REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE MOHAMED BOUDIAF - M'SILA

FACULTE DE TECHNOLOGIE
DEPARTEMENT D'ELECTRONIQUE
N° :2017/CI03/97/482

DOMAINE : SCIENCE ET TECHNOLOGIES
FILIERE : ELECTRONIQUE
OPTION : CONTROLE INDUSTRIEL

Mémoire présenté pour l'obtention
Du diplôme de Master Académique

Par:

REGUIG BERRA Imadeddine

ALLAM Yassine

Intitulé

Etude et réalisation d'une station météo connectée par wifi

Soutenu devant le jury composé de:

GUERMAT Noubeil	Université de M'sila	Président
BOUCHAMA Idris	Université de M'sila	Rapporteur
MEZACHE Hatem	Université de M'sila	Examinateur

Année universitaire : 2016 /2017

Remerciement

*Nous remercions **ALLAH** qui nous aide et nous donne*

la patience et le courage durant ces

langues années d'étude.

Nous tenons à remercier grandement notre Encadreur

Dr: BOUCHAMA IDRIS

pour sa grande disponibilité et ses précieux conseils.

Nous remercions également tous les enseignants du

département d'électronique d'université de M'sila

*plus spécialement les **membres de jury** de notre travail.*

Merci à tous.

Dédicace

À la plus belle créature que Dieu a crée sur terre ,,,

À cet source de tendresse, de patience et de générosité,,,

À ma mère

À le personne qui à toujours me donné la volonté

À mon père

À tous mes frères et sœurs qui ont été toujours avec moi

À tous mes amis et collègues

À tous les étudiants de la promotion 2016/2017

Département d'électronique

A tous ceux qui, par un mot, m'ont donné la force de continuer …

Allam yassine

Reguig berra imadeddine

Liste des abréviations

ADC (Analog to Digital Converter) : Convertisseur Analogique/Numérique.

APIs (Application Program Interface) : Interface de Programme d'Application.

DC (Direct Current) : Courant Continu.

EEPROM (Electrically-Erasable Programmable Read-Only Memory) : Mémoire à Lecture Seule Programmable Effaçable Electriquement.

GPIO (General-Purpose Input/Output) : Usage Général Entrée/Sortie.

IoT (Internet of Things) : IdO(Internet des Objets)**.**

I/O (Input/Output) : Entrée/Sortie.

IP (Internet Protocol) : Protocole Internet.

I2C (Inter-Integrated Circuit) : Circuit Inter-Intégré.

PC (Personal Computer) : Ordinateur Personnel.

PCINT (Pin Change Interrupts) : Interruptions Pin de Changement.

PWM (Pulse Width Modulation) : Modulation de Largeur d'Impulsion.

RF (Radio Frequency) : Fréquence Radio.

RTD (Resistance Temperature Detectors) : Résistance Détecteurs de Température.

SRAM (Static Random-Access Memory) : Mémoire Statique à Accès Aléatoire.

TCP (Transmission Control Protocol) : Protocole de Contrôle de Transmission.

VoIP (Voice over Internet Protocol) : Protocole de Voix sur Internet.

WLAN (Wireless Local Area Network) : Réseau Local sans Fil.

Liste des figures

Liste des tableaux

Sommaire

Sommaire

Chapitre II: Présentation d'Arduino

Chapitre III: Conception et réalisation de la station météo

Introduction générale

Introduction générale

Aujourd'hui, l'électronique est de plus en plus remplacée par l'électronique programmée. On parle aussi de système embarquée ou d'informatique embarquée. Son but est de simplifier les schémas électroniques et par conséquent réduire l'utilisation des composants électroniques, réduisant ainsi le coût de fabrication d'un produit. Il en résulte des systèmes plus complexes et performants pour un espace réduit [1].

Depuis quelque décennies, le besoin d'observer et de contrôler des phénomènes physiques tels que la température, la pression, l'humidité ou encore la luminosité, est essentiel pour de nombreuses applications industrielles et scientifiques. Par exemple, dans le domaine de l'écologie, la surveillance de polluants comme l'Ozone, le NO_2 ou encore le CO_2, pourrait considérablement augmenter la qualité de vie dans les villes [2].

Dans notre travail nous allons réaliser une station météo de mesure des phénomènes physiques existent. Cette station sera connectée par le réseau WIFI au réseau internet. Dans l'ensemble, la station comprenant plusieurs capteurs tels que: Capteur de Température, Capteur d'Humidité, Capteur de Pression et Capteur d'Ensoleillement. Une carte d'acquisition à base d'Arduino UNO est pour but de transférer les données mesurées de ces capteurs vers le Micro-ordinateur (PC) à travers une carte WIFI série sans fil.

Deux principaux objectifs sont visés: Le premier objectif est de regrouper suffisamment d'informations sur une grande catégorie de cartes d'interfaçage à base de l'Arduino: son langage de programmation, sa construction, son principe de fonctionnement.

Le deuxième objectif consiste à réaliser une connexion sans fil entre la carte Arduino et le PC à travers la carte WIFI série sans fil.

Notre projet expérimental est donc consiste à réaliser un système de mesure en temps réel de l'ensemble des grandeurs physiques existent. Les données sont traitées dans l'unité de traitement et de commande à base de l'Arduino UNO afin de les transférer par un câble USB ou par un protocole de transfère des données sans fils WLAN (Wireless Local Area Network) à travers la carte WIFI série sans fil de type ESP8266 (ESP-01). L'affichage des résultats est assuré par des courbes sur le site internet choisi s'appel: 'ThingSpeak'.

Ce mémoire est organisé en trois chapitres, organisés comme suit:

- Dans un premiers temps on commence par une introduction générale;
- Dans le premier chapitre, nous présentons des généralités sur les capteurs et une étude détaillée sur les capteurs de Température, Pression, Humidité et Ensoleillement;
- Le deuxième chapitre est consacré à la présentation du système Arduino UNO, on mettra la lumière sur le logiciel IDE Arduino;
- Le troisième chapitre est réservé pour l'étude et la réalisation de notre station météo. On présente les caractéristiques essentielles du tout les capteurs utilisés dans ce travail tel: (DHT22, BMP180, TEMT6000, ESP8286), et la méthode d'affichage sur ThingSpeak;
- Enfin, on termine par une conclusion générale.

Chapitre I

Généralités sur les capteurs

I.1. Introduction

Durant ces dernières années la technologie de fabrication des cartes de commande à base de microcontrôleurs a connue une évolution remarquable, cette révolution prodigieuse de la microélectronique a conduit à la fabrication des systèmes de commande de plus en plus complexes, offrant des avantages meilleurs tels que: la simplicité de la programmation, la vitesse d'exécution, les ports d'entrée/sortie... etc. Les progrès réalisés dans les domaines de la microélectronique permettent aussi de produire des composants de quelques millimètres cubes de volume, appelés le capteur [2]. Le besoin d'observer et de contrôler des phénomènes physiques tels que la Température, la Pression, l'Humidité ou encore la Luminosité est essentiel pour de nombreuses applications industrielles et scientifiques.

Dans ce qui suite de ce chapitre, nous allons voir la conception du capteur à travers un certains nombre de points et un plan méthodologique que nous avons adopté. Nous commencerons par quelques définitions d'un capteur, classification des capteurs, ensuite les caractéristiques des capteurs. Avant de conclure ce chapitre nous allons montré une étude détaillée sur les capteurs de Température, Pression, Humidité et Ensoleillement.

I .2. Définition principale du capteur

Un capteur est un dispositif ayant pour tache de transformer une mesure physique observée en une mesure généralement électrique qui sera à son tour traduite en une donnée binaire exploitable et compréhensible par un système d'information.

Parmi les différents types de mesures enregistrées par les capteurs, on peut citer entre autres: la Température, l'Humidité, la Luminosité, l'Accélération, la Distance, les Mouvements, la Position, la Pression, la Présence d'un Gaz, la Vision (Capture d'Image), le Son... etc. (voir Figure I.1).

La notion de capteur s'est évoluée avec le temps puisque leur domaine d'application s'est élargi. Les premiers capteurs n'étaient dédiés qu'à un unique type de mesure, les capteurs contemporaine sont la combinaison de plusieurs dispositifs capables de mesurer différentes mesures physiques. En outre, à ces possibilités de mesures multiples, les capteurs actuels ont vu se gérer des fonctionnalités qui leur permettent, en plus de l'enregistrement et de la détection d'événements mesurables, le traitement de ces données et leur communication vers un autre dispositif. On parle alors de capteur intelligent, capable à la fois de mesurer des données et de les communiquer avec d'autre capteurs au sein d'un réseau, tel qu'il est

caractérisé par sa capacité à effectuer une collecte des mesures, les traiter et à les communiquer au monde extérieur [3].

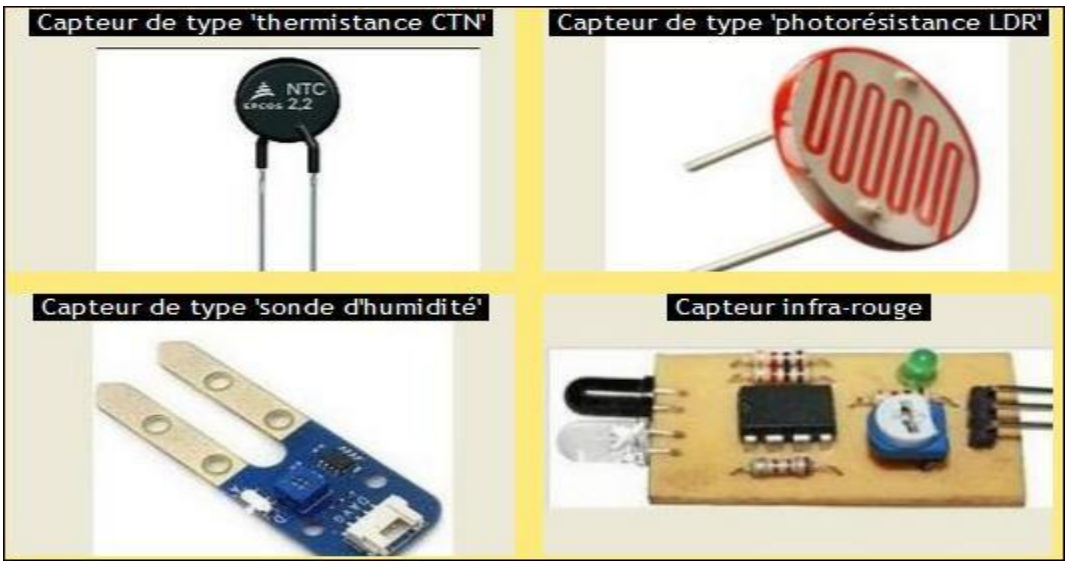

Figure I.1: Quelques exemples de capteurs.

a- Mesurande: c'est la grandeur physique d'entrée du capteur, que ce soit direct ou intermédiaire qu'on cherche à mesurer. Dans les automatiques industriels on cherche souvent à mesurer: la Température, la Pression, le Niveau, le Débit, le Couple, le Déplacement, la Vitesse, l'Accélération, la Distance ... etc.

b-Grandeur de sortie: elle est généralement de type électrique. Elle peut être de type analogique ou numérique. La Figure (I.2) montre les différents types de signaux que l'on peut rencontrer.

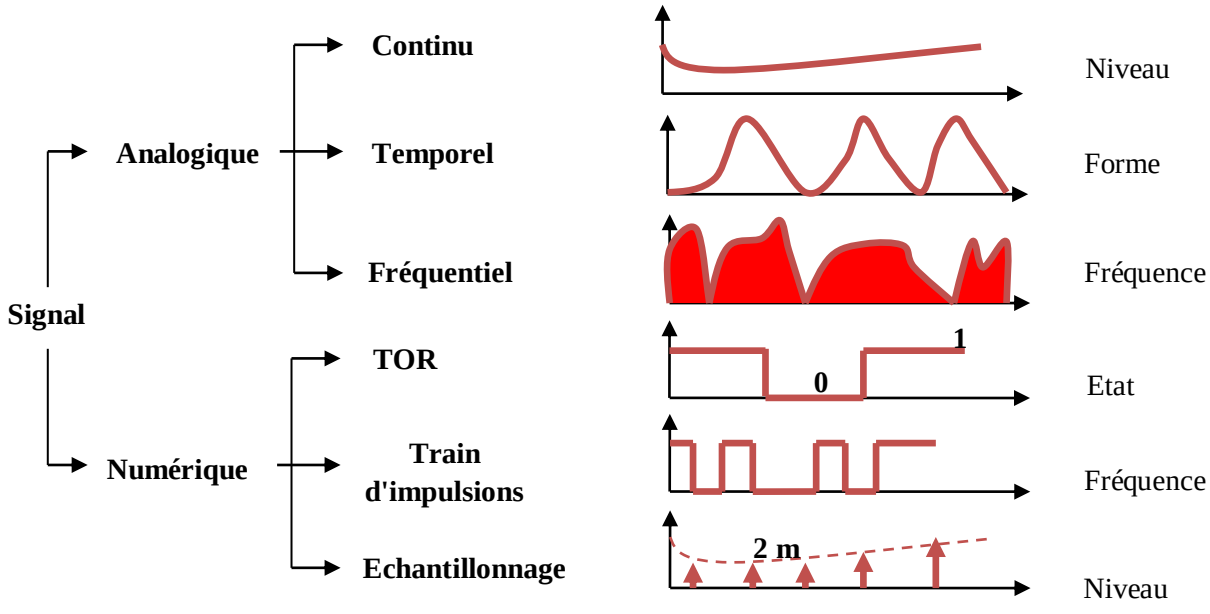

Figure I.2: Différent type de signaux.

- **Signal analogique**

Un signal est dit analogique si l'amplitude de la grandeur physique qui le représente peut prendre une infinité de valeurs dans un intervalle donné.

- **Signal continu:** c'est un signal qui varie "lentement" dans le temps: Température, Débit, Niveau.
- **Forme:** C'est la forme de ce signal qui est important: Pression Cardiaque, Chromatographie, Impact.
- **Fréquentiel:** C'est le spectre fréquentiel qui transporte l'information désirée: analyse vocale, sonar, spectrographie.

- **Signal numérique**

Un signal est dit numérique si l'amplitude de la grandeur physique qui le représente ne peut prendre qu'un nombre fini de valeurs. En général ce nombre fini de valeurs est une puissance de 2.

- **Tout ou rien (TOR) :** il informe sur un état bivalent d'un système. Exemple: une vanne ouverte ou fermée.
- **Train d'impulsion:** chaque impulsion est l'image d'un changement d'état. Exemple: un codeur incrémental donne un nombre fini et connu d'impulsion par tour.
- **Echantillonné:** c'est l'image numérique d'un signal analogique. Exemple: Température, Débit, Niveau.

c-Chaine de mesure: généralement, le signal de sortie n'est pas directement utilisable. On appelle chaine de mesure l'ensemble des circuits ou appareils qui amplifient, adaptent**,** convertissent, linéarisent, digitalisent le signal avant sa lecture sur le support de sortie.

d-Transducteur: c'est tout capteur intermédiaire qui permet de convertir le mesurande en une grandeur physique mesurable par le capteur qui fournie la grandeur électrique avant conditionnement.

e-Corps d'épreuve: en mécanique, notamment la conversion du mesurande en signal de sortie n'est pas directe. par exemple: la mesure d'une force nécessite de l'appliquer à un solide déformable auquel sera fixé un capteur de déformation. Ce solide et plus généralement tout corps intermédiaire entre le capteur et le mesurande est appelé corps d'épreuve.

f-Conditionneur: le signal de sortie du capteur peut être directement exploitable ou non. S'il n'est pas directement exploitable, il faut alors recourir à un élément nommé conditionneur. Il faut savoir que le capteur peut générer des signaux de plus ou moins grande amplitude. Ainsi, il faut donc que le conditionneur adapte le signal de sortie du capteur à celui du système de contrôle, de commande ou de mesure. Si le signal est par exemple faible, il devra l'amplifier.

Certains capteurs génèrent simplement des variations d'impédance. Cela nécessite une alimentation électrique de ces capteurs. La variation d'impédance se traduit par une variation de courant ou de tension électrique. Dans ce cas, le conditionneur fournira l'alimentation électrique au capteur et amplifiera le signal électrique (si besoin) en provenance de ce dernier.

I .3. Classification des capteurs

Si l'on s'intéresse aux phénomènes physiques mis en jeux dans les capteurs, on peut classer ces derniers en deux catégories.

- Capteurs actifs
- Capteurs passifs

I .3.1. Capteurs actifs

Fonctionnant en générateur, un capteur actif est généralement fondé dans son principe sur un effet physique qui assure la conversion en énergie électrique de la forme d'énergie propre à la grandeur physique à prélever, énergie thermique, mécanique ou de rayonnement.

Donc, un capteur actif produit lui-même un signal électrique de sortie par conversion de l'énergie fournie par le grandeur d'entrée ou de ces variations.

On va schématiser dans la Figure (I.3) ce type de capteur par un bloc possédant un accès "physique" et un accès "signal".

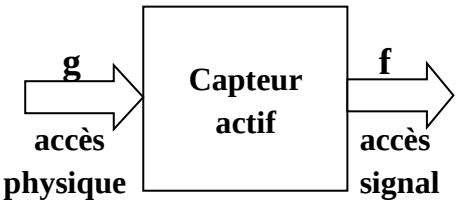

Figure I.3: Schéma bloc d'un capteur actif.

Les effets physiques les plus rencontrés en instrumentation sont:

- **Effet thermoélectrique:** un circuit formé de deux conducteurs de nature chimique différente, dont les jonctions sont à des température T1 et T2, est le siège d'une force électromotrice d'origine thermique Fe(T1,T2).
- **Effet piézo-électrique:** l'application d'une contrainte mécanique à certains matériaux dits piézo-électrique (le quartz par exemple) entraine l'apparition d'une déformation et d'une même charge électrique de signe différent sur les faces opposées.
- **Effet d'induction électromagnétique:** la variation du flux d'induction magnétique dans un circuit électrique induit une tension électrique (détection de passage d'un objet métallique).
- **Effet photo-électrique:** la libération de charges électriques dans la matière sous l'influence d'un rayonnement lumineux ou plus généralement d'une onde électromagnétique.
- **Effet de hall:** un champ magnétique B et un courant électrique I créent dans le matériau une différence de potentiel U_H.
- **Effet photovoltaïque:** des électrons et des trous sont libérés au voisinage d'une jonction PN illuminée, leur déplacement modifie la tension à ses bornes.

<table>
<tr><th>Grandeur physique mesurée</th><th>Effet utilisé</th><th>Grandeur de sortie</th></tr>
<tr><td>Température</td><td>Thermoélectricité</td><td>Tension</td></tr>
<tr><td rowspan="3">Flux de rayonnement optique</td><td>photoémission</td><td>Courant</td></tr>
<tr><td>Effet photovoltaïque</td><td>Tension</td></tr>
<tr><td>Effet photo-électrique</td><td>Tension</td></tr>
<tr><td>Force</td><td rowspan="2">Piézo-électricité</td><td rowspan="2">Charge électrique</td></tr>
<tr><td>Pression</td></tr>
<tr><td>Accélération</td><td rowspan="2">Induction électromagnétique</td><td rowspan="2">Tension</td></tr>
<tr><td>vitesse</td></tr>
<tr><td>Position (Aimant)</td><td rowspan="2">Effet hall</td><td rowspan="2">Tension</td></tr>
<tr><td>Courant</td></tr>
</table>

Tableau I.1: Grandeurs d'entrée et de sortie et effet utilisé pour les capteurs actifs.

I .3.2. Capteurs passifs

Il s'agit généralement d'impédance dont l'un des paramètres déterminants est sensible à la grandeur mesurée. La variation d'impédance résulte:

- Soit d'une variation de dimension du capteur, c'est le principe de fonctionnement d'un grand nombre de capteur de position, potentiomètre, inductance à noyaux mobile, condensateur à armature mobile.
- Soit d'une déformation résultant de force ou de grandeur s'y ramenant, pression accélération (Armature de condensateur soumise à une différence de pression, jauge d'extensomètre liée à une structure déformable).

L'impédance d'un capteur passifs et ses variations ne sont mesurables qu'en intégrant le capteur dans un circuit de conditionnement électronique qui permet son alimentation et l'adaptation du signal à la sortie (Figure I.4). Le Tableau I.2 résume les types des matériaux utilisés et la caractéristique électrique des capteurs passifs.

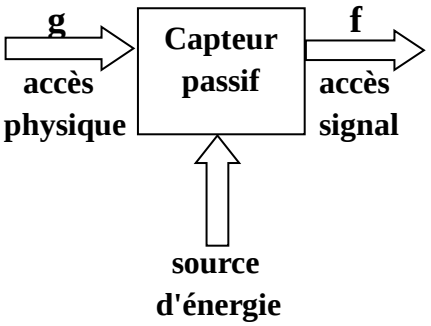

Figure I.4: Schéma bloc d'un capteur passif.

Grandeur mesurée	Caractéristique électrique	Type de matériau utilisé
Température	Résistivité	Métaux : platine, nickel, cuivre...
Très basse température	Constante diélectrique	Verre
Flux de rayonnement optique	Résistivité	Semi-conducteur
Déformation	Résistivité	Alliage de nickel, silicium dopé
	Perméabilité	Alliage ferromagnétique
Position (aimant)	Résistivité	Matériaux magnéto résistants : bismuth, antimoine d'indium
Humidité	Résistivité	Chlorure de lithium

Tableau I.2 : Type de matériau utilisé et caractéristique électrique des capteurs passifs.

I .4. Performances d'un capteur: caractéristiques métrologiques

De manière à classer les capteurs en fonction de leurs performances, on est amené à définir des paramètres qui permettent de les sélectionner en fonction de l'application.

Chaque capteur présente certaines caractéristiques métrologiques qui définissent ses limites d'utilisation et de précision. Ces limites dépendent non seulement du mesurande, mais aussi des grandeurs d'influence qui viennent perturber l'élément de mesure .

I.4.1. Caractéristique de transfert ou d'entrée-sortie

Elle donne la relation dévolution de la grandeur de sortie en fonction de la grandeur d'entrée. Elle est donnée classiquement par une courbe en régime permanent. il s'agit d'une courbe en régime permanent qui ne donne pas d'informations sur les caractéristiques transitoires du capteur.

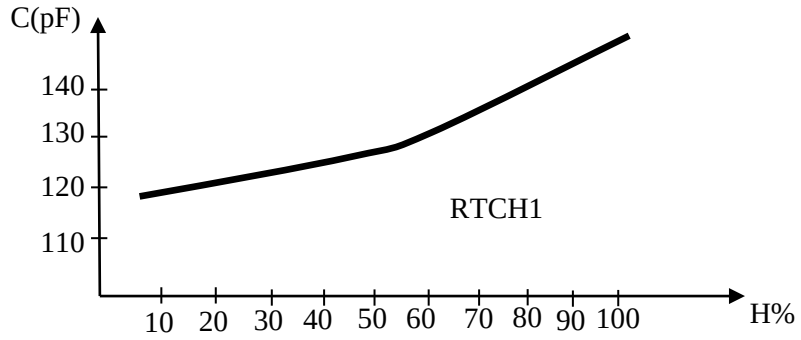

Figure I.5: Exemple de caractéristique d'un capteur d'humidité du type capacitif.

I.4.2. Etendue de mesure

L'étendue de mesure d'un capteur correspond à l'intervalle entre la valeur minimale et la valeur maximale du mesurande. Ces deux valeurs sont respectivement appelées portée minimale et portée maximale. Elle est exprimée dans l'unité de mesure du mesurande.

De l'étendue de mesure, on peut obtenir l'étendue d'échelle qui représente l'écart entre la portée minimale et maximale de l'étendue de mesure.

I.4.3. Domaine de fonctionnement

Ils définissent les zones dans lesquelles les caractéristiques du capteur sont assurées par rapport à des spécifications données. On peut classer cette zone en trois familles:

- Zone nominale d'emploi: Zone dans laquelle le mesurande peut évoluer sans modification des caractéristiques du capteur.
- Zone de non-détérioration: Valeurs limites des grandeurs influençant le capteur (mesurande, Température environnante, etc ...) sans que les caractéristiques du capteur ne soient modifiées après annulation de surcharges éventuelles.
- Zone de non-destruction: Elle définit les limites garantissant la non-destruction du capteur mais dans laquelle il peut y avoir des modifications permanentes des caractéristiques du capteur.

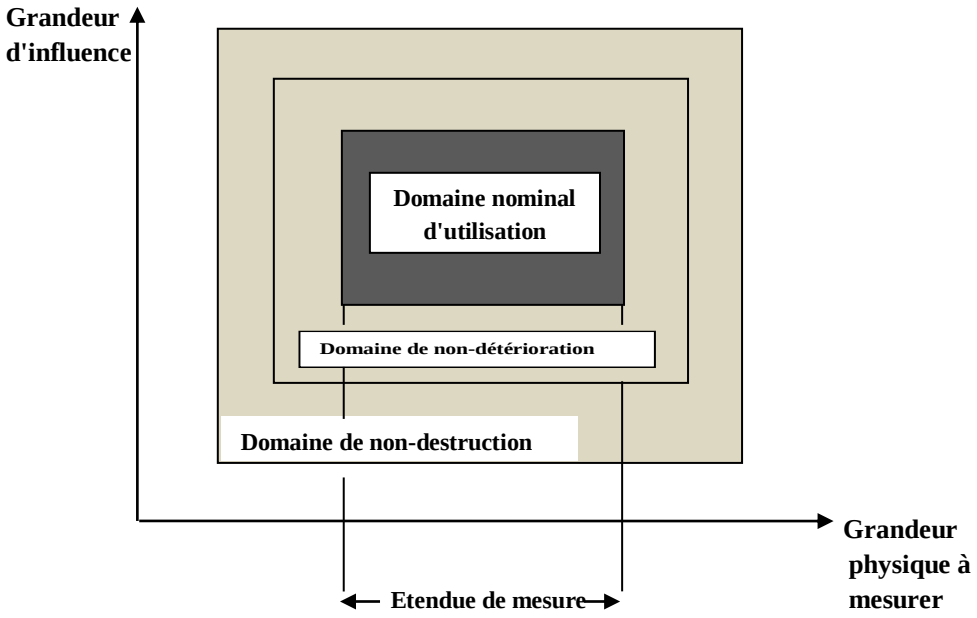

Figure I.6: Domaine de fonctionnement d'un capteur.

I.4.4. La sensibilité

La sensibilité d'un capteur représente le rapport de la variation du signal de sortie à la variation du signal d'entrée, pour une mesure donnée. C'est donc la pente de la courbe de réponse de ce capteur:

$$S = \frac{\Delta_{sortie}}{\Delta_{entrée}} \quad (1)$$

Si le capteur est linéaire, une seule valeur de sensibilité est nécessaire, car la pente de la courbe de la caractéristique entrée/sortie du capteur est constante. La caractéristique est alors une droite.

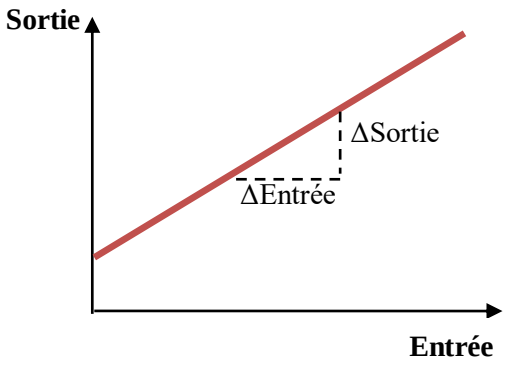

Figure I.7: Caractéristique linéaire.

Si le capteur est non-linéaire alors la sensibilité pour différentes mesures. La sensibilité est une indication de la pente locale de la caractéristique pour une mesure donnée (exemple: pente locale à la mesure e1 sur la Figure I.8).

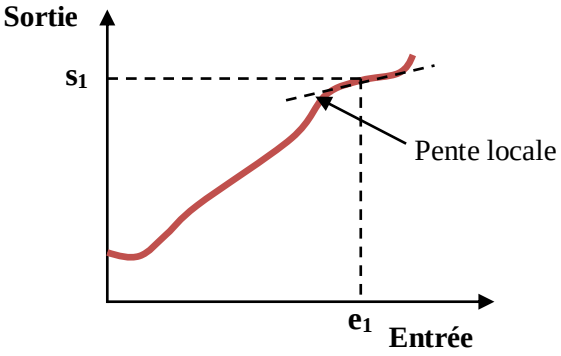

Figure I.8: Caractéristique non-linéaire (sensibilité = pente locale).

I.4.5. La linéarité

La linéarité est une caractéristique qui définit la constance de la sensibilité sur toute la plage de mesure.

L'équation décrivant la relation entre le signal d'entrée x et le signal de sortie y doit être de premier ordre (y = ax+b) pour que le capteur soit linéaire. Si le capteur n'est pas linéaire, la relation entrée/sortie peut être approximée par une équation du premier ordre, il faut accepter l'imprécision causée par cette approximation. L'écart de linéarité est exprimé par un pourcentage de l'étendue de mesure.

On parle aussi de domaine linéaire, la caractéristique est une portion de droite. Dans ce domaine, la variation de la grandeur de sortie est proportionnelle à la variation du mesurande [4].

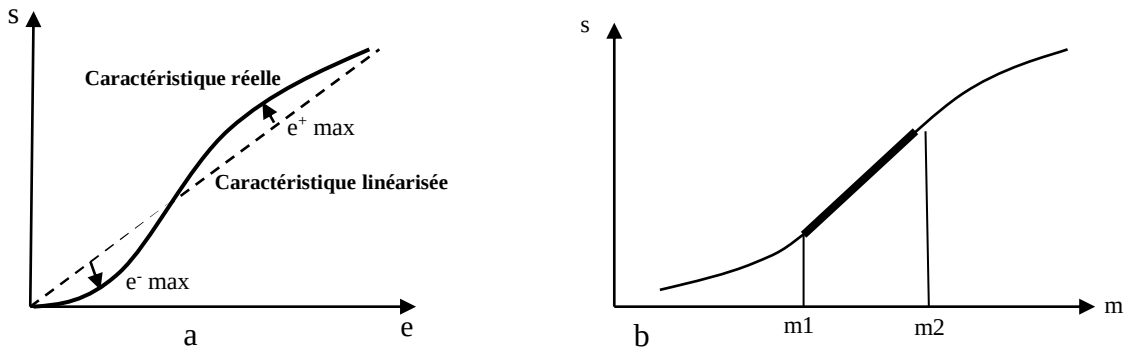

Figure I.9: Linéarisation de la caractéristique et domaine de linéarité.

I.5. Capteurs de Température

Sans doute la Température est une des grandeurs les plus importante dans le milieu industriel (génie chimique, industrie agro-alimentaire, analyse de fonctionnement: moteurs, navettes spatiales, gestion de bains de peinture ...).

Elle est mesurée de façon indirecte, par le biais d'un autre principe physique. Il est donc essentiel de bien connaitre les principales techniques de mesure et les principes physiques qui les permettent.

I.5.1. Types de capteurs de Température

Il est possible de mesurer la température de plusieurs façons différentes qui se distinguent par le cout des équipements et la précision ainsi que le temps de réponse.

Les types les plus courants de capteurs sont RTD, les thermistances et les thermocouples.

I.5.1.1. Principe de fonctionnement des sonde RTD

Les RTD (en Anglais: Resistance-Temperature Detector) fonctionnent sur le principe des variations de résistances électrique des métaux purs et se caractérisent par une modification positive linéaire de la résistance en fonction de la température .

Concrètement, une fois chauffée, la résistance du métal augmente et inversement une fois refroidie, elle diminue.

Les éléments types utilisés pour les RTD incluent le nickel (Ni) et le cuivre (Cu) mais le platine (Pt) est de loin le plus courant, en raison de l'étendue de sa gamme de températures, de sa précision et de sa stabilité.

Faire passer le courant à travers une sonde RTD génère une tension à travers la sonde RTD. En mesurant cette tension, vous pouvez déterminer sa résistance et ainsi, sa température.

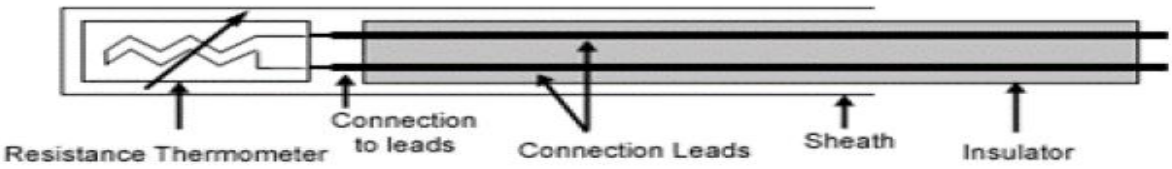

Figure I.10: Architecture physique d'un RTD.

I.5.1.2. Principe de fonctionnement d'une thermistance

Les thermistances, comme les capteurs de température à résistance (RTD), sont des conducteurs thermosensibles dont la résistance varie avec la température.

Les thermistances sont constituées d'un matériau semi-conducteur d'oxyde métallique encapsulé dans une petite bille d'époxy ou de verre.

En outre, les thermistances présentent généralement des valeurs de résistance nominale plus élevées que les RTD (de 2000 à 10000 Ω) et peuvent être utilisées pour de plus faibles courants.

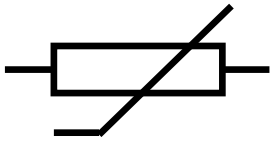

Figure I.11: Symbole communément utilisé pour représenter une thermistance.

I.5.1.3. Principe de fonctionnement d'un Thermocouple

Un conducteur génère une tension lorsqu'il est soumis à une variation de température, cette tension thermoélectrique est appelée tension Seebeck.

La mesure de cette tension nécessite l'utilisation d'un second matériau conducteur générant une tension différente pour une même variation de température (sinon la tension générée par le deuxième conducteur qui effectue la mesure annule tout simplement celle du premier conducteur).

En s'appuyant sur le principe de Seebeck, il est clair que les thermocouples ne peuvent mesurer que des différences de température entre le point de référence (soudure froide) et le point de mesure (soudure chaude). Ceci nécessite que la température de référence soit connue.

I.6. Capteur de pression

La pression constitue une variable essentielle pour l'étude métrologique d'un milieu environnant qui peut être soit un gaz soit un fluide.

La mesure de cette variable est réalisée à l'aide d'un capteur de pression, dispositifs capable d'associer à la grandeur mesurée un signal électrique reconnaissable appelé "réponse".

Le capteur de pression comme étant un système constitué de deux partie de traitement de l'information par l'intermédiaire d'un circuit électronique que l'on peut appeler "Circuit électronique de traitement" ou encore "Circuit convertisseur". La partie détection est constituée d'un "corps d'épreuve" et d'un "transducteur" qui transforme la déformation de ce corps d'épreuve en une grandeur physique, ensuite à une grandeur électrique [5].

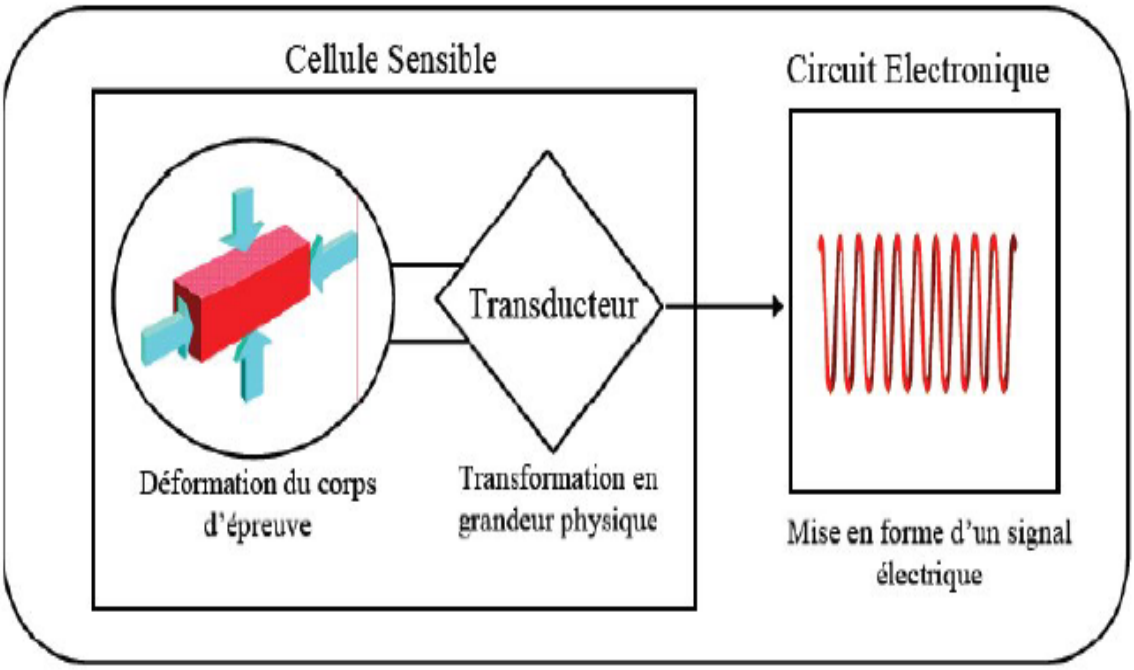

Figure I.12: Schéma de principe du capteur de pression [6].

I.6.1. Types de capteurs de pression

I.6.1.1.Capteur de pression capacitif

Les capteurs étudiés comportent une armature plane fixe et une armature plane déformable. En l'absence de pression différentielle entre les deux faces de la membrane, les armatures sont parallèles [7].

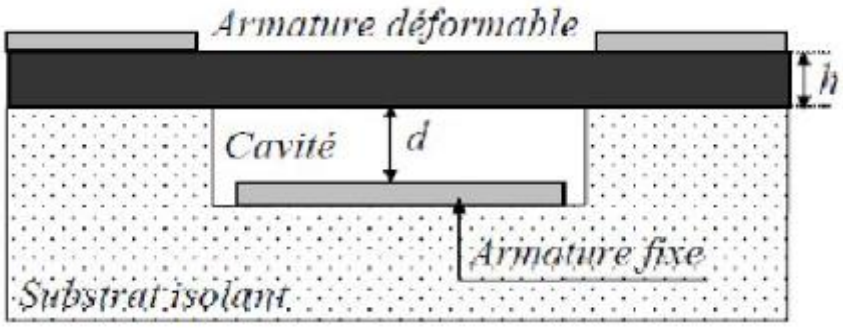

Figure I.13: Capteur de pression capacitif au repos [7].

En l'absence de pression appliquée, la capacité intrinsèque de la cellule est celle d'un condensateur plan. Elle est donc définie par :

$$C_i(0) = \varepsilon \frac{A}{d} \tag{2}$$

Où: ε_0 représente la permittivité électrique du vide, **A** est l'aire de la surface des électrodes en regard c'est-à-dire l'aire de l'armature fixe, et **d** est la distance entre les armatures.

Pour simplifier l'écriture, la capacité définie à pression nulle $C_i(0)$ sera notée Cio dans ce qui suit:

$$\mathrm{Ci(p)} = \varepsilon \iint \frac{dA}{d - w(x,y,p)} \tag{3}$$

Où dA est un élément de surface de l'armature fixe et w(x,y,p) représente la déflexion de la membrane en fonction de la pression au point de coordonnées (x,y); l'origine du repère étant définie au centre géométrique de la face inférieure de la membrane à P = 0.

Autrement dit, lorsque la pression extérieure augmente, la distance inter-armatures diminue, et par suite, la capacité augmente [8].

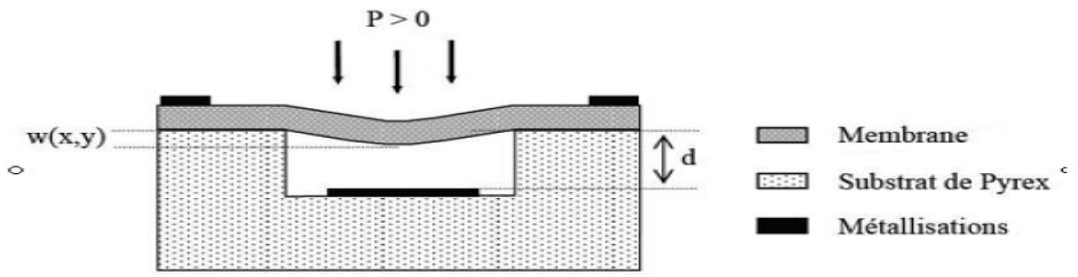

Figure I.14: Capteur de pression en fonctionnement [6].

I.6.1.2. Capteur de pression piezorésistif

Un capteur de pression piezorésistif à membrane est un capteur composite. Une membrane en silicium oxydée de quelques millimètres de côté et quelques microns d'épaisseur recouverte d'oxyde constitue le corps d'épreuve qui se déforme sous l'effet d'une pression appliquée. Des jauges piezorésistives en surface constitue le corps d'épreuve forme un conditionnement passif. La déformation de ces jauges se transforme en variation de résistance.

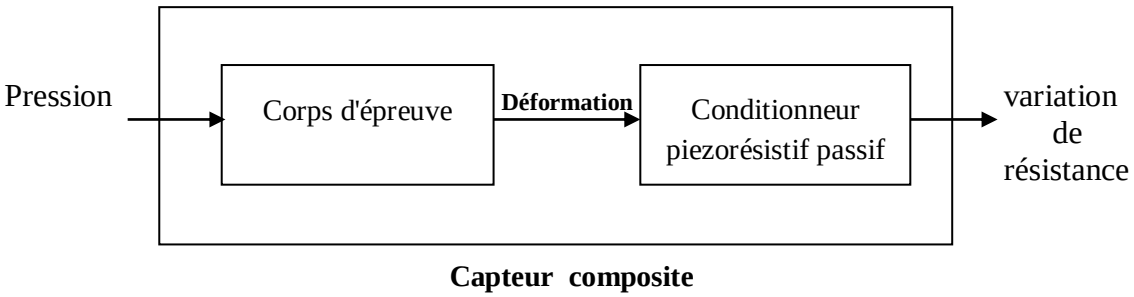

Figure I.15: Structure d'un capteur de pression piezorésistif à membrane.

Pour avoir une sensibilité élevée le conditionneur du capteur peut être constitue de deux jauges longitudinales et deux jauges transversales disposées en bordures de membrane et interconnectées en pont de Wheatstone par des pistes d'aluminium [9].

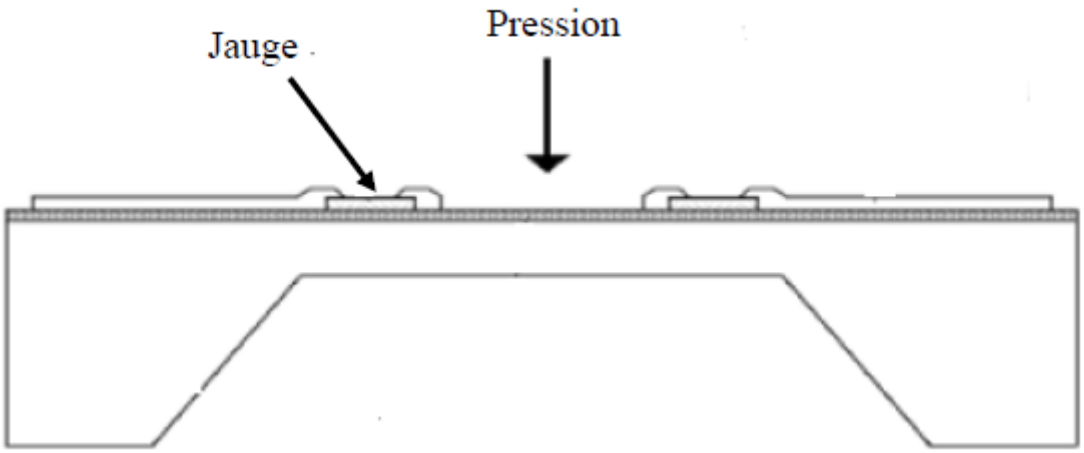

Figure I.16: Schéma en coupe d'un capteur piezorésistif à membrane de type SOI.

I.7. Capteur d'humidité

L'humidité exprime la présence d'un mélange d'air sec et de vapeur d'eau dans l'air ambiant. Quand on parle de mesure d'humidité, on fait allusion au taux d'humidité exprime en % qui correspond à l'humidité relative. Le taux d'humidité est étroitement lié à d'autres grandeurs physiques, telles que la température et la pression [10].

I.7.1. Types de capteurs d'humidité

Le taux d'humidité, avec la température comptent parmi les grandeurs physiques les plus fréquemment mesurée, par l'influence de ces paramètres sur le fonctionnement des systèmes, à ce jour, on distingue les capteurs capacitifs, résistifs, hygrométriques, gravimétriques et optiques [11].

I.7.1.1. Capteurs hygrométriques

Le principe de transduction repose sur la déformation d'un solide: membrane, cheveu... après absorption d'humidité. L'avantage de cette technique de transduction est qu'elle n'est pas sujette à la dérive en température, le matériau est généralement fiable sur une longue durée [12].

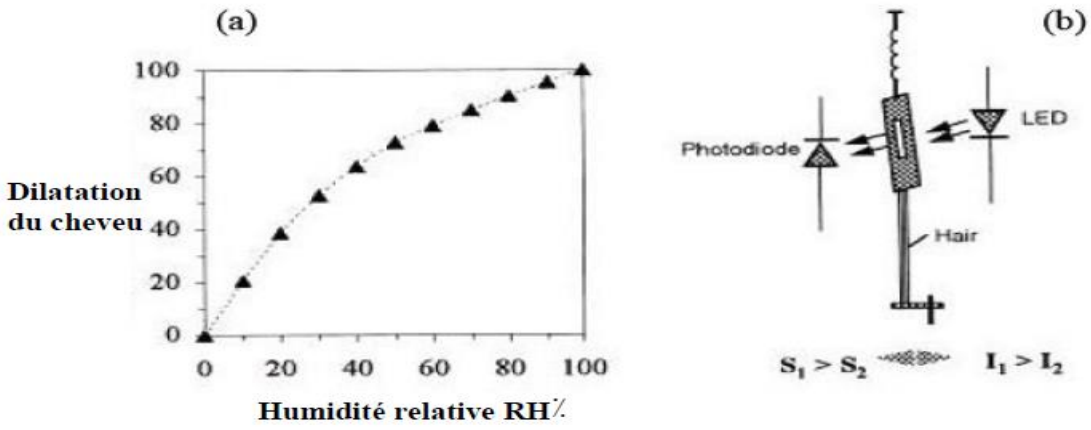

Figure I.17: Capteur d'humidité hygrométrique.

I.7.1.2. Capteur résistif

Les capteurs résistifs sont basés sur le changement d'impédance d'une couche sensible après absorption d'humidité. Trois types de matériaux sont généralement utilisés: céramiques, polymères et électrolytes.

Un exemple de capteur d'humidité résistif Figure (I.18): le matériau utilisé est un trioxyde de tungstène, préparé par sol-gel et déposé sur les résistances dédiées à la transduction, le capteur comporte des résistances chauffantes pour l'évacuation de l'humidité dans la couche sensible [13].

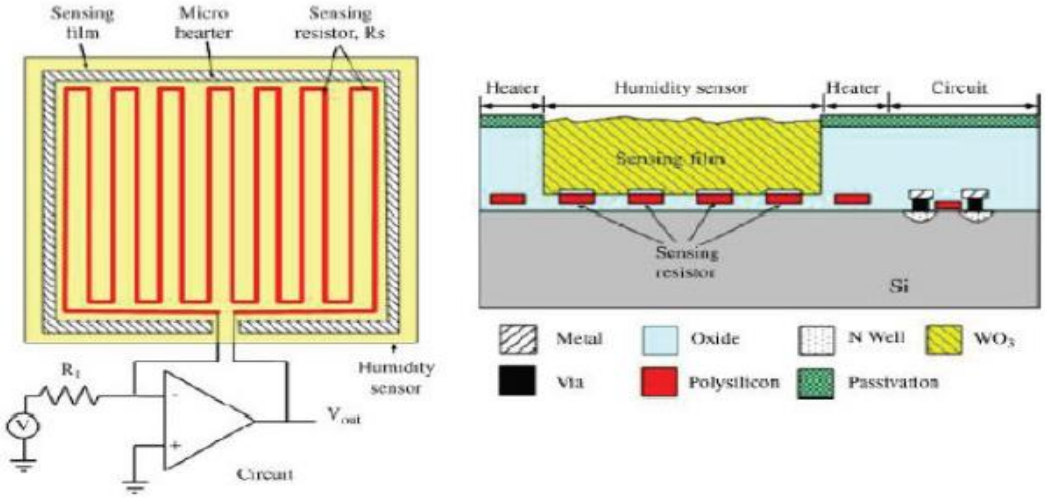

Figure I.18: Capteur résistif avec un film sensible de WO_3, et le principe du circuit d'instrumentation avec amplificateur opérationnel.

I.6.1.3. Capteurs capacitifs

Les capteurs d'humidité capacitifs sont des condensateurs à deux bornes. La valeur de la capacité augmente quand des molécules d'eau sont absorbées dans son polymère diélectrique actif. Le principe de ce capteur est basé sur la variation de la capacité d'une couche diélectrique exposée à un changement du taux d'humidité. Il représente la majeure partie des systèmes dotés d'un capteur d'humidité [14].

La valeur de la capacité peut se mettre sous la forme suivante :

$$C = \varepsilon \frac{S}{e} \tag{4}$$

ou C : capacité en Farad;

ε : permittivité relative du diélectrique;

S : surface des armatures;

e : épaisseur du diélectrique.

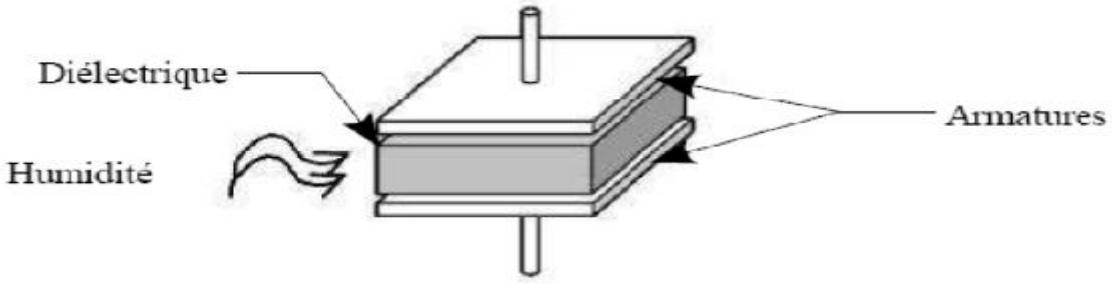

Figure I.19: Capteur d'humidité capacitif.

I.8. Capteur d'ensoleillement

I.8.1. Photorésistance

La photorésistance (également appelée résistance photo-dépendante ou cellule photoconductrice) est un composant électronique dont la résistivité varie en fonction de la quantité de lumière incidente: plus elle est éclairée, plus sa résistivité baisse.

Figure I.20: Une Photorésistance.

I.8.1.1. Principe de fonctionnement de la photorésistance

La photorésistance est composée d'un semi-conducteur à haute résistivité. Si la lumière incidente est de fréquence suffisamment élevée (donc d'une longueur d'onde inférieure à la longueur d'onde seuil), elle transporte une énergie importante. Au-delà d'un certain niveau propre au matériau, les photons absorbés par le semi-conducteur donneront aux électrons liés assez d'énergie pour passer de la bande de valence à la bande de conduction. La compréhension de ce phénomène entre dans le cadre de la théorie des bandes. Les électrons libres et les trous ainsi produits abaissent la résistance du matériau. Lorsque le photon incident est suffisamment énergétique, la production des paires électron-trou est d'autant plus importante que le flux lumineux est intense. La résistance évolue donc comme l'inverse de l'éclairement. Cette relation peut être modélisée par la relation suivante:

$$R(L) = R_0 \, L^{-k} \qquad (5)$$

Les matériaux utilisés dans les photorésistances sont le plus souvent des composés des colonnes II-VI de la classification périodique des éléments. Pour une utilisation dans le domaine visible et à faible coût, on utilise le plus souvent le sulfure de cadmium (CdS) ou le séléniure de cadmium (CdSe). Pour des utilisations dans l'infrarouge on utilise le sulfure de plomb (PbS) [15].

I.9. Conclusion

Dans ce chapitre nous avons vu les concepts généraux liés aux capteurs, avant de représenter les types et les différentes caractéristiques d'un capteur.

Nous avons ensuite mettre la lumière sur les capteurs de température, pression, humidité et ensoleillement, ces quatre capteurs que nous avons choisis par la suite dans notre réalisation pour la conception d'une carte météo.

Dans le chapitre qui suit on va mettre le point sur la présentation de la carte de commande à base de l'Arduino UNO et ses caractéristiques.

Chapitre II

Présentation d'Arduino

II.1. Introduction

Les cartes Arduino sont conçues pour réaliser des prototypes et des maquettes de cartes électroniques pour l'informatique embarquée, ces cartes permettent un accès simple et peu couteux à l'informatique embarquée. De plus, elles sont entièrement libres de droit, autant sur l'aspect du code source (Open Source) que sur l'aspect matériel (Open Hardware). Ainsi, il est possible de refaire sa propre carte Arduino dans le but de l'améliorer ou d'enlever des fonctionnalités inutiles au projet.

Le langage Arduino se distingue des langages utilisés dans l'industrie de l'informatique embarquée de par sa simplicité. En effet, beaucoup de librairies et de fonctionnalités de base occulte certains aspects de la programmation de logiciel embarquée afin de gagner en simplicité. Cela en fait un langage parfait pour réaliser des prototypes ou des petites applications dans le cadre de hobby.

Les possibilités des cartes Arduino sont énormes, un grand nombre d'application ont déjà été réalisée et testées par bon nombre d'internautes. On retrouve par exemple diverse forme de robot (voir Figure II.1), contrôleur de moteur pas à pas par joystick avec Arduino. (voir Figure II.2) [16].

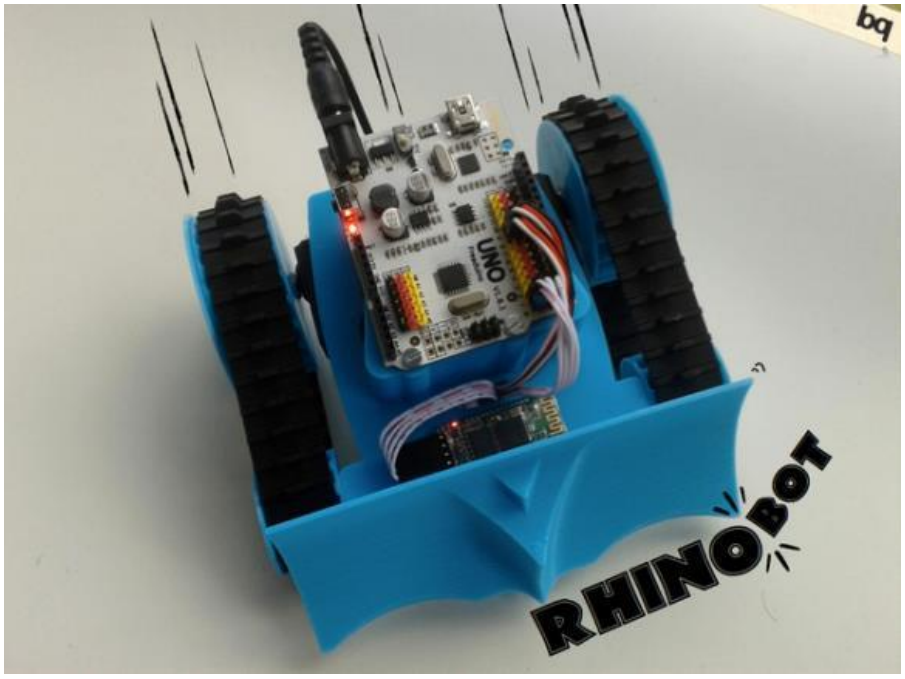

Figure II.1: Robot Arduino.

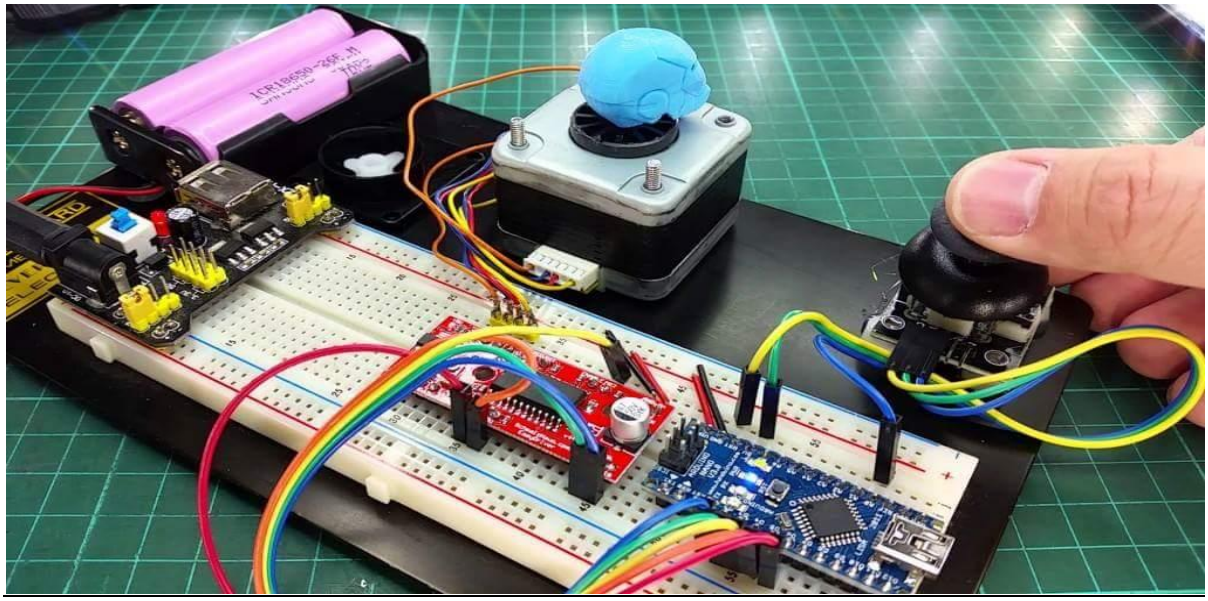

Figure II.2: Contrôleur de moteur pas à pas par joystick avec Arduino.

II.2. Historique

Arduino est un projet créé par une équipe de développeurs, composée de six individus: Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, David Mellis et Nicholas Zambetti. Cette équipe a créé le "système Arduino". C'est un outil qui va permettre aux débutants, amateurs ou professionnels de créer des systèmes électroniques plus ou moins complexes.

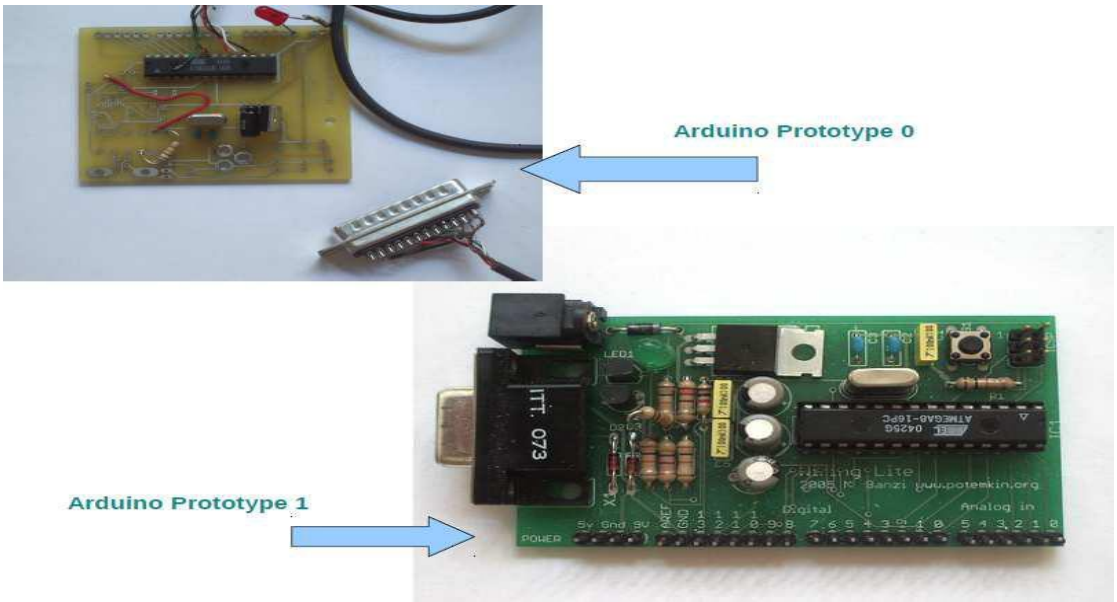

Figure II.3: Les prototypes d'Arduino.

II.3. Les avantages d'Arduino

Le système Arduino, nous donne la possibilité d'allier les performances de la programmation à celles de l'électronique. Plus précisément, nous allons programmer des systèmes électroniques. Les principaux avantages de l'électronique programmée sont :

- Simplifie grandement les schémas électroniques.
- Diminué le coût de la réalisation.
- La charge de travail à la conception d'une carte électronique.
- Environnement de programmation clair et simple.
- Multiplateforme : tourne sous Windows, Macintosh et Linux.
- Nombreuses bibliothèques disponibles avec diverses fonctions implémentées.
- Logiciel et matériel open source et extensible.
- Nombreux conseils, tutoriaux et exemples en ligne (forums, site perso, etc.).
- Existence de « shield » (boucliers en français).

II.4. Domaine d'utilisation et ces applications

Le système Arduino nous permet de réaliser un grand nombre de choses, qui ont une application dans tous les domaines, l'étendue de l'utilisation de l'Arduino est gigantesque. Pour vous donner quelques exemples, vous pouvez :

- Électronique industrielle et embarquée.
- contrôler les appareils domestiques.
- fabriquer votre propre robot.
- faire un jeu de lumières.
- communiquer avec l'ordinateur.
- télécommander un appareil mobile (modélisme).
- Physical computing: Au sens large, construire des systèmes physiques interactifs qui utilisent des logiciels et du matériel pouvant s'interfacer avec des capteurs et des actionneurs.
- Art / Spectacle.
- Hacker, Prototypage, Education, Etc [17].

II.5. Présentation de la carte

II.5.1. Qu'est ce qu'un microcontrôleur

Les cartes Arduino font partie de la famille des microcontrôleurs. Un microcontrôleur est une petite unité de calcul accompagné de mémoire, de ports d'entrée/sortie et de périphériques permettant d'interagir avec son environnement. Parmi les périphériques, ont recense généralement des Timers, des convertisseurs analogique-numérique, des liaisons Séries, ... etc. On peut comparer un micro contrôleurs à un ordinateur classique, mais avec un autre système d'exploitation et avec une puissance de calcul considérablement plus faible.

Les microcontrôleurs sont inévitables dans les domaines de l'informatique embarquée, de l'automatique et de l'informatique industrielle. Ils permettent de réduire le nombre de composant et de simplifier la création de cartes électroniques logiques [16].

II.5.2. Microcontrôleur ATMEL ATMega328

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

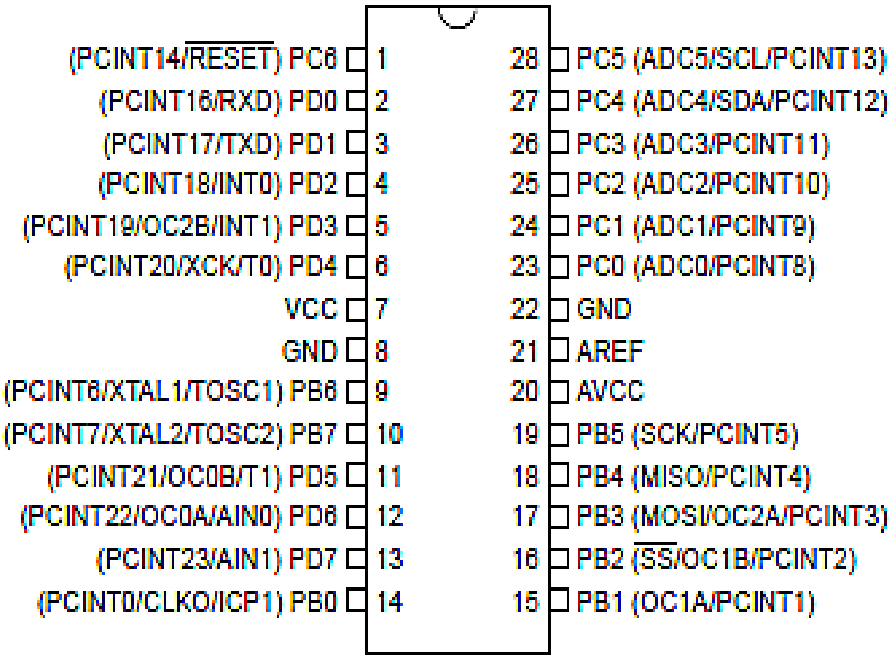

Figure II.4: Datasheet ATMega328.

Les principales caractéristiques de ATMega328 sont :

- **FLASH** = mémoire programme de 32Ko.
- **SRAM** = données (volatiles) 2Ko.
- **EEPROM** = données (non volatiles) 1Ko.
- **Digital I/O (entrées-sorties Tout Ou Rien)** = 3 ports PortB, PortC, PortD (soit 23 broches en tout I/O).
- **Timers/Counters** : Timer0 et Timer2 (comptage 8 bits), Timer1 (comptage 16 bits) Chaque timer peut être utilise pour générer deux signaux PWM. (6 broches OCxA/OCxB).
- **Plusieurs broches multi-fonctions:** certaines broches peuvent avoir plusieurs fonctions différentes choisies par programmation.
- **PWM** = 6 broches OC0A(PD6), OC0B(PD5), 0C1A(PB1), OC1B(PB3), OC2A(PB3), OC2B(PD3)
- **Analog to Digital Converter** (resolution 10 bits) = 6 entrees multiplexees ADC0(PC0) à ADC5(PC5).
- **Gestion bus I2C** (TWI Two Wire Interface) = le bus est exploite via les broches SDA(PC5)/SCL(PC4).
- **Port série (USART)** = émission/réception serie via les broches **TXD(PD1)/RXD(PD0)**
- **Comparateur Analogique** = broches AIN0(PD6) et AIN1 (PD7) peut déclencher interruption
- **Watchdog Timer programmable.**
- **Gestion d'interruptions (24 sources possibles (**cf interrupt vectors)**) :** en résumé
    - Interruptions liées aux entrées INT0 (PD2) et INT1 (PD3).
    - Interruptions sur changement d'état des broches PCINT0 a 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 serie USART.
    - Interruption du bus TWI (I2C) [18].

II.5.3. Caractéristiques techniques de l'Arduino UNO

Un des modèles les plus répandu de carte Arduino est l'Arduino UNO (voir Fig II.5). C'est la première version stable de carte Arduino. Elle possède toutes les fonctionnalités d'un microcontrôleur classique en plus de sa simplicité d'utilisation. Elle utilise une puce ATmega328P (1) cadencée à 16 Mhz. Elle possède 32 ko de mémoire flash destinée à recevoir le programme, 2 ko de SRAM (mémoire vive) et 1 ko d'EEPROM (mémoire morte destinée aux données). Elle offre 14 pins (broches) d'entrée/sortie numérique (données acceptée 0 ou 1) (2) dont 6 pouvant générer des PWM (Pulse Width Modulation, détaillé plus tard). Elle permet aussi de mesurer des grandeurs analogiques grâce à ces 6 entrées analogiques (3). Chaque broche est capable de délivré un courant de 40 mA pour une tension de 5 V.

La carte Arduino peut aussi s'alimenter et communiquer avec un ordinateur grâce à son port USB (4). On peut aussi l'alimenter avec une alimentation comprise en 7V et 12V grâce à sa connecteur Power Jack (5) [16].

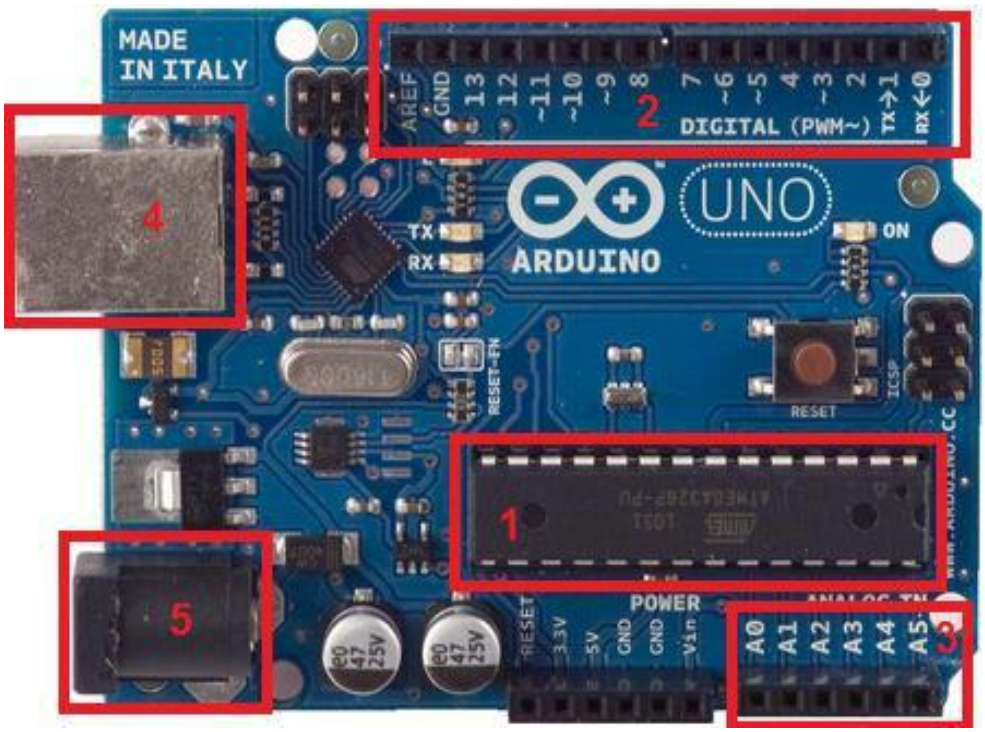

Figure II.5: La carte Arduino UNO.

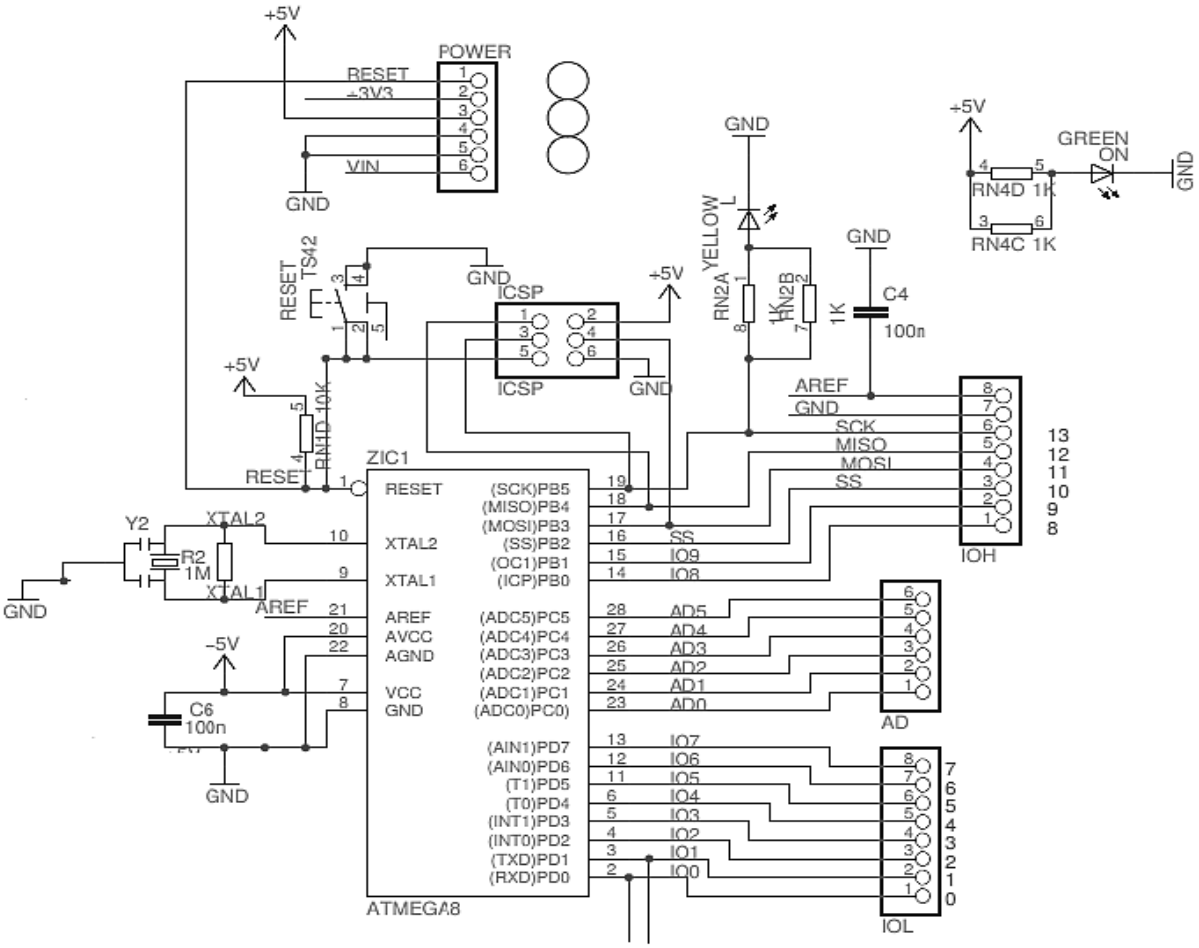

Figure II.6: Schéma simplifié de la carte Arduino UNO.

Les différents éléments de la carte de commande sont regroupés dans le Tableau II.1.

Tension de fonctionnement	5 V
Tension d'alimentation (recommandée)	7-16 V
Tension d'alimentation (limites)	6-20 V
Broches E/S numériques	14(dont 6 disposent d'une sortie PWM).
Broches d'entrées analogiques	6 (utilisables en broches E/S numériques).
Intensité maxi disponible par broche E/S (5 V)	40 mA (attention : 200 mA cumulé pour l'ensemble des broches E/S).
Intensité maxi disponible pour la sortie 3.3 V	50 mA
Intensité maxi disponible pour la sortie 5 V	Fonction de l'alimentation utilisée - 500 mA max si port USB utilisée seul

Mémoire programme flash	32 KB (ATmega328) dont 0.5 KB sont utilisés par le boot loader (programme de base préprogrammé conçu pour établir la communication entre l' ATmega et le logiciel Arduino).
Mémoire SRAM (mémoire volatile)	2 KB (ATmega).
Mémoire EEPROM (mémoire non volatile)	1 KB (ATmega).
Vitesse d'horloge	16 Mhz

Tableau II.1: Différents éléments de la carte de commande.

II.6. Présentation des shields

Pour la plupart des projets, il est souvent nécessaire d'ajouter des fonctionnalités aux cartes Arduino. Plutôt que d'ajouter soit même des composants extérieurs (sur une platine d'essai, circuit imprimé, etc.), il est possible d'ajouter des shields.

Un shield est une carte que l'on connecte directement sur la carte Arduino qui a pour but d'ajouter des composants sur la carte. Ces shields viennent généralement avec une librairie permettant de les contrôler. On retrouve par exemple, des shields Ethernet, de contrôle de moteur, lecteur de carte SD, etc.

Le principal avantage de ces shields est leurs simplicités d'utilisation. Il suffit des les emboiter sur la carte Arduino pour les connecter, les circuits électronique et les logiciels sont déjà faits et on peut en empiler plusieurs. C'est un atout majeur pour ces cartes pour pouvoir tester facilement de nouvelles fonctionnalités. Cependant il faut bien garder à l'esprit que les shields ont un prix. Suivant les composants qu'ils apportent, leurs prix peuvent aller de 2 à 100$.

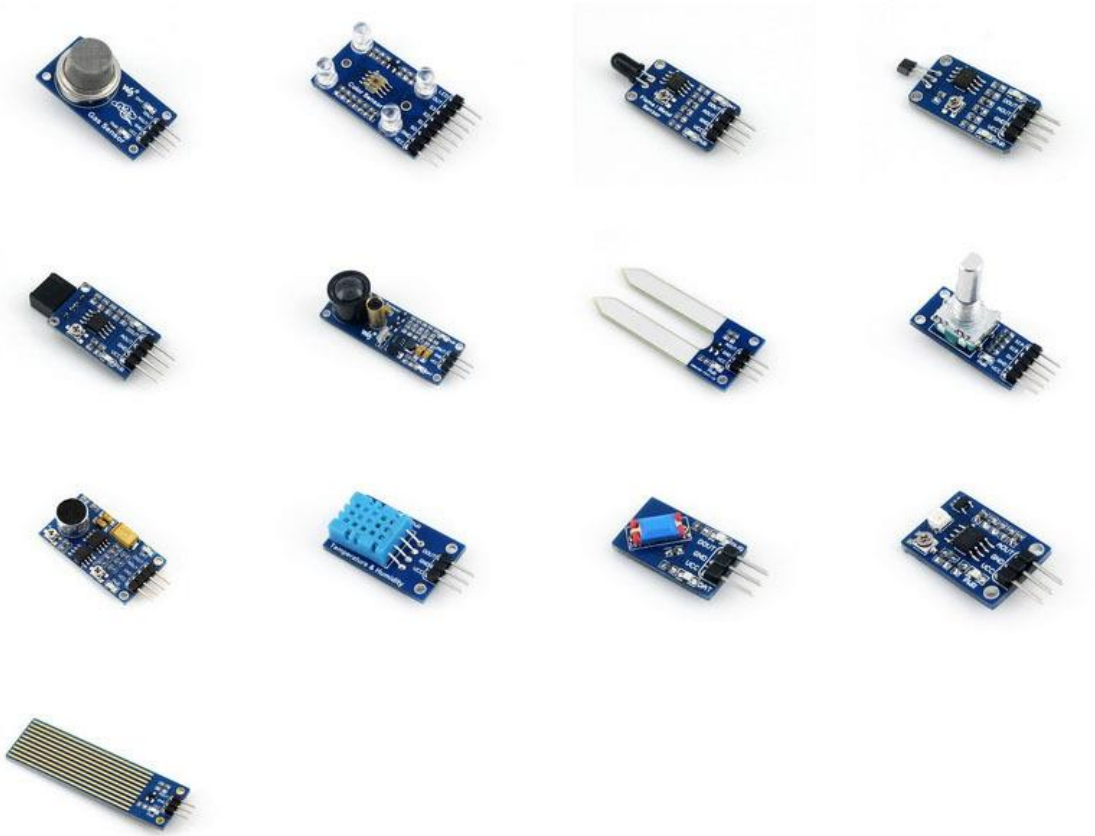

Figure II.7: Liste des différents Shields.

II.7. Présentation du logiciel

II.7.1. IDE Arduino

Un IDE (environnement de développement) libre et gratuit est distribué sur le site d'Arduino (compatible Windows, Linux et Mac) à l'adresse http://arduino.cc/en/main/software. D'autres alternatives existent pour développer l'Arduino (extensions pour CodeBlocks, Visual Studio, Eclipse, XCode, etc.) mais nous n'aborderons dans ce chapitre que l'IDE officiel.

L'interface de l'IDE Arduino est plutôt simple (Figure II.8), il offre une interface minimale et épurée pour développer un programme sur les cartes Arduino. Il est doté d'un éditeur de code avec coloration syntaxique (1) et d'une barre d'outils rapide (2). Ce sont les deux éléments les plus importants de l'interface, c'est ceux que l'on utilise le plus souvent. On retrouve aussi une barre de menus (3) plus classique qui est utilisé pour accéder aux

fonctions avancées de l'IDE. Enfin, une console (4) affichant les résultats de la compilation du code source, des opérations sur la carte, etc.

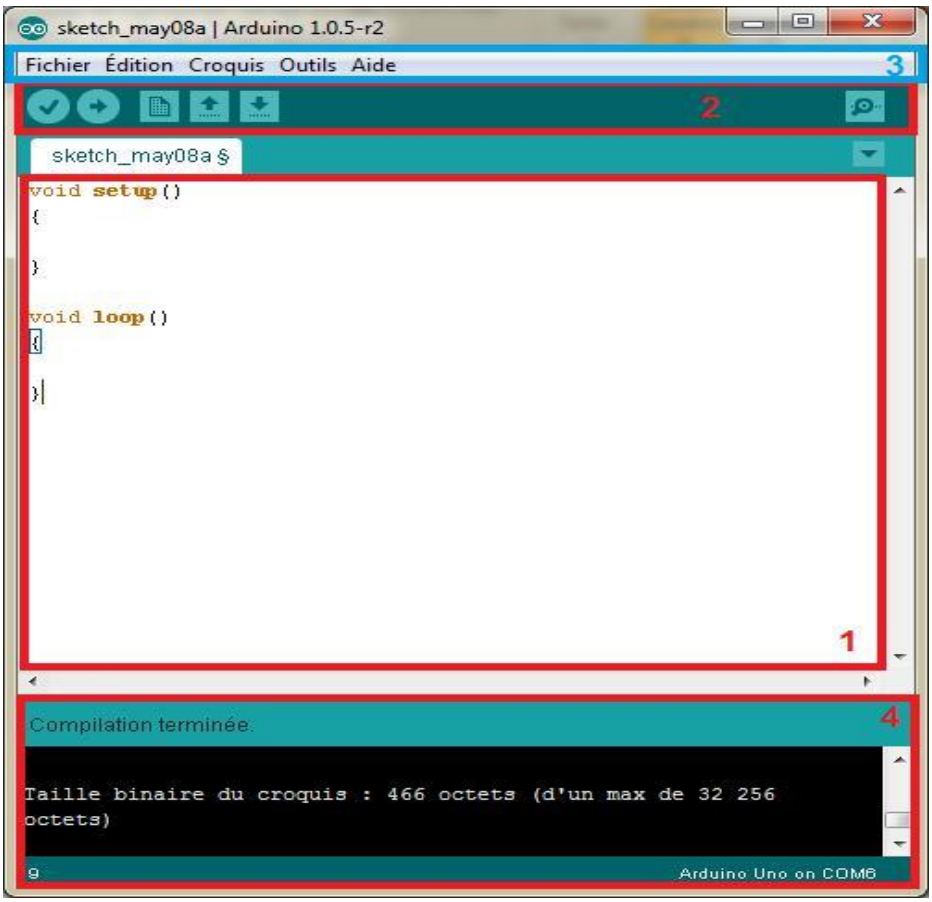

Figure II.8: IDE Arduino.

II.7.1.1. Les boutons

Voyons à présent à quoi servent les boutons numérotés en rouge.

Figure II.9: Présentation des boutons.

- Bouton 1 (verify): Ce bouton permet de vérifier le programme, il actionne un module qui cherche les erreurs dans votre programme.
- Bouton 2 (upload): Compiler et envoyer le programme vers la carte.
- Bouton 3 (new): Créer un nouveau fichier.
- Bouton 4 (open): Charger un programme existant.

- Bouton 5 (save): Sauvegarder le programme en cours.
- Bouton 6 (serial monitor): Permet d'accéder au port série (en RX/TX).

II.7.2. Langage Arduino

Le langage Arduino est inspiré de plusieurs langages. On retrouve notamment des similarités avec le C, le C++, le Java et le Processing. Le langage impose une structure particulière typique de l'informatique embarquée. La fonction setup (voir Fig.10) contiendra toutes les opérations nécessaires à la configuration de la carte (directions des entrées sorties, débits de communications série, etc.). La fonction loop, elle est exécutée en boucle après l'exécution de la fonction *setup*. Elle continuera de boucler tant que la carte n'est pas mise hors tension, redémarrée (par le bouton ***reset***). Cette boucle est absolument nécessaire sur les microcontrôleurs étant donné qu'il n'on pas de système d'exploitation. En effet, si l'on omettait cette boucle, à la fin du code produit, il sera impossible de reprendre la main sur la carte Arduino qui exécuterait alors du code aléatoire.

```
void setup()          //fonction d'initialisation de la carte
{
                      //contenu de l'initialisation
}

void loop()           //fonction principale, elle se répète
(s'exécute) à l'infini
{
                      //contenu de votre programme
}
```

Figure II.10: Un code minimal.

Au niveau de la syntaxe, on retrouve des similarités avec les langages précédemment cités. La déclaration des variables se fait généralement dans l'espace global (de façon à partager les variables les plus importantes entre les deux fonctions principales). On retrouve les types de base suivant (voir Tableau II.2):

Nom	Contenu	Taille (en octet)	Plage de valeurs
(unsigned) char	Entier ou caractère	1	(0->255) -128 -> 127
(unsigned) int	Entier	2	(0->65 535) -32 768 -> 32 767
(unsigned) long	Entier	4	(0 -> 4 294 967 295) -2 147 483 648 -> 2 147 483 647
float/double	Nombre à virgule flottante	4	-3,4028235E+38 -> 3,4028235E+38
String	Chaine de caractères (Objet)	variable	Aucune
boolean	Booléen	1	True / False

Tableau II.2: Les types de base.

Il existe d'autres types de base mais ils ne sont qu'un alias de ceux cités précédemment, la liste des types est disponible sur la page des références du site Arduino (http://arduino.cc/en/Reference/HomePage). La déclaration des variables suit cette syntaxe:

(const) <type> <nom>([<longueur du tableau>]) (= valeur);

Exemples

```
const int constante = 12 ;
float univers = 42.0 ;
char lettre = 'b' ;
String chaine = "Hello World ";
long tableau[12] ;
boolean vrai = true ;
```

On retrouve les opérateurs les plus courants pour les types de bases. Parmi eux, = (affectation), == (comparaison), != (différence), <, >, <=, >=, && (et logique), || (ou logique), ! (non logique). On retrouve aussi les opérateurs mathématiques (+, -, *, /, %) et les opérateurs logiques bit à bit (^ (XOR), & (et), | (ou), ~(non), << (décalage logique à gauche), >> (décalage logique à droite)).

Les structures de contrôle sont elles aussi similaires aux langages de références. On y retrouve toutes les structures de contrôle standard, conditions, boucle, switch, fonctions, etc. On peut aussi écrire des structures et des classes. Chaque structure de contrôle est suivie

d'un bloc d'instructions délimitées par des accolades. Voici une liste des structures de contrôles les plus utilisées :

Nom	Utilité	Syntaxe
If-else	Condition logique	*If (<valeur booléenne>) {* *<instruction>* *} else {* *<instruction>* *}*
If-else if - else	Condition logique multiples	*If (<valeur booléenne>) {* *<instruction>* *} else if (<valeur booléenne>) {* *<instruction>* *} else {* *<instruction>* *}*
Switch	Sélecteur	*Switch (<variable>) {* *case <valeur> :* *<instruction>* *break ;* *default :* *<instruction>* *}*
While	Boucle	*While (<valeur booléenne>) {* *<instruction>* *}*
For	Boucle itérative	*For (<initialisation> ; <valeur booléenne> ; <évolution>) {* *<instruction>* *}*

Tableau II.3: Les structures de contrôle.

Les fonctions, les structures et les classes se déclarent de la même façon qu'en C++. Elles se déclarent sous cette forme :

Structure:

struct <nom> {

<type> <nom du champ>

;}

Fonction :

<type de retour> <nom>(<paramètre>) {

<instruction> }

Classe :

class <nom>

{ public :

<attributs et champ publics>

private :

<attributs et champ privés>

protected :

<attribut et champ privés> ; }

II.7.3. Fonctionnalité de base

II.7.3.1. Les entrées/sorties

Le langage Arduino vient avec un nombre important de fonction de base permettant d'interagir avec son environnement. Les fonctions les plus utilisée sont les fonctions d'entrée/sorties. Ce sont elles qui permettent d'envoyer ou de mesurer une tension sur une des broches de la carte.

Dans un premier temps, avant d'effectuer une mesure ou d'envoyer une commande. Il est nécessaire de définir la direction des broches utilisées. Pour cela on fait appel à la fonction pinMode en lui donnant d'une part, la broche concernée, et d'autre part, la direction :

```
void setup() {
pinMode(1,OUTPUT) ; // Broche 1 en sortie
pinMode(2,INPUT) ; // Broche 2 en entrée
 }
```

Une fois cette configuration faite, on peut procéder à l'utilisation des broches. Toutes les broches sont capables d'écrire et de lire des données numériques (c'est-à-dire des 0 (0V) ou des 1 (5V)). Mais, certaines disposent de fonctionnalité supplémentaire.

Tout d'abord, toutes les cartes Arduino possèdent des entrées analogiques. Ce sont les broches A0-A1-A2 etc. Elles permettent de lire des tensions analogiques (comprise entre0 et 5V) et de le convertir en entier (compris entre 0 et 1023) proportionnellement à la tension mesurée. Certaines cartes Arduino possède des sorties analogique faisant l'opération inverse

(met une tension sur la broche proportionnellement à l'entier donné), mais ce n'est pas le cas pour l'Arduino UNO.

Pour pouvoir tout de même contrôler des composants autrement qu'en « tout ou rien » il est possible d'utiliser des broches PWM. Ce sont les broches annotés par un tilde ~ sur la carte. Les PWM (Pulse Width Modulation) sont utilisées pour synthétiser des signaux analogiques en modulant le temps passé à l'état 1 (5V). Le signal obtenu est représenté figure II.11. En utilisant une fréquence relativement élevée, les PWM permettent de commander certains composants comme si il recevait une tension analogique. Cela provient du fait que les composants utilisés dans l'électronique analogique, ne changes pas d'états instantanément. Par exemple, une ampoule à incandescence reste chaude et éclaire un court instant après avoir été éteinte. Ce phénomène est généralement invisible à l'œil nu. Grâce à elles, on pourra par exemple faire varier l'intensité d'une LED. La plupart des cartes Arduino utilisent des PWM cadencées à 490 Hz environ.

Toutes ces fonctionnalités sur les broches d'entrées sorties sont utilisables par le biais de quatre fonctions :

- **digitalRead(pin)**: mesure une donnée numérique sur une des broches, la broche en question doit être réglée en entrée.
- **digitalWrite(pin, value)**: écrit une donnée numérique sur une des broches, la broche concernée doit être réglée en sortie. Le paramètre *value* doit être égal à *HIGH* (état 1 soit 5V) ou *LOW* (état 0 soit 0V).
- **analogRead(pin)**: mesure une donnée analogique sur une des broches (compatible seulement), la broche doit être réglée sur entrée.
- **analogWrite(pin, value)**: écrit une donnée sous forme de PWM sur une des broches (compatible uniquement), la broche doit être réglée en sortie. Le paramètre value doit être compris dans l'intervalle *[0;255]*.

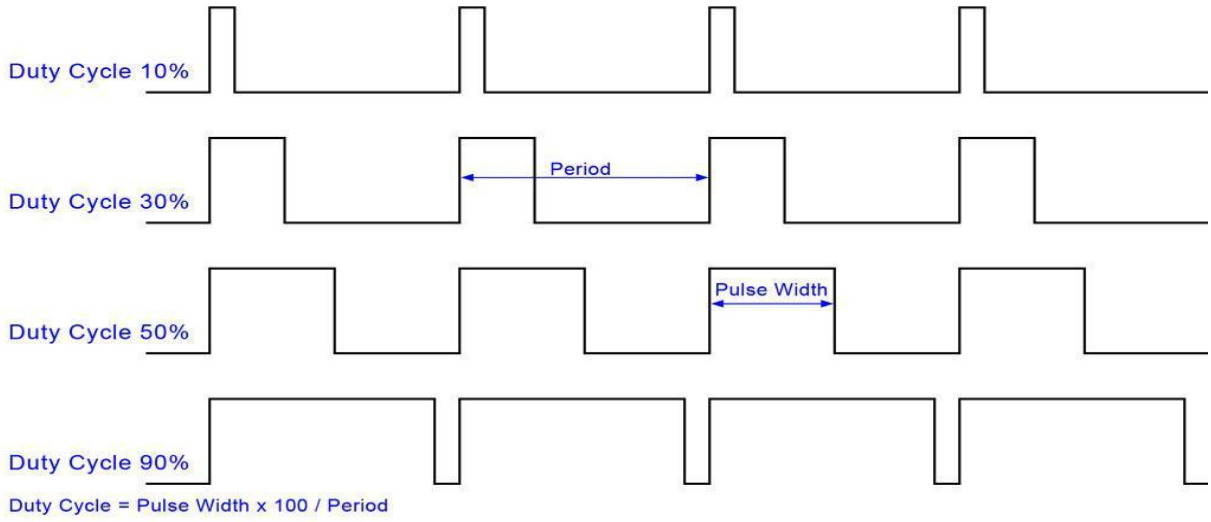

Figure II.11: Signal PWM.

II.7.3.2. Gestion du temps

Pour la plupart des applications de domotique, il est nécessaire de faire intervenir des intervalles de temps. Par exemple, pour gérer le temps d'appui sur un bouton ou pour faire une sonnerie qui se répète un certains nombre de fois. Le langage Arduino fournis quelques fonctions permettant de gérer le temps.

Il est possible d'insérer une pause dans son programme pendant un instant. Pour cela, on utilise les fonctions delay et delayMicroseconds qui insère une pause suivant le paramètre passé (en milliseconde pour l'un, en microseconde pour l'autre). Cependant ces fonctions bloquent le microcontrôleur, on ne peut alors plus effectuer aucune action.

En plus d'insérer une pause, il est possible de mesurer le temps. De la même manière que les fonctions de délai, on utilise les fonctions millis et micros qui donnent le nombre de milliseconde (respectivement microseconde) depuis le lancement de la carte. Attention, ces fonctions incrémente une variable (interne). Ces variables se remettent à zéro une fois le maximum atteint (overflow). La variable utilisée pour les millisecondes atteint sont maximum au bout de 49 jours et 17 heures et la variable utilisée pour les microsecondes au bout de 71 minutes et 34 secondes environ. Il faut donc faire attention lors de l'utilisation de ces fonctions pour des utilisations longues durées.

II.7.3.3. Les interruptions

Il est parfois nécessaire en informatique embarquée, d'attendre un événement externe (appui sur un bouton, données d'un capteur, etc.) pour effectuer une action. Pour ce type de problème, on utilise les interruptions. Les interruptions sont des portions de code (fonctions) appelés lorsque qu'un événement (interne ou externe) survient et à besoin d'être traité sur le champ. Il faut cependant faire attention, ce mécanisme interrompt le code exécuté, il est prioritaire par rapport au reste du code. Vu qu'il est possible de mesurer les événements ponctuellement (via les fonctions d'entrées/sorties) on utilise généralement les interruptions pour du code critique (arrêt d'urgence par exemple) ou des événements non-ponctuels (transmissions de données depuis un ordinateur par exemple).

Aussi, le nombre d'interruption externe est limité sur à 2 sur la plupart des cartes Arduino. Les interruptions sont utilisables sur les broches compatibles seulement (broches 2 et 3 sur l'Arduino UNO). Pour choisir la fonction et la broche utilisée pour l'interruption, on utilise la fonction attachInterrupt. On peut utiliser detachInterrupt pour supprimer l'interruption. Il est possible de partir en interruptions sur 4 types d'événements :

- **LOW** : Lorsque la broche est à l'état 0 (0V)
- **RISING** : Lorsque la broche passe de l'état 0 (0V) à l'état 1 (5V) (front montant).
- **FALLING** : Lorsque la broche passe de l'état 1 (5V) à l'état 0 (0V) (front descendant).
- **CHANGE** : Lorsque la broche change d'état (front montant et front descendant).

Voici un exemple d'utilisation :

```
volatile boolean etat = false ;
void appuiBouton() {
    etat = !etat ; // Changement d'état }
void setup() {
    pinMode(2,INPUT) ; // Broche 2 en entrée
    attachInterrupt(0,appuiBouton,RISING) ; // On attache à l'interruption 0 (broche 2) la
                                            fonction appuiBouton sur un front montant
    }
```

On remarque l'apparition du mot clef volatile avant la déclaration de la variable état. Ce mot clef est nécessaire pour toutes les variables qui sont modifiée dans une interruption. Cela à une incidence sur la manière dont le compilateur traite l'accès à la variable.

Il est parfois nécessaire de désactiver temporairement les interruptions par exemple lorsque l'on exécute du code critique (activation d'un moteur, etc.). Deux fonctions permettent de changer l'activation des interruptions interrupts et noInetrrupts pour activer (respectivement désactiver) les interruptions.

Exemple

Nous allons maintenant passer à une exemple. Il est de faire clignoter une LED de façon régulière. On doit utiliser une LED avec une résistance en série sur une breadboard. Ou en peut utiliser la LED directement sur l'Arduino (broche 13 sur la carte Arduino UNO).

Code avec delay :

```
const int pinLed = 13;

void setup()

{ pinMode(pinLed, OUTPUT); // Broche 13 en sortie }

void loop()

{

delay(500); // Attente d'une demi seconde
digitalWrite(pinLed, HIGH); // Allumage de
laLED delay(500);

digitalWrite(pinLed, LOW); // Eteignage de
laLED

}
```

Code avec millis :

```
const int
pinLed=13;

int temps;

int etat;

void setup(){
```

```
    pinMode(pinLed, OUTPUT); // Broche 13
    en sortie etat = LOW; // LED éteinte }

void loop()

{ int present = millis();

    if ( temps+500 < present ) // Vérification du chronomètre

{   temps = present; // Actualisation du chronomètre

    etat = !etat; //    Changement d'état

digitalWrite(pinLed,etat); // Changement d'état de la
LED

    } }
```

II.7.3.4. Quelques librairies

En plus de la simplicité du langage et des nombreuses fonctionnalités qu'offre. L'IDE vient avec un nombre important de librairies évitant ainsi d'implémenter des fonctions courantes dans l'informatique embarquée.

Une des librairies les plus utilisée est celle implémentant la communication série. La majorité des cartes Arduino possède un émulateur de connexion série pour communiquer au travers de l'USB. Ainsi, on peut communiquer avec l'ordinateur sur lequel la carte Arduino est connectée. Cela permet, par exemple, de déboguer un programme en affichant la valeur des variables ou simplement afficher la valeur des capteurs. Cette librairie à été directement implémentée dans le langage Arduino. On peut accéder à la communication série (au travers l'USB) grâce à l'objet Serial. Une autre librairie existe pour communiquer par liaison série via une des broches de la carte.

Il est parfois nécessaire de stocker des informations même après l'arrêt de la carte. Il est possible de stocker une petite quantité d'information sur la mémoire EEPROM (Electricaly Erasable Programmable Read Only Memory) intégrée. Une librairie est aussi fournie pour dialoguer avec cette mémoire. Il est possible de lire et d'écrire sur cette mémoire sans rien ajouter sur la carte. Cette mémoire est accessible via l'objet EEPROM et en ajoutant la librairie du même nom. Attention, la mémoire EEPROM à une durée de vie limitée (environ 100 000 cycles d'écritures). Il faut donc veiller à ne pas écrire répétitivement les données.

D’autres alternatives existent pour ajouter de plus grandes quantités de mémoires mortes à l’Arduino (carte SD notamment) mais cela demande l’ajout de composants externes.

Pour terminer, il existe aussi des librairies permettant de contrôler des composants externes. Parmi ces librairies, une des plus populaires est celle contrôlant des servomoteurs.

Un servomoteur est un composant électronique composé d’un moteur et d’une carte d’asservissement. On peut le contrôler en position (c'est-à-dire choisir l’angle du moteur) grâce aux PWM. Une librairie Arduino est implémentée pour contrôler simplement ces moteurs : la librairie Servo. Il faut toutes fois faire attention à éviter d’alimenter les servomoteurs directement sur une des broches d’alimentation de la carte. En effet, un servomoteur consomme une quantité important de courant (suivant le couple exercé par le moteur). Il est donc fortement conseillé de relier les servomoteurs sur des alimentations externes (batteries, piles de 9V, etc.) [16].

II.8. Conclusion

On peut conclure sur le fait que les cartes Arduino sont des outils puissants pour les cartes électroniques. Mais aussi, elles permettent un accès facile et intuitif à l’informatique embarqué. On pourra ainsi enrichir tout ces projets à base d’un microcontrôleur pour leurs donner une valeur plus importante.

La carte d’Arduino UNO est la plus simple carte et la plus courante. Il existe d’autres versions de cartes Arduino plus adaptées pour certains projets. Dans notre travail on va utilisé la carte d'Arduino UNO pour réaliser une simple station météo connectée par wifi. Pour certains projets plus complexes, il va falloir d'utiliser l’Arduino Méga.

Pour plus d’informations sur le projet Arduino, de l’aide pour la réalisation d’un projet ou des documentations, reportez vous sur le site officiel Arduino : http://www.arduino.cc/.

Chapitre III

Conception et réalisation de la station météo

III.1. Introduction

Le besoin d'observer et de contrôler des phénomènes physiques tels que la Température, la Pression, l'Humidité ou encore la Luminosité est essentiel pour de nombreuses applications industrielles et scientifiques, cette nécessité nous à diriger de réfléchir à la réalisation d'une nouvelle station météo qui peut mesurer ces grandeurs et l'envoyer vers le PC par une connexion sans fil (wifi) à base d'une carte Arduino UNO.

Dans ce chapitre nous allons donné une description détaillée des capteurs utilisés et le mode de fonctionnement de ces capteurs dans notre station et comment peut-on connecter avec la carte Arduino UNO, ensuite on va essayer de faire une petite application sur l'internet des objets pour utiliser le site ThingSpeak, qui va nous permettre de connecter notre système par wifi. On va présenter à la fin de ce chapitre les résultats obtenu après la réalisation de cette station.

III.2. Schéma en bloc du système à réaliser

Ce système est divisé en trois parties, la première partie est une carte qui reçoit les phénomènes physiques (température, humidité, pression et ensoleillement) et la deuxième partie l'unité de traitement des données (Arduino UNO) et la dernière partie est l'unité de gestion à distance des données (ESP8266).

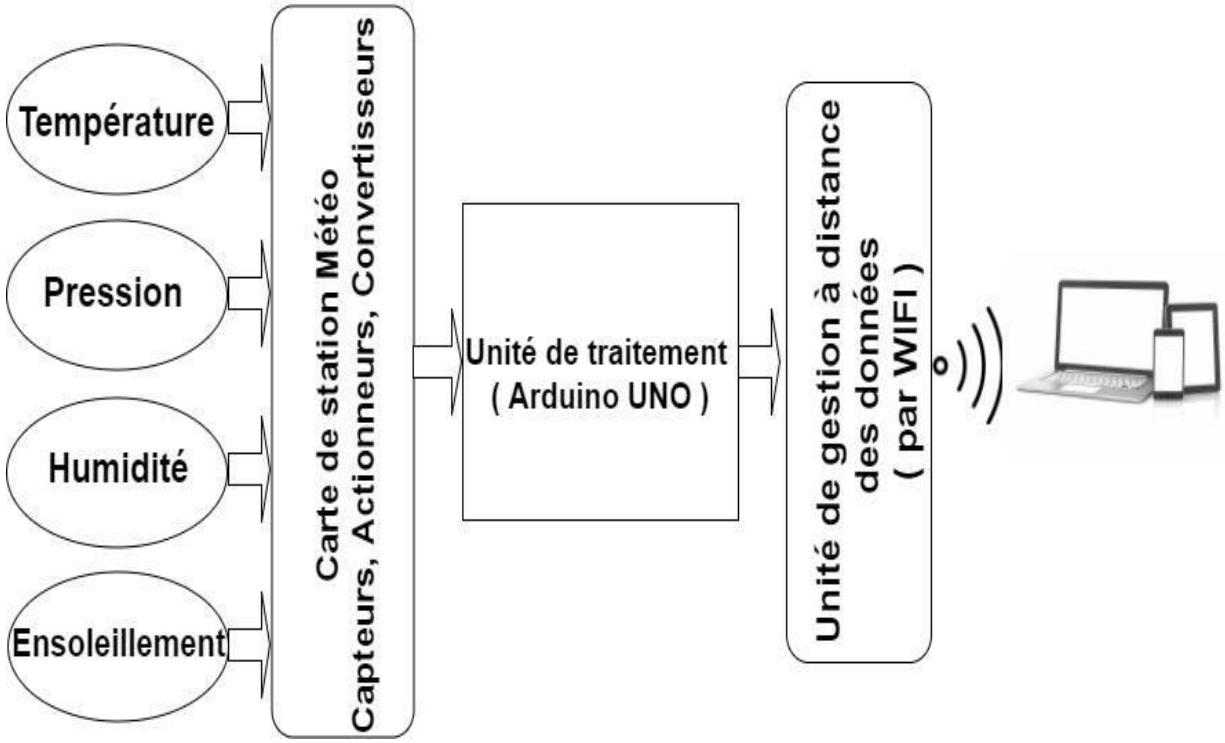

Figure III.1: Schéma bloc de système à réaliser.

III.3. Schéma électrique du montage sur Fritzing

Tous d'abord on à tracé le schéma électrique du système sur le logiciel Fritzing. Le logiciel Fritzing est un outil de création des projets et des circuits électroniques, il permet aussi l'édition de circuit imprimé, il est disponible gratuitement sur internet. Il a notamment pour vocation de favoriser l'échange de circuits électroniques libres et d'accompagner l'apprentissage de la conception de circuits.

Le logiciel comporte trois vues principales:

- La « Platine d'essai », où l'on voit les composants tels qu'ils sont dans la réalité et ou l'on construit le montage.
- La « Vue schématique », représentant le schéma fonctionnel du circuit.
- Le « Circuit imprimé », représentant la vue du circuit imprimé tel qu'il sera sorti en PDF pour être imprimé [19].

Dans les figure III.2 et III.3 on présente, respectivement, le schéma électrique tracé sur Fritzing et le montage de notre système sur plaque d'essai obtenu aussi par Fritzing.

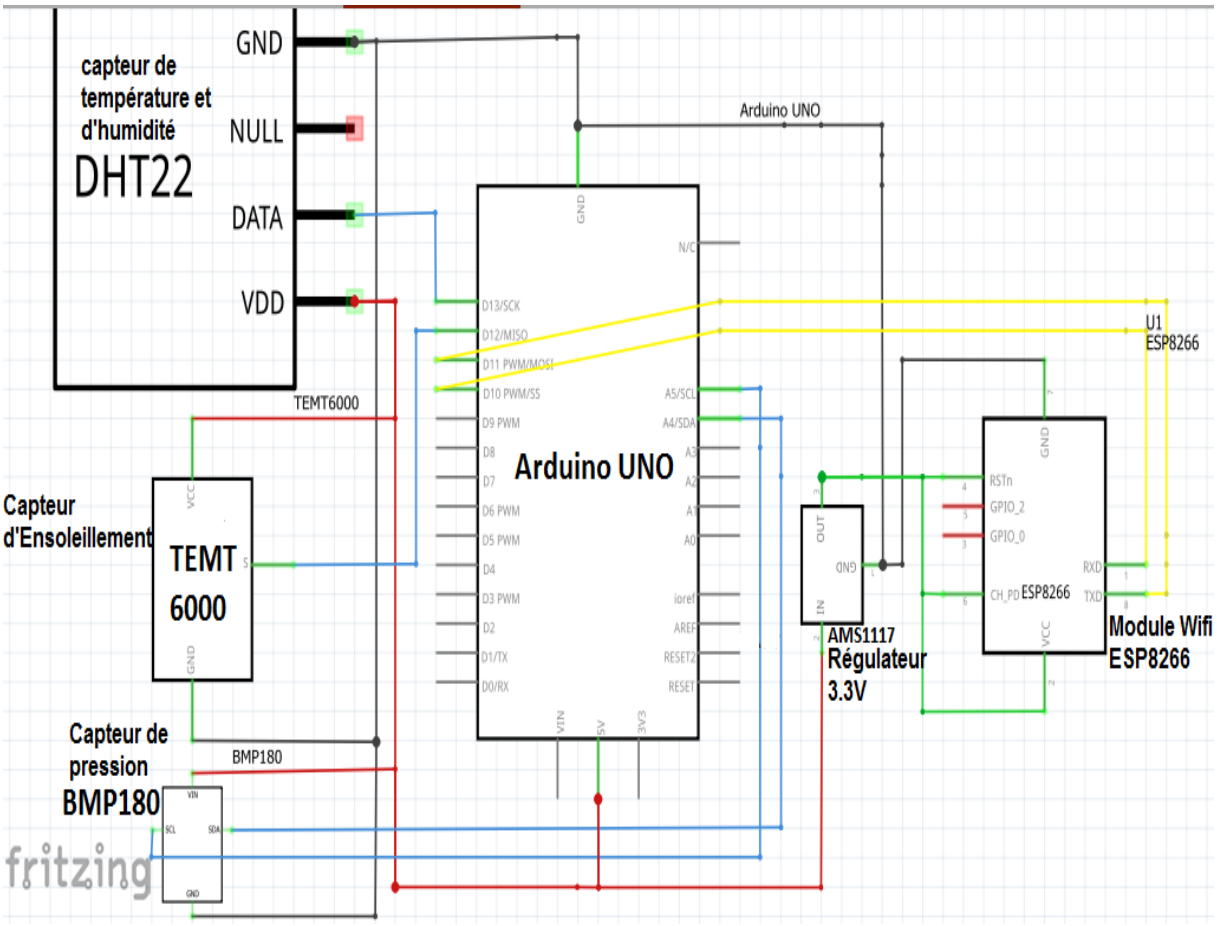

Figure III.2: Schéma électrique du montage.

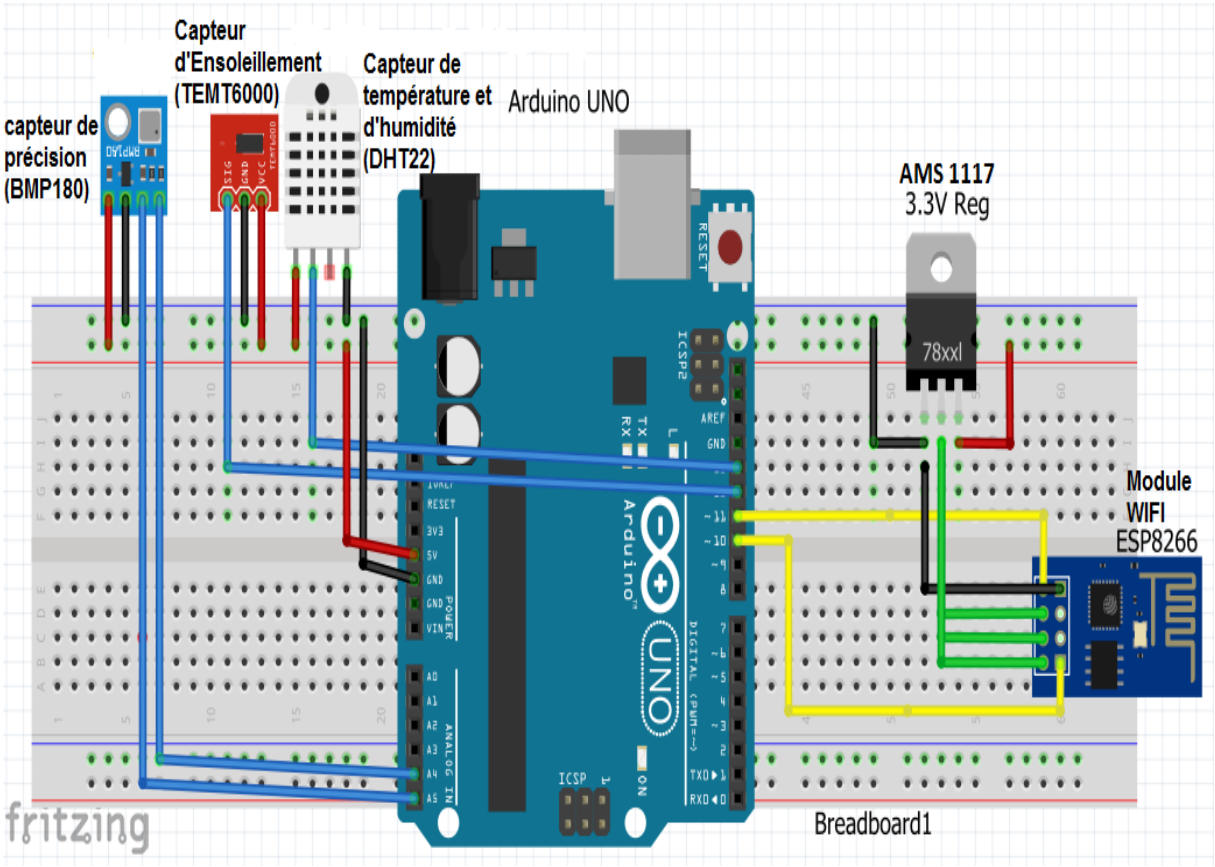

Figure III.3: Montage en plaque d'essai.

III.4. Description des capteurs utilisés

Pour réaliser notre station météo, il faut tout d'abord choisir les composants sensibles aux phénomènes physiques ce que l'on souhaite mesurés. On a fixé le choix sur la Température, l'Humidité, la Pression et l'Ensoleillement. Dons ce qui suit, on va décrire une description détaillée des différents capteurs utilisés:

III.4.1. Capteur de Température et d'Humidité (DHT22)

III.4.1.1. Description du capteur

Le DHT22 est un capteur de température et d'humidité numérique basique à faible coût. Il utilise un capteur d'humidité capacitif et une thermistance pour mesurer l'air environnant et émet un signal numérique sur la broche de données (aucune broche d'entrée analogique n'est requise). Il est assez simple à utiliser (voir Figure III.4), mais nécessite un temps minutieux pour saisir les données. Le seul inconvénient réel de ce capteur est que vous ne pouvez obtenir de nouvelles données qu'une fois toutes les 2 secondes, alors lorsqu'on déclare la bibliothèque de ce capteur dans le programme, la lecture de ce capteur peut durer jusqu'à 2 secondes.

Figure III.4: Image réelle du capteur de Température et d'Humidité DHT22.

Il suffit de connecter la première broche à gauche à l'alimentation de 3 à 5 V, la deuxième broche doit brancher à la carte Arduino, juste au pin 2 numérique (voir la figure III.5). La troisième broche est montée à la masse GND.

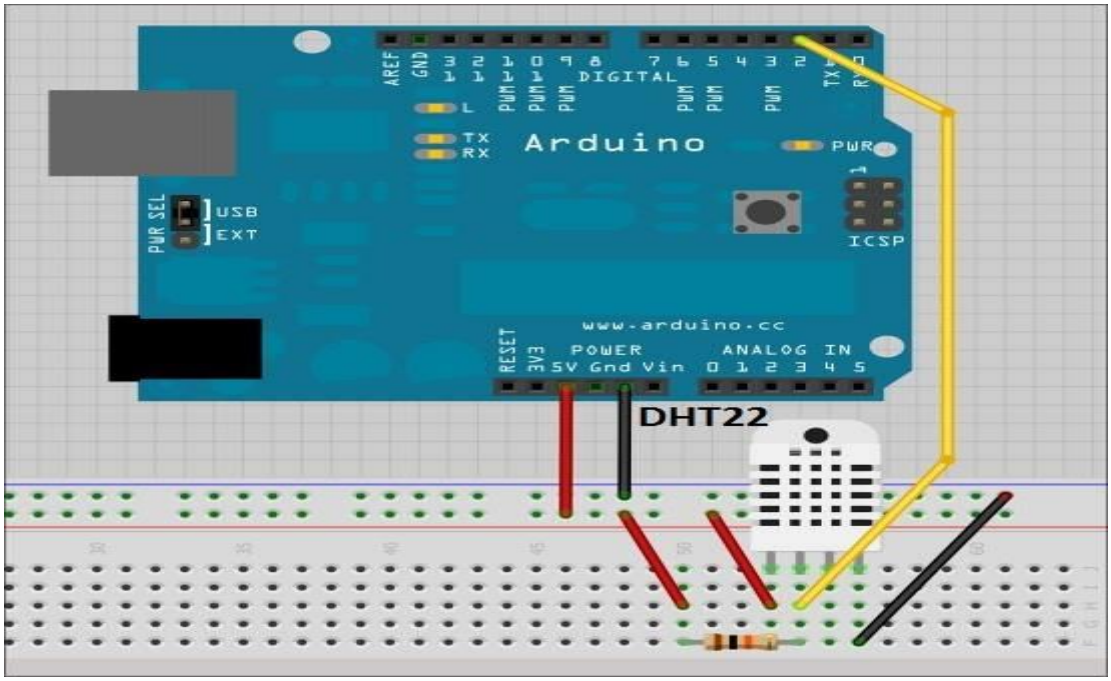

Figure III.5: Câblage du capteur de Température et d'Humidité DHT22 avec l'Arduino UNO.

III.4.1.2. Principales caractéristiques de DHT22

- À bas prix.
- Puissance et E / S de 3 à 5 V.
- 2.5 mA max courant d'utilisation pendant la conversion (lors de la demande de données).
- Bon pour 0-100% de lectures d'humidité avec 2-5% de précision.
- Bon pour les lectures de température de -40 à 80 ° ± Précision ± 0,5 ° C.

- Pas plus de 0,5 Hz de fréquence d'échantillonnage (une fois toutes les 2 secondes).
- Taille du corps 27 mm × 59 mm × 13.5 mm (1.05 "× 2.32" × 0.53 ").
- Poids (juste le DHT22): 2,4 g [20].

III.4.2. Capteur de pression (BMP180)

III.4.2.1. Description du capteur

Le capteur de pression BMP180, fabriqué par Bosch, est la meilleure solution de détection à faible coût pour mesurer la Pression et la Température barométriques. Parce que la pression change avec l'altitude, vous pouvez aussi l'utiliser comme altimètre!.

Le BMP180 est la prochaine génération de capteurs de Bosch, et remplace le BMP085. La bonne nouvelle est qu'il est complètement identique au BMP085 en termes du logiciel firmware, vous pouvez utiliser le tutoriel BMP085 et n'importe quel exemple de code bibliothèques en remplacement. La broche XCLR n'est pas présente physiquement sur le BMP180 (voir figure III.6).

Figure III.6: Image réelle du capteur de Pression BMP180.

L'utilisation du capteur BMP180 est facile. Si on utilise un Arduino UNO, on connecte simplement la broche V_{in} à la broche de tension 5V, GND à la terre, SCL à la broche Analogue 5 et SDA à la broche Analogue 4 (Figure III.7). Ensuite, on fait déclarer la bibliothèque Arduino BMP085/BMP180 et un code d'exemple pour le calcul de la pression doit l'ajouter dans le programme.

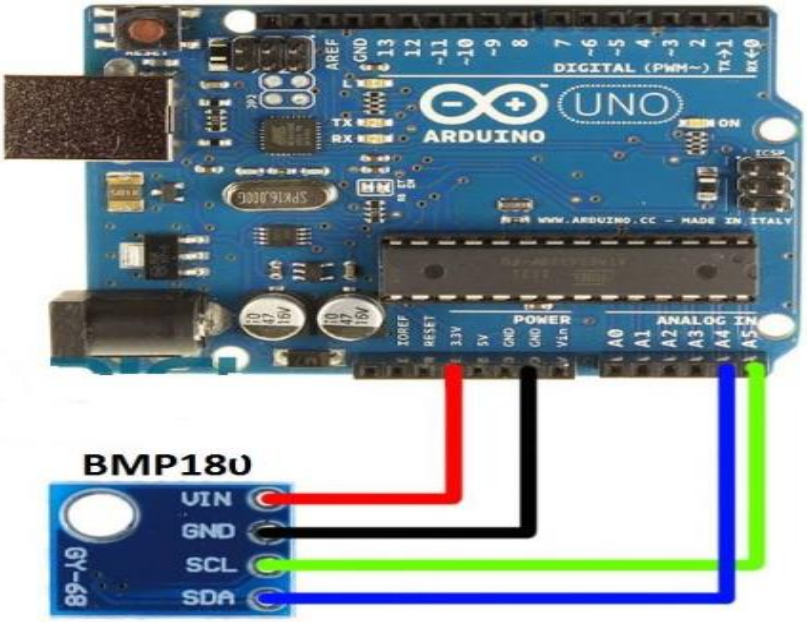

Figure III.7: Câblage du capteur de Pression BMP180 sur l'Arduino UNO.

III.4.2.2. Principales caractéristiques de BMP180

- V_{in}: 3 V à 5 V.
- Plage de détection de pression: 30-110 KPa (9000 m à -500 m au-dessus du niveau de la mer).
- Jusqu'à 0.003 KPa / 0.25 m de résolution.
- plage de mesure de -40 à + 85 ° C de, $^{+}$-2 ° C de précision de température [21].

III.4.3. Capteur d'Ensoleillement (TEMT6000)

III.4.3.1. Description du capteur

Le TEMT6000 est un phototransistor planaire épit axial NPN au silicium. Dans un moule transparent miniature pour surface. Le dispositif est sensible au spectre visible. La figure III.8 présente une image réelle de ce capteur.

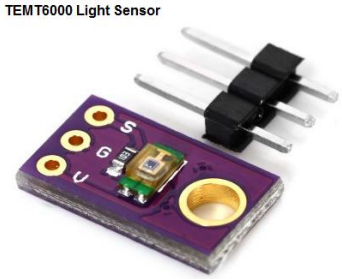

Figure III.8: Image réelle du capteur d'ensoleillement TEMT6000.

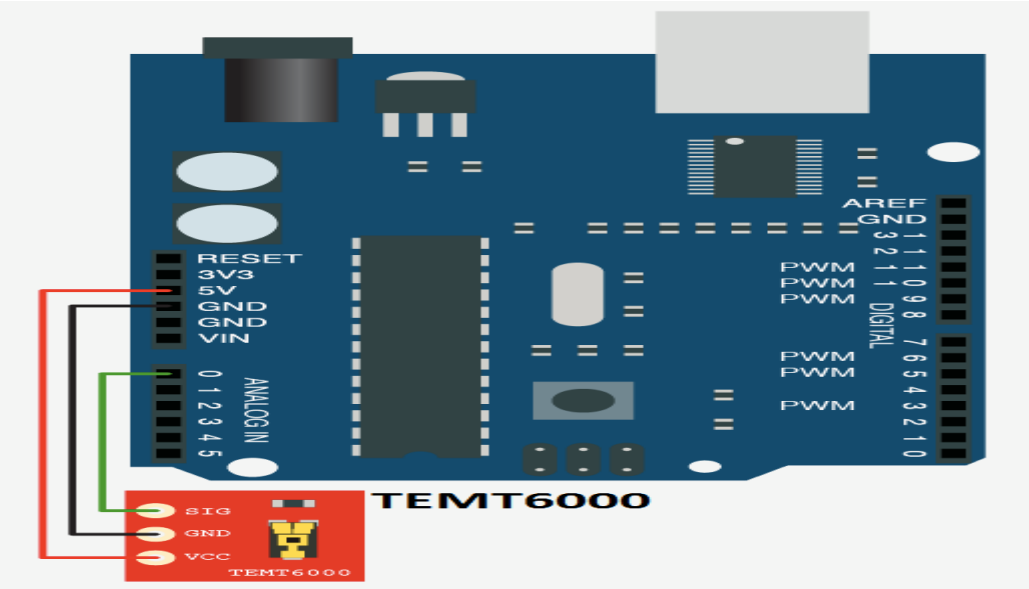

Figure III.9: Câblage du capteur d'ensoleillement TEMT6000 sur l'Arduino UNO.

III.4.3.2. Principales caractéristiques de TEMT6000

- ❖ Alimentation: 3 à 5 V
- ❖ Adapté à la réactivité des yeux humains.
- ❖ Température de service: -40°C à +85°C
- ❖ Dimensions: 10 x 10 mm.
- ❖ Composant sans plomb [22].

III.5. Présentation de l'interfaçage Arduino-PC par câble USB

Premièrement pour tester le fonctionnement des capteurs utilisés: (DHT22, BMP180 et TEMT6000), On a essayé de faire un câblage entre ces capteurs et la carte Arduino UNO. La carte météo réalisée est reliée de façon simple par un câble USB. Une image réelle de la carte météo réalisée sur une plaque d'essai et connectée par un câble USB est présentée à la figure III.10.

Le programme de gestion des données permet d'afficher toutes les valeurs en temps réel et chaque deux secondes sur une fenêtre du moniteur sérié (Voir figure III.11). Dans les annexes vous allez trouver le programme exécuté (voir l'annexe programme 01).

Figure III.10: Image réel du Montage réalisé connecté par USB.

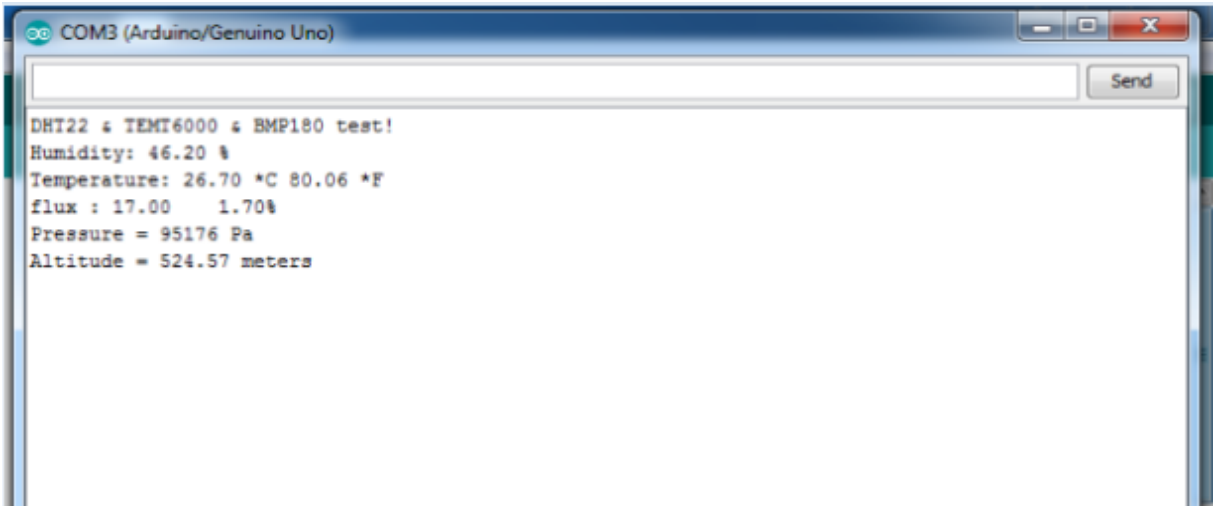

Figure III.11: Fenêtre d'affichage des résultats de mesures des capteurs sur le moniteur sérié.

Après le test, nous avons remarqué que l'alimentation par USB est instable, elle affecte sur le fonctionnement des capteurs, le risque de détruire les composants. Cet inconvénient n'est pas la seule, pour le fonctionnement en permanence de notre carte, il est obligatoire de laisser tous le temps la carte branchée sur le PC via USB.

Pour résoudre ces problèmes nous avons réfléchi de faire une connexion sans fil (par WiFi) afin d'assurer un bon fonctionnement de notre carte.

III.6. Présentation de l'interfaçage Arduino-PC par WiFi

L'interfaçage permet de faire une connexion sans fil entre notre carte Arduino UNO et le PC à travers le module WiFi ESP8266 et d'afficher les données sur le site ThingSpeak sous forme des courbes graphiques.

On notée qu'on doit alimenter la carte Arduino UNO avec un transformateur de tension continue (DC) 9V/1A (figure III.12).

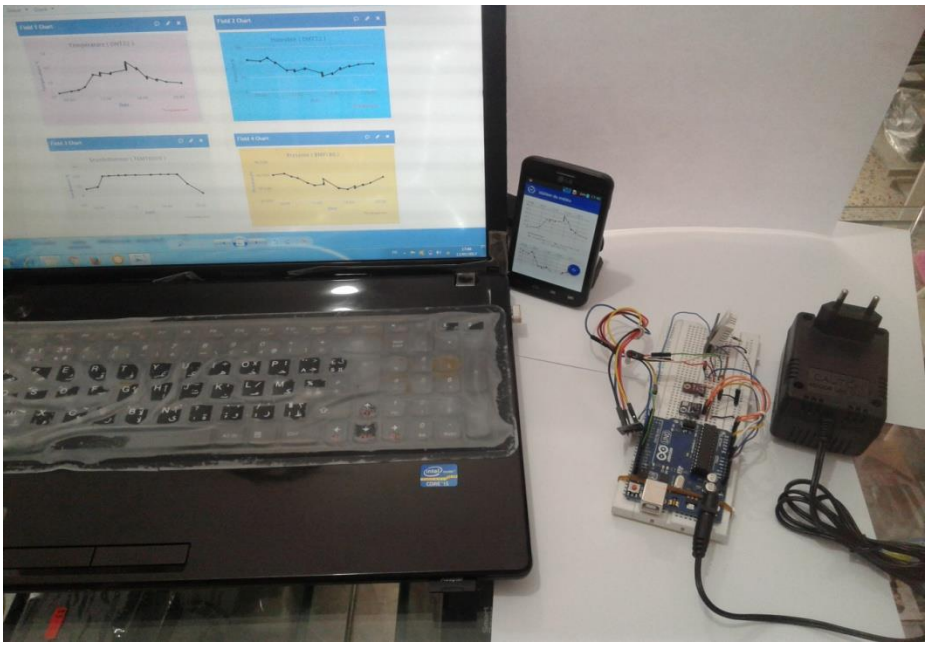

Figure III.12: Image réel du montage réalisé connecté par WiFi.

III.6.1. Description du module de connexion WiFi ESP8266

Les cartes wifi basées sur le micro contrôleur ESP8266 sont programmables comme les cartes Arduino et peuvent communiquer par wifi avec d'autres appareils (ordinateurs, Smartphones, etc.). Il existe plusieurs modèles : l'ESP-01, l'ESP-03, l'ESP-12 … etc. L'ESP-01 que nous allons utiliser dans notre travail est présenté dans la figure III.13.

Figure III.13: Image réelle du module WiFi ESP8266 – ESP01.

Le module WiFi ESP8266 est un réseau autonome avec une pile de protocoles TCP/IP intégrée qui peut donner accès au réseau WiFi. L'ESP8266 est capable d'héberger une application ou de décharger toutes les fonctions de réseau WiFi d'un autre processeur d'application. Chaque module ESP8266 est préprogrammé avec un microprogramme de configuration de commande AT (voir l'annexe pour la liste des commandes de l'ESP8266), ce qui signifie que vous pouvez simplement le brancher sur votre appareil Arduino et obtenir autant de fonctionnalités WiFi. Le module ESP8266 est un tableau extrêmement rentable avec un ensemble des commandes [23].

III.6.2. Alimentation du module ESP8266

L'alimentation de ESP8266 à partir de la carte Arduino n'est pas toujours opérationnelle, d'une part le module est prévu pour fonctionner en 3.3V et un courant de fonctionnement moyenne 80 mA, et d'autre part les broches de la carte Arduino sortent 50 mA. Pour cela, nous sommes obligés d'utiliser un régulateur AMS1117, ce régulateur permet de délivrer une tension fixe égale à 3.3 V et un courant jusqu'a 800 mA.

III.6.3. Brochage et câblage de ESP8266

Le tableau III.1 montre le brochage de l'ESP8266. Dans les figures III.14 et III.15 on présente, respectivement, l'emplacement des pins de l'ESP8266 et le câblage de l'ESP8266 sur la carte Arduino et avec le régulateur AMS1117.

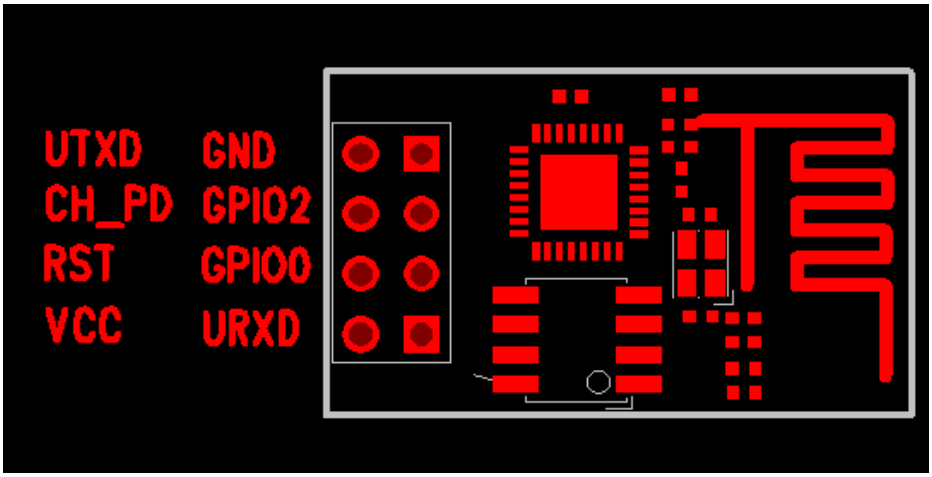

Figure III.14: Description des pins de l'ESP8266.

Pin	Working mode	Flash mode	Description
VCC **GND**			Alimentation, utiliser une alimentation indépendante et partager le GROUND avec le composant UART si possible.
TXD **RXD**			Uart interface, communication asynchrone.
RST			Restart on low TTL.
CH_PD	High TTL	High TTL	Chip select, constant High TTL for both mode.
GPIO0	High TTL(optionally)	Low TTL	Switch Wroking/Flash Mode.
GPIO 15	Low TTL	Low TTL	Constant Low TTL for both mode, N/A for ESP-01.
GPIO 2	High TTL	High TTL	Constant High TTL for both mode, maybe not necessary.
Red Leds			Sous tension la LED rouge est allumé constamment.
Blue Leds			La Led bleue est allumée lors du transfert de données.

Tableau III.1 : Tableau des pins de l'ESP8266.

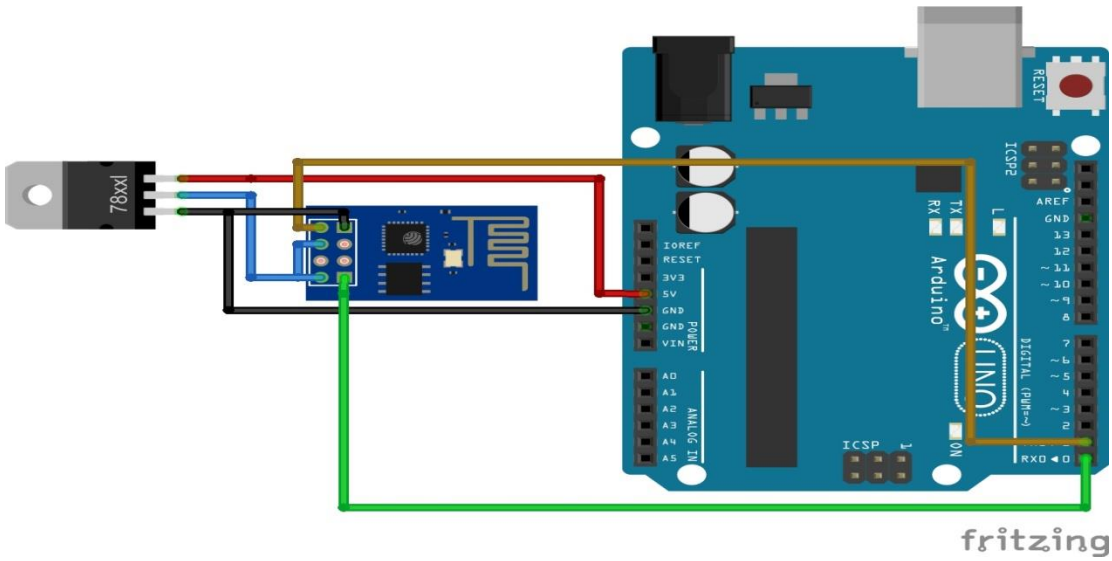

Figure III.15: Câblage du module ESP8266 sur l'Arduino UNO.

III.6.4. Mise sous tension

- Alimenter le module en reliant V_{cc} et CH PD, et câbler GND;
- La LED rouge doit être allumée et le rester;
- La LED bleue peut clignoter au démarrage;
- Vérifiez si votre module ne présente pas son réseau dans la liste des réseaux WIFI (par exemple un nouveau réseau ESP_99b22 [24].

III.6.5. Principales caractéristiques de ESP8266

- ❖ Norme sans fil: IEEE 802.11.
- ❖ Gamme de fréquence: 2.4 GHz ~ 2.5 GHz.
- ❖ Tension de fonctionnement: 3.3v.
- ❖ Courant de fonctionnement moyenne 80MA.
- ❖ Taille de paquet: 14.3 mm ×24.8 mm × 3 mm.
- ❖ Sans fil mode réseau: station/softAP/SoftAP + station.

III.7. Choix du support d'affichage

Il y'a plusieurs méthodes qu'on peut utilisés pour afficher les résultats des mesures, dans notre cas nous avons opté l'internet des objets (IdO) en anglais : **«**internet of thing (IoT)**»**.

L'internet des objets appelé la 3ème révolution industrielle, va profondément modifier le quotidien des personnes avec la domotique, la santé et loisirs, l'énergie, la distribution et notre environnement avec la cité intelligente ou les transports connectés.

Durant ces dernières années, le monde a connu un développement impressionnant de l'univers multimédia. Cela s'explique par le progrès technique et technologique ainsi que des innovations majeures qui ont bouleversé le monde de télécommunication : Informatique des nuages (**Cloud** Computing), social media, internet des objets…

Ce dernier représente l'extension d'internet à des choses et à des lieux du monde physique. Alors qu'internet ne se prolonge habituellement pas au-delà du monde électronique, l'internet des objets représente les échanges d'informations et de données provenant de dispositifs présents dans le monde réel vers le réseau Internet. L'internet des objets est considéré comme la troisième évolution de l'Internet, baptisée Web 3.0. Les objets constituant « l'Internet des objets », qualifiés de « connectés », « communicants » ou « intelligents » [25].

Actuellement les objets connectés ou intelligents sont partout, ils ont envahi le monde et impacté notre vie personnelle et professionnelle. Ils génèrent des milliards d'informations qu'il faut traiter et analyser pour les rendre utilisable. Selon les statistiques du Cisco [26] comme on peut voir sur la figure III.16, 50 à 80 milliard d'appareils connecté seront en circulation dans le monde en 2020. Bien qu'il ne s'agisse que d'une estimation, une chose est sûr c'est qu'actuellement, en moyenne dans le monde une personne dispose de plus de trois objets connectés. Une réalité qui a vu le jour et qui s'est développé depuis l'an 2003.

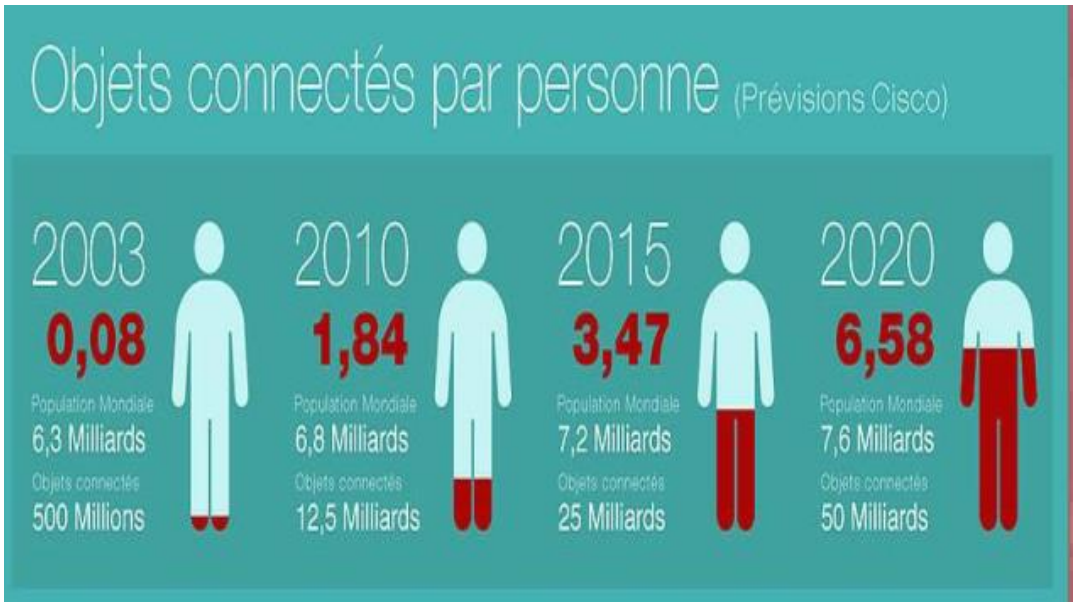

Figure III.16: Nombre des objets connectés à l'horizon 2020.

En effet, on désigne par objet connecté [27] toute appareil composé de capteur qui envoient des données vers une application mobile ou un service web, pour de multiples domaines d'application.

En outre un objet connecté est un objet dont la vocation première n'est pas d'être des périphériques informatiques ni des interfaces d'accès web mais auxquels l'ajout d'une connexion internet a permis d'ajouter une valeur supplémentaire en terme de fonctionnalités, d'information et d'interaction avec l'environnement.

Aujourd'hui les objets connectes commencent à prendre part à notre vie quotidienne et se traduisent par plusieurs et différents objets dans des multiples domaines d'applications.

En les retrouvons partout où nous allons [28]:

- chez nous : ordinateur, téléviseur, voir des objets plus simples comme un réveil ou un réfrigérateur (figure III.17).

- sur nous : les Smartphones et maintenant les montres.
- dans nos moyens de transports : la voiture, avions, peut être demain les vélos.

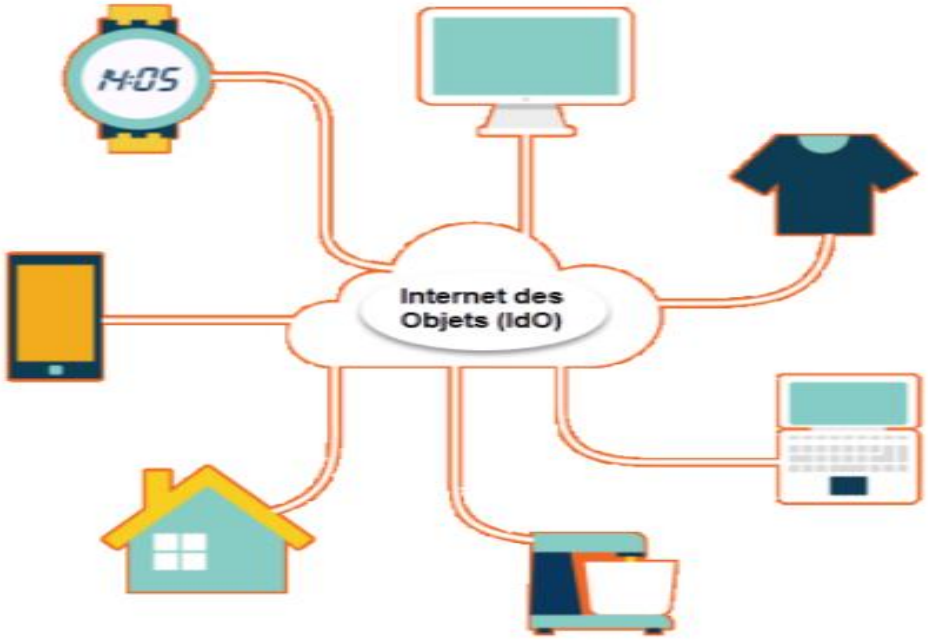

Figure III.17: Illustration des objets.

Voila les types de famille internet des objets :

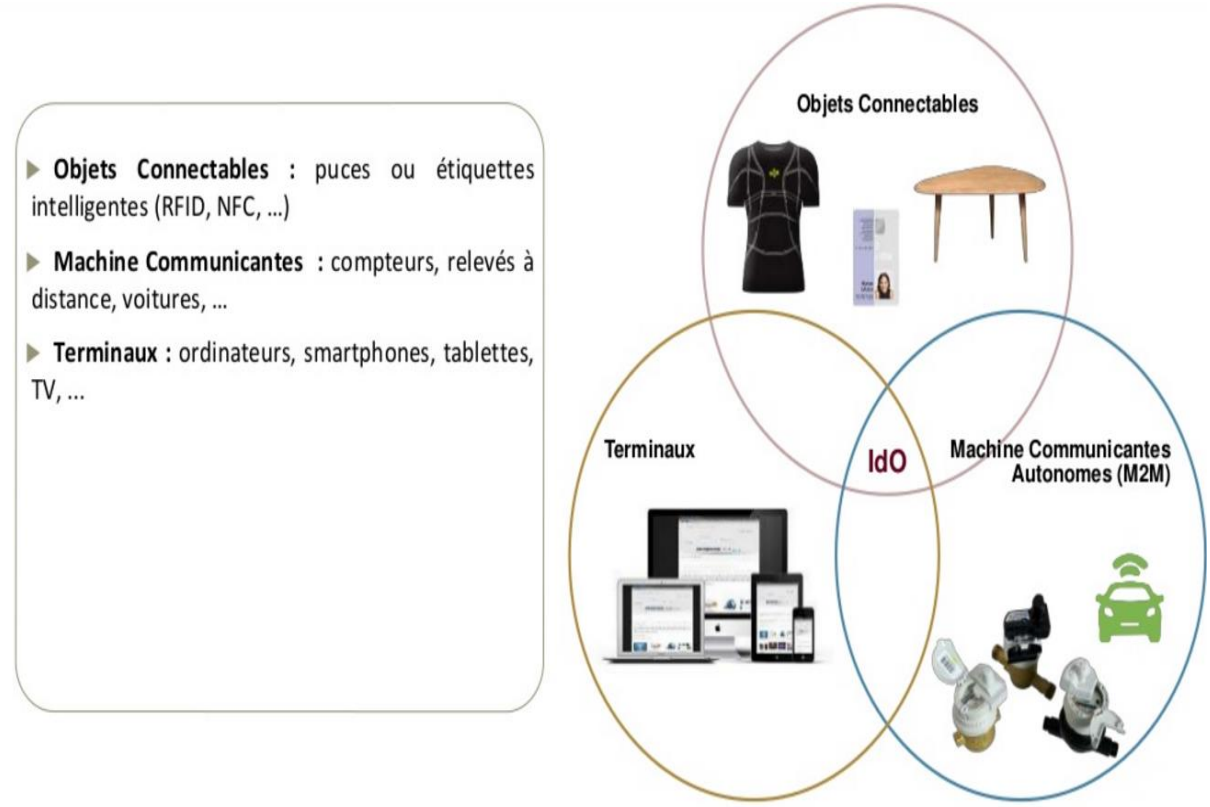

Figure III.18: Les types de famille IdO.

La fonctionnement internet des objets via Cloud:

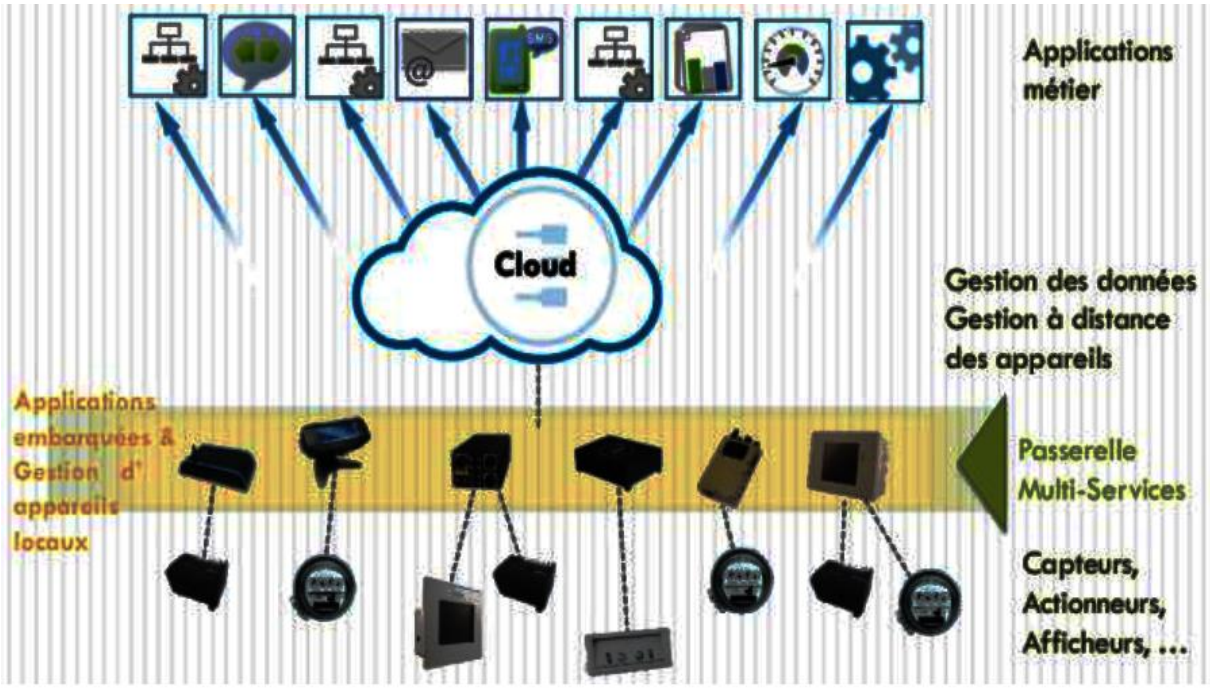

Figure III.19: Fonctionnement IdO via Cloud.

III.7.1. Avantages de l'Internet des objets

- gagné beaucoup Temps.
- Perfection.
- Accessibilité et mobilité : à tout moment, en tout lieu, tout appareil».
- Les objets connectés feront partis intégrante de notre quotidien dans tous les domaines (santé, voiture, life style, météo).

III.7.2. Choix du site d'affichage

Après le choix de l'internet des objets pour afficher les résultats, la dernière étape est de choisi un site d'affichage permet d'afficher les résultats en temps réel, nous avons opté le site "**ThingSpeak**".

Le site ThingSpeak est une entreprise qui propose différents services exclusivement destinés à la construction d'applications IoT.

Le site propose d'être un support d'iOT en permettant de:

- collecter les données en temps réel (fréquence supérieure ou égale à 15 secondes)
- visualiser les données collectées sous forme de graphes

- créer des plugins et des applications pour collaborer avec des web services, des réseaux sociaux et d'autres APIs.

Le centre de ThingSpeak est le "Channel". Un Channel enregistre les données envoyées vers ThingSpeak et comprend les éléments suivants :

- 8 champs pour stocker n'importe quel type de données : voila de quoi stocker les données d'Arduino !
- 3 champs de localisation, latitude, longitude et élévation, ce qui sera utile pour les objets en mouvements
- d'autres champs anecdotiques (adresse youtube ou vimeo,...) [29, 30].

Maintenant on doit créer un projet qui envoie des informations de notre station de météo vers ThingSpeak, en suivre les étapes suivants:

- s'inscrire
- créer un Channel
- créer cinq "Fields" dans le Channel
- récupérer la clé de mise à jour (API Key ; Write KEY)

III.7.3. Application Android dans les Smartphones

Pour faciliter l'accès aux résultats sur un Smartphone nous avons ajouté une application sur Android qui s'appelle "**ThingView**".

L'application ThingView est une application (parmi d'autres) permet de tracer les données de ThingSpeak sous forme des graphes. Notez que cette application n'est pas propriété de ThingSpeak, c'est seulement une application qui utilise les services de ThingSpeak (Figure III.20)

Figure III.20: Application ThingView.

III.8. Installation de la carte à l'extérieur de Laboratoire - les résultats obtenus

Nous avons installé la carte à l'environnement afin d'effectuer des mesures de tous les phénomènes physiques à différents moments de la journée pour savoir comment changer la température, l'humidité, la pression et la luminosité tout au long de la journée, notez que il est nécessaire la présence du réseau WiFi dans le site choisi, en fin de journée nous avons acquis les mesures suivantes (figure III.21.a, b, c et d):

Remarque : ces résultats concernant le centre de la wilaya de M'sila en date de 07 mai 2017.

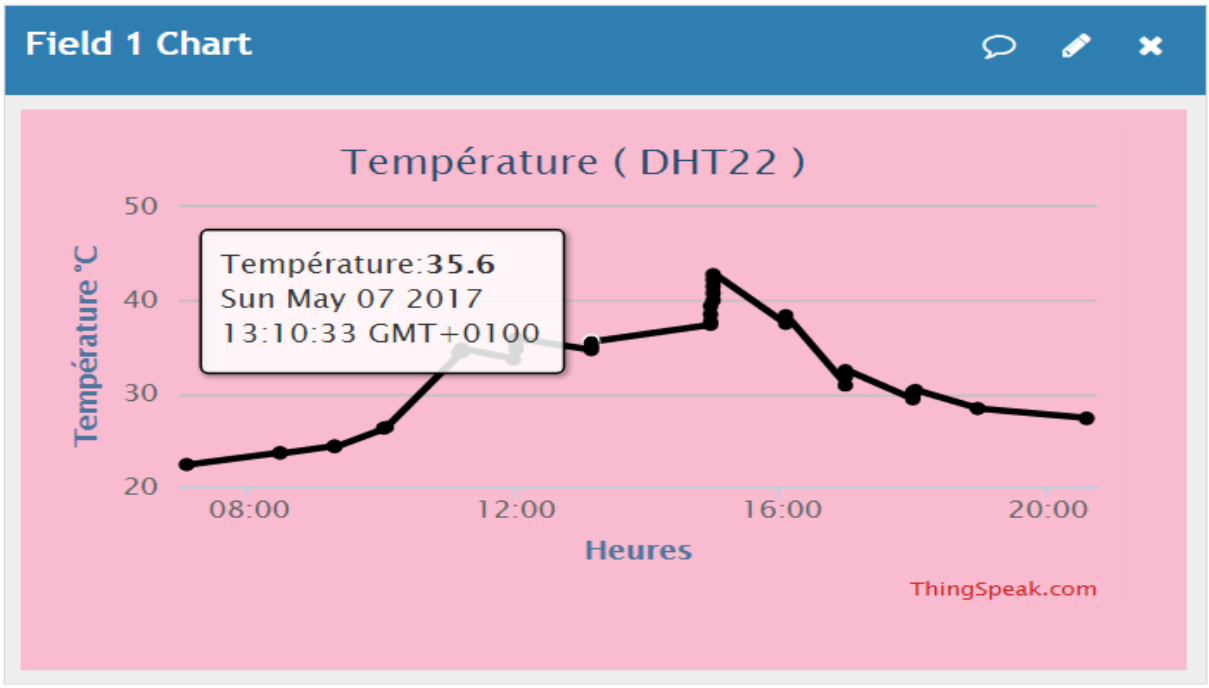

Figure III.21.a: Courbe graphique exprime le changement de la Température.

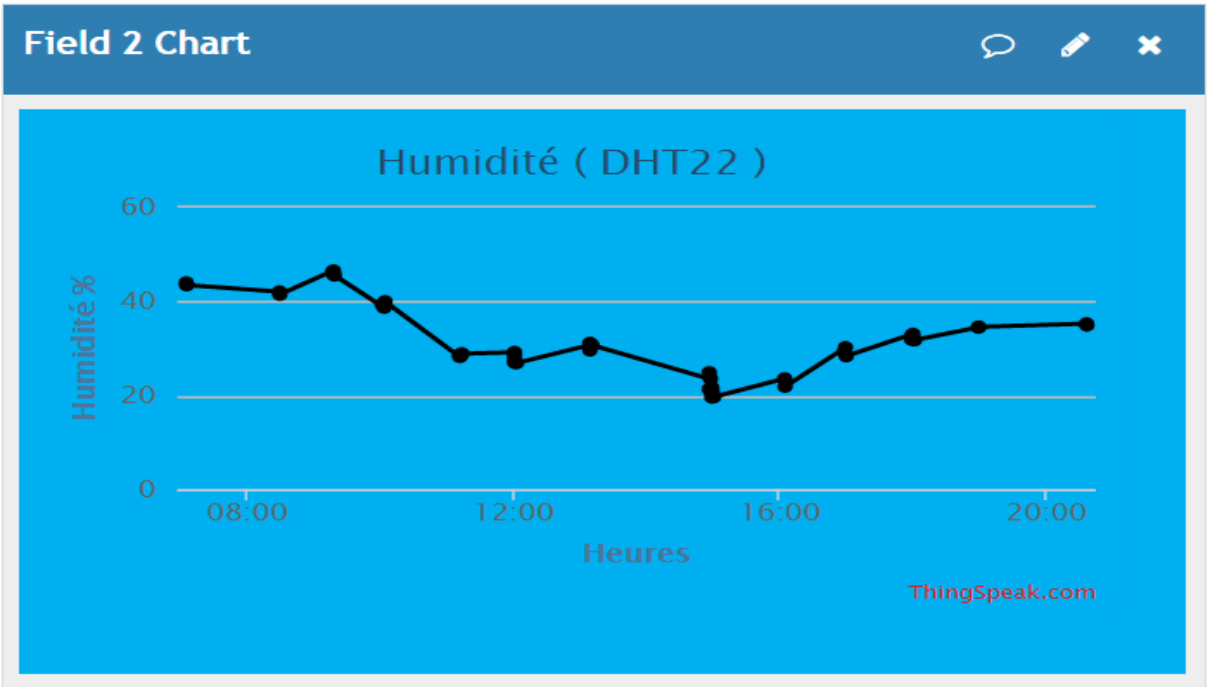

Figure III.21.b: Courbe graphique exprime le changement de l'Humidité.

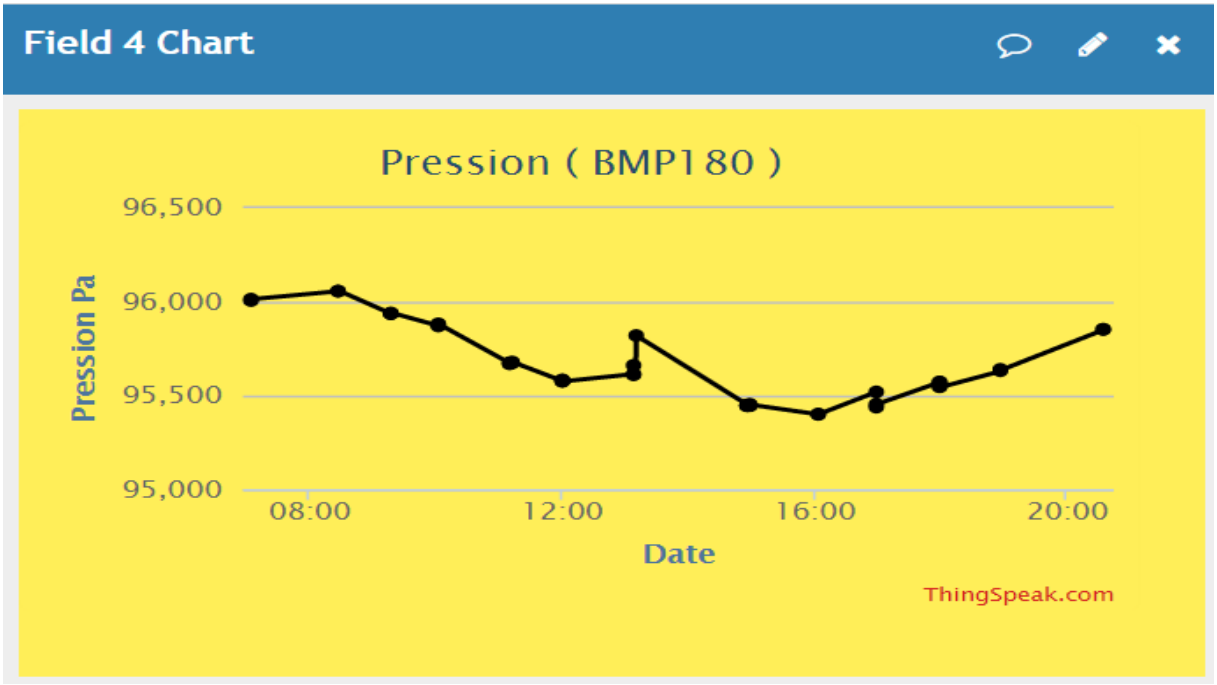

Figure III.21.c : Courbe graphique exprime le changement de la Pression.

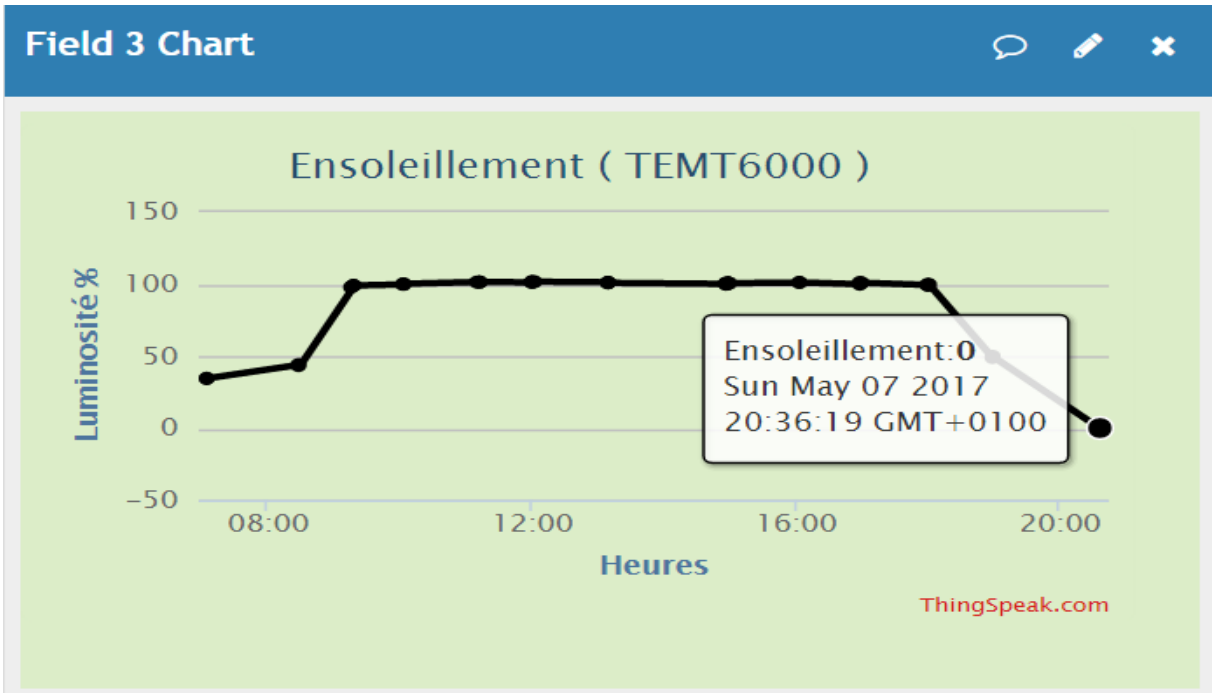

Figure III.21.d : Courbe graphique exprime le changement de la Luminosité.

III.9. Conclusion

Au début de ce chapitre, nous avons donnée une description et des caractéristiques de toutes les capteurs utilisées, puis nous avons fourni une explication détaillée de la méthode de conception et réalisation de notre station météo et la méthode suivie pour faire une connexion sans fil avec un PC ou un Smartphone.

Nous avons fait des mesures expérimentales en dehors de laboratoire pour voir comment les phénomènes physiques variés (la Température, la Pression, l'Humidité et la Luminosité), les résultats des mesures en une journée sont montrée sous forme des graphes en fonction du temps.

Conclusion générale

Conclusion générale

Dans ce projet nous nous sommes concentrés à la conception et la réalisation d'une station météo pour l'acquisition des données grâce aux quatre capteurs: de Température, d'Ensoleillement, de Pression et d'Humidité, utilisant un protocole de communication sans fil (wifi) avec un PC ou un Smartphone.

Nous avons réalisé un système de mesure en temps réel de l'ensemble des phénomènes physiques à base d'une carte Arduino UNO comme une unité de commande, le rôle de la carte Arduino UNO est de traiter les données délivrées par les capteurs utilisées. Au début on a essayé de relier le système de mesure au support d'affichage (PC) par un câble USB pour assurés le bon fonctionnement des capteurs. Le programme écrit sur IDE Arduino permet d'afficher les résultats sur le moniteur série.

La deuxième partie de ce travail consiste à faire une connexion sans fil (par wifi) à travers le module ESP8266, nous avons crées un autre programme capable d'afficher les résultats en temps réel sur le site internet ThingSpeak sous forme des graphes.

Une telle réalisation n'est pas dénuée de difficultés. Il est à noter que nous nous sommes confrontés à plusieurs problèmes surtout dans la partie de la connexion sans fil. Cependant, on peut dire que malgré ces difficultés, les résultats obtenus à travers cette étude qu'ils soient pratiques ou théoriques, permettent d'ouvrir la porte à d'autres études. Nous espérons que ce mémoire sera une référence aux personnes désirant développer et réaliser des projets et systèmes à base des carte Arduino.

Ce travail de réalisation ouvre la voie à de plusieurs perspectives dans le domaine de la commande des systèmes embarqués et de la communication des données. Nous suggérons de: a) Faire la conception et la réalisation d'un robot mobil autonome à base d'un petit panneau solaire gérée par une carte Arduino, b) Réaliser un système de vidéo surveillance dans les laboratoires pédagogiques et de recherches universitaires, c) Optimiser les systèmes de commande par Arduino pour la domotique sans fils, d) Nous suggérons aussi de proposer notre projet comme une station pour mesurer la météo à université de M'sila pour obtenir des valeurs réelles et les injectées dans le site de l'université .

Bibliographie

Bibliographie

[1] Krama. A, Gougui. A, "Etude et réalisation d'une carte de contrôle par Arduino via le système Androïde", Mémoire Master Académique, Université Kasdi Merbah Ouargla, Algérie, 2015.

[2] Yacine. Y, "Minimisation d'énergie dans un réseau de capteur", Mémoire de Magister, Université Mouloud Mammeri de Tizi Ouzou, Algérie, 2012.

[3] Vernon S. Somerset, "Intelligent and Biosensors", Edited by Vernon S. Somerset, Intech, January 2010.

[4] Bensaid. S, "Cours Capteurs et Actionneurs", Université de Bouira, 2014.

[5] Erik. V et Jacob. R, "Piezo Résistive MEMS Devices: Theory and Applications", Thèse de Magister, Departement for Micro and Technology, DTU, 2005.

[6] Philipe. M, "Faisabilité d'un capteur de pression capacitif miniature sur silicium", Thèse de Doctorat, Université de Paul Sabatier de Toulouse, 1998.

[7] El Bahri. M, "Influence de la température sur le comportement statique et dynamique des capteurs de pression capacitif sur silicium", Thèse de Doctorat, Institut National de Sciences Appliquées de Toulouse, 2005.

[8] Rouri. J, "Développement of memes sensors for mesurments of pressure, relative humidity and température", Thèse de Doctorat, Worcester Poly Technic Institue, 2003.

[9] Jingbo. X, "A monolithic silicon multi-sensor for measuring three-axis acceleration, pressure and temperature", Journal of Mechanical Science and Technology, Vol: 22 pp: 731-739, 2008.

[10] Jarne. C, "Dew points of binary carbon dioxide + water and ternary carbon dioxide + water + methanol mixtures Measurement and modeling", Fluid Phase Equilibria, Vol: 216, pp: 85-93, 2004.

[11] Grégory. M, "Absorption de l'eau par les polymères", Thèse de Doctorat, Université de Savoie, 2009.

[12] Nguyen. T, "Study and performance of humidity sensor based on the mecanical optoctronic principle for the mesurement and control of humidity in storehouses", Sensors and Actuators, Vol: B66, pp: 200-202, 2000.

[13] Pi Guey. S, "A micromachined resistive type humidity sensor with a composite material as sensitive film", Sensors and Actuators, Vol: B113, pp: 837-842, 2006.

[14] Ming. Y, "Cobalt oxide nanosheet humidity sensor integrated with circuit on chip" Microelectronic Engineering, 2011.

[15] https://fr.wikipedia.org/wiki/Photor%C3%A9sistance (consulter le 07/05/2017).

[16] Lechalupé. J, "cours d'initiation à Arduino", Université Paul Sabatier, Mai 2014.

[17] Astalaseven, Eskimon et Olyte, "Arduino pour bien commencer en électronique et en programmation", Licence Créative Commons BY-NC-SA 2.0.

[18] Cottenceau. B, "Carte ARDUINO UNO Microcontrôleur ATMega328".

[19] https://fr.wikipedia.org/wiki/Fritzing (consulter le 07/05/2017).

[20] https://www.adafruit.com/product/385 (consulter le 18/04/2017).

[21] https://www.adafruit.com/product/1603 (consulter le 18/04/2017).

[22] https://www.sparkfun.com/datasheets/Sensors/Imaging/TEMT6000.pdf (consulter le 18/04/2017).

[23] https://www.sparkfun.com/products/13678 (consulter le 20/04/2017).

[24] https://sites.google.com/site/arduinoencore/home/esp8266 (consulter le 20/04/2017).

[25] Le Mag Web4. 10 Juillet 2015 https://lemag.agenceweb4.ch/site/fr/lemag-web4?tag=1 (consulter le 02/05/2017).

[26] Evans. D, "L'internet des objets", Avril, 2011

[27] Coulon. A, "L'internet des objets un gisement à exploiter", Hiver, 2010.

[28] Vineela1. A, Sudha, L, "Internet of Things –Overview". Sudha Rani International Journal of Research in Science & Technology Vol: 2, Issue: 4, ISSN: 2349-0845, April 2015.

[29] https://sites.google.com/site/arduinoencore/esp8266 (consulter le 07/05/2017).

[30] https://sites.google.com/site/arduinoencore/esp8266/thingspeak#TOC-iOT-et-ThinkSpeak (consulter le 08/05/2017).

Annexes

Programme (01) avec câble USB

```
#include "DHT.h"  // sensor temperature & humidity
#define DHTPIN 2 // what digital pin we're connected to
                 // Connect a 10K resistor from pin 2 (data) to pin 1(power)
                    of the sensor
#define DHTTYPE DHT22  //#define DHTTYPE DHT22 (AM2302), AM2321

#include <Wire.h>
#include <Adafruit_BMP085.h>
Adafruit_BMP085 bmp; // sensor Presion
DHT dht(DHTPIN, DHTTYPE);
float temt6000Pin = 0; // sensor flux
float a;
//--------------------------------
void setup()
{
  Serial.begin(9600);
  Serial.println("DHT22 & TEMT6000 & BMP180 ");
  dht.begin();
  bmp.begin();
}
void loop()
{
 // Wait a few seconds between measurements.
  delay(2000);
 // Reading temperature or humidity takes about 250 milliseconds!
 // Sensor readings may also be up to 2 sec 'old' (its a very slow  sensor)

  float h = dht.readHumidity();
  float t = dht.readTemperature(); //Read temperature as Celsius
  float f = dht.readTemperature(true); // Read temperature as Fahrenheit
(isFahrenheit = true)
  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f))
  {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
    float value = analogRead(temt6000Pin);
    float a = value*100/1000;

  Serial.print("Humidity: ");
  Serial.print(h);
  Serial.println(" %\t");
  Serial.print("Temperature: ");
  Serial.print(t);
  Serial.print(" *C ");
  Serial.print(f);
  Serial.println(" *F\t");

  Serial.print("flux : ");
  Serial.print(value);
  Serial.print("\t");
  Serial.print(a);
  Serial.println("%");
```

```
  Serial.print("Pressure = ");
  Serial.print(bmp.readPressure());
  Serial.println(" Pa");

    // Calculate altitude assuming 'standard' barometric
    // pressure of 1013.25 millibar = 101325 Pascal
  Serial.print("Altitude = ");
  Serial.print(bmp.readAltitude());
  Serial.println(" meters");

  delay(2000);
}
```

Programme (02) sans câble USB (par WIFI)

```
#include <SoftwareSerial.h>
SoftwareSerial espSerial =  SoftwareSerial(10,11);// esp8266-01 : RX to
pin=11 , TX to pin=10

//DHT22============================
#include <DHT.h>
#define DHTPIN 2 // Connect the signal pin of DHT22 sensor to digital pin 2
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);

//temt6000====================
float temt6000Pin = 0;
float a;
//bmp180 ======================
#include <Wire.h>
#include <Adafruit_BMP085.h>
Adafruit_BMP085 bmp;
//============================
String apiKey = "VK5D36WY9SAUABAM";     // replace with your channel's
thingspeak WRITE API key

String ssid="LG";                    // Wifi network SSID
String password ="20091993";  // Wifi network password

boolean DEBUG=true;

//showResponce=====================================================
void showResponse(int waitTime){
    long t=millis();
    char c;
    while (t+waitTime>millis()){
      if (espSerial.available()){
        c=espSerial.read();
        if (DEBUG) Serial.print(c);
      }
    }
}
//================================================================================
boolean thingSpeakWrite(float value1, float value2, float value3, float
value4, float value5)
{
  String cmd = "AT+CIPSTART=\"TCP\",\"";              // TCP connection
  cmd += "184.106.153.149";                          // api.thingspeak.com
  cmd += "\",80";
  espSerial.println(cmd);
```

```
  if (DEBUG) Serial.println(cmd);
  if(espSerial.find("Error")){
    if (DEBUG) Serial.println("AT+CIPSTART error");
    return false;
 }
  String getStr = "GET /update?api_key=";   // prepare GET string
  getStr += apiKey;

  getStr +="&field1=";
  getStr += String(value1); // t
  getStr +="&field2=";
  getStr += String(value2); // h
  getStr +="&field3=";
  getStr += String(value3); // a
  getStr +="&field4=";
  getStr += String(value4); // pression
  getStr +="&field5=";
  getStr += String(value5); // altitude
  // ...
  getStr += "\r\n\r\n";

  // send data length
  cmd = "AT+CIPSEND=";
  cmd += String(getStr.length());
  espSerial.println(cmd);
  if (DEBUG)  Serial.println(cmd);

  delay(100);
  if(espSerial.find(">")){
    espSerial.print(getStr);
    if (DEBUG)  Serial.print(getStr);
  }
  else{
    espSerial.println("AT+CIPCLOSE");
    // alert user
    if (DEBUG)   Serial.println("AT+CIPCLOSE");
    return false;
  }
  return true;
}
//setup====================================================================
void setup() {
  DEBUG=true;           // enable debug serial
  Serial.begin(9600);

  dht.begin();              // Start DHT22 sensor
  bmp.begin();              // Start bmp180 sensor
  espSerial.begin(115200); // enable software serial
  // Your esp8266 module's speed is probably at 115200.
 // For this reason the first time set the speed to 115200 or to your
esp8266 configured speed and upload. Then change to 9600 and upload again

 espSerial.println("AT+RST");      // Enable this line to reset the module
 showResponse(1000);

  //espSerial.println("AT+UART_CUR=9600,8,1,0,0");  // Enable this line to
set esp8266 serial speed to 9600 bps
  //showResponse(1000);

  espSerial.println("AT+CWMODE=1");   // set esp8266 as client
  showResponse(1000);
```

```
  espSerial.println("AT+CWJAP=\""+ssid+"\",\""+password+"\"");  // set your
home router SSID and password
  showResponse(5000);

   if (DEBUG)  Serial.println("Setup completed");
}
//Loop ================================================================
void loop() {
  // Read sensor values
   float t = dht.readTemperature();
   float h = dht.readHumidity();
   float value = analogRead(temt6000Pin);
   float a = value*100/1000;

        if (isnan(t) || isnan(h)) {
        if (DEBUG) Serial.println("Failed to read !");
      }
      else {
          if (DEBUG) Serial.println("Temp="+String(t)+" *C");
          if (DEBUG) Serial.println("Humidity="+String(h)+" %");
          if (DEBUG) Serial.println("flux="+String(a)+" %");
          if (DEBUG) Serial.println("Pressure =
"+String(bmp.readPressure())+" Pa");
          if (DEBUG) Serial.println("Altitude =
"+String(bmp.readAltitude())+" meters");
           thingSpeakWrite(t,h,a,bmp.readPressure(),bmp.readAltitude());
// Write values to thingspeak
      }
  // thingspeak needs 15 sec delay between updates,
  delay(20000);  //50000
}
```

la liste des commandes ESP8266

ESP8266 AT Command Set

Function	*AT Command*	*Response*
Working	AT	OK
Restart	AT+RST	OK [System Ready, Vendor:www.ai-thinker.com]
Firmware version	AT+GMR	AT+GMR 0018000902 OK
List Access Points	AT+CWLAP	AT+CWLAP +CWLAP:(4,"RochefortSurLac",-38,"70:62:b8:6f:6d:58",1) +CWLAP:(4,"LiliPad2.4",-83,"f8:7b:8c:1e:7c:6d",1) OK
Join Access Point	AT+CWJAP? AT+CWJAP="SSID","Password"	Query AT+CWJAP? +CWJAP:"RochefortSurLac" OK
Quit Access Point	AT+CWQAP=? AT+CWQAP	Query OK
Get IP Address	AT+CIFSR	AT+CIFSR 192.168.0.105 OK
Set Parameters of Access Point	AT+ CWSAP? AT+ CWSAP= <ssid>,<pwd>,<chl>, <ecn>	Query ssid, pwd chl = channel, ecn = encryption
WiFi Mode	AT+CWMODE? AT+CWMODE=1 AT+CWMODE=2 AT+CWMODE=3	Query STA AP BOTH
Set up TCP or UDP connection	AT+CIPSTART=? (CIPMUX=0) AT+CIPSTART = <type>,<addr>,<port> (CIPMUX=1) AT+CIPSTART= <id><type>,<addr>, <port>	Query id = 0-4, type = TCP/UDP, addr = IP address, port= port
TCP/UDP Connections	AT+ CIPMUX? AT+ CIPMUX=0 AT+ CIPMUX=1	Query Single Multiple
Check join devices' IP	AT+CWLIF	
TCP/IP Connection Status	AT+CIPSTATUS	AT+CIPSTATUS? no this fun
Send TCP/IP data	(CIPMUX=0) AT+CIPSEND=<length>; (CIPMUX=1) AT+CIPSEND= <id>,<length>	
Close TCP / UDP connection	AT+CIPCLOSE=<id> or AT+CIPCLOSE	
Set as server	AT+ CIPSERVER= <mode>[,<port>]	mode 0 to close server mode; mode 1 to open; port = port
Set the server timeout	AT+CIPSTO? AT+CIPSTO=<time>	Query <time>0~28800 in seconds
Baud Rate*	AT+CIOBAUD? Supported: 9600, 19200, 38400, 74880, 115200, 230400, 460800, 921600	Query AT+CIOBAUD? +CIOBAUD:9600 OK
Check IP address	AT+CIFSR	AT+CIFSR 192.168.0.106 OK
Firmware Upgrade (from Cloud)	AT+CIUPDATE	1. +CIPUPDATE:1 found server 2. +CIPUPDATE:2 connect server 3. +CIPUPDATE:3 got edition 4. +CIPUPDATE:4 start update
Received data	+IPD	(CIPMUX=0): + IPD, <len>: (CIPMUX=1): + IPD, <id>, <len>: <data>
Watchdog Enable*	AT+CSYSWDTENABLE	Watchdog, auto restart when program errors occur: enable
Watchdog Disable*	AT+CSYSWDTDISABLE	Watchdog, auto restart when program errors occur: disable

* New in V0.9.2.2 (from http://www.electrodragon.com/w/Wi07c)

Résumé

Dans ce travail nous allons réaliser une station météo pour l'acquisition des données grâce à des capteurs de Température, de l'Ensoleillement, de Pression et d'Humidité, réunis sur une seule plaque électronique. Toutes les données récupérées sont traitées par une unité de traitement à base d'un Arduino UNO et transférées au PC, Tablet ou Smartphone à travers une liaison WiFi (sans fil).

La carte Arduino UNO permet d'établir une bonne liaison WiFi entre la station et le PC afin d'afficher toutes les données en temps réel.

Mots clés: Capteurs Température, Ensoleillement, Arduino UNO, Shield, Station météo, WiFi, Internet des objets.

ملخص

في هذا العمل سنقوم بإنشاء محطة رصد جوي للحصول على معطيات من خلال أجهزة استشعار درجة الحرارة وأشعة الشمس و الضغط والرطوبة، جمعت على لوحة إلكترونية واحدة. يتم جمع كل هذه البيانات بواسطة بطاقة اردوينو اينو ويتم نقلها إلى جهاز الكمبيوتر، هاتف ذكي أو لوحة ذكية من خلال وصلة واي فاي (لاسلكي).

بطاقة الاردوينو اينو تسمح باتصال لا سلكي جيد (عن طريق الواي فاي) بين المحطة و جهاز الكمبيوتر لعرض كافة البيانات في الوقت الحقيقي.

كلمات مفتاحية : جهاز استشعار الحرارة، أشعة الشمس، اردوينو اينو، محطة جوية، واي فاي، إنترنت الأشياء.

Summary

In this work we realize a weather station for data acquisition through sensors of Temperature, Sunshine, Pressure and Humidity assembled on breadboard. All these data are retrieved by the card Arduino UNO and transferred to the Laptop, Smartphone or Tablet through a wireless transmission WiFi.

The Arduino UNO card allows a good wireless connection (with WiFi) between the station and the Laptop in order to display all the data in real time.

Keywords: Temperature sensors, Sunshine, Arduino UNO, Shield, weather station, WiFi, Internet of things.