REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

Université de Mohamed El-Bachir El-Ibrahimi - Bordj Bou Arreridj

Faculté *des Sciences et de la technologie*

Département d'Electronique

Rapport

Projet de Fin de Cycle (PFC)

MCIL 3

FILIERE : **Électronique**

Spécialité : Industries Électroniques

Par

- **BENHAMZA Houssam Eddine**
- **DJOUADA Ayoub**

Intitulé

Étude et Réalisation d'une Carte Électronique de Gestion d'un Afficheur LCD

Présenté le : ……

Devant le Jury composé de :

Nom & Prénom	*Grade*	*Qualité*	*Etablissement*
Mme A. LAOUAMRI	*MAA*	*Président*	*Univ-BBA*
Mr A. BOUSSAHOUL	*MAA*	*Examinateur*	*Univ-BBA*
Mme N. BIOUD	*MCA*	*Encadreur*	*Univ-BBA*
Mr L. BOUTAHAR		*Coencadreur*	*Univ-BBA*

Année Universitaire 2021/2022

Remerciements

NOUS remercions dieu le Tout Puissant de nous avoir donner les aptitudes pour mener à bien ce modeste travail.

Un grand merci pour notre promoteur Mme N. BIOUD et co-encadreur Mr L.BOUTAHAR pour sa disponibilité, son suivi et ses conseils tout au long de notre travail.

Nous tenons également à remercier les membres du jury qui nous ont fait l'honneur de juger notre travail.

Nous remercions aussi tous les enseignants du département d'électronique.

Pour finir nous remercions nos parents et nos amis qui nous ont soutenus durant notre cursus universitaire.

Dédicaces

✓ Nous dédions ce travail à :

✓ Nos chers mères et nos chers pères

✓ Nos Chers frères et sœurs

✓ Toute la famille

✓ Tous les amis

✓ Tous les collèges de la promotion 2022

✓ Tous ceux que nous aimons

✓ Tous ceux qui nous aiment

✓ Tous ceux qui nous connaissent :

- Djouada ayoub
- Ben hamza houssam eddine

Liste des figures

Liste des tableaux

Sommaire

CHAPITRE 2 : SIMULATION ET REALISATION PRATIQUE

Introduction Générale

Introduction Générale

Les afficheurs à cristaux liquides ou LCD (Liquid Crystal Display), sont des modules qui nécessitent peu de composants externes pour un bon fonctionnement. Ils consomment relativement peu (de 1 à 5 mA), sont relativement bons marchés et s'utilisent avec beaucoup de facilité.

Notre projet consiste à réaliser un système de gestion d'un afficheur LCD en manipulant ou en modifiant les message saisis dans le code source téléversé dans le microcontrôleur de la carte Arduino, aussi les valeurs des différentes grandeurs physiques mesurées par des capteurs câblés à la carte Arduino, tel que le capteur de température.

Notre mémoire est esquissée en deux chapitres : le premier traite l'étude théorique de la carte de commande ainsi que les afficheurs LCD, le deuxième consiste à simuler, réaliser et tester notre dispositif.

Enfin, nous terminerons notre mémoire avec une conclusion générale.

CHAPITRE 01
CARTE DE COMMANDE ET AFFICHEURS LCD

1.1 Introduction

Dans ce chapitre, nous allons définir la carte Arduino et ses applications , aussi nous allons présenter une étude détaillée sur ses (entrées /sorties) numériques et analogiques, sa structure et son alimentation.

Une étude descriptive de l'afficheur LCD est nécessaire pour faciliter son implantation dans notre réalisation.

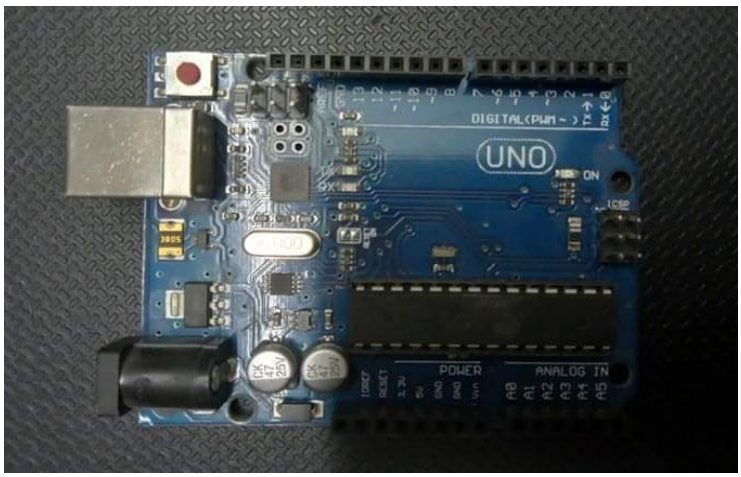

Figure 1.1 : carte Arduino

1.2 Carte Arduino

Arduino est un ensemble matériel et logiciel qui permet d'apprendre l'électronique (en s'amusant) tout en se familiarisant avec la programmation informatique. Arduino est en source libre ; vous pouvez donc télécharger le schéma d'origine et l'utiliser pour élaborer votre propre carte et la vendre sans payer des droits d'auteur. [1]

1.2.1matériel Arduino

Ce sont des cartes électroniques programmables (donc dotées d'un processeur et de mémoire) sur lesquelles nous pouvons brancher des capteurs de température, d'humidité, de vibration ou de lumière, une caméra, des boutons, des potentiomètres de réglage, des contacts électriques…Il y a aussi des connecteurs pour brancher des LED, des moteurs, des relais, des afficheurs, un écran…Une carte Arduino est un cerveau qui permet de rendre intelligent des systèmes électroniques et d'animer des dispositifs mécaniques. [2]

1.2.2 Logiciel Arduino

Les créateurs de Arduino ont développé un logiciel pour que la programmation des cartes Arduino soit visuelle, simple et complète à la fois.
C'est ce que l'on appelle une IDE, qui signifie *Integrated Development Environment* ou Environnement de Développement « Intégré » en français (donc EDI). L'IDE Arduino est le logiciel qui permet de programmer les cartes Arduino . L'IDE affiche une fenêtre graphique qui contient un éditeur de texte et tous les outils nécessaires à l'activité de programmation. Vous pouvez donc saisir votre programme, l'enregistrer, le compiler, le vérifier, le transférer sur une carte Arduino …

A la date de rédaction de cette page, la version la plus récente de l'IDE Arduino est la *1.8.10*. L'aspect est à peu près identique sur chaque plate-forme (Windows, Mac et Linux). L'image suivante montre l'écran initial qui apparaît au lancement de l'IDE. [3]

Figure 1.2: logo de l'Arduino

Figure 1.3: L'écran principal de l'IDE Arduino au démarrage

1.3 Comparatif des cartes Arduino utilisables

Tableau 1.1:Tableau comparatif des différentes carte Arduino

Carte Arduino	Microcontrôleur	Type (bits)	CLK (MHZ)	Alim(V)
Micro	ATmega32u4	8	16	5
Leonardo	ATmega32u4	8	16	5
Uno	ATmega328p	8	16	5
Nano	ATmega328P	8	16	5
MKR Zero	ATSAMD21G18	32	48	3.3
101	Intel curie	32	32	3.3
Due	AT91SAM3X8E	32	84	3.3
Mega	ATmega2560	8	16	5
Zero	ATSAMD21G18	32	48	3.3
Mo pro	ATSAMD21G18	32	48	3.3

1.4 Entrées-sorties numériques

Quasiment toutes les broches d'un Arduino peuvent être programmées en entrée ou sortie numérique (et non les deux en même temps). Par exemple,48 sur l'Arduino Uno , il s'agit d'une part des broches numérotées de 0 à 13 mais également32 des broches A0 à A5. Une broche programmée ainsi peut être : [3]

Une entrée. Le programme peut lire une tension présente sur cette broche en utilisant digitalRead**(...).** Comme cette tension est interprétée comme un chiffre binaire (0 ou 1), la datasheet du MCU de l'Arduino Uno garantit que toute tension inférieure à 0,3 x Vcc, Vcc étant égale à 5V, soit 1,5V sera comprise comme un 0 et que toute tension supérieure à 0,6 x Vcc, soit 3V, sera comprise comme un 1. Entre les deux, c'est flou. digitalRead**(...)** renverra de toutes façons un 0 ou un 1 mais de manière plus ou moins aléatoire et variable selon l'Arduino utilisé.[3]

Une sortie. Le programme peut écrire un chiffre binaire, au moyen de digital Write(...), chiffre qui dans le programme sont nommées HIGH pour le 1 et LOW pour le 0, qui sera traduit en une tension de 5V pour le 1 et de 0V pour le 0. Attention toutefois, cette tension peut respectivement être plus basse ou plus haute si le courant qui est tiré de la broche commence à être important.[4]

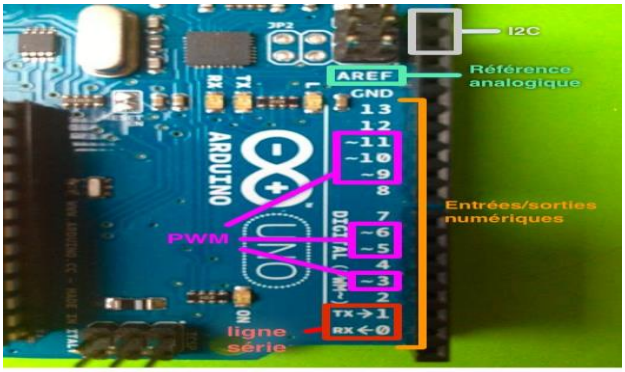

Figure 1.4:Entrées-sorties numériques de l'Arduino Uno

Les broches numérotées de 0 à 13 sont les entrées sorties numériques. Certaines d'entre elles peuvent également être utilisées en sortie PWM (en rose), les broches 0 et 1 servent aussi à la communication série (en rouge). L'I2C en gris est en haut et non étiqueté sur la carte.

Il est préférable de ne pas dépasser une consommation de 20 mA sur une broche programmée en sortie et absolument nécessaire de ne pas dépasser 40 mA sous peine de risque la destruction de la sortie du MCU. Il s'agit aussi de ne pas dépasser au total 200 mA.

Les sorties numériques sont employées de nombreuses manières : alimentation d'une DEL pour un éclairage ou un témoin, commande d'un transistor, connexion à un afficheur LCD, etc.

Les entrées numériques permettent également de connecter de nombreuses choses : interrupteur, bouton poussoir, interrupteurs à lame souple REED mais aussi des capteurs de présence plus sophistiqués.[3]

1.4.1 Entrées analogiques

Les Arduino sont pourvus de 6, pour le Uno, à 16 entrées analogiques, pour le Méga. Ce sont les broches étiquetées *A* suivie d'un nombre. Les tensions, toujours entre 0 et 5V, présentes sur ces broches, peuvent être numérisées via un convertisseur analogique-numérique ou ADC (Analog Digital Converter). La fonction analog Read**(...)** remplit ce rôle.[5]

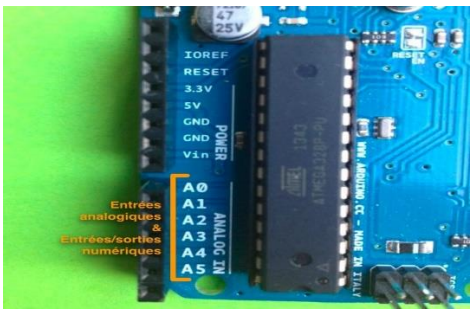

Figure 1.5:Entrées analogiques de l'Arduino Uno

Ces broches peuvent également être utilisées comme entrées-sorties numériques.

Le convertisseur des Arduino à base d'AVR effectue une conversion sur 10 bits, c'est à dire qu'il convertit la tension en un nombre entier ayant une valeur de 0 à 1023.[4]

0 correspond au 0V de la tension et 1023 au 5V. La résolution, c'est à dire la différence entre deux valeurs successives de la tension correspondant à une différence de 1 sur l'entier résultat, est donc d'environ 5mV.[4]

Il est possible de changer la tension de référence de la conversion analogique-numérique, c'est à dire la tension comprise comme 1023. Le MCU des Arduino possède une référence de tension interne de 1,1V qui peut être sélectionné à la place de la tension d'alimentation. Cela permet une meilleure résolution, environ 1mV, sur les faibles valeurs de tension à convertir. Il est également possible de sélectionner la tension fournie sur la broche AREF comme tension de référence.[4]

Plusieurs types de capteurs analogiques peuvent trouver place sur ces broches. On citera la mesure de courant traversant une résistance ou la mesure d'intensité lumineuse via une photorésistance.[5]

1.5 Schéma d'une platine Arduino Uno

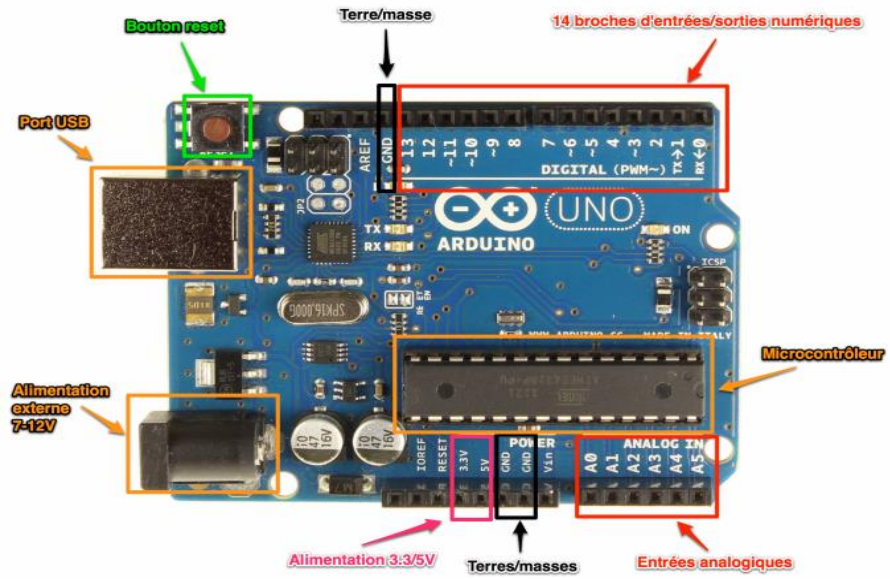

Figure 1.6:Source : arduino.develoopez.com

1.6 Microcontrôleur

C'est le cerveau de notre carte. Il va recevoir le programme que nous allons créer et va le stocker dans sa mémoire avant de l'exécuter. Grâce à ce programme, il va savoir faire des choses, qui peuvent être : faire clignoter une LED, afficher des caractères sur un écran, envoyer des données à un ordinateur, mettre en route ou arrêter un moteur… Il existe deux modèles d'Arduino Uno: l'un avec un microcontrôleur de grande taille, et un autre avec un microcontrôleur dit SMD (SMD : Surface Mounted Device, soit composants montés en surface, en opposition aux composants qui traversent la carte électronique et qui sont soudé du

côté opposé). D'un point de vue utilisation, il n'y a pas de différence entre les deux types de microcontrôleurs. [6]

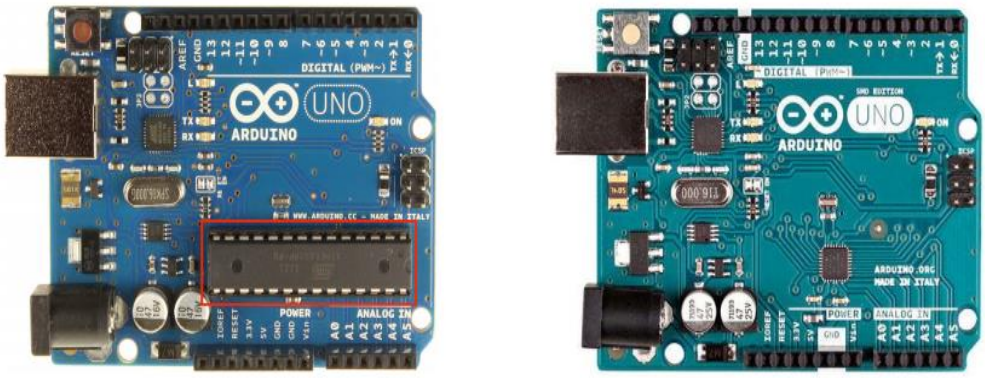

Figure 1.7 : Microcontrôleur

1.7 Alimentation des cartes Arduino

En général, on considère que tout est à peu près bon pour alimenter une carte Arduino : c'est une grosse erreur, et il faut au contraire soigneusement étudier l'alimentation du module Arduino et des périphériques et autres dispositifs associés.

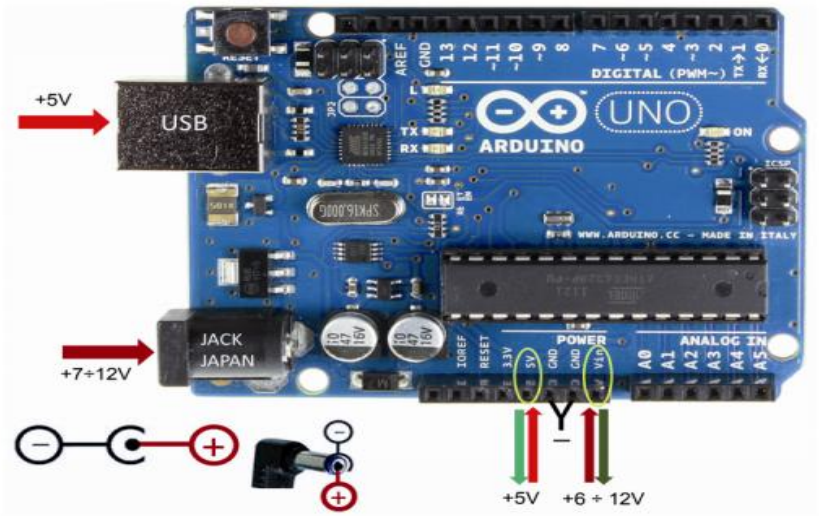

Figure 1.8: Alimentation des cartes Arduino

On voit qu'on a 3 possibilités pour alimenter la carte (qui demande du 5v) :

Alimentation directe sur la broche 5v : dans ce cas, il faut impérativement que le jack ne soit pas alimenté, mais l'USB peut l'être (utilisation de la transmission de données), le circuit constitué du comparateur LM85 U1A et du MOSFET T1 évitera le court-circuit entre la tension 5v du port USB et la tension mise sur la broche 5V. Il faudra protéger l'entrée contre les surtensions, inversions, transitoires (selon la longueur de la ligne d'alimentation, une paire de condensateurs chimiques/ céramique, une self, une diode de clamping …). Aucun régulateur ne limite la puissance fournie par cette entrée 5v. Il faudra fournir sur cette broche l'intensité nécessaire :

- À l'alimentation de l'Arduino et aux entrées sorties,
- Aux shields alimentés par lui
- Au régulateur 3.3v

Alimentation sur la broche Vin : dans ce cas la broche 5v est alimentée par la tension Vin abaissée par le régulateur IC1/IC2. Là aussi, l'absence impérative de tension sur le jack permet au circuit comparateur MOSFET d'isoler l'USB. Par contre dans ce cas le régulateur 5V de l'Arduino est actif, et son échauffement limite l'intensité délivrable aux circuits alimentés ; en reprenant les valeurs de l'article ci-dessous, il faudra limiter à :

- Alimentation 12 V : I = 2 / (12-5) = 2 / 7 = 285mA
- Alimentation 9 V : I = 2 / (9-5) = 2/4 = 500mA (vous pouvez utiliser un adaptateur pour pile 9V par exemple.)
- Alimentation 7 V : I = 2 / (7-5) = 2/2 = 1A

Alimentation par le jack : ce cas est identique au cas précédent, à une petite différence prête : une diode M7 (équivalent de 1N4007 en montage de surface) est placée pour protéger contre l'inversion de polarité, la tension nécessaire sera donc un peu plus élevée que pour l'alimentation directe de Vin. Sinon, le 5v sera toujours obtenu par le régulateur et les intensités acceptables seront les mêmes, en tenant compte toutefois de la puissance dissipée par la diode (grosso modo 1vxIntensité).

Alimentation par le port USB : dans ce cas, l'ensemble des composants alimentés ne devront pas consommer plus de 500mA (fusible automatique sur la carte), à vérifier que

l'appareil peut les fournir ! Le 5v sera fourni directement, en passant par le MOSFET dont la résistance en mode conduction est faible ; on se retrouvera donc dans les mêmes conditions que pour l'alimentation en direct sur le 5V, avec la limitation de l'intensité utilisable à 500mA.[7]

En sortie de la carte, on pourra selon les cas utiliser :

- Si la carte est alimentée par le 5v, on pourra relier celui-ci aux éléments extérieurs sans influence sur la carte
- Si la carte est alimentée par Vin, on pourra alimenter des composants extérieurs via le 5v dans la limite des intensités vues plus haut (échauffement du régulateur)
- Si la carte est alimentée par USB, on pourra alimenter des composants extérieurs via le 5v dans la limite de la capacité de l'appareil connecté et du fusible, soit 500mA maximum.
- Si la carte est alimentée par le jack, on pourra alimenter par Vin des éléments extérieurs, avec une tension égale à environ Vin – 1V et une intensité inférieure à 1A (diode M7 de protection)

Dans tous les cas, on pourra utiliser la sortie 3.3V pour alimenter des éléments 3.3v, mais ATTENTION, l'intensité de sortie est très limitée (50mA) et notamment ne suffit même pas pour alimenter un simple module ESP8266. [7]

1.8 Connectique

Part une LED sur la broche 13, la carte Arduino ne possède pas de composants (résistances, diodes, moteurs À …) qui peuvent être utilisés pour un programme. Il est nécessaire de les rajouter. Mais pour cela, il faut les connecter à la carte. C'est là qu'interviennent les connecteurs, aussi appelés broches (*pins*, en anglais).
Sur les Arduino et sur beaucoup de cartes compatibles Arduino, les broches se trouvent au même endroit. Cela permet de fixer des cartes d'extension, appelée shields en les empilant.

1.9 Exploration des broches Arduino

- ✓ **0 à 13** Entrées/sorties numériques ;
- ✓ **A0 à A5** Entrées/sorties analogiques ;
- ✓ **GND** Terre ou masse (0 V) ;
- ✓ **5 V** Alimentation +5 V ;
- ✓ **3.3 V** Alimentation +3.3 V ;
- ✓ **Vin** Alimentation non stabilisée (= le même voltage que celui à l'entrée de la carte).

Figure 1.9: files de connexion

1.10 Afficheurs LCD

L'afficheur LCD est en particulier une interface visuelle entre un système (projet) et l'homme (utilisateur). Son rôle est de transmettre les informations utiles d'un système à un utilisateur. Il affichera donc des données susceptibles d'être exploiter par l'utilisateur d'un système.

Afficheur peut désigner : afficheur, une personne qui appose des affiches (on parle le plus souvent de « colleur d'affiche ») ; afficheur numérique, en électronique, un dispositif comportant un ou plusieurs emplacements où l'on peut représenter un chiffre, une lettre ou un symbole

Plusieurs afficheurs sont disponibles sur le marché et diffèrent les uns des autres, non seulement par leurs dimensions, (de 1 à 4 lignes de 6 à 80 caractères), mais aussi par leurs caractéristiques techniques et leur tension de service. Certains sont dotés d'un rétroéclairage de l'affichage. Cette fonction fait appel à des LED montées derrière l'écran du module, cependant, cet éclairage est gourmand en intensité (de 80 à 250 mA) [8]

Figure 4.10 :Afficheur LCD

1.11 Présentation d'un afficheur LCD

LCD signifie « Liquide Crystal Display » et se traduit, en français, par « Écran à Cristaux Liquides.

Il y'a une différence avec des écrans à LED. Il en existe deux types :

- les écrans à rétro-éclairage LED : ceux sont des écrans LCD tout à fait ordinaires qui ont simplement la particularité d'avoir un rétro-éclairage à LED à la place des tubes néons. Leur prix est du même ordre de grandeur que les LCD « normaux ». En revanche, la qualité d'affichage des couleurs semble meilleure comparés aux LCD « normaux ».

- les écrans à affichage LED : ceux si ne disposent pas de rétro-éclairage et ne sont ni des écrans LCD, ni des plasma. Ce sont des écrans qui, en lieu et place des pixels, se trouvent des LED de très petite taille. Leur coût est prohibitif pour le

moment, mais la qualité de contraste et de couleur inégale tous les écrans existants !

1.12 Principe de fonctionnement

Comme son nom l'indique, un écran LCD possède des cristaux liquides. Mais ce n'est pas tout ! En effet, pour fonctionner il faut plusieurs choses. Si vous regardez de très près votre écran (éteint) vous pouvez voir une grille de carré. Ces carrés sont appelés des pixels (de l'anglais « Picture Element », soit « Élément d'image » en français).

Chaque pixel est un cristal liquide. Lorsque aucun courant ne le traverse, ses molécules sont orientées dans un sens (admettons, 0°). En revanche lorsqu'un courant le traverse, ses molécules vont se tourner dans la même direction (90°). Voilà pour la base.

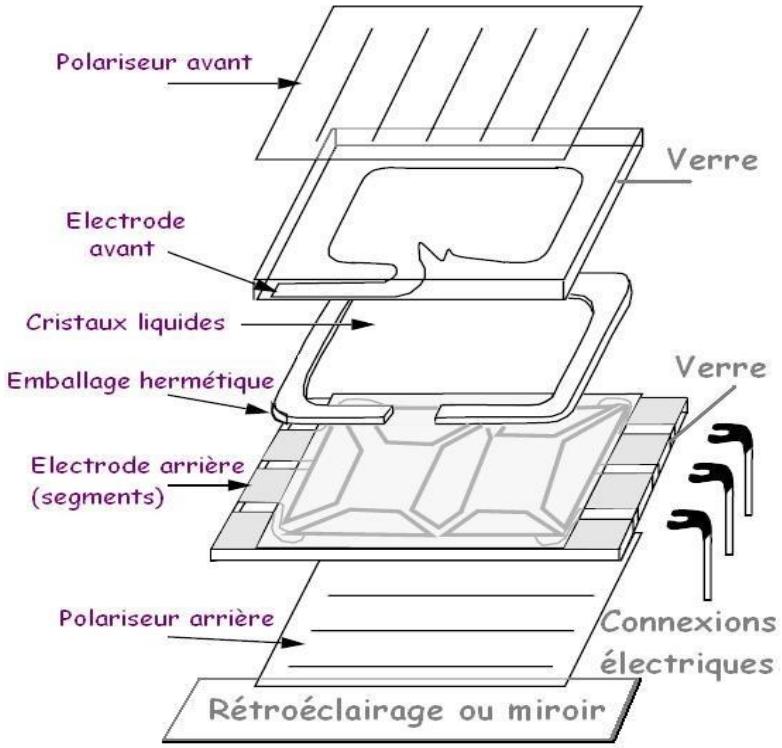

Figure 1.11 : principe de fonctionnement d'un l'écran LCD

1.13 Principe de fonctionnement

Dans la grande famille afficheur LCD, on distingue plusieurs catégories :

- Les afficheurs alphanumériques
- Les afficheurs graphiques monochromes
- Les afficheurs graphiques couleur

Les premiers sont les plus courants. Ils permettent d'afficher des lettres, des chiffres et quelques caractères spéciaux. Les caractères sont prédéfinis (voir table juste audessus) et on a donc aucunement besoin de gérer chaque pixel de l'écran. Les seconds sont déjà plus avancés. On a accès à chacun des pixels et on peut donc produire des dessins beaucoup plus évolués. Ils sont cependant légèrement plus onéreux que les premiers. Les derniers sont l'évolution des précédents, la couleur en plus (soit 3 fois plus de pixels à gérer : un sous-pixel pour le rouge, un autre pour le bleu et un dernier pour le vert, le tout forme la couleur d'un seul pixel).

1.14 Types d'un afficheur LCD

1.14.1 Afficheurs alphanumériques

Les afficheurs alphanumériques sont utilisés pour la visualisation de paramètres ou pour donner des indications à l'utilisateur d'une machine. L'afficheur est un afficheur à segments ou à matrice de points, de technologie LED ou LCD. Il comporte une ou plusieurs lignes de caractères affichables.[9]

1.14.2 Afficheurs graphique

C'est un afficheur LCD graphique 128x64 pixels

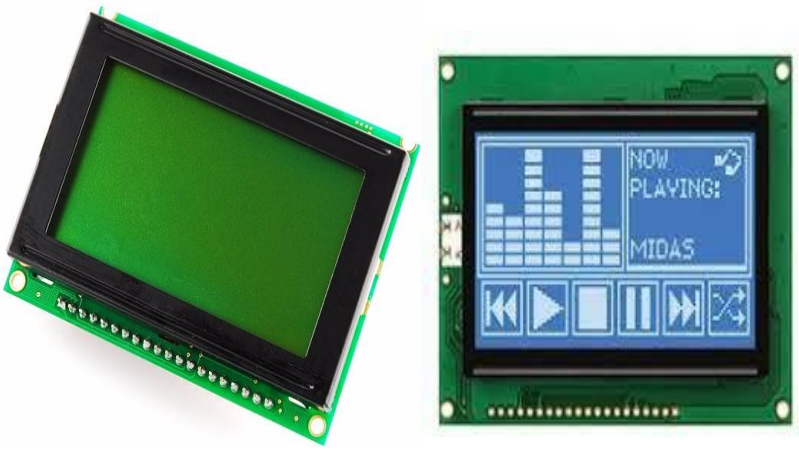

Figure1.12 : Afficheur graphique (monochrome)

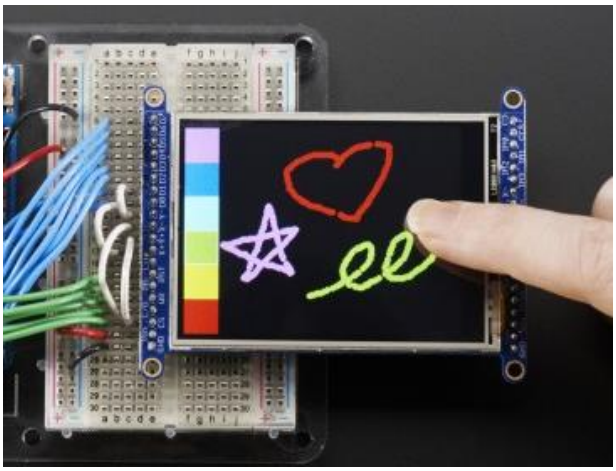

Figure 1.13 : Afficheur graphique (couleur)

1.15 Broches et schéma fonctionnel

l'afficheur lcd utilise 6 à 10 broches de données ((D0 à D7) ou (D4 à D7)+RS+E) et deux d'alimentations (+5V et masse). La plupart des écrans possèdent aussi une entrée analogique pour régler le contraste des caractères nous brancherons dessus un potentiomètre de 10 KOhms .Les 10broches de donnés peuvent être placées sur n'importe quelles entrée/sorties numériques de l'Arduino. En effet , nous indiquerons ensuite à la librairie LiquidCrystal qui est branché ou le montage à 8 broches de données

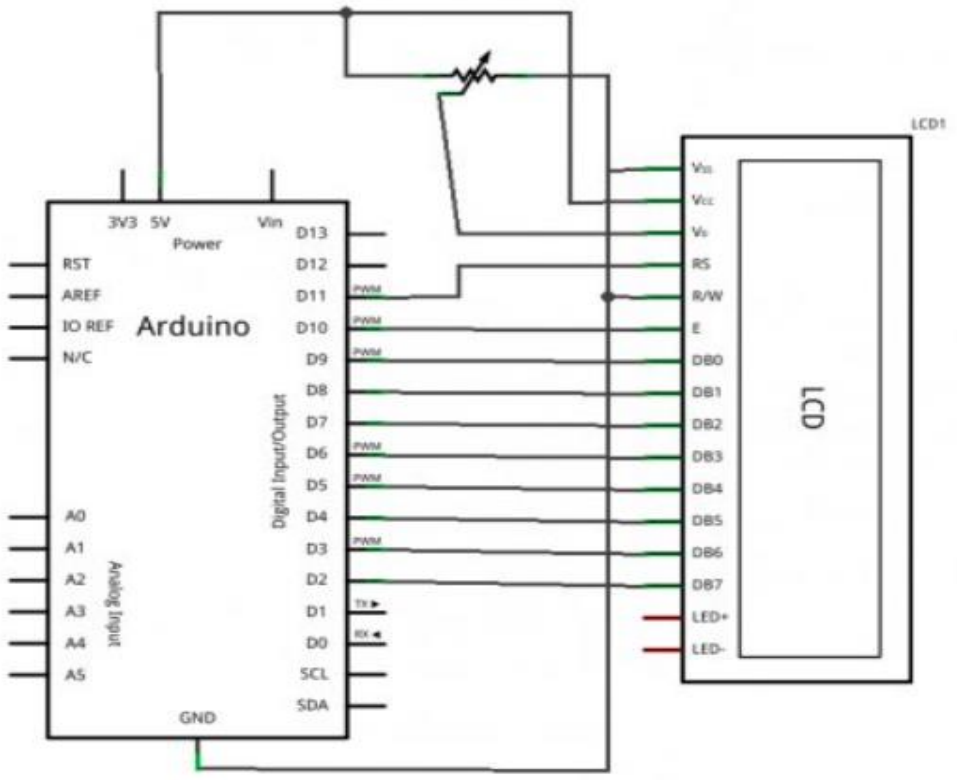

Figure 1.14 :schéma fonctionnel

1.16 Commande

Normalement, pour pouvoir afficher des caractères sur l'écran il nous faudrait active individuellement chaque pixel de l'écran.

Un caractère est représenté par un bloc de 7*5 pixels. Ce qui fait qu'un écran de 16 colonnes et 2 lignes représente un total de 16*2*7*5 = 1120 pixels ! Heureusement c'est la tâche est plus simplifiée.

1.16.1 Décodeur de caractères

Tout comme il existe un driver vidéo pour votre carte graphique d'ordinateur, il existe un driver « LCD » pour votre afficheur.

Ce composant va servir à décoder un ensemble « simple » de bits pour afficher un caractère à une position précise ou exécuter des commandes comme déplacer le curseur par exemple. Ce composant est fabriqué principalement par Hitachi et se nomme le

HC44780. Il sert de décodeur de caractères. Ainsi, plutôt que de devoir multiplier les signaux pour commander les pixels un à un, il nous suffira d'envoyer des octets de commandes pour lui dire « écris moi 'LICENCE' à partir de la colonne 3 sur la ligne 1 ».

Ce composant possède 16 broches :

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

Normalement, pour tous les écrans LCD (non graphiques) ce brochage est le même. Donc pas d'inquiétude lors des branchements, il vous suffira de vous rendre sur cette page pour consulter le tableau. Par la suite, les broches utiles qu'il faudra relier à l'Arduino sont les broches 4, 5 (facultatives), 6 et les données (7 à 14 pouvant être réduite à 8 à 14) en oubliant pas l'alimentation et la broche de réglage du contraste. Ce composant possède tout le système de traitement pour afficher les caractères.

Il contient dans sa mémoire le schéma d'allumage des pixels pour afficher chacun d'entre eux.

1.17 Table des caractères affichables

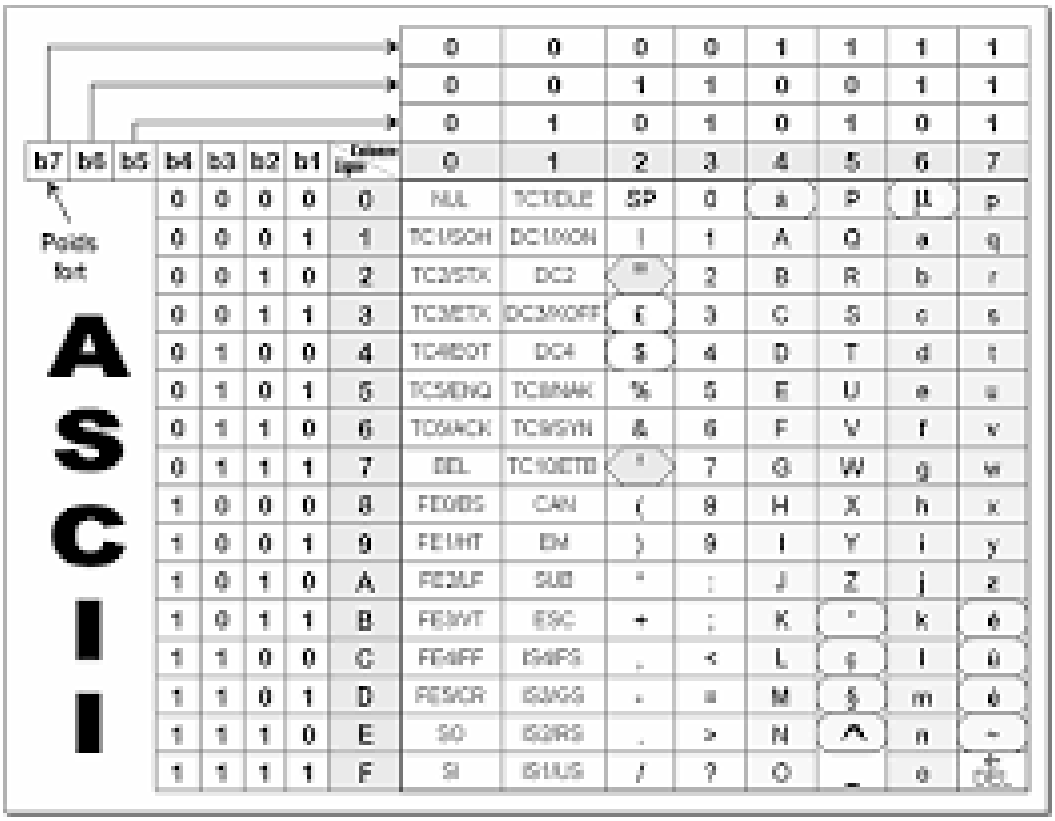

Figure 1.15: tableau des caractères affichables

1.18 Communication avec l'écran

1.18.1 communication parallèle

De manière classique, on communique avec l'écran de manière parallèle. Cela signifie que l'on envoie des bits par blocs, en utilisant plusieurs broches en même temps(opposée à une transmission série où les bits sont envoyés un par un sur une seule broche).Nous utilisons 10 broches différentes, 8 pour les données (en parallèle donc) et 2 pour de la commande (E : Enable et RS : Registre Selector).

La ligne R/W peut être connecté à la masse si l'on souhaite uniquement faire de l'écriture. Pour envoyer des données sur l'écran, c'est en fait assez simple. Il suffit de suivre un ordre logique et un certain timing pour que tout se passe bien. Tout d'abord, il nous faut placer la broche RS à 1 ou 0 selon que l'on veut envoyer une commande, comme par exemple « déplacer le curseur à la position (1;1) » ou que l'on veut envoyer une donnée : « écris le caractère 'a' « . Ensuite, on place sur les 8 broches de données (D0 à D7) la valeur de la donnée à afficher. Enfin, il suffit de faire une impulsion d'au moins 450 ns pour indiquer à l'écran que les données sont prêtes.

Cependant, comme les ingénieurs d'écrans sont conscients que la communication parallèle prend beaucoup de broches, ils ont inventé un autre mode que j'appellerai « semi-parallèle ». Ce dernier se contente de travailler avec seulement les broches de données D4 à D7 (en plus de RS et E) et il faudra mettre les quatre autres (D0 à D3) à la masse. Il libère donc quatre broches. Dans ce mode, on fera donc deux fois le cycle « envoi des données puis impulsion sur E » pour envoyer un octet complet.

Nous utiliserons une libraire nommée LiquidCrystal qui se chargera de gérer les timings et l'ensemble du protocole.

Pour continuer, le mode « semi-parallèle » sera choisi. Il nous permettra de garder plus de broches disponibles pour de futurs montages et est souvent câblé par défaut dans de nombreux shields (dont le mien). La partie branchement.

1.18.2 Communication série

Lorsque l'on ne possède que très peu de broches disponibles sur notre Arduino, il peut être intéressant de faire appel à un composant permettant de communiquer par voie série avec l'écran. Un tel composant se chargera de faire la conversion entre les données envoyées sur la voie série et ce qu'il faut afficher sur l'écran. Le gros avantage de cette solution est qu'elle nécessite seulement un seul fil de donnée (avec une masse et le VCC) pour fonctionner là où les autres méthodes ont besoin de presque une dizaine de broches.

En effet, elle nous permet de garder l'approche « standard » de l'écran et nous permet de garder la liaison série pour autre chose (encore que l'on pourrait en émuler une sans trop de difficulté).

1.18.3 Communication I2C

Un dernier point à voir, c'est la communication de la carte Arduino vers l'écran par la liaison I2C. Cette liaison est utilisable avec seulement 2 broches (une broche de donnée et une broche d'horloge) et nécessite l'utilisation de deux broches analogiques de l'Arduino (broche 4 et 5) [10]

1.19 Utilisation l'afficheur LCD 16×2

L'afficheur texte 16×2 est utilisé pour afficher les informations du capteur, afficher les menus ou les invites. L'écran affiche des caractères noirs d'une taille de 5×8 pixels. Le rétroéclairage intégré s'allume en appliquant une alimentation aux broches du module

1.20 Conclusion

La collecte et la maitrise des connaissances théoriques est une étape primordiale préparative pour toute réalisation pratique, car elle permet de lister les besoins et d'optimiser la planification de l'élaboration du dispositif . Dans le chapitre suivant, nous aborderons la réalisation pratique.

CHAPITRE 02
SIMULATION
ET
REALISATION
PRATIQUE

2.1 Introduction

Dans ce chapitre, nous allons présenter le schéma électrique de notre projet, ainsi que la simulation qui est réalisée par tinkercad.

Aussi, on présentera le code source de notre projet qui est téléversé au sein du microcontrôleur de l'Arduino responsable de la gestion de l'afficheur.

2.2 Notre projet

Notre projet c'est l'étude et réalisation d'une carte électronique de gestion d'un afficheur public avec l'Arduino , Nous avons réalisé le suivant :

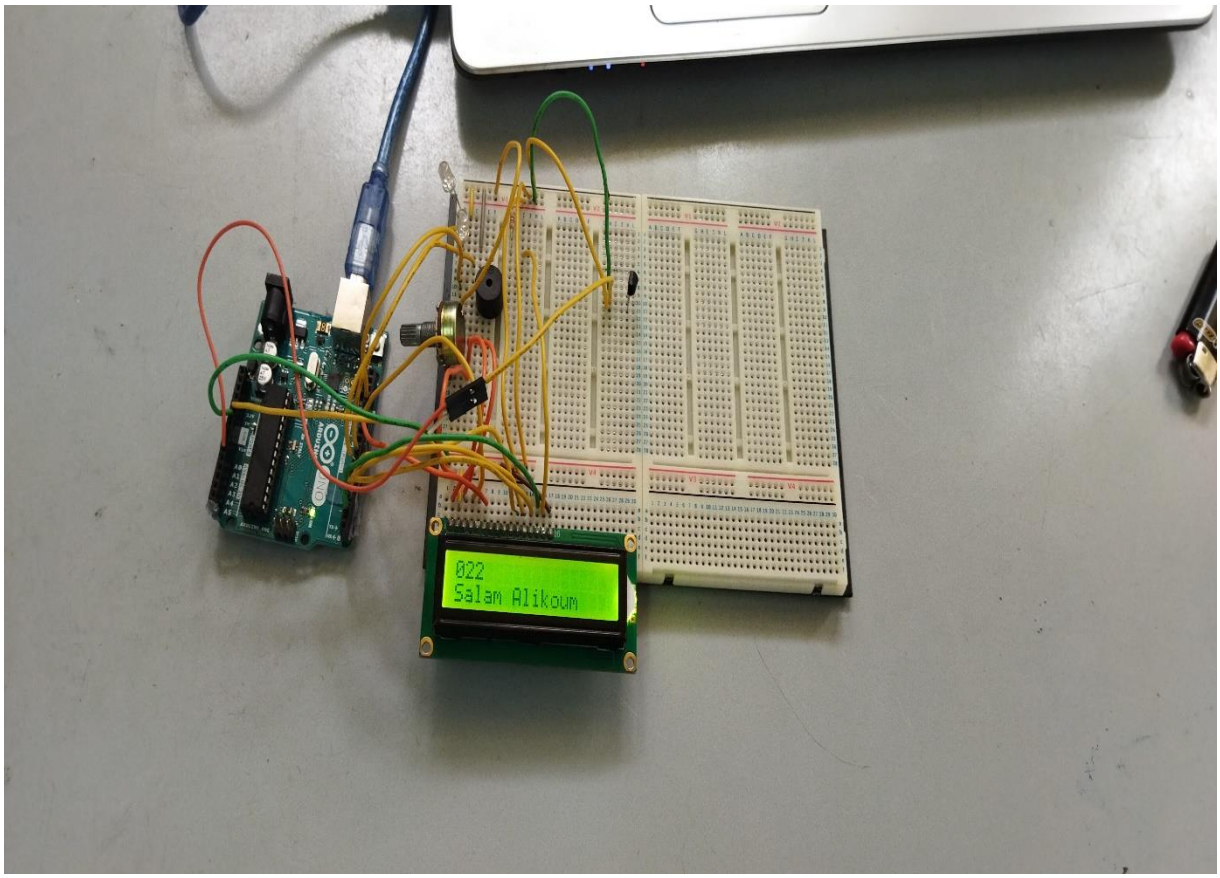

Figure 2.1: projet

2.3 Travail de projet

Ce projet mesure la température au moyen d'un Captur de température TMP36, qui la mesure et l'affiche sur un afficheur LCD 16x2, en plus d'afficher la phrase ¨ MCIL03 2022 ¨ et ¨ Salam Alikoum¨ Lorsque la température dépasse un certain degré, le bazar émet un son et le LED rouge s'allume, indiquant qu'il y a un danger en plus du changement de température sur l'afficheur LCD 16x2, Lorsque la température descend au point où le danger est passé, le buzzer arrête de faire du bruit et le LED vert allume, Voici une indication que le danger est passé.

2.4 composants nécessaires dans ce projet

Tableau 2.1 : tableau des composants

NOM	QUANTITE	COMPOSANT
U4	01	Arduino Uno U3
U5	01	Ecran LCD 16x2
Rpot 2	01	10Kohm Potentiomètre
R2	01	220ohm Resistance
U2	01	Capteur de température
D1	01	Rouge LED
D2	01	Vert LED
PIEZ01	01	Elément piézoélectrique

Figure 2.1:Arduino Uno

Figure 2.4:Potentiomètre 10kohm

Figure 2.2:LED

Figure 2.3:Afficheur LCD 16*2

Figure 2.4 : TMP36

Figure 2.5 : Résistance 220 ohm

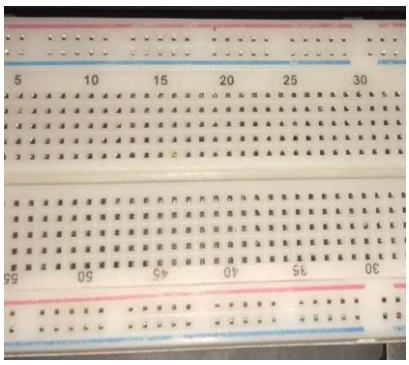

Figure 2.6:carte d'essai

Figure 2.7:PIEZ01

2.5 Schéma électrique

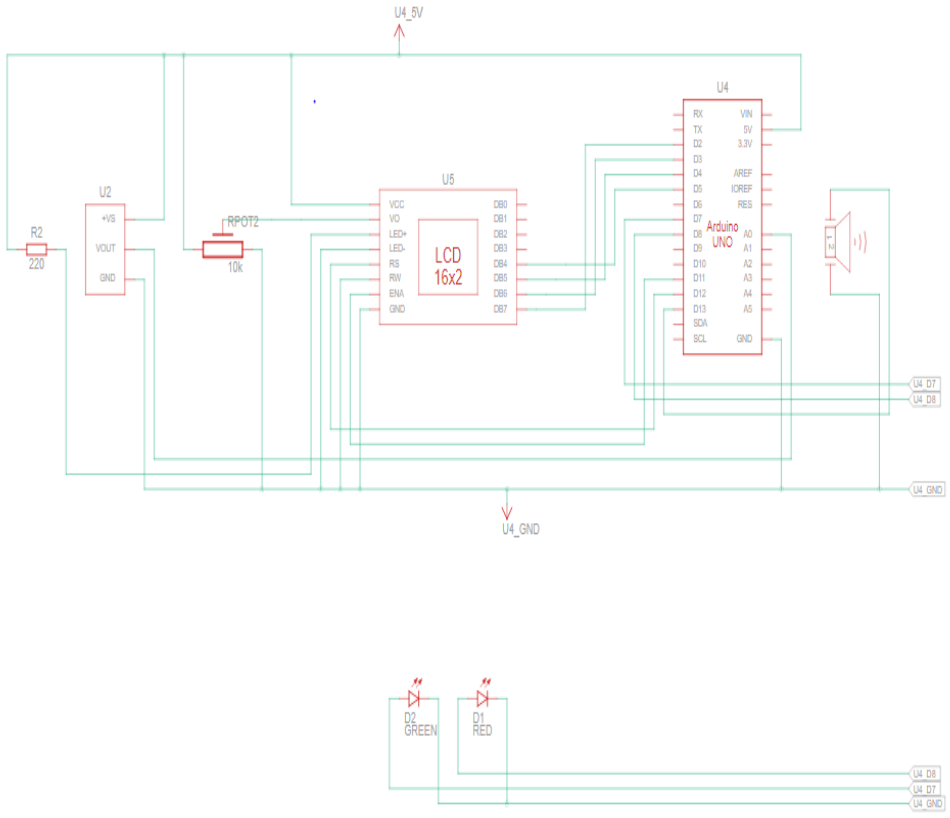

Figure 2.8:schéma électrique de notre projet. (tinkercad)

2.6 Simulation

Dans ce projet, nous avons utilisé un site de simulation appelé **tinkercad** :

Figure 2.9: logiciel tinkercad

2.7 Définition tinkercad

Tinkercad est un programme de modélisation 3D en ligne gratuit qui s'exécute dans un navigateur Web et est connu pour sa simplicité et sa facilité d'utilisation. Depuis qu'il est devenu disponible en 2011, il est devenu une plate-forme populaire pour créer des modèles d'impression 3D ainsi qu'une introduction pour les débutants à l'ingénierie solide constructive dans les écoles :[1]

- **Date de sortie** : 2011.
- **Propriétaire** : Autodesk.
- **Rédigé en** : Web, JavaScript.

2.8 Photo de simulation par (tinkercad)

Voila la photo de simulation de notre projet :

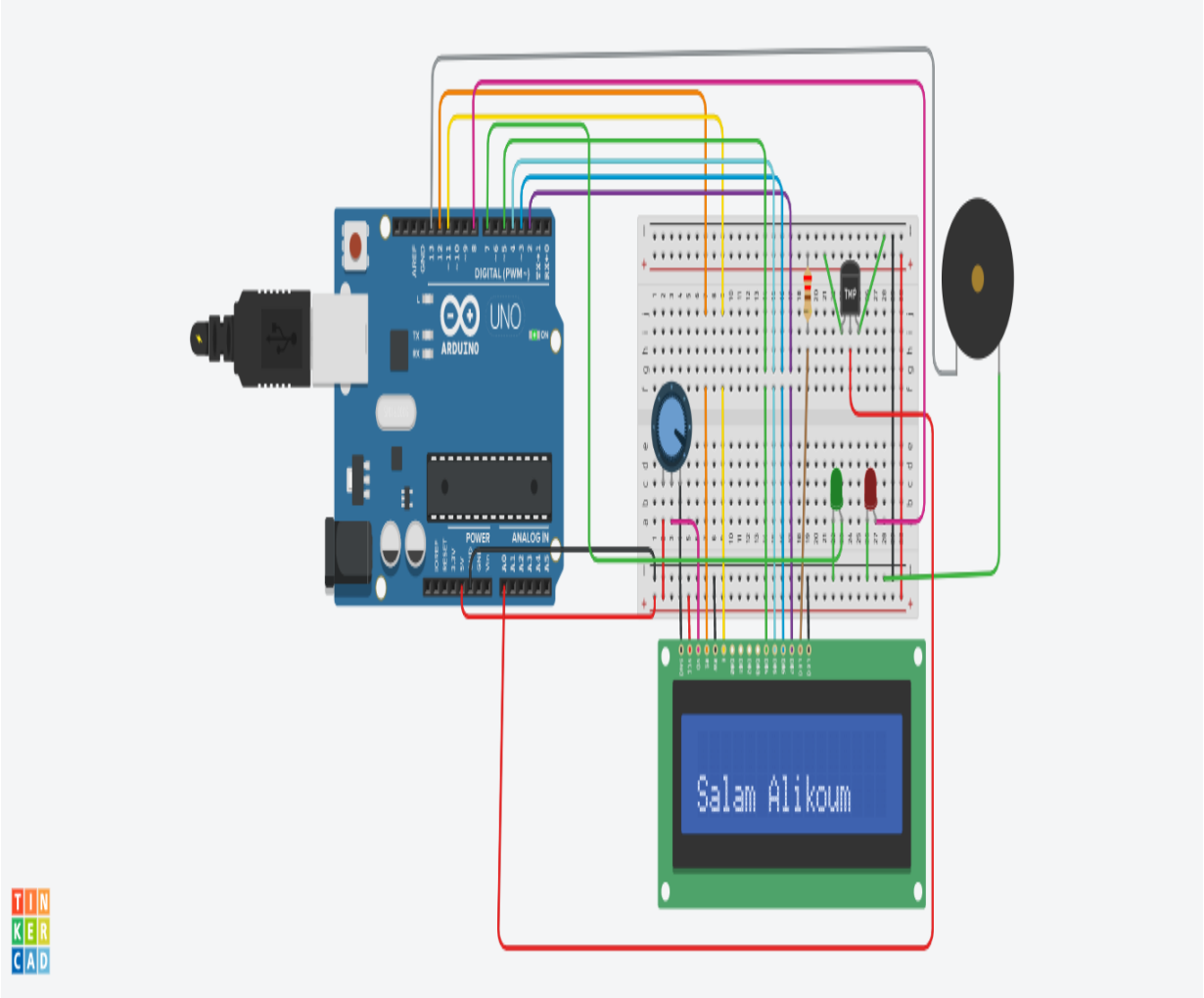

Figure 2.10: Simulation

2.9 Programme de commande

Voilà le programme de commande de notre projet :

sketch_may26a | Arduino 1.8.19

Fichier Édition Croquis Outils Aide

sketch_may26a §

```
int x=16;
int led1=8;
int led2=7;
int buzzer=13;
float temp;
#include <LiquidCrystal.h>

LiquidCrystal  lcd(12, 11, 5, 4, 3, 2);

void setup() {
//put your setup code here,to run cnce;


  lcd.begin(16,2);
  lcd.clear();
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(buzzer, OUTPUT);
  Serial.begin(9600);
}
void loop()
{
  lcd.setCursor(x,0);
  lcd.print("MCIL03 2022");
    x--;
  if (x<-44)
  {x=45;
delay(150);
  }
```

Figure 2.14 programme (partie 1)

sketch_may26a | Arduino 1.8.19

Fichier Édition Croquis Outils Aide

sketch_may26a §

```
delay(150);
  }

  lcd.setCursor(0,1);
  lcd.print("Salam Alikoum");
  delay (2000);
  lcd.clear();
  lcd.setCursor(0,1);
  temp=analogRead(A0);
  temp=temp*500/1023;
  lcd.print("Temperature:");
  lcd.print(temp);
  lcd.print("°C");
  lcd.println();
  delay(2000);
  lcd.clear();

  int Y= analogRead(A0);
  //Serial.println(Y);
 //delay(1000);

  if(Y > 50)
  {
  digitalWrite(led2,LOW);
  digitalWrite(led1,HIGH);
    digitalWrite(buzzer,HIGH);
  }
  else
  {digitalWrite(led2,HIGH);
```

Figure 2.15 programme (partie 2)

sketch_may26a | Arduino 1.8.19

Fichier Édition Croquis Outils Aide

Vérifier

sketch_may26a §

```
  delay (2000);
  lcd.clear();
  lcd.setCursor(0,1);
  temp=analogRead(A0);
  temp=temp*500/1023;
  lcd.print("Temperature:");
  lcd.print(temp);
  lcd.print("°C");
  lcd.println();
  delay(2000);
  lcd.clear();

  int Y= analogRead(A0);
  //Serial.println(Y);
 //delay(1000);

  if(Y > 50)
  {
  digitalWrite(led2,LOW);
  digitalWrite(led1,HIGH);
    digitalWrite(buzzer,HIGH);
  }
  else
  {digitalWrite(led2,HIGH);
   digitalWrite(led1,LOW);
    digitalWrite(buzzer,LOW);
  }
}
```

Figure 2.16 programme (partie 3)

2.10 Conclusion

La simulation de notre projet nous a permet des prédire la réponse de notre système. Aussi, les différents essais effectuées prouvent que l'opération d'affichage via un afficheur LCD 16*2 est fiable et efficace et peut être appliquée sur un écran de grande surface dans un lieu public.

Conclusion Générale

Conclusion Générale

Notre travail consistait à réaliser un gestionnaire d'un afficheur LCD par le moyen d'une carte de commande câblée et programmée dans le but est d'afficher des chaines de caractères saisies par le manipulateur ou des valeurs de grandeurs mesurées par des capteurs.

Notre mémoire comportait deux volets: le premier est consacré à l'identification et à la fragmentation de la carte de commande (Arduino Uno) et de la présentation des afficheurs LCD 16*2, le deuxième chapitre traite la partie simulation et l'énumération des différents éléments qui entrent dans l'élaboration de notre dispositif tels que le capteur de température, le buzzer, ...

Notre application à petite échelle sur un afficheur 16*2 peut être généraliser à grande échelle ou surface tel que les afficheurs publics, aussi, la maitrise de la gestion des entrées/sorties d'un afficheur facilite la communication des utilisateurs avec la machine.

Résumé

Notre projet vise à montrer et à commander des messages formées d'un ensemble de caractères sur un écran LCD grâce à une carte de commande : l'Arduino Uno programmée par une application IDE. Le manipulateur peut animer le message affiché, ainsi que la vitesse de translation. Des grandeurs physiques mesurées par des capteurs peuvent être affichées telle que la température dans notre cas.

الملخص

يهدف مشروعنا إلى إظهار الرسائل المكونة من مجموعة من الأحرف على شاشة LCD والتحكم فيها باستخدام بطاقة التحكم: Arduino Uno المبرمج بواسطة تطبيق IDE. يمكن لعصا التحكم تحريك الرسالة المعروضة، وكذلك سرعة السفر.

يمكن عرض الكميات المادية المقاسة بواسطة أجهزة الاستشعار مثل درجة الحرارة في حالتنا.