionné son point nominal. Exemple :

- Vitesse : 1500 tr/min.
- Puissance en KW.

La variation de la vitesse est un dispositif de réglage de la vitesse du moteur. Il est choisi selon le point de fonctionnement nominal. Ainsi, le constructeur du moteur doit fournir à l'utilisateur la courbe caractéristique du couple résistant/vitesse du moteur. De ce fait, le variateur peut fournir un couple important c'est-à-dire une puissance croissante jusqu'à arriver sa puissance nominale indiquée sur la plaque signalétique.

Au-delà de cette puissance, le moteur peut devenir instable provoquant un déséquilibre u rotor et une défection des roulements.

I.3.Influence des vibrations :

I.3.1.Les sources de vibration :

Toutes les machines tournantes en fonctionnement produisent des vibrations qui sont la conséquence des pièces en mouvement. Ainsi, un moteur neuf en excellent état de fonctionnement produit très peu de vibrations.

La figure suivante montre les vibrations dans un moteur électrique tournant.

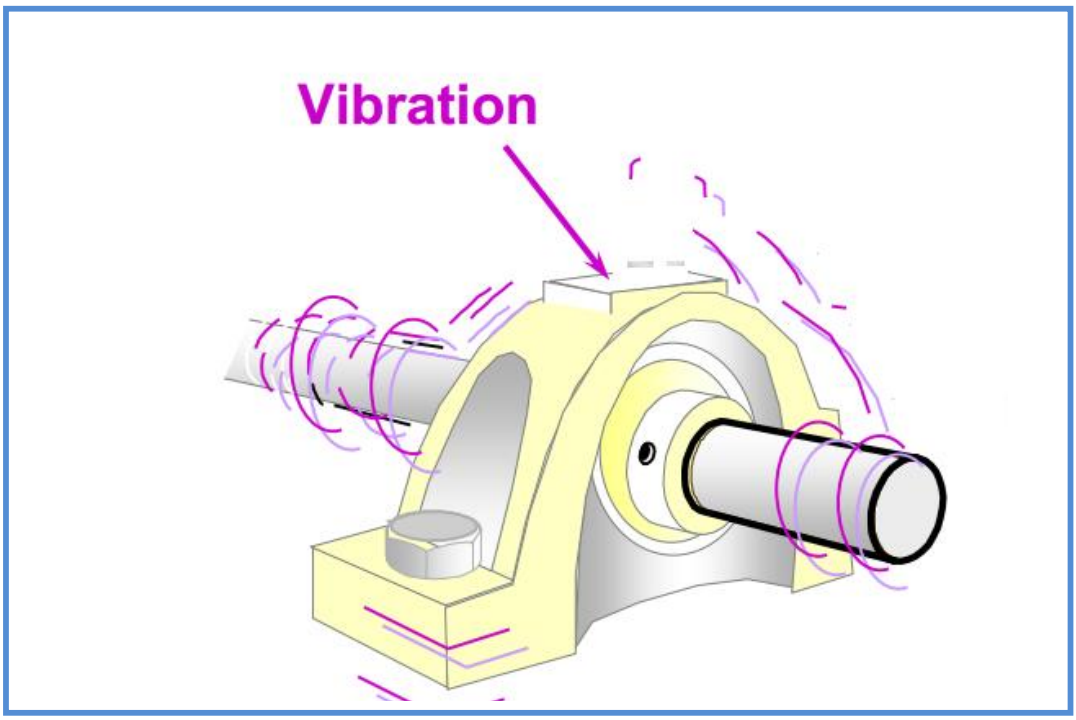


Figure 1.2.Les vibrations dans un moteur électrique tournant.

La détérioration du fonctionnement conduit le plus souvent à un accroissement du niveau de vibration et peut engendrer des dissipations de chaleur. Ces vibrations peuvent être transmises au reste du moteur suivant les points de fixation. En observant l'évolution de ce niveau, il est par conséquent possible d'obtenir des informations très utiles sur l'état du moteur ainsi prévenir la panne. Ces vibrations occupent une place privilégiée parmi les paramètres à prendre en considération pour faire un diagnostic.

La modification de la vibration d'un moteur constitue souvent la première manifestation physique d'une anomalie, cause potentielle de dégradation, voire une panne.

Le comportement dynamique des moteurs est identifié par l'utilisation des capteurs d'accélération qui transforment les vibrations engendrées en signaux électriques.

La figure qui suit représente l'emplacement d'un capteur d'accélération sur un moteur électrique.

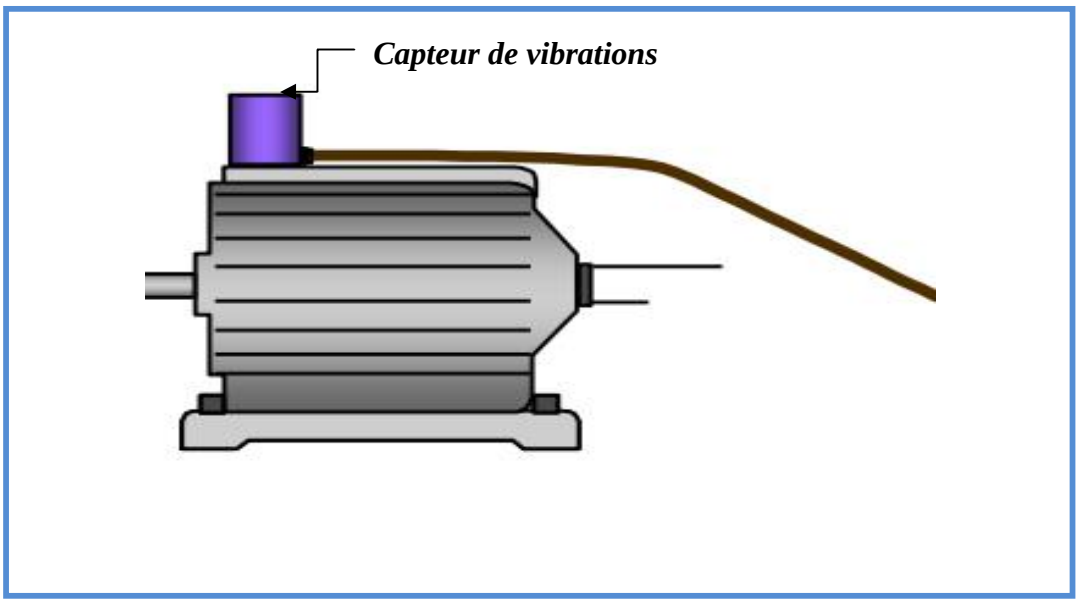


Figure 1.3.capteur d'accélération dans un moteur électrique.

I.3.2.L'analyse vibratoire :

L'analyse des vibrations permet la détection précoce des défaillances ce qui permet d'éviter d'atteindre un stade critique de la panne. Son emploi cible trois niveaux d'analyse :

- La surveillance.
- Le diagnostique.
- Le suivi de l'état d'endommagement des composants.

I.3.3.Le but de l'analyse vibratoire :

Son but est d'identifier le comportement vibratoire des moteurs ou bien le comportement dynamique d'un ensemble de composants prédéfinis. En se basant sur les résultats obtenus, nous pourrons identifier la partie des vibrations qui nous intéresse. Cette identification va permettre d'établir un diagnostic fiable en vue d'une maintenance.

L'intérêt est de séparer la contribution des différentes sources vibratoires liées à une défaillance plus au moins importante d'un composant à partir de plusieurs mesures réalisées via l'accélérateur. Cette séparation permettra la localisation des défaillances des composants et de suivre l'évolution de l'endommagement à fin d'améliorer les conditions de fonctionnement mécaniques et électriques.

I.3.4.Structure des signaux vibratoires :

Les signaux vibratoires prélevés sur les systèmes mécaniques en fonctionnement contiennent l'information nécessaire relative à l'état des composants de la machine. L'isolation de l'information relative à chaque composant est un sujet très important pour le diagnostic.

La figure suivante représente un exemple de la DSP de l'enveloppe carrée d'un signal vibratoire.

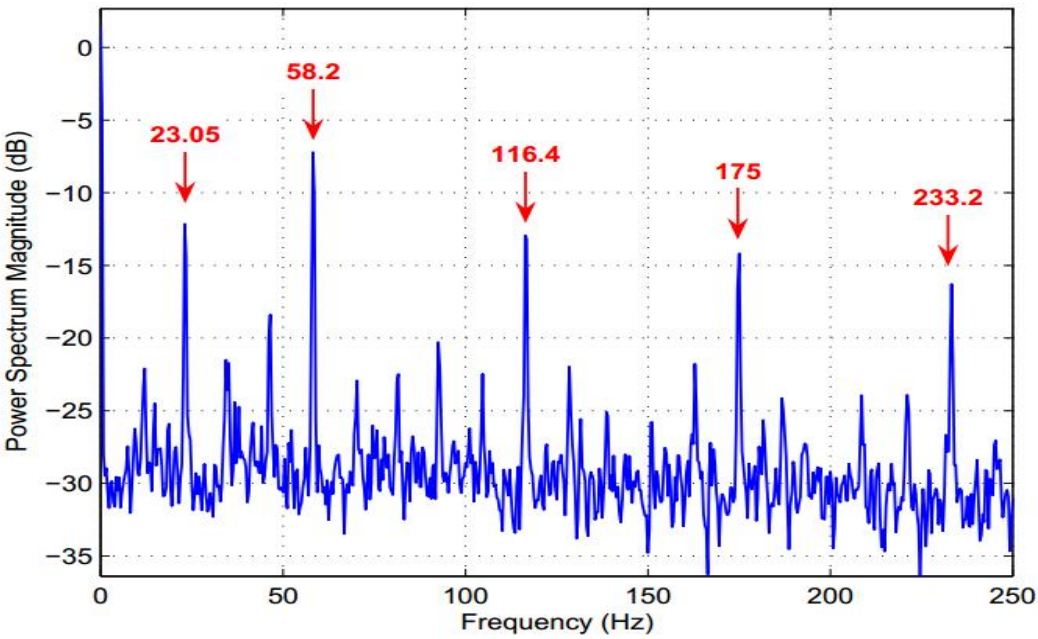


Figure 4.DSP de l'enveloppe carrée du signal vibratoire.

Conclusion :

Le but de notre projet est de déterminer les signes précurseurs de panne d'une machine électrique tournante et de l'éviter en intervenant avant que cela à savoir la te arrive. Les paramètres qu'on va mesurer à savoir la température, l'accélération, la vitesse et le bruit sont les principaux signes déterminant le mauvais état de la machine. Nous allons voir en détails, dans les chapitres à venir, comment qu'on est arrivé à notre but qui est la conception et la réalisation d'un banc de mesure de ces paramètres.

Chapitre II

Description de la carte Arduino uno et des capteurs utilisés.

Introduction :

Nous allons maintenant voir notre carte et les capteurs utilisés dans notre projet. Ce chapitre est consacré à la description des composants utilisés ainsi qu'à la réalisation pratique de notre projet.

II.1. Description de la carte ARDUINO UNO :

Arduino Uno est une carte électronique basée autour d'un microcontrôleur ATMEL de type ***Atmega 328*** de la famille ***AVR*** pour réaliser des fonctions plus ou moins évoluées à un coût bas. Elle possède une interface ***USB*** pour la programmer et l'alimenter. C'est une carte multiplateformes et open-source qui possède un logiciel de programmation, véritable environnement de développement intégré, pour écrire, compiler et transférer le programme vers la carte. L'intérêt principal des cartes ***Arduino*** (d'autres modèles existent) est leur facilite de mise en œuvre. ***Arduino*** fournit un environnement de développement s'appuyant sur des outils open source.

Le chargement du programme dans la mémoire du microcontrôleur se fait de façon très simple via port USB. En outre, des bibliothèques de fonctions "clé en main" sont également fournies pour l'exploitation d'entrées-sorties courantes : gestion des E/S TTL, gestion des convertisseurs ADC, génération de signaux PWM, exploitation de bus TWI/I2C, exploitation de servomoteurs et d'autres applications.

Elle comporte six entrées analogiques, repérées **A0 à A5** (au niveau du Atmega 328 **PC0 à PC5,** voir ***figure 2.1***), peuvent admettre toute **tension analogique** comprise entre **0 et 5 V**. Ces entrées analogiques sont gérées par un **convertisseur analogique/numérique de 10 bits** dont la sortie peut varier de **0 à 1023**. Les entrées **A4** et **A5** peuvent également être utilisées respectivement comme la ligne de donnée **SDA** et la ligne d'horloge **SCL** de **l'interface série I2C**. La carte ***Arduino Uno*** dispose aussi de 14 entrées/sorties numériques.

Chacune des 14 broches numériques (repérées 0 à 13 et dont 6 pouvant être utilisées comme étant des sorties **PWM (*Pulse Width Modulation)*** qui sont les broches numérotées 3, 5, 6, 9,10 et la broche numéro 11) peut être utilisée en entrée (input) ou en sortie (output) sous le contrôle du programme. Le sens de fonctionnement pouvant même changer de manière dynamique pendant son exécution. Elles fonctionnent en logique TTL (0V/5V) ; chacune pouvant fournir (source) ou recevoir un courant maximal de 40 mA et dispose si besoin d'une résistance interne de 'pull-up'. Elle est équipée aussi d'un quartz de 16 MHz, d'un connecteur ***ICSP (In Circuit Serial Programmation)*** qui permet d'injecter le bootloader à l'intérieur du microcontrôleur, un connecteur jack pour une alimentation extérieure, un bouton de reset pour remettre le processus à zéro. L'avantage avec cette carte c'est qu'elle n'a pas besoin de pilote pour faire

la conversion **FTDI USB/série**, elle a juste un petit microcontrôleur ***Atmega 8*** (pour la version 2) programmé comme convertisseur **USB/série**.

La figure ci-dessous représente la carte ***Arduino Uno*** et ses différents composants.

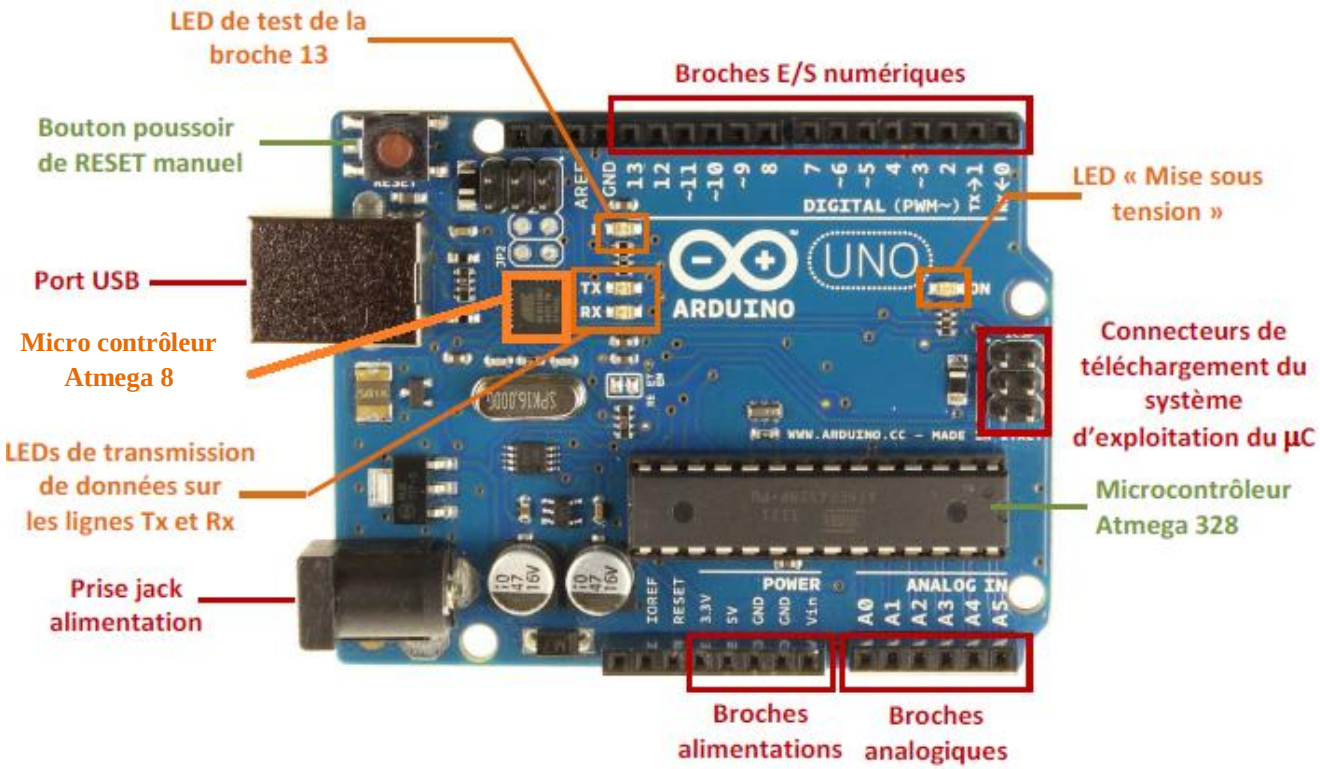


Figure 2.1.Description de la carte ARDUINO UNO.

II.1.1. Avantages de la carte ARDUINO UNO :

Les cartes ***ARDUINO*** sont peu couteuses relativement aux autres cartes de développement, les moins chères peuvent même être assemblées à la main et les pré-assemblées coutent moins de 25 €.On n'a pas besoin d'un programmateur extérieur. Grâce au ***bootloader*** on peut transférer le nouveau programme dans le microcontrôleur sans avoir à utiliser un matériel de programmation externe.

L'intérêt principal des cartes ARDUINO est leur facilité de mise en œuvre.

ARDUINO fournit un environnement de développement s'appuyant sur des outils open-source. Le chargement du programme dans la mémoire du microcontrôleur se fait de façon très simple par port USB.
En outre, des bibliothèques de fonctions "clé en main" sont également fournies pour l'exploitation d'entrées-sorties courantes : gestion des E/S, gestion des convertisseurs ADC, génération de signaux ***PWM***, exploitation de bus **I2C**, exploitation de servomoteurs, émission /réception série. Le logiciel ***Arduino*** écrit en ***Java*** est multi-plate forme, il tourne sous les systèmes d'exploitation Windows, Macintosh et Linux. Il existe aussi des ***Modules « shield »*** qui sont des cartes supplémentaires se connectant sur la carte ***ARDUINO*** pour augmenter les possibilités comme les afficheurs graphiques en couleur, interface ethernet, GPS et BLUETOOTH.

II.1.2.Alimentation de la carte ARDUINO:

La carte ***ARDUINO UNO*** peut être alimentée par l'USB ou par une alimentation externe. La source est sélectionnée automatiquement.
La tension d'alimentation extérieure (hors USB) peut venir soit d'un adaptateur ***AC-DC*** ou de piles. L'adaptateur peut être connecté grâce à une prise 'jack' de 2.1mm positif au centre. Le raccordement vers un bloc de piles peut utiliser les bornes ***Gnd*** et ***Vin*** du connecteur d'alimentation (POWER). La carte peut fonctionner à l'aide d'une tension extérieure de 7 à 12 volts. Les broches (pins) d'alimentation sont les suivantes :

- ***VIN (à*** distinguer du 5V de la connexion USB ou autre source 5V régulée). C'est la tension d'entrée positive lorsque la carte Arduino est utilisée avec une source de tension externe (à distinguer du 5V de la connexion USB ou autre source 5V régulée). Vous pouvez alimenter la carte à l'aide de cette broche, ou, si l'alimentation est fournie par le jack d'alimentation, accéder à la tension d'alimentation sur cette broche.

- **5V**. C'est la tension régulée utilisée pour faire fonctionner le microcontrôleur et les autres composants de la carte (la tension régulée est obtenue grâce à un régulateur intégré dans la carte Arduino). Le 5V régulé fourni par cette broche peut donc provenir soit de la tension d'alimentation VIN via le régulateur de la carte, ou bien de la connexion USB (qui fournit du 5V régulé) ou de tout autre source d'alimentation régulée.

- ***3V3***. Une alimentation de 3.3V fournie par le circuit intégré ***FTDI*** (circuit intégré faisant l'adaptation du signal entre le port USB de votre ordinateur et le port série de l'**Atmega**) de la carte est disponible : ceci est intéressant

pour certains circuits externes nécessitant cette tension au lieu du 5V). L'intensité maximale disponible sur cette broche est de 50mA.

Ø ***GND***. Broche de masse (ou 0V).

II.1.3.Protection du port USB contre la surcharge en intensité :

Un **fusible réarmable** ou « **Poly fuse** » est présent sur la connexion d'alimentation 5 V de la prise USB. Toute consommation **supérieure à 500 mA** provoque le **déclenchement de ce fusible**, protégeant ainsi le port USB de l'ordinateur auquel la carte est relié. Le fusible étant de type **réarmable**, il retrouvera son état normal quelques secondes après que la consommation excessive aura cessée. 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 500mA 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.

II.1.4. Gestion de la mémoire :

Le microcontrôleur ***ATmega 328*** dispose de **32 ko de mémoire de programme Flash** (dont 0.5Ko également utilisés par le bootloader). Il contient aussi **2 ko de mémoire vive** (SRAM). Cette mémoire est généralement utilisée pour stocker les résultats temporaires lors de calculs. Elle peut être lue et écrite à tout instant par le microcontrôleur mais son contenu est perdu dès que la n'est plus alimentée. ***L'ATmega 328*** dispose également **1ko mémoire EEPROM**. Le contenu de cette mémoire est accessible grâce aux fonctions de la librairie ***« EEPROM »***.

*II.*1.5.Les entrées/sorties numériques :

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

- **Communication série :** Broches 0 (RX) 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 ATmega8U2 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 à la transmission série a travers ces broches avec l'instruction **Serial.print()**, à condition que le câble USB soit déconnecté, sinon il va y avoir un chevauchement.

- **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. Voir l'instruction **attachInterrupt()** pour plus de détails.

- **Impulsion PWM (largeur d'impulsion modulée) :** Broches 3, 5, 6, 9, 10, et 11. Fournissent une impulsion PWM 8-bits à l'aide de l'instruction **analogWrite().**

- **SPI (Interface Série Périphérique) :** Broches 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Ces broches supportent la communication SPI (*Interface Série Périphérique*) disponible avec la librairie pour communication SPI. Les broches SPI sont également connectées sur le connecteur ICSP.

- **I2C :** Broches 4 (SDA) et 5 (SCL), Supportent les communications de protocole I2C, disponible en utilisant la librairie **Wire/I2C** ou **Two Wire**.

- **LED :** Broche 13. Il y a une LED incluse dans la carte connectée à la broche 13. Lorsque la broche est au niveau HAUT, la LED est allumée, lorsque la broche est au niveau BAS, la LED est éteinte.

-Autres broches :

Il y a deux autres broches disponibles sur la carte :

- **AREF :** Tension de référence pour les entrées analogiques (si différent du 5V), utilisée avec l'instruction **analogReference ().** Elle s'utilise pour réduire
- **Reset :** Mettre cette broche au niveau BAS entraîne la réinitialisation du microcontrôleur. Typiquement, cette broche est utilisée pour ajouter

un bouton de réinitialisation sur le circuit qui bloque celui présent sur la carte.

La figure suivante montre les différentes broches de la carte ***Arduin Uno*** :

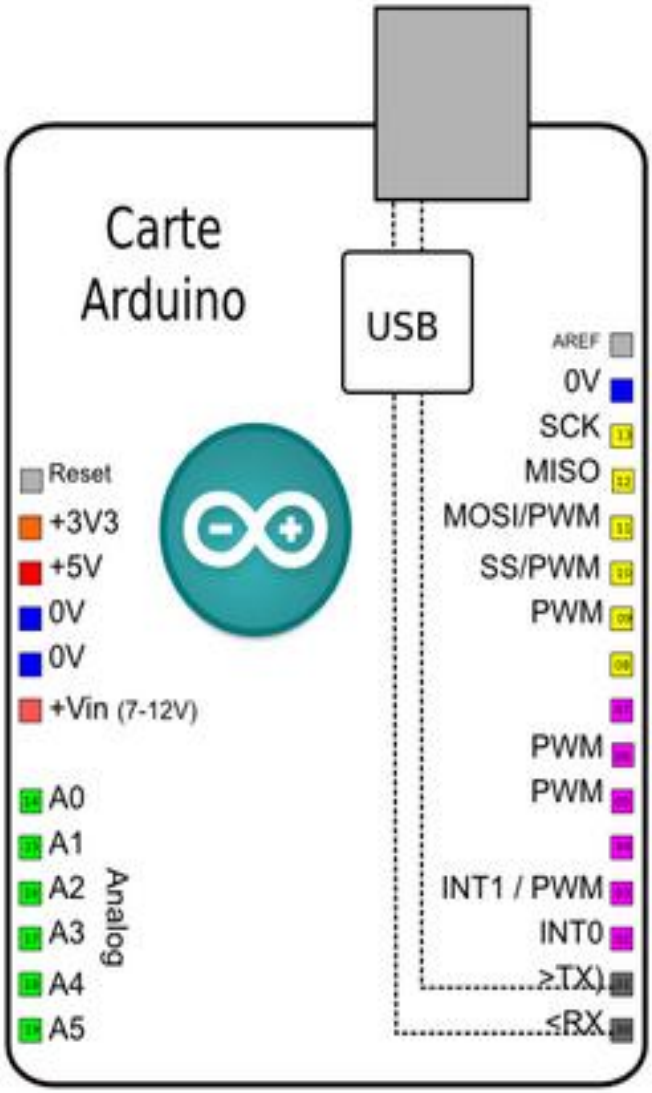


Figure 2.2 .Brochage de la carte Arduin Uno.

II.1.6. Communication avec l'extérieur :

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

L'ATmega 328 dispose d'une ***UART*** (Universal Asynchronous Receiver Transmitter ou émetteur-récepteur universel asynchrone en français) pour une communication série de niveau TTL (5V) et qui est disponible sur les broches 0 (RX) et 1 (TX).

Un circuit intégré ATmega8U2 sur la carte assure la connexion entre cette communication série et le port USB de l'ordinateur et apparaît 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 driver externe n'est nécessaire.

Le logiciel Arduino (dont l'on va parler dans le chapitre suivant) inclut une fenêtre terminal série (ou moniteur série) sur l'ordinateur et qui permet d'envoyer des textes simples depuis et vers la carte Arduino. Les LEDs RX et TX sur la carte clignotent lorsque les données sont transmises via le circuit intégré USBvers-série et la connexion USB vers l'ordinateur (mais pas pour les communications série sur les broches 0 et 1). Une librairie Série Logicielle permet également la communication série (limitée cependant) sur n'importe quelle broche numérique de la carte UNO.

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

- Ø Le logiciel Arduino inclut la librairie **Wire** qui simplifie l'utilisation du bus I2C.

- Ø 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.

-Dimensions de la carte :

Les longueurs et largeurs maximales de la Uno sont respectivement 6.86 cm et 5.33 cm, avec le connecteur USB et le connecteur d'alimentation Jack s'étendant au-delà des dimensions de la carte. Quatre trous de vis permettent à la carte d'être fixée sur une surface ou dans un boîtier (pour l'embarquer sur un système). Noter que la distance entre les broches 7 et 8 est de 0.16 pouces, et 0.1pouces séparant les autres broches.

II.1.7. Le microcontrôleur ATMega328 :

Le microcontrôleur utilisé sur la carte ARDUINO UNO est un microcontrôleur ATMega328. C'est un microcontrôleur ATMEL de la famille AVR 8bits.

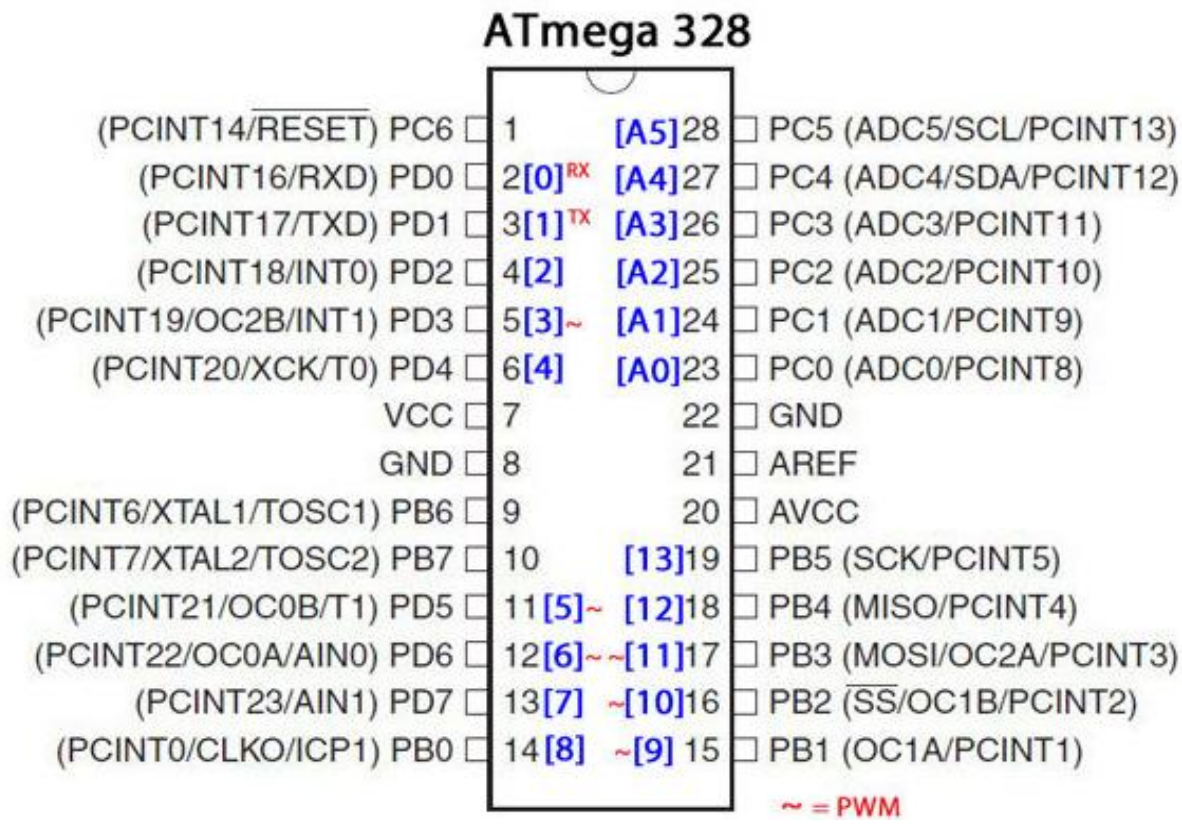


Figure 2.3. Le microcontrôleur ATMega328.

II.1.8.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), OC0B(PD5), 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/sorties 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 32KB dont 0.5KB 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 Timer2 sont à 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, l' 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).

Convertisseur Analogique/Numérique : l'ATMega328 possède un convertisseur Analogique/Numérique d'une résolution de 10bits, ce convertisseur peut être utilisé à travers 6 entrées multiplexées de ADC0(PC0) jusqu'à ADC5(PC5).

Gestion bus I2C : ce bus est exploité via les deux broches SDA(PC5)/SCL(PC4).

Port série (USART) : émission/réception série via les broches TXD(PD1)/RXD(PD0).

Comparateur Analogique : le comparateur analogique intégré dans l'ATMega peut être utilisé à travers les deux broches AIN0(PD6) et AIN1 (PD7), ce comparateur peut déclencher une interruption.

Watchdog Timer programmable : 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 Timers 0, 1 et 2 (plusieurs causes configurables).
– Interruption liée au comparateur analogique.
– Interruption de fin de conversion **ADC.**
– Interruptions du port série **USAR.**
– Interruption du bus **I2C.**

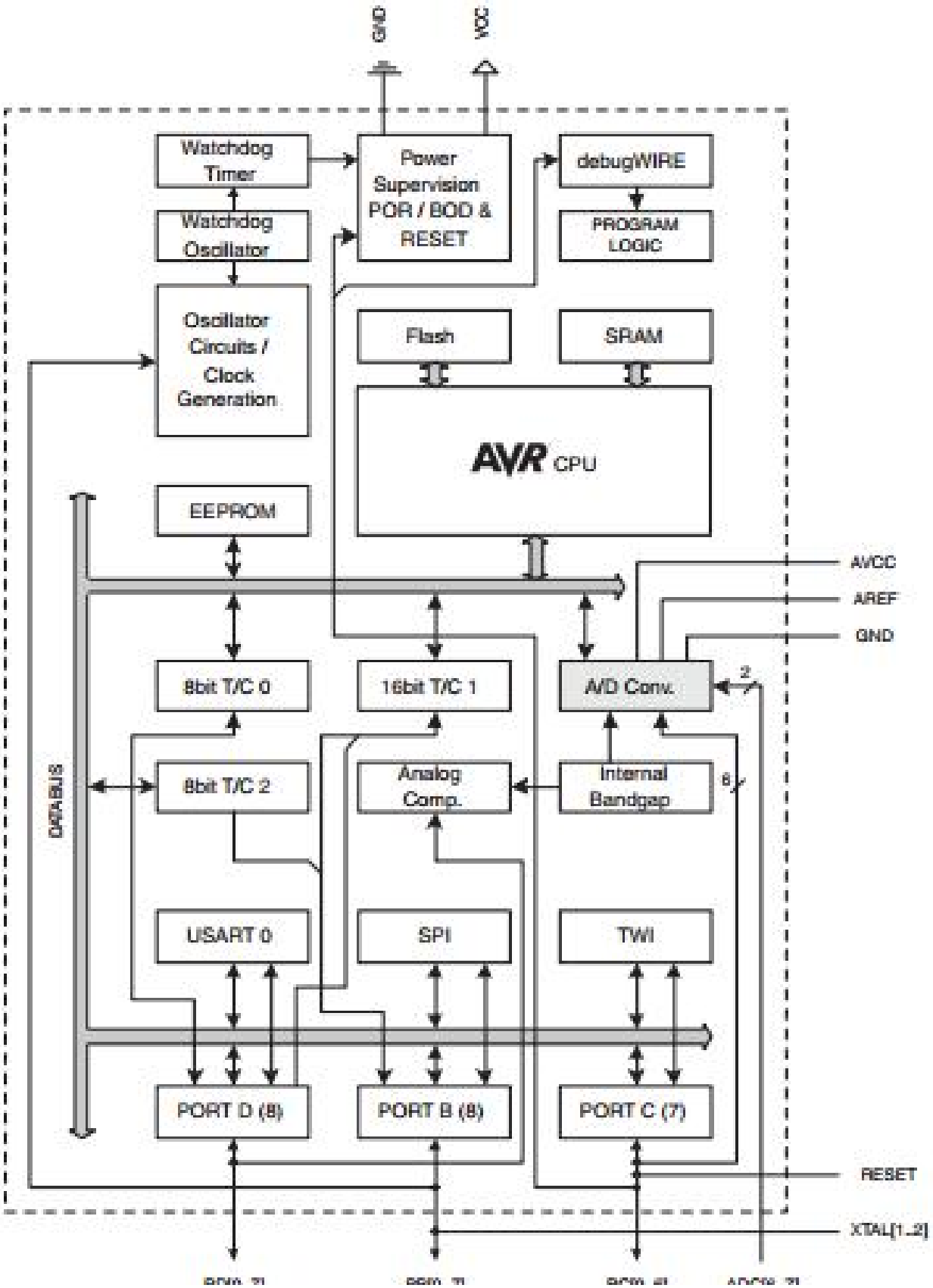


Figure 2.4.Architecture interne de l'ATMega328.

II.2. Description des capteurs utilisés :

Les capteurs sont les premiers éléments d'une chaine d'acquisitions. Ils transforment une grandeur physique d'un processus ou d'un phénomène en signaux électriques exploitables. Dans notre projet, on a utilisé quatre capteurs :

- Capteur de température DS18B20.
- Capteur de vibration BMA180.
- Capteur de vitesse de type effet hall UGN 3503.
- Capteur de son (microphone).

II.2.1. Le capteur de température DS18B20 :

Le DS18B20 est capteur de température numérique du fabricant DALLAS. Il communique via le bus 1-Wire du même fabricant. Sa plage de fonctionnement se situe entre -55°C et +125°C, il possède une résolution de 9, 10, 11 ou 12 bits, ce qui correspond à une précision de 0.5°C, 0.25°C, 0.125°C et 0.0625°C respectivement. La conversion analogique -numérique prend un temps de conversion de 0.75 secondes.

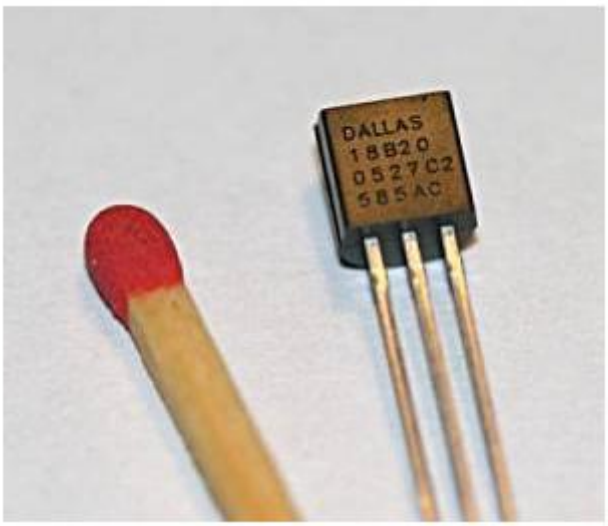


Figure. Le DS18B20.

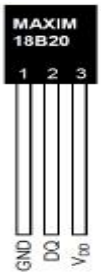
Le DS18B20 peut fonctionner sans aucune alimentation externe. L'alimentation est fournie par la résistance de tirage par la broche DQ quand le bus est au niveau haut. Ce niveau du bus charge également une capacité interne Cpp qui alimente ensuite le capteur quand le bus est en niveau bas. Cette méthode d'alimentation est appelée « parasite power ». Le capteur peut aussi être alimenté da façon standard par la broche d'alimentation Vcc avec une

tension variant entre 3 à 5.5 volts. À sa mise sous tension, il démarre par un état de veille en basse consommation et la valeur du registre de température est de +85°C.

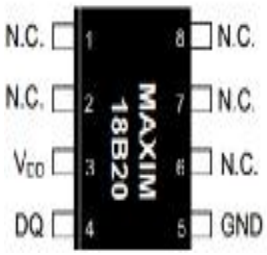
Remarque :

En mode parasite, les valeurs de la température peuvent être erronées d'où la préférence d'utiliser le mode d'alimentation standard qu'on a utilisé dans notre projet.

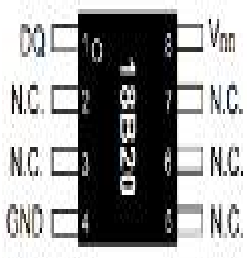
II.2.1.1. Brochage du DS18B20 :



-DS18B20.



-DS18B20Z.



-DS18B20U.

N .C : non connecté.

Figure 2.5. Brochage des différents types du DS18B20.

II.2.1.2. Schéma bloc du DS18B20 :

La figure suivante montre l'architecture interne du DS18B20 :

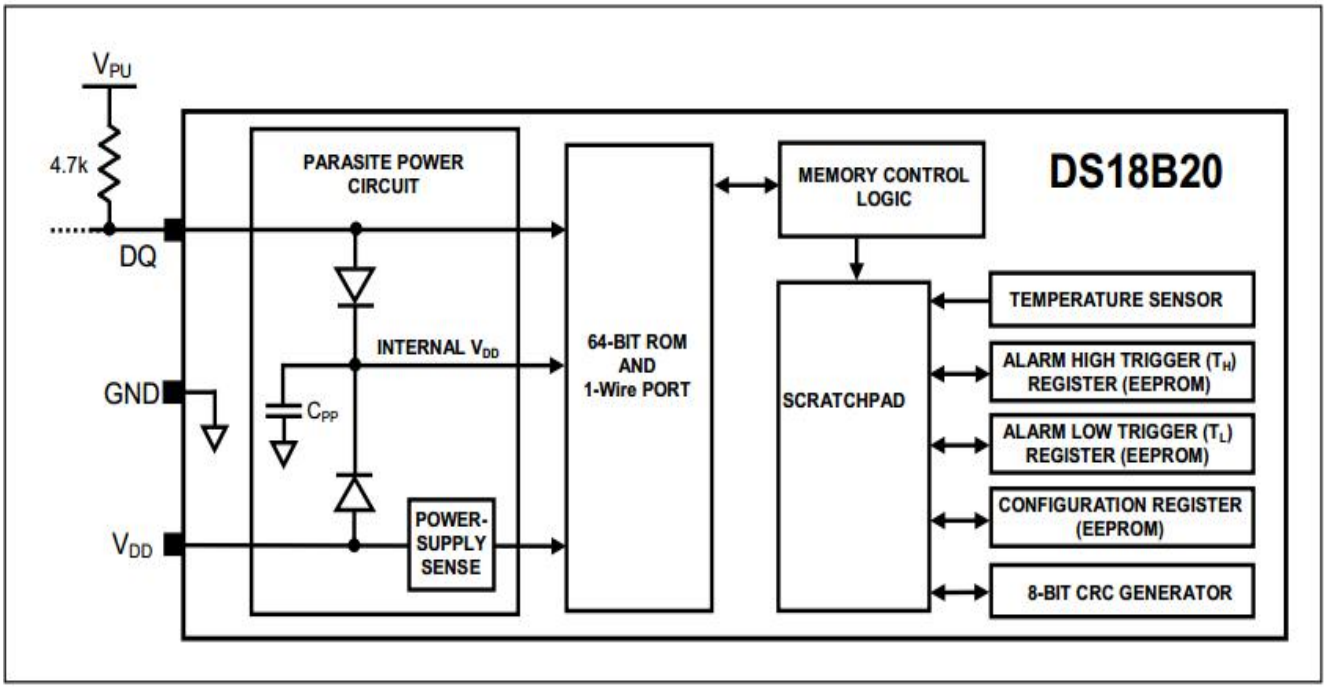


Figure 2.6. Architecture interne du DS18B20.

II.2.1.3.Mesure de la température par le DS18B20 :

Les résultats de mesure de la température sont stockés dans le registre de température qui est sur deux octets. La figure suivante montre le contenu du registre de température :

	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
LS BYTE	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
	BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
MS BYTE	S	S	S	S	S	2^6	2^5	2^4

S = SIGN

Figure 2.7. Registre de température du DS18B20.

Le tableau suivant montre la relation entre les valeurs mesurées et la température :

TEMPERATURE (°C)	DIGITAL OUTPUT (BINARY)	DIGITAL OUTPUT (HEX)
+125	0000 0111 1101 0000	07D0h
+85*	0000 0101 0101 0000	0550h
+25.0625	0000 0001 1001 0001	0191h
+10.125	0000 0000 1010 0010	00A2h
+0.5	0000 0000 0000 1000	0008h
0	0000 0000 0000 0000	0000h
-0.5	1111 1111 1111 1000	FFF8h
-10.125	1111 1111 0101 1110	FF5Eh
-25.0625	1111 1110 0110 1111	FE6Fh
-55	1111 1100 1001 0000	FC90h

Figure 2.8. Relation entre les mesures et la température.

II.2.1.4. Structure de la mémoire du DS18B20 :

La mémoire est constituée de ram statique SRAM pour la partie bloc notes (Scratchpad) et de mémoire non volatile EEPROM pour les registres de seuil d'alarme TH et TL. Si la fonctionnalité d'alarme n'est pas utilisée, ces registres peuvent servir de mémoire à usage général.

SCRATCHPAD

Byte 0	Temperature LSB (50h)	(85°C)
Byte 1	Temperature MSB (05h)	
Byte 2	T_H Register or User Byte 1*	
Byte 3	T_L Register or User Byte 2*	
Byte 4	Configuration Register*	
Byte 5	Reserved (FFh)	
Byte 6	Reserved	
Byte 7	Reserved (10h)	
Byte 8	CRC*	

Figure 2.9.Structure de la mémoire du DS18B20.

- Les octets 0 et 1 du bloc-notes représentent respectivement les poids faible et fort du registre de température.
- Les octets 2 et 3 de la mémoire bloc-notes constituent les registres d'alarme TH et TL.
- L'octet 4 contient les données du registre de configuration.
- Les octets 5, 6 et 7 sont réservés pour l'usage interne par le dispositif.
- L'octet 8 est en lecture seule et contient le code de redondance cyclique (CRC) calculé en fonction du contenu des octets 0 à 7 de la mémoire bloc-notes.

Remarque :

L'avantage du capteur DS18B20 est que chacun contient un code unique sur 64 bits (voir Figure suivante) stocké dans la ROM. Les moins importants 8 bits du code de la ROM contiennent le code 1-Wire de la famille DS18B20 ***(28h).*** Les prochaines 48 bits contiennent un unique numéro de série propre à chaque DS18B20. Les plus significatifs 8 bits contiennent un contrôle de redondance cyclique (CRC) qui est un octet calculé à partir des 56 premiers bits du code de la ROM. Le code de la ROM 64 bits et la logique de commande de fonction de la ROM associée permettent au DS18B20 de fonctionner comme un dispositif 1-Wire en utilisant le protocole 1-wire.

Ce code unique qui caractérise les composants DS18B20 permet d'utiliser plusieurs capteurs de ce genre en même temps sans se soucier de se tromper lors de l'adressage, il faut juste spécifier dans le programme quel composant veut-on adresser.

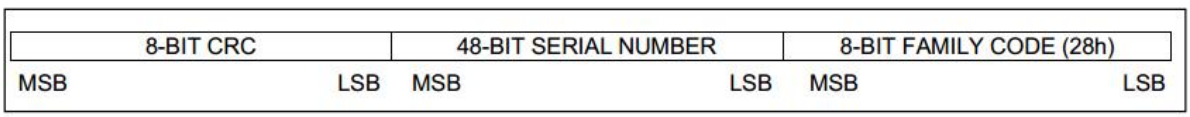


Figure 2.10.Le registre d'adresse du DS18B20.

II.2.1.5. Le bus One-Wire:

Le bus One-Wire conçu par Dallas Semi-conducteur permet de connecter et de faire dialoguer des composants avec seulement un fil. L'état de repos du bus est un état haut ce qui permet d'alimenter les différents composants à partir de la

ligne data en mode parasite. Le fil unique du bus doit être tiré vers +Vcc par une résistance de 4.7KΩ.

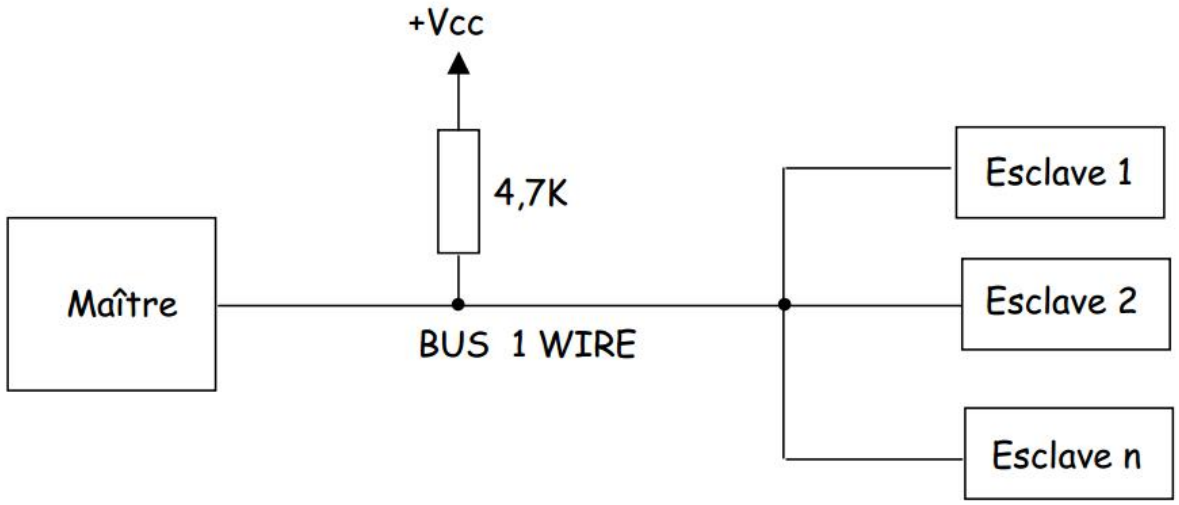


Figure 2.11. Le bus One-Wire.

Si le bus est maintenu à l'état bas plus de 480 µs par le maître, tous les composants sur le bus sont remis à zéro. C'est le pulse d'initialisation ou de Reset. Après un délai de 15 à 60 µs, le ou les esclaves raccordés forcent le bus à l'état bas pendant 60 à 240 µs pour signaler leur présence.

U bus

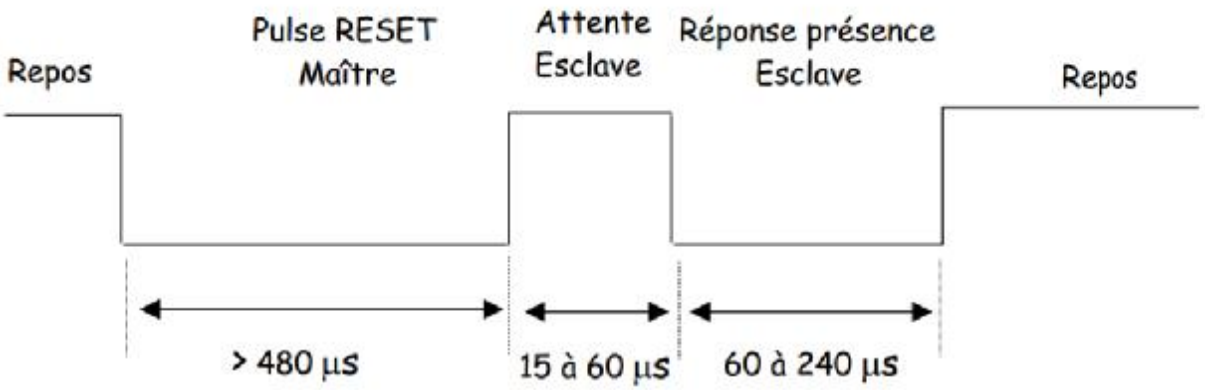


Figure 2.12. Chronogramme du bus 1-Wire.

Toute transaction entre un maître et un ou plusieurs esclaves, débute par une initialisation, constituée par l'envoi du pulse de Reset par le maître. Le maître doit ensuite envoyer une commande de type ROM qui est propre au protocole 1 Wire, et que tous les circuits de ce type vont reconnaître. Cela va permettre entre autre de sélectionner un circuit parmi les différents esclaves qui ont répondu

présents au pulse de Reset. Le dialogue et l'échange des données pourront ensuite commencer entre le maître et l'esclave sélectionné.

II.2.1.6.Emission d'un bit du maître vers l'esclave:

Le maître force le bus à "0" pendant au moins 15 µs. L'esclave va lire le bus entre 15 et 45 µs après le front descendant (valeur typique 30 µs). Si on veut émettre un "1", il faut repasser le bus à "1" immédiatement, et ne plus rien faire jusqu'à t = 60 µs. Pour émettre un "0" il faut laisser le bus à "0" jusqu'à t = 60 µs, puis repasser le bus à "1". La durée du bit est donc de 60 µs, ce qui donne un débit de 16 kbits/sec.

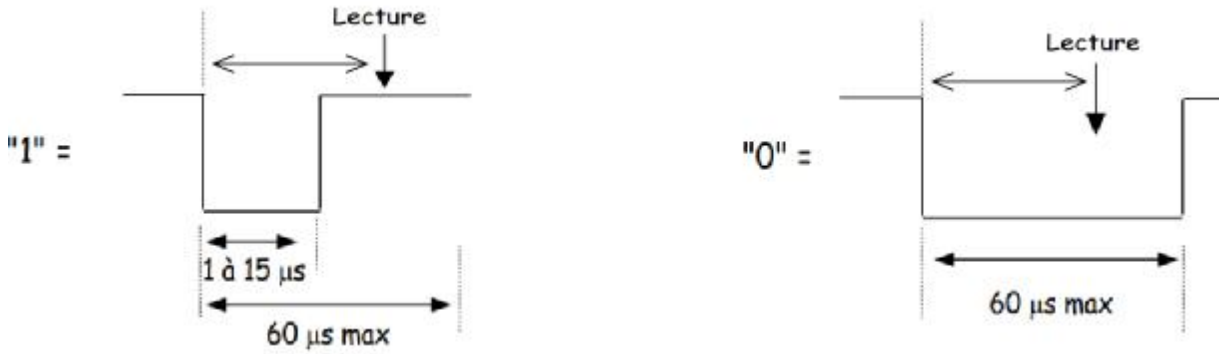


Figure 2.13. Emission d'un bit par le mettre.

II.2.1.7.Réception d'un bit par le maître:

Le maître force le bus à "0" pendant au moins 1 µs. Si l'esclave veut émettre un "1", il laisse le bus libre donc tiré à "1". Pour émettre un "0", l'esclave doit tirer le bus à "0" pendant 15 µs au minimum. Le maître devra donc dans tous les cas lire le bus 15 µs maximum après avoir tiré le bus à "0" pendant 1 µs. L'état du bus donnera alors le bit transmis par l'esclave.

Figure 2.14.Réception d'un bit par le maitre.

II.2.1.8.Organigramme du fonctionnement du DS18B20 :

La figure suivante montre l'organigramme du DS18B20.

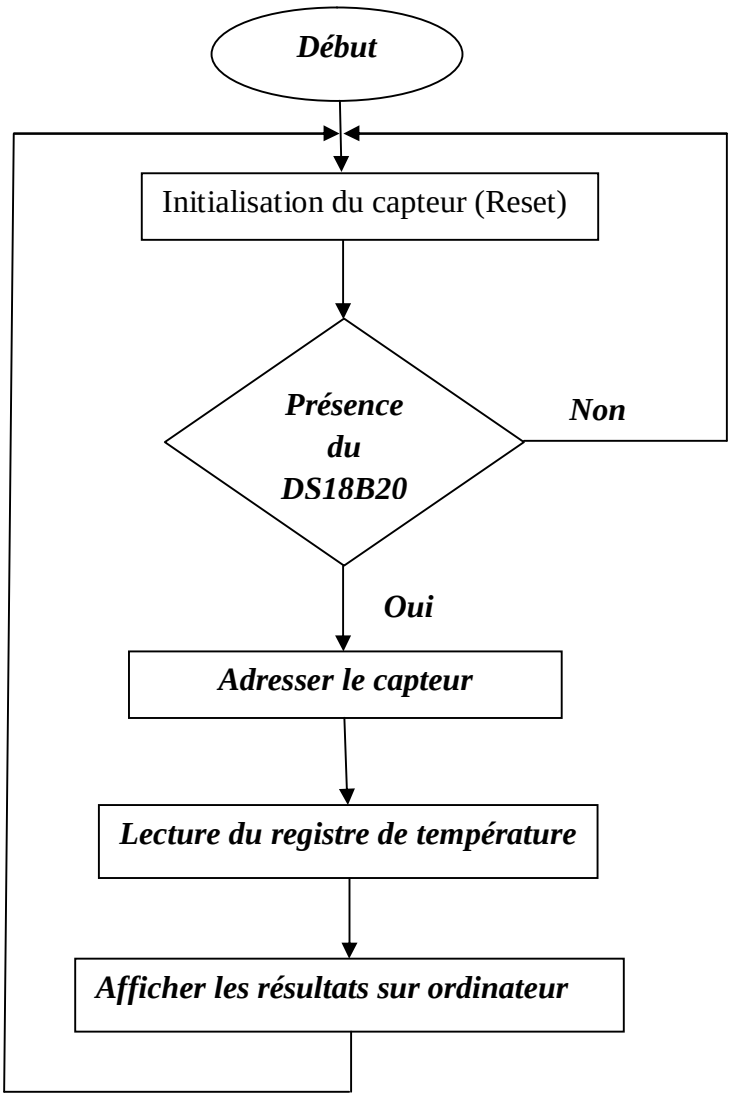


Figure 2.15.Organigramme du fonctionnement du DS18B20.

II.2.1.9. Schémas du branchement du DS18B20 :

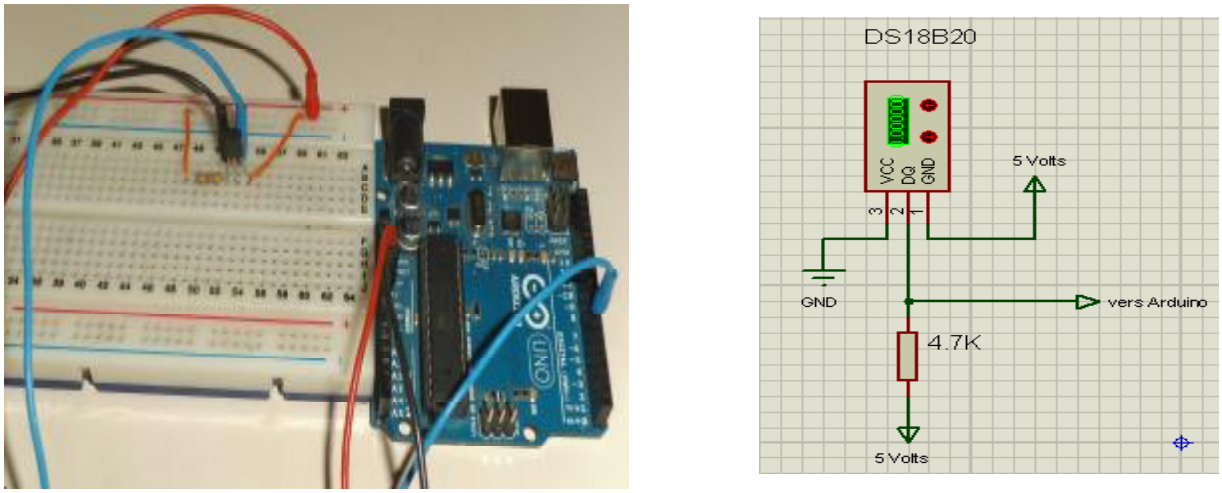


Figure 2.16.Mise en œuvre et circuit du DS18B20.

II.2.2. Le capteur d'accélération BMA180 :

L'accéléromètre BMA180 de Bosch est un capteur numérique triaxial d'ultra haute performance, destiné à des applications à faible consommation de puissance. Le BMA180 permet la mesure précise des accélérations dans les trois axes perpendiculaires (X, Y et Z) et par conséquent détecte le sens d'inclinaison, le mouvement, les chocs et les vibrations. Il permet aussi la mesure de la température dans la plage -45 et +85°C. Il fournir une sortie 14 bits numérique soit par une interface SPI ou I2C (qui est utilisée dans notre projet).

Le BMA180 peut être alimenté avec une tension de 1.62 jusqu'à 3.6 volts avec une consommation de 650µA en mode normal.

La figure suivante montre le BMA180.

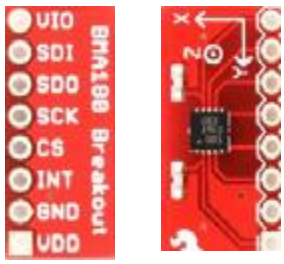


Figure 2.17. Le BMA180.

II.2.2.1. Bloc diagramme du BMA180 :

Le bloc diagramme du BMA180 est composé de :

- Deux capteurs, le premier pour la température et le second pour l'accélération.
- Deux préamplificateurs du signal d'entrée (analogique).
- Un multiplexeur.
- Un convertisseur analogique/numérique 14 bits.
- Un bloc numérique responsable du filtrage numérique et calibrage.
- Deux interfaces de sorties (SPI et I2C).
- Une sortie d'interruption.

La figure suivante illustre le bloc diagramme de l'accéléromètre BMA180 :

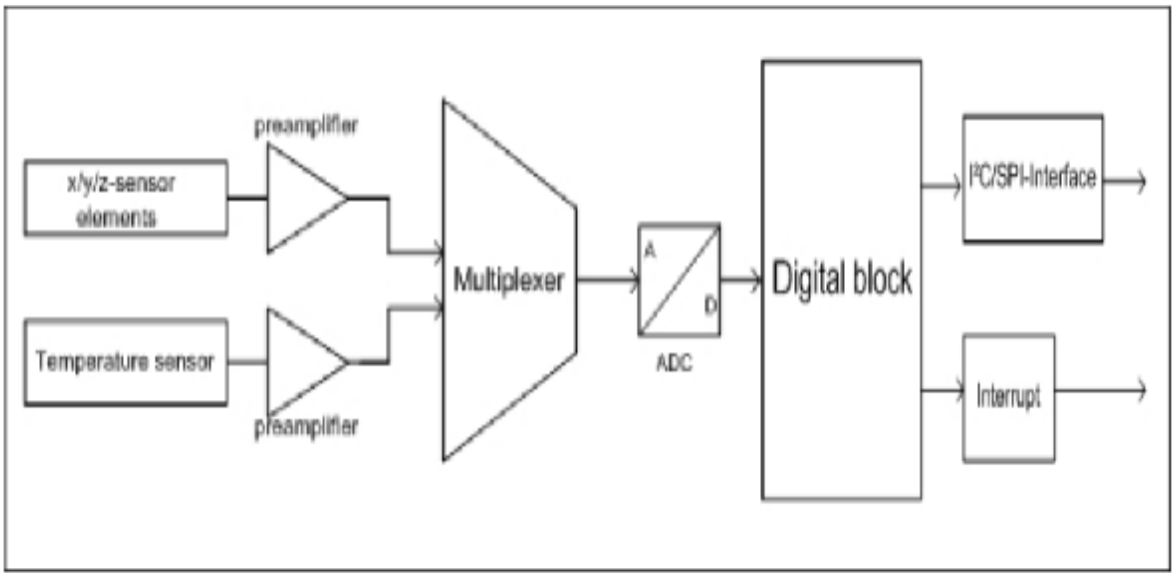


Figure 2.18.Bloc diagramme deBMA180.

II.2.2.3. Le bus I2C :

Le bus I2C (***Inter Integrated Circuit***) fait partie des bus avec 2 fils de données et un fil de masse. Il est apparu au début des années 80, développé par la société ***Philips*** pour minimiser les liaisons entre les circuits intégrés numériques. Il utilise :

- Un fil de donnée (SDA).
- Un fil d'horloge (SCL).
- Un fil de référence électrique (la masse).

Les données sont transmises en série à 100Kbits/s en mode standard et peut aller jusqu'à 400Kbits/s en mode rapide. Ce qui ouvre la porte de cette technologie à toutes les applications dont la vitesse de transmission n'est pas primordiale.

II.2.2.4.Le protocole I2C :

Le protocole I2C définit la succession des états logiques possibles sur SDA et SCL, et la façon dont doivent réagir les circuits en cas de conflits.

Avant d'utiliser le bus I2C, le circuit voulant communiquer doit vérifier que les lignes SDA et SCL sont au repos, c'est-à-dire à l'état haut. Pour transmettre des données sur le bus, il faut surveiller deux conditions particulières : la condition de départ et la condition d'arrêt.

- La condition de départ : SDA passe à '0' alors que SCL reste à 1.
- La condition d'arrêt : SDA passe à '1' alors que SCL reste à 1.

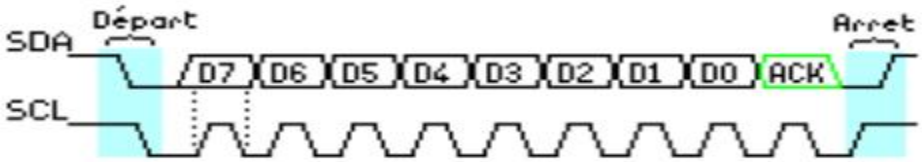


Figure 2.19. Condition de départ et d'arrêt du bus I2C.

II.2.2.5.Transmission d'un octet :

Après avoir imposé la condition de départ, le maître applique sur SDA le bit de poids fort D7. Il valide ensuite la donnée en appliquant pendant un instant un niveau '1'sur la ligne SCL. Lorsque SCL revient à '0', il recommence l'opération jusqu'à ce que l'octet complet soit transmis. Il envoie alors un bit ACK à '1' tout en scrutant l'état réel de SDA. L'esclave doit alors imposer un niveau '0' pour signaler au maître que la transmission s'est effectuée correctement.

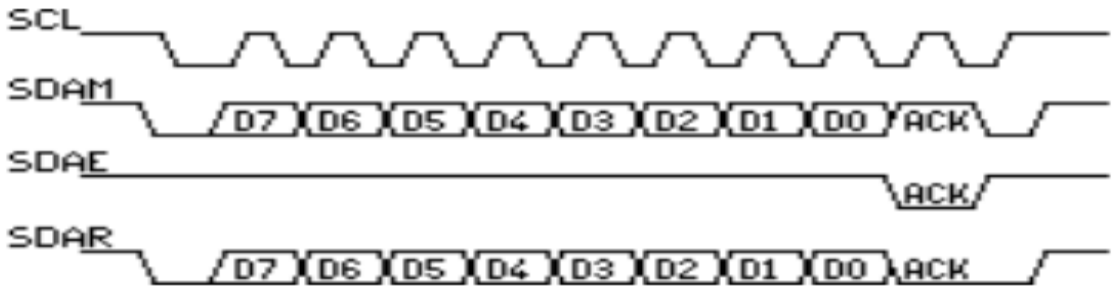


Figure 2.20.Transmission d'un octet.

- SCL : Horloge imposée par le maître.
- SDAM : Niveaux de SDA imposés par le maître.
- SDAE : Niveaux de SDA imposés par l'esclave.
- SDAR : Niveaux de SDA réels résultants.

II.2.2.6.Transmission d'une adresse :

Le nombre de composants qu'il est possible de connecter sur un bus I2C étant largement supérieur à deux, il est nécessaire de définir pour chacun une adresse unique. L'adresse d'un circuit, codée sur sept bits, est définie d'une part par son type et d'autre part par l'état appliqué à un certain nombre de ces broches. Cette adresse est transmise sous la forme d'un octet au format particulier.

Figure 2.21.Transmission d'une adresse.

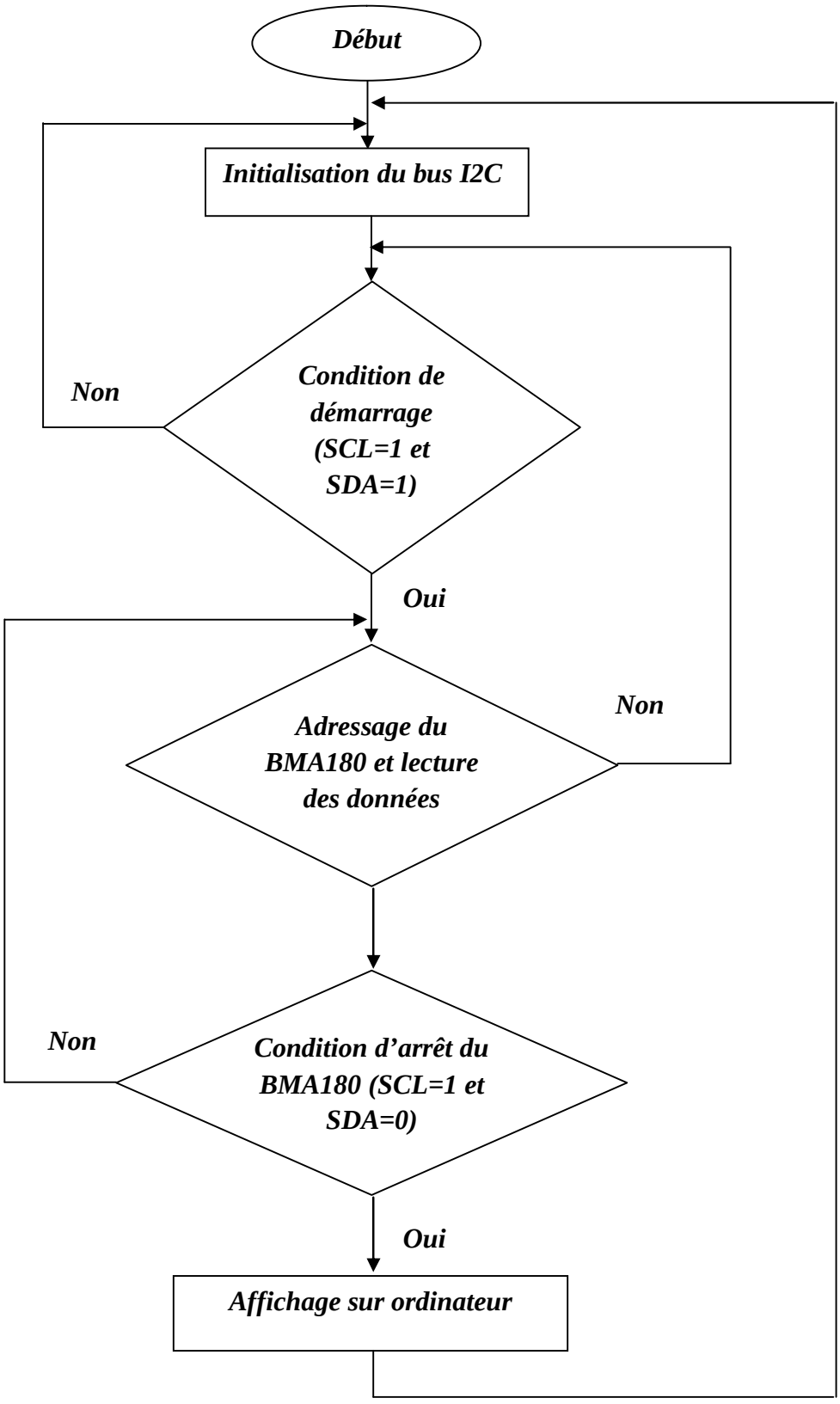


Figure 2.22.Organigramme du fonctionnement du BMA180.

II.2.2.7.Brochage du BMA180 :

La figure suivante montre comment connecter le BMA180 avec la carte Arduino Uno.

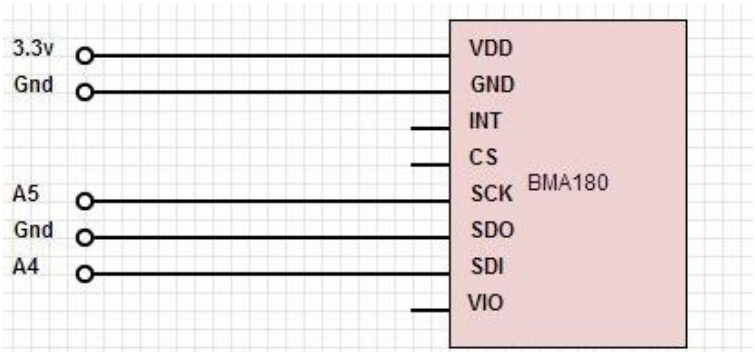


Figure 2.23.a. Circuit du BMA180.

La figure suivante montre la connexion avec l'Arduino.

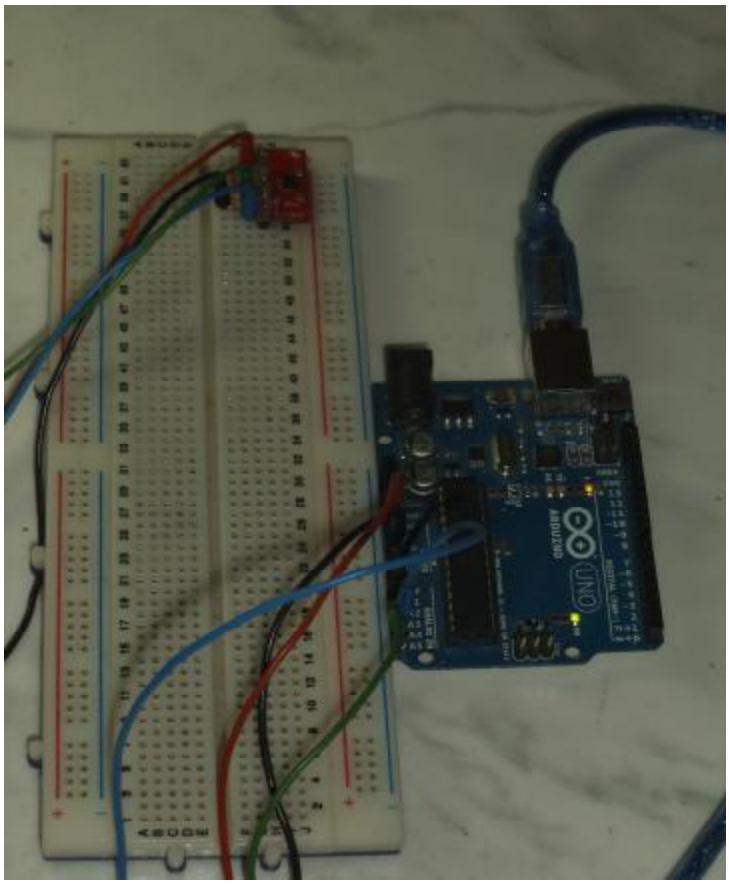


Figure 2.23.b. Arduino et le BMA180.

II.2.3.Mesure de la vitesse de rotation du moteur :

II.2.3.1. L'effet Hall :

L'effet hall est l'apparition d'une différence de potentiel et d'un champ électrique dans un conducteur lorsque ce dernier est parcouru par un courant électrique et plongé dans un champ magnétique. Les capteurs effet Hall sont les mieux adaptés pour la meure de la vitesse pour leur facilité de mise en œuvre et leur temps de réponse.

Pour la mesure de la vitesse de rotation du moteur, nous avons utilisé un capteur ***UGN3503.***

II.2.3.2.Le capteur effet hall UGN3503 :

C'est un capteur linéaire très sensible. Il fonctionne dans une température comprise entre -20 à 85°C, sa tension d'alimentation est comprise entre 4.5 jusqu'à 6 volts avec une réponse de 23KHz. Il est utilisé pour de nombreuses applications comme détecteur de dents d'engrenage, la détection de niveau, le positionnement, la détection de mouvement, bouton sans contact, mesure de la vitesse de rotation et bien d'autre d'application.

Les figures suivantes montrent le capteur UGN3503 et son bloc diagramme:

Figure 2.24.a.UGN3503.

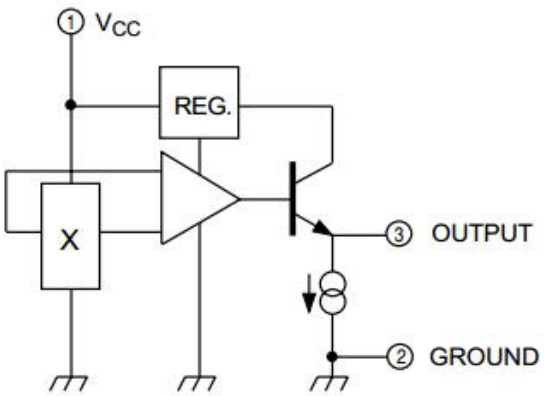


Figure 2.24.b.Diagramme de l'UGN3503.

II.2 .3.3.Quelques applications :

Les applications sont très nombreuses, nous allons voir deux applications utilisant ce capteur effet Hall.

- **Détecteur de proximité :**
  Dans la figure 3A, le pôle magnétique SUD est proche du capteur à effet Hall, il est donc activé (ON). Dans la figure 3B, le pole magnétique SUD s'est déplacé trop loin. Le capteur est désactivé (OFF).

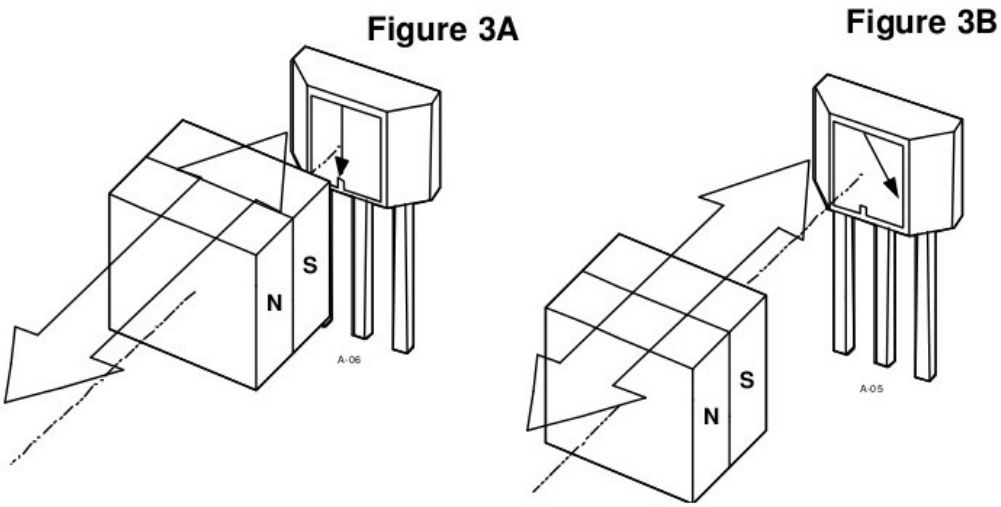


Figure 2.25.Détecteur de proximité.

- Une autre application élémentaire est la détection de niveau. Ce type d'assemblage peut être utilisé pour une détection de niveau dans un réservoir d'essence.

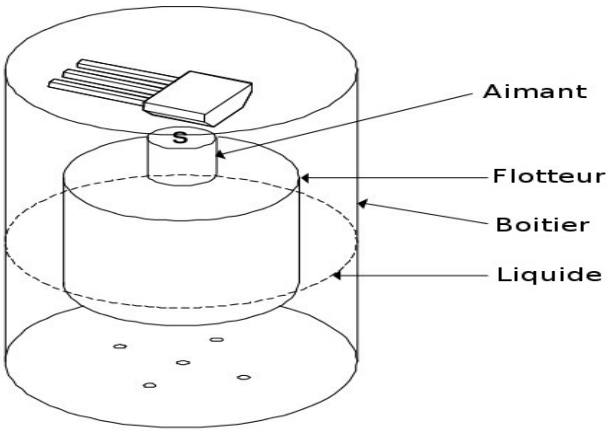


Figure 2 .26.Détection de niveau.

II.2.3.4.Principe de la mesure de la vitesse :

Sans la présence d'un champ magnétique, le capteur fournit une tension égale à la moitié de la tension d'alimentation (dans notre cas, il fournit environ 2.5 volts). Lorsque le pôle sud est détecté, une tension de 2 volts est mesurée sur la broche de sortie du capteur tandis qu'on mesure 3 volts en la présence du pôle nord.

Pour la mesure de vitesse, on va placer un aimant sud suffisamment puissant sur la partie tournante du moteur qui est le rotor pour qu'il soit détecté par notre capteur qui sera fixé juste à coté de la tige de l'aimant de sorte que chaque fois qu'il passe devant le capteur, une variation de la tension sera détecée. Chaque passage à zéro signifiera que le rotor a fait un tour et ainsi de suite. Le nombre de changement d'état va donc nous donner la vitesse du moteur.

La figure suivante montre comment placer le capteur effet Hall sur le moteur.

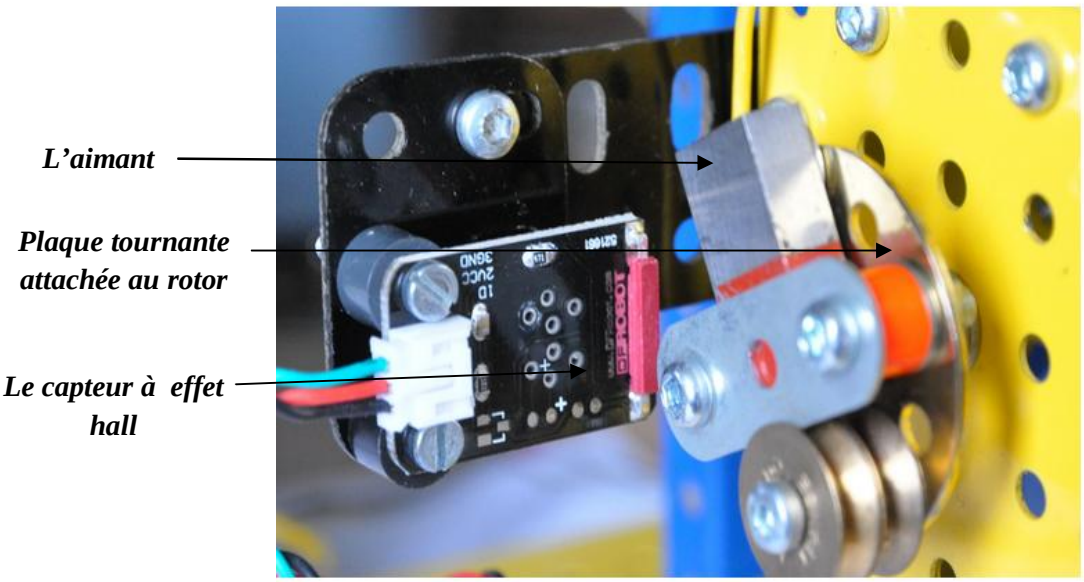


Figure 2.27.Fixation du capteur à effet Hall.

II.2.3.5.Schéma électrique :

Le schéma électrique de l'UGN3503 est représenté dans la figure suivante.

.

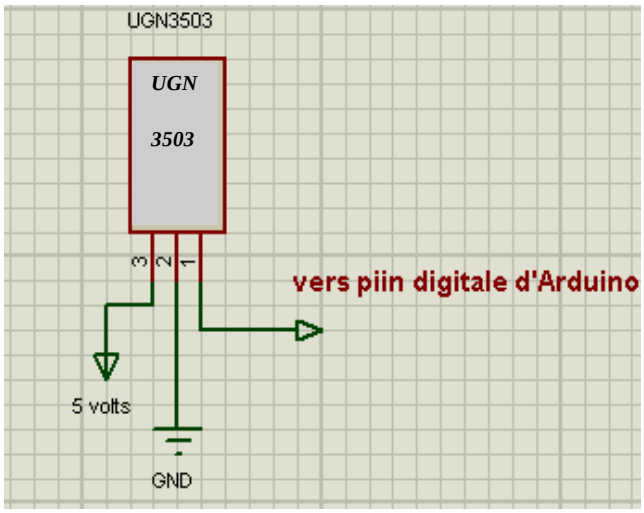


Figure 2.28.Schéma électrique du capteur effet Hall.

II.2.4.Détecteur de bruit :

Pour cette partie, nous avons utilisé pour la détection du bruit un microphone analogique. Pour faire l'acquisition de ce signal, nous avons ajouté un circuit amplificateur du type TDA2822 à 16 broches pour permettre si on veut utiliser un autre microphone sur un deuxième moteur.

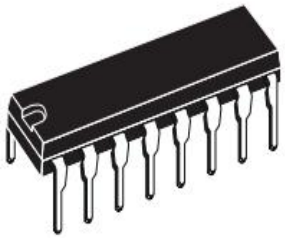


Figure 2.28.TDA2822.

II.2.4.1. Le TDA2822 :

Le TDA2822 est un circuit intégré monolithique utilisé comme amplificateur audio. Il est configurable en mode stéréo ou en mode pont.

Ses principales applications sont amplificateur casque, système audio portable, mini radio et les appareils auditifs. Il peut fournir une puissance de sortie de 0.65 Watts par canal dans un haut parleur 4 Ohms avec une tension d'alimentation 6 Volts en mode stéréo et 1.35 Watts dans les mêmes conditions en mode pont.

La figure suivante montre le brochage du TDA2822.

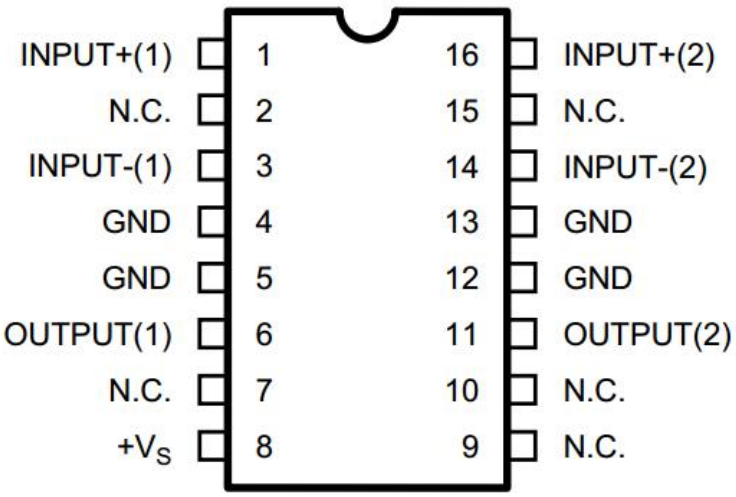


Figure 2.29.Brochage du TDA2822.

La figure suivante montre les deux types de montages avec le TD2822 :

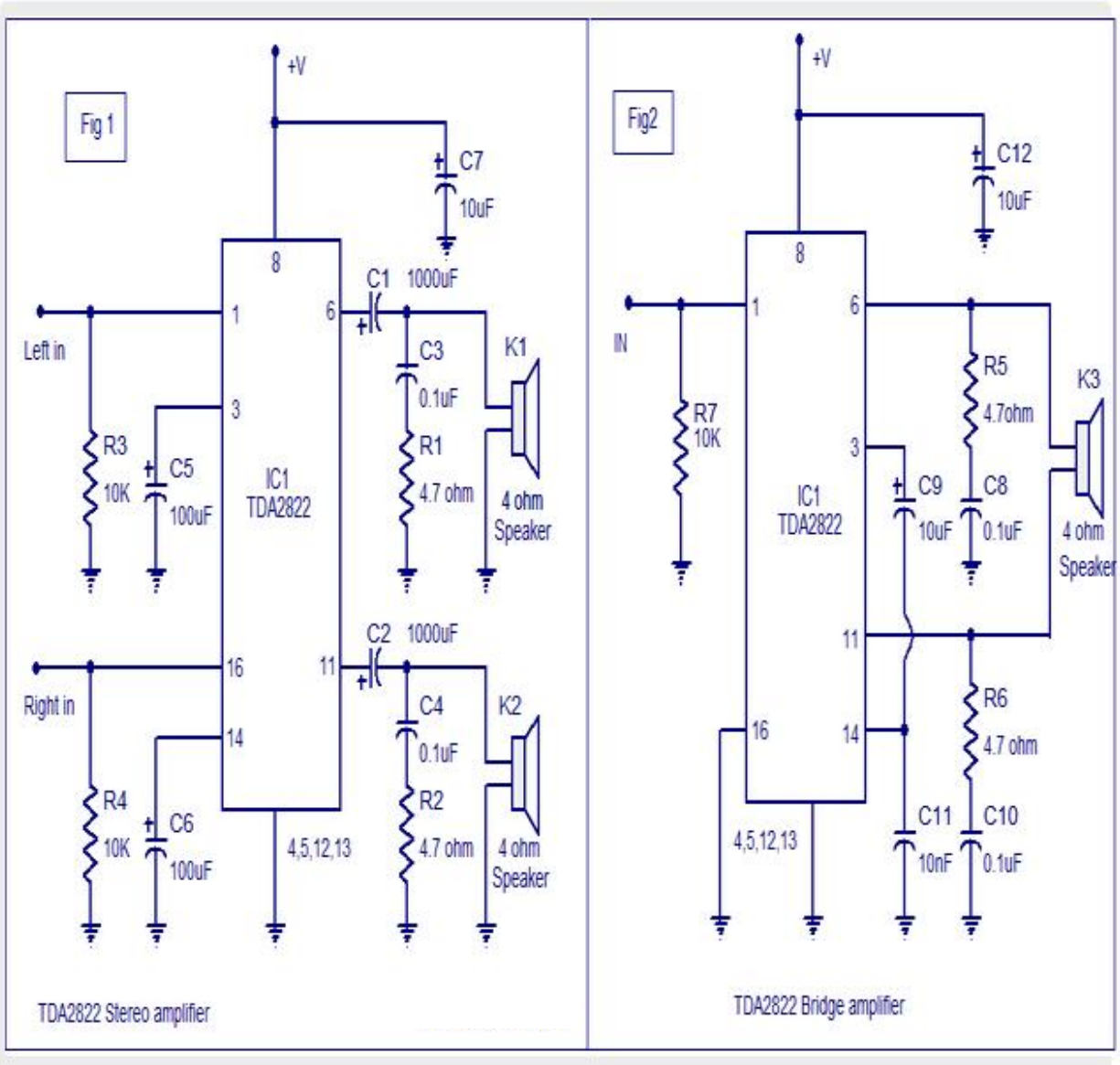


Figure 2.30.Montage avec haut parleur.

Remarque :

- Ø Dans notre projet, l'entrée ***Left in*** sera la broche + de notre microphone et la sortie ***K1*** sera reliée à la carte Arduino.
- Ø Les valeurs des composants indiquées dans ces schémas sont des valeurs exigées par le constructeur.

La figure suivante montre le circuit du TDA2822 avec le microphone.

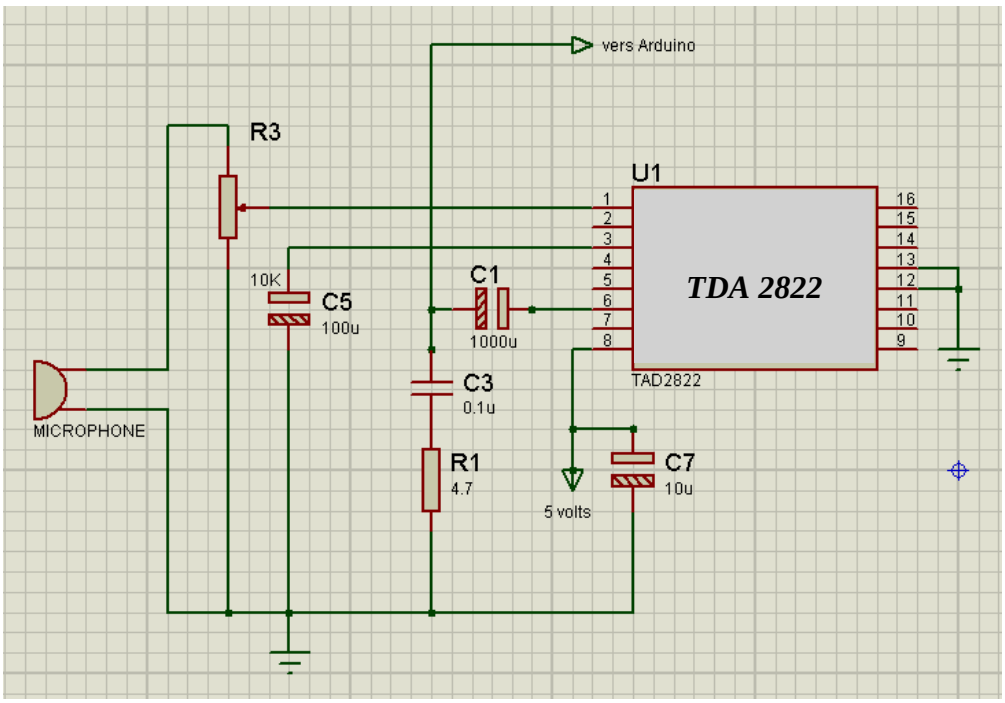


Figure 2.31.Montage du microphone avec le TDA2822.

II.3. Le module de transmission :

Il existe plusieurs technologies de transmission sans fil dans le domaine des télécommunications, et chaque une de ces technologies diffère de l'autre en fonction de ses performances, ses avantages, sa gamme de fréquence de travaille…

Dans notre projet on a utilisé la technologie Bluetooth vu ses avantages et ses performances et sa simplicité de mise en œuvre.

II.3.1.Historique :

Le mot bluetooth fait références a un roi Viking du 10 éme siècle Harald Blatand. Son nom, Blatand est devenu bluetooth dans un anglais récent. La traduction littérale de ce mot donne Harald à la dent bleue, ce surnom lui viendrait du fait qu'il appréciait énormément les bleuets (plus connu sous le nom de myrtille qui est un fruit d'une couleur bleu et d'un goût sucré) et qu'elles lui coloraient les dents en bleu. Il unifia la Norvège et le Danemark en préférant la consultation et la coopération plutôt que d'utiliser la puissance des armes.

L'instigateur de la norme bluetooth, Ericsson, a trouvé que ce nom serait parfait pour une technologie qui à pour but d'unifier les connections entre les ordinateurs et les appareils de télécommunication. Ericsson était un géant des télécoms norvégiens c'est pourquoi leur choix c'est porté sur un roi Viking plutôt qu'un autre.

Figure 2.32.Symbole du bluetooth.

II.3.2 Le module bluetooth de Sparkfun :

Les résultats des mesures des capteurs sont transmis par la carte Arduino vers l'ordinateur via Bluetooth, et pour cela on a choisis le module ***" BlueSMiRF"*** du groupe ***Sparkfun***, qui est compatible avec la carte Arduino et sans utiliser aucun circuit externe. Ce module dispose de six broches de connexion : ***Vcc***, ***Gnd***, ***Rx***, ***Tx***, ***CTS***, ***RTS***, et deux Leds (verte et rouge), une antenne miniaturisée intégré (la description de ces broches se fera tout à l'heure ainsi que le rôle de ces deux Leds).

II.3.3. Les caractéristiques du module BlueSMiRF :

Le module ***BlueSMiRF*** possède plusieurs caractéristiques techniques, ces caractéristiques sont présentées dans le tableau suivant :

caractéristiques	Propriétés
Classe	1
Consommation	25mA
Gamme de fréquence	De 2,4 à 2,524 GHz
Rapidité de modulation	De 2400 à 115200 bps
Tension d'alimentation	Entre 3,3 et 6V
Plage de température de fonctionnement	Entre -40 et +70°C
Porté	Environ 100m
Protocole de communication	UART
Dimensions	0.15x0.6x1.9 inch

Figure 2.33. Caractéristiques du module BlueSMiRF.

II.3.4. Brochage du module BlueSMiRF :

Comme nous l'avons vu précédemment le module BlueSMiRF dispose de six pins qui sont :

- ***VCC-Gnd :*** BlueSMiRF peut être alimenté avec une alimentation comprise entre 3.3V et 6V, dans notre projet nous l'avons alimenté avec les 5V que dispose la carte Arduino .

- ***RX-I (Receive into) :*** cette broche est utilisée pour la réception série des Données. Cette broche doit être connectée à la broche ***Tx*** de l'émetteur, dans notre cas elle va être connectée à la broche 1de la carte Arduino.
- ***Tx-O (Transmit from)*** : cette broche se charge de la transmission série des données. Elle doit être connectée à la broche ***Rx*** du récepteur, qui est la broche 2 de la carte Arduino.
- ***CTS-I (Clear To Send) et RTS-O (Ready To Send) :*** s'utilisent pour le contrôle du flux (dans notre projet nous les s'avons pas utilisé).

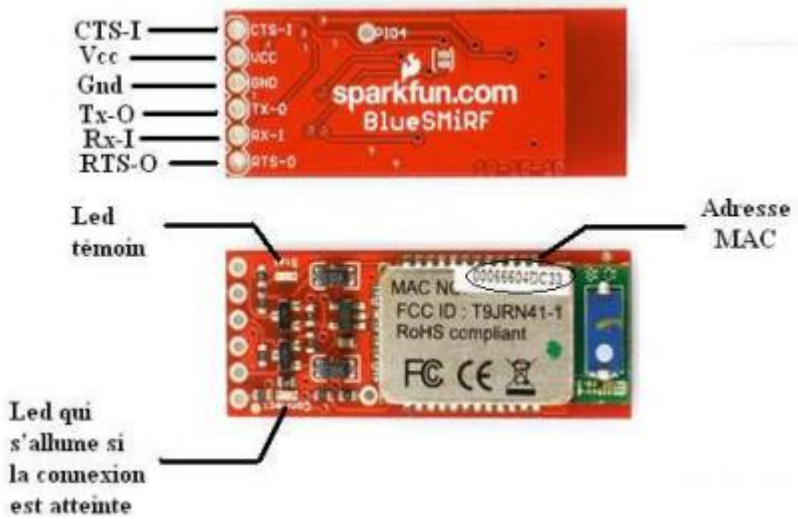


Figure 2.34.Brochage du module BlueSMiRF.

-Brochage du module BluesMiRF avec la carte Arduino:

On a utilisé pour la communication avec le module Bluetooth les deux broches Tx et Rx de la carte Arduino : Rx du BlueSMiRF vers Tx de la carte et la pin Tx du BlueSMiRF vers Rx de la carte.

La figure suivante nous permet de voir comment connecter le module Bluetooth avec la carte Arduino.

Le fil bleu est connecté au Bluetooth sur sa broche Tx vers Rx de la carte Arduino, tandis que le fil vert est connecté au Bluetooth sur la broche Rx vers Tx de la catre. Les fils rouge et noir sont l'alimentation et le GND comme le montre l'image de notre montage.

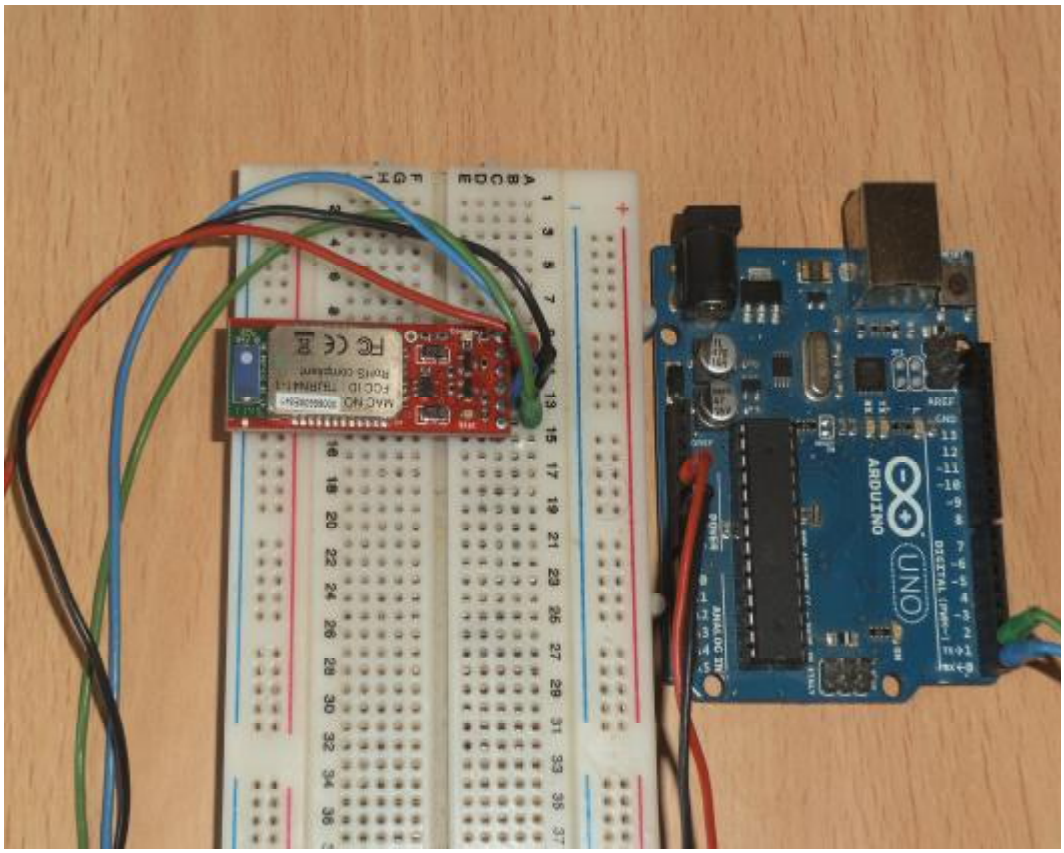


Figure 2.35.Brochage du BlueSMiRF à la carte Arduino.

Chapitre III

Programmation et conception logicielle.

Introduction :

Ce dernier chapitre est consacré pour la description des différents logiciels utilisés au cours de notre travail, ainsi que les différentes étapes suivies pour le développement de notre maquette.

III.1.L'environnement de programmation Arduino :

La carte Arduino présente le noyau de notre système, et pour qu'on puisse la programmer, un logiciel compatible est mis à notre disposition avec cette carte, ce logiciel est Arduino ***EDI*** (***Espace de développement Intégré***), qui porte le même nom de la carte Arduino.

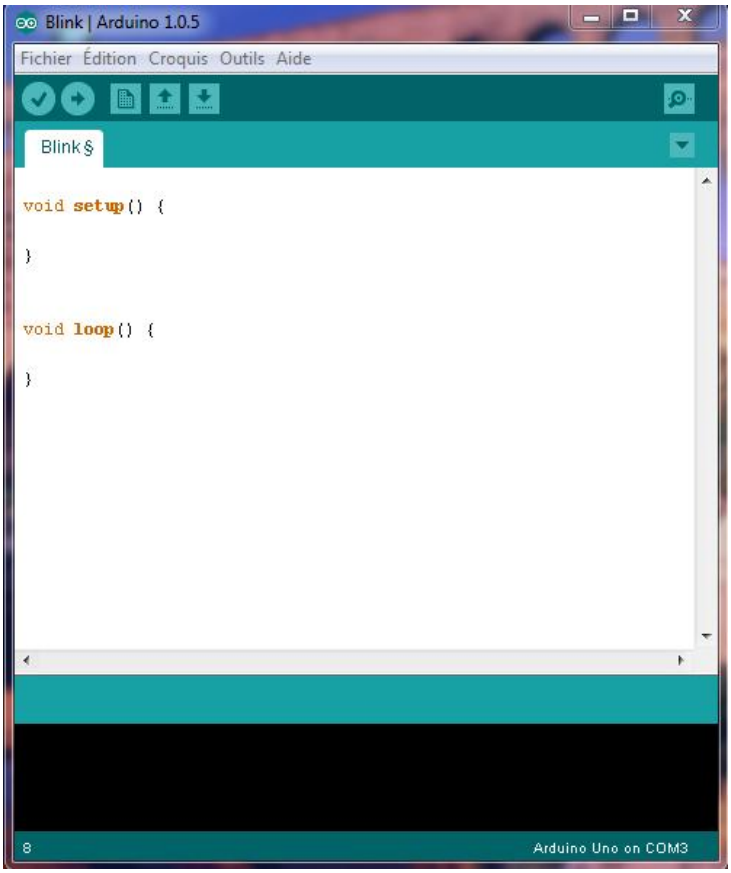


Figure3.1.Fenêtre principale de l'environnement de programmation Arduino.

III.1.1.Description du logiciel Arduino :

Le logiciel Arduino a pour fonctions principales :

• Ecrire et compiler des programmes pour la carte Arduino.

• Se connecter avec la carte Arduino pour y transférer les programmes.

• Communiquer avec la carte Arduino.

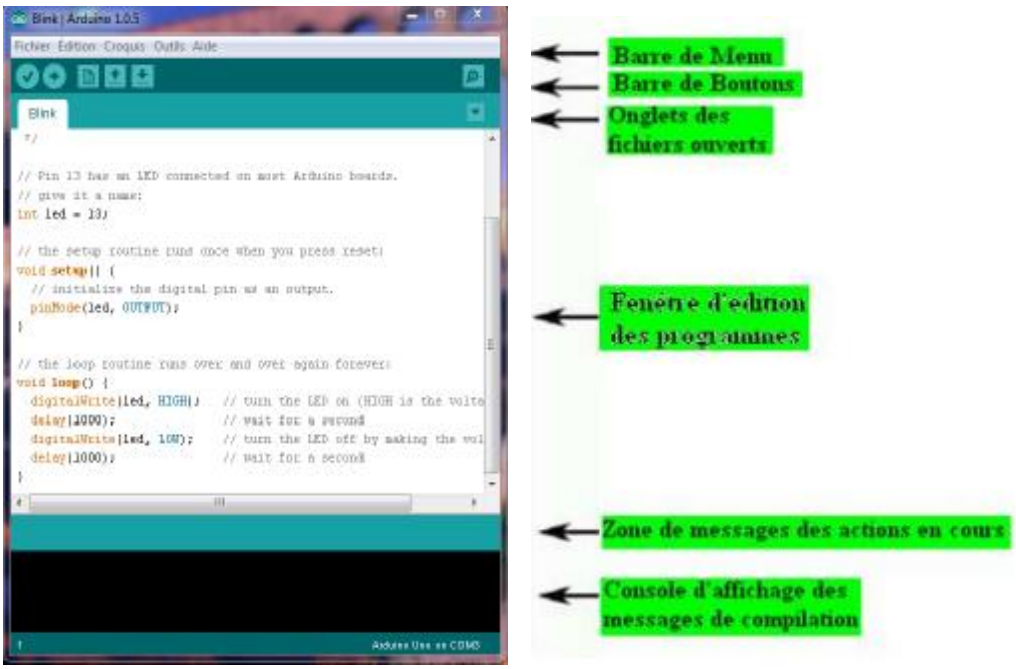


Figure 3.2.Les différentes parties de la fenêtre principale du logiciel Arduino.

Cet environnement comporte :

- ***Une barre de menus :*** comme pour tout logiciel une interface graphique.
- ***Une barre de boutons:*** elle donne un accès direct aux fonctions essentielles du logiciel, elle donne au logiciel une simplicité d'utilisation.
- ***Un éditeur:*** c'est dans cette espace qu'on va écrire notre programme.
- ***Une zone de messages:*** permet d'afficher et d'indiquer l'état des actions en cours (ex : vérification, téléversement).

- ***Une console texte:*** elle permet d'afficher les messages concernant le résultat de la compilation du programme (il nous indique s'il y a des erreurs).
- ***Un moniteur série :*** ce moniteur est utilisé pour la visualisation des données (message, valeurs, caractères) transmises vers l'ordinateur via le câble USB connecté à la carte Arduino et d'envoyer des caractères vers la carte Arduino.

  Cette fonctionnalité permet une mise au point de la facilité des programmes et d'afficher sur l'ordinateur l'état des variables d'un programme où il y a des calcules ou des mesures, aussi le résultat de la conversions analogique-numérique : un élément essentiel pour améliorer, tester et corriger ses programmes.

  Nous l'avons utilisé dans notre projet pour vérifier et visualiser les résultats des mesures des capteurs utilisés.

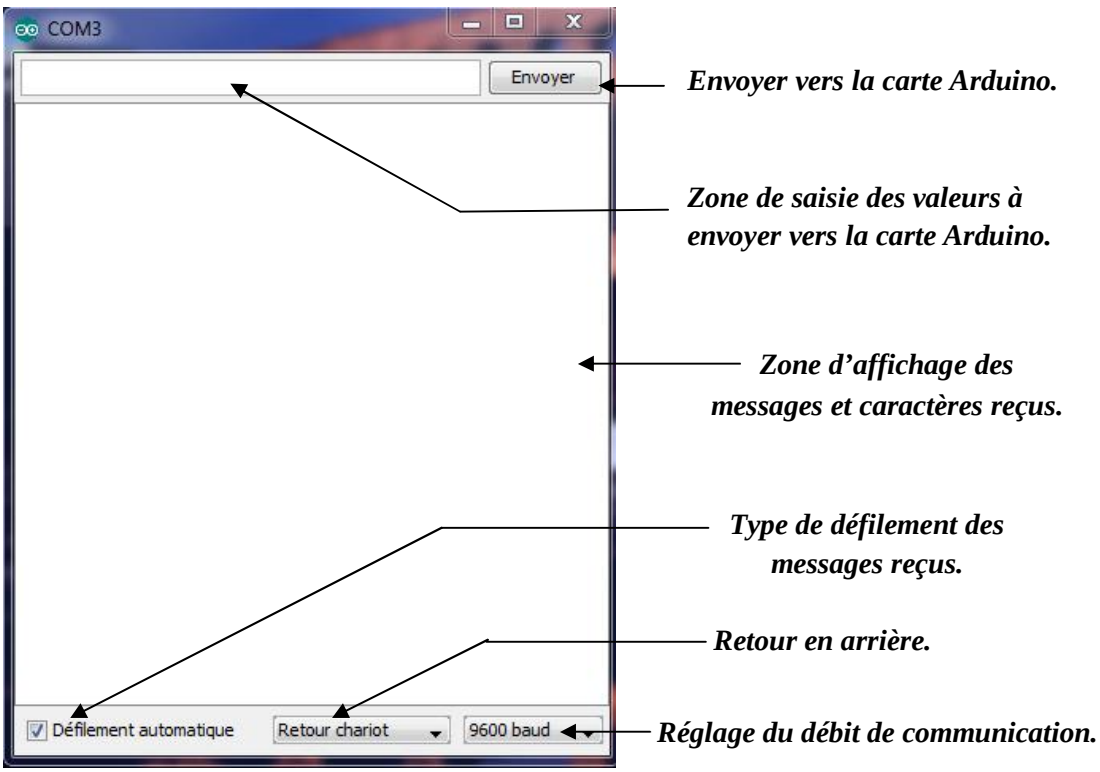


Figure 3.3.Fenêtre du moniteur série.

III.1.2.Description de la barre des boutons :

La barre des boutons nous permet de vérifier la syntaxe, le transfère des programmes, aussi la création, l'ouverture et la sauvegarde d'un code écrit. Elle permet aussi d'ouvrir le moniteur série. Elle dispose de six boutons ou icônes (ça diffère d'une version du logiciel à un autre):

Figure 3.4.Barre des boutons.

Vérifier/compiler : ce bouton nous permet de vérifie s'il y a des erreurs dans le programme en précisant la ligne où se trouve l'erreur.

Compiler/télévrser : ce bouton permet de compiler et de téléverser le programme vers la carte Arduino.

Nouveau : en appuyant sur cette icône, une fenêtre d'édition vide s'ouvre pour la saisie d'un nouveau programme.

Ouvrir : permet d'accéder à la liste de tous les programmes qui se trouvent dans le "livre des programmes". On peut ouvrir un exemple de programme dans la fenêtre courante en cliquant dessus.

Sauvegarder : avant de quitter le programme saisi, il faut l'enregistrer, et ça peut se faire en cliquant sur cette icône.

Moniteur série : pour ouvrir la fenêtre du moniteur série, il suffit d'appuyer sur ce bouton et on pourra voir les résultats des mesures.

III.1.3.Description de la barre des menus :

Le logiciel Arduino est équipé d'une barre des menus qui contient un ensemble de menus (d'où le nom "Barre des menus") permettant de faire la gestion du programme.

Figure 3.5.La barre des menus.

- ***Fichier :*** contient les différentes options de création, d'ouverture, de sauvegarde, d'impression d'un programme, ou l'ouverture d'un exemple parmi les exemples qui accompagnent le logiciel Arduino.
- ***Edition :*** contient les options de copier/coller, sélection, et les options de recherche.
- ***Croquis :*** ce menu contient les différentes fonctions de la barre des boutons, ainsi que les options d'ajout de librairies ou de fichiers.
- ***Outils :*** c'est dans ce menu qu'on sélectionne le type de carte à programmer, et le port série utilisé ainsi que la fonction de chargement du bootloader dans l'ATmega.
- ***Aide :*** ce menu est fait pour donner de l'aide concernant les différents problèmes rencontrés au niveau du logiciel Arduino.

III. 1.4.Les étapes de téléversement d'un programme vers la carte Arduino :

Une fois le programme est saisi (première phase), une deuxième phase consiste à vérifier s'il y a des erreurs dans le programme. Pour cela il suffit de cliquer sur le bouton de vérification se trouvant à l'extrémité gauche de la barre des boutons et attendre un petit moment.

Apres cette vérification s'il y a des erreurs, un message en orange s'affiche en dessus de l'onglet d'édition des programmes indiquant qu'elle est l'erreur et la ligne où se trouve (pour la correction des erreurs il suffit de consulter l'aide disponible dans la barre des menus). Une fois la correction des erreurs terminée, on passe à la phase de la sélection du type de la carte à programmer et le port série sur lequel est branchée cette carte, ce petit réglage peut se faire à l'aide du menu ***Tools (Outils)***. Une fois terminé il reste qu'à cliquer sur le bouton ***vérifier*** qui se trouve dans la barre des boutons.

Si tout marche bien les deux LEDs Rx et Tx de la carte s'allument, sinon un message d'erreur s'affiche dans la console des messages de compilation.

A ce niveau la carte est programmée et prête pour l'exécution du programme, il suffit juste de la mettre sous tension et de lui brancher les composants ou les éléments nécessaire pour le bon déroulement du processus à exécuter.

III.2. Le logiciel BlueSoleil :

Les résultats de nos mesures sont transmises vers l'ordinateur via bluetooth, mais il se trouve que les bluetooths des ordinateurs ne transmettent et reçoivent que des objets ou fichiers (comme les images, textes, vidéos), or dans notre cas on doit transmettre un ensemble de données et non pas des objets ou fichiers et de manière continue.

C'es pour cela la qu'on doit créer un ***PAN (Personal Area Network).*** Il existe un logiciel qui nous permet de créer ce type de réseau avec différents éléments (PDA (Personal Digital Assistant), comme les ordinateurs et les imprimantes), et la transmission des données va se faire de manière continue. Ce logiciel est ***BlueSoleil.***

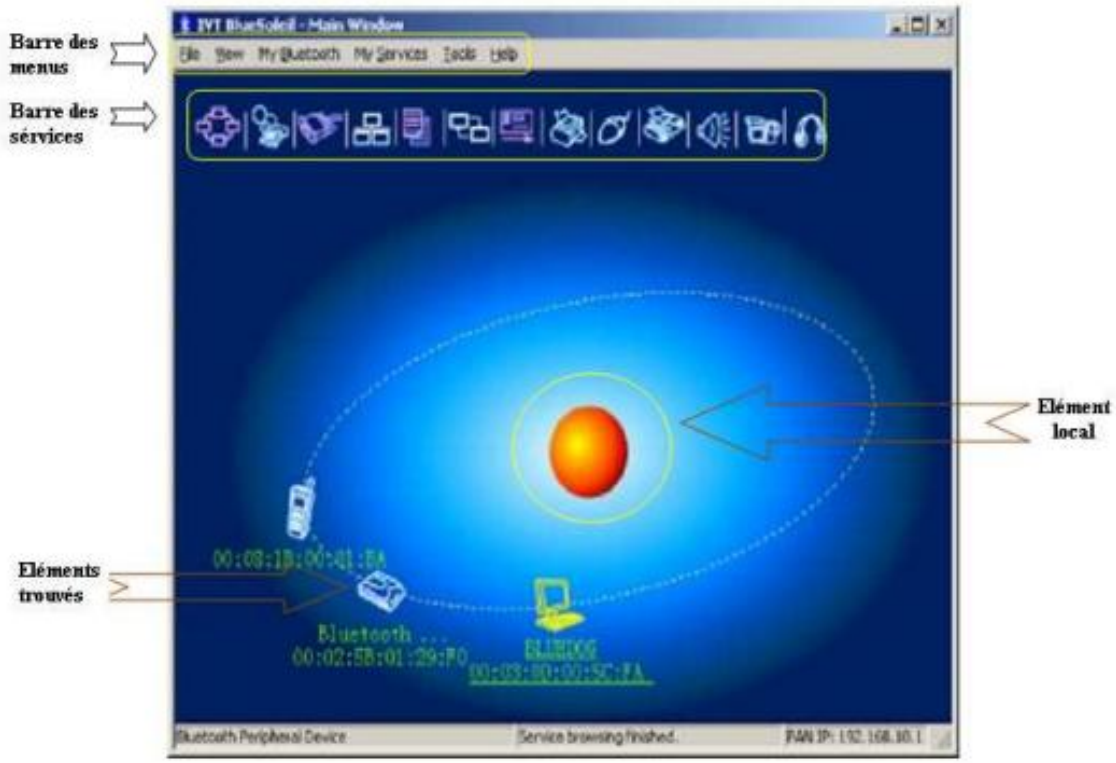


Figure 3.6.Fenêtre principale de Bluesoleil.

III.2.1.Description de BlueSoleil :

Bluesoleil comporte un nombre important de fonctions et services qui sont :

- Ø Possibilité de créer un PAN.
- Ø Création d'un port série virtuel.
- Ø Accès au réseau LAN (Local Area Network).
- Ø Transfert de fichiers.
- Ø Transfert d'images.
- Ø Partage d'imprimante et Fax.

Ce qui nous intéresse dans notre projet est la création du PAN. Au démarrage du Bluetooth, BlueSoleil fait une recherche des dispositifs ayant un Bluetooth allumé. Pour se connecter à un dispositif, il suffit de faire un double clique dessus et l'envoi de données commence, mais parfois le dispositif comme dans notre cas demande une clé pour accepter la communication. Le module BlueSMiRF demande une clé dite d'appairage qui est par défaut 1234.

III.3. Le logiciel de programmation graphique LabVIEW :

Après avoir transmis les résultats des mesures faites via bluetooth, nous avons développé une interface graphique avec le logiciel LabVIEW ***(Laboratory Virtual Instrumentation Engineering Workbench)***. C'est un logiciel de développement graphique qui permet de créer de des applications modulaires et extensibles pour la conception d'applications de mesure, de contrôle et de test. Sa principale caractéristique est son langage de programmation qui est un langage graphique (le langage G) destiné à créer un programme sous forme de diagramme.

Figure 3.7. Les différentes fonctions de LabVIEW.

Des bibliothèques étendues de fonctions et de routines répondant à la plupart des besoins en programmation sont disponibles. Il contient aussi des bibliothèques de fonctions spécifiques pour l'acquisition de données qui proviennent des éléments connectés sur une liaison série. Il comporte également des outils de mise au point permettant de placer des points d'arrêt et animer l'exécution du programme.

III.3.1.Les instrument virtuels (VI) :

On appelle Instruments Virtuels ***(VI)*** les programmes LabVIEW. Les Vis se composent de deux parties : la face avant (front panel) et la face arrière ou le diagramme (Block Diagram).

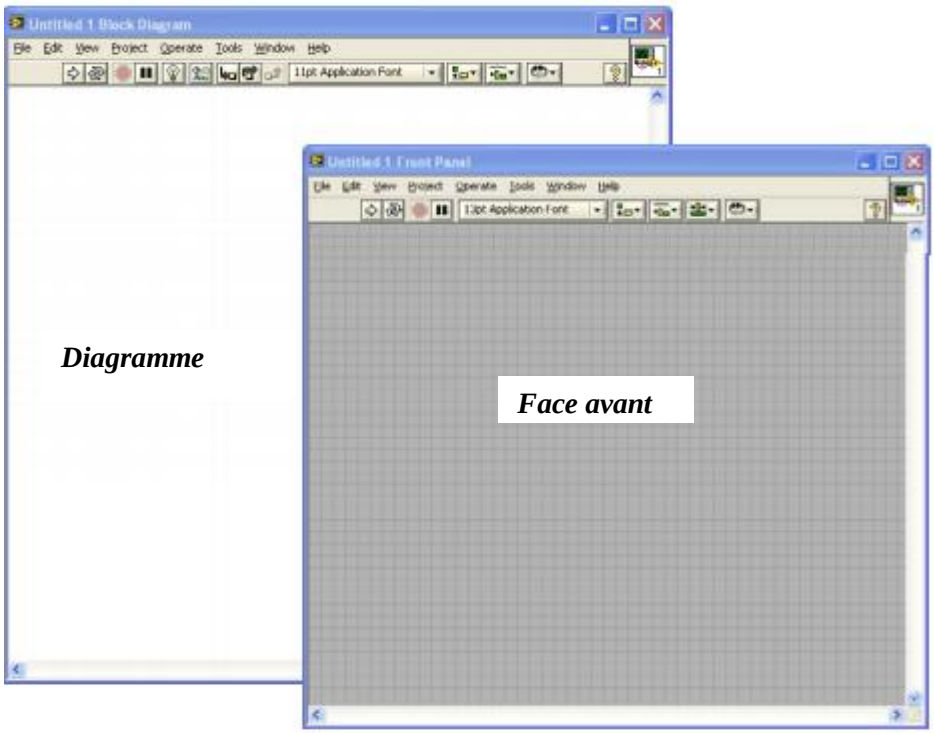


Figure 3.8.Fenêtres principales de LabVIEW.

- ***La face avant :***

C'est l'interface utilisateur du VI. Elle est construite avec des commandes et des indicateurs qui sont respectivement les terminaux d'entrées et de sorties interactifs du VI. On trouve dans les commandes des boutons poussoir, des cadrans et d'autres périphériques d'entrée. Les indicateurs par exemple sont les afficheurs et les graphes. Les commandes simulent les périphériques d'entrée d'instruments et fournissent les données au diagramme du VI tandis que les indicateurs simulent les périphériques de sortie d'instruments et ils affichent les données générées par le diagramme.

La figure suivante représente les différents indicateurs et outils de décoration qui sont rangés dans la ***palette des commandes***.

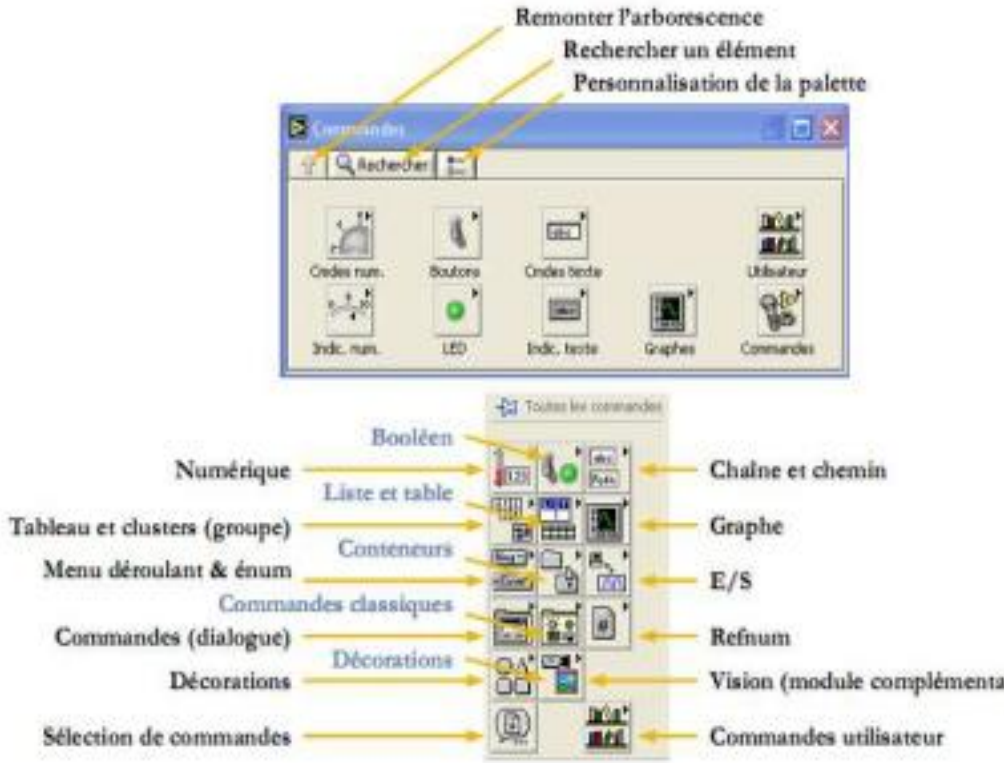


Figure 3.9.Palette des commandes.

- ***Le diagramme :***

Une fois la face avant construite, on doit ajouter le code qui est contenu dans le diagramme en utilisant les représentations graphiques des fonctions pour le contrôle des objets de la face avant. Les objets de la face avant apparaissent comme des terminaux sur le diagramme et un terminal disparait que lorsque son objet correspondant dans la face avant est supprimé. C'est dans la palette des fonctions qu'on trouve les objets du diagramme (comme les portes logiques et les boucles).

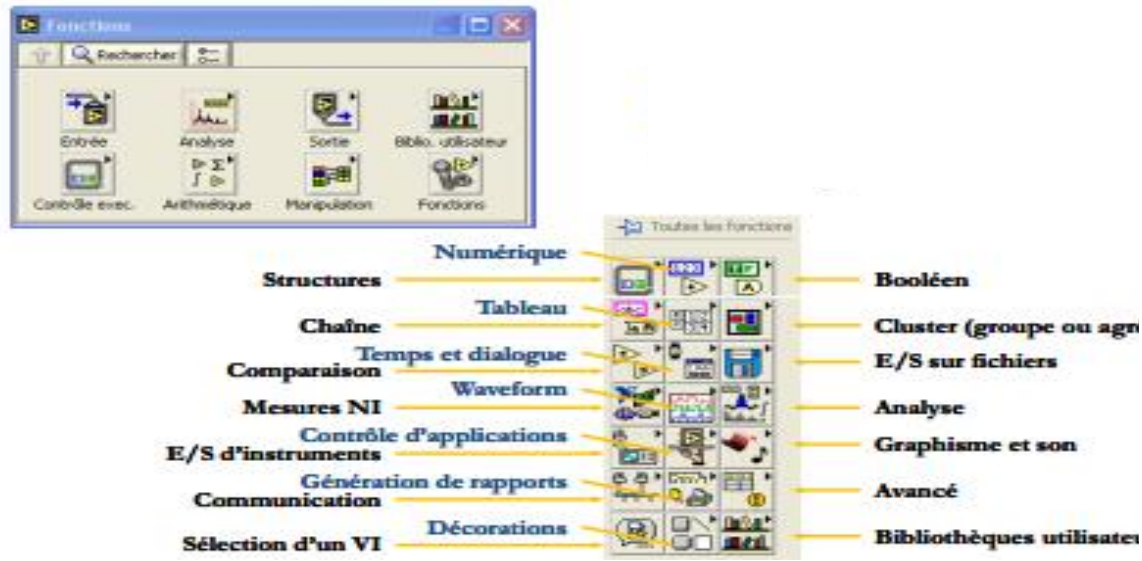


Figure 3.10.Palette des fonctions.

Une autre palette existe et elle peut être utilisée pour les deux faces, c'est la ***palette d'outils.***

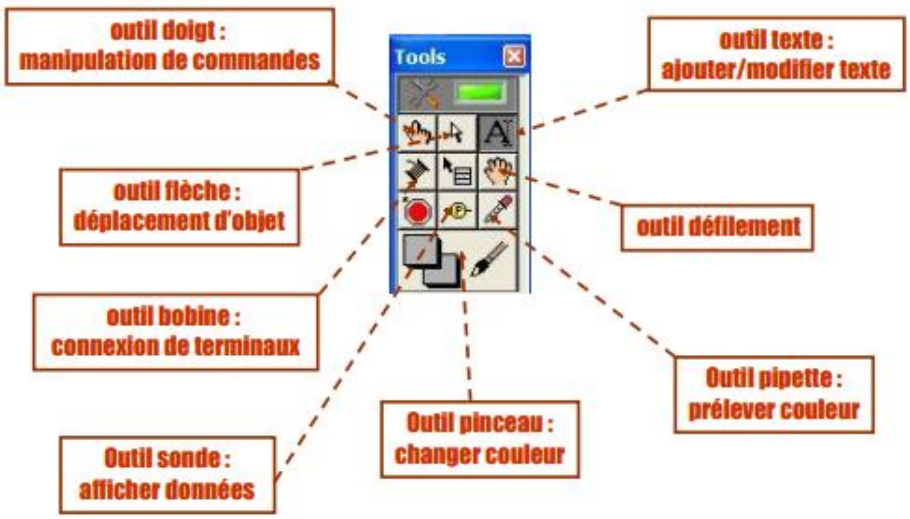


Figure 3.11.Palette d'outils.

III.3.2.Les données de bases dans LabVIEW :

Chaque donnée dans LabVIEW a son type et à chaque type de base correspond une couleur. Le tableau suivant illustre les différents types de données :

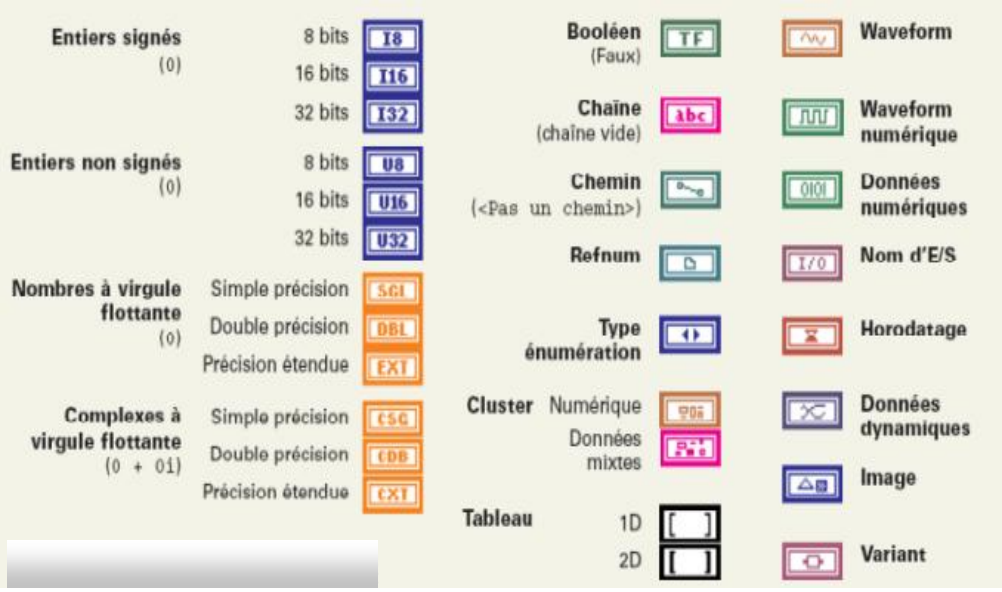


Figure 3.12.Les différents types de données.

III.3.3.Description de l'interface réalisé :

Pour visualiser les données transmises par notre carte, nous avons réalisé une interface qui contient un écran comme un oscilloscope pour visualiser le bruit et des fenêtres pour visualiser la température, la vitesse de rotation et l'accélération.

L'acquisition des données va se faire en utilisant la bibliothèque **VISA** ***(Virtual Instrumentation Software Architecture)*** qui permet de communiquer avec des instruments reliés au pc et de configurer tous les types de communication comme étant une communication série.

Figure 3.13.Bus de liaison s'adaptant avec VISA.

Une palette des différentes opérations et attributs de VISA existe et à laquelle on peut accéder en sélectionnant ***Fonctions E/S d'instruments VISA.***

Figure 3.14.Opérations et attributs de VISA.

III.3.4.Configuration du port :

Pour pouvoir acquérir des données avec LabVIEW, on doit d'abord configurer le port par lequel va se faire l'acquisition, la palette de VISA contient une fonction permettant de le faire qui est ***« VISA Configuration Port ».***

Cette fonction c'est le noyau du système d'acquisition de notre interface sous LabVIEW, elle permet la configuration du port USB où se trouve le Bluetooth comme étant un port série comportant plusieurs paramètres comme le débit, nombre de bits de parité et le bit d'arrêt.

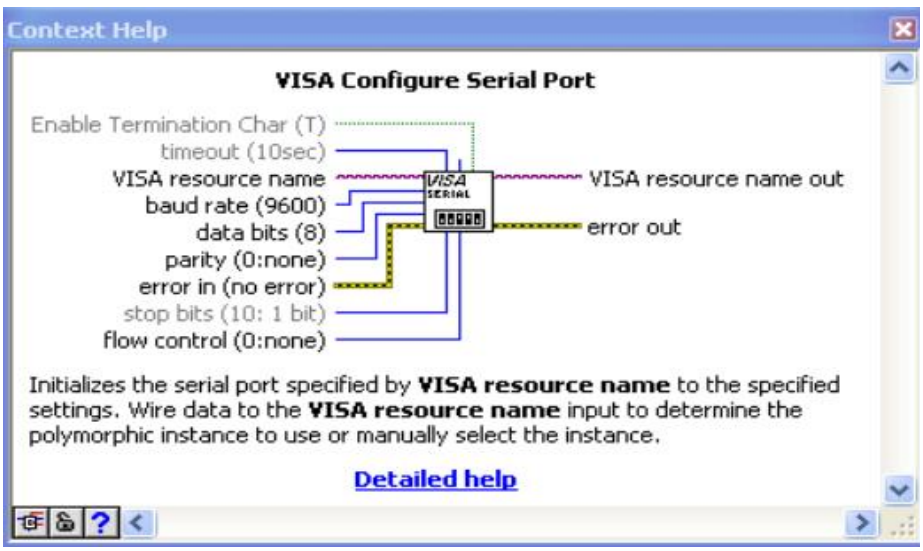


Figure 3.15.La fonction « VISA Configure Serial Port ».

III.3.5.Configuration de la taille du tampon d'entrée/sortie :

On trouve aussi dans la palette VISA la fonction ***VISA Set I/O Buffer Size Fonction*** qui permet de définir la taille du tampon d'entrée/sortie c'est-à-dire pour la transmission et la réception des données. Cette fonction définir la

taille du tampon uniquement pour la réception (masque de 16 bits), ou bien uniquement pour la transmission (masque de 32 bits), ou pour la transmission et la réception au même temps (le masque est de 48 bits).

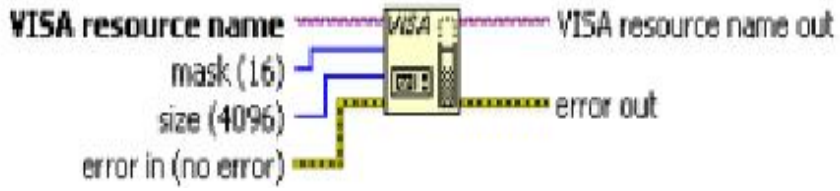


Figure 3.16.La fonction « VISA Set I/O Buffer Size Fonction.

Il reste maintenant à lire les données provenant du port. Cela se fait en utilisant la fonction ***VISA Read Fonction***. Elle nous permet de lire les données sous forme d'octets et les transférer vers l'instrument de visualisation ou de sauvegarde.

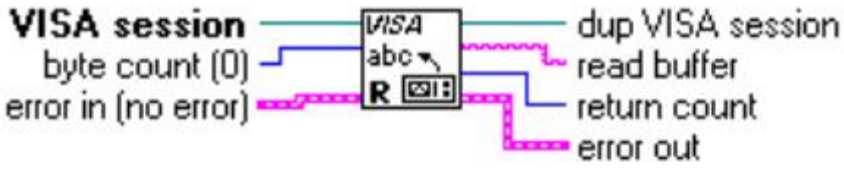


Figure 3.17.La fonction « VISA Read Fonction ».

Après avoir fini la conception des deux faces, nous avons obtenu l'interface suivant :

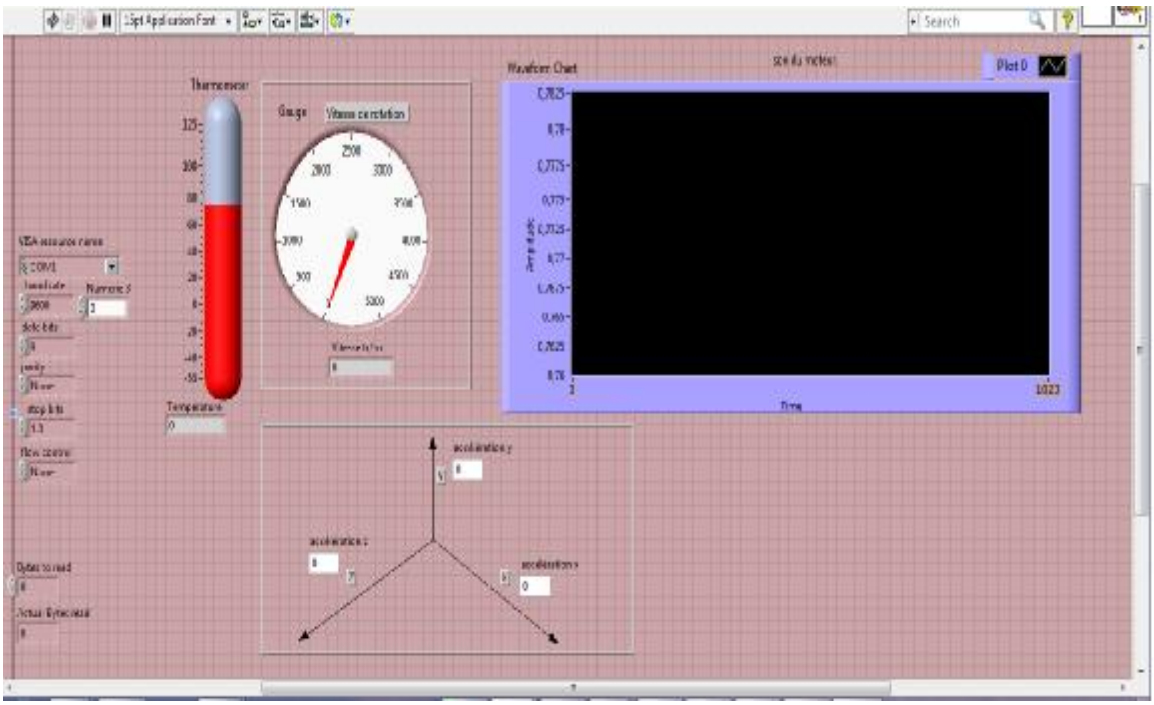


Figure 3.18.L'interface graphique.

La figure qui suit montre le circuit global de notre conception et réalisation qu'on a fait dans notre projet qui est présenté dans ce mémoire. Toute fois, des modifications peuvent être apportées dans le schéma comme le brochage du DS18B20 qu'on peut relier vers une pin analogique.

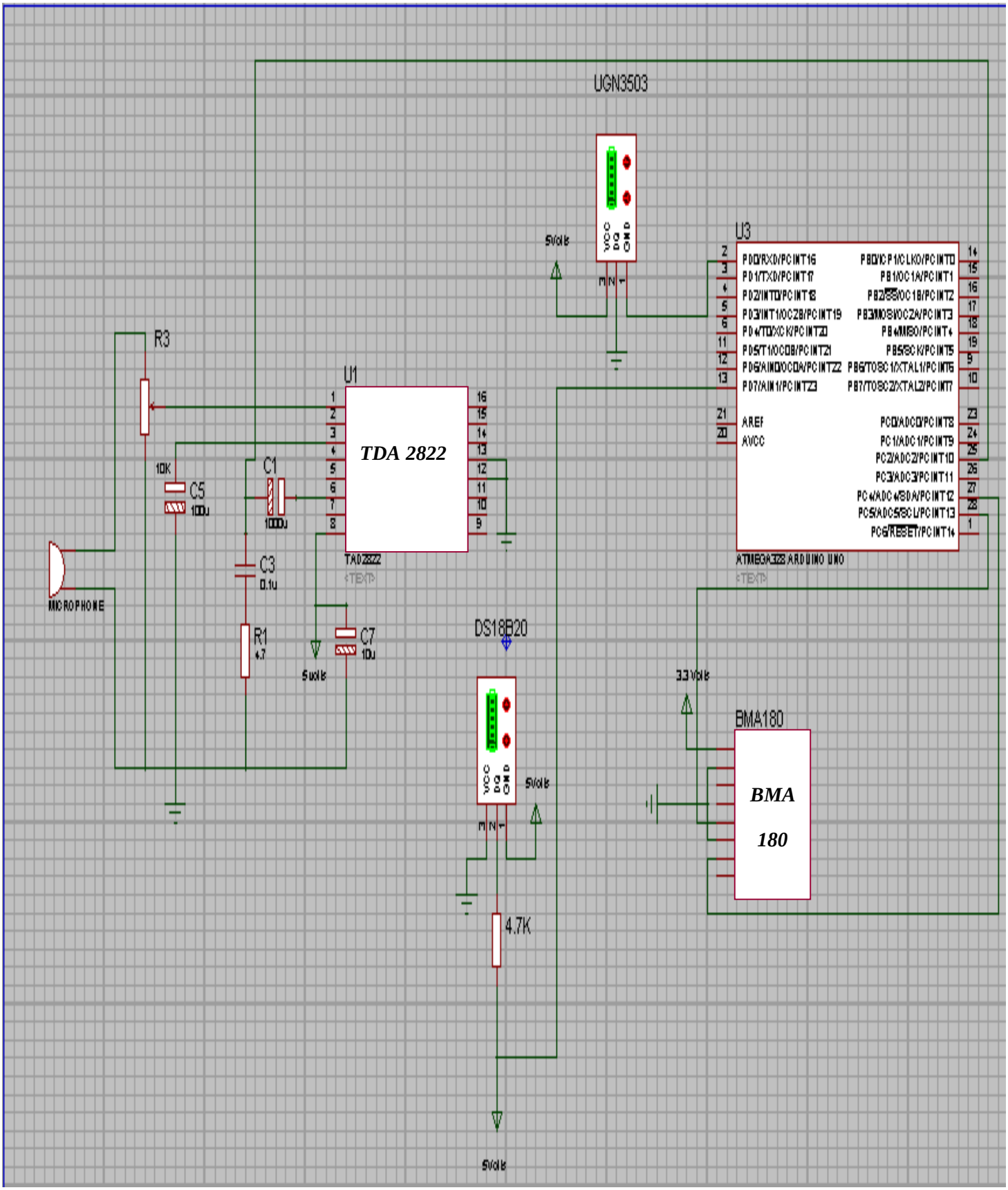


Figure 3.19. Schéma global réalisé.

Conclusion générale

Dans notre projet, nous avons conçu, étudié et réalisé un banc de mesure pour la surveillance d'un moteur électrique tournant. Nous avons pu mesurer les paramètres sensibles et principaux permettant de diagnostiquer les défaillances dans les moteurs électriques qui sont la température, l'accélération, la vitesse de rotation et le bruit. Nous avons pu utiliser pour le transfert de ces données la liaison Bluetooth puis les afficher sur notre ordinateur.

Le choix des capteurs utilisés dans notre projet est basé sur leur temps de réponse, facilité d'implantation, la plage de fonctionnement et le domaine d'application.

Toute fois, des améliorations et des modifications peuvent être apportées à notre projet comme l'utilisation de plusieurs capteurs de température et d'accélération, par exemple pour surveiller plusieurs moteurs en même temps ou ajouter des capteurs d'humidité et de pression ou utiliser une liaison Zigbee pour augmenter la portée de la transmission.

En fin, nous souhaitons que notre travail serve de support pour les promotions à venir à fin d'attirer l'attention sur l'importance de la pratique dans le domaine de l'électronique.

Bibliographie

[1] Ali Ibrahimi, thèse DOCTORA, Contribution au diagnostique des machines électromécaniques : exploitation des signaux électriques et de la vitesse instantanée, 2009.

[2] Nachef Mhena, mémoire MASTER, conception et réalisation d'un système de télésurveillance médicale à base d'une carte ARDUINO « Transmission Bluetooth, 2013. UMMTO, département d'électronique.

[3] Nechiche Sofiane, Remichi Hichem, étude et réalisation d'un banc de mesure pour la surveillance d'un moteur à base de la carte de développement OLIMEX, UMMTO, département d'électronique, 2013.

[4] T. Toumi, N. Guerbas, mémoire ingénieur, Conception et réalisation d'un système d'acquisition et de commande d'un four de chimie avec une interface LabVIEW, UMMTO.département d'électronique, 2012.

[5] N. Tandon G.S. Yadava, K.M. Ramakrishna, A comparison of some condition monitoring techniques for the detection of defect in induction motor ball bearing, Mechanical Systems and Signal Processing (2007).

[6] TD[99] O. V. Thorsen and M. Dalva. Failure identification and analysis for highvoltage induction motors in the petrochemical industry. IEEE Transactions on Industry Applications, vol. 35(no. 4) :pp. 810–818, July-Aug. 1999.

[7] Baudoin Fabrice, « les capteurs. »

[8] Migeon André, « Application industrielle des capteurs. »

Sites:

WWW.arduino.cc.com

www.abcelectronique.com

www.alldatasheet.com