N° d'ordre……/FSI/UMBB/2018

République Algérienne Démocratique et Populaire

Ministère De l'enseignement Supérieur et de la Recherche Scientifique

Université M'hamed Bougara-Boumerdes

Faculté des sciences de l'Ingénieur

Département génie électrique

Mémoire de fin d'études

Présente pour l'obtention du diplôme de MASTER

Filière : Electronique

Spécialité : **E**lectronique **D**es **S**ystème **E**mbarqués

THEME

Etude, Simulation et Réalisation d'un système d'acquisition et de contrôle à base du microcontrôleur PIC16f877A

Réalisé par :

- ✓ **BOUINOUNE Saloua**
- ✓ **KARABADJI Ouarda**

Suivi par :

Mr. RAHMOUNE Fayçal

Soutenu le : 28/06/2018

Devant le jury composé de :

Mr. FELLAG Sid Ali	Prof/(UMBB)	President de jury
Mme. MAHDI Ismahane	MAA/(UMBB)	Examinatrice
Mr. AKROUM. H	MCB/(UMBB)	Examinateur
Mr. RAHMOUNE Fayçal	Prof/(UMBB)	Promoteur
Mr. DERGUINI Nour Eddin	ING/(CDTA)	Encadereur

Promotion: 2017/2018

REMERCIMENT

Louange à Dieu le clément, le miséricordieux, qui nous a donné le courage et la Patience de mener à bien ce travail.

Plus particulièrement, nous tenons à formuler à notre promoteur Mr ***Fayçal Rahmoune*** *un grand merci, pour nous avoir guidés dans notre travail ainsi que pour sa disponibilité, ses conseils et ses encouragements.*

Notre gratitude s'adresse particulier à notre encadreur Mr ***Nour eddin. DERGUINI*** *qui nous a mis dans les meilleures conditions de travail au centre de recherche CDTA tout au long de la période du stage.*

C'est ainsi nous exprimons notre gratitude et nos vifs remerciements à monsieur ***Loutfi houcini*** *pour ses orientations et ses conseils judicieux qui nous a trop aidé à maintenir le projet.*

Nous voudrions aussi exprimer l'honneur que nous font les membres du jury, d'avoir accepté d'examiner et d'évaluer notre travail.

Enfin, nous tenons à remercier nos chères familles qui nous ont encouragé et aider tout le long des années d'études.

Sommaire

Chapitre II Etude conceptuelle

Chapitre III Conception et Réalisation

Liste des figures

Chapitre I Mise en œuvre matérielle

Chapitre II Etude conceptuelle

Chapitre III Conception et Réalisation

Liste des tableaux

Chapitre I Mise en œuvre matérielle

Chapitre II Etude conceptuelle

Chapitre III Conception et Réalisation

INTRODUCTION GENERALE

Introduction générale

La température constitue une information importante dans plusieurs processus industriels et de laboratoire. Elle intervient comme une grandeur principale dont la valeur doit être connue avec précision ou comme paramètre influant sur la qualité d'autres mesures. Certain procèdes industriels nécessitent des environnements de températures spécifiques, ainsi la régulation de température s'impose. Cette régulation passe par la mesure de température de manière continue c'est le cas d'un capteur linéaire, à long terme il sera exposé à des contraintes des précision, dans ce cas nous allons utiliser une méthode d'étalonnage dans le but de le rendre plus précis et de diminuer ses erreurs absolues.

Dans ce travail, l'intérêt majeur est attribué à la recherche d'une solution permettant d'automatiser la commande de contrôle de température avec un circuit programmable à un prix réduit et il sera pour objectif d'être valable pour n'importe quel type de capteurs soient (pression, position, humidité…).

Notre système devrait être capable de lire une grandeur physique ou chimique via un capteur dédié pour une application qui sera choisie au préalable (i.e. Relai, Moteur ou Alarme).

Ce rapport contient trois chapitres répartis comme suit :

Le premier chapitre consiste à la présentation de quelques notions théoriques utiles sur les composants utilisés dans système de contrôle de température.

Le deuxième chapitre concerne la présentation de l'algorithme et l'organigramme du programme ainsi que l'outil de programmation.

Le dernier chapitre contient deux parties la première concerne la description des différentes fonctions du dispositif de contrôle de température avec un capteur linéaire d'une façon générale et consacré à la simulation, la conception et la réalisation.

La deuxième partie concerne la description des différentes fonctions du dispositif de capteur non linéaire y compris leur étalonnage pour objectif d'avoir un capteur linéaire

Enfin, une conclusion et des perspectives futures.

CHAPITRE I

Mise En Œuvre Matérielle

Introduction

Dans ce chapitre nous présenterons une description détaillée de la carte électronique utilisée dans notre système. D'abord nous commencerons par définir les contrôleurs de capteur et nous présenterons quelques domaines d'utilisations. Ensuite nous présenterons brièvement quelques notions théoriques utiles sur les composants que nous allons utiliser dans ce projet.

- Le PIC 16F877
- Le capteur lm35
- Le moteur pas à pas
- L'afficheur LCD16x2
- L'ULN2003
- Les LED

I.I Les microcontrôleurs

I.I.1 Présentation des microcontrôleurs

Les microcontrôleurs sont aujourd'hui à la portée des amateurs de la programmationdes circuits et permettent des réalisations aux possibilités étonnantes. Leurs utilisationspermettent de réaliser la conception de deux façons différentes :

- Considérer que ce sont des circuits « comme les autres », intégrés à certaines réalisations, et tout ignorer de leur fonctionnement Interne ;
- Profiter de leurs possibilités de programmation pour concevoir ses réalisations ou bien encore pour modifier le comportement d'appareils existants. Pour ce faire, il faut évidemment savoir les programmer.

Les microcontrôleurs les plus utilisés actuellement sont ceux du constructeur MICROCHIP [1] [2] [5].

I.I.2 Définition

Un microcontrôleur est un circuit qui intègre un maximum de fonctions sur le même boîtier. Il comprend une unité de traitement de type microprocesseur et des périphériques internes. Il permet la réalisation d'applications autonomes sans ajout de composants externes [2] [16] [8]

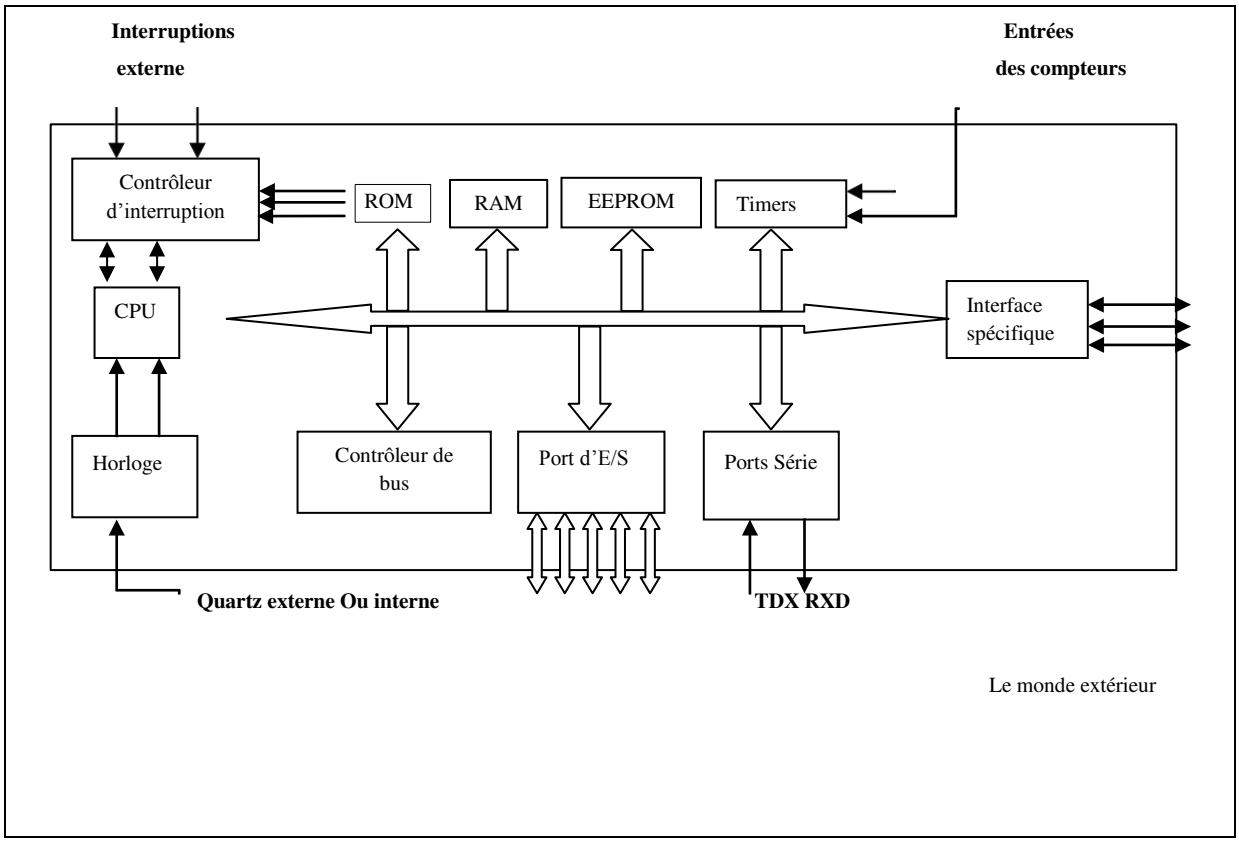


Figure I.1 : Architecture d'un microcontrôleur

I.I.3 Les avantages des microcontrôleurs

L'utilisation des microcontrôleurs pour les circuits programmables à plusieurs points forts et bien réels :

- Un plus haut degré d'intégration
- Encombrement moindre
- Le microcontrôleur contribue à réduire les coûts à plusieurs niveaux
- L'augmentation de la fiabilité du système [3] [5].

I.I.4 Choix du microcontrôleur

Le choix d'un microcontrôleur pour une application dépend principalement :

- Du nombre d'entrées/sorties de l'application
- Du type de mémoire programme : flash, EPROM et de sa taille
- De la présence ou non des convertisseurs Analogique/Numérique
- De l'existence ou non d'une mémoire EEPROM.
- Du coût et de la disponibilité sur le marché [3] [4] [5].

I.I.5 Les microcontrôleurs PIC

Les PIC sont des composants dit RISC (Reduced Instruction Construction Set), ou encore composants à jeu d'instruction réduit, car plus on réduit le nombre d'instructions, plus facile et plus rapide est le décodage, et plus vite le composant fonctionne [3] .

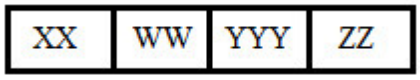
I.I.6 Différentes familles des PIC

La famille des PIC est subdivisée à l'heure actuelle en 3 grandes catégories :

- **Base-Ligne (Base-Line) :** ils utilisent des mots d'instruction de 12 bits, tel que 12F.
- **Milieu de gamme (Mid-Range) :** ils utilisent des mots d'instruction de 14 bits, tel que 16F.
- **Haut de gamme (High-End) :** ils utilisent des mots d'instruction de 16 bits. Tel que 18F. [3][5][17]

I.I.7 Identification d'un PIC

Pour identifier un PIC, on utilise simplement son appellation du type :



- **WW:** Représente la catégorie du composant (12, 14, 16, 17, 18),
- **XX:** Type de mémoire de programme:
  - **C:** EPROM ou EEPROM.
  - **CR:** PROM.
  - **F:** FLASH.
  - **YYY:** Identification.
  - **ZZ:** fréquence d'horloge en MHz.

Exemple : 16F877-20

16 : indique la famille mid-range

F : mémoire utilisé de type Flash

877 : identité **:** fréquence d'horloge

20 : fréquence d'horloge. [3] [4]

I.II Le PIC16F877

I.II.1 Architecture interne du PIC16F877

Le PIC 16F877 est caractérisé par :

- Une fréquence de fonctionnement élevée, jusqu'à 20MHz.
- Une mémoire vive RAM de 368 octets.
- Une mémoire morte EEPROM de 256 octets pour la sauvegarde des données.
- Une mémoire de type FLASH de 8 K mots (1mot = 14 bits)
- Chien de garde WDT (Watchdog Timer).
- 33lignes d'entrées /sorties. Chaque sortie peut générer un courant maximum de 25 mA.
- Trois Temporisateurs :
  - ✓ TIMER0 : compteur de taille de 8 bits avec pré-diviseur de valeurs de (0-255).
  - ✓ TIMER1 : compteur de taille de 16 bits avec pré-diviseur de valeurs de (0-65535).
  - ✓ TIMER2 : compteur de taille de 8 bits avec pré-diviseur de valeurs de (0-255).
- Deux entrées de captures et de comparaison.
- Un convertisseur Analogique / Numérique 10 bits avec 8 entrées multiplexées.
- Une interface de communication série asynchrone et synchrone (USART/SPI).
- Une tension d'alimentation entre 2 et 5.5 V.

Le schéma ci-dessous représente l'architecture interne d'un microcontrôleur 16F877 [4] [5] [8] .

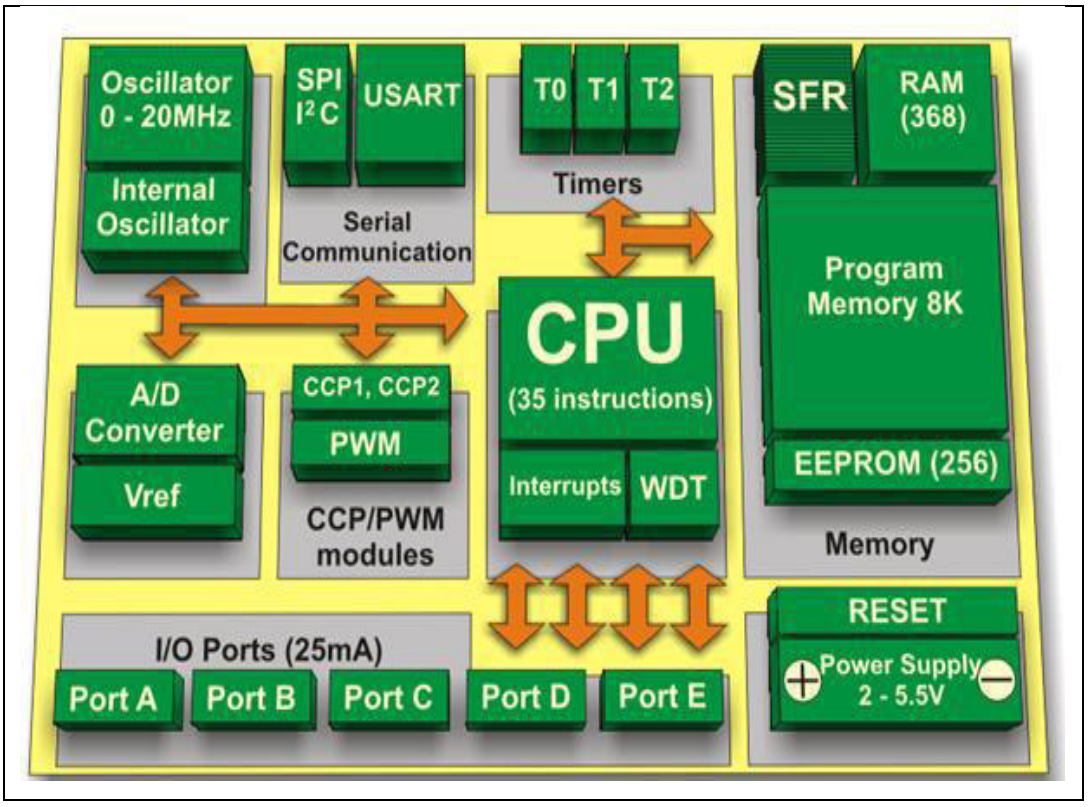


Figure I.2 : Architecture interne d'un microcontrôleur 16F877

I.II.2 Organisation de la mémoire

- **Mémoire FLASH :** C'est dans celle-ci qu'est stocké le programme du PIC.
- **Mémoire RAM :** Fait partie de la zone d'adressage des données.
- **Mémoire EEPROM :** L'EEPROM est une mémoire de stockage de données **[3] [5] [16].**

I.II.3 Architecture externe

Le boîtier du PIC 16F877 décrit par la figure ci-dessous comprend :

40 pins : 33 pins d'entrées/sorties,

4 pins pour l'alimentation,

2 pins pour l'oscillateur

1 pin pour le reste (MCLR) [5] [17].

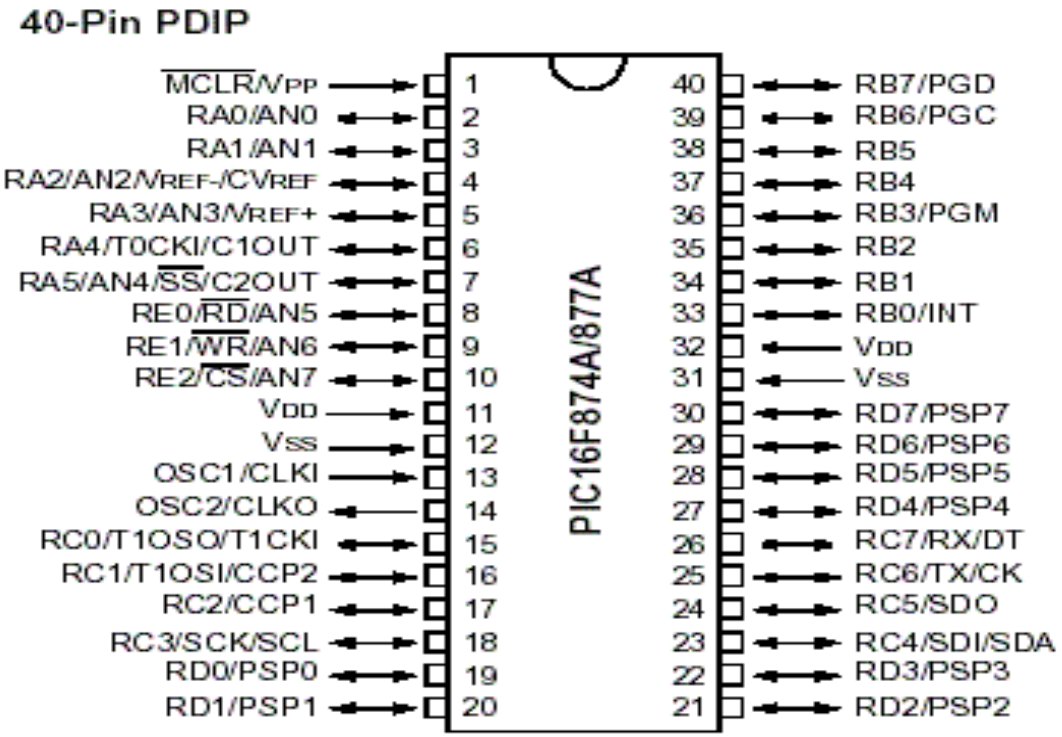


Figure I. 3 : Architecture externe d'un PIC16F877A [8]

I.II.4 Les différents ports d'entrées/sorties du PIC 16F877

Le microcontrôleur 16F877 dispose 5 ports (A, B, C, D, E). Tous les ports d'entrées/sorties sont bidirectionnels. Ils sont pilotés par deux registres :

Le registre de PORTx : Si le PORTx ou certaines lignes de PORTx sont configurées en sortie, ce registre détermine l'état logique des sorties.

Le registre TRISx : C'est le registre de direction. Il détermine si le PORTx ou certaines lignes de PORTx sont en entrée ou en sortie.

L'écriture d'une 1 logique correspond à une entrée et l'écriture d'une 0 logique correspond à une sortie. Au RESET toutes les lignes de ports sont configurées en entrées.

Port A : C'est un port d'entrée sortie, il contient 6pins d'entrées /sorties de RA0 à RA5 repartie sur deux registres : le registre PORTA et le registre TRISA. Le bit 6 et7 ne sont pas implémenté, ils seront lus comme 0.

Au moment de reset on doit forcer une valeur dans le registre ADCON1, pour pouvoir utiliser ce port en entrée/sortie de type générale.

Port B : Le port B est formé de huit pins d'entrées/sorties numérotées de RB0 à RB7. Il peut être configuré pour générer une interruption sur un changement d'état des broches RB4 à RB7.

Port C : Le port C possède huit pins entrées/sorties numérotées de RC0 à RC7.

Port D : Ce port n'est présent que sur le 16F877.il fonctionne d'une façon identique aux autres, dans son mode de fonctionnement général.

Port E : Ce port n'est présent que sur les PIC 16F877.

Il ne comporte que 3 pins, RE0, RE1 et RE2, les bits non concernés de TRIS sont implémentés pour d'autres fonctions.

Les 3 bits de PORTE peuvent être utilisés soit comme E/S numérique soit comme entrées analogiques du CAN. La configuration se fait à l'aide du registre ADCON1. [4] [5]

I.II.5 Les particularités électriques

Les broches VDD (Broche 11 et 32) et VSS (Broche 12 et 31) : Servent à alimenter le PIC.Ils sont placés d'une part et d'autre en position centrale du PIC.

La broche MCLR : Est connecter au +5v, il sert à initialiser le pic qui dispose de plusieurs sources de RESET.

Le quartz : Est peut être remplacé par un oscillateur ou par un simple réseau RC. Les condensateurs de découplage, du fait de la fréquence plus importante du quartz utilisé, sont de valeur environ 15pF. [4] [5] [16]

I.III Les capteurs

I.III.2 Définition

Par définition, un capteur est un dispositif électronique capable de transformer une grandeur physique, chimique, biologique… (mesurande) en une grandeur électrique, généralement une tension ou un courant (la figure I.4 présente le principe d'un capteur). [9]

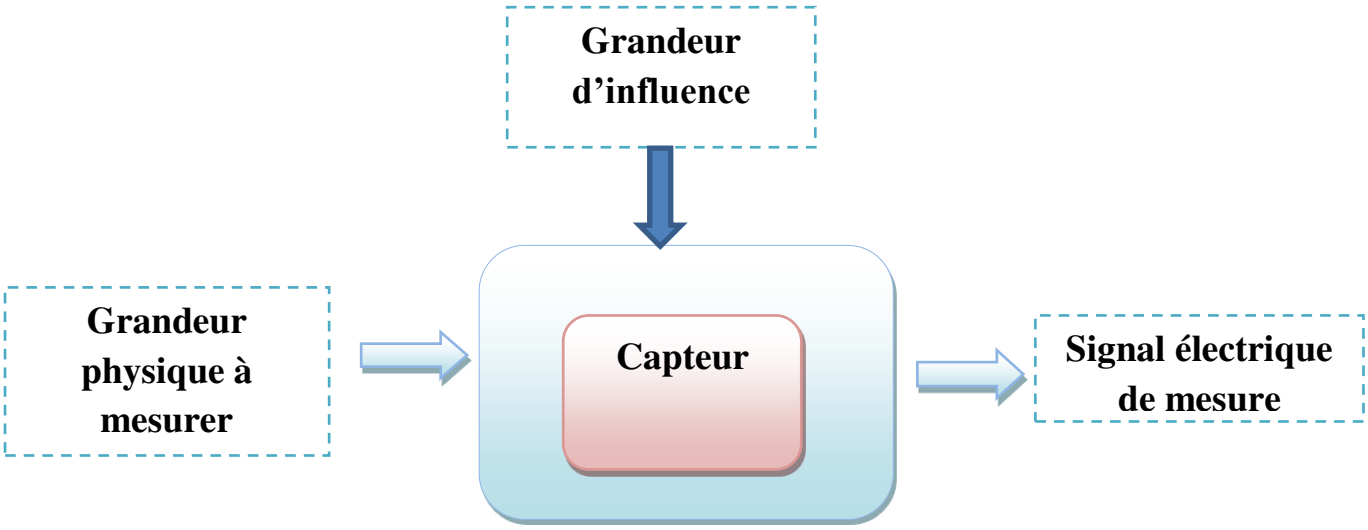


Figure I.4 : Principe d'un capteur. [9]

I.III.3 Structure d'un capteur

Les différentes parties constitutives d'un capteur sont décrites ci-dessous (figure I.5) : [9]

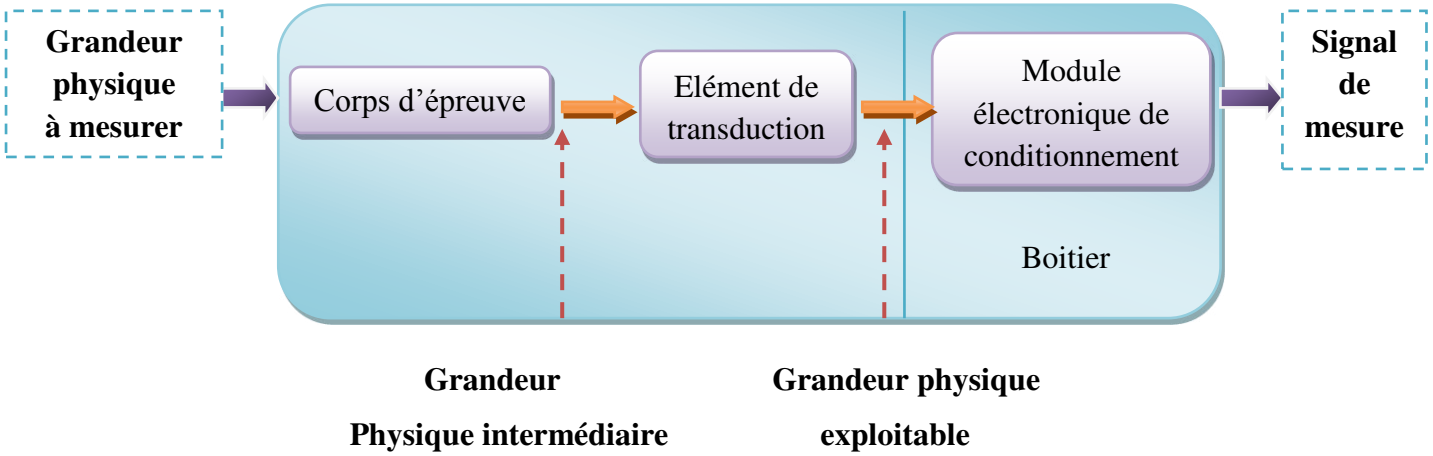


Figure I.5 : Structure d'un capteur [9]

A. Le corps d'épreuve

Le corps d'épreuve est un élément sensible qui réagit à la grandeur à mesurer. Il a pour rôle de transformer la grandeur à mesurer en une autre grandeur physique dite mesurable. [9] [8]

B. Transducteur

Le transducteur est un élément sensible lié au corps d'épreuve. Il traduit les réactions du corps d'épreuve en une grandeur électrique constituant le signal de sortie. [9] [10]

C. Le boîtier

Le boîtier est un élément mécanique de protection, de maintien et de fixation du capteur. [9]

D. L'électronique de conditionnement

C'est un dispositif qui convertit le signal de sortie du capteur en un signal de mesure standard. Il fait le lien entre le capteur et le système de contrôle commande. Il a pour rôle d'amplifier et de faire le traitement du signal électrique. [10] [8]

I.III.4 Différents types d'un capteur [9]

Suivant l'origine du signal électrique de sortie, on peut classer les capteurs en deux types :

A. Capteurs actifs

Ce type de capteurs fonctionne en générateur, dont une partie de l'énergie physique prélevée sur la mesurande est transformée directement en énergie électrique qui constitue le signal de sortie (tension ou courant). Les principes physiques mis en jeu sont présentés ci-dessous (tableau I.1). [3]

Grandeur physique à mesurer	Effet utilisé	Grandeur de sortie
Température	Thermoélectricité	Tension
Température	Pyroélectrique	Charge
Flux de rayonnement optique	Photoémission	Courant
Flux de rayonnement optique	Effet photovoltaïque	Tension
Flux de rayonnement optique	Effet photoélectrique	Tension
Force ou pression	Piézoélectrique	Charge
Accélération ou vitesse	Induction électromagnétique	Tension
Position (aiment) ou courant	Effet hall	Tension

Tableau I.1 : Exemples de capteurs actifs [9]

B. Capteurs passifs

Ils sont constitués d'un matériau spécifique dont l'impédance (résistance, capacité ou inductance) est sensible au mesurande. Le tableau I.2 résume, en fonction du mesurande, les matériaux et les effets utilisés pour réaliser la mesure. [9]

Grandeur physique à mesurer	Caractéristique sensible	Matériaux utilisé
Température	Résistivité	Métaux : platine,nickel,cuivre…
Très basse température	Constante diélectrique	Verre
Flux de rayonnement optique	Résistivité	Semi-conducteur

Déformation	Résistivité	Alliage de Nickel, silicium dopé
Déformation	Perméabilité magnétique	Alliage ferromagnétique
Position	Résistivité	Matériaux magnéto résistant
Humidité	Résistivité	Chlorure de lithium

Tableau I.2 : Exemples de capteurs passifs. [9]

I.III.5 Caractéristiques métrologiques d'un capteur

Les caractéristiques métrologiques d'un capteur constituent les liens effectifs entre le capteur et la grandeur qu'il mesure. [9]

I.III.5.1 Etalonnage

L'étalonnage permet d'ajuster et de déterminer, sous forme graphique, la relation entre le mesurande et la grandeur électrique de sortie (figure I.6). Très souvent l'étalonnage n'est valable que pour une seule situation d'utilisation du capteur. [9][8]

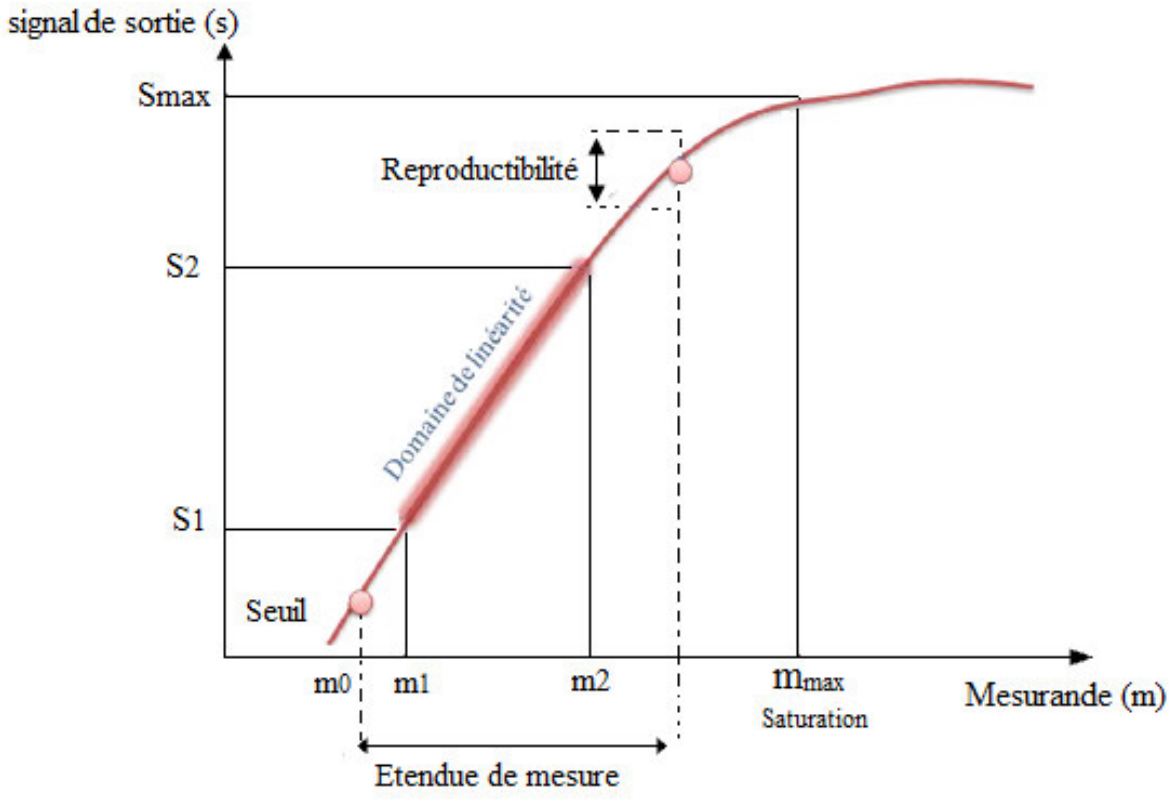


Figure I.6 : Courbe d'étalonnage d'un capteur [9]

I.III.5.2 Étendue de mesure

L'étendue de mesure est définie sur la courbe d'étalonnage du capteur (figure I.6). A l'extérieur de cette zone se trouvent deux valeurs particulières : le seuil et la saturation.
Le phénomène de saturation est fréquemment rencontré en physique. Même si la valeur du mesurande augmente, la grandeur de sortie ne peut dépasser une valeur maximale Smax : pour m>mmax, S=Smax. On ne peut donc pas effectuer de mesurage pour des valeurs au-dessus de mmax.

Le seuil ou limite de détection correspond à la valeur minimale du mesurande nécessaire pour obtenir une grandeur de sortie non nulle : pour m=m0, S correspond au bruit de fond de la mesure.

$$S=LOD=3.3* SD / s \qquad \text{Eq.I. 1}$$

Avec **SD** : l'écart type de la réponse.

S : la pente.

En résumé, on ne peut mesurer que des mesurandes compris entre m0 et mmax. [12]

I.III.5.3 Domaine de linéarité

Dans ce domaine de linéarité, la variation de la grandeur de sortie est proportionnelle à la variation du mesurande. [9][11]

I.III.5.4 Sensibilité

La sensibilité (***s***) est une caractéristique importante pour l'exploitation et l'interprétation des mesures. Elle est définie comme étant la variation du signal de sortie (ΔS) par rapport à la variation du mesurande (Δm) (pente de la portion linéaire de la courbe d'étalonnage) et s'écrit : [9]

$$S=\frac{\Delta S}{\Delta m} \qquad \text{Eq I. 2}$$

I.III.5.5 Temps de réponse

La rapidité est caractérisée par le temps que met le capteur à réagir à une variation brusque du mesurande. Cependant la valeur finale étant le plus souvent atteinte de manière

asymptotique, elle correspond au temps nécessaire pour que le capteur délivre une certaine portion α de la pleine amplitude du signal. Le temps de réponse noté tα est tel que α vaut généralement 90%.

La connaissance du temps de réponse d'un capteur est un élément essentiel lors de la réalisation de mesures.[9]

I.III.5.6 Résolution - Précision

C'est la plus petite variation de mesurande que peut détecter le capteur. [10]

I.III.5.7 Reproductibilité ou répétabilité

Ce paramètre est probablement le plus important, tant pour les capteurs physiques que chimiques. C'est l'aptitude d'un capteur à donner, dans des conditions définies, des réponses très voisines lors de la mesure répétée d'une même valeur du mesurande. Pour une fabrication de capteurs, on définit la reproductibilité d'un capteur à l'autre.

Toute mesure sera entachée d'une erreur qu'il est important de connaître.

- L'erreur systématique. Elle est toujours dans le même sens et de la même amplitude. On peut la détecter en effectuant des mesures avec un autre appareillage.
- L'erreur accidentelle possède une amplitude et un sens aléatoires. Ses causes peuvent être variées. [10]

I.III.5.8 Sélectivité

Un capteur est dit sélectif, si la variation du signal de sortie est due uniquement à la seule grandeur (physique, chimique, biologique…) qu'on veut mesurer. [8][12]

I.III.5.9 Grandeurs d'influence

Les grandeurs d'influence sont les paramètres qui influent sur le signal de sortie du capteur. On retrouve les grandeurs de type mécanique (variations de pression, les forces qui provoquent des déformations...) ou thermique (variation de température qui engendre la dilatation des corps et la modification des propriétés électriques tels que le changement de conductibilité et de caractéristiques diélectriques) mais aussi des grandeurs électriques (paramètres électriques, tels que courant, tension, fréquence, des circuits d'alimentation du capteur).

Dans le cas des capteurs chimiques et des biocapteurs, la présence d'espèces différentes de l'espèce cible peuvent influer sur le signal de sortie du capteur. [10] [11]

I.III.6 Capteur lm35

I.III.6.1 Définition

Le capteur LM35 est un capteur de température où la tension de sortie est linéairement proportionnelle à la température en Celsius centigrade. Ce capteur ne nécessite pas de calibrage externe pour fournir une précision de ± _°C sur une gamme de température de –55°C à +150°C. Son coefficient est de 10mV/°C et dans notre cas le capteur est alimenté en 0-5V, on ne peut mesurer par conséquent que des températures positives. [10]

I.III.6.2 Branchage de lm35

On utilise le capteur LM335 pour détecter la température, ce capteur fonctionne à partir de -40 ° C à 100 ° C et alimenté par 5 v. [10] [11]

Figure I.7 : LM335 [10]

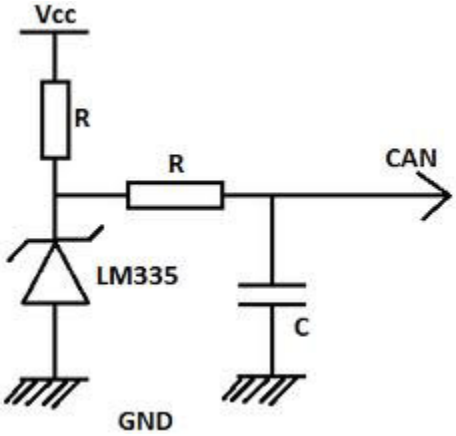


Figure I.8 : Circuit équivalent de LM335 [10]

I.III.6.3 Caractéristiques de lm35

- Fonctionne à partir du 400 µA à 5 mA
- Impédance dynamique réduite
- Facilement calibré
- Grand choix de température de fonctionnement
- À prix réduit [11]

I.III.6.4 Avantage du capteur LM35

Parmi les avantages de ce capteur, mentionnons sa consommation très faible (de l'ordre de 60 µA), d'où une puissance dissipée également très faible, et sa linéarité qui demeure excellente sur toute sa plage de sensibilité. Cette plage va de -40°C à +110°C pour les LM35C et de 0°C à 100°C pour les LM35D [9] [11].

I.III.6.5 Différentes versions de Lm 35

Voici le brochage des différentes versions du LM35, disponible sous 3 boîtiers différents. Les références CZ et DZ se présentent en boîtier TO-92 : on s'orientera vers l'une ou l'autre [11].

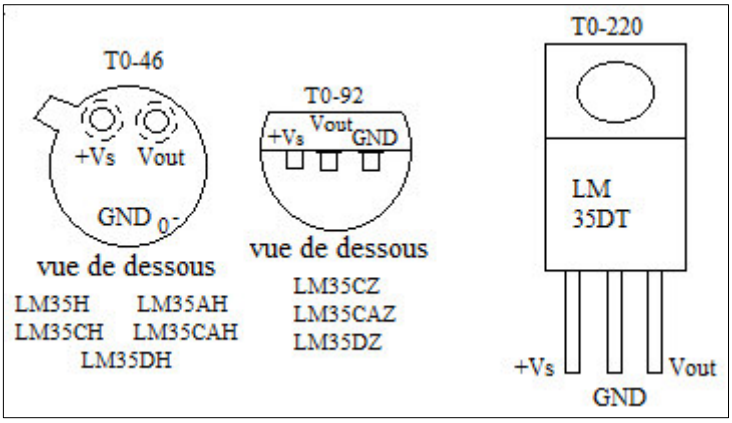


Figure I.9 Les différentes versions du lm 35 [11]

I.III.6.6 le Circuit Intégré de Lm35

Brochage de la puce

1 : Alimentation du capteur (+5 V).

2 : Tension de sortie Vs. [11]

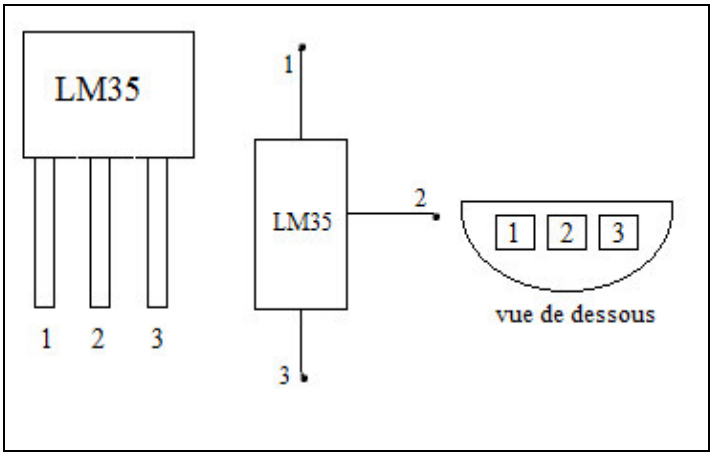


Figure I.10 le circuit intégré du lm35 [11]

I.IV L'actionneur

I.IV.1 Définition

Convertisseur électromécanique (énergie électrique en énergie mécanique) conçu pour mettre en mouvement linéaire des systèmes mécaniques à partir de commandes électriques. [12]

I.IV.2 Sens mécanique

Partie d'une machine ou d'un système de commande à distance qui permet de convertir l'énergie reçue travail utile pour exécuter les taches d'un système automatisé. [12]

I.IV.3 Critères de base de conception d'un actionneur

- Facilité et précision avec laquelle on peut commander électriquement la force ou le couple
- Rapport existant entre la taille de l'actionneur et la force ou le couple qu'il peut développer : plus le couple développé élevé pour une taille donnée, meilleur est l'actionneur. [12]

I. IV.4 Moteur pas à pas

I. IV.4.1 Définition

Le moteur pas à pas fut inventé par Marius Lavet en 1936 pour l'industrie horlogère. Il permet de transformer une impulsion électrique en un mouvement angulaire. Ce type de moteur est très courant dans tous les dispositifs où l'on souhaite faire du contrôle de vitesse ou de position en boucle ouverte, typiquement dans les systèmes de positionnement. L'usage le plus connu du grand public est dans les imprimantes reliées à un ordinateur (positionnement de la tête d'impression et rotation du rouleau porte-papier dans les imprimantes matricielles, à marguerite et à jet d'encre, et rotation du rouleau porte-papier seulement dans les imprimantes à xérographie à laser). [12]

Figure I.11 le moteur pas à pas

I. IV.4.2 Principe de fonctionnement d'un moteur pas à pas

Le fonctionnement d'un moteur pas à pas nécessite la présence des éléments suivants :

- Une unité de commande (microcontrôleur par exemple) qui fournit des impulsions dont la fréquence est proportionnelle à la vitesse de rotation du moteur, elle imposera également le sens de rotation.
- Un séquenceur qui aiguillera les impulsions sur les différentes bobines du moteur.
- Une alimentation de puissance. [12]

I. IV.4.3 Caractéristiques des moteurs pas à pas

- **Nombre de pas par tour**

Plus il est élevé, plus la précision obtenue sera grande.

Exemple : moteur de 200 pas/tour, à chaque pas, le moteur tournera de 360°/200=1,8°. Généralement, les moteurs pas à pas ont des pas variants de 15° (24 pas par tour) à 0,9° (400 pas par tour).[11][12]

- **Tension d'alimentation**

Trois volts à quelques dizaines de volts. Suivant la résistance ohmique des bobinages, la consommation des moteurs pas à pas peut atteindre plusieurs ampères. [12]

- **Couples du moteur**
  - Couple dynamique : c'est le couple disponible sur l'arbre lorsque le moteur est en marche
  - Couple de détente : c'est le couple obtenu lorsque le moteur à aimant permanent ou hybride est hors tension
  - Couple de maintien : couple auquel peut résister un moteur à l'arrêt, ses enroulements correspondants restant alimentés de façon permanente. [12]

I. IV.4.4 Avantages

- Rotation constante pour chaque commande (précision meilleure que 5% d'un pas).
- Existence de couple à l'arrêt.
- Contrôle de la position, de la vitesse et synchronisation de plusieurs moteurs (pas de besoin de contre-réaction). [12]

I. IV.4.5 Inconvénients

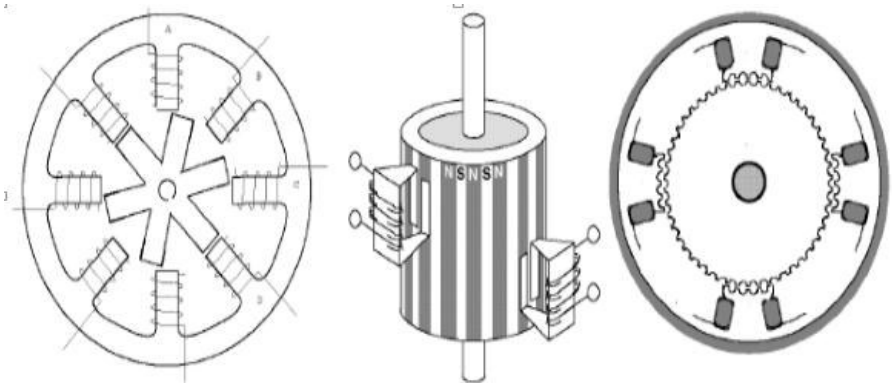
- Plus difficile à faire fonctionner qu'un moteur à courant continu.
- Vitesse et couple relativement faible.
- Couple décroissant rapidement lorsque la vitesse augmente. [12]

I. IV.4.6 Les différents types de moteurs pas à pas

Trois catégories de moteurs :

- **A réluctance variable** : On ne sent pas les pas, a caractéristique électrique identique, un tel moteur est moins puissant, mais plus rapide que les moteurs à aimant permanent. Sans doute les plus anciens [12].
- **A aimants permanents** : Rotor fabriqué en acier doux non magnétique dans lequel sont taillées des dents.
  On sent les pas, ce sont des moteurs à faible coût de revient, et de résolution moyenne (jusqu' à 100 pas/tour) [12].
- **Hybrides** : Ces moteurs combinent les 2 technologies précédentes, et sont plus chers. Leur intérêt réside dans un meilleur couple, une vitesse plus élevée, et une résolution de 100 à 400 pas/tour.

Les moteurs les plus courants sont ceux à aimants permanents et les hybrides [12] [21].



Reluctance variable Aimants permanents Hybride

Figure I.12 Les différents types de Pas à pas

Type de moteurs	Aimant permanent	Reluctance variable	Hybride
Nombre de pas par tour	Moyen	Assez important	Elevé
Couple moteur	Elevé	Faible	Elevé
Sens de rotation	Dépend du sens du courant et de l'ordre d'alimentation des bobines	Dépend seulement de l'ordre d'alimentation des bobines	Dépend du sens du courant et de l'ordre d'alimentation des bobines
Vitesse de rotation	Faible	Grande	Grande

Tableau I.3 : caractéristiques des types de moteur pas à pas

Parmi les moteurs à aimants permanents et hybrides, on distingue les moteurs unipolaires et bipolaires [11]

I. IV.4.6.1 Les bipolaires

– 4 fils, avec 2 bobines indépendantes (appelé « 2 phases »).

I. IV.4.6.2 Les unipolaires

– 5 fils, 1 commun aux 4 bobines, plus un par bobine.

– 6 fils, 2 bobines avec point milieu (parfois appelé « 2 phases »).

– 8 fils, 4 bobines indépendantes, également appelé universel.

– Par abus de langage, ces moteurs sont tous appelés « 4 phases ».

Ce sont les moteurs pour GP99 et BB00.

• Et les moins classiques

– 4 ou 6 fils avec un point commun à toutes les bobines. Ce sont des moteurs unipolaires avec 3 ou 5 bobines. Ces moteurs sont appelés 3 ou 5 phases. [12]

I. IV.4.6.3 Comparaison entre le moteur bipolaire et unipolaire

- **Bipolaire**

  Puissance disponible plus élevée pour à caractéristiques mécaniques identiques. [12]

- **Unipolaire**

  Les moins chers et sont plus facile à mettre en œuvre. C'était surtout vrai avant l'arrivée de circuits intégrés spécialisés. [12]

I. IV.4.7 Commandes des moteurs pas à pas

I. IV.4.7.1 cas de pas entier monophasé et biphasé

	Phase 1	Phase 2	Phase 3	Phase 4
Pas 1	1	0	0	0
Pas 2	0	1	0	0
Pas 3	0	0	1	0
Pas 4	0	0	0	1

Tableau I.4 Commande pas entier monophasé [12]

	Phase 1	Phase 2	Phase 3	Phase 4
Pas 1	1	0	0	0
Pas 2	1	1	0	0
Pas 3	0	1	1	0
Pas 4	0	0	1	1

Tableau I.5 Commande pas entier biphasé [12]

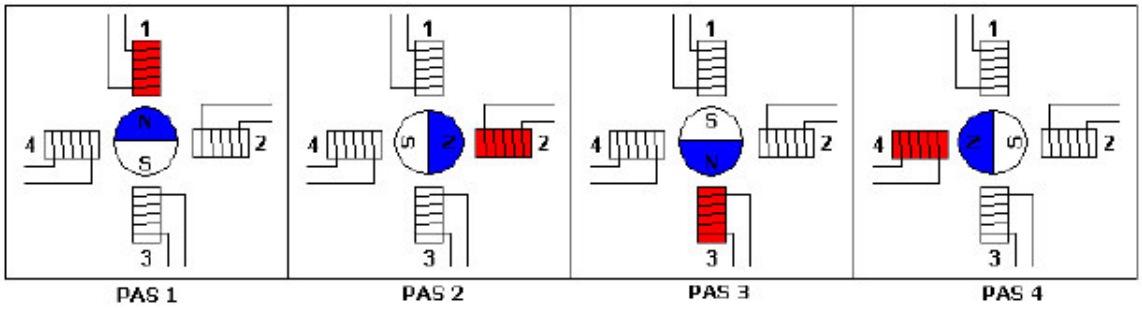


Figure I.13 Commande pas entier monophasé [12]

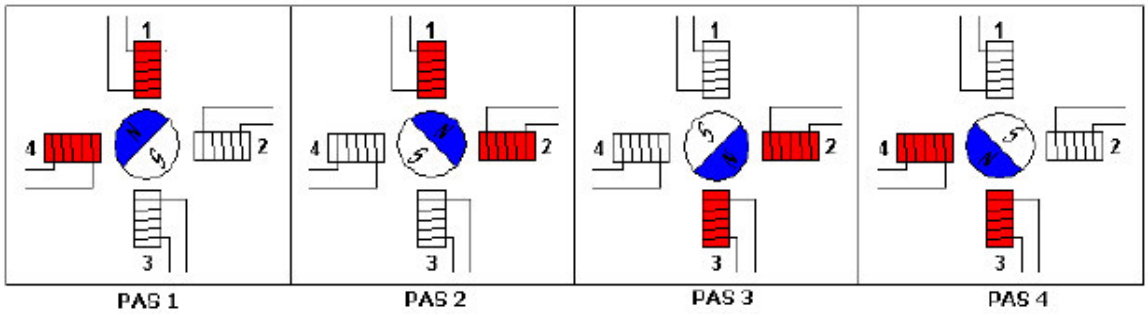


Figure I.14 Commande pas entier biphasé [12]

I. IV.4.7.2 cas de demi pas

	Phase 1	Phase 2	Phase 3	Phase 4
Demi pas 1	1	0	0	0
Demi pas 2	1	1	0	0
Demi pas 3	0	1	0	0
Demi pas 4	0	1	1	0
Demi pas 5	0	0	1	0
Demi pas 6	0	0	1	1
Demi pas 7	0	0	0	1
Demi pas 8	1	0	0	1

Tableau I.6 Commande de moteur avec un cas de demi pas [12]

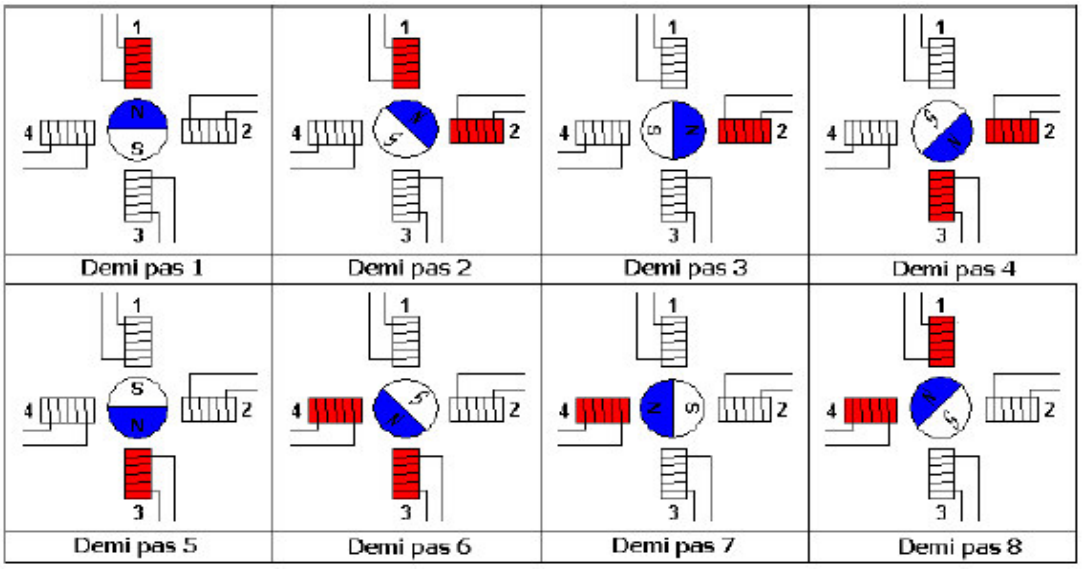


Figure I.15 Commande de moteur avec un cas de demi pas [12]

I. IV.4.8 Alimentation des moteurs pas à pas

– Un moteur Pap alimenté à son courant nominal peut chauffer de
60 à 70 degrés par rapport à la température ambiante. Attention aux brûlures !
– Un moteur qui chauffe plus à l'arrêt qu'en fonctionnement est sous-alimenté (cas typique du moteur alimenté en tension, dès que la vitesse de rotation augmente). Ce n'est un problème que si la puissance disponible est insuffisante, il faut alors passer à un mode d'alimentation « en courant ». [12]

- Calcul de la résistance pour une alimentation en courant contrôlé par résistance série (choisir une résistance non inductive) :

• R = (VAlim - VMoteur) /IMoteur
• Puissance dissipée P = (VAlim - VMoteur) * Imoteur
Ex: un moteur prévu pour 5V, 200mA, alimenté sous 15V nécessite une résistance de
(15 - 5)/0,2 = 50 Ohms, (15-5)*0,2 = 2Watts.

I. IV.5 Moteur pas à pas « 55SI-25 DAWC »

Un moteur pas à pas est un moteur dont la rotation du rotor s'effectue par déplacements angulaires successifs sous l'action d'impulsions électriques appliquées sur les bobinages statiques. Malgré les différences existantes entre les moteurs pas à pas, le résultat recherché est l'avance d'un seul pas, c'est-à-dire la rotation de leur axe suivant un angle déterminé à chaque impulsion que l'une ou l'autres bobines différentes recevra. Cet angle, qui varie selon la constitution interne du moteur, est en général compris entre 0.9° et 90°. [6][7]

Le moteur 55SI-25DAWC (Figure I.16), connu aussi sous le nom de 55SI-25DAYA, est un moteur pas à pas unipolaire très connu et très facile a utilisé pour des petits automatismes ou des réalisations à commande numérique.

Figure I.16 Moteur 55SI-25DAWC [11]

I.IV.5.1 Caractéristiques générales du 55SI-25 DAWC

- Tension nominale : 12 Volts.
- Résistance de bobine : 36 ohms.
- Degré d'angle : 7.5°.
- Diamètre moteur : 55mm
- Courant nominale : 330 mA.
- Inductance : 37mH.
- Nombre de pas par tour : 48.
- Hauteur moteur hors axe et palier : 25m m

[6][7]

I. IV.5.2 Câblage et code couleur

Le moteur unipolaire possède 5,6 ou 8 fils, Il a toujours un, deux ou quatre fils communs selon le nombre de fils que contient le moteur, il est parfois facile de distinguer ces fils, par leur ressemblance de couleur [7]. Sur le (**Tableau I.7)** ci-dessous le moteur

55SI-25DAWC contient six fils, la reconnaissance de ces derniers nécessite un multimètre afin d'établir le tableau suivant :

	Blanc 1	Blanc 2	Brun	Jaune	Bleu	Rouge
Blanc 1	X	inf	36Ω	inf	inf	36Ω
Blanc 2	inf	X	inf	36Ω	36Ω	inf
Brun	36Ω	inf	X	inf	inf	72Ω
Jaune	inf	36Ω	inf	X	72Ω	inf
Bleu	inf	36Ω	inf	72Ω	X	inf
Rouge	36Ω	inf	72Ω	inf	inf	X

Tableau I.7 Câblage et code couleur [11]

Le symbole X signifie que c'est le même fil, pour le symbole inf il signifie que c'est une résistance infinie, c'est-à-dire, que les deux fils ne sont pas connectés. Evidemment, c'est un moteur a quatre bobines, qui ont chacun un connecteur commun blanc, le tableau ci-dessus nous montre que le rouge et le brun sont les bobines connectées avec blanc 1 et que le jaune et le bleu sont la paire avec blanc 2. [11]

A partir de ce tableau, on peut mettre un schéma, représenté par la (Figure I.17), qui nous aidera à piloter notre moteur.

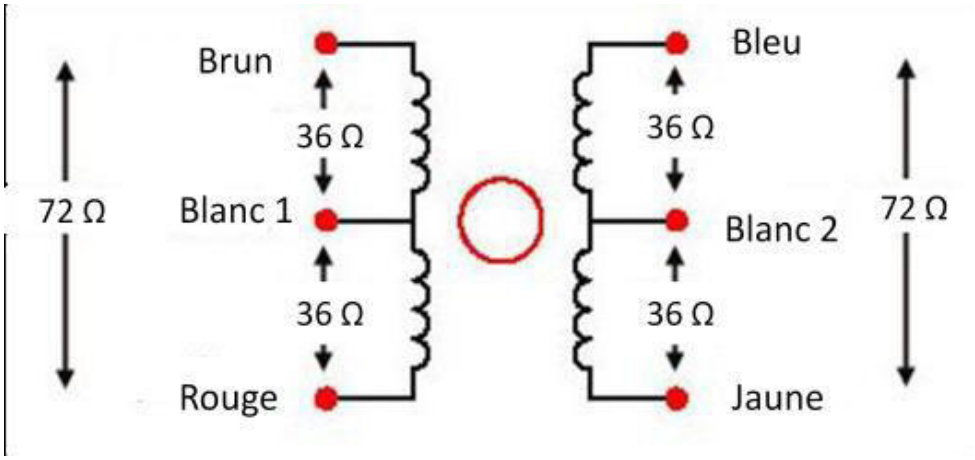


Figure I.17 Schéma du moteur 55SI-25DAWC [6]

I. IV.5.3 Mode de commande

La rotation du moteur s'effectue par une séquence d'alimentation des divers enroulements de phases en unipolaire ou en bipolaire. Cependant, avec un pas unipolaire les bobines doivent être entraînées dans une certaine séquence, sinon le moteur ne tourne pas du tout, ou n'aura que peu de pouvoir. Il existe trois méthodes distinctes pour piloter les moteurs :

- Commande par pas entier, une phase ON (dite mode monophasé)
- Commande symétrique en pas entier, deux phase ON (dite mode biphasé)
- Commande asymétrique en demi pas, une ou deux phases ON [6] [7]

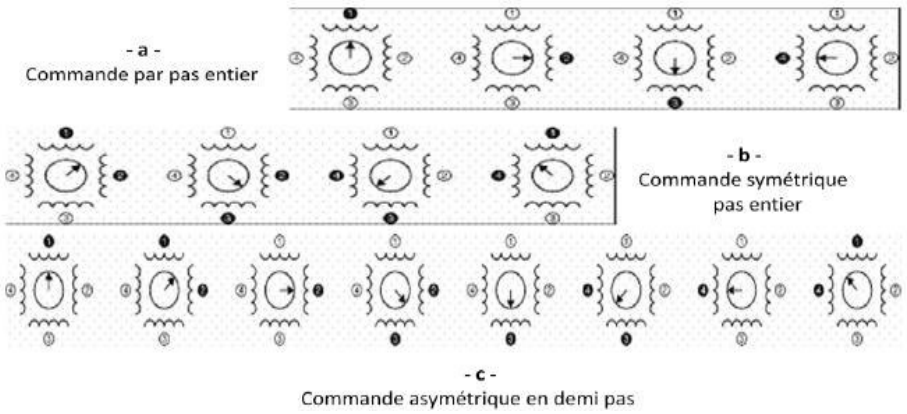


Figure I.18 Commande d'un moteur PAP [11]

I.V Afficheur LCD (Liquide Crystal Display)

I.V.1 Définition

Les afficheurs à cristaux liquides, autrement appelés afficheurs LCD (Liquide Crystal Display), sont des modules compacts intelligents et nécessitent peu de composants externes pour un bon fonctionnement. Ils consomment relativement peu (de 1 à 5 mA), ils sont relativement bons marchés et s'utilisent avec beaucoup de facilité [8]

Figure I.19 Afficheur LCD16x2 [8]

I. V.2 Brochage

Au-dessus de l'écran à cristaux liquides, on trouve une série de 14 broches aux rôles suivantes :

- Broche 1 : masse GND ;
- Broche 2 : VCC ;
- Broche 3 : luminosité ;
- Broche 5, R/W : sélection du mode lecture ou écriture.
- Broche 6, E : Commande des opérations d'écriture ou de lecture ;
- Broche 7 à 14 : utilisées pour le transfert des données ou des instructions.

Le transfert peut se faire sur 8 bits, toutes les broches sont alors utilisées, ou sur 4 bits, dans ce cas, seules les broches 11 à 14 sont utilisées [11]

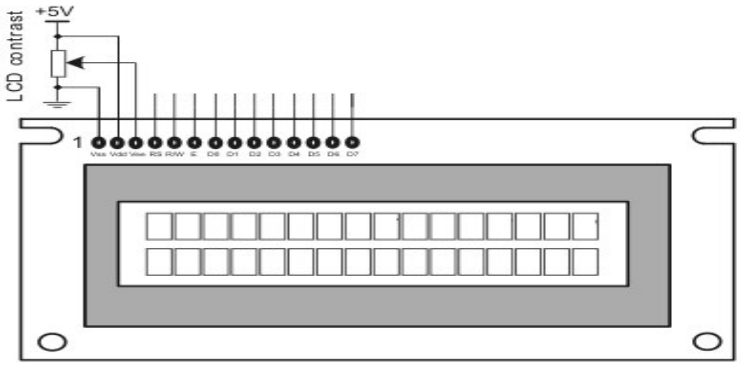


Figure I.20 Brochage d'un afficheur LCD [11]

I.V.3 Fonctionnement

Un afficheur LCD est capable d'afficher tous les caractères alphanumériques usuels et quelques symboles supplémentaires.

Chaque caractère est identifié par son code ASCII qu'il faut envoyer sur les lignes D0 à D7 broches. Ces lignes sont aussi utilisées pour la gestion de l'affichage avec l'envoi d'instructions telles que l'effacement de l'écran, l'écriture en ligne 1 ou en ligne 2, le sens de défilement du curseur [8] [10]

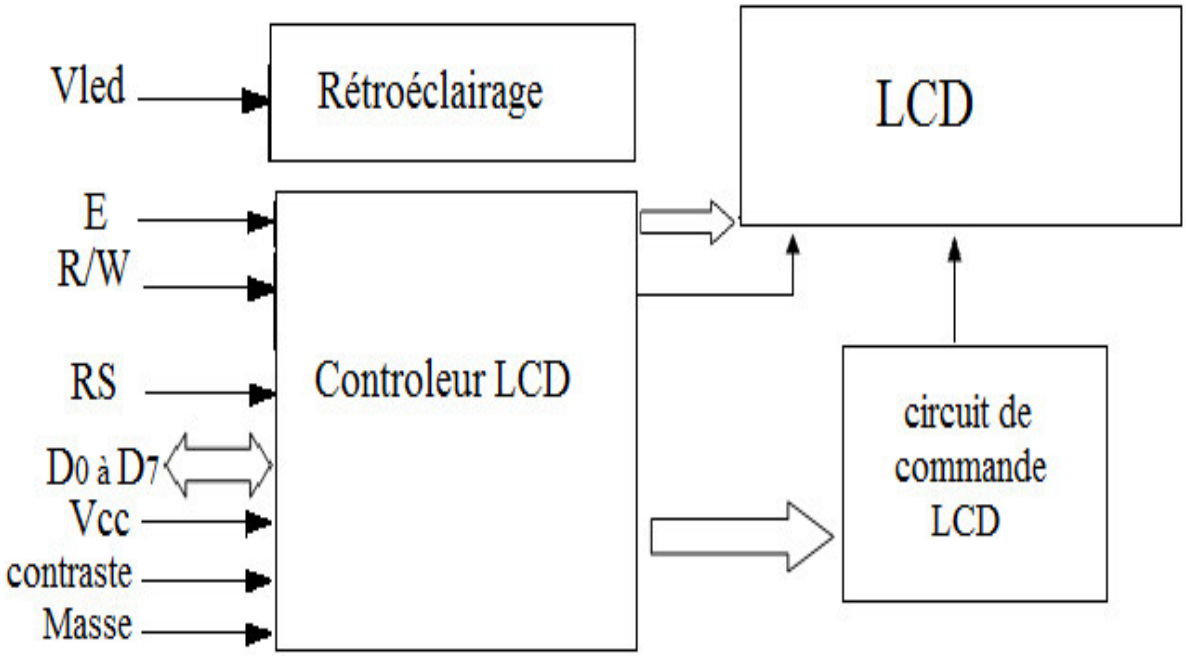


Figure I.21 Schéma fonctionnel d'un LCD [11]

I.VI Diode LED

I.VI.1 Définition

Une diode électroluminescente, est un composant optoélectronique capable d'émettre de la lumière lorsqu'il est parcouru par un courant électrique. [8]

Figure I.22 Diodes LED [8]

I.VI.2 Brochage

Les Diodes ont deux extrémités, une anode et une cathode. Cette diode émet de la lumière quand elle est polarisée en direct. Donc, il faut la connecter correctement à la tension d'alimentation. [11]

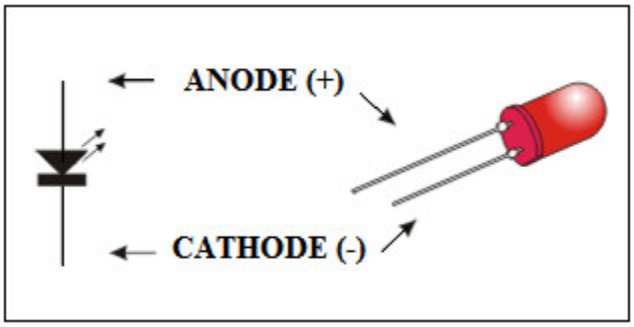


Figure I.23 Brochage de diode LED [8]

Conclusion

Dans ce chapitre, on a cerné les différents composants qu'on va utiliser pour réaliser le dispositif contrôleur de température, en donnant une idée générale sur les caractéristiques techniques et le brochage de chaque composant. Ce chapitre relève ainsi une utilité majeure pour ce qui suit puisqu'il détaille des concepts fondamentaux qu'on vise d'exploiter dans la réalisation de ce projet.

CHAPITRE II
Etude Conceptuelle

Introduction

La programmation des microcontrôleurs PIC est supportée par plusieurs langages de programmation. Dans notre projet nous avons opté pour le compilateur de MikroC qui est un compilateur en langage C (langage évolué). Ce choix est à la fois un choix personnel et un choix technologique parce qu'il est basé sur le langage C qui est facile et en douceur.

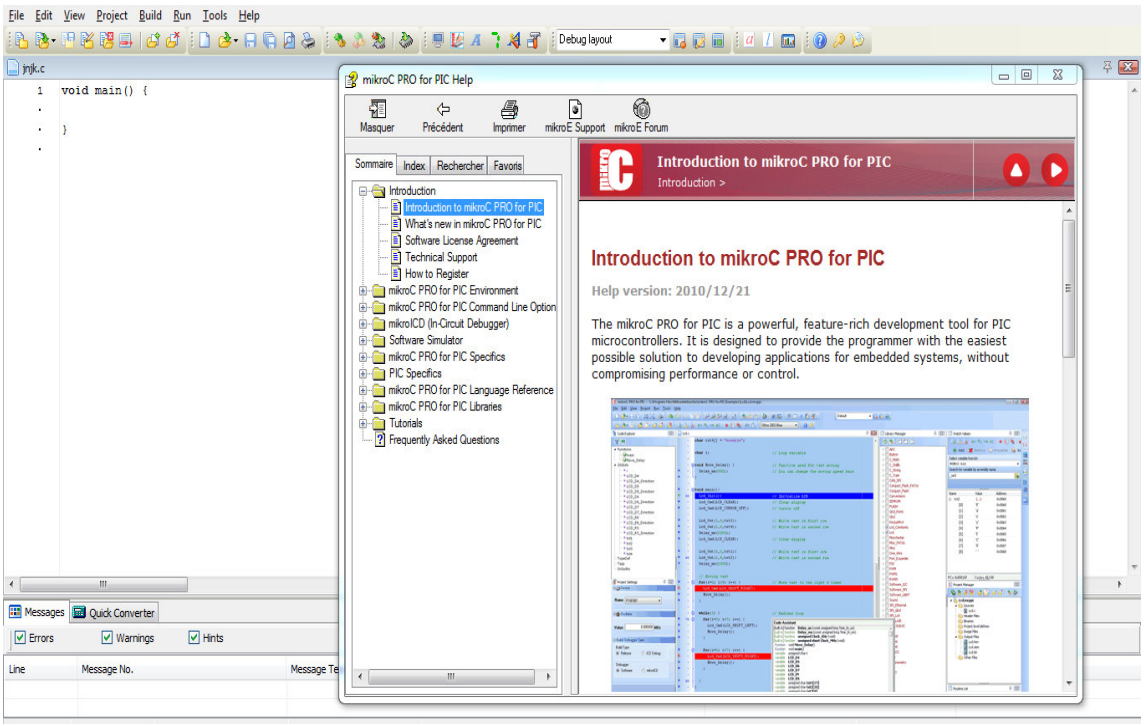
Dans ce chapitre nous présenterons l'algorithme et l'organigramme du programme ensuite nous le synoptique de la carte du contrôleur de température ainsi que le programme de fonctionnement de chaque bloc.

II.1 L'outil de programmation (MikroC)

Le « MikroC » est un compilateur pour PIC Conçu par la société « Mikroelektronika », le compilateur C nouvelle génération "MikroC" pour microcontrôleurs PIC bénéficie d'une prise en main très facile. Il comporte plusieurs outils intégrés (mode simulateur, terminal de communication, gestionnaire 7 segments,...etc); Il a une capacité à pouvoir gérer la plupart des périphériques rencontrés dans l'industrie (Bus I2C, 1Wire, SPI, RS485, Bus CAN, signaux PWM, afficheurs LCD et 7 segments...); de ce fait il est un des outils de développement incontournable et puissant.

Il est conçu pour fournir les solutions les plus faciles que possibles pour des applications se développant pour les systèmes à microcontrôleur. Il contient un large ensemble de bibliothèques de matériel, de composant et la documentation complète.

Toutes ces caractéristiques nous encouragent à choisir ce langage de programmation. De plus, le nombre des utilisateurs de ce logiciel devient de plus en plus important [3] [4].



FigureII.1 Interface du logicielle MikroC

II.1.1 Avantage du C

Pour la programmation de base, le C'est intéressant .Il permet rapidement, sans gros effort, de développer des programmes fonctionnels Il permet aussi de s'affranchir de connaissances complexes sur L'architecteur des PIC. Il a l'avantage de gérer facilement les boucles, les choix, ainsi que l'affichage[8].

II.1.2 Inconvénient du C

Le langage C de haut niveau propre au microcontrôleur, il permet de programmer plus intuitivement les logiciels de programmation en C transforment alors les lignes en C en lignes assembleurs directement compréhensibles par le microcontrôleur. Pour programmer efficacement, il est souvent nécessaire d'aller voir le code assembleur qui est le langage de bas niveau, il est donc conseillé d'avoir des bases solides en assembleur parceque plus qu'on est proche de la machine, plus la programmation est meilleure et bien contrôlée pour cela, il faut bien connaitre l'architecture du microcontrôleur[8].

II.2 L'Algorithme

1. Démarrage
2. Configuration des registres
3. Initialisation des ports
4. Initialisation du LCD
5. Fixer latempérature de référence TH à 25 (température ambiante)
6. Afficher le message de choix de changement de TH

- Si Non
  - Appuyer sur le bouton (NON) (RD7=1) pour confirmer le choix etpassez au programme principal ;
- Si Oui
  - Appuyer sur le bouton (OUI) (RD4=1) pour confirmer le choix ;

Réglage de TH ;

- Appuyer sur le bouton (+) (RD5=1) pour incrémenter TH ;
- Appuyer sur le bouton (-) (RD6=1) pour décrémenter TH ;

Réglage de TH en appuyant sur le bouton (SWITCH) (RD7=1) ;

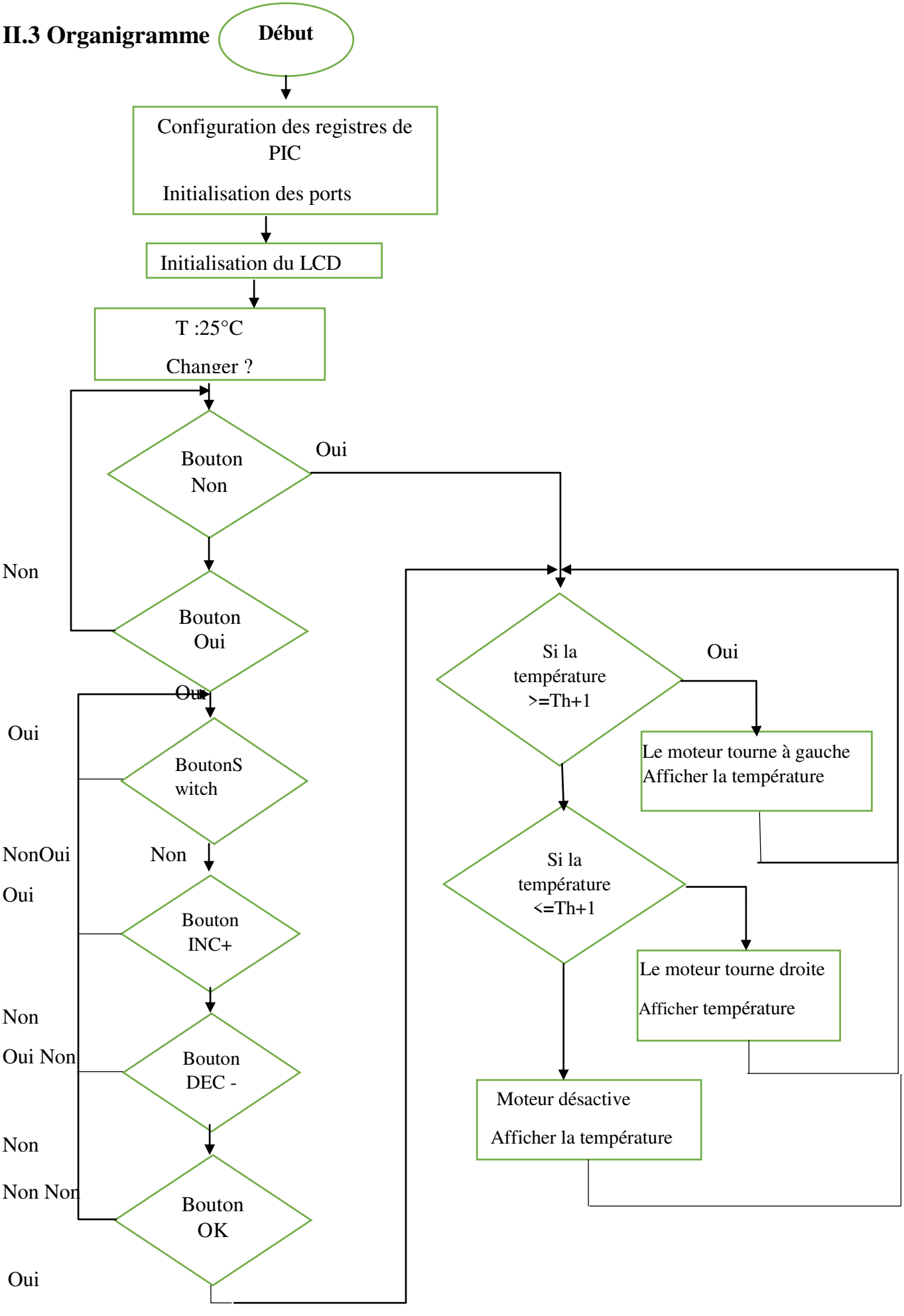
- Appuyer sur le bouton (+) (RD5=1) pour incrémenter TH ;
- Appuyer sur le bouton (-) (RD6=1) pour décrémenter TH ;

9. appuyer sur le bouton (OK) (RD4=1) pour confirmer la référence TH ;
10. Passez au programme principal ;

❖ Programme principale :

- Si la température mesurée est supérieure à TH le moteur tourne à gauche, la température s'afficheet la led rouge s'allume.
- Si la température mesurée est inférieure à TH Le moteur tourne à droite, la températures'affiche et la ledjaune s'allume.
- Si la température mesurée est égale à TH le moteur s'arrête, la température s'affiche et la ledvertes'allume.

II.3 Organigramme



II.4 Programme

II.4.1 Initialisation et configuration

Dans la première étape, on configure les entrées et les sorties du Pic l'on initialise les ports, comme le montre le programme suivant :

```
sbit LCD_RS at RC0_bit;

sbit LCD_EN at RC1_bit;

sbit LCD_D4 at RC3_bit;

sbit LCD_D5 at RC4_bit;

sbit LCD_D6 at RC5_bit;

sbit LCD_D7 at RC6_bit;

sbitLCD_RS_Direction at TRISC0_bit;

sbitLCD_EN_Direction at TRISC1_bit;

sbit LCD_D4_Direction at TRISC3_bit;

sbit LCD_D5_Direction at TRISC4_bit;

sbit LCD_D6_Direction at TRISC5_bit;

sbit LCD_D7_Direction at TRISC6_bit;// initialisation et configuration de l'afficheur LCD

TRISE=0;    //configuration de port E en sortie

PORTE=0 ;    //initialisation du port Eà 0

TRISA.F5 = 1 ;

TRISD=0b11110000 ; //configuration de port D bit de poids fort en entré et des poids faibles en sortie

PORTD=0;//initialisation du port D à 0
```

II.4.2 Convertisseur analogique numérique

Pour Initialiser le module ADC avec les paramètres par défaut nous avons déclarél'équation suivante :

ADC_Init(); //initialisation des ports analogiques numériques ADC

Le registre ADCON1 configure les fonctions des broches du port. Les broches du port peuvent être configurées comme entrées analogiques ou comme E / S numériques en fonction de la valeur du registre ADCON1. Dans notre programme, nous avons utilisé ADCON1 = 0x80 ; parce que nous utilisons PORTA comme port d'entrée analogique [8].

ADCON1=0b10000000 ; //configuration des ports analogiques numériques ADC

4Bits PCFG	**N°7 PE2**	**N°6 PE1**	**N°5 PE0**	**N°4 PA5**	**N°3 PA3**	**N°2 PA2**	**N°1 PA1**	**N°0 PA0**	**V_{ref+}**	**V_{ref-}**
0000	**A**	**A**	**A**	**A**	**A**	**A**	**A**	**A**	**VDD**	**VSS**

Tableau II.1 configuration de registre PCFG

A = Entrée Analogique.

D = I/O Digitale.

II.5 Schéma synoptique

L'idée est de réaliser un dispositif de commande automatique d'un système d'acquisition et decontrôle de température.

La partie commande qui compte le PIC16f877A, le cerveau du système, dont le rôle est la distribution des tâches à exécuter par chaque composant,en fonction de l'évolution des paramètres à surveiller, acquis par la partie capteur qui contient les différents capteurs utilisés.

La partie communication est responsable d'informer sur l'état du système. Par une comparaison entre les conditions prédéterminées et l'évolution des paramètres transmises par les capteurs, le microcontrôleur doit ordonner, si c'est nécessaire, à la partie actionneur de changer son état afin d'obtenir un retour convenable aux conditions recherchées.

Un capteur de température permet de convertir la température capturée vers une valeur numérique bien correspondante.

Cette valeur numérique est traitée par le microcontrôleur et affichée sur un écran LCD.

Nous avons rajouté un moteur pas à pas dont le rôle actionneur, en cas ou la valeur de température TH est préalablement programmée inférieure ou supérieure à la température capturée T celui-ci est actionné etune LED rouge ou jaune s'allume, il ne s'arrêtera que lorsque la température capturée T est égale à la température de référence TH.

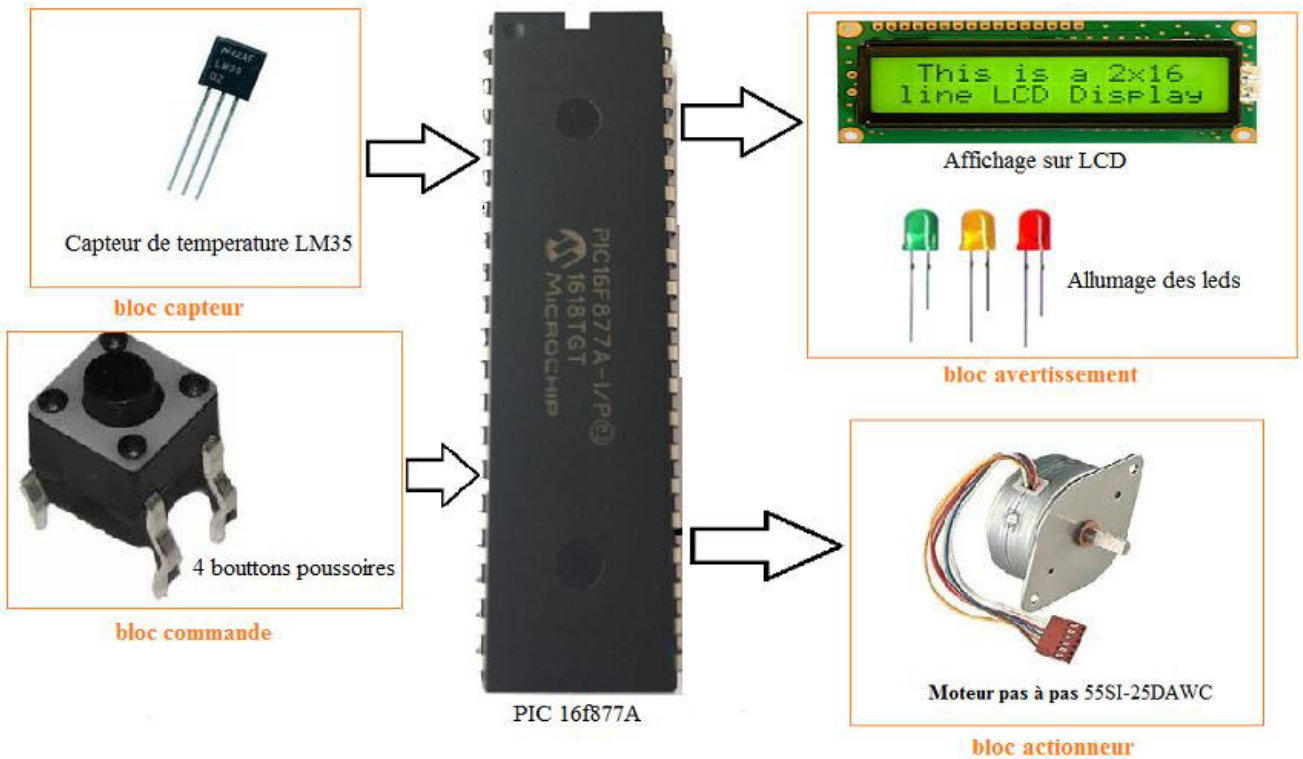


Figure II.2 Schéma synoptique

II.5.1 Bloc capteur

La collection des données qui sont la présence de la température se fait à l'aide des capteurs de température LM35. La donnée collectée à des variables qu'on utilise dans les taches suivantes de programme.

Pour les capteurs de température LM35 e, la tension délivrée par les capteurs et lue au moyen des instructions suivantes :

```
unsignedint valeurAD1 ;
  valeurAD1 = ADC_Get_Sample(4);//Récupérer la valeur numérique de AN4 (ou RA5)
  valeurAD1 = valeurAD1 * 4.8876;// x*5000/1023 mV //
```

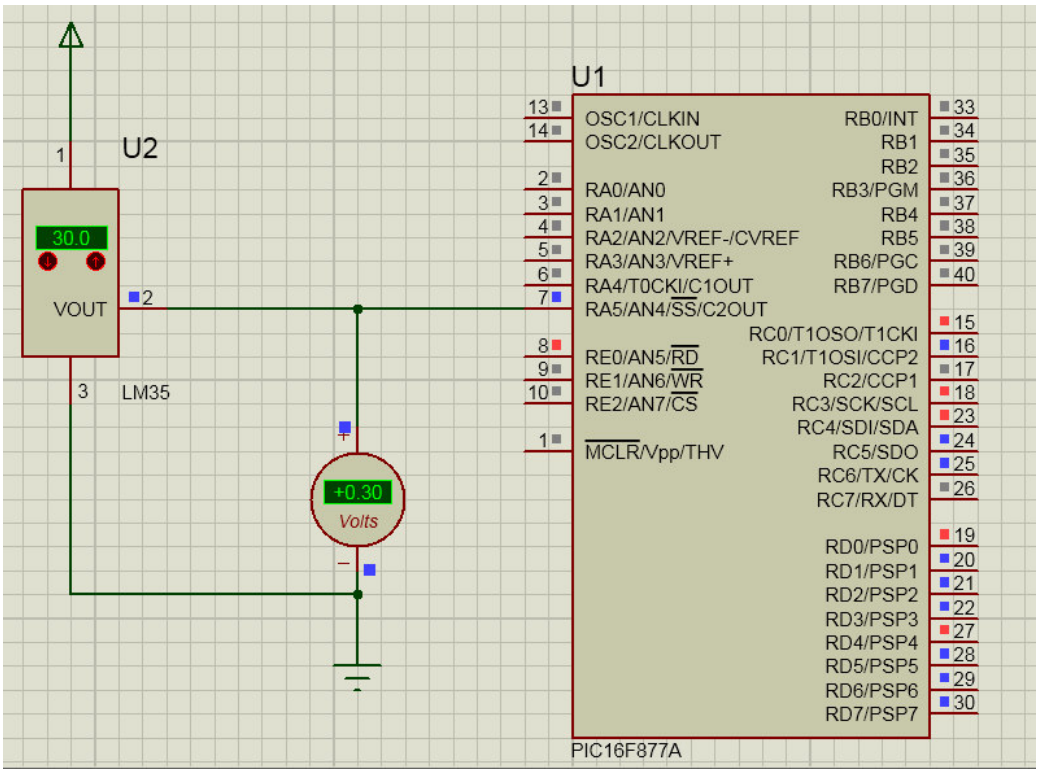
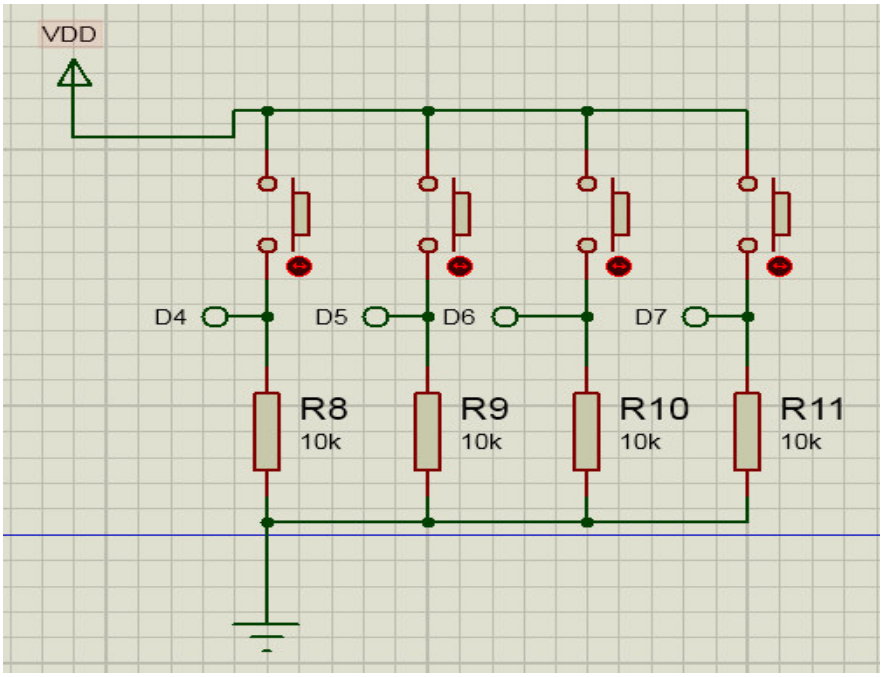


Figure II.3 circuit capteur microcontrôleur

II.5.2 bloc commande

Pour une facilité de variation de la température de référence à régler, dite consigne de température, il est essentiel de mettre un dispositif pratique, sans avoir à modifier le programme dans le PIC, dans notre système.

La référence varie à l'aide de 4 boutons poussoir,2 bouttons pour choisir le changement ou bien fixer la température de référence se trouvent sur le port D du PIC aux pattes 27 et 30, et 2 bottons l'un pour augmenter (Réf TH) et l'autre pour diminuer (Réf TH), se trouvent sur le port D du PIC aux pattes 28 et 29.



FigureII.4 Circuit boutons poussoirs de variation de température de référence

D4 : Oui/ok

D5 : (+)

D6 :(-)

D7 : Non/switch

II.5.3 Bloc d'affichage

L'affichage est une partie importante dans la plupart des applications à base de microcontrôleur. L'écran LCD (liquidcrystal display) est utilisé afin de permettre une communication de l'état du système à l'opérateur. Ces dispositifs viennent dans différentes formes et tailles. Certains écrans LCD n'ont qu'une seule ligne, tandis que d'autres peuvent avoir jusqu'à quatre. L'écran utilisé est un 2x16, c'est-à-dire, deux lignes et seize colonnes, facilitant ainsi l'affichage de message envoyé par le PIC. L'afficheur est connecté au port C, aux pattes RC0, RC1, RC3, RC4, RC5, RC6.

L'affichage sera au moyen des instructions suivantes :

```
chartxt_temp[3];

txt_temp[0]= (valeurtemp/100) + 48; //récupérer le 1er chiffre

txt_temp[1]= (valeurtemp/10)%10 + 48;// récupérer le 2eme chiffre

txt_temp[2]= (valeurtemp%10) + 48; //récupérer le 3eme chiffre

LCD_Chr(ligne,colonne,txt_temp[0]); //Afficher le 1er chiffre dans la colonne et définir lafonction printLCDtemp(ligne,colonne,valeurtemp)

LCD_Chr(ligne, colonne+1,txt_temp[1]); //Afficher le 2eme chiffre dans la colonne +1 , printLCDtemp(ligne,colonne,valeurtemp)

LCD_Chr(ligne, colonne+2,txt_temp[2]); //Afficher le 3eme chiffre dans la colonne +2 , printLCDtemp(ligne,colonne,valeurtemp)

LCD_Chr_CP('C') ; //afficher le caractère 'C'à la position actuelle du curseur
```

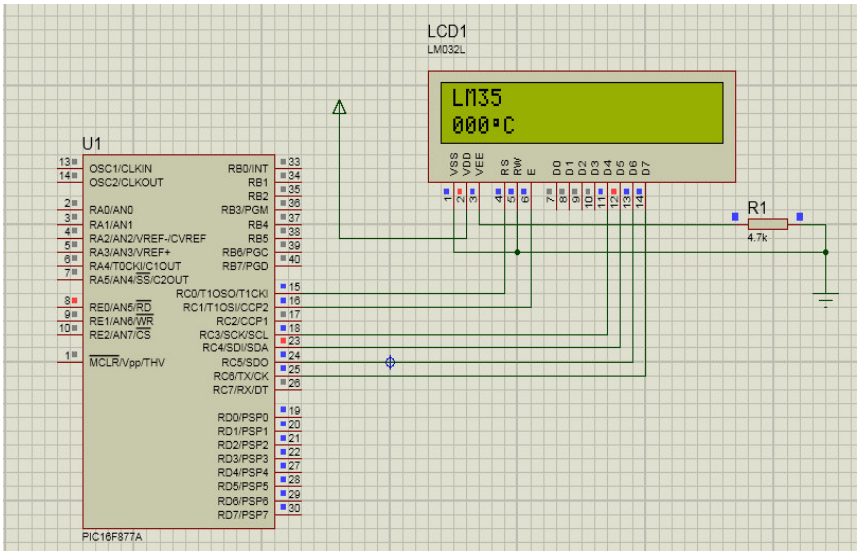


Figure II.5 Circuitmicrocontrôleur afficheur LCD

II.5.4 bloc actionneur (moteur pas à pas)

Le séquenceur utilisé pour l'aiguillage des impulsions sur les différentes bobines du moteur est le circuit intégré ULN2003.

L'ULN2003 est un réseau de transistors Darlington, à sept entrées de niveau logique qui passent à sept sorties, qui supporte une tension maximale de 50V et des charges de courant jusqu'à 500mA pour chaque sortie. Le moteur sera alimenté par les deux fils communs, en 12V, et le séquenceur fournira les quatre fils restant en 0V. Le port responsable de commander le moteur pas à pas est le Port D, donc il sera configuré comme sortie.

pasmoteur[0] = 0b00000001;

pasmoteur[1] = 0b00000010;

pasmoteur[2] = 0b00000100;

pasmoteur[3] = 0b00001000;

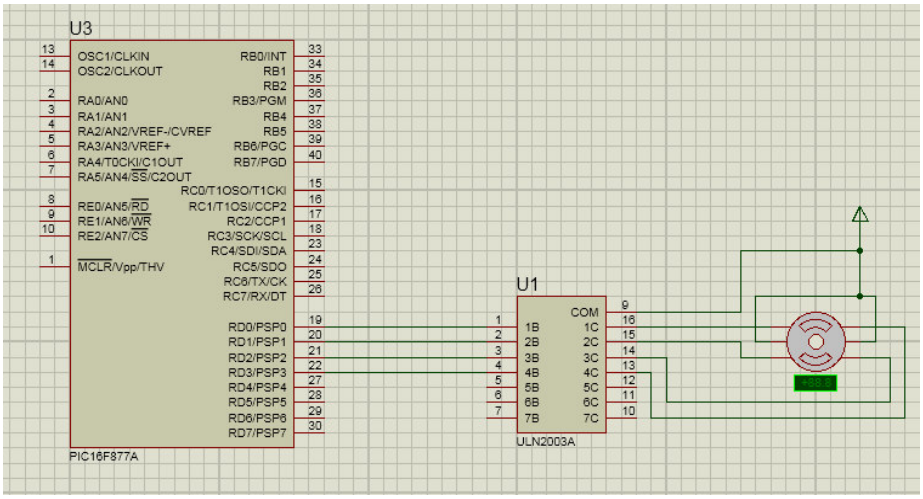


Figure II.6 circuit microcontrôleur moteur

II.6 Circuit Reset

Cette patte se trouve concrètement au premier emplacement dans le PIC, elle permet la remise à zéro du programme, c'est-à-dire, que le programme redémarre du début.

L'initialisation du programme s'effectue si une tension basse, zéro volt, est mise sur cette patte. Un système de commutation avec un bouton poussoir et une résistance, pour la fixation du courant entrant, est mis sur cette patte pour aider à la réinitialisation.

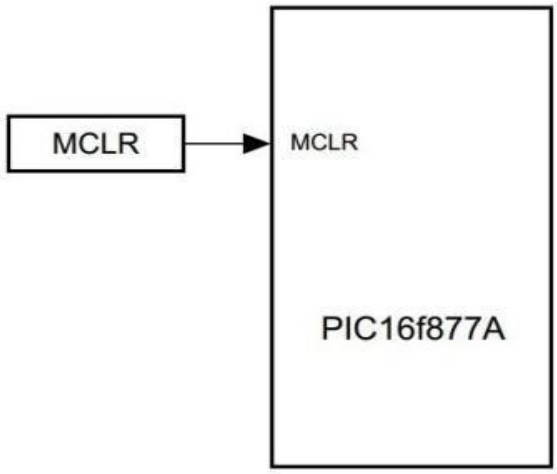


Figure II.7 Circuit reset

II.7 Circuit Quartz

La synchronisation du microcontrôleur doit être présente pour le pilotage de ce dernier, elle se fait à l'aide d'un quartz de 4 Mhz, mis avec deux condensateurs de filtrage, de valeur prédéterminée par le **datasheet** du PIC16f877A, pour avoir un signal carré.

Les pattes concernées par le cortège du pilotage à l'aide d'un quartz sont présentes dans le PIC aux numéros 13 et 14.

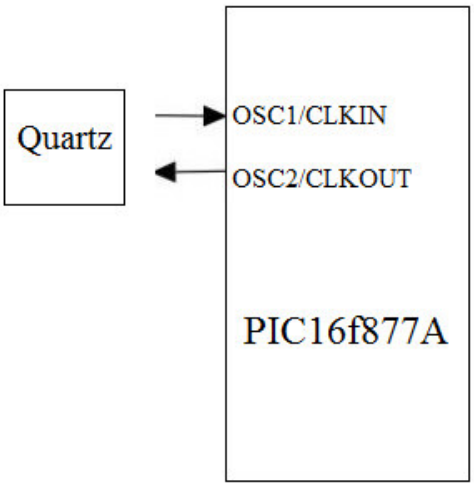


Figure II.8 circuit quartz

II.8 Circuit Alimentation

Pour alimenter notre circuit, il est nécessaire d'utiliser un régulateur de voltage. Le circuit fonctionne avec 5 volts, donc on utilise le L7805 pour générer la tension demandée.

Pour améliorer la régulation de la tension il est préférable d'utiliser en entrée une tension supérieure à 7 volts, en ajoutant des condensateurs, pour éliminer les oscillations et avoir une tension continue.

Conclusion

La programmation du microcontrôleur est nécessaire après la réalisation du projet. Cela consiste à créer un programme source. Après l'édition du programme, sa compilation permet de voir s'il n'y a pas d'erreurs. Si tout est correct, le fichier de format hexadécimal (.hex) est créé. En effet cette étape est indispensable pour que le programme soit reconnu et exécuté par le microcontrôleur. Ainsi, une fois le fichier hexadécimal créé, on le transfert vers la mémoire programme du PIC.

CHAPITRE III
Conception et Réalisation

Introduction

Ce chapitre est composé en deux parties essentielles, la première consiste à la présentation de l'environnement de simulation ISIS Proteus, la réalisation du système de contrôle de température avec le capteur linéaire et la représentation des résultats des tests obtenus.

Pour la deuxième partie nous allons étudier un capteur non linéaire et de l'exposer à l'étalonnage pour objectif de le rendre linéaireen effectuant un programme sous l'outil de programmation Matlab et afin de valider notre approche, nous avons développé notre programme avec le logiciel MikroC.

La partie conception du circuit et simulation a été réalisée avec le logiciel Porteus.

Partie I Cas d'un capteur linéaire

III.1 L'outil de simulation Proteus

Proteus est une suite logicielle destinée à l'électronique développée par la société labcenter Electronics.

Le logiciel ISIS de Proteus est principalement connu pour éditer des schémas électriques. Par ailleurs il permet également de simuler ces schémas ce qui permet de déceler certaines erreurs dès l'étape de conception. Les circuits électriques conçus grâce à ce logiciel peuvent être utilisés dans des documentations, car le logiciel permet de contrôler la majorité de l'aspect graphique des circuits. [11]

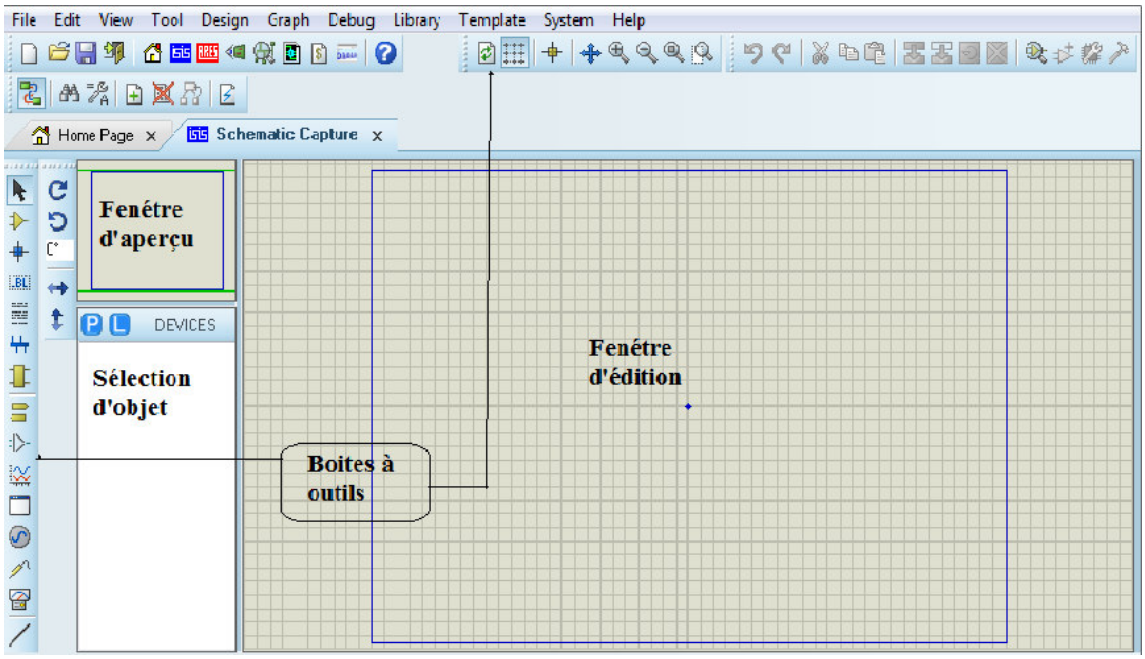


Figure III.1 Interface du logicielProteus

Cette suite logicielle est très connue dans le domaine de l'électronique de nombreuse entreprise et organismes de formation. Cet outil possède des avantages :

- L'outil de création de prototypes virtuel permet de réduire les couts matériels et logiciel lors de la conception d'un projet.[11]

III.2. Étapes de développement du programme

- La 1ère étape : l'algorithme.
- La 2ème étape : écriture de programme.
- La 3ème étape : Simulation de schéma électrique.
- La 4ème étape : Transfert du programme vers le pic

III.3 Simulation de la carte

Avant de passer à la réalisation pratique du système nous avons eu recours à la simulation des différentes parties du système.

Pour cela on a utilisé le logiciel ISIS qui est un logiciel pratique de simulation en électronique.

Le schéma électrique du dispositif réalisé est donné par la figure ci-dessous

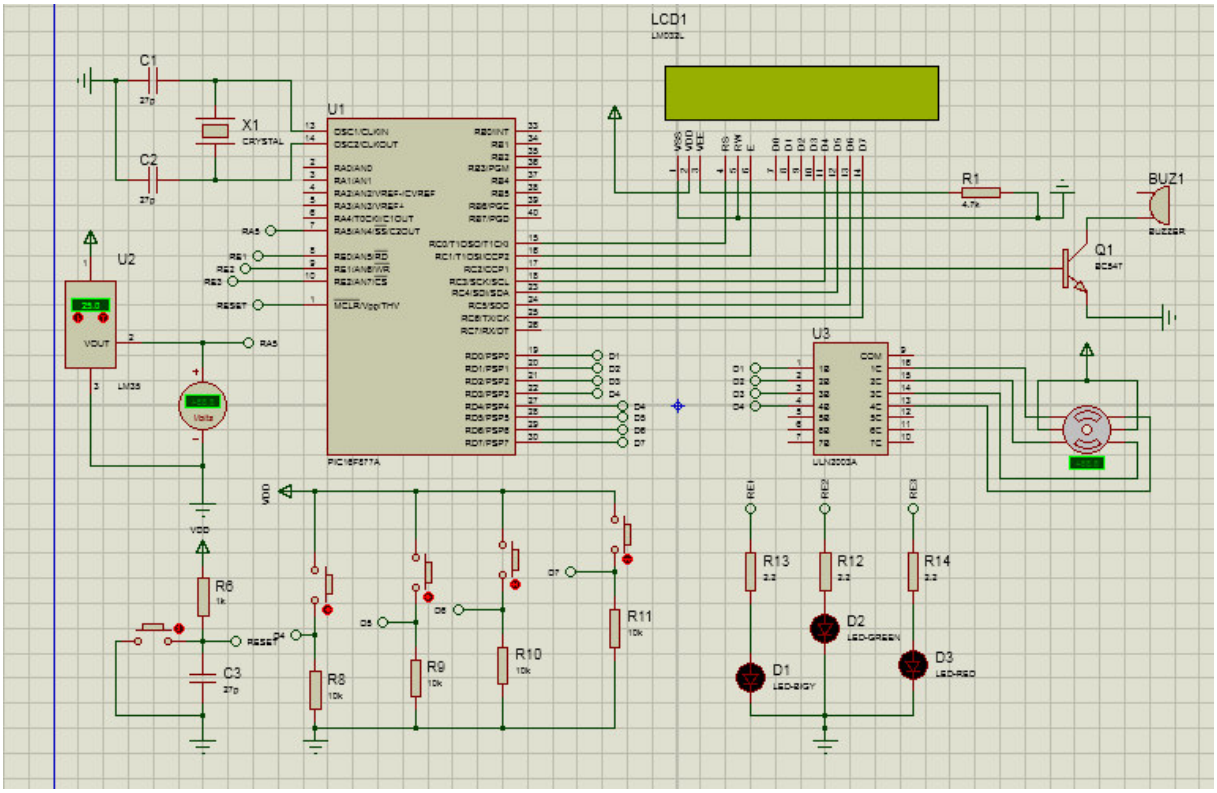


Figure III.2 Schéma électrique sous ISIS

III.4 Simulation et Tests

Avant d'effectuer le test final nous avons reproduit le circuit dans le logiciel de simulation PROTEUS_ISIS les figures ci-dessous**,** illustrent les résultats obtenus.

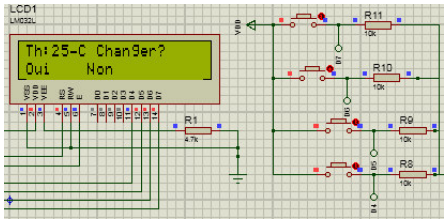
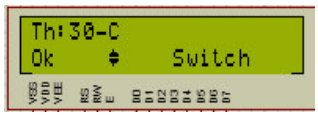
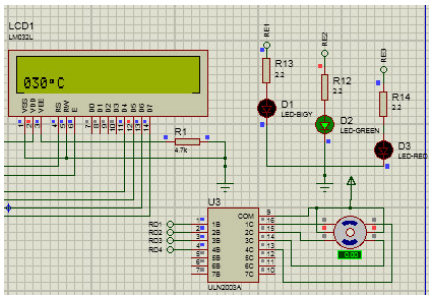
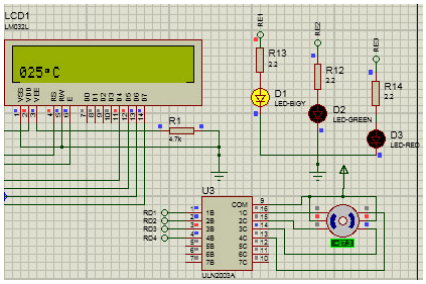
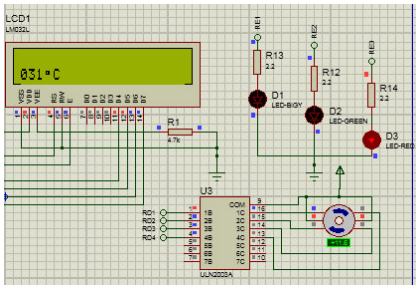
Simulation de l'état du système	**Résultats obtenues**
 	• A l'état initial : affichage de la température de référence (TH) fixée à 25° c et du message de choix de changement • Changer la température de référence (TH) à 30°c
	• T= TH : - Led verte allumée - moteurà l'état d'arrêt
	• T<= TH+1 : - Led jaune allumée - Moteur tourne à gauche
	• T>=TH+1 : - Ledrouge allumée - Moteur tourne à droite

Tableau III.1 résultats de la simulation sur Proteus

TH (température de référence)T (température de l'environnement)

III.5 Interprétation des résultats

Après avoir effectué la simulation sous Proteus en implémentant le programme dans le microcontrôleur qui est vérifié avec le fonctionnement du moteur et l'affichage de température avec précision sur l'afficheur LCD.

Partie IICas d'un capteur non linéaire

III.7 capteur non linéaire

III.7.1 Définition

Un capteur est dit non linéaire si ses valeurs changent quand on les compare à des valeurs étalon, ou à des valeurs données par d'autre capteurs normalisés et si sa sensibilité n'est pas constante. La relation entre grandeur physique à mesurer et grandeur électrique est alors non linéaire. [14]

III.8 Erreur

III.8.1 Définition

Une erreur est une différence entre la valeur vraie de la mesure et celle obtenue à partir de la réponse du capteur. [13]

III.9Étalonnage

Ensemble des opérations établissant, dans des conditions spécifiées, la relation entre les valeurs de la grandeur indiquée par un appareil de mesure ou un système de mesure, ou les valeurs représentées par une mesure matérialisée ou par un matériau de référence, et les valeurs correspondantes de la grandeur réalisée par des étalons.

- Note 1 : Le résultat d'un étalonnage permet soit d'attribuer aux indications les valeurs correspondantes du mesurage soit de déterminer les corrections à appliquer aux indications.
- Note 2 : Un étalonnage peut aussi servir à déterminer d'autres propriétés métrologiques telles que les effets des grandeurs d'influence.
- Note 3 : Le résultat d'un étalonnage peut être consigné dans un document parfois appelé certificat d'étalonnage ou rapport d'étalonnage. [9]

III.10 conception d'un capteur étalonné

III.10.1 chois de la fonction d'approximation

Dans notre étude nous avons choisis une fonction connue plus proche à un phénomène physique par exemple soit la charge d'un condensateur

$$U_c(t)=E\ (1-e^{-t/\tau}) \quad \text{(eq III.1)}$$

III.10.2 la fonction d'interpolation de LaGrange

Le problème de l'approximation d'une fonction f intervient dans plusieurs situations,comme par exemple :

1) La fonction f(x) est connue, mais difficile `à manipuler. L'approximation a pour but de remplacer f par une fonction plus simple, yy qui est plus accessible pour l'intégration, la différentiation, etc.
2) La fonction f(x) n'est pas connue, on ne connait que les valeurs dans certains points xi.

Les quantités données f(xi) = yi peuvent ˆêtre par exemple des mesures expérimentales.

Le but de l'approximation est alors de trouver une représentation synthétique (analytique)des données expérimentales.

On parle d'interpolation polynomiale quand v est un polynôme

Il existe un unique polynôme pn∈Pn (espace vectoriel des polynômesde degré inférieur ou égal à n) tel que

$\forall i,\ 0 \le i \le n,\ p_n(x_i) = f_i$

Il s'écrit sous la forme

$$p_n(x) = \sum_{i=0}^{n} f_i \ell_i(x), \quad avec \quad \ell_i(x) = \prod_{j \neq i} \frac{x - x_j}{x_i - x_j} \quad \text{(eq III.2)}$$

Les ℓi sont les polynômes d'interpolation de Lagrange. pn est le polynôme d'interpolationaux points xi pour les mesures fi. [18]

III.10.3 Programmation

III.10.3.1 Programmation sur Matlab

Nous avons choisis une fonction connue plus proche à Uc(t)=E (1-$e^{-t/\tau}$) qui est :

yy=100-80*exp(-0.5*xx)(eq III.3)

Le programme suivant montre les étapes des instructions à suivre pour le choix de la fonction :

```
close all;
clear all;
clc;
holdon;
xx=0:0.1:5;
yy=100-80*exp(-0.5*xx);% la fonction d'approximation
v=[xx;yy];
plot(xx,yy,'blue');
x=[0 1 2.5 4];% les valeurs mesurées par le capteur
y=[20 65 85 93];% les valeurs correspondantes aux valeurs mesurées
s=size(xx);
for i =1:s(2)
ty(i)=approche(xx(i),x,y); % fonction approchée
end
plot (x, y, 'ro')
Plot (xx,ty, 'red')
mean_error= mean(abs(yy-ty));% calcul de l'erreur absolue moyen
```

Le programme suivant montre les étapes des instructions à suivre pour la fonction d'interpolation de LaGrange

```
function v = approche(myx,x,y)
v=0
for i =1:4
      li=1;
for j = 1:4
if(i~=j)
       li=li*(myx-x(j))/(x(i)-x(j));% polynôme d'interpolation de LaGrange
end
end
    v=v+y(i)*li;% polynôme d'interpolation aux points x sur les mesures
end
```

III.10.3.1.1Tests et résultats

Essai 1

Et après du programme suivant (sous Matlab on obtient le graphe III.3)

```
Untitled.m | approche.m

1  close all;
2  clear all;
3  clc;
4  hold on;
5  xx=0:0.1:5;
6  yy=100-80*exp(-0.5*xx);
7  v=[xx;yy];
8  plot(xx,yy,'blue');
9  x=[0 1.65 3.3 4.95];
10 y=[20 45 52  99];
11 s=size(xx);
12
13 for i =1:s(2)
14     ty(i)=approche(xx(i),x,y);
15 end
16 plot(x,y, 'ro')
17
18 plot(xx,ty, 'red')
19
20 mean_error= mean(abs(yy-ty));
21
```

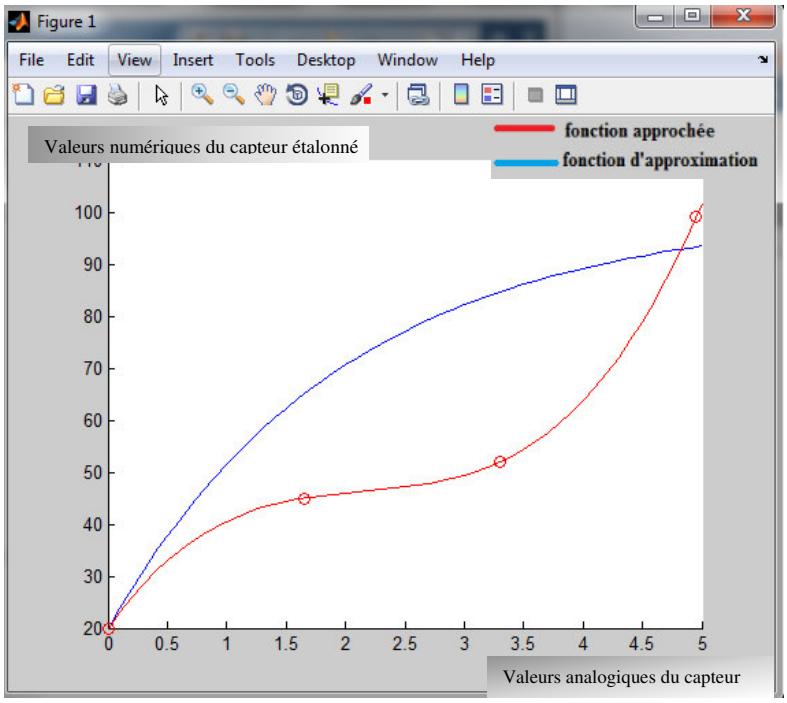


Figure III.3 graphe essai 1 obtenu après exécution du programme

>>mean_error= mean(abs(yy-ty))

mean_error =18.8788

Interprétation : on a entré des valeurs arbitrairement, on constate que les deux grapheséloignés **résultat** : l'écart entre les 2 courbes est large.

Essai 2 On a changé les valeurs de x et y

x=[0 1 1.5 5] ;

y=[20 5065 99] ;

Et après l'exécution nous avons abstenu le graphe et l'erreur suivants

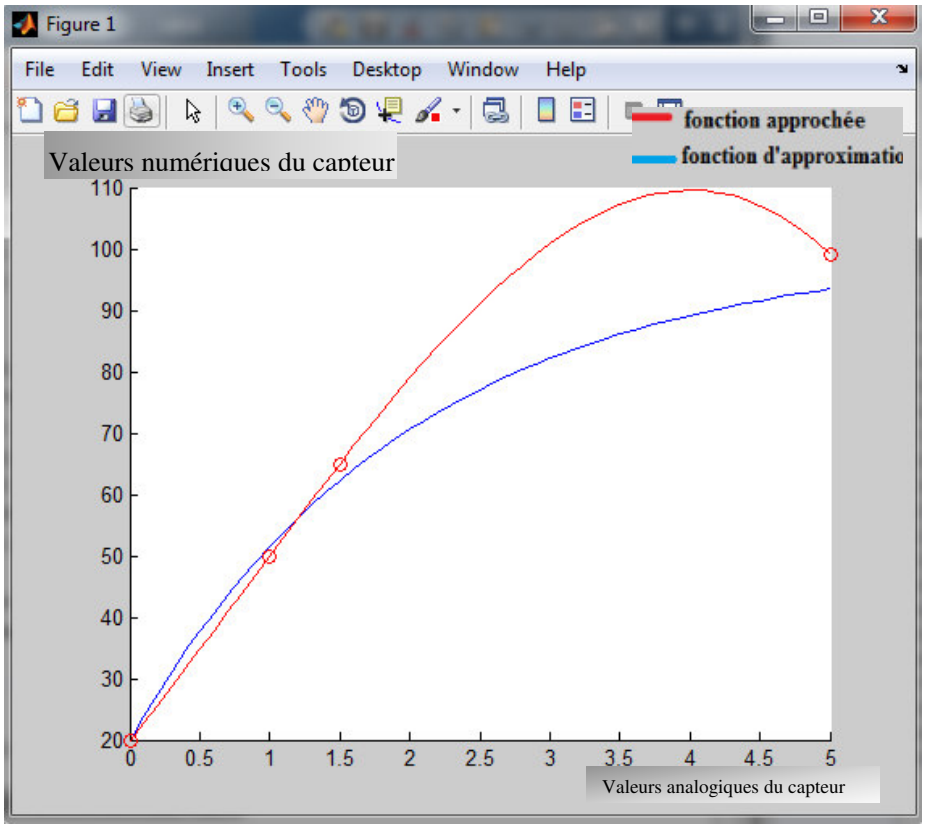


Figure III.5graphe essai 2 obtenu après exécution du programme

>>mean_error= mean(abs(yy-ty))

mean_error =10.7840

Interprétation : nous avons choisis des valeurs qui sont proches par rapport à la valeur initiale 20

Résultat : Les deux graphes sont proches au début et s'éloignent vers la valeur maximale

L'écart est moin large par rapport à l'essai 1

Essai 3 On a changé les valeurs de x et y

x=[0 1 2.54] ;

y=[20 65 85 93] ;

Et après l'exécution nous avons abstenu le graphe et l'erreur suivants :

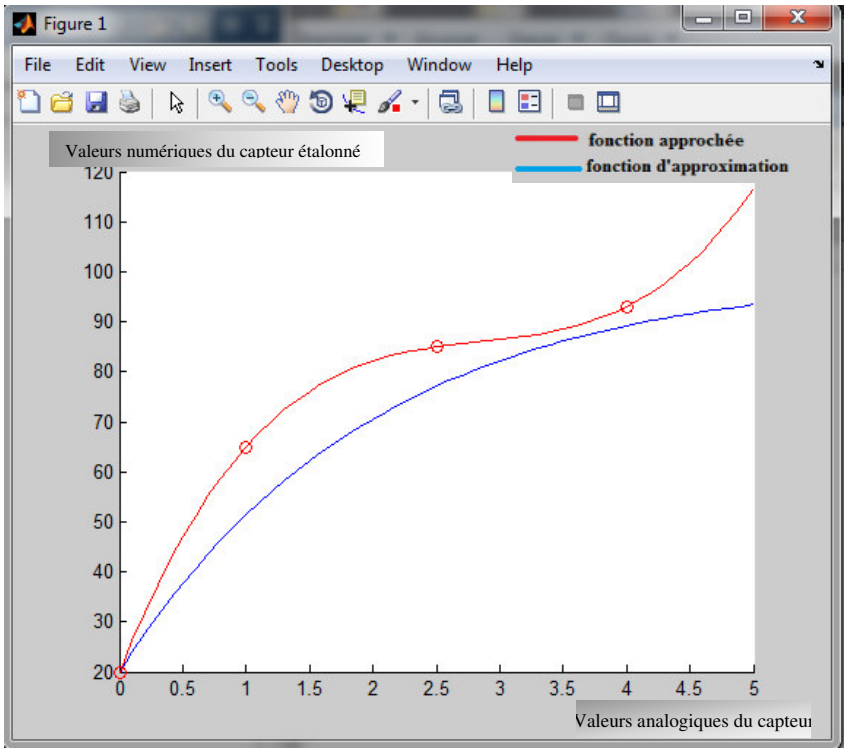


Figure III.6graphe essai 3 obtenu après exécution du programme

>>mean_error= mean(abs(yy-ty))

mean_error = 14.9483

Interprétation : nous avons choisis des valeurs qui sont proches par rapport à la valeur maximale 99

Résultat Les deux sont éloignés au début après il se rapprochent vers la valeur maximale

L'écart entre les 2 courbes est large

Essai 4

On a changé les valeurs de x et y

x=[0 1.5 35] ;

y=[20 62 82 93] ;

Et après l'exécution nous avons abstenu le graphe et l'erreur suivants :

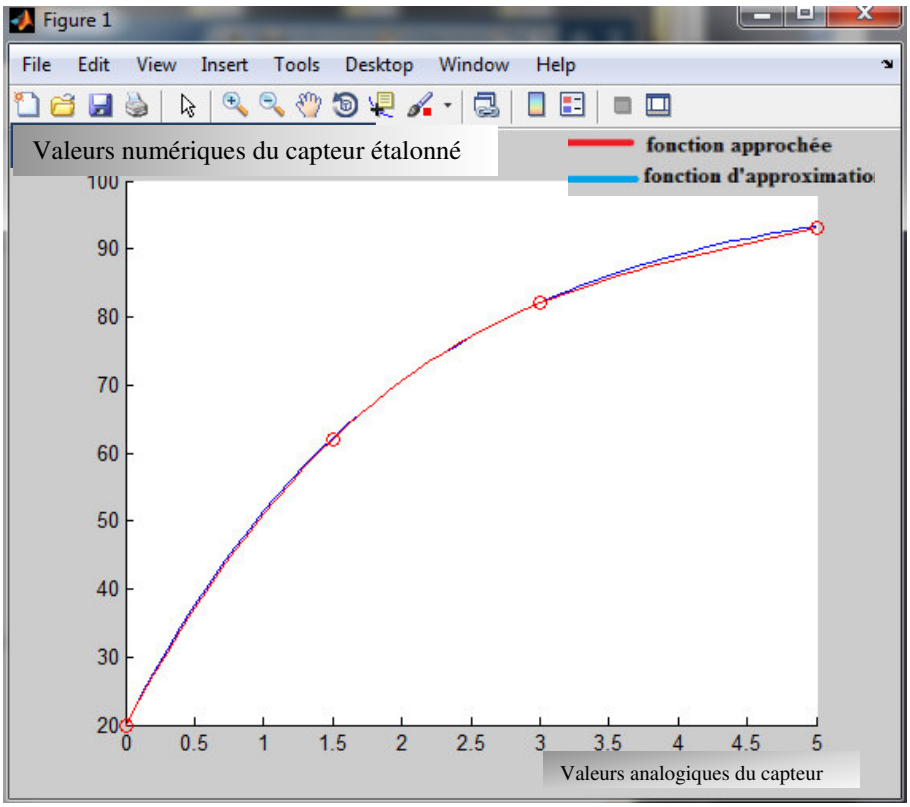


Figure III.7graphe essai 4 obtenu après exécution du programme

>>mean_error= mean(abs(yy-ty))

mean_error =

0.3890

Interprétationon constate que les deux graphes sont presque superposés

Résultat L'erreur diminue donc on approche au meilleur résultat

Essai 5 On a changé les valeurs de x et y

x=[0 1.65 3.3 4.95] ;

y=[20 65 85 93] ;

Et après l'exécution nous avons abstenu le graphe et l'erreur suivants :

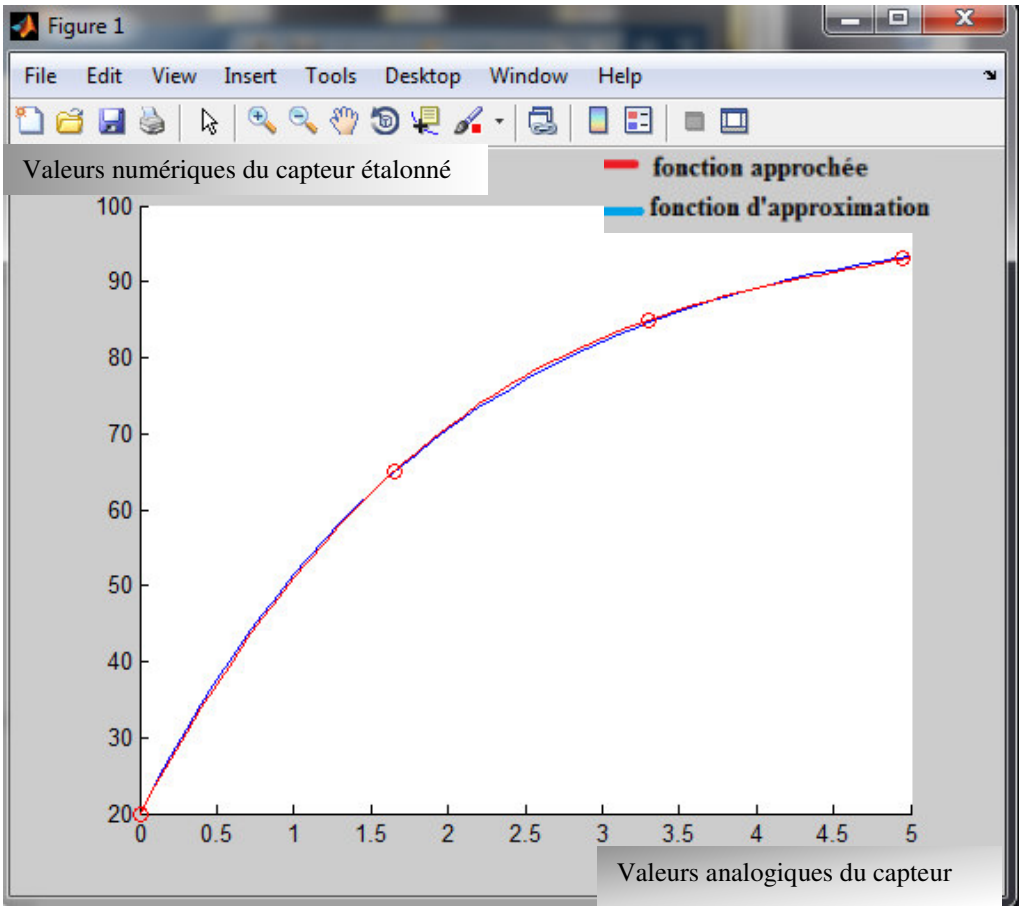


Figure III.8graphe essai 5 obtenu après exécution du programme

>>mean_error= mean(abs(yy-ty))

mean_error =

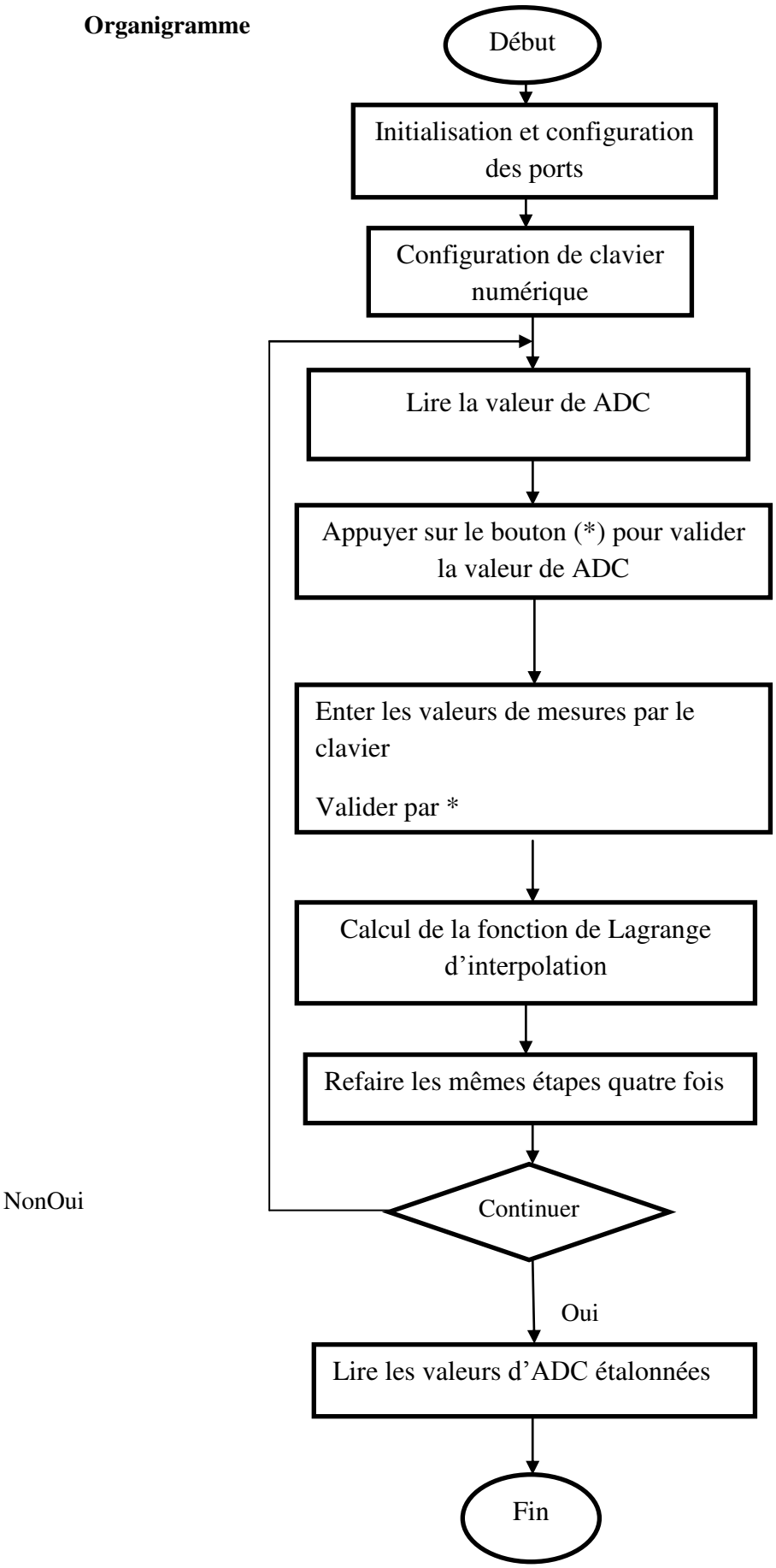
0.3384

Interprétation : Après avoir entré des valeurs qui sont équidistantes et on constate que les deux graphes sont superposés.

Résultat nous avons eu un meilleur résultat de l'erreur qui est de 0.3 donc une meilleure précision, le choix de la fonction d'approximation est plus évident.

III.10.3.2 Programmation sur MikroC

Organigramme



III.10.3.2.1Tests et résultats

On a suivi les étapes dans l'organigramme et on a obtenu les résultats ci-dessous

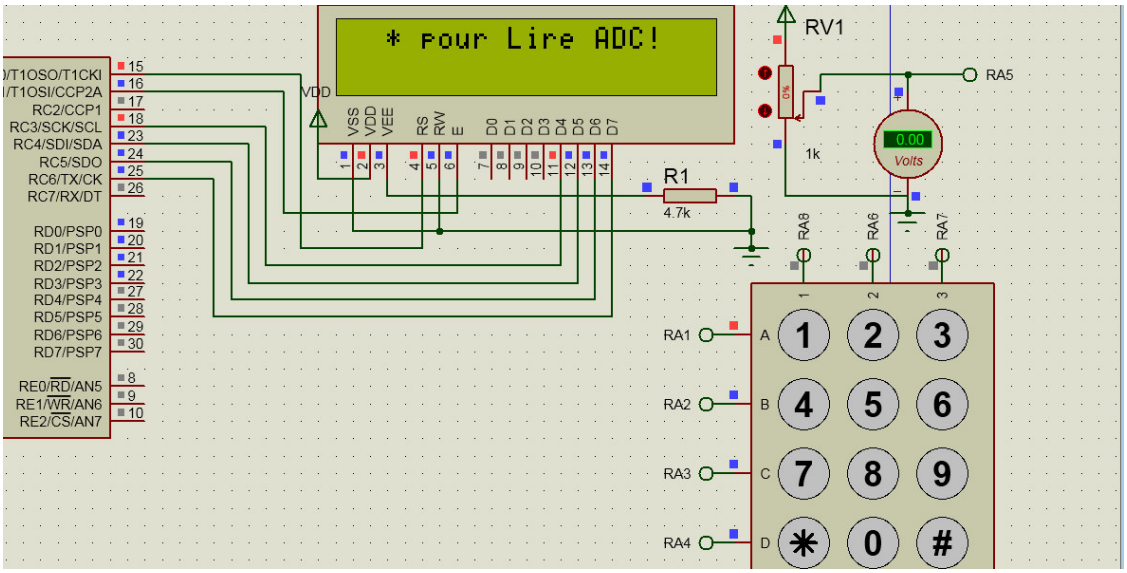


Figure III.9 lire ADC

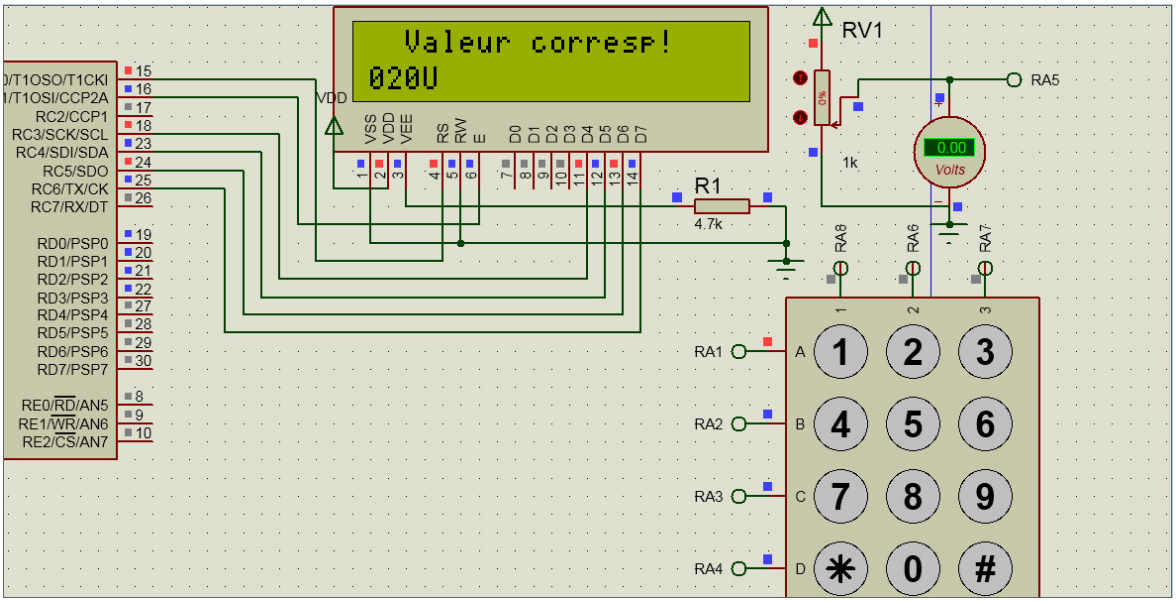


Figure III.10 Entrer la première valeur de mesures

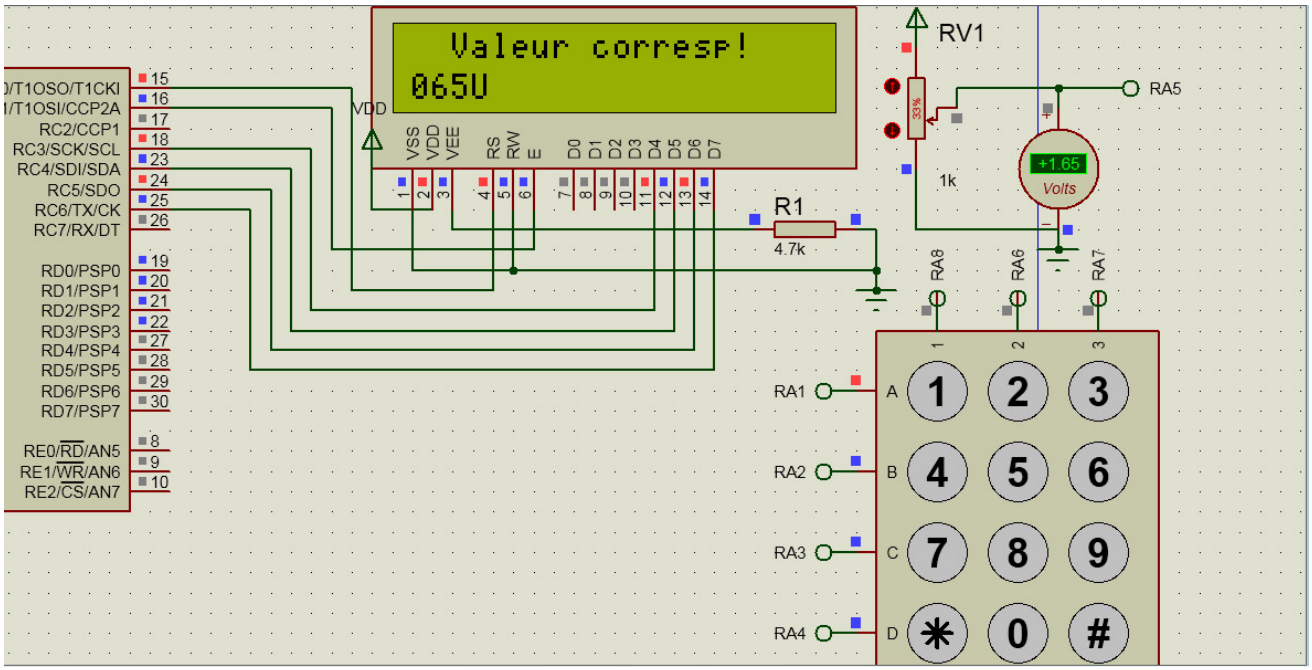


Figure III.11 Entrer la deuxième valeur de mesures

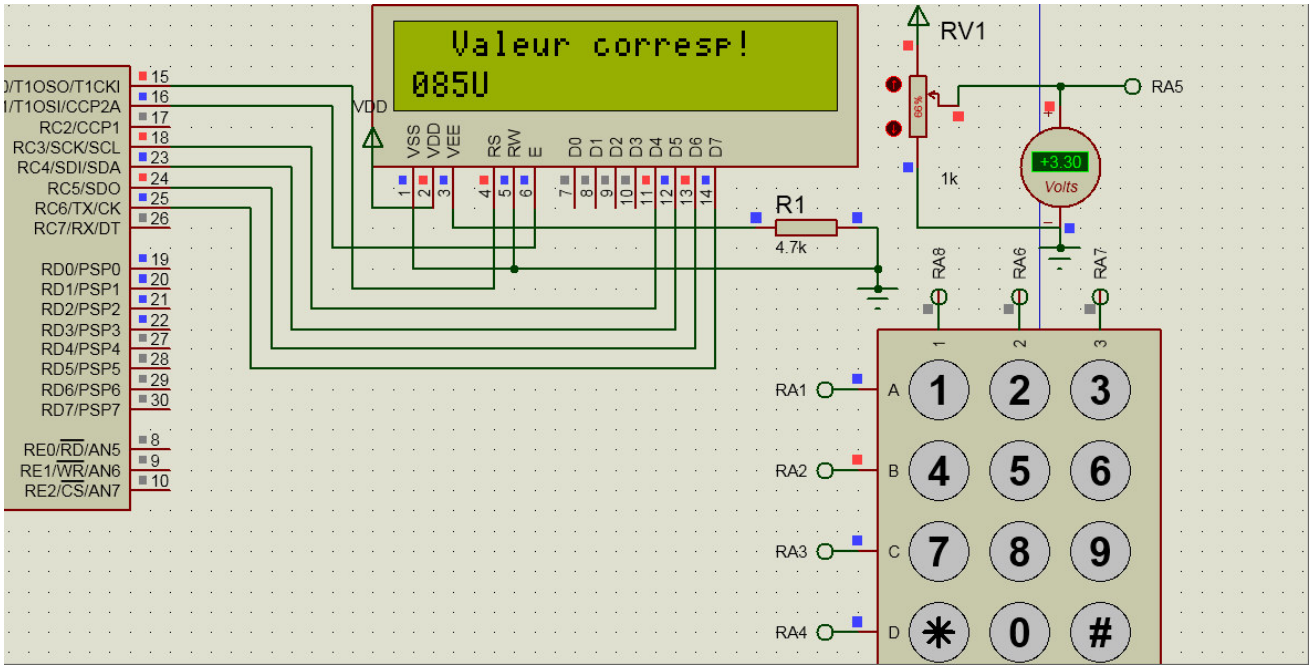


Figure III.12 Entrer la troisième valeur de mesures

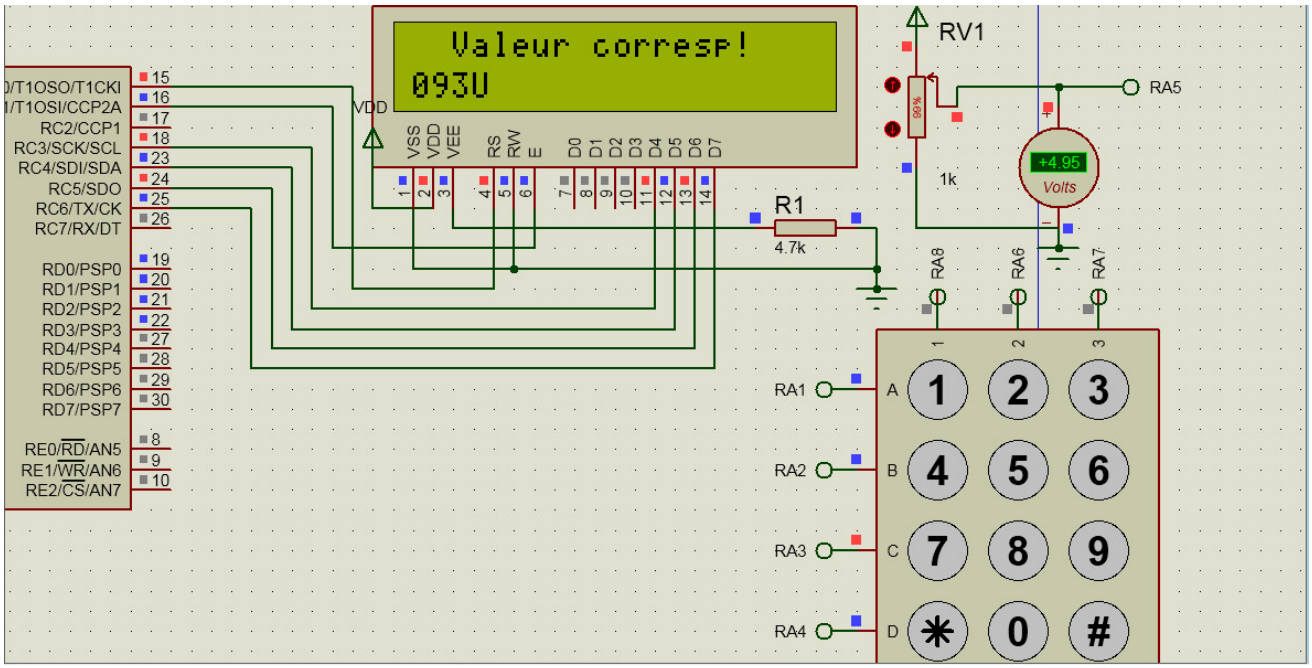


Figure III.13 Entrer la quatrième valeur de mesures

III.10.3.2.2 Interprétation des résultats

Après la validation des valeurs entrées on a obtenu les résultats suivants dans le tableau ci-dessous

On a pris le dernier exemple qui représente un meilleur résultat et on le compare avec les résultats obtenus de l'environnement de simulation Proteus

Valeurs analogiques de capteur (v)	0.1	0.6	1	1.4	1.7	2	3	3.4	4	4.6	5
v=v+y(i)*li sur Matlab	23.90	40.73	51.4	60.24	65.8	70.56	82.14	85.38	89.17	91.9	93.4
Valeurs numériques de capteur étalonné Sur MiKroC	23	40	50	60	65	70	82	85	89	91	93

Tableau III.2tableau comparatif des résultats obtenus après l'étalonnage avec ceux de la fonction v

Remarque

Si on compare les valeurs de l'interpolation de la fonction de LaGrange v et les valeurs étalonnées on constate que les résultats obtenus lors de la simulation sur Proteus sont les très proches de celles obtenueslors de l'exécution du programme de la fonction d'approximation 100-80*exp (-0.5*xx) sur Matlab.

Conclusion

Dans ce chapitre, on a étudié les deux cas de capteur, linéaire et non linéaire et d'après les résultats obtenus nous avons validé notre approchepour qu'elle soit valable pour n'importe quel type de capteur.

D'après les résultats obtenus dans la première partie nous avons constaté le bon fonctionnement du système avec le capteur linéaire.

A force d'utiliser le capteur à long terme sa précision diminue et l'erreur augmente et ses valeurs changent, si on mesure deux fois la même grandeur à deux moments différents on obtient deux grandeurs différentes donc on parle de la non fidélité du capteur pour cela on aura besoin d'effectuer l'opération de l'étalonnage des capteurs non linéaires dans la deuxième partie.

Et d'après les résultats obtenus dans la deuxième partie on a constaté les points suivants :

1- Pour que notre système suit la fonction d'approximation il faut choisir des valeurs qui sont équidistantes
2- Pour avoir de meilleur résultat de l'erreur donc une meilleure précision il faut choisir une fonction plus proche à un phénomène physique connu

CONCLUSION GENERALE

Conclusion générale

Le but de ce travail a été de réaliser un système contrôleur de température à base de PIC16F877A, pour permettre la mesure de température à travers un capteur numérique LM35, et de contrôler la valeur de cette dernière en mettant en marche ou en arrêt un moteur pas à pas. Pour cela nous avons cerné les différents composants qu'on va utiliser pour réaliser le dispositif contrôleur de température, en donnant les caractéristiques techniques et le brochage de chaque composant.

Ensuite on est passé au développement logiciel de ce projet qui consistait à l'élaboration de l'algorithme et du programme approprié. Le programme a été développé sous l'environnement MikroC. Enfin nous avons procédé à la réalisation du circuit Contrôleur de Température avec évaluation des performances.

En fait, ce projet a été une source de découverte de plusieurs domaines d'études tels que l'informatique pour la programmation embarquée, sans oublier le savoir-faire dans le domaine électronique qui consiste à réaliser pratiquement les circuits électroniques.

L'élaboration de ce travail dans le cadre du projet de fin d'étude, nous a permis d'approfondir nos connaissances théoriques en électronique et d'acquérir une bonne expérience au niveau de la réalisation pratique.

Les perspectives de ce travail sont :

- La gestion d'un réseau de capteurs et d'actionneurs.

- Associer d'autre type de capteurs (humidité, fumée, gaz toxique… etc.).

- Etalonner les différents types de capteurs non linéaires et même toute la chaine d'acquisition et d'action

- Etalonner les différents composants en mode pratique

- la supervision du système à distance en introduisant une application de contrôle et les modules de communication GPS-GPRS et GSM

- introduire les application androïde pour commander le système

ANNEXES

Annexe I

```
// Configuration des connections d'afficheur LCD avec PIC
sbit LCD_RS at RC0_bit;
sbit LCD_EN at RC1_bit;
sbit LCD_D4 at RC3_bit;
sbit LCD_D5 at RC4_bit;
sbit LCD_D6 at RC5_bit;
sbit LCD_D7 at RC6_bit;
sbit LCD_RS_Direction at TRISC0_bit;
sbit LCD_EN_Direction at TRISC1_bit;
sbit LCD_D4_Direction at TRISC3_bit;
sbit LCD_D5_Direction at TRISC4_bit;
sbit LCD_D6_Direction at TRISC5_bit;
sbit LCD_D7_Direction at TRISC6_bit;
// Fin de configuration
//Creation d'un caractère 'º' à l'aide de "LCD Custom Character"
   const char character[] = {0,14,10,14,0,0,0,0,0,0,0};
   const char characterud[] = {4,14,31,0,31,14,4,0};

void CustomChar(char pos_row, char pos_char) {
  char i;
    Lcd_Cmd(64);
    for (i = 0; i<=7; i++) Lcd_Chr_CP(characterud[i]);
    Lcd_Cmd(_LCD_RETURN_HOME);
    Lcd_Chr(pos_row, pos_char, 0);
}
void graus(char pos_row, char pos_char) {
  char i;
    Lcd_Cmd(64);
    for (i = 0; i<=10; i++) Lcd_Chr_CP(character[i]);
```

```
    Lcd_Cmd(_LCD_RETURN_HOME);
    Lcd_Chr(pos_row, pos_char, 0);
}
//fin de Creation
  void printLCDtemp(char ligne,char colonne,unsigned int valeurtemp);   // Initialiser
printLCDtemp
   char pasmoteur[4],pasmoteuree[4];
   int pas,i,valeur;valmax;
   char th;
   char r,a,b,p;
   void init_temp(){
       th=25;
       a=2;
       b=5;
       LCD_Out(1,1,"Th:25°C Changer?");
       LCD_Out(2,1,"Oui    Non");

       while(1){
          if(portd.b7==1){    // si boutton non app
               r='n';
               while(1){
                   if(portd.b7==0){
                               break;
                   } }
               break; }
           if(portd.b4==1){    // si boutton non app
               r='o';
               while(1){
                   if(portd.b4==0){
                               break;
                   } }
```

```
        break;
   } }
// process the response
if(r=='o'){
    LCD_Cmd(_LCD_CLEAR);
    LCD_Out(1,1,"Th:25°C");
    LCD_Out(2,1,"Ok         Switch");
    CustomChar( 2, 7)  ;
    p=4;
   while(1){
      if(portd.b4==1){    // boutton ok
        while(1){
            if(portd.b4==0){
                        th =a*10+b;
                        LCD_Cmd(_LCD_CLEAR);
                        break;
            } }
       break;
      }
       if(portd.b7==1){    // bouton switch
        if(p==4){
              p=5;
        }else{
             p=4;
        }
        while(1){
            if(portd.b7==0){
                        break;
            } } }
       if(portd.b5==1){    //  boutton up
```

```
        if(p==4){
              if(a!=9){
                    a=a+1;
              }
              LCD_Chr(1,p,a+48)   ;
        }
         if(p==5){
              if(b!=9){
                    b=b+1;
              }
              LCD_Chr(1,p,b+48) ;
        }
        while(1){
               if(portd.b5==0){
                    break;
                }  }}
if(portd.b6==1){    //  boutton down

        if(p==4){
              if(a!=0){
                    a=a-1;
              }
              LCD_Chr(1,p,a+48)     ;
        }
         if(p==5){
              if(b!=0){
                    b=b-1;
              }
              LCD_Chr(1,p,b+48)  ;
```

```
                }
                while(1){
                        if(portd.b6==0){
                            break;
        } } } } }
      //  LCD_Chr(ligne,colonne,txt_temp[0]       }
int ps=0;
void main(){
 LCD_Init();                  // Initialiser LCD
 LCD_Cmd(_LCD_CURSOR_OFF);     // Curseur est en off
 LCD_Cmd(_LCD_CLEAR);          // Effacer un texte sur l'ecrant LCD
 LCD_Out(1,1,"LM35");          // Ecrire le texte sur la 1ère ligne
ADC_Init();
 ADCON1=0b11000000;
 TRISA.F5 = 1;
 trisd=0b11110000;
 portd=0;
 pasmoteur[0]=0b00010001;
 pasmoteur[1]=0b00010010;
 pasmoteur[2]=0b00010100;
 pasmoteur[3]=0b00011000;
  init_temp();
 for (;;) {
  unsigned int valeurAD1 ;
  valeurAD1 = ADC_Read(4); //Récupérer la valeur numérique de AN4 (ou RA5)
  valeurAD1=valeurAD1*0.48876;    //x*5000/1023 mV    (/10 deg c)
  valeur= valeurAD1;
  printLCDtemp(2,1,valeur);   // afficher la valeur numérique de capteur LM35 sur la 2ème
ligne
  if(valeur>th){
       ps=ps+1;
```

```
        if(ps>3){
                ps=0 ;  }
        portd=pasmoteur[ps];
        Delay_ms(100);
   }
   else {
     if(valeur<th){
        ps=ps-1;
        if(ps<0){
                ps=3; }
        portd=pasmoteur[ps];
        Delay_ms(100);
     }  else{portd=0;}
   }}}
//Creation de la fonction printLCDtemp
void printLCDtemp(char ligne,char colonne,unsigned int valeurtemp) {


char txt_temp[3];
  txt_temp[0]= (valeurtemp/100) + 48; //recupére la 1er chiffre
  txt_temp[1]= (valeurtemp/10)%10 + 48;//recupére la 2eme chiffre
  txt_temp[2]= (valeurtemp%10) + 48; //recupére la 3eme chiffre
  LCD_Chr(ligne,colonne,txt_temp[0]); /*Afficher la 1er chiffre dans la colonne est définie de
la fonction printLCDtemp(ligne,colonne,valeurtemp)*/
  LCD_Chr(ligne,colonne+1,txt_temp[1]); //Afficher la 2eme chiffre dans la colonne +1 ,
printLCDtemp(ligne,colonne,valeurtemp)
  LCD_Chr(ligne,colonne+2,txt_temp[2]); //Afficher la 3eme chiffre dans la colonne +2 ,
printLCDtemp(ligne,colonne,valeurtemp)
   graus(ligne,colonne+3); //afficher le caractère 'º' sur LCD dans la colonne +3 ,
printLCDtemp(ligne,colonne,valeurtemp)
   LCD_Chr_CP('C'); //afficher le caractère 'C' à la position actuelle du curseur
}
```

Annexe II

```
// Configuration des connections d'afficheur LCD avec PIC
sbit LCD_RS at RC0_bit;
sbit LCD_EN at RC1_bit;
sbit LCD_D4 at RC3_bit;
sbit LCD_D5 at RC4_bit;
sbit LCD_D6 at RC5_bit;
sbit LCD_D7 at RC6_bit;
sbit LCD_RS_Direction at TRISC0_bit;
sbit LCD_EN_Direction at TRISC1_bit;
sbit LCD_D4_Direction at TRISC3_bit;
sbit LCD_D5_Direction at TRISC4_bit;
sbit LCD_D6_Direction at TRISC5_bit;
sbit LCD_D7_Direction at TRISC6_bit;
// Fin de configuration
//Creation d'un caractère "º" à l'aide de "LCD Custom Character"
//fin de Creation
char lire_clavier(void){
      char key;
      portb=0;
      while(1){
        portb=1;
        Delay_ms(30);
        if(portb.b4 ==1){
          key=1;
          break;
        }
        if(portb.b5 ==1){
          key=2;
          break;
```

```
}
 if(portb.b6 ==1){
   key=3;
   break;
}
portb=0b10;
Delay_ms(30);
if(portb.b4 ==1){
   key=4;
   break;
}
 if(portb.b5 ==1){
   key=5;
   break;
}
 if(portb.b6 ==1){
   key=6;
   break;
}
portb=0b100;
Delay_ms(30);
if(portb.b4 ==1){
   key=7;
   break;
}
 if(portb.b5 ==1){
   key=8;
   break;
}
 if(portb.b6 ==1){
```

```
          key=9;
          break;
        }
        portb=0b1000;
        Delay_ms(30);
        if(portb.b4 ==1){
          key=10;
          break;
        }
         if(portb.b5 ==1){
          key=0;
          break;
        }
         if(portb.b6 ==1){
          key=11;
          break;
        }
      }
       return key;
}
  void printLCDtemp(char ligne,char colonne,unsigned int valeurtemp);   // Initialiser
printLCDtemp


   int i;
int ps=0;
int j;
char tttt;
unsigned int y[4];
unsigned int x[4];
float aaa,bbb;
unsigned int x_mesur;
```

```
float sum,li, yy;
void approche(){
  sum=0;
  for (i=0;i<4;i++){
     li=1;
     for (j=0;j<4;j++){
        if(i!=j){
              aaa= (float)  x_mesur-(float)x[j];
              bbb= (float)x[i]-(float)x[j];
              aaa=aaa/bbb;
              li=li*aaa;
        }
     }
     yy=y[i];
     sum=sum+yy*li;
  }
}
int nc;
void main()
{
    trisb=0b1110000;
    ADC_Init();
    ADCON1=0b11000000;
     TRISA.F5 = 1;
     trisd=0b11110000;
     LCD_Init();                  // Initialiser LCD
     LCD_Cmd(_LCD_CURSOR_OFF);     // Curseur est en off
     LCD_Cmd(_LCD_CLEAR);         // Effacer un texte sur l'ecrant LCD
     for(i=0;i<4;i++){
              LCD_Cmd(_LCD_CLEAR);
```

```
Lcd_Out(1, 3, "* pour Lire ADC!");
while(1){
    tttt=lire_clavier();
    if( tttt==10){    //si boutton * appuye'
      x[i]= ADC_Read(4);
      delay_ms(1000);
      break;
    }
}
 LCD_Cmd(_LCD_CLEAR);
 Lcd_Out(1, 3, "Valeur corresp!");
y[i]=0;
nc=0; //nombre de caracters sesies
printLCDtemp(2,1,y[i]) ;
while(1){
     tttt=lire_clavier();
     if(tttt==10){delay_ms(1000); break;}
     if(tttt==11){
       if(nc>0){
       nc=nc-1 ;
       y[i]=y[i]/10;
       }
       printLCDtemp(2,1,y[i])  ;
       delay_ms(500);
        continue;
     }
     if(nc==0){
       y[i]=tttt;
        printLCDtemp(2,1,y[i])  ;
        nc=1;
```

```
                    delay_ms(500);
                    continue;
                  }
                  if(nc==1){
                    y[i]=y[i]*10+tttt;
                    printLCDtemp(2,1,y[i])  ;
                    nc=2;
                    delay_ms(500);
                    continue;
                  }
                  if(nc==2){
                    y[i]=y[i]*10+tttt;
                    printLCDtemp(2,1,y[i])  ;
                    nc=3;
                    delay_ms(500);
                    continue;
                  }
            }   }
     LCD_Cmd(_LCD_CLEAR);
      for (;;) {
        unsigned int valeurAD1,v,l[4];
        unsigned int val_corr;
       x_mesur = ADC_Read(4); //Récupérer la valeur numérique de AN4 (ou RA5)
       approche(); // qui calcule sum = valeur correspondente de de l'adc
      val_corr=(unsigned)sum;
        printLCDtemp(2,1,val_corr);   // afficher la valeur numérique de capteur LM35 sur la
2ème ligne
        delay_ms(100);
     }
}
```

BIBLIOGRAPHIE

Bibliographie

[1] Christian Tavernier « Les microcontrôleurs PIC Description et mise en œuvre. Nouvelle présentation de la deuxième édition », Edition mai 2002.

[2] Christian tavernier « Microcontrôleurs PIC 24 Description et mise en œuvre ». Edition octobre 2010.

[3] E. Alibi&S.Jawadi. 'Conception et réalisation d'un enregistreur de données', Mémoire de licence, Université Virtuelle de Tunis Edition 2010/2011.

[4] W. HENI & I. Hmaied. 'Mise en place d'une plateforme de télécommande des équipements électrique à distance « Smart House »', Mémoire de Licence, Université Virtuelle de Tunis. 2010/2011

[5] M. Taboui. 'Commande de l'éclairage public et mesure de la température à base de PIC18F455', Mémoire de licence ISET de Bejaia. 2009/2010.

[6] OGUIC, P. (2004). Moteurs pas-à-pas et PC ; 2ème Edition. Dunod. Paris

[7] Wildi, T.. & Sybille, G.(2000). Electrotechnique. De Boeck Supérieur

[8] Hassani Rima 'conception et réalisation de contrôleur de température à base d'un microcontrôleur Pic 16f877', Mémoire de master système embarqué de ENST 2013/2014

[9] Hafaid Imen 'étude physiquo-chimique de capteurs à base de nanomatériaux des applications biomédicales', Thèse de doctorat nanomatériau, France,2009

[10] Rahmoune Mohamed 'système d'alerte de fuite de gaz à base d'Arduino', Mémoire de master système embarqué de ENST 2013/2014

[11] Remita Tarek 'conception d'un système d'asservissenment standard (température/pression) à base d'un Pic16f877A

Webographie

[12]https://www.leselectroniciens.com/site/default/files/cours/gpa668_pas_a_pas_e2011.pdf

[13] //www-lgis-univ-llille1.fr/bonnet/GSI/capteurs2_GSI.pdf

[14] https://philipp.berger2.frer.fr/automatique/cours/cpt/les_capteurs.htm

[15] https://www.researchgate.net/publication/definition-d'une/procedure/d'etalonnage

[16] http://www.mikroe.com/chapters/view/17/chapter-4-examples/

[17]http://www.technologuepro.com/TP-miniprojet-electronique/miniprojet-2 MICROCONTROLEURS-PIC-MICROSHIP.pdf

[18] http://www.math.univ.Toulous.fr/cnegules/article/interp.pdf

Abréviations

- **A:** analogique
- **ADC:** Analog-to-Digital Converter
- **ASCCII:** American Standard Code for Information Interchange
- **D:** Digital
- **E** Enable
- **EEPROM:** Electrically Erasable Programmable Read Only Memory
- **GND :** Ground
- **PCFG:** Port Configuration Function Gate
- **PIC :** Programmable Integrated Circuit7
- **PWM :** Pulse Width Modulation
- **LCD:** Liquide Crystal Display
- **LED:** Light-Emitting Diode
- **MCLR:** Master Clear
- **R:** Resistance
- **RAM** Read access memory
- **RISC**: Reduced Instruction Construction Set
- **R/W :** Read/ Write
- **USART :** Universal Synchronous/Asynchronous Receiver Transmitter
- **$V_{reff:}$** Tension de référence
- **WDT :** Watchdog Timer