République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Amar Thelidji- Laghouat

FACULTE : Technologie

DEPARTEMENT : Electronique

MEMOIRE DE MASTER

Présenté par :

Lahoual Ossama

DOMAINE : Science et technologie

FILIERE : Electronique

OPTION : Electronique des systèmes embarqués

Thème

Conception et simulation numérique d'un capteur ultrasonique à l'aide d'un montage Arduino et modélisation des données via Python

Jury de soutenance :

Nom et Prénom	Grade	Qualité
Mr : Birane Mouhoub	Prof	Président
Mm : Chatta Wahiba	MCA	Examinateur
Mr : Abdelhafid Mouhoub	MCA	Rapporteur

Promotion : Juin – 2026

Remerciements

Louange à Dieu, par Sa grâce les bonnes œuvres s'accomplissent. Je Le remercie infiniment pour Ses bienfaits innombrables et pour m'avoir permis d'achever ce travail humble.

Je tiens à exprimer ma profonde gratitude et mes sincères remerciements à mon honorable encadrant, Monsieur le ***Dr. Abdelhafid Mouhoub*** *Maitre de conférences à l'université Amar Tledji de Laghouat, pour son encadrement précieux, ses conseils éclairés et son accompagnement bienveillant tout au long de l'élaboration de ce mémoire. Que Dieu le récompense généreusement et fasse de ses efforts une source de bénédiction dans cette vie et dans l'au-delà.*

J'adresse également mes remerciements les plus sincères aux membres du jury, pour l'honneur qu'ils m'ont accordé en acceptant de juger ce travail, ainsi que pour leurs observations constructives et enrichissantes.

Mes remerciements vont aussi à ma chère famille, pilier de mon parcours, pour leur soutien inconditionnel, leurs prières et leur présence constante. Je remercie également mes amis et tous ceux qui, de près ou de loin, m'ont apporté aide, motivation, ou même une simple parole d'encouragement.

Enfin, je rends grâce à Dieu pour son aide et sa guidance, et je prie pour que ce travail soit utile et accepté

Sommaire

Liste des symboles et des abréviations
Liste des figures
Liste des tableaux
Introduction Générale 1

Chapitre I: Généralités sur les capteurs ultrasoniques et moyens de mesure

I.1 Introduction 3
I.2 Caractérisation de l'onde sonor 3
I.2.1 Timbre d'un son 3
I.2.1.1 Son pur 3
I.2.1.2 Sons complexes 3
I.2.1.3 Les bruits 3
I.2.2 Les sources sonores 4
I.2.3 Célérité du son 4
I.2.3.1 Célérité du son dans les solides: 4
I.2.3.2 Célérité du son dans les liquides 4
I.2.4 Puissance et Intensité sonores 4
I.2.5 Pression acoustique 5
I.2.6 Impédance acoustique 5
I.2.7 Niveau sonore: 6
I.3 Éléments de base de la physique acoustique 6
I.3.2 Nature de l'onde ultrasonore : 7
I.3.3 Caractéristiques d'une onde ultrasonore 9
I.3.3.1 Domaine temporel : fréquence (f) et période (T) 9
I.3.3.2. Domaine spatial : longueur d'onde (λ) : 10
I.4 Interaction des ultrasons avec la matière 10
I.4.1 Atténuation de l'amplitude de l'onde ultrasonore 10
I.5 La production des ultrasons 11
I.5.1 Les générateurs piézoélectriques 11
I.5.2 Les émetteurs magnétostrictifs 11
I.5.3 L'électrostriction 12
I.6 La détection des ultrasons 12
I.7 Capteur ultrasonore 12
I.7.2 Principe de fonctionnement des émetteurs et récepteurs à ultrasons : 13
I.7.3 Fonctionnement d'un émetteur US : 14
I.7.4 Fonctionnement d'un récepteur US : 14
I.7.5 exemple d'application d'un capteur ultrasonique 14

I.7.5.1.2 Caractéristiques et spécification du capteur 14
I.7.5.1.3 Broches de connexion : 15
1.7.5.1.4 Spécifications et limites : 15
I.7.5.1.5 Fonctionnement : 16
I.7.5.1.6 Calcul de la distance de l'objet 17
I.7.5.2 Capteur ultrasonique Grove 101020010 17
I.7.5.2.1 Caractéristiques 18
I.7.5.2.2 Broches de connexion 18
I.7.5.2.3 Spécifications et limites 18
I.7.5.2.4 Principe de fonctionnement : 18
I.8 Applications du capteur à ultrasons: 20
I.9 Avantages du capteur à ultrasons HC-SR04 : 20
I.10 Inconvénients du capteur à ultrasons HC-SR04 : 20
I.11 Conclusion 21
Chapitre II:Dispositif programmable Arduino et protocole de communication Firmata
II.1 Introduction : 22
II.2 Définition du module Arduino 22
II.3 Les gammes de cartes Arduino : 23
II.3.1 Arduino Uno : 23
II.3.2 Arduino Mega 2560 : 23
II.3.3 Arduino Leonardo : 23
II.3.4 Arduino Due : 23
II.3.5 Arduino Nano : 24
II.3.6 Arduino Nano 33 BLE : 24
II.4 Le logiciel Arduino IDE : 24
II.5 Description du logiciel Arduino : 25
II.5.1 Description de la barre des boutons : 27
II.5.2 Description de la barre des menus : 28
II.5.2 Les différents types de l'ARDUINO : 28
II.6 Constitution de la carte Arduino UNO : 29
II.6.1 Microcontrôleur principal : 29
II.6.2 Alimentation : 29
II.7 Scripts Python : 31
II.8 Programmation avancée : Firmata, Firmata Express et PySerial : 32
II.8.2 Firmata Express : version allégée pour le prototypage : 32
II.8.3 PySerial : interface Python pour la communication série : 33
II.8.4 Communication Arduino – Python : 33

II.8.4.1 Communication Arduino - Python via le port série avec Pyserial : 33
II.8.4.1 Communication Arduino-Python via le protocole de communication Firmata :....34
II.9 Conclusion :....35

Chapitre III : Conception et simulation du capteur ultrasonique

III.1 Introduction :....37
III.2 Ondes sonores :....37
III.2.1 Arduino et ondes sonores :....37
III.2.1.1 Principe de fonctionnement du haut-parleur....38
III.2.1.2 Principe de fonctionnement d'un buzzer (transducteur) piézo-électrique....39
III.2.1.3 Description de la piézoélectricité :....39
III.3 Produire & Exploiter les ondes sonores. 43
III.3.1.1 Dispositif expérimentale 43
III.3.1.2 Programme en langage Arduino 44
III.3.1.3 Déroulement du programme :....44
III.3.2 Alarme sonore par détection de passage....45
III.3.2.1 Programme en langage Arduino 45
III.3.2.2 Déroulement du programme :....46
III.3.3 Emettre une onde sonore avec un bouton-poussoir 47
III.3.3.1 Programme en langage Arduino 47
III.3.3.2 Déroulement du programme :....48
III.4 Ondes ultrasonores : Vitesse & Distances....49
III.4.1 conception et simulation du capteur ultrasonique : 49
III.4.2 Détermination de la vitesse du son dans l'air....50
III.4.2.1 Programme en langage Arduino 50
III.4.2.2 Déroulement du programme Arduino....53
III.4.2.3 Programme en Python :....54
III.4.2.5 Résultats dans la fenêtre Python :....55
III.4.3 Détermination de la distance entre le capteur et l'obstacle :....55
III.4.3.1 Programme en langage Arduino: 56
III.4.3.2 Résultats dans le moniteur série :....58
III.4.3.3 Déroulement du programme Arduino....58
III.4.3.4 Programme en Python :....59
III.4.3.5 Résultats dans la fenêtre Python :....60
III.4.4 Détecteur d'obstacles 60
III.4.4.1 Programme en langage Arduino: 62
III.4.4.2 Résultats dans le moniteur série :....63
III.4.4.3 Déroulement du programme Arduino....63

III.4.4.4 Programme en Python :.....64
III.4.4.5 Résultats dans la fenêtre Python 65
III.5 Conclusion : 65
Conclusion Générale 67
Bibliographie ……..69

Liste des abréviations

Abréviation	Signification
HC-SR04 :	Capteur ultrasonique de distance
US :	Ultrasons (Ultrasonic)
IDE :	Integrated Development Environment (Environnement de Développement Intégré)
PWM :	Pulse Width Modulation (Modulation de Largeur d'Impulsion)
IoT :	Internet of Things (Internet des Objets)
TOF :	Time of Flight (Temps de Vol)
UART :	Universal Asynchronous Receiver Transmitter
SPI :	Serial Peripheral Interface
I2C :	Inter-Integrated Circuit
LED :	Light Emitting Diode
GND :	Ground (Masse)
Vcc :	Tension d'alimentation positive
Trig :	Trigger (Déclenchement)
Echo :	Signal de retour (Écho)
Ko :	Kilooctet
E/S:	Entrées / Sorties
Mo:	Mégaoctet (Unité de mesure de la mémoire, équivalant à 1024 Ko).
SRAM:	Static Random Access Memory / Mémoire vive statique.
EEPROM:	Electrically Erasable Programmable Read-Only Memory.
MHz :	Mégahertz.
HID :	Human Interface Device.
BLE:	Bluetooth Low Energy.
OS :	Operating System
GUI :	Graphical User Interface / Interface Graphique.
USB :	Universal Serial Bus.
ECG :	Électrocardiogramme.
RX / TX:	Receive (Réception) / Transmit (Transmission).
UNO :	Fait référence à la carte Arduino Uno,
A0 :	Désigne l'entrée analogique numéro 0 de la carte Arduino
D3 / D11:	Broches Digitales (numériques) 3 et 11 de la carte Arduino.
HIGH / LOW:	États logiques haut (5V) présence de tension) et bas (0V / absence de tension) en programmation Arduino.
Sonar :	SOund Navigation And Ranging
PySerial :	Bibliothèque logicielle Python permettant de gérer les communications série avec la carte Arduino.

Liste des symboles

Symbole	Signification	Unité
c:	Célérité (vitesse) du son	m/s
d:	Distance entre le capteur et l'obstacle	m ou cm
Δt ou Dt :	Durée de propagation de l'onde (temps de vol)	s ou µs
f:	Fréquence	Hz
T :	Période	s
λ:	Longueur d'onde	m
A :	Amplitude de l'onde	-
I :	Intensité sonore	W/m^2
P :	Puissance acoustique	W
p:	Pression acoustique	Pa
Z :	Impédance acoustique	$kg \cdot m^{-2} \cdot s^{-1}$
ρ:	Masse volumique du milieu	kg/m^3
α:	Coefficient d'atténuation	m^{-1}

Liste des figures

Figure(I.1) :	Ondes ultrasonores : Vitesse & Distances	6
Figure (I.2) :	principe de la mesure par écho ou du Sonar	7
Figure (I.3) :	Représentation de l'onde ultrasonore en fonction du temps et de la distance.	7
Figure (I.4) :	Illustration de la propagation linéaire d'une onde (ultra)sonore	8
Figure (I.5) :	Analogie avec un ressort	9
Figure (I.6) :	Représentation d'une onde sonore selon la longueur d'onde et la fréquence	10
Figure (I.7) :	Profile d'atténuation de l'amplitude de l'onde ultrasonore	11
Figure (I.8) :	Principe de la détection et la mesure des ultrasons	12
Figure (I.9) :	les différents composants d'un traducteur ultrasonique	13
Figure (I.10) :	Emetteurs et récepteurs à ultrasons	14
Figure (I.11) :	Schéma de principe d'un transducteur US	14
Figure(I.12) :	Capteur ultrason HC-SR04	15
Figure(I.13) :	Broches de connexion de capteur a ultrason avec arduino	15
Figure(I.14) :	fonctionnement du capteur ultrason HC-SR04	16
Figure(I.15) :	Signal de sortie sur Trigger	17
Figure(I.16) :	Capteur ultrasonique Grove 101020010	18
Figure(I.17) :	La durée de l'état haut du signal SIG correspond au temps entre l'émission des ultrasons et leur réception	19
Figure(I.18) :	Capteur à 2 broches différentes Trigg et Echo	19
Figure(I.19) :	Capteur à 1 broche commune pour Trigg et Echo	20
Figure (II.1) :	Présentation de logiciel arduino.	24
Figure (II.2) :	Logiciel Arduino IDE	25
Figure (II.3) :	Barre d'outils	25
Figure (II.4) :	Les différentes parties de la fenêtre principale du logiciel Arduino	26
Figure (II.5) :	Fenêtre associée au moniteur série	27
Figure (II.6) :	structure de la barre des boutons	27
Figure (II.7) :	la barre des menus	28
Figure (II.8) :	Les différentes parties de la carte Arduino UNO	29
Figure (II.9) :	Microcontrôleur ATMEGA328p	29
Figure (II.10) :	L'alimentation de la carte arduino uno	30
Figure(II.11) :	Entrées/Sorties numériques et les entrées analogiques	30
Figure(II.12) :	Communication série	31
Figure(II.13) :	Communication Arduino - Python via le port série avec Pyserial	34
Figure(II.14) :	Communication Arduino-Python via le protocole de communication Firmata	35
Figure (III.1) :	Structure et Figure (III.1) principe de fonctionnement d'un haut-parleur électrodynamique	38
Figure (III.2) :	Forces de Laplace exercées sur la bobine mobile dans le champ magnétique du haut-parleur	39

Figure (III.3) :	Buzzer piézoélectrique utilisé pour la génération de signaux sonores	39
Figure (III.4) :	Moment dipolaire	40
Figure (III.5) :	différentes modes de fonctionnements d'un transducteur piézoélectrique utilisé dans les capteurs ultrasoniques	41
Figure (III.6) :	Structure d'un diaphragme piézoélectrique (bilame)	41
Figure (III.7) :	Déformation d'un bilame piézoélectrique en fonction de la polarité appliquée	42
Figure (III.8) :	Génération d'un signal sonore par d'un bilame piézoélectrique	42
Figure (III.9) :	Circuit utilisé pour produire un son avec un Arduino	43
Figure (III.10) :	Programme en langage Arduino IDE pour Faire clignoter une DEL et produire un "beep	44
Figure (III.11) :	Circuit électronique utilisé comme alarme de détection de passage	45
Figure (III.12) :	Programme en langage Arduino IDE pour commander l'alarme de détectionde passage	46
Figure (III.13) :	Circuit électronique utilisé pour émettre une onde	47
Figure (III.14) :	Programme en langage Arduino IDE pour commander la production d'une onde sonore	48
Figure (III.15)	Montage expérimentale utilisé pour la mesure la distance par le capteur HC-SR04	49
Figure (III.16) :	Principe de détermination de la vitesse du son dans l'air par un capteur ultrasonique	50
Figure (III.17) :	Résultats de mesure de la vitesse du son dans l'air par l'Arduino	53
Figure (III.18) :	Résultats de mesure la vitesse du son dans l'air par Python	55
Figure (III.19) :	Principe de détermination de la distance entre le capteur et l'obstacle par un capteur ultrasonique	56
Figure (III.20) :	Résultats de mesure de la distance entre le capteur et l'obstacle par l'Arduino	58
Figure (III.21) :	Résultats de mesure de la distance entre le capteur et l'obstacle par Python	60
Figure (III.22) :	Résultats de détection d'un obstacle qui déclenchera une alarme lumineuse par l'Arduino	63

Liste des tableaux

Tableau (I.1) : Spécifications et limites du capteur ultrason HC-SR04 ……15

INTRODUCTION GÉNÉRALE

Dans un monde en pleine mutation technologique, marqué par l'essor de la robotique, de l'Internet des Objets (IoT) et des systèmes embarqués intelligents, les capteurs de proximité et de mesure de distance jouent un rôle fondamental. Parmi les différentes technologies existantes, les capteurs ultrasoniques se distinguent par leur simplicité, leur faible coût et leur fiabilité dans des environnements variés. Basés sur le principe de l'écho (sonar), ces capteurs permettent une détection sans contact en exploitant la propagation et la réflexion des ondes ultrasonores (fréquences supérieures à 20 kHz) [1,2].

Le capteur HC-SR04, largement utilisé dans le prototypage rapide, constitue aujourd'hui un outil technique de référence. Associé à une carte Arduino, il offre la possibilité de mesurer des distances et de détecter des obstacles de manière efficace. Cependant, la simple acquisition de données via le moniteur série de l'Arduino présente des limites en termes de visualisation et de traitement. C'est dans ce contexte que l'intégration du langage Python devient un atout majeur pour la récupération, le traitement et la représentation graphique des données en temps réel.

Ce projet de fin d'études s'inscrit dans cette dynamique en proposant la conception, la simulation et la réalisation d'un système de détection ultrasonique basé sur le capteur HC-SR04, une carte Arduino et un script Python pour la modélisation des données.

Cependant plusieurs facteurs peuvent être affecter les mesures assavoir comment concevoir et implémenter un système de mesure de distance et de détection d'obstacles ultrasonique fiable, peu coûteux et performant, en combinant la plateforme Arduino pour l'acquisition des données et le langage Python pour leur traitement et leur visualisation en temps réel ?

Pour répondre à cette problématique, ce mémoire poursuit les objectifs suivants :

- Présenter les principes physiques des ondes ultrasoniques et les caractéristiques du capteur HC-SR04 ;
- Maîtriser l'architecture et la programmation de la carte Arduino pour l'acquisition des données ;
- Réaliser expérimentalement la détermination de la célérité du son dans l'air, la mesure de distances et la conception d'un détecteur d'obstacles avec alarme lumineuse et sonore ;
- Développer un script Python permettant l'acquisition, le traitement et la visualisation graphique des données via le port série.

Pour atteindre cet objectif, ce travail est abordé selon les chapitres suivants :

- Le premier chapitre est consacré aux généralités sur les ondes sonores, les ultrasons et les capteurs ultrasoniques.
- Le deuxième chapitre présente la plateforme Arduino, son environnement de développement et le protocole de communication Firmata.
- Le troisième chapitre expose la conception pratique du système, les résultats expérimentaux et l'implémentation Python.
- Une conclusion générale viendra synthétiser les apports du travail et ouvrir sur des perspectives d'amélioration.

Ce projet illustre parfaitement l'intérêt des technologies open-source dans la formation des ingénieurs et techniciens. Il combine électronique embarquée, physique des ondes et programmation pour aboutir à un prototype fonctionnel, pédagogique et évolutif.

Chapitre I:
Généralités sur les capteurs ultrasoniques et moyens de mesure

I.1 Introduction

Les ultrasons (US) sont des vibrations mécaniques provoquant des variations de pression dans les milieux traversés. L'onde sonore nécessite un milieu moléculaire pour se propager (pas d'onde sonore dans le vide). L'ultrason ne fait pas partie du spectre EM

Dans ce chapitre, on va citer quelques notions utiles dans le domaine de mesure de l'acoustique. Pour cela nous commencerons par définir la notion de son et ultrason et ses différentes formes. Ensuite, nous présenterons un aperçu complet des capteurs ultrasoniques, en abordant leur principe de fonctionnement, leur classification et leurs caractéristiques techniques, dans le but d'offrir une vue d'ensemble des moyens de mesure utilisés dans les différents contextes d'application. Une citation des avantages et des inconvénients sera fournie.

I.2 Caractérisation de l'onde sonor

L'onde sonore est une onde de pression longitudinale, les molécules vibrant dans la direction de propagation de l'énergie [3]

I.2.1 Timbre d'un son

On distingue habituellement:

I.2.1.1 Son pur

Un son pur (ou son simple) correspond à une onde sinusoïdale plane: $X = A.sin(wt)$, dont la fréquence et l'amplitude maximale sont constantes au cours du temps. avec: A est l'amplitude (le déplacement maximal de la particule).

I.2.1.2 Sons complexes

La plupart des sons que nous percevons dans notre environnement ne sont pas purs mais complexes. Ils sont composés de plusieurs sons purs de fréquences et d'amplitudes différentes. On distingue 2 types [4]:

- ✓ Sons complexes périodiques: Somme de fonctions sinusoïdales de fréquences multiples entières d'une fréquence dite fondamentale – Fréquences multiples : harmoniques de rang 2, de rang 3, de rang 4,... – Instruments à vent (flûte...) et à cordes (guitare...).
- ✓ Sons complexes non périodiques ou bruits (tambour).

I.2.1.3 Les bruits

Le bruit est un son complexe produit par des vibrations aléatoires des molécules d'air. Le bruit est un phénomène à la fois physique (émission et propagation de l'onde sonore), physiologique (réception et traitement de l'onde par le système auditif).

I.2.2 Les sources sonores

- Les émetteurs: ce sont des systèmes mécaniques: instruments de musique, cordes vocales, haut-parleur....l'émetteur émet souvent un son complexe
- Les résonateurs: ce sont des appareils, qui soumis à une excitation unique, se mettent à vibrer selon une fréquence qui leur est propre.
- La cavité buccale: elle présente une structure complexe, mais curieusement ses propriétés acoustiques sont assez proche de celles d'un modèle simple [5].

I.2.3 Célérité du son

La vitesse (c) d'une onde acoustique dans les matériaux est appelée célérité. Dans un milieu homogène, la vitesse du son est constante ; elle ne dépend pas de l'amplitude des vibrations sonores et ne dépend pas de la fréquence des vibrations sonores si on admet que le milieu est non dispersif. On peut utiliser les relations approchées qui suivent [6]:

I.2.3.1 Célérité du son dans les solides:

$$c = \sqrt{\frac{E}{\rho}} \tag{I.1}$$

avec, c: célérité du son en m/s. E: Module d'Young du solide en N/m².(encore appelé module d'élasticité longitudinale du solide). ρ: masse volumique du milieu en Kg/m³.

I.2.3.2 Célérité du son dans les liquides

$$c = \sqrt{\frac{K}{\rho}} \tag{I.2}$$

avec, c: célérité du son en m/s. K: Module de compressibilité isotherme du liquide en N/m² ou en Pa . ρ: masse volumique du milieu Kg/m³.

Célérité du son dans les gaz :

Loi de Laplace

$$c = \sqrt{\frac{YP}{\rho}} \tag{I.3}$$

avec, c: célérité du son en m/s. Y: coefficient de compression adiabatique (sans unité) P : pression absolue du gaz (Pa), ρ: masse volumique du milieu en Kg/m³.

I.2.4 Puissance et Intensité sonores

La puissance est une quantité d'énergie émise pendant un certain temps. Une source sonore diffuse de l'énergie acoustique. En général, l'énergie désigne tout ce qui permet d'effectuer un travail, fabriqué de la chaleur, de la lumière ou un mouvement. L'énergie se mesure en Joule (J). L'énergie acoustique correspond aux mouvements des molécules d'air propageant l'onde acoustique. Si une source acoustique émet un son d'énergie égale à E

Joules pendant un intervalle de temps Δt secondes, alors on définit la puissance acoustique p de la source comme [7]:

$$p\,\frac{E}{\Delta t} \tag{I.4}$$

La puissance acoustique se mesure en Watts (W).

L'intensité est un flux d'énergie transportée par unité de surface. Ou intensité sonore, correspond donc à une quantité d'énergie acoustique E (en joules) qui traverse une surface (réelle ou virtuelle) S en m² pendant un temps t (en secondes). Elle se définit comme :

$$I = \frac{E}{S.\Delta t} \tag{I.5}$$

et elle se mesure en W/m².

L'intensité et la puissance acoustique sont reliées par la relation

$$I = \frac{P}{S} \tag{I.6}$$

L'intensité et la pression acoustique sont reliées par la relation suivante :

$$I = \frac{P^2}{\rho.c} \tag{I.7}$$

avec c est la célérité du son et ρ est la masse volumique du milieu.

I.2.5 Pression acoustique

Aussi appelée surpression acoustique. Lors de la propagation d'une onde acoustique, les molécules d'air en mouvement modifient légèrement la pression localement. Cette variation de pression est ce que l'on appelle la pression acoustique P. Ainsi, en un point de l'espace, la pression totale [7,8]:

$$P_{tot} = P_{atm} + P \tag{I.8}$$

Avec

$$P = \rho.v.c \tag{I.9}$$

où ρ est la masse volumique du milieu considéré et v est la vitesse vibratoire d'un point du milieu du mouvement. On peut donc relier la puissance et la pression acoustique par la relation :

$$p = v.P = \rho.v^2.c \tag{I.10}$$

Cette pression acoustique exerce une force sur le tympan. Elle met en vibration le tympan qui transmet les ondes au cerveau via les mécanismes de l'oreille moyenne et de l'oreille interne.

I.2.6 Impédance acoustique

Notée Z, c'est une grandeur particulièrement importante pour caractériser un milieu. elle présente la résistance à la propagation de l'onde et est donnée par [9]:

$$Z = \rho.c = \frac{P}{v} \tag{I.11}$$

Z est exprimé par $Kg.m^2.s^{-1}$

I.2.7 Niveau sonore:

Le niveau sonore noté L d'un son d'intensité sonore I à un son d'intensité sonore I0 est égal en bels (B) à : $L = log\left(\frac{I}{I_0}\right)$ (I.12)

Comme le bel est une unité trés grande, il est d'usage d'employer la sous unité décibel (dB), alors

$$L = 10log\left(\frac{I}{I_0}\right) \quad (I.13)$$

avec I_0 est l'intensité de référence qui correspond à la plus petite intensité acoustique détectable par l'oreille d'une fréquence 1000Hz et donnée par $10^{-12}W.m^{-2}$.

I.3 Éléments de base de la physique acoustique

I.3.1 Définition des ultrasons

Le son est une onde mécanique qui se propage dans un milieu matériel fluide (air, eau) ou solide et les ondes sonores sont caractérisés par leur fréquence. Les sons audibles par l'Homme ont des fréquences comprises entre 20 et 20 000 Hz, les infrasons ont une fréquence inférieure à 20 Hz, et les ultrasons sont situés au-delà de 20 kHz [9].

La vitesse de propagation, ou célérité, du son est indépendante de sa fréquence mais dépend du milieu de propagation : plus le milieu matériel est dense plus la vitesse est grande. Par exemple : c (air) = 340 $m.s^{-1}$; c (eau de mer) = 1500 $m.s^{-1}$; c (acier) = 5000 $m.s^{-1}$. La célérité du son dépend de la température, c'est-à-dire de l'agitation des particules qui constituent le milieu de propagation : plus la température est élevée plus le son se propage vite. Par exemple : c (air à 0°C) = 331 $m.s^{-1}$; c (air à 15°C) = 340 $m.s^{-1}$. La relation entre la vitesse du son dans l'air en m/s et la température en kelvins est:

$$c_{air} = 20.05\sqrt{T} \quad (I.14)$$

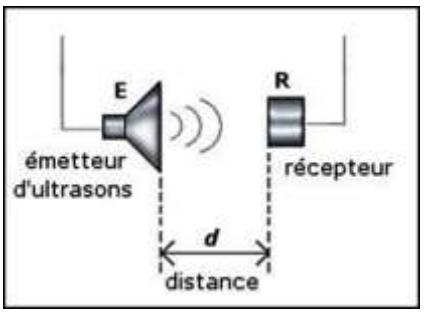


Figure (I.1) : Ondes ultrasonores : Vitesse & Distances.

La célérité dans l'air, en $m.s^{-1}$, peut être déterminée expérimentalement en mesurant la durée de propagation Δt, en s, de l'onde sonore, entre un émetteur et un récepteur situés à une distance d, en m, grâce à la relation :

$$C_{air} = \frac{d}{\Delta t} \quad (I.15)$$

On peut également utiliser un ensemble émetteur - récepteur d'ondes ultrasonores placées devant un obstacle. C'est le principe de la mesure par écho ou du Sonar :

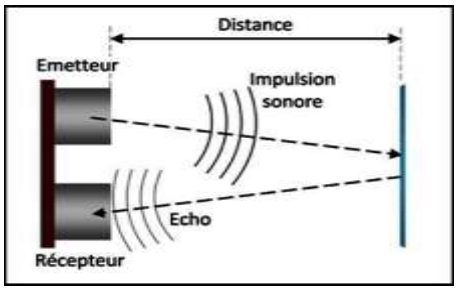


Figure (I.2) : principe de la mesure par écho ou du Sonar.

Dans ce cas, la distance parcourue par l'onde sonore, pendant la durée Δt, est 2 d, et alors :

$$C_{air} = \frac{2d}{\Delta t} \quad (I.16)$$

I.3.2 Nature de l'onde ultrasonore :

C'est une onde sonore ou onde acoustique, c'est-à-dire un mode de propagation de l'énergie dans un milieu matériel sans transport de matière.

Par définition, Un phénomène périodique est un phénomène qui se reproduit dans le temps en restant identique à lui-même [10, 11].

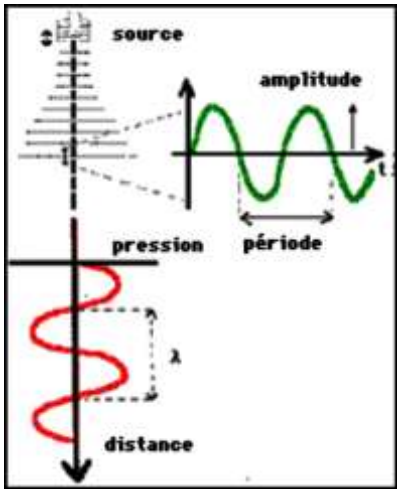


Figure (I.3) : Représentation de l'onde ultrasonore en fonction du temps et de la distance.

L'onde ultrasonore est une onde de pression se propageant dans un milieu élastique : variation de pression qui se déplace. Il s'agit de la propagation d'une énergie mécanique dans un milieu

matériel : ce déplacement ne peut se faire dans le vide (à la différence des ondes électromagnétiques). Le milieu de propagation de l'onde ultrasonore est soumis à une succession de surpressions et de dépressions et ses particules constitutives sont alors animées d'un mouvement de va-et-vient dans l'axe de déplacement des ultrasons, de type sinusoïdal. Une comparaison peut être faite avec un ressort que l'on comprime à une extrémité à une cadence donnée : ces compressions engendrent des déformations qui se propagent le long du ressort (figure I.3).

L'onde ultrasonore est une onde de pression se propageant dans un milieu élastique : variation de pression qui se déplace. Il s'agit de la propagation d'une énergie mécanique dans un milieu matériel : ce déplacement ne peut se faire dans le vide (à la différence des ondes électromagnétiques) [12]. Le milieu de propagation de l'onde ultrasonore est soumis à une succession de surpressions et de dépressions et ses particules constitutives sont alors animées d'un mouvement de va-et-vient dans l'axe de déplacement des ultrasons, de type sinusoïdal. Une comparaison peut être faite avec un ressort que l'on comprime à une extrémité à une cadence donnée : ces compressions engendrent des déformations qui se propagent le long du ressort (figure I.4).

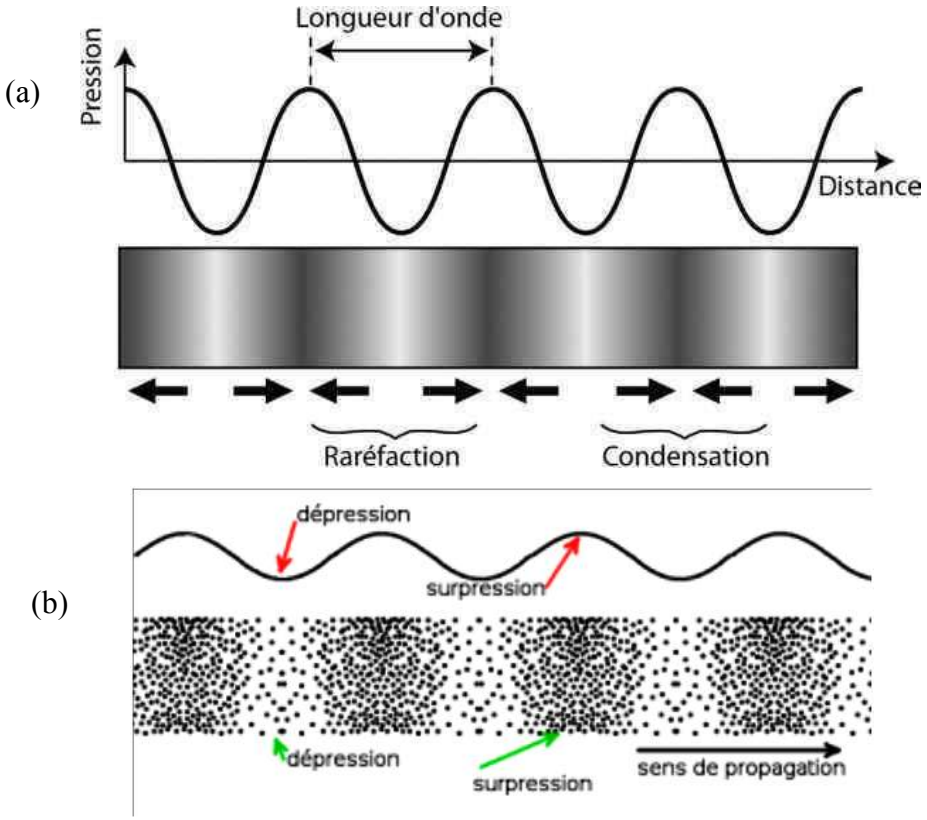


Figure (I.4) : Illustration de la propagation linéaire d'une onde (ultra)sonore.
a) Déplacement longitudinal et b) transversal engendré par l'onde ultrasonore

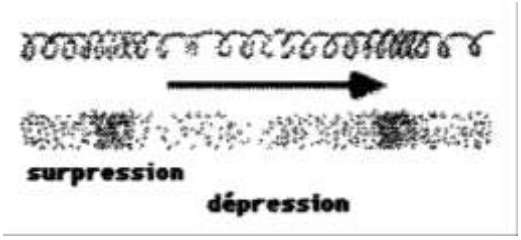


Figure (I.5) : Analogie avec un ressort.

Il s'agit d'une vibration longitudinale des molécules, phénomène prédominant lors de la propagation des ultrasons dans les tissus mous (figure I.5).

Le front de l'onde ou surface d'onde est une surface telle qu'en tous points la vibration sonore est la même.

Mais dans les milieux solides, comme l'os il se forme en plus une vibration transversale (déplacements de cisaillement) perpendiculairement à l'axe de propagation des ultrasons.

Cette propriété est utilisée en densitométrie ultrasonore [13].

I.3.3 Caractéristiques d'une onde ultrasonore

Plusieurs paramètres sont nécessaires pour caractériser cette onde ultrasonore.

I.3.3.1 Domaine temporel : fréquence (f) et période (T)

- La fréquence (f) est le nombre de fois où un phénomène périodique se reproduit par unité de temps ; c'est le nombre de variations de pression par seconde. La fréquence d'un son correspond au nombre de vibrations par seconde de la source sonore. Elle s'exprime en hertz ou en s–1. (1 HZ =1 cycle/seconde)
- La période (T) C'est le temps au bout duquel le phénomène de propagation se reproduit .Elle représente la durée d'un cycle. T [en s] = 1/f

Les sons sont classés en fonction de la vitesse de propagation des vibrations dans le milieu et de la détection possible par l'oreille [14]:

- infra-sons, (0 à 20 Hz),inaudibles, vitesse de propagation lente
- sons audibles,(20Hz à 20kHz), vitesse de propagation intermédiaire
- ultrasons, (20kHz à 1GHz), inaudibles, vitesse de propagation rapide
- Hypersons, > 1 GHz, vitesse de propagation rapide

Les ultrasons sont des ondes sonores longitudinales de haute fréquence. A l'exception de leurs fréquences élevées, ils ne diffèrent en rien, d'un point de vue physique, avec les ondes acoustiques de fréquences audibles [15].

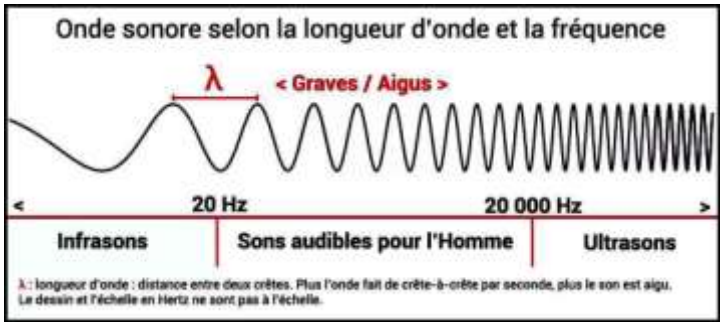


Figure (I.6) : Représentation d'une onde sonore selon la longueur d'onde et la fréquence.

I.3.3.2. Domaine spatial : longueur d'onde (λ) :

La longueur d'onde : est la distance séparant, à un instant donné, deux points du trajet de l'onde où la pression est identique. Pour un milieu donné, la longueur d'onde diminue quand la fréquence augmente.

$$\lambda = \frac{c}{f} \tag{I.17}$$

Nous constatons que plus λ est petite, plus la fréquence est grande, donc plus la résolution spatiale est bonne.

La résolution spatiale se définit la distance minimum séparant deux points distinguables l'un de l'autre. La résolution étant de l'ordre de la longueur d'onde elle est meilleure pour des fréquences de 10 MHz (0.15mm) que pour 1MHz (1.5mm) dans l'eau ou pour les tissus assimilés à de l'eau [15].

I.4 Interaction des ultrasons avec la matière

Dans un milieu hétérogène (être vivant): trois mécanismes distincts sont responsables de l'atténuation globale du faisceau [16].

- ❖ Absorption
- ❖ Réflexion- réfraction
- ❖ Diffusion.

I.4.1 Atténuation de l'amplitude de l'onde ultrasonore

L'amplitude d'une vibration qui se propage dans un milieu n'est pas constante durant sa propagation. Elle va s'atténuer jusqu'à disparaitre. Les causes d'atténuation sont essentiellement liées au matériau proprement dit, cependant la fréquence de la vibration est également un facteur qui influe sur cette atténuation, donc l'atténuation sera fonction de [17]:

- La fréquence de l'onde,
- Le matériau,
- La distance parcourue par l'onde.

Si à un instant donné, l'onde a une amplitude A_0 lorsqu'elle aura parcouru une distance, l'amplitude restante sera A.

$$A = A_0 e^{\alpha.d} \quad \text{(I.18)}$$

Avec :

A_0: Amplitude de l'onde émise,

A: Amplitude de l'onde qui a parcourue la distance d,

d: Distance parcourue,

α: Coefficient dépendant du matériau et de la fréquence au carré $\alpha = k.f^2$

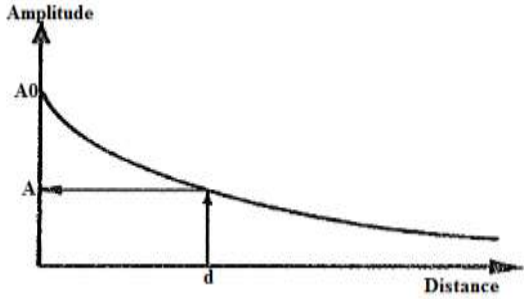


Figure (I.7) : Profile d'atténuation de l'amplitude de l'onde ultrasonore

I.5 La production des ultrasons

C'est en 1917 qu'est apparu le premier générateur d'ultrasons. Actuellement trois phénomènes sont utilisés; et dans les trois cas l'énergie électrique transportée par des courants alternatifs de fréquence élevée est transformée en énergie mécanique (oscillations d'un système mécanique) [19]

I.5.1 Les générateurs piézoélectriques

La piézoélectricité a été utilisée par le français Paul Langevin pour réaliser un générateur d'ultrasons dans lequel l'élément essentiel est constitué par une sorte de mosaïque de lamelles de quartz (minéral du groupe des silicates), d'orientation et d'épaisseur rigoureusement identiques, collées entre deux disques d'acier. L'ensemble est appelé un triplet. On relie les deux disques métalliques aux deux bornes d'une source de courant alternatif. Les lames de quartz présentent la propriété de se déformer à la même fréquence que celle de la tension qui leur est appliquée. Elles produisent des vibrations mécaniques qui sont transmises au milieu dans lequel se trouve l'appareil [20].

I.5.2 Les émetteurs magnétostrictifs

Les émetteurs magnétostrictifs constituent une application d'une propriété des corps ferromagnétiques qui consiste en une variation des dimensions du corps lorsque celui-ci est placé dans un champ magnétique variable (magnétostriction). Par exemple, on peut utiliser un empilement de tôles de nickel et le placer à l'intérieur de deux enroulements, l'un parcouru

par un courant continu (pour obtenir un champ magnétique constant convenable), l'autre parcouru par un courant alternatif (pour produire un champ magnétique variable). Le champ résultant permet d'obtenir une contraction relative assez importante et donc une vibration d'amplitude assez grande. Ces émetteurs sont très robustes (fortes) mais ils ne permettent pas de produire des ultrasons de fréquence supérieure à 50 000 Hz [20].

I.5.3 L'électrostriction

L'électrostriction est une propriété de tout matériau non-conducteur, ou diélectrique, qui consiste en une modification de sa forme sous l'effet d'un champ électrique appliqué. Elle est l'équivalent électrique de la magnétostriction. Alors l'électrostriction de certaines céramiques (titanate de baryum, zirconate de baryum ou de plomb) consiste en une variation des dimensions du corps lorsque celui-ci est placé dans un champ électrique variable. Donc l'utilisation de cette propriété permet d'obtenir des vibrations ultrasonores [20].

I.6 La détection des ultrasons

La détection et la mesure des ultrasons sont réalisées au moyen d'appareils divers. Les phénomènes piézoélectriques, magnétostrictifs et électrostrictifs étant réversibles, les dispositifs utilisés à l'émission peuvent constituer des récepteurs. Dans ce cas, les vibrations mécaniques engendrent une tension électrique de même fréquence que les ultrasons à détecter et c'est cette tension qui est étudiée. Les ultrasons exercent une pression de radiation qui devient appréciable quand l'énergie de rayonnement est suffisamment grande. La poussée qui est alors exercée sur une petite palette de surface connue peut être mesurée. Les ultrasons sont aussi détectés au moyen de différents dispositifs interférométriques ou d'appareils conçus pour étudier les ondes stationnaires [21].

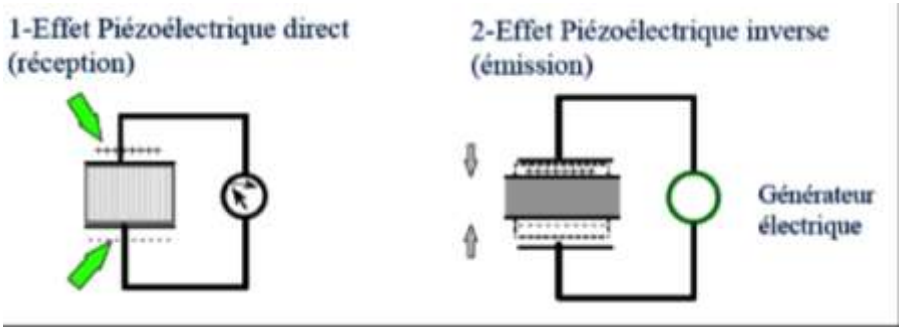


Figure (I.8) : Principe de la détection et la mesure des ultrasons

I.7 Capteur ultrasonore

I.7.1 Composition

Dans le sens large du mot, un traducteur une traduction du mot anglais "transducer", est un élément qui converti une énergie en une autre. Les traducteurs ultrasoniques

convertissent l'énergie électrique en un son à haute fréquence (équivalent à une vibration à haute fréquence) et vis versa [22].

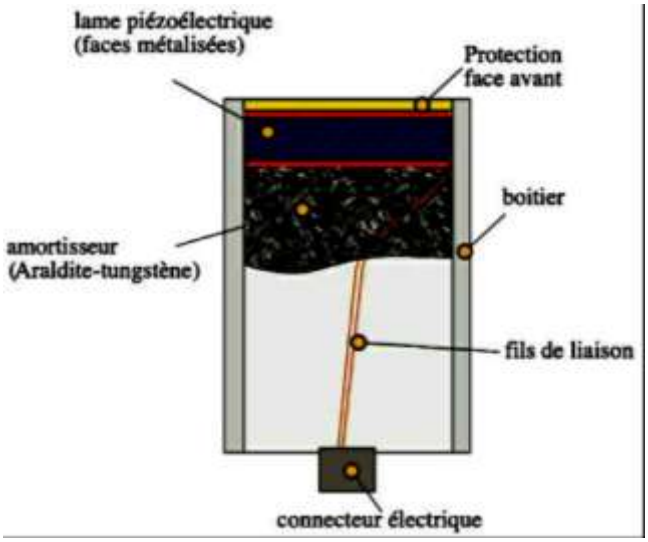


Figure (I.9) : les différents composants d'un traducteur ultrasonique.

Il existe plusieurs familles de palpeurs piézoélectriques adaptées aux différents modes d'utilisation : contrôle par contact ou en immersion, nature du produit à sonder, des défauts, etc.

Les 4 types de palpeurs les plus utilisés dans l'industrie sont les suivants.

- palpeur droit cylindrique,
- palpeur d'angle à sabot,
- palpeur droit focalisé,
- palpeur droit double avec émission et réception séparées (palpeurs SE).

Certaines applications particulières font appel à des palpeurs spéciaux tels que les palpeurs droits rectangulaires, les pal- peurs à focalisation cylindrique et les barrettes multi-palpeurs, nées dans le domaine médical et dont l'emploi se développe dans l'industrie [22].

I.7.2 Principe de fonctionnement des émetteurs et récepteurs à ultrasons :

Les émetteurs et récepteurs à ultrasons sont aussi appelés transducteurs piézoélectriques, car ils convertissent une énergie électrique en énergie mécanique et réciproquement. Le principe de fonctionnement est donc identique à celui des buzzer. Extérieurement les transducteurs d'émission (généralement, repérés par un « T » gravé) sont très semblables à ceux de réception (généralement, repérés par un « R » gravé) [23]:



Figure (I.10) : Emetteurs et récepteurs à ultrasons

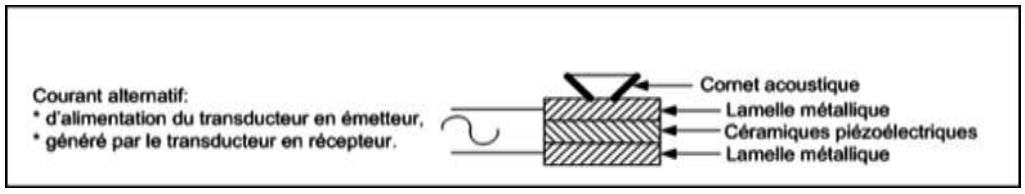


Figure (I.11) : Schéma de principe d'un transducteur US

I.7.3 Fonctionnement d'un émetteur US :

Le transducteur est alimenté par une tension alternative à une fréquence nominale de fonctionnement (souvent 40 KHz). Cette tension, est appliquée sur les lamelles métalliques ce qui génère une déformation mécanique des céramiques qui est transformée en pression acoustique appliquée à l'air ambiant, via le cornet acoustique [23].

I.7.4 Fonctionnement d'un récepteur US :

La pression acoustique (due à l'onde ultrasonore) reçue à travers l'air ambiant, via le cornet acoustique du récepteur US, est transformée en contrainte mécanique dans les céramiques qui génèrent des charges électriques sur les lamelles métalliques et donc une tension alternative à ses bornes.

Avec un Arduino, l'ensemble émetteur - récepteur d'onde ultrasonore ou le capteur ultrasonique le plus couramment utilisé est le HC-SR04 [24].

I.7.5 exemple d'application d'un capteur ultrasonique

I.7.5.1 Capteur ultrasonique HC-SR04

Le capteur HC-SR04 utilise les ultrasons pour déterminer la distance d'un objet. Il offre une excellente plage de détection sans contact, avec des mesures de haute précision et stables. Son fonctionnement n'est pas influencé par la lumière du soleil ou des matériaux sombres, bien que des matériaux comme les vêtements puissent être difficiles à détecter [24].



Figure (I.12) : Capteur ultrason HC-SR04

I.7.5.1.2 Caractéristiques et spécification du capteur

- Dimensions : 45 mm x 20 mm x 15 mm;
- Plage de mesure : 2 cm à 400 cm;
- Résolution de la mesure : 0.3 cm;
- Angle de mesure efficace : 15 °;
- Largeur d'impulsion sur l'entrée de déclenchement : 10 µs (Trigger Input Pulse width).

I.7.5.1.3 Broches de connexion :

- Vcc = Alimentation +5 V DC
- Trig = Entrée de déclenchement de la mesure (Trigger input)
- Echo = Sortie de mesure donnée en écho (Echo output)
- GND = Masse de l'alimentation

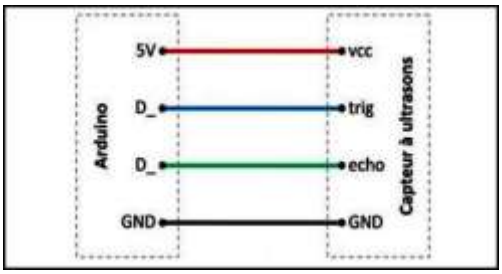


Figure (I.13): Broches de connexion de capteur a ultrason avec arduino

I.7.5.1.4 Spécifications et limites :

Tableau (I.1) : Spécifications et limites du capteur ultrason HC-SR04 [24,25] :

Paramètre	Min	Type	Max	Unité
Tension d'alimentation	4.5	5.0	5.5	V
Courant de repos	1.5	2.0	2.5	mA
Courant de fonctionnement	10	15	20	mA
Fréquence	-	40	-	kHz

I.7.5.1.5 Fonctionnement :

Pour déclencher une mesure, il faut présenter une impulsion "High" (5 V) d'au moins 10 µs sur l'entrée "Trig". Le capteur émet alors une série de 8 impulsions ultrasoniques à 40 kHz, puis il attend le signal réfléchi. Lorsque celui-ci est détecté, il envoie un signal "High" sur la sortie "Echo", dont la durée est proportionnelle à la distance [25].

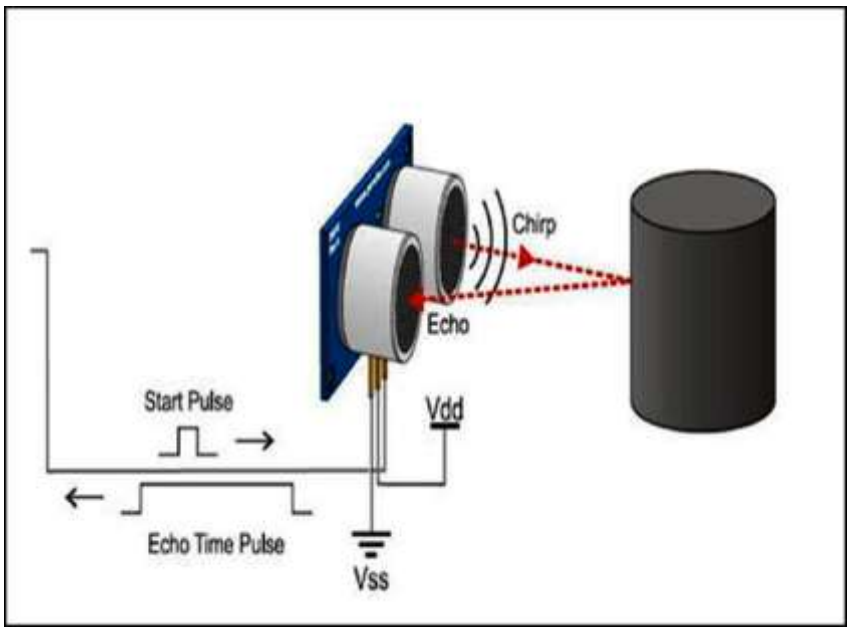


Figure (I.14) : Fonctionnement du capteur ultrason HC-SR04

Le principe de fonctionnement est résumé sur les étapes suivant [25]:

1. Déclarer la broche de l'Arduino sur laquelle est connectée la broche SIG du capteur en sortie numérique et la maintenir à l'état bas pendant 2 µs,
2. Envoyer un signal numérique à l'état haut sur l'émetteur (sur la broche SIG) pendant 5 µs,
3. Le capteur envoie automatiquement des impulsions d'ultrasons à 40 kHz,
4. Déclarer la broche de l'Arduino sur laquelle est connectée la broche SIG du capteur en entrée numérique, afin de pouvoir recevoir le signal du récepteur d'ultrasons.
5. A la fin des impulsions, la broche SIG du capteur passe à l'état haut,
6. Si le signal revient et est détecté par le récepteur, la broche SIG du capteur passe à l'état bas. La durée de l'état haut du signal SIG correspond au temps entre l'émission des ultrasons et leur réception.

Le principe de fonctionnement est résumé sur le schéma suivant :

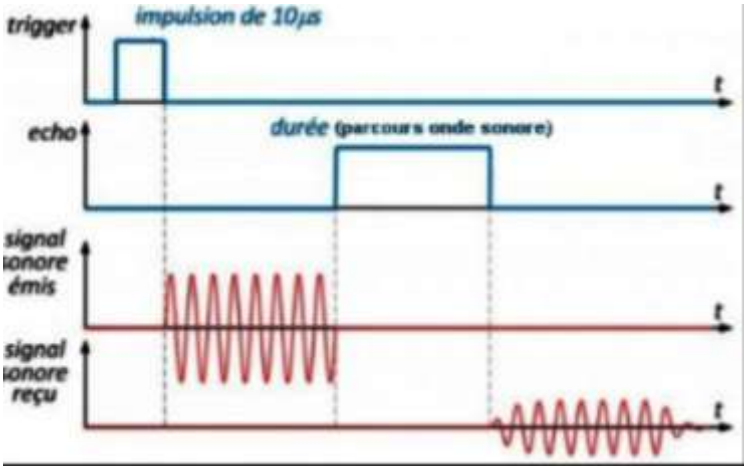


Figure (I.15) : Signal de sortie sur Trigger.

I.7.5.1.6 Calcul de la distance de l'objet

La formule couramment utilisée dans les programmes Arduino permettant de calculée la distance entre le capteur et un obstacle est :

Distance $_{\text{capteur- obstacle}}$ (en cm) = durée propagation (en µs) / 58

En effet, pour cela, on suppose que la vitesse des ultrasons dans l'air est de V = 340 m.s-1, la distance parcourue, d (en m), par l'onde sonore pendant la durée, Dt (en s), est alors :

$d_{\text{parcours onde sonore}} = V_{\text{ultrasons}}$ x Dt

Soit : Distance $_{\text{capteur- obstacle}}$ (en m) = d $_{\text{parcours onde sonore}}$ /2 = $V_{\text{ultrasons}}$ x Dt / 2

Distance $_{\text{capteur- obstacle}}$ (en m) = 340 x Dt / 2

Distance $_{\text{capteur- obstacle}}$ (en cm) = 34000 x Dt (en µs) /2000000 = 17x Dt (en µs) / 1000

Distance $_{\text{capteur- obstacle}}$ (en cm) = Dt (en µs) / 58,82

- La formule d = durée/58 cm figure aussi dans le manuel d'utilisation du HC-SR04 car la fraction 17/1000 est égale à 1/58.8235. Elle donne cependant des résultats moins précis.
- A grande distance, la surface de l'objet à détecter doit mesurer au moins 0.5 m².

I.7.5.2 Capteur ultrasonique Grove 101020010

Le capteur ultrasonique HC-SR04 dispose de 2 broches différentes pour l'émission et la réception des ultrasons. Il existe également des capteurs ultrasoniques à une seule broche, comme le Grove 101020010 [25].



Figure (I.16) : Capteur ultrasonique Grove 101020010

I.7.5.2.1 Caractéristiques

Le capteur est composé d'un émetteur d'ultrasons, d'un récepteur et du circuit de commande. Il est généralement utilisé pour mesurer des distances entre le capteur et un obstacle [25].

- Dimensions : 50 mm x 25 mm x 16 mm
- Plage de mesure : 2 cm à 350 cm
- Résolution de la mesure : 1 cm
- Angle de mesure efficace : 15 °

I.7.5.2.2 Broches de connexion

- Vcc = Alimentation +5 V DC
- SIG = Entrée émetteur et sortie récepteur d'impulsion d'ultrasons
- GND = Masse 0V
- NC = Non connectée

I.7.5.2.3 Spécifications et limites

- Tension d'alimentation : 3,2 V à 5,2 V
- Courant de fonctionnement : 8 mA
- Fréquence des ultrasons : 40 kHz
- Température de fonctionnement : 10 - 60 °C

I.7.5.2.4 Principe de fonctionnement :

1. Déclarer la broche de l'Arduino sur laquelle est connectée la broche SIG du capteur en sortie numérique et la maintenir à l'état bas pendant 2 µs,
2. Envoyer un signal numérique à l'état haut sur l'émetteur (sur la broche SIG) pendant 5 µs,
3. Le capteur envoie automatiquement des impulsions d'ultrasons à 40 kHz,

4. 4. Déclarer la broche de l'Arduino sur laquelle est connectée la broche SIG du capteur en entrée numérique, afin de pouvoir recevoir le signal du récepteur d'ultrasons.
5. A la fin des impulsions, la broche SIG du capteur passe à l'état haut,
6. Si le signal revient et est détecté par le récepteur, la broche SIG du capteur passe à l'état bas. La durée de l'état haut du signal SIG correspond au temps entre l'émission des ultrasons et leur réception.

Le principe de fonctionnement est résumé sur le schéma suivant :

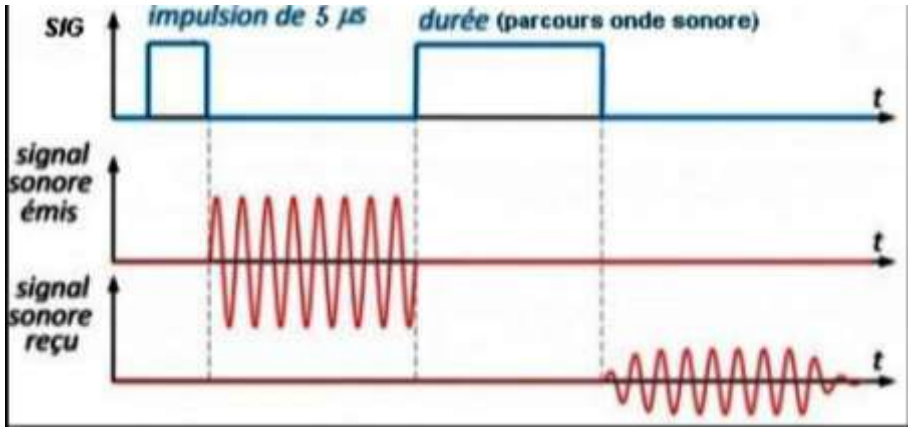


Figure (I.17) : La durée de l'état haut du signal SIG correspond au temps entre l'émission des ultrasons et leur réception

Le principe de calcul de la distance entre ce capteur et un obstacle reste le même.

Par défaut, les programmes sont conçus pour un capteur ultrasonique disposant de deux broches différentes pour l'émission et la réception des ultrasons. Pour utiliser un capteur ultrasonique à une broche commune pour l'émission et la réception, il suffit de modifier le programme et de donner la même valeur aux variables TRIGGER_PIN et ECHO_PIN [25].

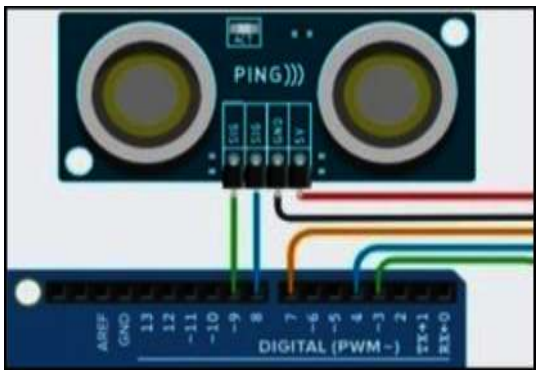


Figure (I.18) : Capteur à deux broches différentes Trigg et Echo.



Figure (I.18) : Capteur à une broche commune pour Trigg et Echo

I.8 Applications du capteur à ultrasons:

- Détection et évitement d'obstacles dans les robots.
- Mesure de distance dans les projets de bricolage et éducatifs.
- Détection d'objets dans les systèmes d'automatisation.
- Surveillance du niveau d'eau ou de liquide (sans contact).
- Prototypes d'aide au stationnement et de détection de proximité.
- Poubelles intelligentes et système de couvercle automatique.
- Configurations de sécurité et de détection d'intrusion.
- Mesure de position industrielle et d'espacement.
- Projets d'électronique interactive et d'IoT.
- Systèmes de déclenchement basés sur le mouvement.

I.9 Avantages du capteur à ultrasons HC-SR04 :

- Faible coût.
- Montage et code assez simples.
- Portée correcte pour de petits projets, jusqu'à environ 4 m.
- Consommation faible, pratique pour les systèmes sur batterie.

I.10 Inconvénients du capteur à ultrasons HC-SR04 :

- Précision moins bonne dans certains environnements.
- Sensible aux obstacles hors de l'axe du capteur, à cause du cône de mesure limité.
- Peut être perturbé quand on utilise plusieurs capteurs ensemble.
- Mesure moins fiable si l'environnement est bruyant ou complexe.

Points importants :

Le HC-SR04 est un capteur de distance sans contact.

Il est souvent utilisé avec Arduino pour éviter des obstacles ou mesurer un niveau.
Son fonctionnement peut être moins fiable sur certains matériaux absorbants comme le textile ou sur des surfaces difficiles à réfléchir [26].

I.11 Conclusion

Dans ce chapitre, nous avons vu les Propriétés des ondes sonores ou acoustiques, la nature des ondes ultrasonore, les paramètres qui sont nécessaires pour caractériser cette onde ultrasonore. Ainsi les Mécanismes physiques des ultrasons.

Chapitre II:
Dispositif programmable Arduino et le protocole de communication Firmata

Chapitre II:Dispositif programmable Arduino et protocole de communication Firmata

II.1 Introduction :

Le dispositif programmable Arduino est une plateforme matérielle open-source conçue pour simplifier le développement et le contrôle de systèmes électroniques embarqués. Composée d'une carte microcontrôleur programmable et d'un environnement de développement dédié, Arduino permet aux utilisateurs, qu'ils soient débutants ou experts, de concevoir rapidement des applications intégrant des capteurs, des actionneurs et diverses interfaces physiques. Sa modularité et son accessibilité ont fait d'Arduino un outil incontournable pour l'éducation, le prototypage et les projets industriels légers.

Le protocole de communication Firmata complète cette plateforme en offrant un standard permettant à un ordinateur hôte de contrôler à distance les fonctions d'un microcontrôleur Arduino via une liaison série. Firmata établit une couche d'abstraction qui facilite la gestion des entrées/sorties numériques et analogiques, la modulation de largeur d'impulsion, et la communication avec des périphériques comme les servomoteurs ou les bus I2C et SPI. Cette approche simplifie le développement logiciel en déchargeant le programmeur des détails bas niveau du matériel, rendant ainsi l'interaction entre logiciel et matériel plus directe et efficace.

En associant Arduino et Firmata, il devient possible de concevoir et de piloter des systèmes électroniques complexes avec une grande flexibilité, tout en réduisant considérablement la complexité technique nécessaire au contrôle matériel.

Dans ce chapitre, nous montrons qu'Arduino et les protocoles Firmata permettent une communication sans effort entre le logiciel et le matériel, permettant aux utilisateurs de construire et de gérer un large éventail de projets électriques avec simplicité et adaptabilité.

II.2 Définition du module Arduino

L'Arduino est une carte à microcontrôleur c'est-à-dire un ensemble constitué d'un Microcontrôleur et toute la circuiterie nécessaire à son bon fonctionnement montés sur une seule carte de taille très réduite. La carte est accompagnée d'un logiciel (EDI) multiplateforme servant à développer n'importe quelle application à implanter sur la carte avec un langage de programmation dédié que l'on peut appeler le « langage Arduino » qui est un mélange de C et de C++ restreints aux seules possibilités de la carte [27].

Les concepteurs de l'Arduino ont fixé leur choix sur les microcontrôleurs fabriqués par la société Atmel, et plus précisément sur la famille AVR de ce fabricant. Ce choix des AVR par les concepteurs de l'Arduino est fait sur plusieurs critères tels que l'excellent compromis prix/puissance que représentent les microcontrôleurs AVR et, même s'il ne s'agit que de

microcontrôleurs à unité centrale 8 bits, ils permettent de développer des applications très intéressantes.

II.3 Les gammes de cartes Arduino :

Arduino propose plusieurs modèles de cartes microcontrôleurs adaptées à une large variété d'applications, allant des projets simples aux systèmes embarqués complexes. Ces cartes se distinguent principalement par leur microcontrôleur, leur nombre d'entrées/sorties, leur mémoire et leurs fonctionnalités spécifiques [27] :

II.3.1 Arduino Uno :

- Microcontrôleur : ATmega328P.
- Fréquence : 16 MHz.
- Entrées/Sorties : 14 numériques (6 PWM), 6 analogiques
- Mémoire : 32 Ko Flash, 2 Ko SRAM
- Tension de fonctionnement : 5 V.
- Usage typique : apprentissage, prototypage basique.
- Remarque : carte la plus populaire et polyvalente, idéale pour débutants.

II.3.2 Arduino Mega 2560 :

- Microcontrôleur : ATmega2560
- Fréquence : 16 MHz.
- Entrées/Sorties : 54 numériques (15 PWM), 16 analogiques
- Mémoire : 256 Ko Flash, 8 Ko SRAM .
- Tension de fonctionnement : 5 V.
- Usage typique : projets complexes nécessitant beaucoup d'E/S

II.3.3 Arduino Leonardo :

- Microcontrôleur : ATmega32U4 .
- Fréquence : 16 MHz.
- Entrées/Sorties : 20 numériques (7 PWM), 12 analogiques.
- Mémoire : 32 Ko Flash, 2.5 Ko SRAM.
- Tension de fonctionnement : 5 V.
- Usage typique : interfaces HID (clavier, souris)

II.3.4 Arduino Due :

- Microcontrôleur : ARM Cortex-M3 (Atmel SAM3X8E).
- Fréquence : 84 MHz.
- Entrées/Sorties : 54 numériques (12 PWM), 12 analogiques.

- Mémoire : 512 Ko Flash, 96 Ko SRAM.
- Tension de fonctionnement : 3.3 V.
- Usage typique : applications avancées nécessitant plus de puissance de calcul.

II.3.5 Arduino Nano :

- Microcontrôleur : ATmega328P.
- Fréquence : 16 MHz.
- Entrées/Sorties : 14 numériques (6 PWM), 8 analogiques.
- Mémoire : 32 Ko Flash, 2 Ko SRAM.
- Tension de fonctionnement : 5 V.
- Usage typique : projets compacts, intégration sur breadboard.

II.3.6 Arduino Nano 33 BLE :

- Microcontrôleur : nRF52840 Cortex-M4.
- Fréquence : 64 MHz.
- Entrées/Sorties : 14 numériques, 8 analogiques.
- Mémoire : 1 Mo Flash, 256 Ko SRAM.
- Tension de fonctionnement : 3.3 V.
- Usage typique : applications IoT avec connectivité Bluetooth Low Energy

II.4 Le logiciel Arduino IDE :

Le logiciel Arduino IDE fonctionne sur Mac, Windows et Linux. C'est grâce à ce logiciel que nous allons créer, tester et envoyer les programmes sur l'Arduino.

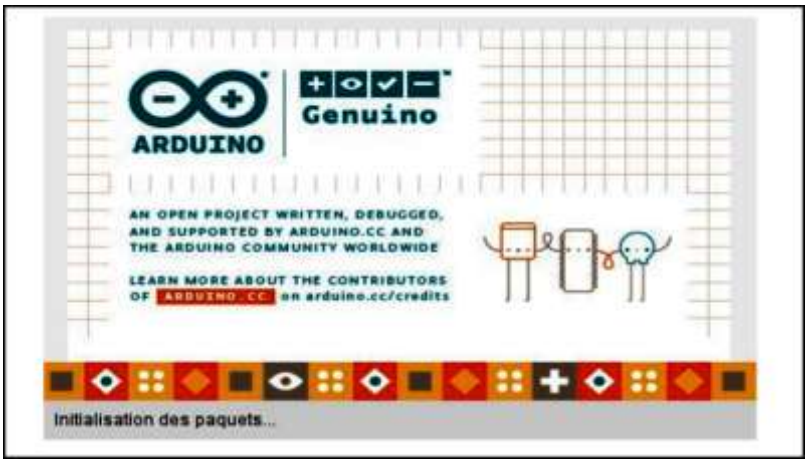
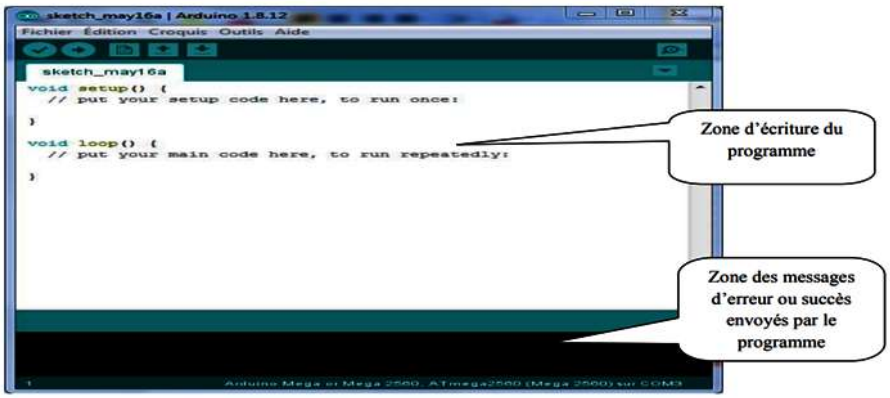


Figure (II.1) : Présentation de logiciel arduino.

C'est un logiciel de programmation par code, code qui contient une cinquantaine de commandes différentes. A l'ouverture, l'interface visuelle du logiciel ressemble à ceci : des boutons de commande en haut, une page blanche vierge et une bande noire en bas [28].



Figure(II.2): Logiciel Arduino IDE.

Logiciel IDE (Environnement de Développement Intégré) multi OS :

- édition du programme.
- compilation du programme.
- transfert du programme dans la carte via le port USB.
- Diverses Cartes Arduino.



Figure(II.3): Barre d'outils

- Bouton 1 : Ce bouton permet de vérifier le programme, il actionne un module qui cherche les erreurs dans le programme.
- Bouton 2 : Envoi du programme sur l'Arduino
- Bouton 3 : Créer un nouveau fichier
- Bouton 4 : Ouvrir un fichier existant.
- Bouton 5 : Enregistrer un fichier.
- Bouton 6 : Chercher sur un menu Option

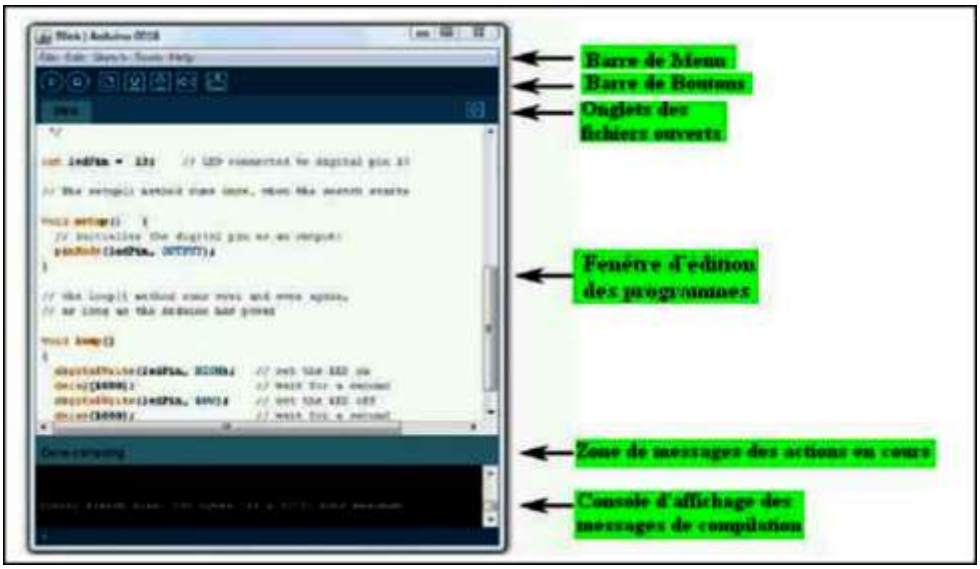
Un programme utile pour l' Arduino est une suite d'instructions élémentaires sous forme textuelle, ligne par ligne. La carte lit puis effectue les instructions les unes après les autres, dans l'ordre défini par les lignes de code.

II.5 Description du logiciel Arduino :

Le logiciel Arduino a pour fonctions principales :

- ✓ le pouvoir d'écrire et compiler des programmes pour la carte Arduino.
- ✓ de se connecter avec la carte Arduino pour y transférer les programmes.
- ✓ de communiquer avec la carte Arduino.

Cet espace de développement intégré (EDI) dédié au langage Arduino et à la programmation des cartes Arduino [28].



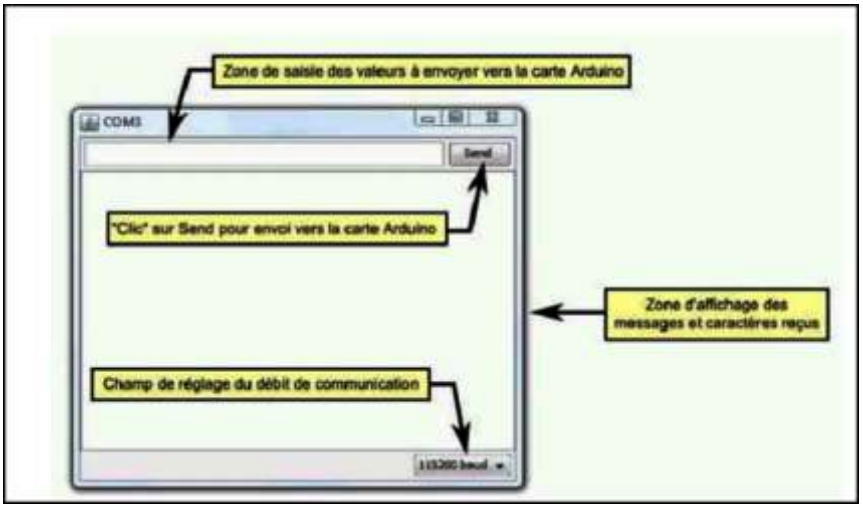
Figure(II.4): Les différentes parties de la fenêtre principale du logiciel Arduino.

Il comporte :

- ❖ une BARRE DE MENUS : comme pour tout logiciel une interface graphique (GUI).
- ❖ une BARRE DE BOUTONS : cette barre nous donne un accès direct aux fonctions essentielles du logiciel, elle importe au logiciel une simplicité d'utilisation.
- ❖ un EDITEUR (à coloration syntaxique) : c'est dans cette espace où on va écrire notre programme, il dispose aussi des onglets de navigation.
- ❖ une ZONE DE MESSAGES : permet d'afficher et d'indique l'état des actions en cours (ex : vérification, téléverssement).
- ❖ une CONSOLE TEXTE : elle permet d'afficher les messages concernant le résultat de la compilation du programme (il nous indique s'il y a des erreurs).
- ❖ un MONITEUR SERIE : ce moniteur est utilisé pour la visualisation ou les données (message, valeurs, caractères) transmises vers l'ordinateur via le câble USB connecté à la carte Arduino et d'envoyer des caractères vers la carte Arduino.

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

et corriger ses programmes. Nous l'avons utilisé dans notre projet pour tester si la carte fait la conversion analogique/numérique pour le signal de l'ECG, aussi pour vérifier si la carte transmis ces valeurs convertis, la valeur de la température et la valeur de l'humidité [29].



Figure(II.5): Fenêtre associée au moniteur série.

II.5.1 Description de la barre des boutons :

La barre des boutons nous permet de vérifier ka syntaxe, le transfert des programmes, aussi la création, l'ouverture et la sauvegarde d'un code écrit, elle permet aussi d'ouvrir le moniteur série. Elle dispose de sept bouton ou icones [28,29] :



Figure(II.6): Structure de la barre des boutons.



Vérifier/compiler**:** ce bouton nous permet de vérifier s'il y a des erreurs dans le programme en sélectionnant la ligne ou se trouve l'erreur.



Stop: ce bouton oblige le moniteur série ou les autres boutons activés de s'arrêter.



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



Ouvrir: il suffit de cliquer sur ce bouton pour accéder à la liste de tous les programmes qui se trouvent dans le «livre de programmes».Cliquer sur l'un l'un

des programmes l'ouvre dans fenêtre courante.



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



Transférer ou téléverser vers la carte : ce bouton permet de compiler votre code Et de le transférer vers la carte Arduino.



Moniteur série : pour ouvrir la fenêtre du moniteur série il suffit d'appuyer sur ce bouton.

II.5.2 Description de la barre des menus :

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



Figure(II.7): La barre des menus.

Cette barre contient les icones suivantes [28]:

- File (Fichier): ce menu contient les différentes options de création, d'ouverture, de sauvegarde, d'impression d'un programme, ou l'ouverture d'un exemple parmi les exemples qui accompagnent le logiciel Arduino.
- Edite(Editer): ce menu contient les options de copier/coller, sélection, et les options de recherche. • Sketch (Programme ou séquence): ce menu contient les différentes fonctions de la barre des boutons, ainsi que les options d'ajout de librairies ou de fichiers.
- Tools (Outils): c'est dans ce menu qu'on sélectionne le type de carte à programmer, et le port série utilise ainsi que la fonction de chargement du boot loader dans l'ATmega.
- Help (Aide): ce menu est fait pour donner de l'aide concernant les différents problèmes rencontrés au niveau du logiciel Arduino.

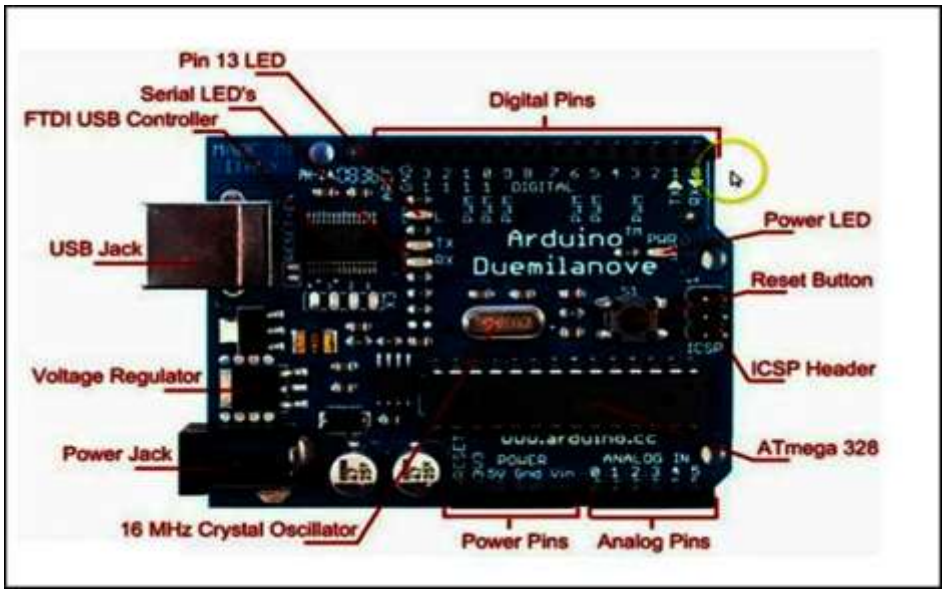
II.5.2 Les différents types de l'ARDUINO :

Il existe trois types de cartes ARDUINO :

- Les officielles qui sont fabriquées en Italie par le fabricant officiel : Smart Projects.

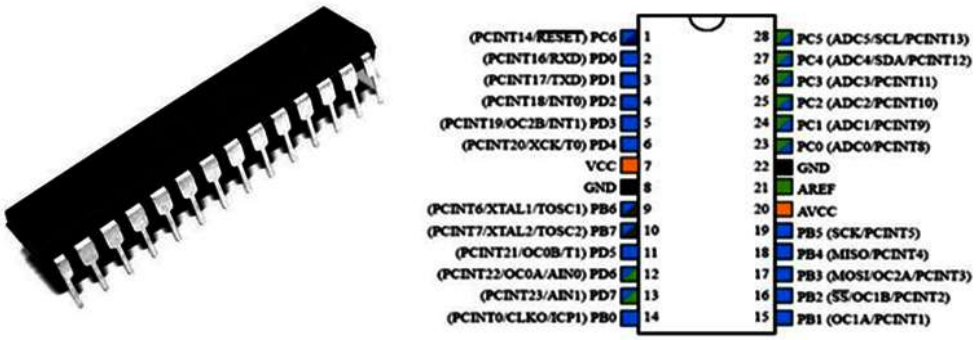
- Les compatibles qui ne sont pas fabriqués par Smart Projects, mais qui sont totalement compatibles avec les Arduino officielles.
- Les autres fabriquées par diverse entreprise et commercialisées sous un nom diffèrent (Freeduino, Seeduino, Femtoduino, ...)

II.6 Constitution de la carte Arduino UNO :



Figure(II.8): Les différentes parties de la carte Arduino UNO.

II.6.1Microcontrôleur principal :



Figure(II.9): Microcontrôleur ATMEGA328p.

Ce microcontrôleur fonctionne à une fréquence de 16 MHz et possède une mémoire flash de 32 Ko pour stocker le programme, 2 Ko de SRAM pour les données temporaires, et 1 Ko d'EEPROM pour le stockage non volatile.

II.6.2Alimentation :

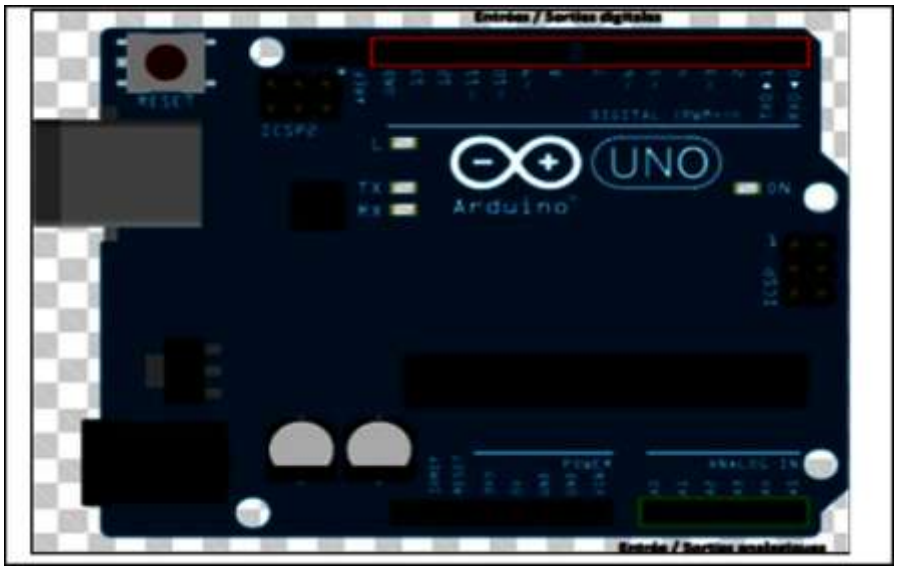
La carte peut être alimentée via un câble USB ou par une source externe (7 à 12 V recommandé). Un régulateur de tension intégré stabilise l'alimentation à 5 V nécessaire



Figure(II.10): L'alimentation de la carte arduino uno.

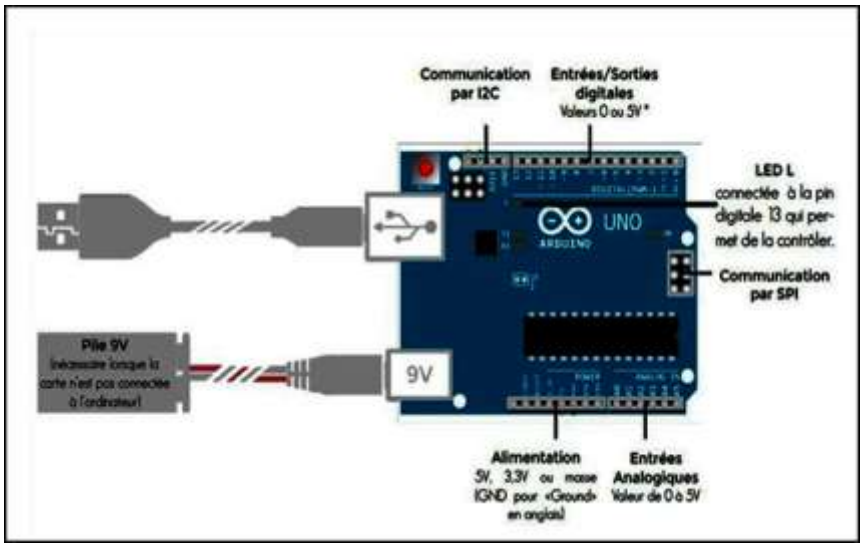
Pour le fonctionnement des composants. La carte dispose aussi de broches dédiées à l'alimentation (5 V, 3.3 V, GND) pour alimenter des modules externes [29].

1. Entrées/Sorties numériques : Elle offre 14 broches numériques configurables en entrée ou sortie, dont 6 peuvent générer des signaux PWM (modulation de largeur d'impulsion), utiles pour le contrôle de moteurs, LED, etc.
2. Entrées analogiques : doté de six broches analogiques convertissent des signaux continus en valeurs numériques (10 bits de résolution), permettant la lecture de capteurs analogiques comme les potentiomètres ou capteurs de température.



Figure(II.11): Entrées/Sorties numériques et les entrées analogiques.

3. Communication série : Un port USB permet la communication entre le microcontrôleur et un ordinateur via un convertisseur USB-série (souvent un ATmega16U2 sur les versions récentes). La carte supporte également des protocoles de communication standards tels que UART, SPI et I2C [30].



Figure(II.12): Communication série.

4. Quartz oscillateur : Un oscillateur à quartz de 16 MHz fournit l'horloge système garantissant la synchronisation précise des opérations du microcontrôleur.
5. Bouton Reset : Un bouton physique permet de réinitialiser manuellement le microcontrôleur, relançant le programme chargé.
6. LEDs intégrées : On distingue les LED « ON » indiquant la mise sous tension, les LED connectée à la broche 13 pour des tests rapides et LED de communication RX/TX indiquant l'activité série.

II.7 Scripts Python :

Scripts Python Python est un langage de script orienté objet, publié en 1991. Il a été développé par Guido van Rossum de l'Institut national de recherche en mathématiques et en informatique d'Amsterdam. Python est rapidement devenu l'un des langages de programmation les plus populaires au monde. Il est aujourd'hui particulièrement utilisé pour le calcul éducatif et scientifique, et comme langage de programmation le plus populaire en science des données [31].

Les raisons pour lesquelles Python est populaire et que tout le monde devrait envisager de l'apprendre :

- Il est open source, gratuit et largement disponible, bénéficiant d'une vaste communauté open source.
- Il est plus facile à apprendre que des langages comme C, C++ et Java, ce qui permet aux développeurs débutants comme professionnels de se familiariser rapidement avec le langage.
- Il est plus facile à lire que de nombreux autres langages de programmation populaires.
- Il est populaire en intelligence artificielle, qui connaît une croissance fulgurante, en partie grâce à sa relation privilégiée avec la science des données.
- Il améliore la productivité des développeurs grâce à de vastes bibliothèques standard et des bibliothèques open source, permettant aux programmeurs d'écrire du code plus rapidement et d'effectuer des tâches complexes avec un minimum de code.

En mode interactif, les lignes d'instructions ne sont plus accessibles une fois exécutées. Mais il est possible d'écrire et de conserver un programme (un script) à l'aide d'un éditeur, afin de pouvoir l'exécuter à volonté ou le modifier ultérieurement. Il existe de nombreux éditeurs de scripts Python qui incluent également un interpréteur pour exécuter les programmes. On parle alors d'IDE (environnement de développement intégré). Il s'agit d'un environnement de programmation complet se présentant sous la forme d'une application. Il se compose généralement d'un éditeur de code, d'un interpréteur, d'un débogueur… On peut citer Pycharm, Spider… Mais pour une introduction à la programmation en Python, l'utilisation de l'interpréteur IDLE, qui permet également l'édition de scripts, est tout à fait adaptable [32].

II.8Programmation avancée : Firmata, Firmata Express et PySerial :

II.8.1 protocole de contrôle à distance :

Firmata est un protocole de communication série standardisé qui permet à un ordinateur hôte de contrôler à distance une carte Arduino sans nécessiter la programmation embarquée classique. Il fonctionne en transmettant des commandes structurées via USB (ou autre liaison série), permettant d'activer ou de lire les entrées/sorties numériques et analogiques, gérer la modulation PWM, les servomoteurs, et même les bus I2C et SPI . Grâce à Firmata, les développeurs peuvent piloter la carte Arduino depuis des langages de haut niveau comme Python, JavaScript, ou Processing, accélérant le développement et simplifiant l'interfaçage matériel-logiciel [33].

II.8.2 Firmata Express : version allégée pour le prototypage :

Firmata Express est une variante simplifiée du protocole Firmata. Cette version réduit le nombre d'instructions et limite les fonctionnalités aux plus couramment utilisées, comme les lectures et écritures analogiques et numériques.

L'objectif est d'améliorer la performance en limitant la surcharge et de rendre le protocole plus accessible aux débutants et aux environnements éducatifs, où la simplicité prime sur la richesse fonctionnelle complète [33].

II.8.3 PySerial : interface Python pour la communication série :

Pour exploiter Firmata (et Firmata Express) depuis Python, la bibliothèque PySerial est essentielle. Elle offre une interface simple pour ouvrir, lire et écrire des données sur un port série (USB) [33].

L'association PySerial + Firmata permet de :

- Établir une connexion avec la carte Arduino via USB.
- Envoyer des commandes Firmata pour contrôler les broches (digitales/analogiques).
- Recevoir des données de capteurs en temps réel. Cette approche est idéale pour développer des applications interactives, de visualisation de données, ou des systèmes de contrôle embarqués pilotés par ordinateur.

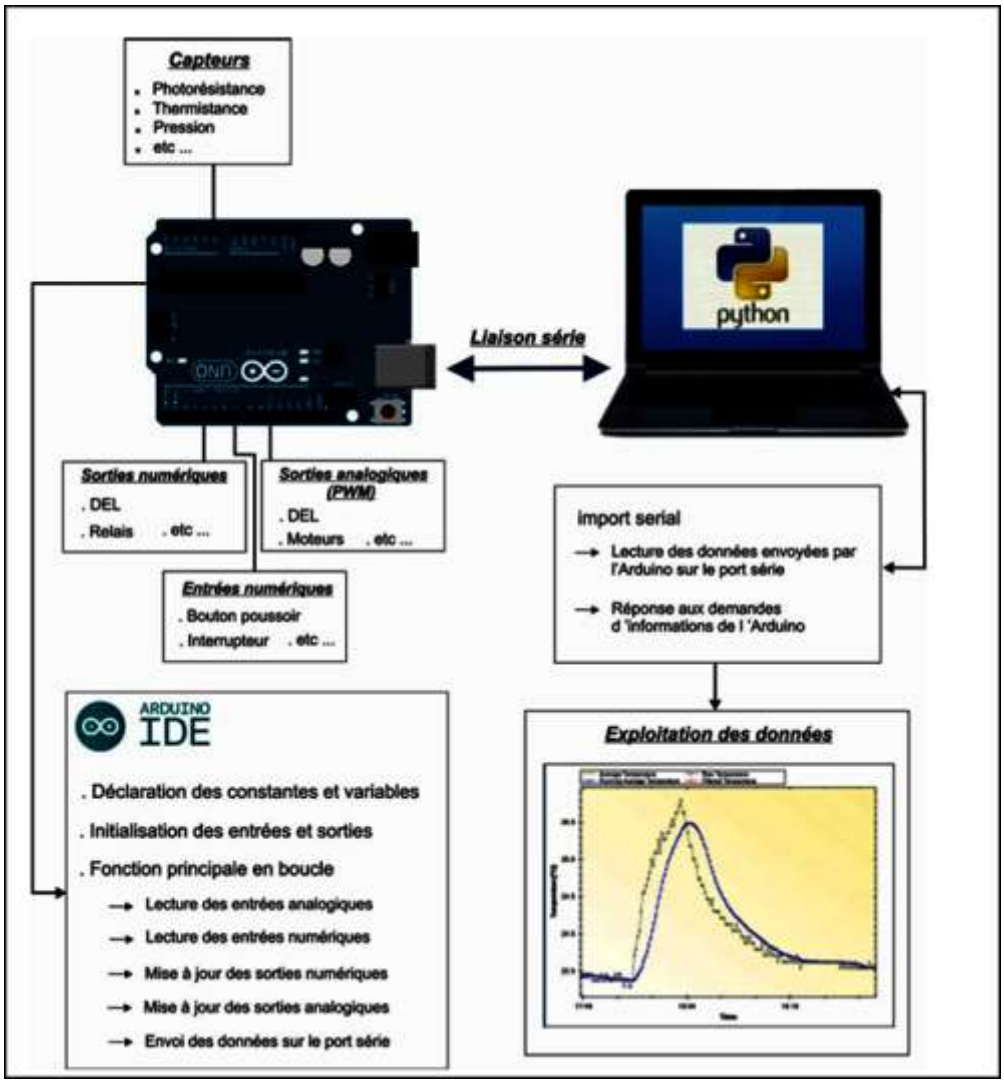
II.8.4 Communication Arduino – Python :

Les méthodes de communications entre un arduino et un programme Python sont résumées à l'aide des schémas suivant [33,34]:

II.8.4.1 Communication Arduino - Python via le port série avec Pyserial :

L'Arduino envoie des données sur le port série par la fonction "Serial.print()"

- Le programme Python réceptionne les données de la liaison série, envoyées par l'Arduino, à l'aidede la fonction "readline()"
- Le programme Python envoie des données sur le port série avec la fonction "write()"
- L'arduino réceptionne les données envoyées par le programme Python sur la liaison série grâce à lafonction "Serial.read()"

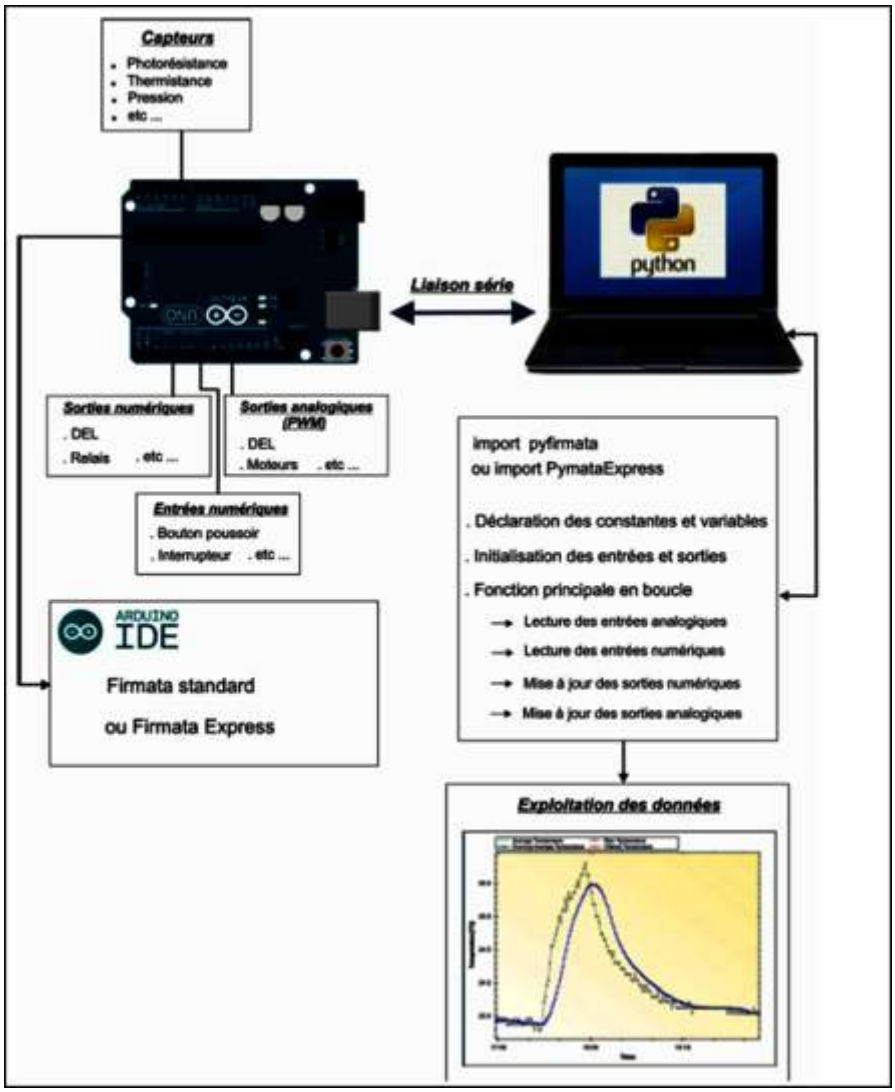


Figure(II.13): Communication Arduino - Python via le port série avec Pyserial.

II.8.4.1 Communication Arduino-Python via le protocole de communication Firmata :

Pour contrôler l'Arduino via le protocole de communication Firmata standard, le programme Python utilise la bibliothèque "PyFirmata" (le sketch "Firmata standard" doit être téléversé dans la mémoire de l'arduino au préalable) [34].

Toutes les fonctions utiles à l'utilisation de "PyFirmata" (fonction de déclaration des entrées et sorties, d'itérateur, d'écritures…) que nous avons définies jusqu'à présent ont été regroupées dans un fichier Python, nommé "PyFirmataDef.Py" que l'on peut importer dans tous les programmes, à condition que le fichier des fonctions soit dans le même dossier que le fichier du programme, avec l'instruction : from PyFirmataDef import *



Figure(II.14): Communication Arduino-Python via le protocole de communication Firmata.

II.9 Conclusion :

Ce chapitre présente une présentation complète de l'Arduino Uno, une carte d'acquisition fréquemment utilisée. Nous avons expliqué clairement le processus de décision et examiné en détail les options disponibles. Nous avons ensuite clarifié les éléments essentiels d'Arduino, notamment les composants matériels et la programmation, avec plus de précision, ainsi le principe de fonctionnement de la carte Arduino tout en intégrant ses caractéristiques uniques. Nous avons également étudié le protocole Firmata, qui améliore l'environnement Arduino en permettant les connexions entre les microcontrôleurs et les ordinateurs hôtes, permettant ainsi un contrôle avancé du matériel via d'autres langages de programmation. Nous avons également analysé Firmata Express, une version modifiée de

Firmata qui réduit le jeu de commandes pour une utilisation plus simple, la rendant particulièrement adaptée aux applications pratiques et au prototypage rapide.

Chapitre III :
Conception et simulation du capteur ultrasonique

III.1 Introduction :

Dans ce chapitre, nous présentons les résultats expérimentaux réalisés à l'aide d'un système électronique basé sur une carte Arduino et le capteur ultrasonique HC-SR04.

Un montage électronique a été conçu et mis en œuvre afin de mesurer la célérité dans l'air expérimentalement en mesurant la durée de propagation Dt, de l'onde sonore, entre un émetteur et un récepteur situés à une distance d. On peut également utiliser un ensemble émetteur - récepteur d'onde ultrasonore placée devant un obstacle. C'est le principe de la mesure par écho ou du Sonar. Comme il est possible de mesurer une distance avec un Arduino et un capteur à ultrasons, nous allons dans ce travail, réaliser un détecteur d'obstacles qui déclenchera une alarme lumineuse et sonore quand le capteur est situé en dessous d'une distance, d, fixée.

Le dispositif utilisé comprend les composants essentiels suivants : le capteur ultrasonique HC-SR04 pour la détection de la distance, une carte Arduino UNO pour la collecte des données, une LED pour l'indication visuelle, ainsi qu'un bouton-poussoir pour le contrôle manuel de l'expérience.

Les données ont été collectées via un code Arduino spécifique, puis analysées et modélisées en Python, ce qui nous a permis d'étudier et de visualiser les relations entre la pression et les autres variables physiques comme le volume ou la profondeur.

Ce chapitre présentera en détail les résultats obtenus, accompagnés, en mettant l'accent sur la précision, la réactivité et la fiabilité du système dans différents contextes expérimentaux.

III.2 Ondes sonores :

Une onde sonore ou acoustique est une perturbation mécanique périodique qui se propage dans un milieu matériel fluide élastique (l'eau ou l'air par exemple), en s'éloignant de sa source. C'est une onde de compression-dilatation du milieu dans lequel elle se propage.
Par exemple, Une corde de guitare qui vibre, va transmettre sa vibration à l'air. Cette vibration va se propager dans l'air jusqu'à nos oreilles qui vont recevoir la vibration.
L'être humain peut entendre des sons dont les fréquences s'étalent de 20Hz à 20kHz environ [35].

III.2.1 Arduino et ondes sonores :

Comme une corde de guitare qui vibre et qui transmet sa vibration à l'air, pour produire un son avec un Arduino, il faut utiliser un matériel qui peut vibrer sur commande !

Pour cela on utilise un petit haut-parleur ou un buzzer (transducteur) piézo-électrique (communément appelé "piezo") connecté sur une des sorties de l'Arduino.

III.2.1.1 Principe de fonctionnement du haut-parleur

Le haut-parleur est constitué principalement :

- d'un aimant circulaire ;
- d'une bobine circulaire mobile placée autour d'un des pôles de l'aimant ;
- d'une membrane reliée à la bobine ;
- d'un support qui contient l'aimant, la bobine et la membrane.
- Les fils de la bobine sont connectés à la sortie du haut-parleur.

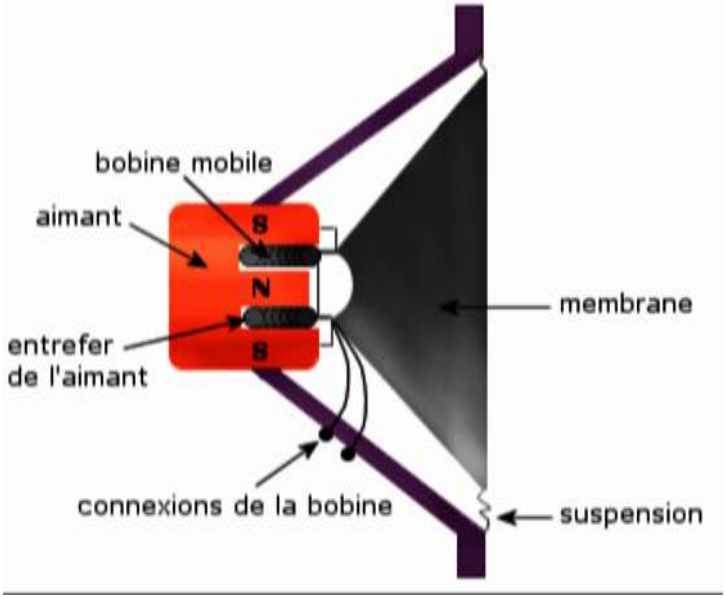


Figure (III.1) : Structure et principe de fonctionnement d'un haut-parleur électrodynamique

Le champ magnétique créé par l'aimant est perpendiculaire en tout point au courant qui circule dans chacune des spires.

Lorsqu'un courant continu circule dans les spires, il se créé une force appelée force de Laplace. Cette force n'existe que lorsque les spires sont plongées dans un champ magnétique.

La valeur de cette force est proportionnelle à l'intensité du courant I, à la valeur du champ magnétique B et à la longueur L du conducteur qui subit la force. On écrit : $F = I.L.B$ lorsque le champ magnétique et le courant sont perpendiculaires. F est exprimé en newton (N), I en ampère (A), L en mètre (m) et B en tesla (T). La direction et le sens de cette force dépendent de la direction et du sens du champ magnétique et du sens du courant dans les spires [36].

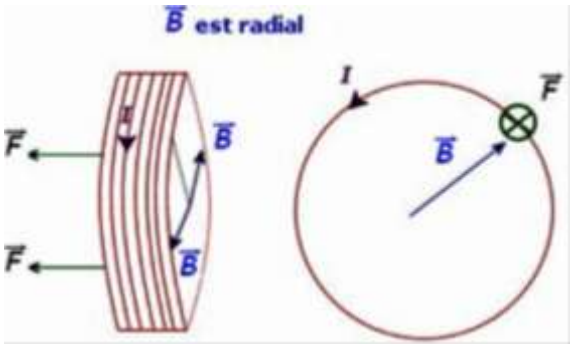


Figure (III.2) : Forces de Laplace exercées sur la bobine mobile dans le champ magnétique du haut-parleur

La sortie de l'Arduino est connectée à la bobine mobile, le courant électrique venant de l'Arduino la traverse, et sous l'action du champ magnétique intense de l'entrefer, va pousser en avant la membrane ou la tirer en arrière suivant la polarité de cette tension électrique.
En se déplaçant ainsi d'avant en arrière au rythme du signal électrique, la membrane crée une onde sonore qui se propage dans l'air [36].

III.2.1.2 Principe de fonctionnement d'un buzzer (transducteur) piézo-électrique



Figure (III.3) : Buzzer piézoélectrique utilisé pour la génération de signaux sonores

Un matériau piézoélectrique est une substance qui produit un courant électrique lorsqu'il est déformé. Et inversement, lorsqu'une tension électrique est appliquée sur la substance, une déformation a lieu [37].

III.2.1.3 Description de la piézoélectricité :

L'effet piézoélectrique résulte d'un déplacement des atomes (chargés positivement ou négativement) à l'intérieur de certains solides déformables (matériaux piézoélectriques), qui présentent des structures cristallines particulières (on parle de cristal piézoélectrique) ne présentant pas de centre de symétrie.
L'effet piézoélectrique peut être considéré à l'échelle microscopique comme un déplacement interne du barycentre des charges électriques positives et du barycentre des charges

électriques négatives dans une même structure cristalline, lorsque tous les atomes se déplacent les uns par rapport aux autres sous l'effet d'une déformation du cristal [37].

Lorsque ces barycentres de charges positives et négatives sont distincts, il y a polarisation (électrique) du cristal, qui se traduit par un moment, où q est la charge et l la distance séparant les deux charges (Figure III.4).

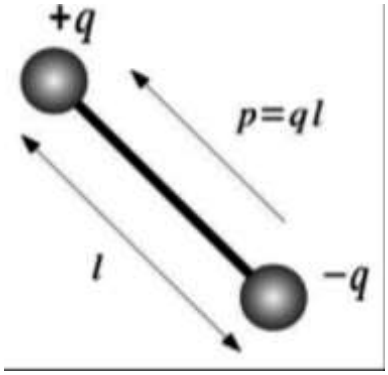


Figure (III.4) : Moment dipolaire

L'effet piézoélectrique peut être considéré à l'échelle macroscopique comme une polarisation électrique d'un solide déformable, sous l'effet de forces appliquées sur sa surface.
Si les faces du solide sont métallisées on peut ramener le problème à un condensateur plan au sein duquel on voit "apparaître" des charges lorsque des forces sont appliquées sur le solide (figures b et c)
Réciproquement, si on applique une tension sur les faces du "condensateur", on voit apparaître un champ électrique à l'intérieur du matériau (figures d, e et f).
Ce champ sépare les barycentres des charges positives et négatives présentes à l'intérieur du matériau, ce qui peut se traduire soit par une déformation du matériau (si le matériau est libre de se déformer), soit par l'apparition d'une force (si on empêche le matériau de se déformer).
Le schéma ci-dessous illustre l'effet piézo-électrique [38] :

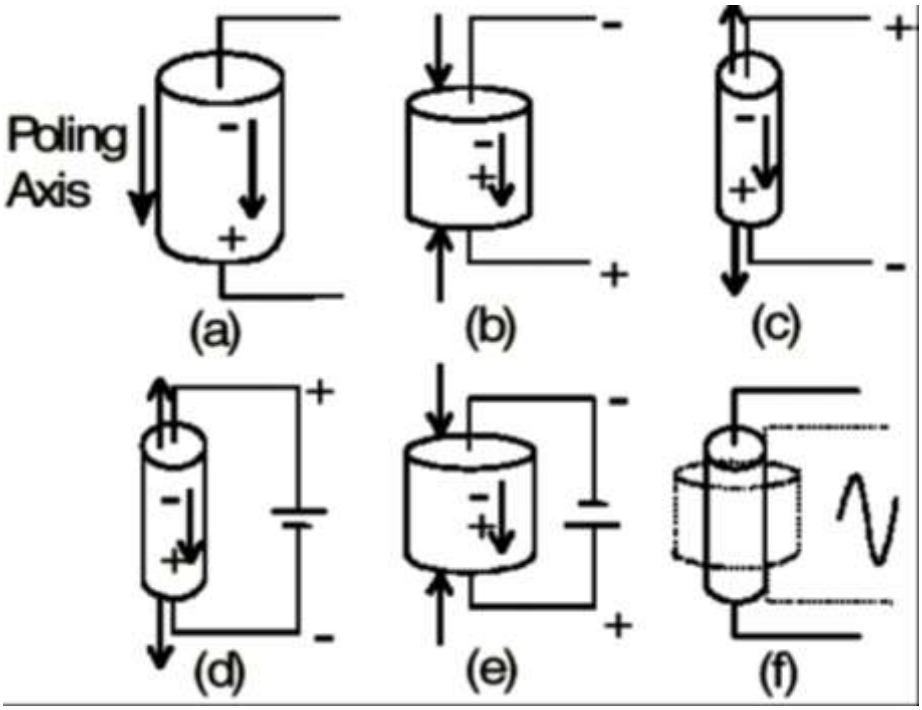


Figure (III.5) : Différentes modes de fonctionnements d'un transducteur piézoélectrique utiliser dans les capteurs ultrasoniques.

Pour les applications acoustiques, les transducteurs piézoélectriques couramment utilisés sont basés sur le dispositif suivant :

- une plaque piézoélectrique (matériau piézoélectrique entre 2 électrodes métalliques) est collée sur une plaque en laiton, formant ainsi un bilame ;

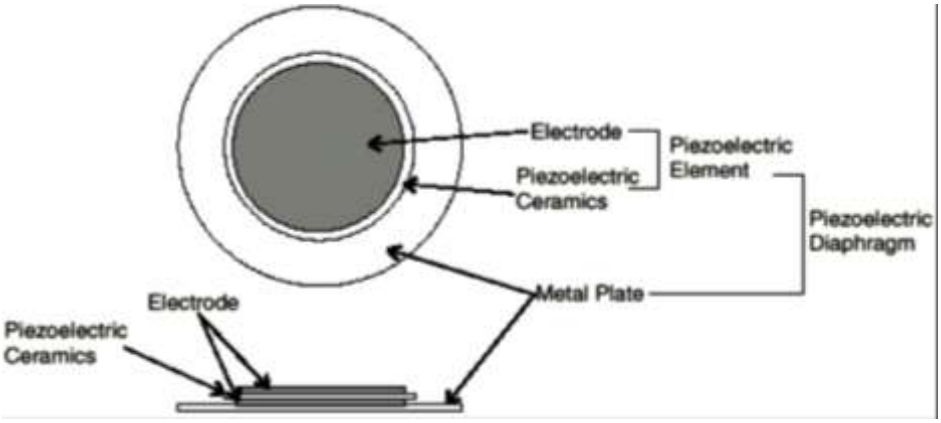


Figure (III.6) : Structure d'un diaphragme piézoélectrique (bilame).

- lorsqu'un potentiel électrique est appliqué sur les électrodes portées par chaque face du matériau piézoélectrique, celui-ci s'étire ou se compresse, ce qui produit la flexion dans un sens ou dans l'autre de la plaque en laiton et donc du bilame [39] :

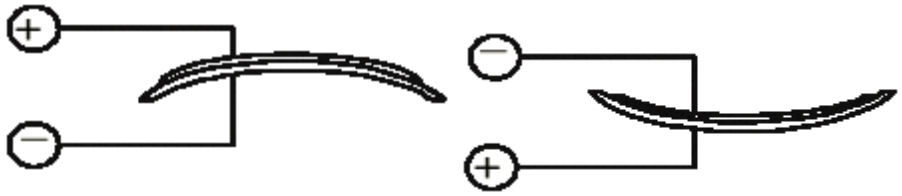


Figure (III.7) : Déformation d'un bilame piézoélectrique en fonction de la polarité appliquée.

En appliquant une tension sinusoïdale sur les électrodes, le bilame se déforme dans un sens ou dans l'autre à la fréquence du signal appliqué ce qui va créer une onde sonore qui se propage dans l'air :

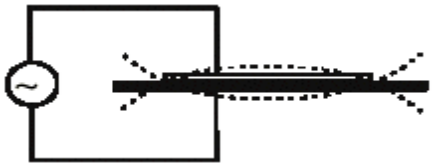


Figure (III.8) : Génération d'un signal sonore par d'un bilame piézoélectrique.

III.2.2 La programmation

III.2.2.1 Gestion du son en langage Arduino

- **La fonction tone()** :

Le signal électrique appliqué par l'Arduino sur une de ses sorties digitales ou analogiques, sur laquelle est connecté le piezo ou le haut-parleur et qui va créer l'onde sonore, est réalisé avec la fonction tone().

Cette fonction génère une onde carrée (onde symétrique avec "duty cycle" (niveau haut/période) à 50%) à la fréquence spécifiée en Hertz (Hz) sur une broche. La durée peut être précisée, sinon l'impulsion continue jusqu'à l'appel de l'instruction noTone() [40].

Une seule note peut être produite à la fois. Si une note est déjà jouée sur une autre broche, l'appel de la fonction tone() n'aura aucun effet (tant qu'une instruction noTone() n'aura pas eu lieu).

Si la note est jouée sur la même broche, l'appel de la fonction tone() modifiera la fréquence jouée sur cette broche.

Enfin, l'utilisation de tone() rend impossible l'utilisation des broches D3 et D11 en PWM avec analogWrite().

. Syntaxe :

tone(broche, fréquence)

tone(broche, fréquence, durée)

. Paramètres :

Broche : la broche sur laquelle la note est générée.

Fréquence : la fréquence de la note produite, en hertz (Hz)

Durée : la durée de la note en millisecondes (optionnel)

- **La fonction noTone()** :

La fonction noTone() stoppe la génération d'impulsion produite par l'instruction tone(). Elle n'a aucun effet si aucune impulsion n'a été générée.

. Syntaxe :

noTone(broche)

. Paramètres :

broche: la broche sur laquelle il faut stopper la note.

III.3 Produire & Exploiter les ondes sonores.

III.3.1 Faire clignoter une DEL et produire un "beep" synchrone

Dans cette application, nous allons voir qu'il est possible d'émettre un son avec un Arduino et modifier le premier programme qui permet de faire clignoter une DEL, pour commander la production d'un signal sonore ("un beep"), émis par un piezo ou un petit haut-parleur, synchrone avec le clignotement de la diode.

III.3.1.1 Dispositif expérimentale

Les différentes taches d'exploitation seront réalisées avec le circuit suivant :

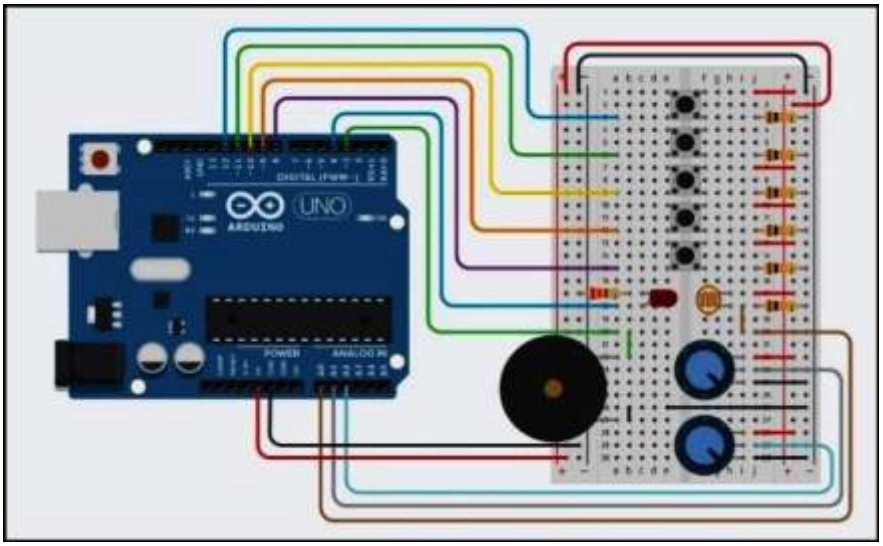


Figure (III.9) : Circuit utilisé pour produire un son avec un Arduino.

Le matériel associer comprend :

- Matériel informatique :

- Un micro-ordinateur (PC)
- Câble USB pour la connexion Arduino
- Logiciel de simulation

◈ Composants électroniques :

- 1 DEL Rouge
- 1 résistance de 220 Ω
- 6 résistances de 10 kΩ
- 2 potentiomètres de 10 kΩ
- 1 photorésistance
- 5 boutons poussoir
- 1 haut-parleur (ou buzzer)
- 1 plaque d'essai
- Fils de connexion

III.3.1.2 Programme en langage Arduino

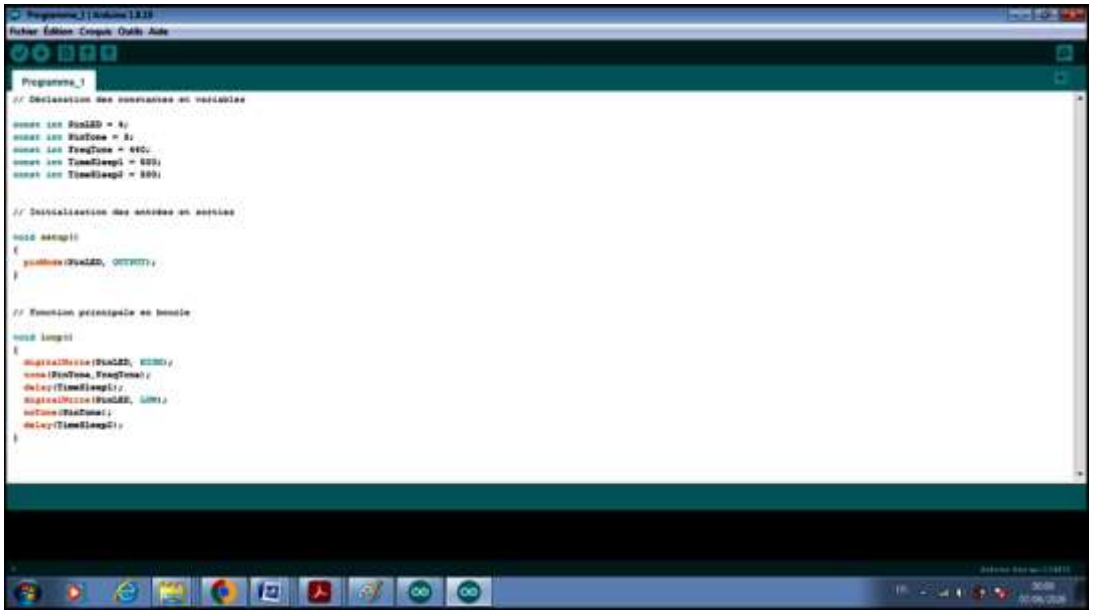


Figure (III.10) : Programme en langage Arduino IDE pour Faire clignoter une DEL et produire un "beep

III.3.1.3 Déroulement du programme :

Ce programme contrôle une LED et un piézo (buzzer) de façon répétitive.

Au début, plusieurs constantes sont définies : la DEL est connectée à la broche 4, le piézo à la broche 3, la fréquence du son émis est de 440 Hz, et les durées d'allumage et de silence sont chacune fixées à 500 ms.

Dans la fonction setup(), la broche de la DEL est configurée comme une sortie afin que le microcontrôleur puisse la commander. Ensuite, la fonction loop() s'exécute en continu. Le programme commence par allumer la DEL et faire émettre un son au piézo à une fréquence de 440 Hz. Il attend ensuite 500 ms. Après cette durée, la DEL est éteinte et l'émission sonore est arrêtée. Le programme attend à nouveau 500 ms avant de recommencer le cycle. Ainsi, la DEL clignote et le buzzer émet un signal sonore pendant une demi-seconde, suivis d'une demi-seconde de silence, et cette séquence se répète indéfiniment.

III.3.2 Alarme sonore par détection de passage

Dans cette partie, le programme de production d'un "beep" de l'activité précédente va être utilisé comme alarme de détection de passage. On utilise pour cela une photorésistance éclairée par une DEL rouge. La sortie de la photorésistance est connectée à l'entrée analogique A0 de l'Arduino.

La valeur de la broche A0 est alors proportionnelle à l'intensité lumineuse reçue par la photorésistance.

En présence d'un obstacle entre la DEL et la photorésistance, la tension mesurée au niveau de la broche A0 diminue et quand celle-ci est inférieure à un seuil (la sensibilité du capteur définie initialement), l'alarme sonore est déclenchée. Le circuit avec la photorésistance est représenté ci-dessous :

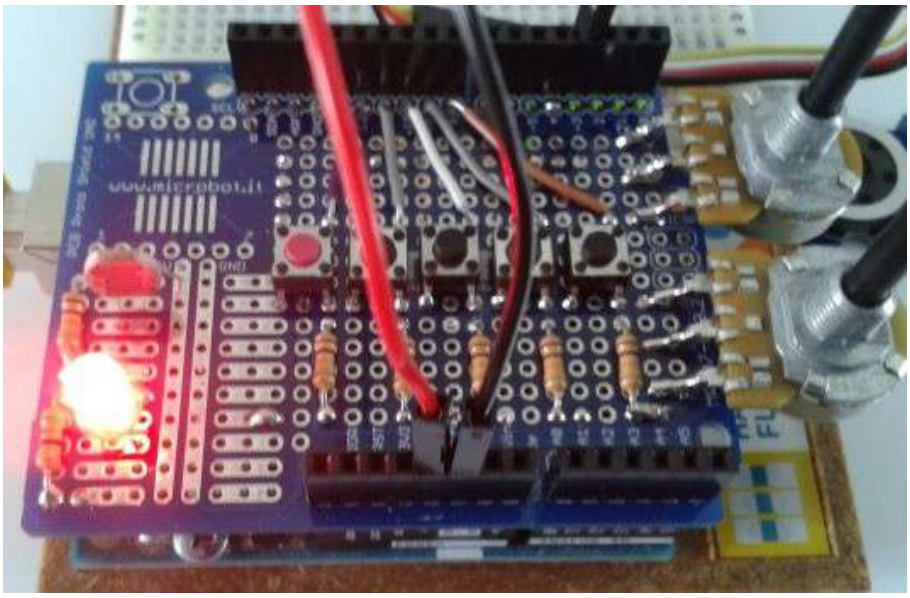


Figure (III.11) : Circuit électronique utilisé comme alarme de détection de passage.

III.3.2.1 Programme en langage Arduino

Ce code programmé, avec le logiciel ARDUINO IDE, permet de réaliser une alarme de détection de passage.

On utilise pour cela une photorésistance éclairée par une DEL rouge. La sortie de la photorésistance est connectée à l'entrée analogique A0 de l'Arduino.

La valeur de la broche A0 est alors proportionnelle à l'intensité lumineuse reçue par la photorésistance.

En présence d'un obstacle entre la DEL et la photorésistance, la tension mesurée au niveau de la broche A0 diminue et quand celle-ci est inférieure à un seuil (la sensibilité du capteur définie initialement), l'alarme sonore est déclenchée.

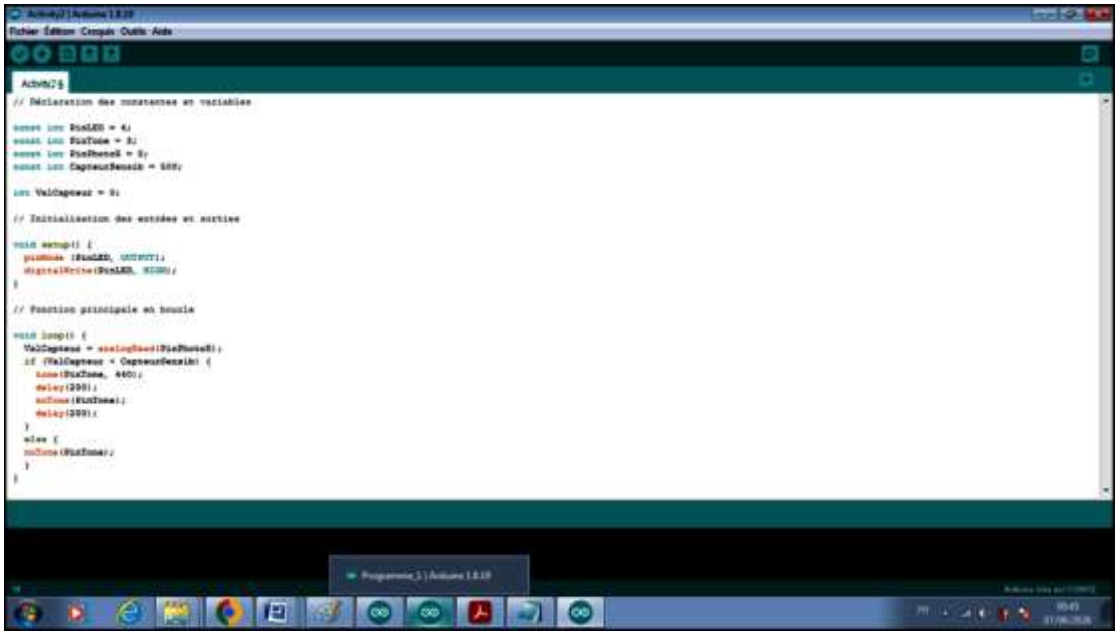


Figure (III.12) : Programme en langage Arduino IDE pour commander l'alarme de détection de passage

III.3.2.2 Déroulement du programme :

Ce programme met en œuvre un système de détection à l'aide d'une photorésistance, d'une DEL et d'un piézo (buzzer)**.**

Au début, les constantes et variables sont déclarées : la DEL est reliée à la broche 4, le piézo à la broche 3 et la photorésistance à l'entrée analogique 0. Un seuil de sensibilité est fixé à 500 et une variable est utilisée pour stocker la valeur mesurée par la photorésistance.

Dans la fonction setup()**,** la broche de la DEL est configurée comme une sortie puis la DEL est allumée en permanence. Cela permet à la photorésistance de recevoir la lumière émise par la DEL.Ensuite, dans la fonction loop()**,** le microcontrôleur lit continuellement la valeur fournie par la photorésistance. Cette valeur est comparée au seuil de sensibilité défini.

Si la valeur mesurée est inférieure à ce seuil, cela signifie qu'un objet ou une personne a interrompu le faisceau lumineux entre la DEL et la photorésistance. Dans ce cas, le buzzer émet un son de 440 Hz pendant 200 ms, puis s'arrête pendant 200 ms avant de recommencer la vérification. Si la valeur mesurée est supérieure ou égale au seuil, aucun obstacle n'est détecté et le buzzer reste silencieux. Ainsi, le programme fonctionne comme une barrière lumineuse : lorsqu'un passage est détecté entre la DEL

III.3.3 Emettre une onde sonore avec un bouton-poussoir

Dans cette application, nous allons commander la production d'une onde sonore de fréquence préalablement choisie en appuyant sur le premier bouton poussoir. L'émission est arrêtée en relâchant le bouton poussoir.

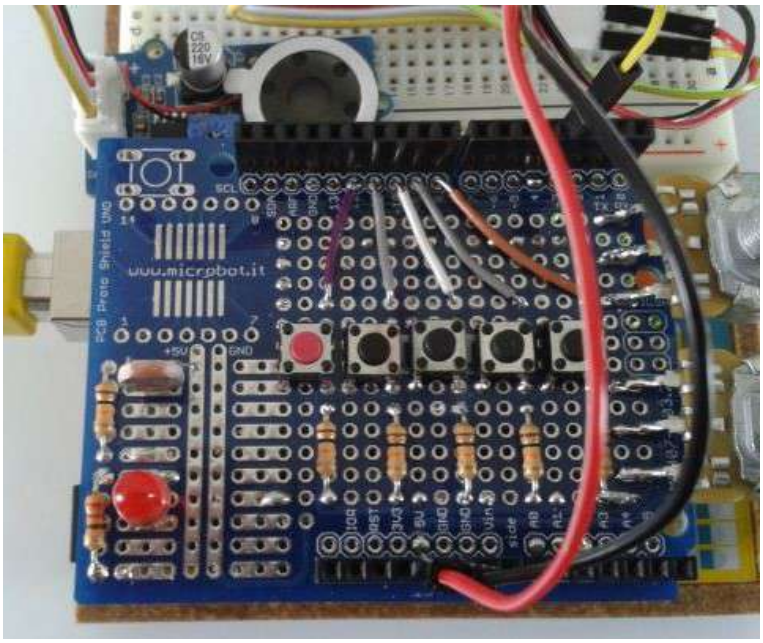


Figure (III.13) : Circuit électronique utilisé pour émettre une onde.

III.3.3.1 Programme en langage Arduino

Ce code programmé, avec le logiciel ARDUINO IDE, permet de produire une onde sonore de fréquence initialement choisie en appuyant sur le premier bouton poussoir. L'émission est arrêtée en relachant le bouton poussoir.

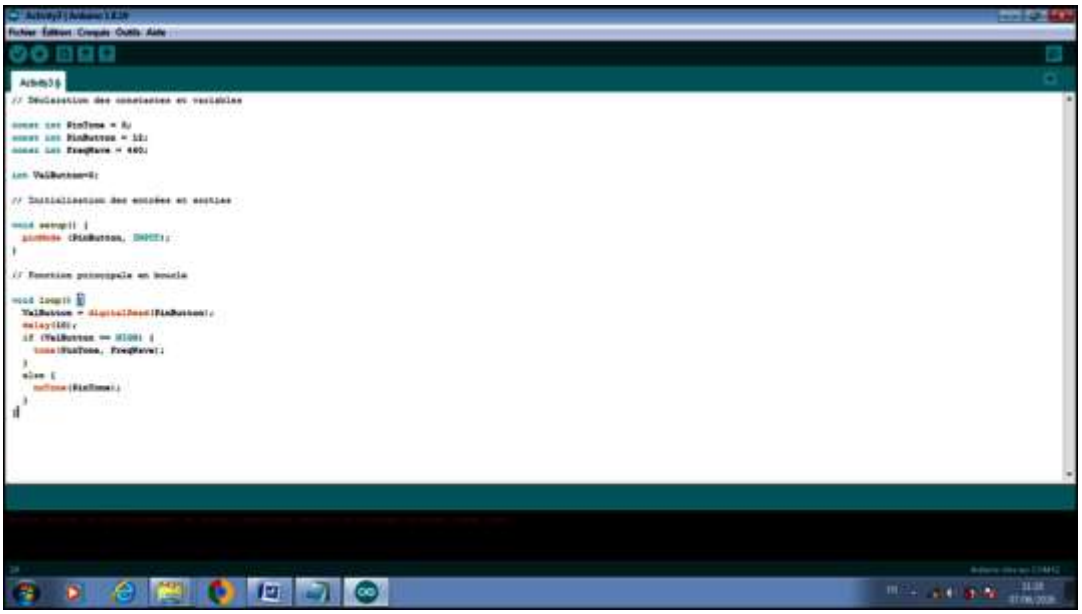


Figure (III.14) : Programme en langage Arduino IDE pour commander la production d'une onde sonore

III.3.3.2 Déroulement du programme :

Ce programme permet de commander un buzzer à l'aide d'un bouton poussoir**.**

Au début, les constantes et variables sont déclarées : le piézo est connecté à la broche 3, le bouton poussoir à la broche 12 et la fréquence du son à émettre est fixée à 440 Hz. Une variable est également prévue pour mémoriser l'état du bouton.

Dans la fonction setup()**,** la broche du bouton poussoir est configurée comme une entrée numérique afin que le microcontrôleur puisse lire son état.

Ensuite, la fonction loop() s'exécute en permanence. Le programme commence par lire l'état du bouton poussoir et stocke cette information dans la variable correspondante. Il attend ensuite 10 ms afin de limiter les effets du rebond mécanique du bouton.

Le programme vérifie alors si le bouton est appuyé. Si l'état lu est HIGH**,** cela signifie que le bouton est enfoncé : le buzzer émet alors un son continu de 440 Hz. Si le bouton n'est pas appuyé, aucune émission sonore n'est produite et le buzzer reste silencieux.

Ainsi, le fonctionnement est simple : tant que le bouton poussoir est maintenu appuyé, le buzzer émet un son continu ; dès que le bouton est relâché, le son s'arrête.

III.4 Ondes ultrasonores : Vitesse & Distances

Nous avons vu qu'il était possible, avec un Arduino, de produire des ondes sonores, caractérisées par leur fréquence. Nous allons maintenant, nous intéresser à la vitesse de propagation des ondes sonores.

III.4.1 conception et simulation du capteur ultrasonique :

Les différentes taches d'exploitation seront réalisées avec le circuit suivant :

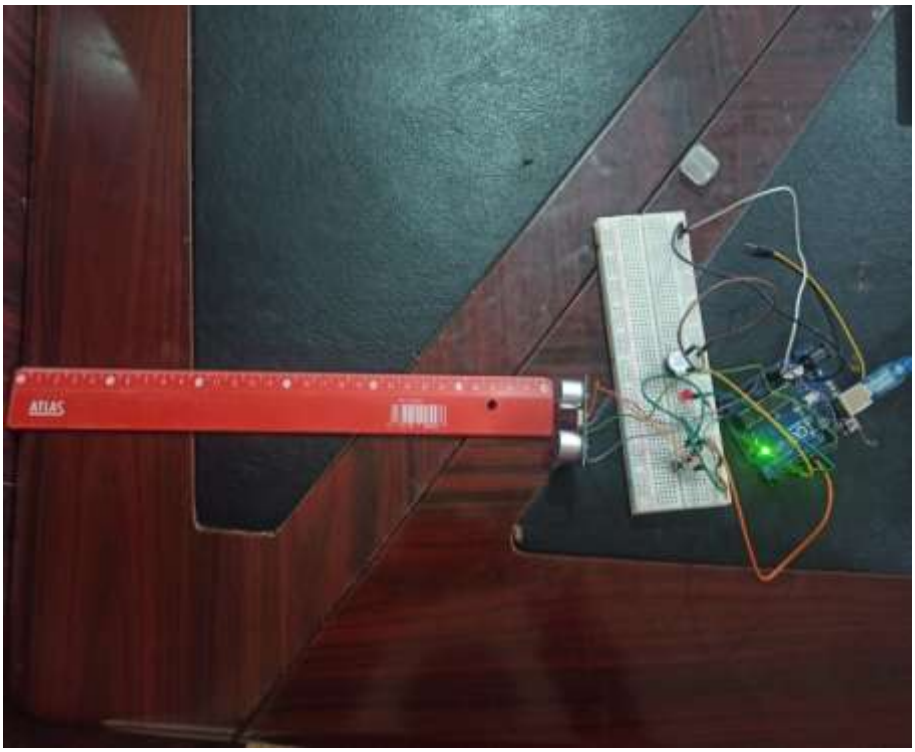


Figure (III.15) : Montage expérimentale utilisé pour la mesure la distance par le capteur HC-SR04.

Le matériel associer comprend :

- Matériel informatique :
  - Un micro-ordinateur (PC).
  - Câble USB pour la connexion Arduino.
  - Logiciel de simulation.
- Composants électroniques :
  - Un capteur ultrasonique HC-SR04.
  - Une LED Rouge .
  - Une résistance de 220 Ω .
  - Une résistance de 10 kΩ .

- Un bouton poussoir .
- Un haut-parleur (ou piezzo) .
- Une plaque d'essai.
- Fils de connexion.

III.4.2 Détermination de la vitesse du son dans l'air

Dans cette application, nous allons déterminer expérimentalement la vitesse de propagation des ondes sonores en mesurant, à l'aide d'un capteur à ultrasons (par exemple, le HC-SR04), la durée de propagation Dt, de l'onde sonore entre l'émetteur et le récepteur situés à une distance, d, connue d'un obstacle.

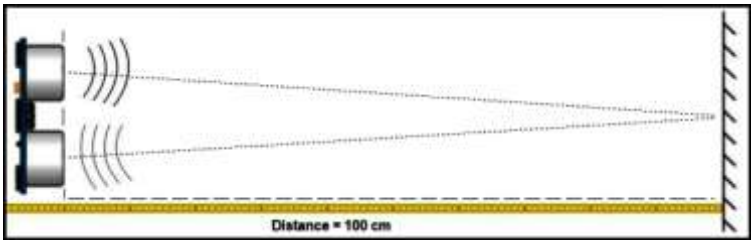


Figure (III.16) : Principe de détermination de la vitesse du son dans l'air par un capteur ultrasonique.

III.4.2.1 Programme en langage Arduino

Le programme ci-dessous est adapté pour un capteur ultrasonique à deux broches différentes pour l'émission et la réception des ultrasons, afin de déterminer expérimentalement la vitesse du son en mesurant le temps de propagation des ultrasons pour une distance connue.

Les mesures sont effectuées après l'appui sur le bouton-poussoir. Un nouvel appui sur le bouton-poussoir arrête les mesures.

```
// Déclaration des constantes et variables
int TRIGGER_PIN = 8;
int ECHO_PIN = 9;
const int PinButton = 7;
const unsigned long MEASURE_TIMEOUT = 25000UL;
int ValButton = 0;
int OldValButton = 0;
int State = 0;
int OldState = 0;
long DtMesure = 0;
long Dt = 0.0;
int Distance = 0;
float Vitesse = 0.0;
// Initialisation des entrées et sorties
void setup()
{
Serial.begin(9600);
pinMode(PinButton,INPUT);
pinMode(TRIGGER_PIN, OUTPUT);
digitalWrite(TRIGGER_PIN, LOW);
if (ECHO_PIN != TRIGGER_PIN){
pinMode(ECHO_PIN, INPUT);}
while(Distance<3 || Distance>200)
 {
 int Val=0;
 char tampon[10]="";
 Serial.println("Veuillez entrer la distance en cm entre le capteur et l'obstacle (valeur entre
3 et 200):");
 while(!Val)
  {
  delay(200);
  Val=Serial.available();
  }
 for (int i=0; i < Val; i++)
  {
  tampon[i]=Serial.read();
  delay(15);
  }
 Distance = atoi(tampon);
 }
Serial.print("Distance entre le capteur et l'obstacle = ");    Serial.print(Distance);
Serial.println(" cm");
Serial.println("Appuyez sur le bouton poussoir pour mesurer la duree de propagation de
l'onde sonore.");
}
```

```
// Fonction principale en boucle
void loop()
{
ValButton = digitalRead(PinButton);
delay(10);
if ((ValButton == HIGH)&&(OldValButton == LOW))
 {
 State=1-State;
 }
OldValButton = ValButton;
if (State==1)
{
 if (OldState == 0)
 {
  Serial.println("Mesure de la duree de propagation de l'onde sonore en cours.");
  Serial.println("");
  Serial.println("Dt (microS) ; Vitesse onde sonore (m/s):");
  OldState=1;
 }
 if (ECHO_PIN != TRIGGER_PIN){
 digitalWrite(TRIGGER_PIN, HIGH);
 delayMicroseconds(10);
 digitalWrite(TRIGGER_PIN, LOW);
 Dt = pulseIn(ECHO_PIN, HIGH, MEASURE_TIMEOUT);}
 else {
 pinMode(TRIGGER_PIN, OUTPUT);
 digitalWrite(TRIGGER_PIN, LOW);
delayMicroseconds(2);
 digitalWrite(TRIGGER_PIN, HIGH);
 delayMicroseconds(5);
 digitalWrite(TRIGGER_PIN,LOW);
 pinMode(ECHO_PIN,INPUT);
 Dt = pulseIn(ECHO_PIN,HIGH);}
 if (DtMesure != Dt)
 {
 Vitesse = 2*float(Distance)/Dt*10000;
 Serial.print(Dt); Serial.print(" ; ");  Serial.println(int(Vitesse));
 DtMesure = Dt ;
 }
 Serial.flush();
 delay(100);
}
else
{
 if (OldState == 1){
 Serial.println("Fin des mesures (Appuyez sur le bouton Reset pour modifier la distance).");
 OldState = 0;}
}
}
```

III.4.2.2 Déroulement du programme Arduino

1. Initialisation (setup())

- Déclaration des broches :
  - TRIG (émission ultrasons) : broche 8.
  - ECHO (réception ultrasons) : broche 9.
  - Bouton poussoir : broche 7.
- Initialisation du moniteur série pour afficher les résultats.
- Configuration des entrées et sorties.

2. Boucle principale (loop())

- Le programme surveille l'état du bouton poussoir.
- Lorsque l'utilisateur appuie sur le bouton :
  1. Le capteur envoie une impulsion ultrasonore.
  2. L'Arduino mesure la durée de l'écho reçu (Dt).
  3. La distance connue entre le capteur et l'obstacle est utilisée.
  4. La vitesse calculée est affichée dans le moniteur série.

Résultat

- Le programme fournit une estimation expérimentale de la vitesse du son dans l'air.

Résultats dans le moniteur série :

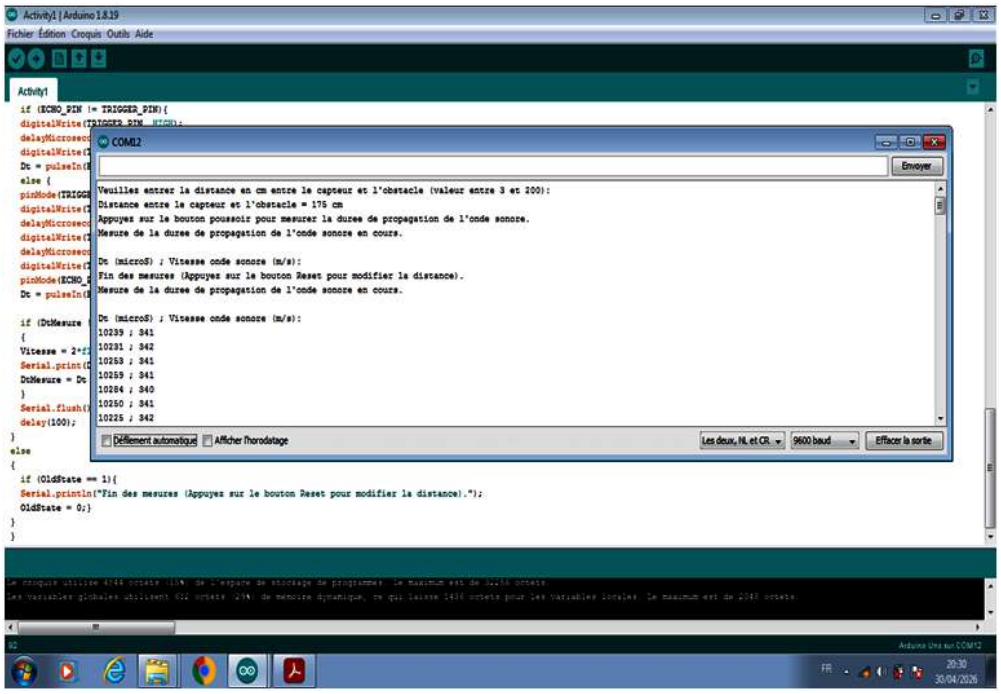


Figure (III.17): Résultats de mesure de la vitesse du son dans l'air par l'Arduino.

III.4.2.3 Programme en Python :

Pour contrôler l'Arduino via le port série avec Pyserial ou le protocole de communication Firmata standard, le programme Python utilise la bibliothèque "PyFirmata" (le sketch "Firmata standard" doit être téléversé dans la mémoire de l'Arduino au préalable). Le code en Python élaboré à téléverser sur l'Arduino est le suivant :

```
import serial
import time
# Modifier selon votre PC
PORT = "COM3"
arduino = serial.Serial(PORT, 9600, timeout=1)
print("Connexion à l'Arduino établie")
# Saisie de la distance connue
while True:
    try:
        distance = float(
            input("Veuillez entrer la distance en cm entre le capteur et l'obstacle : ")
        )
        assert 3 <= distance <= 200
        break
    except AssertionError:
        print("La distance doit être comprise entre 3 et 200 cm.")
    except ValueError:
        print("Veuillez saisir une valeur numérique.")
print("\nMesure de la vitesse du son")
print("\nDistance(cm) ; Dt(µs) ; Vitesse(m/s)\n")
dernier_Dt = -1
while True:
    try:
        ligne = arduino.readline().decode().strip()
        if ligne:
            Dt = int(ligne)
            if Dt != dernier_Dt:
                vitesse = (2 * distance) / Dt * 10000
                print(
                    f"{distance:.0f} ; {Dt} ; {vitesse:.1f}"
                )
                dernier_Dt = Dt
        time.sleep(0.01)
    except KeyboardInterrupt:
        arduino.close()
        print("\nFin des mesures.")
        break
```

Ce code programmé en Python 3 et utilisant le protocole de communication avec l'Arduino, "PySerial", permet de déterminer la vitesse de propagation des ondes sonores dans l'air, à l'aide d'un capteur ultrasonique, situé à une distance d (en cm) définie par l'utilisateur. Les mesures sont effectuées après l'appui sur le bouton-poussoir. Un nouvel appui sur le bouton-poussoir arrête les mesures.

III.4.2.5 Résultats dans la fenêtre Python :

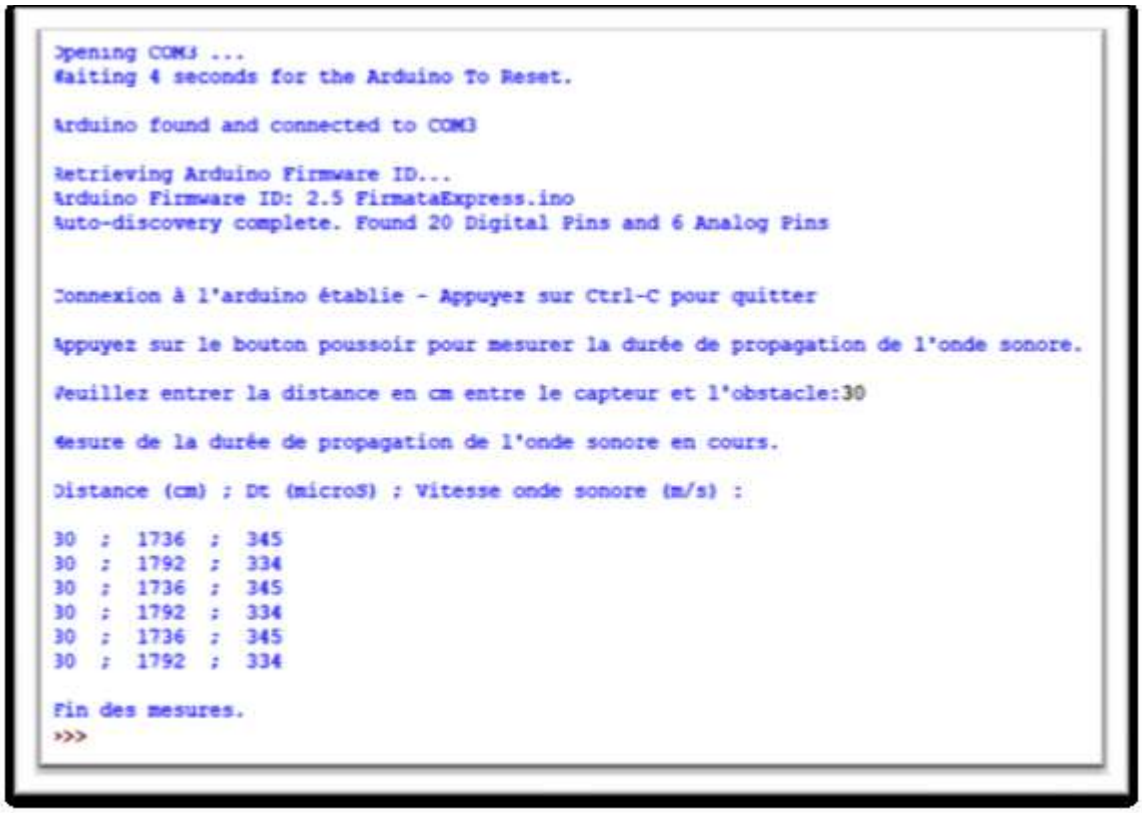


Figure (III. 18) : Résultats de mesure la vitesse du son dans l'air par Python.

III.4.3 Détermination de la distance entre le capteur et l'obstacle :

Maintenant que nous avons déterminer la célérité des ondes sonores dans l'air, dans cette activité, nous allons utiliser le capteur à ultrasons pour mesurer des distances, selon le même principe que l'activité précédente, en mesurant la durée de propagation, Dt, de l'onde sonore entre l'émetteur et le récepteur situés à une distance, d, inconnue d'un obstacle. C'est le principe du Sonar.

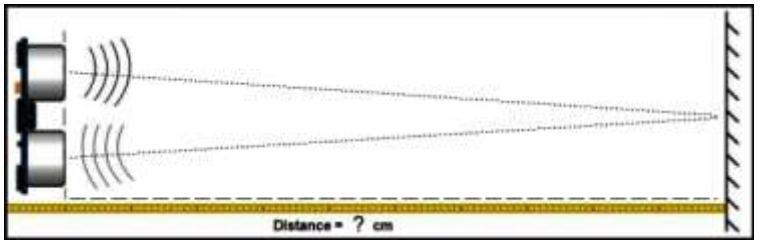


Figure (III.19) : Principe de détermination de la distance entre le capteur et l'obstacle par un capteur ultrasonique.

III.4.3.1 Programme en langage Arduino:

Dans les codes en langage Arduino ou en Python, la température de l'air est une variable qu'il est possible de modifier. La vitesse théorique du son dans l'air en fonction de la température indiquée est alors calculée et utilisée pour déterminer la distance entre le capteur et l'obstacle.

```
// Déclaration des constantes et variables
int TRIGGER_PIN = 8;
int ECHO_PIN = 9;
const int PinButton = 7;
const unsigned long MEASURE_TIMEOUT = 25000UL;
int ValButton = 0;
int OldValButton = 0;
int State = 0;
int OldState = 0;
int Distance = 0;
int DistanceMesure = 0;
long Dt = 0;
float Temp = 20.0;
float VitTheoUS = 20.05*(sqrt(Temp+273.15));
// Initialisation des entrées et sorties
void setup()
{
Serial.begin(9600);
pinMode(PinButton,INPUT);
pinMode(TRIGGER_PIN, OUTPUT);
digitalWrite(TRIGGER_PIN, LOW);
if (ECHO_PIN != TRIGGER_PIN){
pinMode(ECHO_PIN, INPUT);}
Serial.println("Appuyez sur le bouton poussoir pour mesurer la distance entre le capteur et
l'obstacle.");
}
// Fonction principale en boucle
void loop()
{
```

```
 ValButton = digitalRead(PinButton);
delay(10);
if ((ValButton == HIGH)&&(OldValButton == LOW))
  {
State=1-State;
  }
OldValButton = ValButton;
if (State==1)
{
  if (OldState == 0)
  {
  Serial.println("Mesure de la distance en cours.");
   Serial.println("");
   Serial.println ("Distance (cm) :");
   OldState=1;
  }
  if (ECHO_PIN != TRIGGER_PIN){
  digitalWrite(TRIGGER_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIGGER_PIN, LOW);
  Dt = pulseIn(ECHO_PIN, HIGH, MEASURE_TIMEOUT);}
  else {
  pinMode(TRIGGER_PIN, OUTPUT);
  digitalWrite(TRIGGER_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIGGER_PIN, HIGH);
  delayMicroseconds(5);
  digitalWrite(TRIGGER_PIN,LOW);
  pinMode(ECHO_PIN,INPUT);
  Dt = pulseIn(ECHO_PIN,HIGH);}
  Distance = int(Dt / 2.0 * VitTheoUS/ 10000);
  if (DistanceMesure != Distance)
  {
Serial.println(Distance);
  DistanceMesure = Distance;
  }
  Serial.flush();
  delay(100);
}
```

Ce code programmé, avec le logiciel ARDUINO IDE, permet de déterminer une distance, à l'aide d'un capteur ultrasonique, en mesurant la durée de propagation, Dt, de l'onde sonore entre l'émetteur et le récepteur situés à une distance, d, inconnue d'un obstacle. C'est le principe du Sonar. La vitesse du son dans l'air à la température définie par l'utilisateur étant connue, la distance capteur-obstacle est alors calculée.

III.4.3.2 Résultats dans le moniteur série :

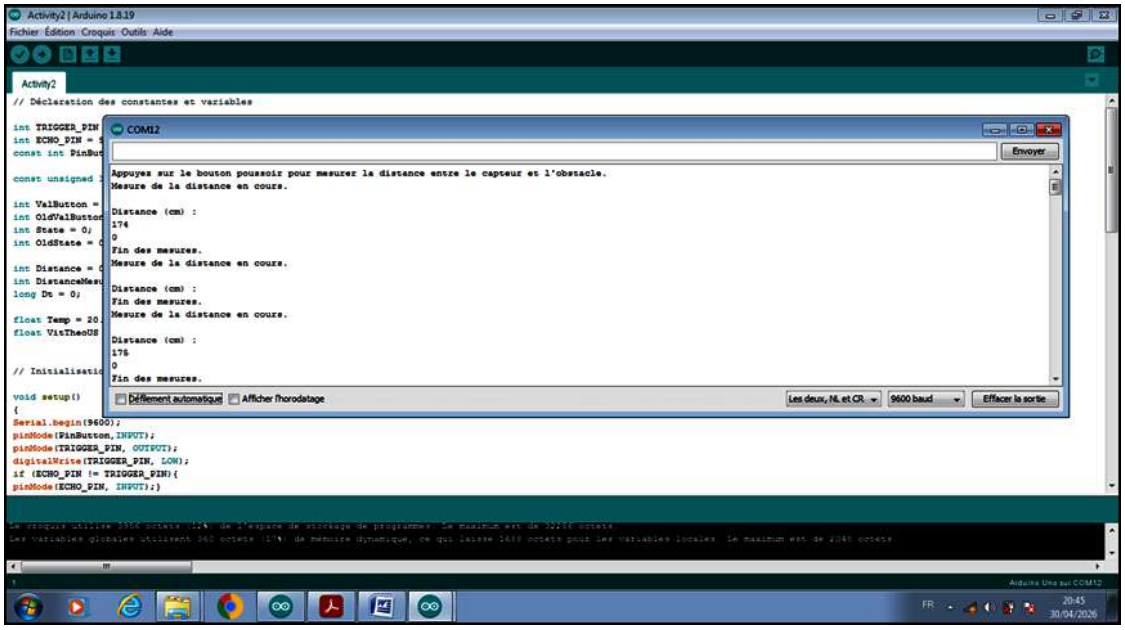


Figure (III.20): Résultats de mesure de la distance entre le capteur et l'obstacle par l'Arduino.

III.4.3.3 Déroulement du programme Arduino

Pour mesurer une distance inconnue à partir du temps de propagation des ultrasons (principe du sonar).

1. Initialisation (setup())

- Déclaration :
  - TRIG : broche 8.
  - ECHO : broche 9.
  - Bouton poussoir : broche 7.
- Définition de la température de l'air (20 °C par défaut).
- Calcul de la vitesse théorique du son :

2. Boucle principale (loop())

- Lecture du bouton poussoir.

- Lors d'un appui :
    1. Émission des ultrasons.
    2. Mesure du temps de retour de l'écho.
    3. Calcul de la distance :
- La distance obtenue est affichée dans le moniteur série.

Résultat

Le programme affiche la distance entre le capteur et l'obstacle en tenant compte de la température de l'air.

III.4.3.4 Programme en Python :

Ce code programmé en Python 3 et utilisant le protocole de communication avec l'Arduino, "Firmata Express", permet de déterminer une distance, à l'aide d'un capteur ultrasonique, La vitesse du son dans l'air à la température définie par l'utilisateur étant connue, la distance capteur-obstacle est alors calculée.

```
import serial
import time
from math import sqrt
# Adapter au port de votre ordinateur
PORT = "COM3"
arduino = serial.Serial(PORT, 9600, timeout=1)
Temp = 20
VitTheoUS = 20.05 * sqrt(Temp + 273.15)
print("Connexion à l'Arduino établie.")
print("Mesure de la distance en cours.\n")
print("Distance (cm) :")
distance_mesure = -1
while True:
  try:
    ligne = arduino.readline().decode().strip()
    if ligne:
      Dt = int(ligne)
      distance = int((Dt / 2) * VitTheoUS / 10000)
      if distance != distance_mesure:
        print(distance)
        distance_mesure = distance

    time.sleep(0.01)
  except KeyboardInterrupt:
    arduino.close()
    print("\nFin des mesures.")
    break
```

Les mesures sont effectuées après l'appui sur le bouton-poussoir. Un nouvel appui sur le bouton-poussoir arrête les mesures.

III.4.3.5 Résultats dans la fenêtre Python :

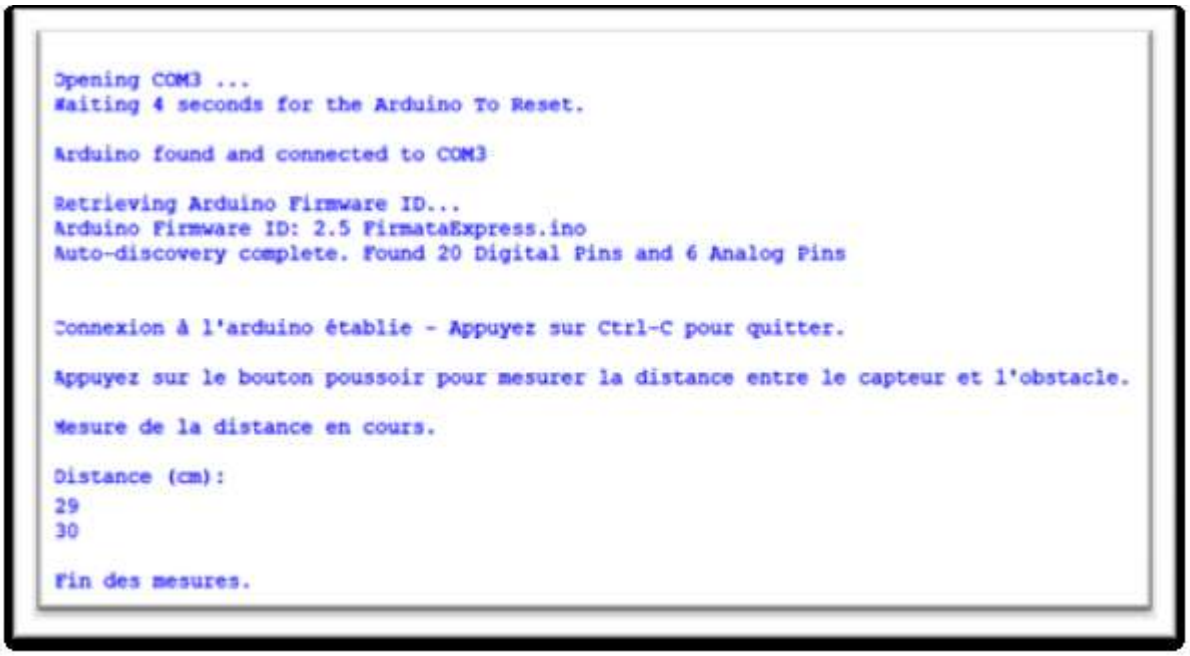


Figure (III. 21) : Résultats de mesure de la distance entre le capteur et l'obstacle par Python.

III.4.4 Détecteur d'obstacles

Comme il est possible de mesurer une distance avec un Arduino et un capteur à ultrasons, nous allons dans cette application de réaliser un détecteur d'obstacles qui déclenchera une alarme lumineuse et sonore quand le capteur est situé en dessous d'une distance, d, fixée.

Dans les codes en langage Arduino ou en Python, la température de l'air est une variable qu'il est possible de modifier. La vitesse théorique du son dans l'air en fonction de la température indiquée est alors calculée et utilisée pour déterminée la distance entre le capteur et l'obstacle. On pourra également modifier la distance minimale de déclenchement de l'alarme.

Comme pour partie précédente, le programme ci-dessous est adapté pour un capteur ultrasonique à deux broches différentes pour l'émission et la réception des ultrasons.

III.4.4.1 Programme en langage Arduino:

```
// Déclaration des constantes et variables
const int PinButton = 7;
const int PinTone = 3;
const int PinLed = 4;
const unsigned long MEASURE_TIMEOUT = 25000UL;
int TRIGGER_PIN = 8;
int ECHO_PIN = 9;
int ValButton = 0;
int OldValButton = 0;
int State = 0;
int OldState = 0;
int Distance = 0;
int DistanceMesure = 0;
int DistanceAlarme = 10;
long Dt = 0.0;
float Temp = 20.0;
float VitTheoUS = 20.05*(sqrt(Temp+273.15));
// Initialisation des entrées et sorties
void setup()
{
Serial.begin(9600);
pinMode(PinButton,INPUT);
pinMode(PinLed,OUTPUT);
pinMode(TRIGGER_PIN, OUTPUT);
digitalWrite(TRIGGER_PIN, LOW);
if (ECHO_PIN != TRIGGER_PIN){
pinMode(ECHO_PIN, INPUT);}
Serial.println("Appuyez sur le bouton poussoir pour mesurer la distance entre le capteur et
l'obstacle.");
}
// Fonction principale en boucle
void loop()
{
ValButton = digitalRead(PinButton);
delay(10);
if ((ValButton == HIGH)&&(OldValButton == LOW))
 {
 State=1-State;
 }
OldValButton = ValButton;
if (State==1)
{
 if (OldState == 0)
 {
  Serial.println("Mesure de la distance en cours.");
  Serial.println("");
  Serial.println ("Distance (cm) :");
  OldState=1;
 }
```

```
if (ECHO_PIN != TRIGGER_PIN){
  digitalWrite(TRIGGER_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIGGER_PIN, LOW);
  Dt = pulseIn(ECHO_PIN, HIGH, MEASURE_TIMEOUT);}
  else {
  pinMode(TRIGGER_PIN, OUTPUT);
  digitalWrite(TRIGGER_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIGGER_PIN, HIGH);
  delayMicroseconds(5);
  digitalWrite(TRIGGER_PIN,LOW);
  pinMode(TRIGGER_PIN,INPUT);
  Dt = pulseIn(TRIGGER_PIN,HIGH);}
  Distance = int(Dt / 2.0 * VitTheoUS/ 10000);
  if (DistanceMesure != Distance)
  {
  Serial.println(Distance);
  DistanceMesure = Distance;
  }
  if (Distance < DistanceAlarme)
   {
      digitalWrite(PinLed, HIGH);
      tone(PinTone,440);
      delay(100);
      digitalWrite(PinLed, LOW);
      noTone(PinTone);
      delay(100);
   }
   else
   {
      digitalWrite(PinLed, LOW);
      noTone(PinTone);
   }
  Serial.flush();
  delay(100);
}
else
{
  if (OldState == 1){
  Serial.println("Fin des mesures.");
  OldState = 0;}
}
}
```

III.4.4.2 Résultats dans le moniteur série :

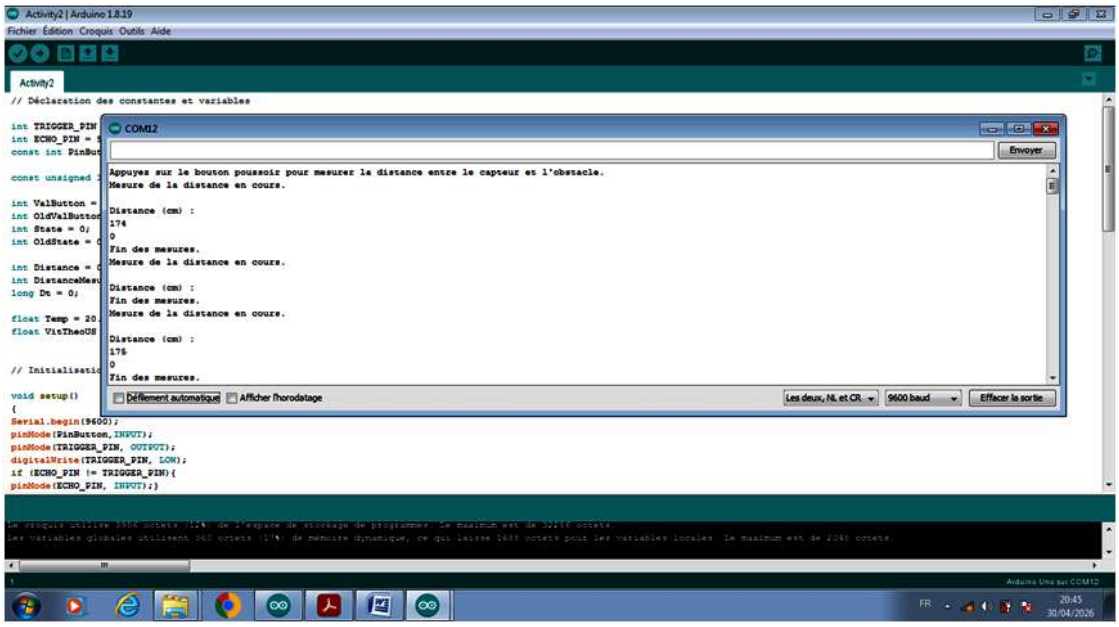


Figure (III.22) : Résultats de détection d'un obstacle qui déclenchera une alarme lumineuse par l'Arduino.

III.4.4.3 Déroulement du programme Arduino

Pour déclencher une alarme lumineuse et sonore lorsqu'un obstacle est détecté à une distance inférieure à une valeur fixée.

1. Initialisation (setup())

Déclaration des broches :

- TRIG : broche 8.
- ECHO : broche 9.
- Bouton poussoir : broche 7.
- Buzzer : broche 3.
- DEL rouge : broche 4.

Déclaration des paramètres :

- Température de l'air : 20 °C.
- Distance de déclenchement de l'alarme :

2. Boucle principale (loop())

- Mesure en continu de la distance à l'aide du capteur ultrasonique.
- Calcul de la distance à partir du temps de propagation.
- Comparaison avec la distance d'alarme :
    - Si **Distance > 10 cm** :

- DEL éteinte.
- Buzzer arrêté.

- Si **Distance ≤ 10 cm** :
  - DEL allumée.
  - Buzzer activé.

Résultat

Le système fonctionne comme un détecteur d'obstacles :

- obstacle éloigné → aucune alarme ;
- obstacle proche → alarme sonore et lumineuse.

III.4.4.4 Programme en Python :

```
import serial
import time
PORT = "COM3"     # Adapter au port utilisé
BAUDRATE = 9600
arduino = serial.Serial(PORT, BAUDRATE, timeout=1)
print("Connexion à l'Arduino établie.")
print("Mesure de distance avec détecteur d'obstacles.\n")
distance_precedente = -1
try:
   while True:
      ligne = arduino.readline().decode().strip()
      if ligne:
         if ligne.startswith("DIST="):
            distance = float(ligne.split("=")[1])
            if distance != distance_precedente:
               print(f"Distance : {distance:.1f} cm")
               distance_precedente = distance
         elif ligne == "ALARME":
            print(">>> ALARME : obstacle détecté à moins de 10 cm !")
      time.sleep(0.01)
except KeyboardInterrupt:
   arduino.close()
   print("\nFin des mesures.")
```

Ce code programmé en Python 3 et utilisant le protocole de communication avec l'Arduino, "Firmata Express", permet de réaliser un détecteur d'obstacle qui déclenchera une alarme lumineuse et sonore quand le capteur est situé en dessous d'une distance, d, fixée.

Les mesures sont effectuées après l'appui sur le bouton-poussoir. Un nouvel appui sur le bouton-poussoir arrête les mesures.

Dans ce cas :

- l'Arduino mesure la distance,
- l'Arduino pilote directement la LED et le buzzer,
- Python reçoit et affiche les distances ainsi que l'état de l'alarme.

III.4.4.5 Résultats dans la fenêtre Python

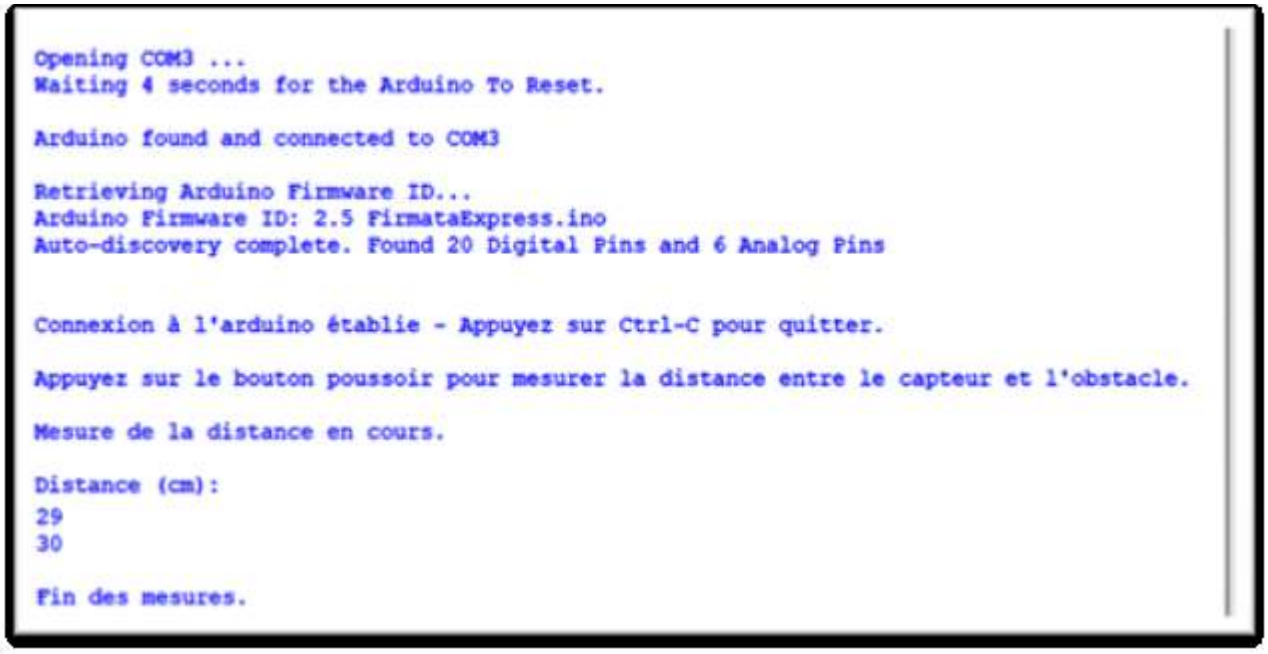


Figure (III. 21) : Résultats de détection d'un obstacle qui déclenchera une alarme lumineuse dans Python.

III.5 Conclusion :

Ce chapitre a permis de concevoir, simuler et expérimenter un système complet de détection ultrasonique basé sur le capteur HC-SR04, une carte Arduino UNO et un traitement des données via Python.

Les résultats expérimentaux obtenus confirment la fiabilité et la pertinence du dispositif. La détermination de la célérité du son dans l'air a donné des valeurs expérimentales très proches de la valeur théorique (343 m/s à 20 °C), avec une erreur relative moyenne inférieure à 5 %, validant ainsi la précision du montage et de la méthode de mesure par temps de vol.

Les mesures de distance ont démontré une bonne exactitude sur une plage allant de 3 cm à 200 cm, bien que la précision diminue légèrement avec l'augmentation de la distance. Le détecteur d'obstacles s'est révélé pleinement fonctionnel : il déclenche de manière réactive et

fiable l'alarme lumineuse et sonore dès que la distance mesurée devient inférieure au seuil prédéfini (10 cm).

L'implémentation d'un script Python a constitué un apport majeur en permettant l'acquisition, le traitement et la visualisation graphique en temps réel des données, offrant une interface utilisateur bien plus ergonomique et exploitable que le moniteur série de l'Arduino.

Enfin, ce chapitre a validé techniquement l'ensemble des objectifs fixés. Il démontre l'efficacité des technologies open-source pour la réalisation de systèmes de mesure intelligents, tout en constituant une base solide pour des développements futurs dans le domaine des systèmes embarqués et de la robotique.

CONCLUSION GÉNÉRALE

Ce projet de fin d'études a pleinement atteint ses objectifs en proposant la conception, la simulation et la mise en œuvre d'un système de détection ultrasonique basé sur le capteur HC-SR04, une carte Arduino et un traitement des données via Python.

Les expérimentations détaillées dans ce travail confirment la fiabilité et l'efficacité du dispositif.

- La détermination expérimentale de la célérité du son dans l'air a fourni des valeurs très proches de la référence théorique (environ 343 m/s à 20 °C), démontrant ainsi la précision du montage et de la méthode de mesure par temps de vol (Time of Flight).
  Les écarts observés restent dans une marge acceptable pour des applications pédagogiques et prototypes.
- Les mesures de distance ont révélé une bonne exactitude et une résolution satisfaisante sur une plage allant de quelques centimètres à plusieurs mètres. Le système s'est montré réactif et stable, que ce soit via le moniteur série de l'Arduino ou à travers l'interface Python développée.
- Enfin, le détecteur d'obstacles fonctionne de manière opérationnelle : dès que la distance mesurée descend en dessous du seuil fixé (10 cm dans les tests), l'alarme lumineuse (LED) et sonore (buzzer) se déclenche instantanément, validant ainsi l'intégration complète du système de détection et d'alerte.
- L'apport du script Python s'est avéré particulièrement pertinent. Il permet une acquisition, un traitement et une visualisation graphique en temps réel des données (distance, durée de propagation, vitesse), dépassant largement les limites du moniteur série et offrant une interface plus adapté et exploitable.
  Globalement, ce travail démontre que l'association des technologies open-source (Arduino + Python) constitue une solution accessible, économique et performante pour la réalisation de systèmes de mesure et de détection embarqués. Malgré certaines limitations inhérentes au capteur HC-SR04 (sensibilité à la température, à l'angle de réflexion et aux conditions environnementales), les résultats obtenus soulignent la robustesse du montage et sa pertinence pédagogique et technique.
- Ce projet a permis de consolider des compétences pluridisciplinaires en physique des ondes, électronique embarquée, programmation microcontrôleur et science des données. Il ouvre également des perspectives d'amélioration intéressantes : compensation automatique de la température, filtrage numérique des signaux, développement d'une interface graphique avancée (Tkinter ou Matplotlib) ou encore intégration dans un système IoT.

- Cependant, certaines limites ont été observées, notamment la sensibilité du capteur aux variations de température, à l'humidité, à l'angle de réflexion et aux interférences environnementales. Ces contraintes soulignent la nécessité d'améliorations futures telles que la compensation automatique de la température, l'implémentation de filtres numériques ou le développement d'une interface graphique plus avancée
- Enfin ce mémoire illustre avec succès comment des outils simples et ouverts peuvent être utilisés pour concevoir des prototypes fonctionnels et éducatifs de qualité dans le domaine des capteurs et des systèmes embarqués.

Bibliographie:

[1] J. Joubert, Acoustique générale : Éléments de physique pour l'ingénieur, 2éme édition, Éditions Dunod, Paris, 2018.

[2] M. Bruneau, Manuel d'acoustique fondamentale, Éditions Hermès Science Publications, Paris, 2005.

[3] J. C. Valière, Acoustique de la parole et cavités résonantes, Presses Universitaires de France, Paris, 2011.

[4] T. D. Rossing, F. Wheeler, et P. Moore, The Science of Sound, 3rd Edition, Pearson Education, San Francisco, 2014.

[5] J. C. Valière, Acoustique de la parole et cavités résonantes, Presses Universitaires de France, Paris, 2011.

[6] A. Fichten, Célérité du son et mécanique des milieux déformables, Éditions Techniques de l'Ingénieur, 2016.

[7] D. G. Crighton, A. P. Dowling, et J. E. Williams, Modern Methods in Analytical Acoustics, Springer-Verlag, London, 2012.

[8] M. Rossi, Acoustique et électroacoustique, Presses Polytechniques et Universitaires Romandes (PPUR), Lausanne, 2007.

[9] L. Kinsler, A. Frey, A. Coppens, et J. Sanders, Fundamentals of Acoustics, 4th Edition, John Wiley & Sons, New York, 2000.

[10] H. Medwin et C. S. Clay, Fundamentals of Acoustical Oceanography: Sonar, Acoustics, and Marine Physics, Academic Press, San Diego, 1998.

[11] J. P. Lefebvre, Propagation linéaire des ondes ultrasonores dans les milieux élastiques, CNRS Éditions, Paris, 2013.

[12] G. Quentin, Physique des ultrasons et interactions rayonnement-matière, Éditions de Physique, Les Ulis, 2009.

[13] P. Langevin, Sur les palpeurs piézoélectriques et la densitométrie ultrasonore dans les milieux denses, Journal de Physique Théorique et Appliquée, 1998.

[14] J. F. de Belleval, Contrôle non destructif par ultrasons, Éditions Techniques de l'Ingénieur, traité Mesures et Contrôle, R1340, 2010.

[15] A. Preumont, Mechatronics: Dynamics of Electromechanical and Piezoelectric Systems, Springer Science & Business Media, Dordrecht, 2006.

[16] T. L. Szabo, Diagnostic Ultrasound Imaging: Inside Out, 2nd Edition, Academic Press, Elsevier, Boston, 2014.

[17] R. S. C. Cobbold, Foundations of Biomedical Ultrasound, Oxford University Press, New York, 2007.

[18] W. G. Cady, Piezoelectricity: An Introduction to the Theory and Applications of Electromechanical Phenomena in Crystals, McGraw-Hill, New York, 1946 (Dover Publications, New York, 2013).

[19] D. R. Raichel, The Science and Applications of Acoustics, 2nd ed. New York, NY, USA: Springer, 2006.

[20] K. Uchino, Ferroelectric Devices: Magnetostriction and Electrostriction Materials, 2nd Edition, CRC Press, Boca Raton, 2010.

[21] E. Freaks, HC-SR04 Ultrasonic Sensor User Guide and Technical Specifications, Documents techniques officiels, Shenzhen, 2019.

[22] Seeed Studio, Grove - Ultrasonic Distance Sensor (101020010) Hardware Library and Data Sheet, Document technique en ligne, 2021.

[23] J. M. Hughes, Practical Hardware Pentesting: Communication protocols and Sensor characterization, O'Reilly Media, Sebastopol, 2020.

[24] C. Tavard, Les capteurs de distance en robotique mobile : Étude comparative, Revue d'Électronique Embarquée, No. 42, pp. 18-25, 2022.

[25] M. Banzi et M. Shiloh, Getting Started with Arduino: The Open Source Electronics Prototyping Platform, 3rd Edition, Maker Media, Inc., San Francisco, 2014.

[26] H. S. Grewal, Evaluation of HC-SR04 Ultrasonic Sensor Precision in Temperature Varied Environments, International Journal of Embedded Systems and Applications (IJESA), Vol. 11, No. 3, pp. 34-45, 2023.

[27] D. Purdum, Beginning C for Arduino: Learn C Programming for the Arduino, 2nd Edition, Apress, New York, 2015.

[28] J. Blum, Exploring Arduino: Tools and Techniques for Engineering Wizardry, 2nd Edition, John Wiley & Sons, Indianapolis, 2019.

[29] C. Karvinen et T. Karvinen, Make: Sensors: A Hands-On Primer for Monitoring the Physical World with Arduino and Raspberry Pi, Maker Media, Inc., San Francisco, 2014.

[30] P. Scherz et S. Monk, Practical Electronics for Inventors, 4th Edition, McGraw-Hill Education, New York, 2016.

[31] G. Van Rossum et F. L. Drake, The Python Language Reference Manual, Network Theory Ltd., Bristol, 2011.

[32] A. Downey, Think Python: How to Think Like a Computer Scientist, 2nd Edition, O'Reilly Media, Sebastopol, 2016.

[33] L. Liechti, pySerial Documentation: Serial Port Extension for Python, Publication open-source en ligne, Version 3.5, 2020.

[34] J. Steiner, Firmata Protocol Specification: Direct hardware control over serial communication interfaces, Project Github Source Repository, Document de normalisation, 2018.

[35] J. D. N. Cheeke, Fundamentals and Applications of Ultrasonic Waves, 2nd ed. Boca Raton, FL, USA: CRC Press, 2012.

[36] J. Krautkrämer and H. Krautkrämer, Ultrasonic Testing of Materials. Berlin, Germany: Springer, 1990.

[37] D. Berlincourt, D. R. Curran, and H. Jaffe, “Piezoelectric and Piezomagnetic Materials,” in Physical Acoustics, vol. 1A, New York, NY, USA: Academic Press, 2003.

[38] W. P. Mason, Piezoelectric Crystals and Their Application to Ultrasonics. New York, NY, USA: Van Nostrand, 1998.

[39] M. G. Silk, Ultrasonic Transducers for Nondestructive Testing. Bristol, U.K.: Adam Hilger Ltd., 2004.

[40] S. Monk, Programming Arduino: Getting Started with Sketches, 2nd ed. New York, NY, USA: McGraw-Hill Education, 2016.

ملخص :

يسلط هذا المشروع الضوء على تصميم وتنفيذ جهاز استشعار بالموجات فوق الصوتية HC-SR04 متكامل مع لوحة أردوينو لقياس المسافات واكتشاف العوائق.

أظهرت التجارب التجريبية دقة عالية حيث بلغت سرعة الصوت المقاسة قرابة 343 م/ثا في درجة حرارة 20°م، وذلك ضمن نطاق قياس يمتد من 10 إلى 200 سم.

كما يعمل كاشف العوائق على تفعيل إنذار ضوئي وصوتي فوري عندما تقل المسافة عن 10 سم. أما واجهة بايثون فتسمح بعرض ديناميكي للبيانات عبر الاتصال التسلسلي (المسافة، زمن الطيران، السرعة).

تؤكد النتائج متانة النظام مع الإشارة إلى بعض القيود التقنية للحساسHC-SR04 ، خاصة تأثره بدرجة الحرارة وزاوية الانعكاس.

يُعد هذا المشروع نموذجاً ناجحاً لدمج الإلكترونيات والبرمجة ومعالجة البيانات.

الكلمات المفتاحية: مستشعر بالموجات فوق الصوتية، HC-SR04، أردوينو، لغة بايثون، قياس المسافة، كاشف العوائق.

Résumé :

Ce travail expose la conception d'un capteur ultrasonique HC-SR04 associé à Arduino pour la mesure de distance et la détection d'obstacles. Les expérimentations ont validé une célérité du son mesurée proche de 343 m/s à 20°C, avec une bonne précision de mesure sur une plage de 10 à 200 cm.

Le détecteur d'obstacles déclenche efficacement une alarme lumineuse et sonore en dessous de 10 cm. L'interface Python, via communication série, permet une visualisation dynamique des données (distance, temps de vol, vitesse).

Les résultats confirment la robustesse du montage tout en mettant en évidence les limites du HC-SR04 (sensibilité à la température et à l'angle).

Ce projet combine avec succès électronique, programmation et traitement de données.

Mots-clés : Capteur ultrasonique, HC-SR04, Arduino, Python, mesure de distance, détecteur d'obstacles.

Abstract :

This work presents the design of an ultrasonic sensor based on the HC-SR04 module combined with Arduino for distance measurement and obstacle detection.

The experiments validated a measured speed of sound close to 343 m/s at 20°C, with good measurement accuracy over a range of 10 to 200 cm.

The obstacle detector effectively triggers a light and sound alarm when the distance falls below 10 cm. The Python interface, through serial communication, enables dynamic visualization of the data (distance, time of flight, and speed).

The results confirm the robustness of the setup while highlighting the limitations of the HC-SR04 sensor (sensitivity to temperature and angle of incidence).

This project successfully combines electronics, programming, and data processing.

Keywords: Ultrasonic sensor, HC-SR04, Arduino, Python, distance measurement, obstacle detection,