République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Abderrahmane MIRA- Bejaia

Faculté de Technologie

Département de Génie Electrique

Mémoire de fin de cycle

En vue de l'obtention du diplôme MASTER en Electronique

Option : *Instrumentation*

Thème

Réalisation d'une montre degitale à base d'un PIC16F84A

Présenté par :

Boudjaoui Halim

Encadré par :

Mr Hanfoug .S

Année universitaire : *2017/2018*

Remerciement

Je remercie ***Dieu*** *de nous avoir donné la force et le courage pour réaliser Ce modeste travail.*

Je tiens à exprimer nos profondes gratitudes envers notre encadreur ***Dr. HANFOUG****, pour le sujet d'actualité qu'il nous a proposé et les précieux conseils qu'il n'a cessé de nous prodiguer.*

Je tiens également à remercier tous ***les membres du jury*** *de soutenance pour leurs intérêts envers notre sujet et leurs contributions pour enrichir notre thématique.*

Je témoigne reconnaissances envers nos ***enseignants*** *et cela de l'école* ***primaire*** *jusqu'à* ***l'université*** *; auxquels nous devons notre formation grâce à l'enthousiasme qu'ils ont su nous communiquer en tant qu'étudiants lors de notre cursus d'étude, Sans oublier tous ceux et celles qui ont contribué de prêt ou de loin pour l'élaboration de ce travail.*

Enfin, je ne pourrai terminer ces remerciements sans remercier ***mes parents*** *pour leurs aides précieux, compréhensions, encouragements et soutiens, qu'ils nous ont apportés durant nos longues années études.*

Dédicaces

Au nom d'ALLAH, le tout puissant, le miséricordieux.

D'abord, je remercie le bon Dieu qui m'a donné le courage pour Arriver à la fin de mes études.

Je dédie ce modeste travail à :

- ***A Mes parents :***

Grâce à leurs tendres encouragements et leurs grands sacrifices durant toute la période d'étude, ils nous ont créé le climat affectueux et propice à la poursuite de nos études.

Nous prions le bon dieu de veiller sur eux et de les bénir, et les acquérir dans son vaste paradis, on espérant qu'ils seront toujours fiers de nous.

- ***A Mes sœurs, Amis et Amies :***

Pour le soutien qu'ils n'ont pas cesser de nous porter.

- ***A nos collègues :***

Ils vont trouver ici le témoignage d'une fidélité et amitié infinie.

- ***A la famille :*** *Boudjaoui*

La liste des figures

Chapitre I :

Chapitre II :

Chapitre III :

La liste des tableaux

Chapitre I :

Chapitre III :

Sommaire

Sommaire

Sommaire

Introduction générale

Au fil du temps, l'électronique connait une grande évolution et par conséquent de nouveaux circuits développés ont fait leur apparition. Les circuits intégrés foisonnent dans notre environnement. Synthétiseurs vocaux, lecteurs multimédia, industrie automobile, électroménager... tous les secteurs sont concernés aujourd'hui par le développement de l'électronique numérique.

Une montre numérique ou montre à affichage numérique est une montre dépourvue d'aiguilles. Souvent appelée aussi par abus de langage montre digitale, elle affiche l'heure sous forme de chiffres, par différents moyens.

Une montre digitale à base du PIC 16F84A est un système qui associe un ensemble de fonctions gérées par un microcontrôleur de la famille PIC16F.

L'objectif de notre projet est d'étudier et réaliser une montre digitale autour d'un PIC16F84A à affichage numérique via des écrans LCD permettant d'indiquer l'heure.

Apres une brève introduction notre projet s'organise de la manière suivante :

Le premier chapitre : intitulé << présentation et Description des outils du projet >> donne un aperçu sur des différents blocs nécessaires à la réalisation pratique de notre carte électronique, ainsi que la description des logiciels utilisés pour la simulation.

Le deuxième chapitre : intitulé << Mise en ouvre du microcontrôleur pic 16F84A >> est consacré à la description du microcontrôleur PIC 16F84A, qui est l'unité de commande et de traitement de notre système.

Le troisième chapitre : intitulé << Conception et Réalisation pratique >> concerne la conception et la réalisation pratique des différents blocs de notre carte électronique.

Nous terminons par une conclusion générale et des perspectives.

I.1 Introduction

Avec les progrès de la technologie, plus particulièrement de l'électronique, il n'est plus rare de nos jours de voir les appareils usuels remplacés par des appareils électroniques, nous pouvons penser entre autres aux montres digitales au tour d'un microcontrôleur.

Pour réaliser notre montre digitale, nous avons utilisé de nouveaux logiciels de simulation ; microcontrôleur et composants électroniques.

I.2 Description matériel :

I.2.1 Afficheur LCD :

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

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 rétro éclairage 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 250mA).

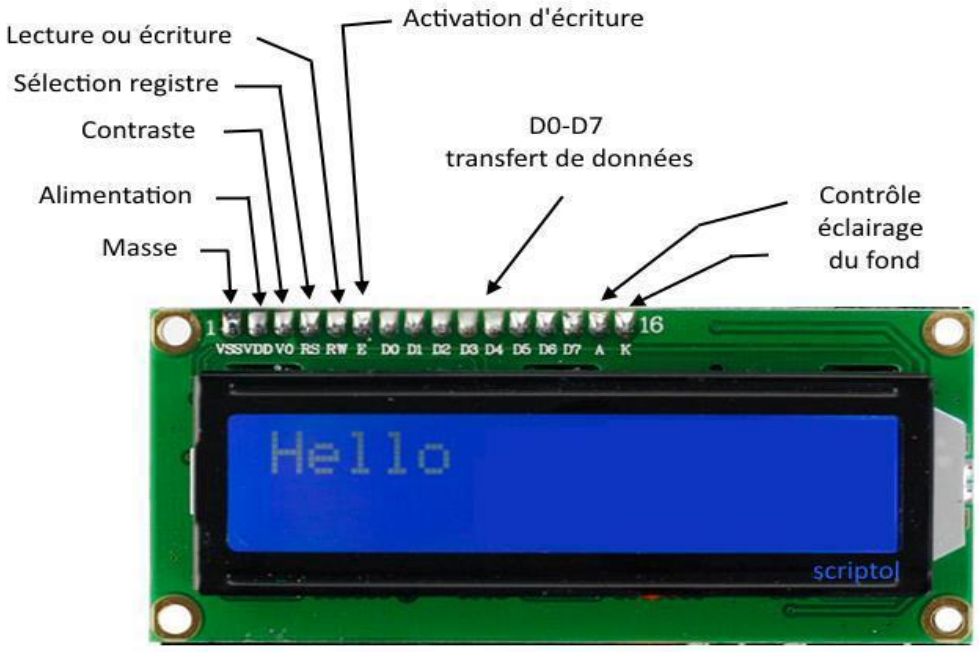


Figure I.1: Afficheur à cristaux liquide LCD

Les afficheurs LCD sont devenus incontournables dans toutes applications qui demandent la visualisation de paramètres.

Auparavant onéreux et difficiles à mettre en œuvre, ils sont maintenant bon marché et l'interface parallèle, au standard Hitachi, permet un pilotage facile.

On rencontre aussi de plus en plus d'afficheur pilotable avec un port série ou I2C. Les afficheurs LCD se ressemblent tous, à part le nombre de lignes et le nombre de colonnes, le fonctionnement et le brochage est standard et identique. Un des points intéressant est de pouvoir contrôler l'afficheur en mode 8bits ou en mode 4bits.

I.2.1.1 Définition:

Un cristal liquide est produit de la chimie organique [**1**], qui possède les propriétés optiques des cristaux solides alors qu'il est lui-même liquide.

Ses molécules ont la forme de cigares susceptible de s'orienter très rapidement dans le sens de tous champs électriques qu'on lui applique. En l'absence de ce dernier, les molécules s'orientent aléatoirement dans toutes les directions.

I.2.1.2 Présentation :

L'afficheur LCD utilise la polarisation de la lumière [**1**], grâce à des filtres polarisants, et à la biréfringence de certains cristaux liquides en phase nématique (phase intermédiaire entre liquide et solide), dont on peut faire varier l'orientation en fonction du champ électrique. Du point de vue optique, l'afficheur à cristaux liquides est un dispositif passif (il n'émet pas de la lumière) dont la transparence varie ; il doit donc être éclairé.

D'abord disponible en monochrome et en petite taille, il est utilisé dans les calculatrices et les montres, du fait de sa faible consommation électrique. Il permet actuellement d'afficher en couleurs dans des dimensions dépassant le mètre de diagonale. Il a pu remplacer le tube cathodique dans la plupart des applications, sauf en très haute définition, lorsque la palette de couleurs doit être précise et fidèle.

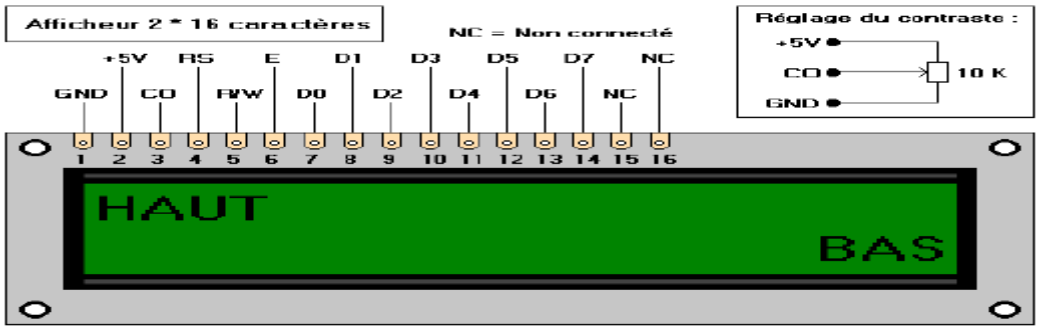


Figure I.2 : Un afficheur LCD 2*16 caractères.

I.2.1.2.1 Les bases des afficheurs LCD :

LCD est l'acronyme de **Liquid Crystal Display** (en anglais), ce qui signifie en français **écran à cristaux liquides [1]**. Par opposition à un afficheur à LED (comme les afficheurs 7 segments par exemple) où il suffit d'allumer une LED pour créer des caractères, l'affichage d'un message textuel sur un afficheur LCD n'est jamais direct. Il faut envoyer une série de commandes à l'afficheur, qui les interprète et qui réalise en fonction certaines actions dont l'affichage des caractères.

On distingue 2 types de commandes : les **instructions** (pour configurer l'afficheur) et les **données** (pour afficher un caractère à partir de son code ASCII).

Le protocole d'envoi des commandes à l'afficheur est très précis et doit être respecté si on veut que la réaction de l'afficheur soit le résultat attendu : afficher un message.

Un afficheur LCD contient :

- une entrée de contrôle **RS** (Register Select)
- une autre entrée **RW**
- une entrée de validation **E** (Enable)
- 8 entrées de données **D0** à **D7**
- 3 entrées d'alimentation **VSS, VDD** et **VEE.**

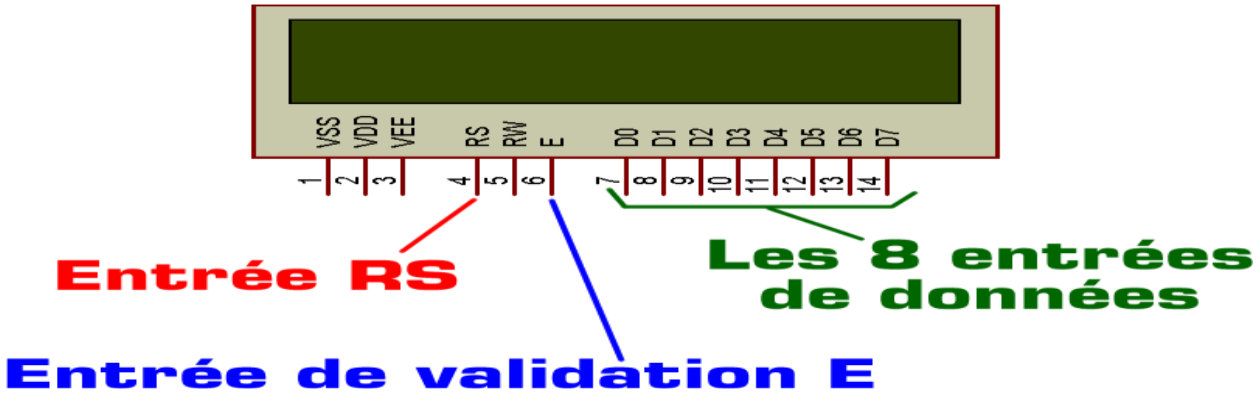


Figure I.3 : Les Entrées d'un afficheur LCD

RS permet de préciser si la commande présente sur les entrées **D0** à **D7** est une **instruction** ou une **donnée**, et **E** permet de valider cette commande.

Une commande est une valeur numérique présente sur les entrées **D0** à **D7** et validée par une impulsion sur **E**.

Le protocole d'envoie des commandes précise la liste des instructions à envoyer pour configurer l'afficheur (**RS=0**) suivie des données à envoyer (**RS=1**).L'entrée **RW** sera mise à zéro(connectée à la masse) et sera inutilisée ici.

Les 3 entrées d'alimentation **VSS, VDD** et **VEE** n'ayant pas besoin d'être obligatoirement alimentées dans ISIS Proteus, elles resteront non connectées.

I.2.1.2.2 Le brochage d'un afficheur LCD :

L'afficheur LCD a 14 broches en standard et souvent 16, les broches 15 et 16 servent au rétro-éclairage (une option) [**1**].

Broche	Nom	Description
1	Vss	Masse
2	Vdd	Alimentation 5v
3	CO	Variables de 0 à 5v permet de modifier le contraste de l'afficheur
4	RS	Indique une commande ou une donnée à afficher (0: Commande / 1: Donnée)
5	R/W	Indique une écriture ou une lecture (0: Écriture / 1: Lecture)
6	E	Indique une validation (Le niveau Haut doit être maintenu 500µs)
7	D0	Bus de données bidirectionnel.
8	D1	
9	D2	
10	D3	
11	D4	
12	D5	
13	D6	
14	D7	
15	A	Anode rétroéclairage (+5V)
16	K	Cathode rétroéclairage (masse)

Tableau I.1: le brochage d'un afficheur LCD.

I.2.1.2.3 Principe de fonctionnement :

On envoie deux types d'information à l'afficheur [**1**]**.**

- ❖ les **commandes** qui permettent de l'initialiser : positionnement du curseur, effacement écran, etc. ;
- ❖ les **données** à afficher :

L'entrée **RS** permet de spécifier si on envoie une commande ou une donnée :

- ✓ RS=0: instruction (commande) ;
- ✓ RS=1: caractère (donnée).
- ✓ L'afficheur dispose d'une entrée **R/W** pour spécifier une lecture ou une écriture :
  - R/W=0: écriture vers l'afficheur.
  - R/W=1: lecture de l'afficheur.

Pour valider tous les échanges sur le bus de données (D7-D0) on utilise l'entrée **E** de l'afficheur. Un **front descendant** sur cette entrée valide la donnée.

En programmation, il faudra placer un court instant **E** à l'état haut puis à l'état bas. Il est possible d'utiliser l'afficheur LCD en mode 8 bits normal ou en mode 4 bits pour économiser les broches de son µcontrôleur par exemple, c'est assez pratique :

A / Mode 8 bits:

En mode 8 bits on place la donnée ou la commande sur le bus **D7** à **D0** et on valide avec **E.**

B /Mode 4 bits:

En mode 4 bits on place déjà les poids forts de la donnée ou la commande sur les bits de **D7** à **D4** et on valide une première fois avec **E**. Puis on va mettre le poids faible sur les bits de **D7** à **D4** et on valide une seconde fois avec **E**.

L'envoie ou la lecture d'un octet s'effectue donc en 2 temps dans ce mode

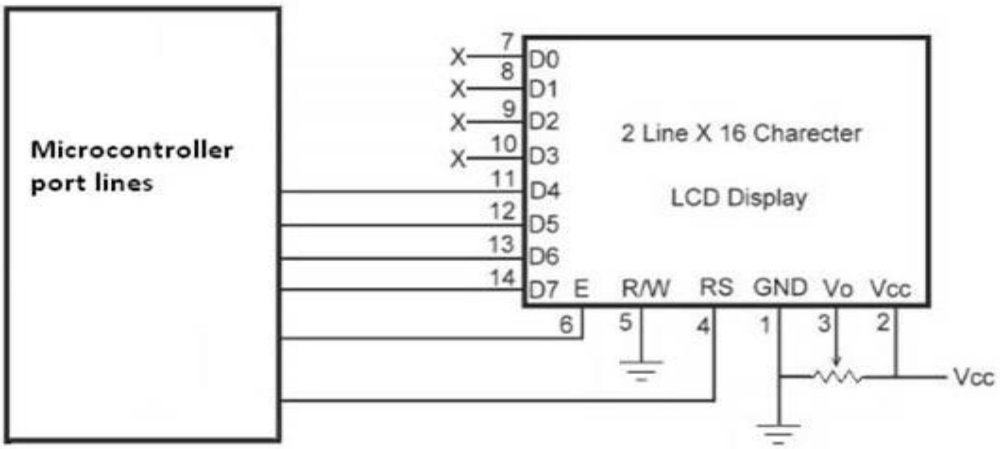


Figure I.4 : affichage en mode 4 bits

I.2.1.2.4 Les mémoires :

Les afficheurs LCD possèdent 2 types de mémoires [**1**] :

- ✓ La **DD RAM** qui mémorise les caractères affichés à l'écran ;
- ✓ La **CG RAM** qui contient la table des caractères affichables.

La DD RAM (Display Data RAM):

La DD RAM commence à l'adresse 0x00 et dans le cas d'un afficheur 16x2 lignes, elle termine à 0x4F. C'est une mémoire d'affichage dont l'adresse contient le caractère affiché à l'écran à une certaine position.

La première ligne commence en 0x00 jusqu'à 0x0F inclue.

La seconde ligne commence en 0x40 jusqu'en 0x4F inclue.

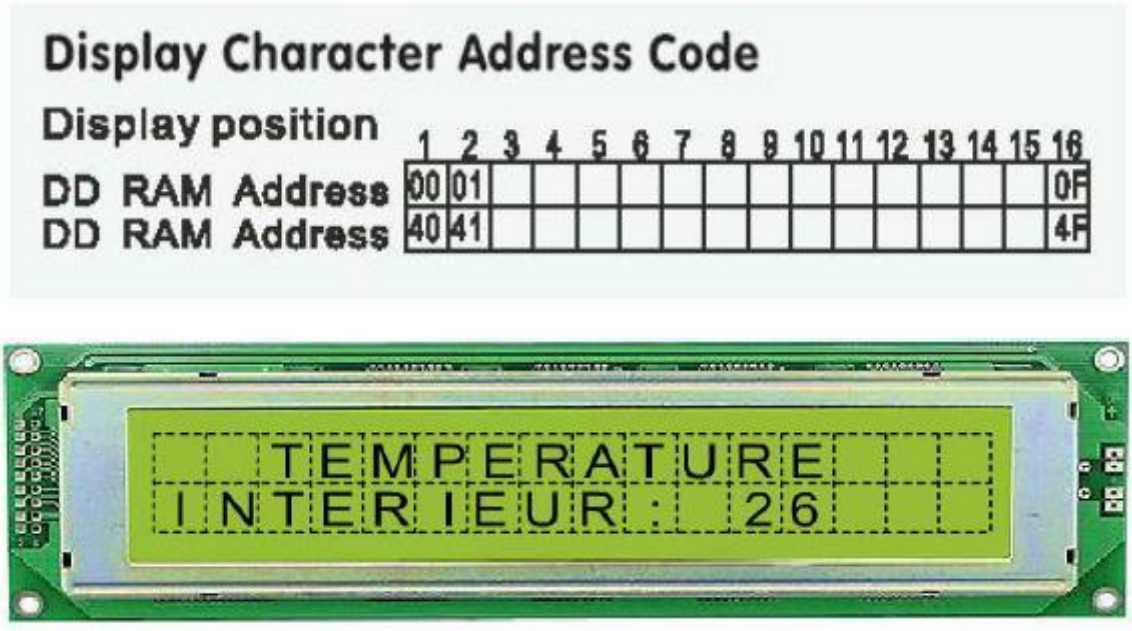


Figure I.5 : l'adresse des caractères dans l'afficheur LCD.

I.2.2 le pic 16F84A :

Le PIC est un élément indispensable dans ce projet **[2]**, il permet de contrôler et d'afficher le temps (jour, l'heure, ... etc.) sur l'afficheur LCD ou bien sur un afficheur 7segments

Le PIC utilisé dans le projet PIC16f84A

Le 16F84A est un circuit intégré de 18 broches, que l'on peut trouver dans un boîtier PDIP

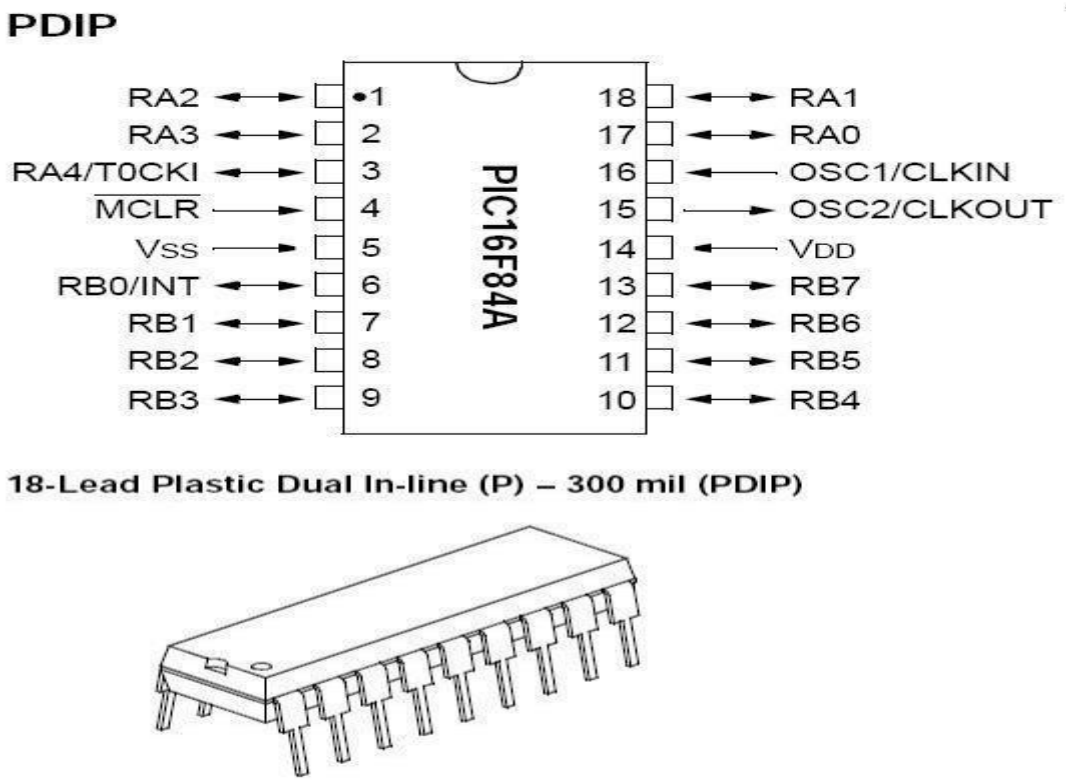


Figure I.6 : Brochage du PIC 16F84A

- V_{ss} et V_{DD} : broches d'alimentation (3 à 5,5 V).
- OSC1 et OSC2 : signaux d'horloges, ces broches peuvent recevoir un circuit RC ou un résonateur.
- CLKIN : peut être connectée à une horloge externe (0 à 4, 10 ou 20 MHz).
- MCLR : Reset (Master Clear).
- RA0, ..., RA4 : 5 entrées/sorties du port A.
- RB0, ..., RB7 : 8 entrées/sorties du port B.
- T0CKI : Entrée d'horloge externe du Timer TMR0.
- INT : entrée d'interruption externe.

I.3 Description logiciels

I.3.1 Le logiciel ISIS Proteus :

I.3.1.1 Présentation générale :

Proteus Professional est une suite logicielle destinée à l'électronique **[3].** Développé par la société **Labcenter Electronics**, les logiciels incluent dans **Proteus Professional** permettent la CAO (Construction Assistée par Ordinateur) dans le domaine électronique. Deux logiciels principaux composent cette suite logicielle: (ISIS, ARES, PROSPICE) et VSM.

Cette suite logicielle est très connue dans le domaine de l'électronique. De nombreuses entreprises et organismes de formation (incluant lycée et université) utilisent cette suite logicielle. Outre la popularité de l'outil, **Proteus Professional** possède d'autres avantages :

• Pack contenant des logiciels facile et rapide à comprendre et utiliser

• Le support technique est performant

• L'outil de création de prototype virtuel permet de réduire les coûts matériel et logiciel lors de la conception d'un projet

1. ISIS

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

2. ARES

Le logiciel ARES est un outil d'édition et de routage qui complète parfaitement ISIS. Un schéma électrique réalisé sur ISIS peut alors être importé facilement sur ARES pour réaliser le PCB (**Printed circuit board)** de la carte électronique. Bien que l'édition d'un circuit imprimé soit plus efficiente lorsqu'elle est réalisée manuellement, ce logiciel permet de placer automatiquement les composants et de réaliser le routage automatiquement.

I.3.1.2 L'écran Isis :

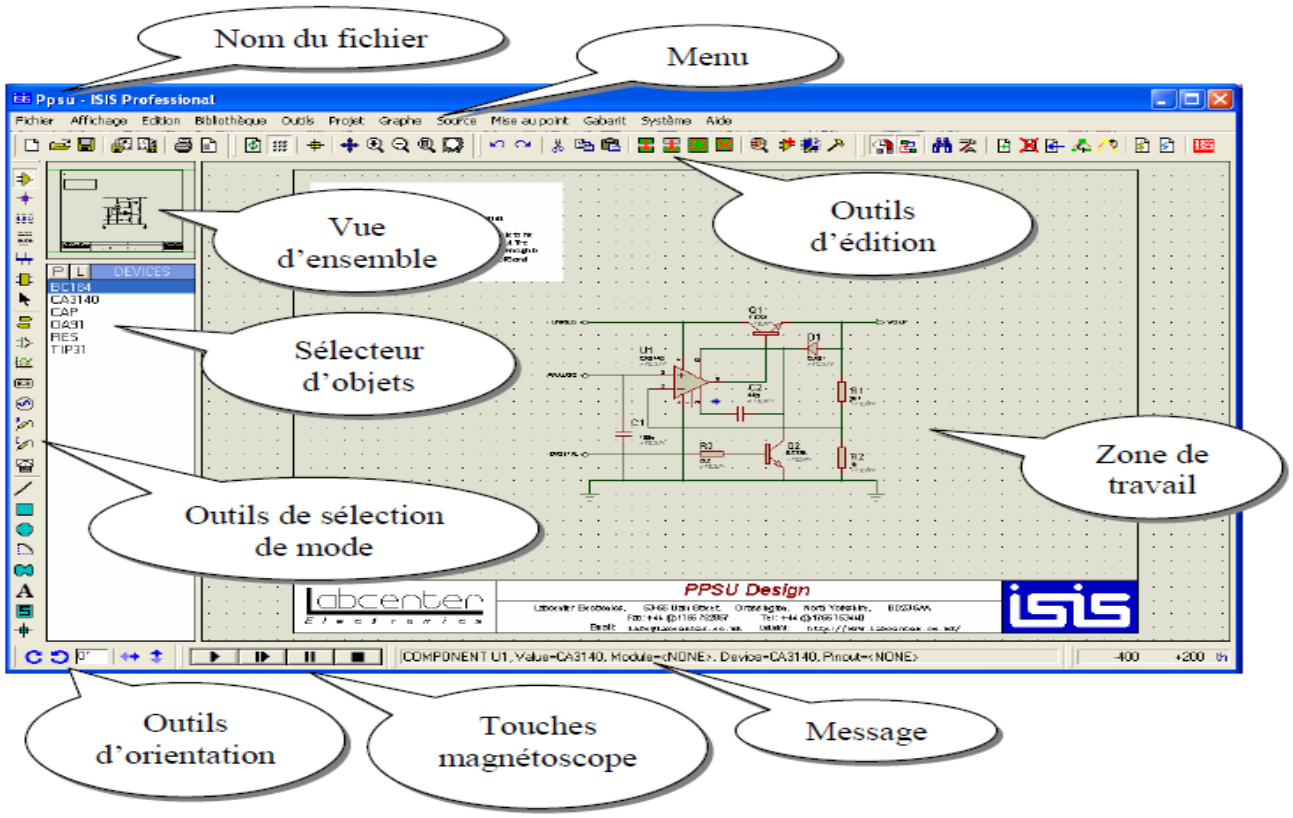


Figure I.7: l'Ecran Isis

I.3.1.3 Les barres d'outils :

❖ **Les outils d'éditions :**

- **Commandes d'édition**

Annuler la dernière opération	Rotation / symétrie des objets sélectionnés
Rétablir la dernière opération annulée	Effacement des objets sélectionnés
Couper	Prendre ou mettre à jour un composant ou un symbole
Copier	Regrouper la sélection en un composant de la bibliothèque
Coller	Outil d'affectation de boîtier
Copie des objets sélectionnés	Décomposer l'objet sélectionné
Déplacement des objets sélectionnés	

- **Outils de projet**

Accrochage en temps réel	Aller à une feuille spécifique
Autorouteur de fils	Aller vers une feuille enfant
Recherche de composant	Quitter la feuille courante pour remonter à la feuille parent
Gestion des propriétés	Rapport de liste du matériel
Nouvelle feuille racine	Contrôle des règles électriques
Supprimer la feuille courante	Générer la netliste et passer sous ARES

❖ **Les outils de sélection de mode :**

- **Modes principaux** 

 Composant
 Point de connexion
 Label
 Texte
 Bus
 Sous-circuit
 Edition

- **Gadgets** 

 Terminaisons
 Broches des composants
 Graphes pour affichage de la simulation
 Enregistreur
 Générateurs
 Sonde de tension
 Sonde de courant
 Instruments virtuels

❖ **Les outils d'orientation** : 

 Rotation ¼ de tour sens horaire
 Symétrie horizontale
 Rotation ¼ de tour sens trigonométrique
 Symétrie verticale


I.3.2 Le compilateur mikroC PRO [4]:

Le langage mikroC pour PIC a trouvé une large application pour le développement de systèmes embarqués sur la base de microcontrôleur. Il assure une combinaison de l'environnement de programmation avancée *IDE*(Integrated Development Environment) , et d'un vaste ensemble de bibliothèques pour le matériel, de la documentation complète et d'un grand nombre des exemples.

Le compilateur *mikroC* pour PIC bénéficie d'une prise en main très intuitive et d'une ergonomie sans faille. Ses très nombreux outils intégrés (mode simulateur, terminal de communication Ethernet, terminal de communication USB, gestionnaire pour afficheurs 7 segments, analyseur statistique, correcteur d'erreur, explorateur de code, mode Débug ICD...) associé à sa capacité à pouvoir gérer la plupart des périphériques rencontrés dans l'industrie (Bus I2C™, 1Wire™, SPI™, RS485, Bus CAN™, USB, gestion de cartes compact Flash et SD™/MMC™, génération de signaux PWM, afficheurs LCD alphanumériques et graphiques, afficheurs LEDs à 7 segments, etc...) en font un outil de développement incontournable pour les systèmes embarqués, sans aucun compromis entre la performance et la facilité de débogage.

I.3.2.1 Compilateur MicroC PRO pour PIC [4]:

La nouvelle version appelée mikroC PRO dispose de très nombreuses améliorations du compilateur mikroC : nouvelles variables utilisables, nouvelle interface IDE, amélioration des performances du linker et de l'optimisateur, cycle de compilation plus rapide, code machine généré plus compact (jusqu'à 40% suivant les cas), nouveaux PIC supportés, environnement de développement encore plus ergonomique, nouveaux exemples d'applications, etc...

- Dans la suite nous utiliserons le compilateur mikroC PRO v.5.6.1
- La simulation des applications de programmation nous réalisons à l'aide Logiciel PROTEUS.

I.3.3.2 Création d'un nouveau projet [4]:

Le mikroC PRO pour PIC organise des applications dans des projets, composé d'un seul fichier de projet (extension. mcppi) et un ou plusieurs fichiers sources (extension). Les fichiers source peuvent être compilés que si elles font partie d'un projet. Le fichier projet contient les informations suivantes :

- Nom du projet et une description facultative
- Composant cible
- Option du composant
- Fréquence d'horloge du composant
- La liste des fichiers source du projet avec les chemins
- Fichiers d'image
- Fichiers binaires (* mcl.)
- D'autres fichiers

La meilleure façon de créer un projet c'est à l'aide de l'Assistant Nouveau projet (menu Project> New Project) ou en cliquant sur l'icône Nouveau projet à partir de la barre d'outils du projet.

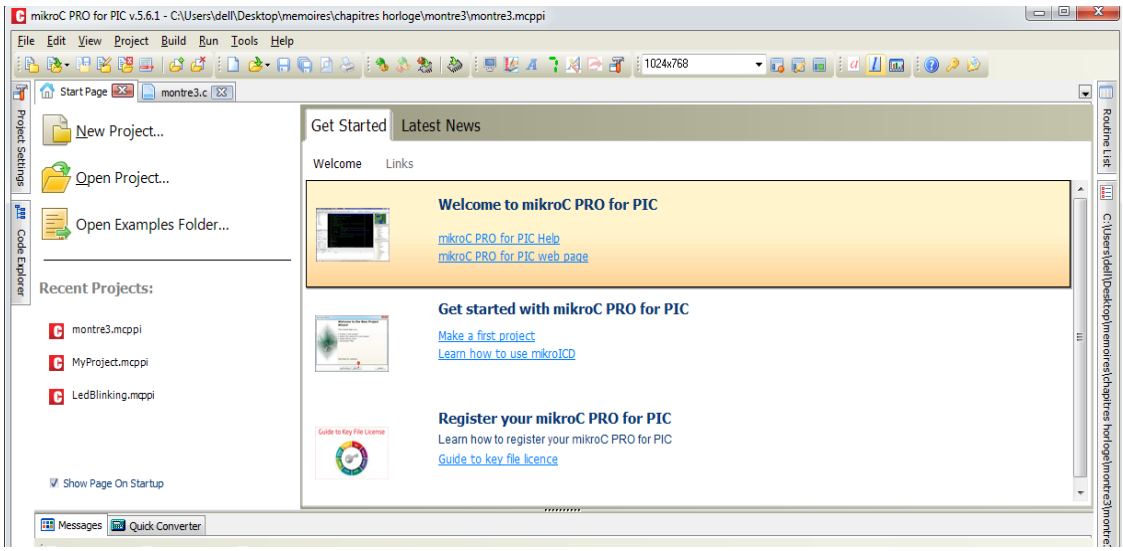


Figure I.8 : Creation d'un nouveau programme .

Les nouvelles étapes de l'Assistant de projet :

Commencez à créer votre nouveau projet, en cliquant sur le bouton **Next** :

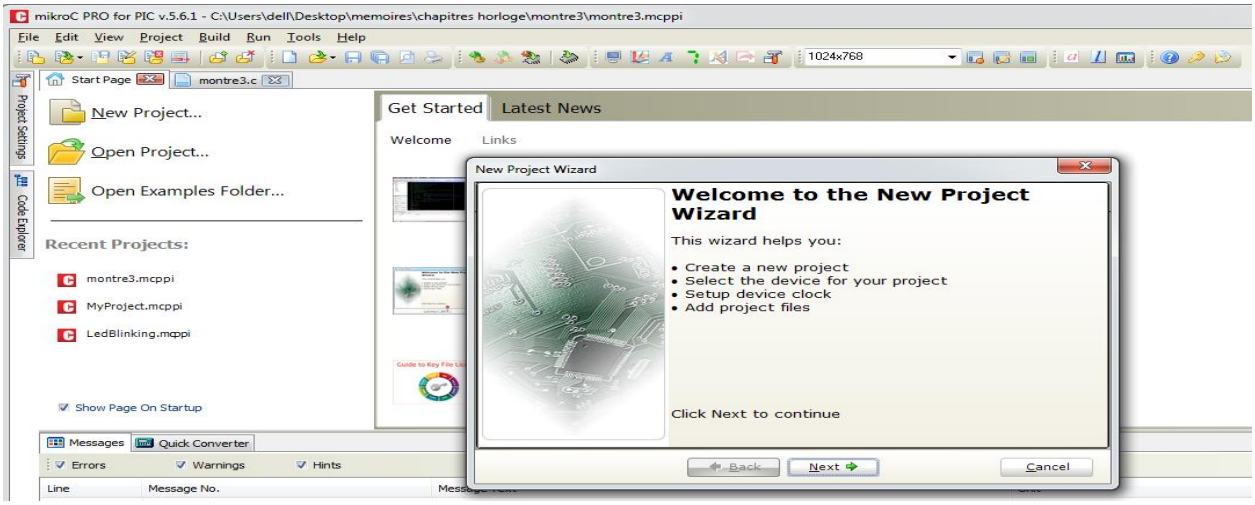


Figure I.9 : création d'un nouveau projet

Premiere étape : Sélectionnez le périphérique dans le périphérique dans la liste déroulante et Saisir la valeur de fréquence de l'oscillateur

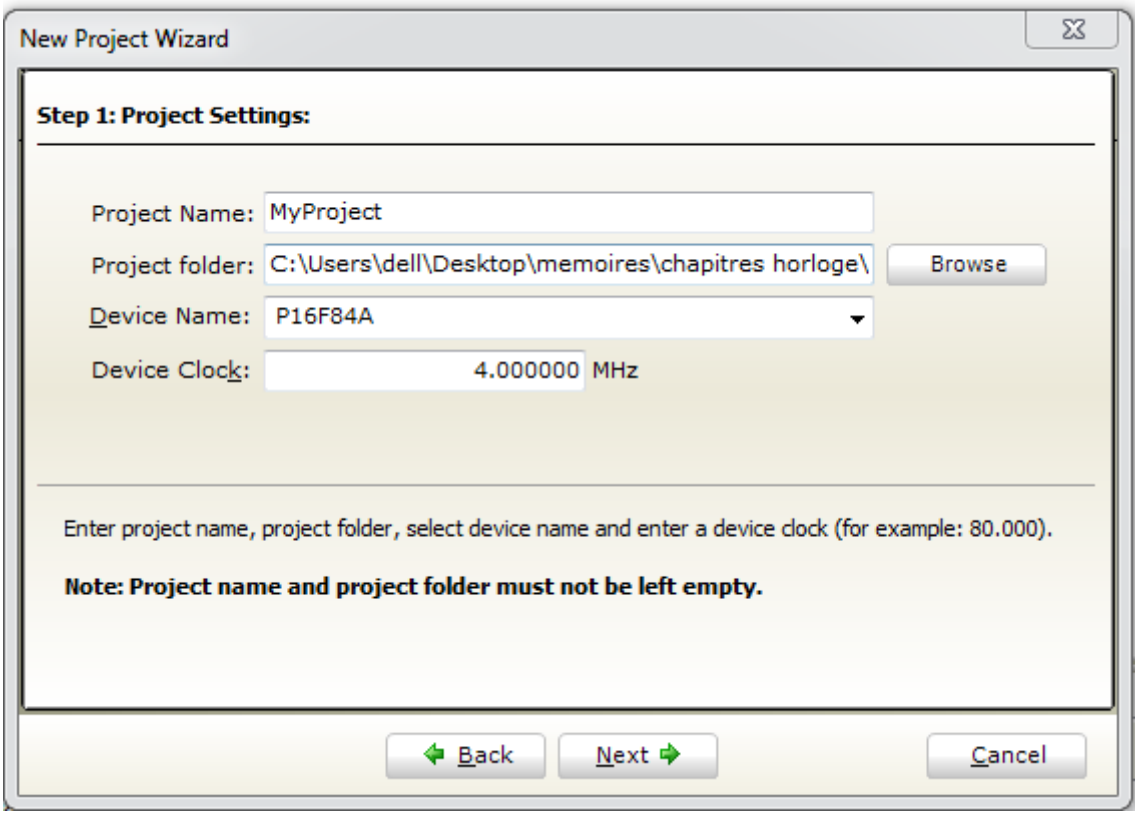


Figure I.10 : fréquence d'oscillateur

Deuxième étape : Spécifiez l'emplacement où votre projet sera enregistré.

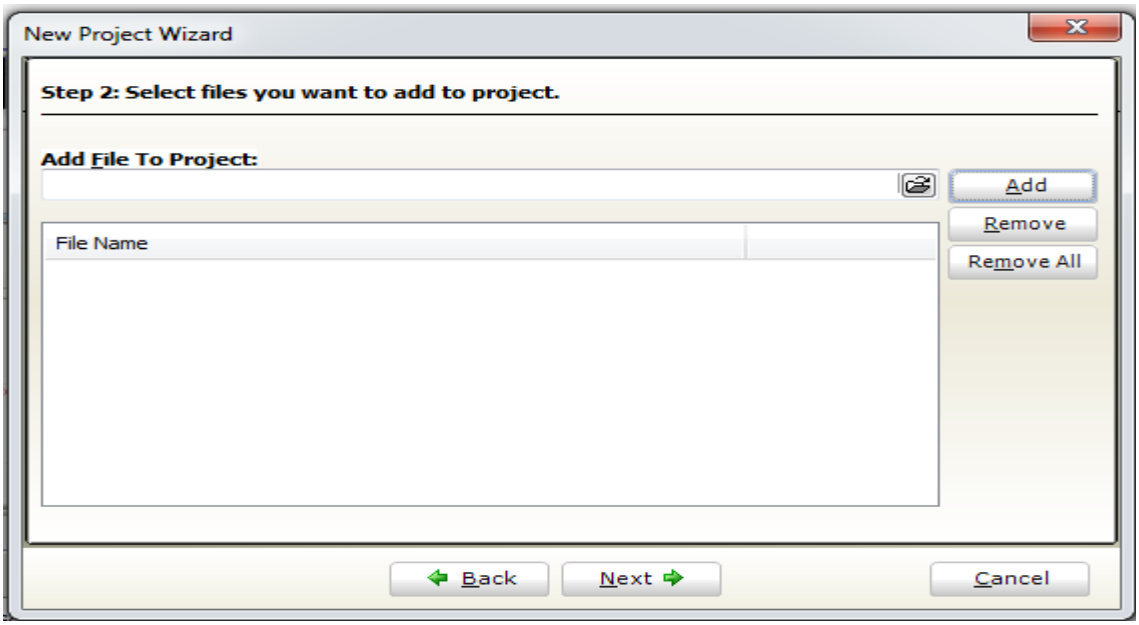


Figure I.11 : L'emplacement du projet

Troisième étape : Cliquez sur Finish pour créer votre nouveau projet. À ce stade, une nouvelle fenêtre vide s'affiche afin de saisir votre programme.

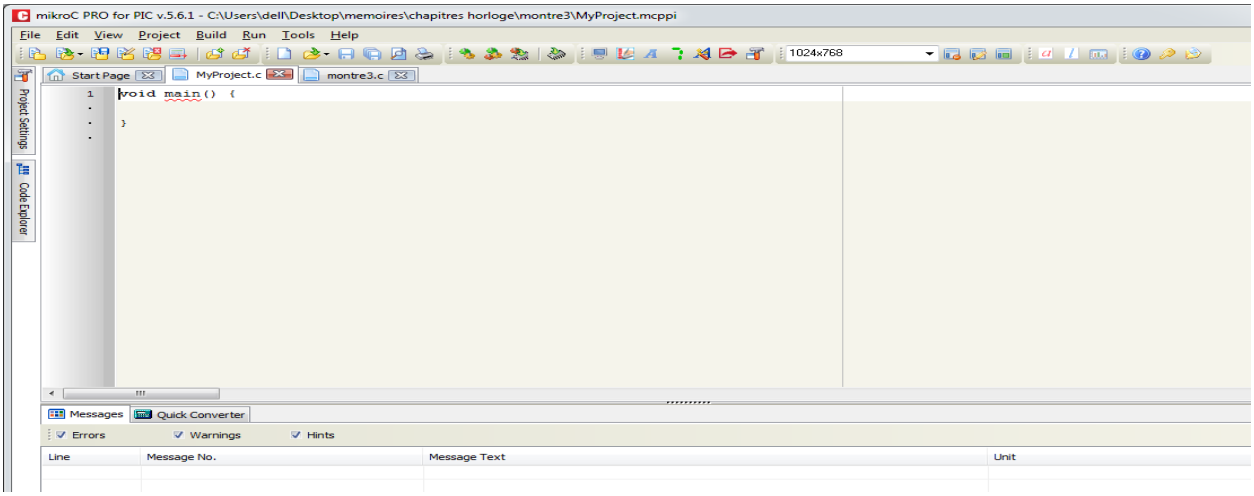


Figure I.12 : fenêtre de la saisie du programme

I.3.2.3 Compilation [4]:

Lorsque vous avez créé le projet et écrit le code source, il est temps de le compiler. Sélectionnez **Project Build** à partir du menu déroulant ou cliquez sur l'icône **Build** dans la barre d'outils du projet. Si plus d'un projet est ouvert, vous pouvez compiler tous ouverts projets en sélectionnant Project > Build All dans le menu déroulant, ou cliquez sur l'icône de la barre d'outils du projet.Barre de progression s'affiche pour vous informer sur l'état de la compilation.

S'il y a quelques erreurs, vous en serez informé dans la fenêtre d'erreur **(fig. I.13).**

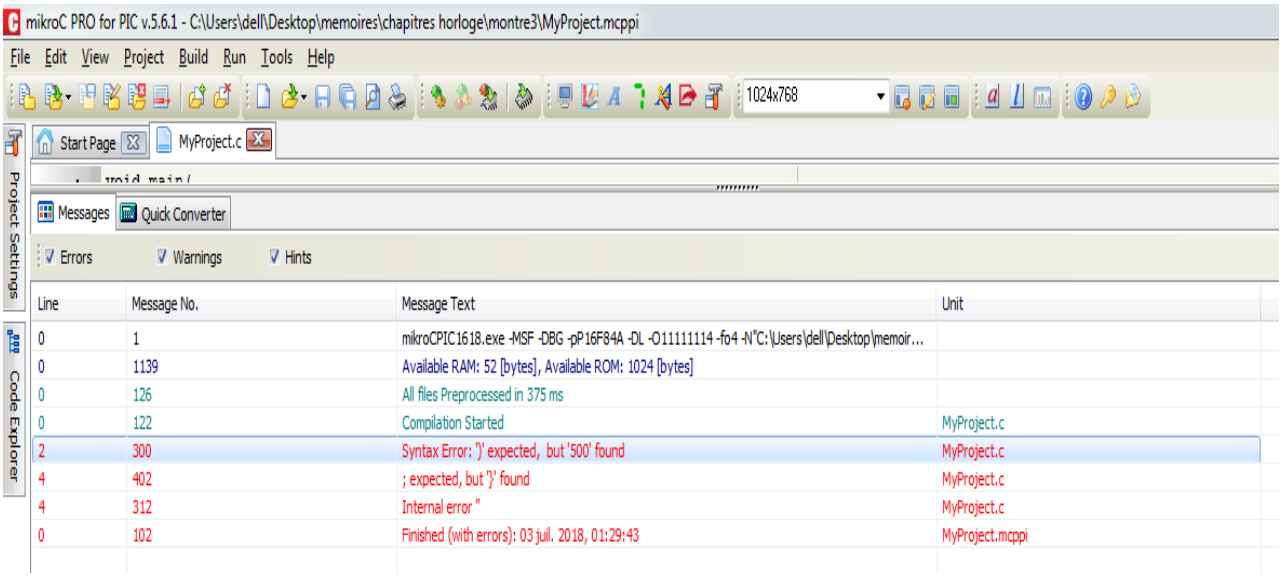


Figure I.13:Avertissement des erreurs.

Après la compilation réussie, le compilateur mikroC PRO pour PIC génère des fichiers de sortie dans le dossier du projet (dossier qui contient le fichier projet. mcppi). Les fichiers de sortie sont résumés dans le tableau ci-dessous:

Format	Description	Type de fichier
Intel HEX	Code hexadécimal dans le format Intel. Fichier est utilisé pour programmer PIC	.hex
Binary	Fichier compilé pour la bibliothèque mikroC. Les distributions binaires sont des routines qui susceptibles d'être inscrits dans d'autres projets.	.mcl
List File	L'mage globale de l'allocation de mémoire du PIC pour : adresses d'instructions, les registres et les étiquettes du programme.	.lst
Assembler File	Le fichier en assembleur avec des noms symboliques, obtenus à partir de liste des fichiers.	.asm

Tableau I.2 : compilation (mikroC)

I.3.2.4 Fichiers Sources [4]:

- **Création d'un nouveau fichier source :**

Pour créer un nouveau fichier source, procédez comme suit : Sélectionne New depuis le menu File ou pressez Ctrl+N ou cliquez sur l'icône (New File) depuis la barre d'outils File. Une nouvelle fenêtre s'ouvrira dans laquelle vous pourrez saisir votre code source. Sélectionnez alors Save depuis le menu File ou pressez Ctrl+S ou cliquez sur l'icône (Save File) depuis la barre d'outils (File) et nommez ce dernier comme vous le voulez.

- **Ouvrir un fichier existant :**

Sélectionnez Open depuis le menu File ou pressez Ctrl+Q ou cliquez sur l'icône (Open File) depuis la barre d'outils (File). Dans la boite de dialogue Ouvrir, sélectionnez alors l'emplacement du fichier que vous désirez ouvrir et cliquez sur le bouton Ouvrir. Le fichier sélectionné s'affiche dans sa propre fenêtre. Si le fichier sélectionné est déjà ouvert, sa fenêtre d'édition (Editor) deviendra alors active.

- **Sauvegarder un fichier :**

Assurez-vous que la fenêtre contenant le fichier que vous voulez sauvegarder est active. Sélectionnez ensuite Save depuis le menu File ou pressez Ctrl+S ou cliquez sur l'icône Save Fail de la barre d'outils File.

- **Sauvegarder un fichier sous un nom différent :**

Assurez-vous que la fenêtre contenant le fichier que vous voulez sauvegarder est active. Sélectionnez ensuite Save As depuis le menu File. Une boîte de dialogue 'Save [nom du fichier] .cas' s'affichera alors. Dans cette boîte, sélectionnez l'emplacement dans lequel vous voulez sauvegarder le fichier. Dans le champ *Nom du fichier,* modifiez le nom du fichier.

- **Règles générale d'écriture en microC :**
  - Les instructions propres au langage microC doivent être écrites en minuscule (**void** main (void)).
  - Les instructions particulières aux microcontrôleurs doivent être écrites en majuscule (TRISB).
  - Les retours à la ligne et les espaces servent uniquement à aérer le code
  - Toutes instructions ou actions se terminent par un point virgule « ; ».

❖ **Début et fin d'un programme :**

En langage microC, un programme commence avec les mots-clés void main ()
Après cela, une accolade ouvrante est utilisée pour indiquer le début du corps de programme .Le programme se termine par une accolade fermante.
le programme a la structure suivante :

```
void main()
{
//Votre code ici

}
```

❖ **Fin d'une instruction :**

Le point virgule « ; » indique la fin d'une instruction, sinon une erreur du compilateur sera générée.

```
j = 5; // correcte
j = 5 // erreur
```

❖ **Espaces blancs :**

Les espaces blancs sont des espaces, des flans, les tabulations et les caractères de nouvelle ligne. Le compilateur *microC* ne tient pas compte de tous les espaces blancs.
Ainsi, les trois séquences suivantes sont identiques :

```
int i; char j;
ou
int i;
char j;
ou
int i;
char j;
```

❖ **Instructions d'itération for, while, do, goto, continue et break :**

Les instructions d'itération nous permettent d'effectuer des boucles dans un programme, où une partie d'un code doit être répété un certain nombre de fois. Dans mikroC l'itération peut être effectuée de quatre façons. Nous se penchera sur chacun des exemples :

- ❖ Utilisation de for
- ❖ Utilisation de while
- ❖ Utilisation de do
- ❖ Utilisation de goto, continue et break

ADC Utilis**é** pour conversion analogique/numérique

PWM Utilisé pour les opérations avec le module de PWM

LCD_CLEAR Effacer l'affichage

LCD Utilisé pour le fonctionnement avec LCD standard

Delay_ms Crée le retard constant dans les unités de millisecondes

ADC	Utilisé pour conversion analogique/numérique
PWM	Utilisé pour les opérations avec le module de PWM
LCD_CLEAR	Effacer l'affichage
LCD	Utilisé pour le fonctionnement avec LCD standard
Delay_ms	Crée le retard constant dans les unités de millisecondes

Tableau I.3 : Instructions d'itération.

I.4 Conclusion :

Dans ce chapitre, nous avons projeté la lumière sur tous les outils utilisés dans notre projet en général, et spécifiquement les logiciels d'assimilation et de programmation.

Le chapitre suivant sera consacré à l'étude des généralités sur les microcontrôleurs en générale et spécifiquement le PIC16F84A.

II.1. L'architecture générale des microcontrôleurs [5] :

II.1.1 Introduction :

Un microcontrôleur est un circuit intégré qui rassemble aux éléments essentiels d'un ordinateur : processeur, mémoires (mémoire morte pour le programme, mémoire vive pour les données), unités de périphériques et interfaces d'entrées/sorties. Les microcontrôleurs se caractérisent par un plus haut degré d'intégration, une plus faible consommation électrique, une vitesse de fonctionnement plus faible et un coût réduit par rapport aux microprocesseurs polyvalents utilisés dans les ordinateurs personnels.

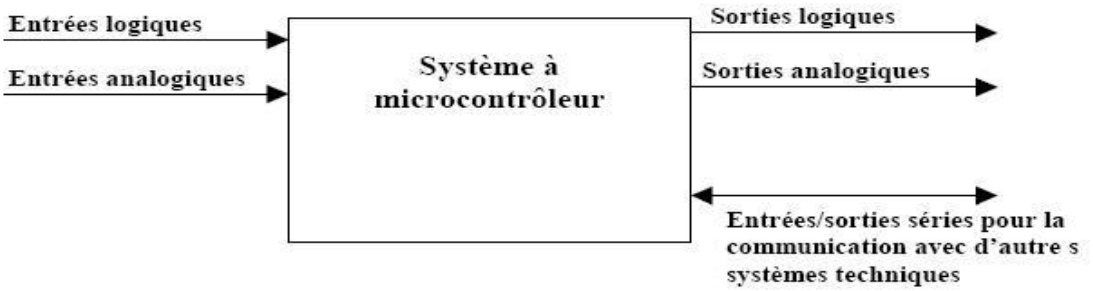
Par rapport à des systèmes électroniques à base de microprocesseurs et autres composants séparés, les microcontrôleurs permettent de diminuer la taille, la consommation électrique et le coût des produits. Ils ont ainsi permis de démocratiser l'utilisation de l'informatique dans un grand nombre de produits et de procédés.

Les microcontrôleurs sont fréquemment utilisés dans les systèmes embarqués, comme les contrôleurs des moteurs automobiles, les télécommandes, les appareils de bureau, l'électroménager, les jouets, la téléphonie mobile, etc.

II.1.2 Rôle d'un système à microcontrôleur:

Un système à microprocesseur permet :

- d'acquérir des entrées logiques et analogiques représentant l'état du système technique,
- d'interpréter, la signification de ces entrées,
- de calculer, mémoriser, récupérer des variables logicielles intermédiaires,
- de gérer le temps,
- d'agir sur des sorties logiques et analogiques en fonction des entrées et des calculs réalisés de manière à modifier le fonctionnement du système technique (commande moteur, affichage d'informations,…)
- de communiquer par des liaisons séries avec d'autres systèmes techniques et/ou un ordinateur



II.1.3 Schéma fonctionnel de l'organisation d'un système à microcontrôleur:

Ce schéma (Figure II.1) de la page suivante représente les différents périphériques internes à un microcontrôleur ainsi que le moyen utilisé pour communiquer avec des périphériques externes au microcontrôleur.

Attention, un même microcontrôleur ne possède pas forcément tous les périphériques représentés.

Le schéma fonctionnel de l'organisation fonctionnelle d'un microcontrôleur se trouve dans la page suivante :

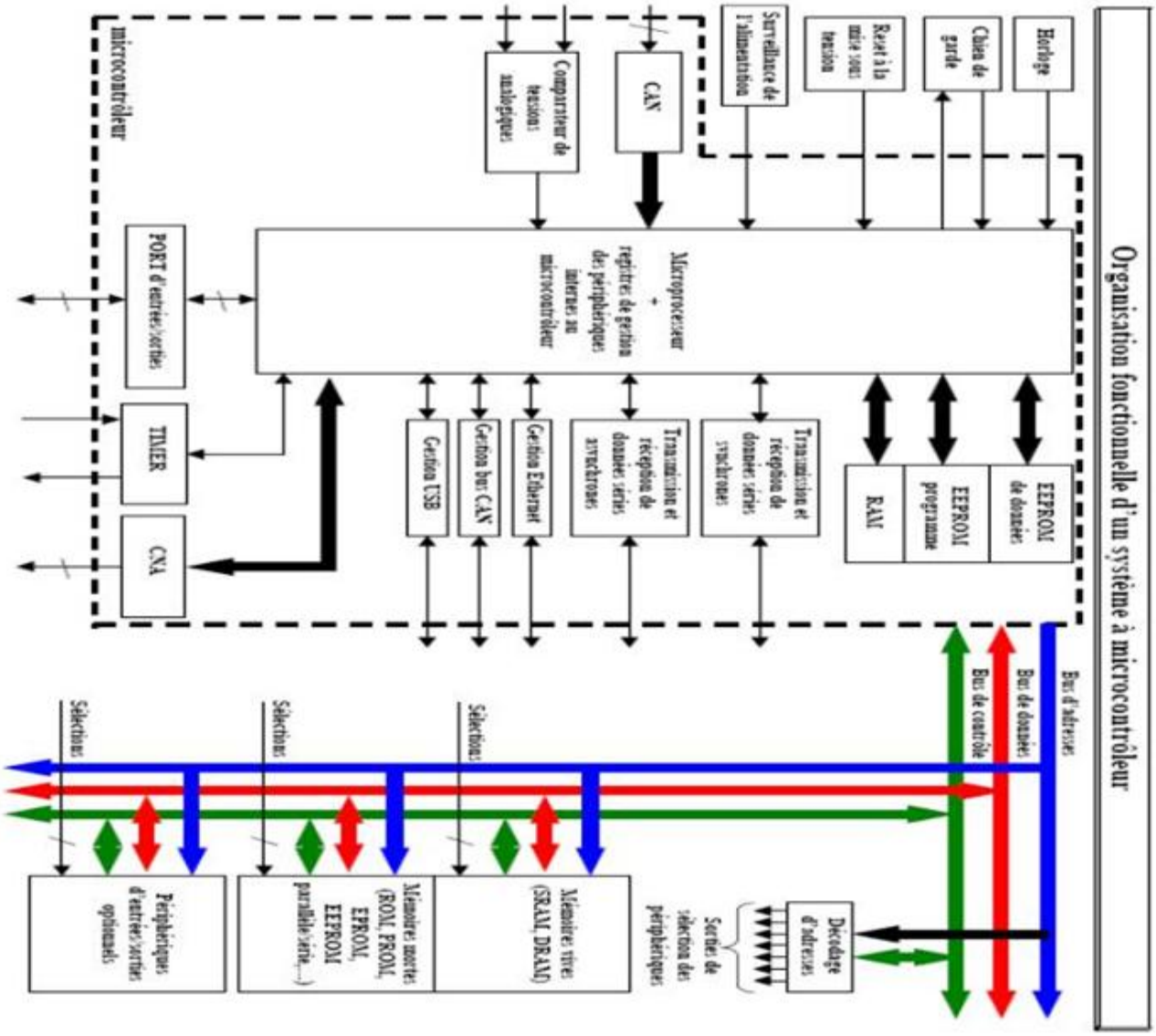


Figure II.1: Organisation fonctionnelle d'un système à microcontrôleur

II.1.4 Rôle des différents éléments composants de l'organisation fonctionnelle d'un système à microcontrôleur :

On appelle microcontrôleur un circuit intégré qui est constitué d'un microprocesseur associé à un ou plusieurs périphériques.

II.1.4.a Le microprocesseur [5] :

Toutes les informations transitées par le microprocesseur, il exécute un programme contenu en mémoire. Ce programme est constitué d'un ensemble d'instructions élémentaires codées, qui seront décodées puis exécutées au fur et à mesure par le microprocesseur**.**

Le microprocesseur est composé entre autre:

- ❖ D'un décodeur d'instruction qui va déterminer la tâche à exécuter.
- ❖ D'un séquenceur qui contrôle le fonctionnement de l'ensemble du microprocesseur.
- ❖ D'une Unité Arithmétique et Logique qui est chargée des opérations élémentaires (opérations logiques, addition, soustraction. comparaison, multiplication division,…).
- ❖ D'un compteur ordinal qui génère l'adresse de l'instruction qui devra être exécutée ou de la donnée qui devra être traitée.

Le microprocesseur utilisera un certain nombre de registres qui permettront de configurer et agir sur les différents périphériques.

II.1.4.b Les mémoires du microcontrôleur [5] :

Il existe différents types de mémoires :

- ✓ **EEPROM programme :** c'est une mémoire morte dans laquelle on va stocker le programme qui va gérer le fonctionnement du système technique.
- ✓ **EEPROM données :** c'est une mémoire vive dans laquelle; on va stocker les données devant être sauvegardées si le système technique est mis hors tension.
- ✓ **RAM :** mémoire vive dans laquelle ; on va stocker des données temporaires nécessaires à l'exécution du programme de gestion du système technique. Ces données ne seront plus disponibles si le système technique est mis hors tension.

II.1.4.c Le contrôle du microcontrôleur [5] :

II.1.4.c.1 L'horloge du microcontrôleur :

Elle va donner la référence temporelle au microprocesseur pour exécuter les instructions. L'horloge d'un microprocesseur est souvent réalisée grâce à un Quartz. Il existe certains microcontrôleurs qui ont la possibilité de sélectionner une horloge interne (sans composants externes) ce qui permet d'utiliser les broches de l'horloge pour d'autres périphériques.

II.1.4. c.2 Le chien de garde du microcontrôleur :

C'est une structure, qui peut être interne ou externe au microcontrôleur, qui permet vérifier le bon déroulement du programme.

Le microcontrôleur envoie des impulsions espacées de durées fixes au chien de garde. tant que les impulsions espacées de durées fixes arrivent au chien de garde, tout se passe bien. par contre dès que le chien de garde détecte l'absence d'une impulsion (le programme est bloqué), il produit une mise à zéro du programme de gestion du système technique de manière à débloquer le programme.

II.1.4.c.3 Le reset à la mise sous tension :

Tout microcontrôleur a besoin d'un temps minimum avant de pouvoir commencer à lancer le programme. Ce temps est donné par la documentation constructrice. Il faut par conséquent produire un signal de reset d'une durée supérieur à la mise sous tension.

II.1.4.c.4 Surveillance de l'alimentation:

C'est une structure qui permet de produire un reset du microcontrôleur si une chute de l'alimentation est détectée (problème sur le système technique).

II.1.5 Les périphériques d'un microcontrôleur [5] :

II.1.5.a Les CAN (Conversion Analogique Numérique) et CNA (Conversion Numérique Analogique) :

- **Les CAN :** ce périphérique se trouve souvent implémenté dans le microcontrôleur, il permet d'acquérir des grandeurs électriques de type analogique directement à partir d'une ou plusieurs broches du microcontrôleur la sortie est un nombre binaire.
- **Les CNA :** ce périphérique permet de produire une tension analogique à partir de mots numériques internes au microcontrôleur.

II.1.5.b Les ports d'entrées/sorties d'un microcontrôleur :

Ces périphériques sont indispensables au microcontrôleur ils permettent :

- ✓ d'acquérir les entrées de types logiques indiquant l'état du système technique,
- ✓ de produire des sorties de types logiques permettant de commander les périphériques du système techniques (afficheurs, moteurs, buzzer,…).

II.1.5.c La transmission de données séries asynchrone et synchrone : ces périphériques permettent la communication avec d'autres systèmes technique et/ou un PC

II.1.5.d La gestion Ethernet :

Les nouveaux microcontrôleurs disposent d'un périphérique permettant de gérer la liaison réseau de type Ethernet.

Ceci permet notamment de commander des systèmes techniques et/ou visualiser son état de fonctionnement à distance grâce à une page internet.

II.1.5.e La gestion de bus CAN :

Ce périphérique permet la communication série de données numériques avec des systèmes techniques dans des milieux perturbés notamment dans le domaine de l'automobile.

II.1.5.f La gestion de bus USB :

Ce périphérique permet de gérer le protocole de communication USB afin de connecter des appareils utilisant ce même protocole.

II.1.6 Les périphériques externes d'un microcontrôleur [5]:

Si les périphériques contenus dans le microcontrôleur ne sont pas suffisants, on peut rajouter certains périphériques externes.

Pour cela; il faut que le microcontrôleur dispose d'un bus d'adresses et d'un bus de données.

II.1.6.a Le décodage d'adresses :

Cette fonction permet d'affecter une plage d'adresses à un seul périphérique de manière à éviter les conflits de bus.

II.1.6.b Les bus du microcontrôleur :

Un bus est un ensemble de lignes, transportant des informations codées binaires.

Chacune de ces lignes est affectée d'un poids binaire. C'est par l'intermédiaire de ces lignes que s'effectuent les échanges entre les différents éléments du système. On distingue 3 types de bus.

- ✓ L**e bus de données :**

Ce bus transporte les données échangées par les différents périphériques externes du microcontrôleur. C'est un ensemble de lignes bidirectionnelles de 8, 16 ou 32 voies. La taille du bus de données détermine l'appartenance du microprocesseur du système: Un microprocesseur avec un bus des données de 16 voies sera appelé « microprocesseur 16 bits ».

Ce bus est bidirectionnel; c'est à dire que les informations qu'il véhicule peuvent transiter:

- Du microcontrôleur vers l'un de ses périphériques,
- D'un périphérique vers le microcontrôleur.

- ✓ **Le bus d'adresse :**

A chaque mot de donnée correspond un numéro: l'adresse. Pour avoir accéder à une donnée, il suffira de présenter son adresse sur le bus d'adresse. De même, pour mémoriser une donnée, il faudra présenter sur le bus d'adresse, l'adresse à laquelle on désire stocker cette donnée. Par conséquent, le bus d'adresse véhicule l'adresse qui spécifie l'origine ou la destination de l'information qui transite sur le bus de données.

Le bus d'adresse est un ensemble de lignes unidirectionnelles. La taille du bus d'adresse caractérise la capacité d'adressage du microprocesseur du système: Un microprocesseur qui a un fil d'adresse peut présenter, sur son bus d'adresse, 2 adresses distinctes.

Ce bus est unidirectionnel; c'est à dire que les informations qu'il véhicule transitent microprocesseur vers l'un de ses périphériques.

Le bus de contrôle C'est un ensemble de lignes transportant les différents signaux de commande et de synchronisation nécessaires pour le bon déroulement des échanges entre les divers éléments du système. Les lignes de ce bus ne sont pas affectées d'un poids binaire, contrairement aux lignes du bus de donnée et du bus d'adresse.

II.1.6.c Les mémoires :

Si les mémoires internes au microcontrôleur sont insuffisantes (programme de gestion trop important, les données temporaires à sauvegarder trop importantes,….),

on choisira des mémoires externes de manière à compléter ou suppléer les mémoires internes au microcontrôleur.

II.1.6.d Les périphériques optionnels :

Si les périphériques internes au microcontrôleur ne sont pas suffisants alors on pourra ajouter des périphériques externes.

II.2 Microcontrôleur 16F84A [6] :

II.2.1 Présentation du microcontrôleur :

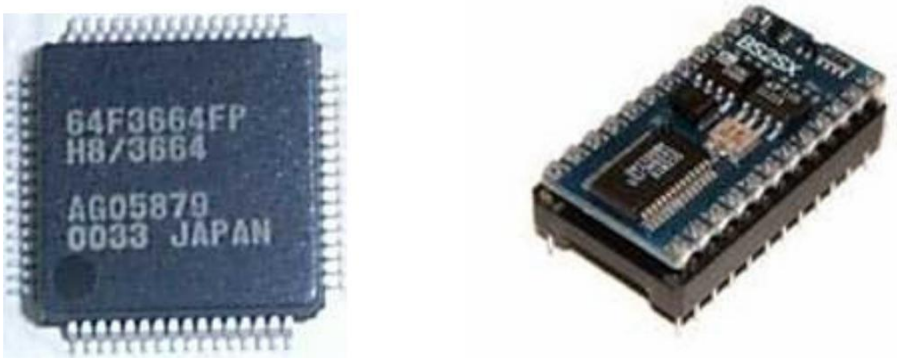


Figure II.2 : Deux exemples de microcontrôleurs.

Aujourd'hui, les microcontrôleurs sont partout : ordinateurs, portable… Assez facile d'utilisation, pour la plupart, et programmable de fort nombreuses fois (plus de 1000), ou définitivement (OTP : One Time Programmable), leur souplesse d'utilisation a séduit rapidement les divers constructeurs de divers domaines. Aujourd'hui, un des géants mondiaux s'appelle Microchip. Mais il ne faut surtout pas confondre les microcontrôleurs et les microprocesseurs. Pour résumer, on peut dire qu'un microcontrôleur est un ordinateur extrêmement miniaturisé et possédant donc assez peu de mémoire, et dont le processeur est relativement simple, alors qu'un microprocesseur ne fait qu'exécuter des instructions qui lui sont communiquées, puis renvoie les résultats.

Les principaux problèmes des microcontrôleurs sont la taille de leur mémoire et le nombre limité de périphériques qu'ils peuvent recevoir en même temps. Cependant, le nombre de ces derniers peut parfois être augmenté en associant, sur les mêmes pattes un périphérique d'entrée et un de sortie, permettant alors de doubler le nombre de périphériques connectables…

Les microcontrôleurs sont tellement miniatures, que la plupart du temps, ils sont implantés sur l'application même qu'ils sont censés piloter, comme par exemple un clavier d'ordinateur, ou la souris. Ces systèmes sont alors appelés « systèmes embarqués ». Ils exécutent tout le temps, en boucle, le même programme. Leur champ d'application ne connaît comme limite, que l'ingéniosité des divers concepteurs. Grâce à eux, la réception radio est de plus en plus fine. Leur avenir s'annonce radieux en ce début de XXI ème siècle.

Pour l'an 2000, le bilan était très positif :

- Communication : 30%
- Consommation générale : 27%
- Industrie automobile : 18%
- Périphériques informatiques : 15%
- Industrie : 10%

Nous pouvons en effet rencontrer des microcontrôleurs tous les jours dans les appareils que nous utilisons. La classification générale de puces se fait par nombre de bits : 4, 8, 16, 32 bits. un microcontrôleur se décompose en diverses parties : la mémoire de programme, la mémoire de données, le processeur, les ressources auxiliaires.

Les modèles les plus vendus sont les PIC de chez Microchip, en raison d'un excellent rapport qualité/prix, et surtout le PIC 16F84, qui est le sujet de ce livre, et qui est utilisé par toute la communauté électronique, que ce soit amateur, ou même par certains professionnels. Facile à programmer et à utiliser, son prix tourne aux alentours des 7.5€, avec une cadence de fonctionnement pouvant aller jusqu'à 20 MHz.

Chez les PIC, il y a la gamme de base (16C5X : 33 instructions de 12 bits, DIL 18 et 28), la gamme intermédiaire (16C/FXXX : 72 modèles DIL 18 à 68, 35 instructions de 14 bits), la gamme mini (12CXXX : DIL 8), la gamme supérieure (58 instructions de 16 bits), la gamme avancée (18CXXX : 10 MIPS à 40 MHz, 77 instructions de 16 bits, DIL28 à 80, 16 Kmots de 16 bits).

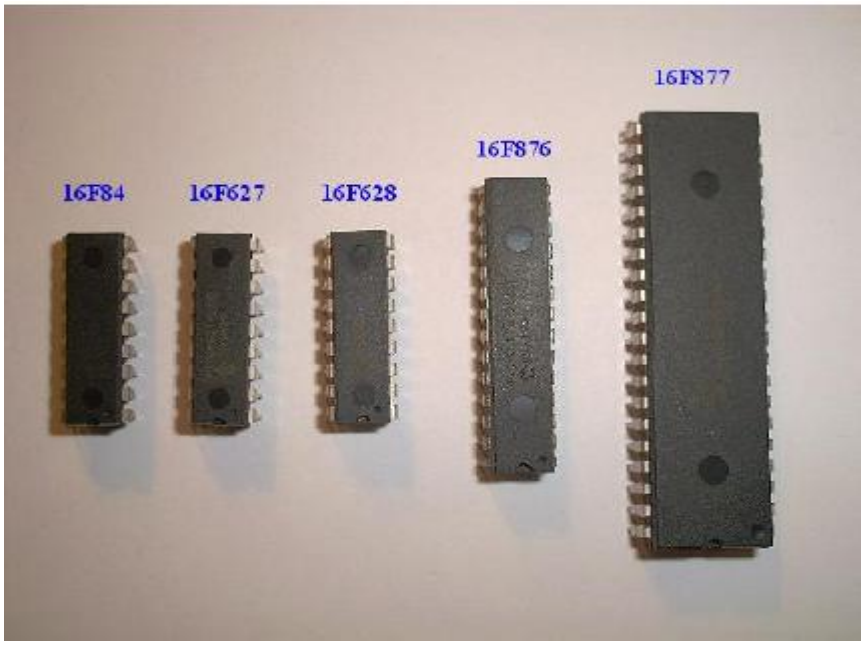


Figure II.3 : Des exemples de la famille PIC de chez Microchip

Jusqu'à une certaine époque, les microcontrôleurs respectaient l'architecture Von Neumann, inventeur de l'ENIAC, premier ordinateur au monde. Cependant, celle-ci présente des inconvénients. En effet, la vitesse d'exécution est limitée, les instructions et les données transitaient par le même bus. Pour résumer, son principal défaut était le fait qu'elle ne possédait qu'un bus pour, simultanément, la mémoire programme et la mémoire donnée.

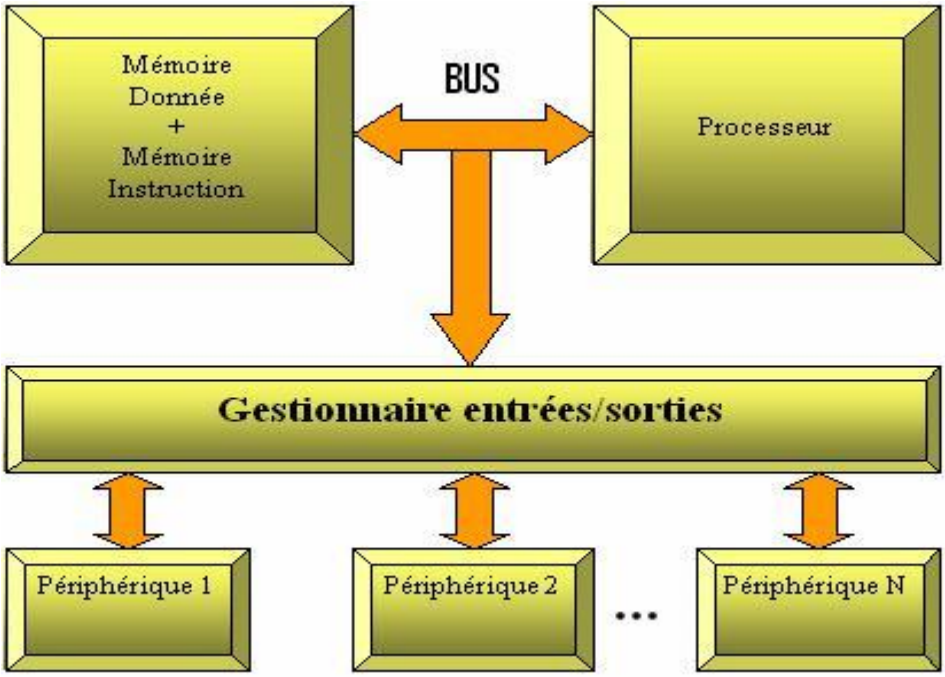


Figure II.4 : L'architecture de Von Neumann

D'où l'architecture Harvard, utilisée maintenant par les PIC. Sa particularité tient dans le fait qu'il y a deux mémoires accessibles en même temps par le processeur, par l'intermédiaire de deux bus spécifiques. l'un sert pour les données, et l'autre pour les instructions. De ce fait, les deux peuvent être accessibles en même temps, d'où un gain de vitesse, au niveau exécution.

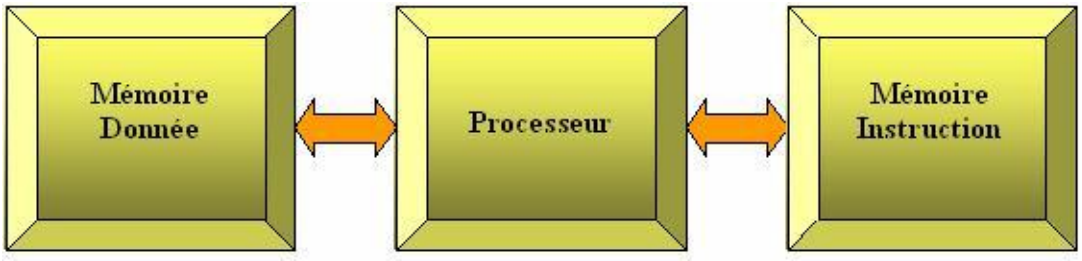


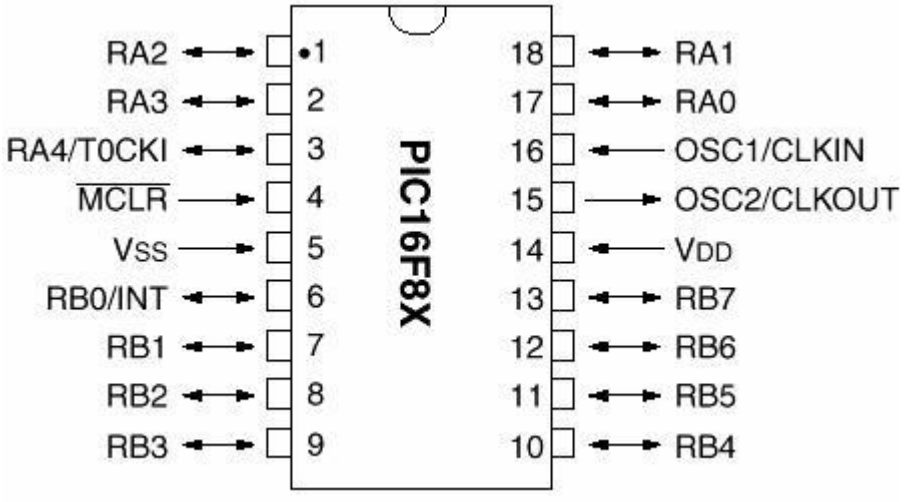
Figure II.5 : L'architecture Harvard

Ceci aidant, il existe depuis quelques années, un nouveau type de mémoires dites « flash », avec écriture et effacement électrique des données dans la mémoire. Elle est de type RAM, mais est associée à une mémoire EPROM pour des données auxiliaires.

Cependant, d'autres architectures existent, mais néanmoins moins répandues, comme par exemple, l'architecture Flynn développée en 1996.chaque microcontrôleur possède des registres. Il s'agit de mémoires intervenant dans les opérations de l'UAL (Unité Arithmétique et Logique). Celle-ci requiert deux opérandes (instructions) pour réaliser une opération.Quand un processeur exécute une instruction, il le fait en deux étapes : d'abord une phase de recherche, puis une phase d'exécution.

Tous les microcontrôleurs, ou presque, possèdent des entrées/sorties programmables. Celles-ci se configurent simplement sur les PIC en entrant une valeur à l'adresse du port correspondant. en plus du programme, les microcontrôleurs ont besoin de ce qu'on appelle un mot de configuration, tenant compte du type d'oscillateur (RC, XT,…), de l'activation ou non du Watchdog (qui empêche un dysfonctionnement), d'une temporisation au démarrage ou non, et du code P (non utilisé et explicité dans ce livre, le code P, pour Protection, sert à verrouiller la programmation du PIC ; c'est-à-dire que pour le programmer, il faut un mot de passe).

II.2.2 Caractéristiques du PIC 16F84 [6] :



Ce microcontrôleur, que nous surnommerons « le pic » pour des raisons de commodité, dans ce livre, possède 13 broches configurables, réparties sur deux ports : le port A et le port B.

Remarque : ***On ne peut affecter que deux valeurs différentes de configuration à chaque patte : un '1' pour la mettre en entrée, ou un '0' pour une sortie.***

- Le port A possède 5 broches (nommées RA0 à RA4), mais la quatrième, également appelée TOCKI peut servir pour une éventuelle temporisation externe.
- Le port B, lui, possède 8 broches (de RB0 à RB7) ; mais la broche RB0 peut également servir comme interruption éventuelle (un peu comme un garde sur un évènement)

  La broche 4, le MCLR barre, sert à indiquer au PIC s'il est en fonctionnement normal (un '1' logique) ou alors s'il est en cours de programmation (un '0' logique). Cette broche sert également à un éventuel Reset du PIC.
- Ne reste que 4 broches : - l'alimentation (la 5 (0 volts) et la 14(+5 volts))
  - l'oscillateur (pattes 16 et 15)

Outre ces caractéristiques, il faut savoir que le PIC 16F84 n'utilise pas de signaux analogiques, mais uniquement numériques (logique). de plus, il faut savoir, qu'il ne fait pas la différence entre les niveaux logiques TTL et CMOS. Il est donc alors préférable d'utiliser les deux extrêmes de signaux logiques TTL (0 et 5 volts), pour éviter tout problème possible.
Voici enfin, les caractéristiques du PIC 16F84 fournissent par Microchip :

- Mémoire de programme : 1KO, type Flash
- Mémoire de données RAM : 68 octets
- Mémoire de données E²PROM : 64 octets

- Niveau de la pile : 8
- Jeux d'instruction RISC : 35 de 14 bits
- Temps d'exécution des instructions normales : 4*Tosc
- Temps d'exécution des instructions de saut : 8*Tosc
- Cause d'interruption : 4
- Fréquence max de travail : 10 MHz
- Lignes E/S numérique : 13
- Temporisateur : un pour l'utilisateur, un pour le Watchdog
- Tension d'alimentation : 2 à 6 V continu.
- Tension de programmation : 12 à 14 V continu.
- Boîtier : DIL 18

Enfin, autre point fort du PIC : la consommation, car en tant que système embarqué, il faut que ce microcontrôleur consomme peu.

Ainsi, des 2 mA de consommation en fonctionnement normal, il peut passer à 10µA en fonction veille ou sommeil, comme par exemple sur un téléphone portable, quand on ne s'en sert pas, la lumière reste éteinte : il est en veille. En revanche, au moindre appui sur un bouton (interruption externe, broche RB0 ici), il se remet en marche. Le Pic fonctionne de la même manière.

Remarque : il faut cependant savoir que le PIC peut fournir jusqu'à 20 mA par sortie, soit, si tous les ports sont en sortie, un débit de 260 mA. Il est aisé de comprendre alors que les 2 mA sont une consommation moyenne et que celle-ci varie selon l'utilisation du PIC.

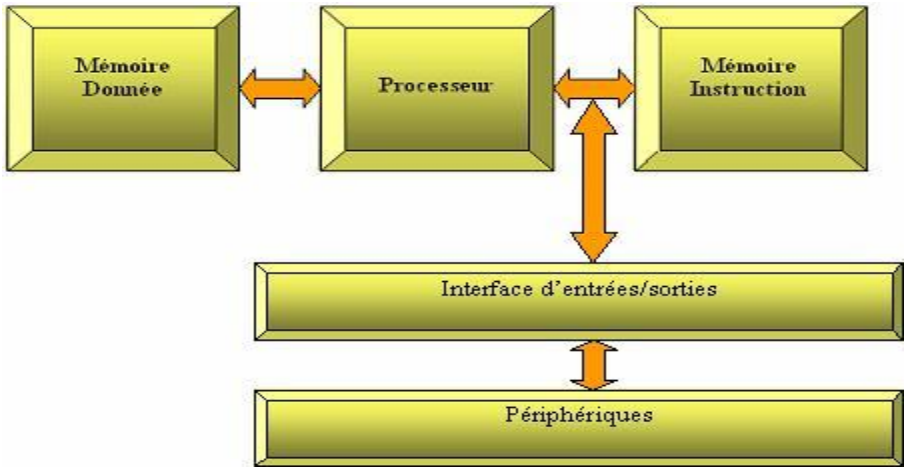
II.2.3 Fonctionnement du PIC 16F84 :

Dans cette partie du livre, nous traiterons des principaux systèmes du PIC, plus ou moins utiles. Nous illustrerons ces systèmes par des schémas, les plus clairs et les plus compréhensibles possibles. Certains de ces systèmes, étant des paramètres configurables, devront faire l'objet d'attention selon que vous déciderez où non de les mettre en œuvre.

Nous verrons donc :

- les entrées/sorties
- les différents types d'oscillateurs possibles (pour l'horloge) ›le reset
- la mémoire EEPROM
- la mémoire flash, utilisée dans le PIC
- les interruptions
- le TMR0
- le Watchdog
- le RTCC

II.2.4 Les entrées/sorties :



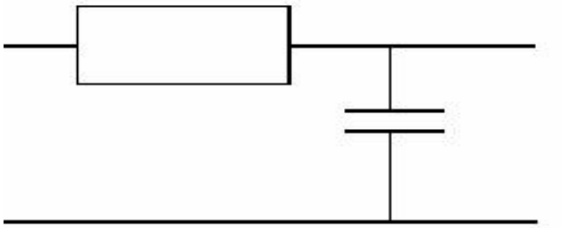
Les entrées/sorties sont l'interface entre le système embarqué et le monde extérieur.

Les entrées/sorties sur le Pic sont réparties sur deux ports : le port A (5 E/S) et le port B (7 E/S). Pour configurer ces dernières, il s'agit de rentrer un '1' pour la mettre en entrée, ou un '0' pour une sortie. Quand chaque bit est déterminé, on obtient alors le mot binaire à rentrer dans le registre de configuration.

II.2.5 Les oscillateurs :

Ils peuvent être de plusieurs types. Nous verrons donc ceux-ci en détail, afin de pouvoir choisir le meilleur possible.

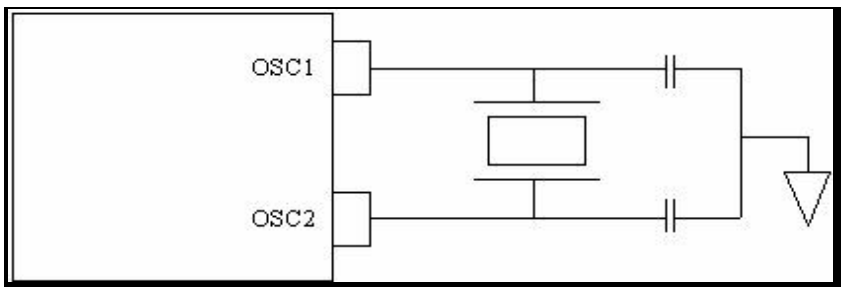
RC :



Cet oscillateur coûte peu cher à fabriquer, mais n'est pas très précis. Il peut convenir si votre système n'a pas besoin d'une horloge précise. La constante de temps est alors : τ=RC

Remarque : $F=1/\tau$

Systèmes à quartz :



Ce système peut se diviser en trois sous-systèmes : HS, XT et LP .La plupart du temps, l'oscillateur utilisé sera un XT, un quartz à 4 MHz. Cependant, le Pic peut monter jusqu'à 20 MHz en utilisant un mode HS (High speed). Quand au mode LP, il s'agit du mode de fonctionnement en basse fréquence, peu utilisé.

II.2.6 Le reset :

Ce système est relativement important. En effet, lorsque le système a des ratés, ou plante, il est utile de pouvoir le réinitialiser. Il est également possible et utile parfois d'effectuer un reset lors de la mise sous tension. Le reset s'effectue en mettant un '0' logique , Le reset s'effectue en mettant un '0' logique sur la patte 4 du microcontrôleur. Il existe plusieurs types de système de reset : le manuel, l'automatique, et un mixte.

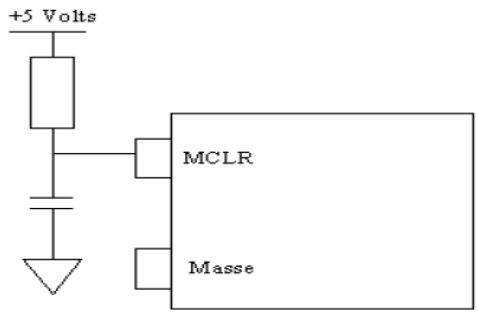


Figure II.6: Le reset automatique (condensateur de 1µF et une résistance de 1 K)

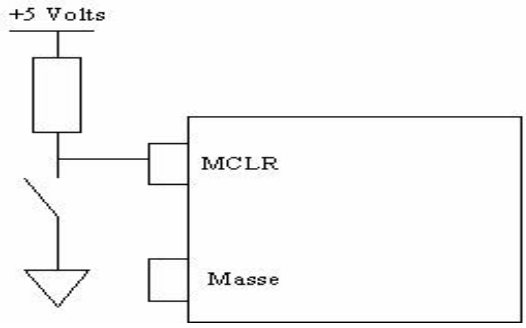


Figure II.7 : Le reset manuel (avec une résistance de 1K

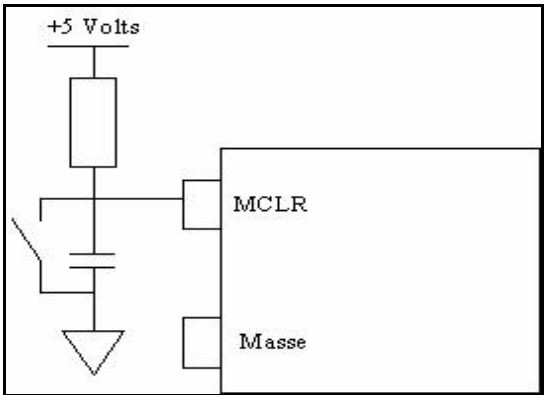


Figure II .8 : Le modèle mixte, permettant un reset automatique lors de la mise sous tension et également un interrupteur de reset manuel.

II.2 .7 La mémoire EEPROM :

L'EEPROM (Electrically Erasable Programmable Read Only Memory) est une mémoire interne au Pic. Il s'agit d'une mémoire non volatile dans laquelle le PIC peut stocker des données, comme par exemple les résultats d'une acquisition.

II.2.8 La mémoire flash :

La mémoire flash est le nouveau type de mémoire E²PROM. Bien plus souple que les premières générations de ces dernières, la mémoire flash permet une écriture/effacement de toute la mémoire ou que d'une partie au choix. Ce type de mémoire possède beaucoup de caractéristiques intéressantes.

Caractéristiques de la mémoire Flash
1-Pas de différence significative entre les différents types de mémoires existantes sur le marché
2-Utilisation simplifié par rapport aux autres mémoires E²PROM
3-Faible coût des outils de développement
4-Simplification du débogage
5-Possibilité de misa a jour du Firmware
6– Grande plage de tension de programmation.

Caractéristiques	Mémoire flash
Alimentation	2-5,5 volts
Tension de programmation	Vdd
Autoprogrammable	OUI
Débogage en circuit	OUI
Technologie	0,5µm
Cycles d'effacement/écriture	100K(données), 1K(programme)
EEPROM Données	OUI
Temps de programmation/cycle	1-2 ms

II.2.9 Les Interruptions :

Une interruption provoque l'arrêt du programme principal pour aller exécuter une procédure d'interruption. A la fin de cette procédure, le microcontrôleur reprend le programme à l'endroit où il s'était arrêté. Le PIC16F84 possède 4 sources d'interruption. A chaque interruption sont associés deux bits: un bit de validation et un drapeau. Le premier permet d'autoriser ou non l'interruption, le second permet au programmeur de savoir de quelle interruption il s'agit. Tous ces bits sont dans le registre INTCON à part le drapeau EEIF de l'interruption EEI qui se trouve dans le registre EECON1.

II.2.9.1 Déroulement d'une interruption :

Lorsque l'événement déclencheur d'une interruption intervient, alors son drapeau est positionné à un (levé). Si l'interruption correspondante a été validée, elle est alors déclenchée : le programme arrête ce qu'il est en train de faire et va exécuter la procédure d'interruption qui se trouve à l'adresse 4 en exécutant les étapes suivantes :

- l'adresse contenue dans le PC (Program Counter) est sauvegardée dans la pile, puis remplacée par la valeur 0004 (adresse de la routine d'interruption).
- Le bit GIE est placé "0" pour inhiber toutes les interruptions (afin qu'on ne soit pas dérangés pendant l'exécution de la procédure d'interruption).
- A la fin de la procédure d'interruption (instruction RETFIE) :
  - le bit GIE est replacé à l'état haut (autorisant ainsi un autre événement)
  - le contenu du PC est rechargé à partir de la pile ce qui permet au programme de reprendre là où il s'est arrêté

Deux remarques importantes sont à faire :

Le drapeau reste à l'état haut même après le traitement de l'interruption. Par conséquent, il faut toujours le remettre à "0" à la fin de la routine d'interruption sinon l'interruption sera déclenchée de nouveau juste après l'instruction RETFI

Seul le PC est empilé automatiquement. Si cela est nécessaire, les registres W et STATUS doivent être sauvegardés en RAM puis restaurés à la fin de la routine pour que le microcontrôleur puisse reprendre le programme dans les mêmes conditions où il l'a laissé.

II.9.2 L'interruption INT (Entrée RB0 du port B) :

Cette interruption est provoquée par un changement d'état sur l'entrée RB0 du port B quand elle est programmée en entrée. Elle est gérée par les bits :

- INTE : bit de validation (1=oui, 0=non)
- INTF : drapeau
- INTEDG : front de déclenchement, 1=montant, 0=descendant (registre OPTION_REG)

II.9.3 L'interruption RBI (RB4 A RB7 du port B) :

Cette interruption est provoquée par un changement d'état sur l'une des entrées RB4 à RB7 du port B, Le front n'a pas d'importance. Les bits associés sont RBIE (validation) et RBIF (drapeau).

II.9.4 L'interruption T0I : Débordement du Timer TMR0 :

Cette interruption est provoquée par le débordement du timer TMR0. Les bits associés sont T0IE (validation) et T0IF (drapeau).

II.9.5 L'interruption EEI : Fin d'écriture dans l'EEPROM :

Cette interruption est déclenchée à la fin d'une écriture réussie dans l'EEPROM.

Les bits associés sont EEIE (validation) et EEIF (drapeau).

INTCON	GIE	EEIE	T0IF	INTE	RBIE	T0IF	INTF	RBIF
EECON1	-	-	-	EEIF	WRERR	WREN	WR	RD
OPTION_REG	RBPU	INTEDG	TOCS	TOSE	PSA	PS2	PS1	PS0

II .2.10 Le TMR0 :

Il s'agit d'un temporisateur interne de 8 bits, qui peut être initialisé à une valeur donnée. A chaque passage de FF à 00 (en hexa), le bit de débordement est activé. Il faut alors le remettre à zéro, pour pouvoir détecter un autre débordement (non automatique).

Il possède deux modes de fonctionnement possible, dont le choix s'effectue par la mise à 1 ou à 0 du bit TOSC (voir chap. nécessaire à la programmation), l'entrée horloge devenant alors la patte RA4, en mode dit « TOCKI ».

Ces deux modes sont :

- temporisateur interne (peut alors servir pour des fonctions de temps)
- Compteur d'évènements (peut servir pour compter des évènements extérieurs par l'intermédiaire de RA4)

Remarque : la patte RA4 doit être définie en entrée dans le cas du compteur d'évènements.

II .2.11 Le Watchdog :

Littéralement le « chien de garde », le Watchdog est un système de surveillance du bon déroulement du programme. Il s'agit d'un compteur, qui est réinitialisé régulièrement dans le cas d'un fonctionnement normal. Mais dans le cas d'un dysfonctionnement, le compteur va jusqu'au bout et déclenche alors un reset interne, par débordement, réinitialisant le Pic.

Le compteur peut fonctionner à la fréquence de l'oscillateur, ou bien à une fréquence spécifique, désignée par la fréquence de l'oscillateur modifiée par un diviseur.

Remarque : il n'est pas obligatoire de l'utiliser. Son utilisation est activée ou non, lors de la programmation du PIC.

II.2.12 Le RTCC:

Il s'agit d'une horloge interne destinée au fonctionnement du timer 0 (TMR0) dans le cas d'un fonctionnement de ce dernier sur horloge interne.

II.3 Conclusion :

Nous avons traité tout au long de ce chapitre les microcontrôleurs, spécifiquement le PIC 16F84A à savoir ses caractéristiques, son architecture interne et externe, sa capacité …etc. Partant de cette étude, on peut déduire que le microcontrôleur 16F84A peut bien jouer le rôle d'une unité de contrôle pour notre système et maintenant nous pouvons passer à la conception et la réalisation, puisque le composant le plus important dans notre système nous est déjà familier.

III.1 Introduction :

Une montre digitale est un système électronique qui remplit un ensemble de fonctions spécialisées. Il s'agit d'une horloge et d'un calendrier numérique avec affichage sur un module LCD alphanumérique à l'aide d'un microcontrôleur 16F84A.

Dans ce chapitre nous décrivons les différentes étapes qui nous permettrons la conception et la réalisation d'une carte électronique : l'étude, le choix des composants nécessaires, le test sur la plaquette d'essai, la réalisation comme circuit imprimé et enfin la soudure des composants.
Notre réalisation pratique a été répartie en trois parties:

Partie I : Description du système:

- Schéma synoptique.
- Présentation des blocs.
- Fonctionnement.

Partie II : Simulation du circuit:

- Schéma électronique.
- Programmation du PIC.
- Routage et Création du circuit imprimé.

Partie III : Réalisation pratique :

- Liste des composants du circuit (résistances, condensateurs, …etc.).
- Gravure du programme sur le pic.
- Implantation des composantes.

III .2 Description du système :

III.2.1 : Schéma synoptique :

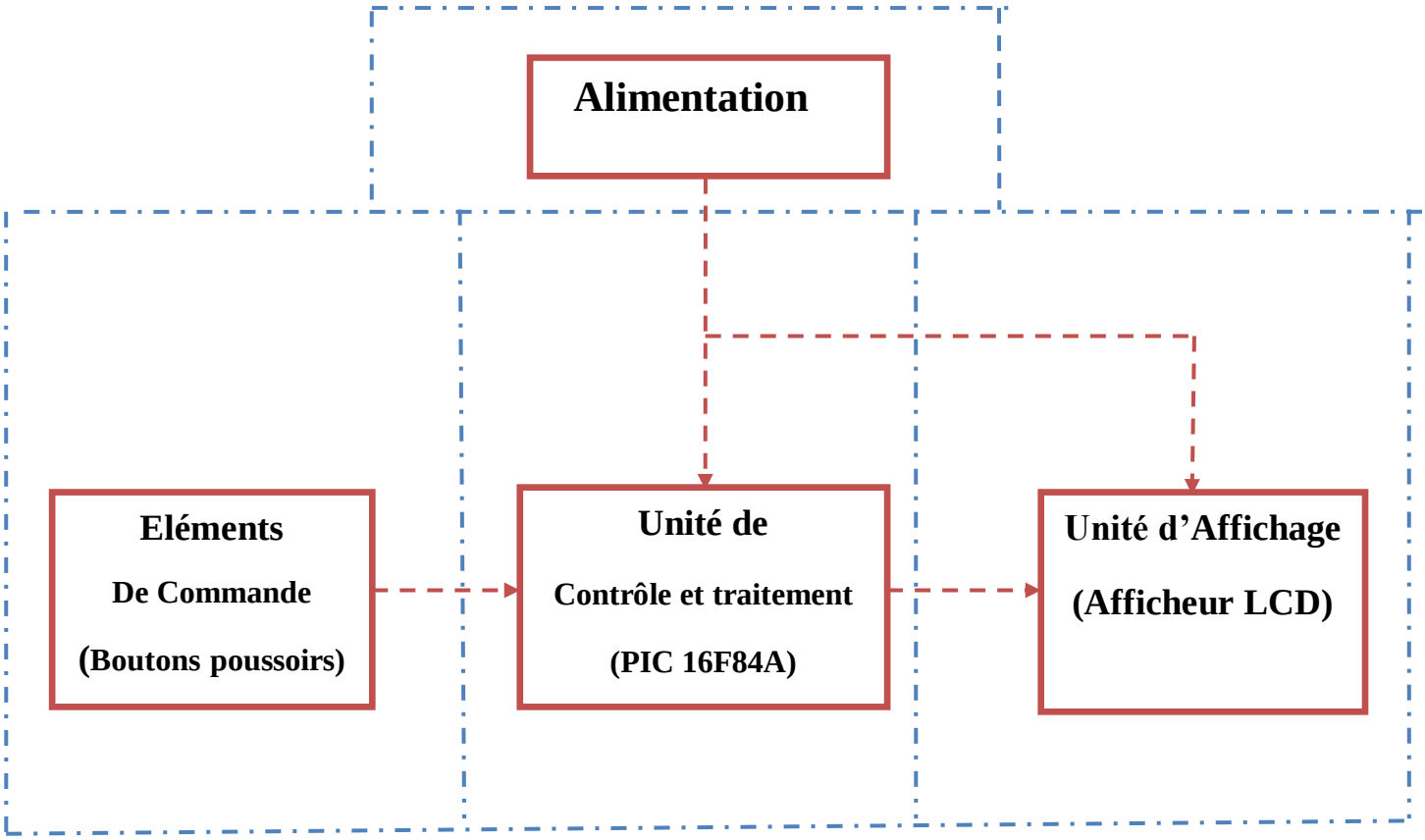


Figure III.1 : schéma synoptique du circuit

III.2.2 : Présentation des blocs :

Le montage est articulé autour de quatre principales parties fonctionnelles :

- Alimentation.
- Eléments de commande (boutons poussoirs).
- Unité de contrôle de traitement (PIC16F84A).
- Unité d'affichage (Afficheur LCD).

Notre projet, Il s'agit d'une horloge (un calendrier) contrôlée par des boutons poussoirs permettent le réglage de l'heure et le jour à l'aide d'un microcontrôleur 16F84A, avec un afficheur LCD.

III.2.3 : Fonctionnement :

❖ **Alimentation :**

Tous les montages électroniques nécessitent une alimentation pour fonctionner. Notre montage nécessite une alimentation de 5 v pour alimenter le PIC et ses périphériques. Pour cela, nous avons opté pour une batterie de 9 volts et un régulateur LM 7805 qui résulte en sortie une tension de 5 v pour alimenter le PIC et les autres circuits.

Un régulateur de tension LM7805 qui résulte en sortie une tension de 5V quelque soit la tension affectée à son entrée.

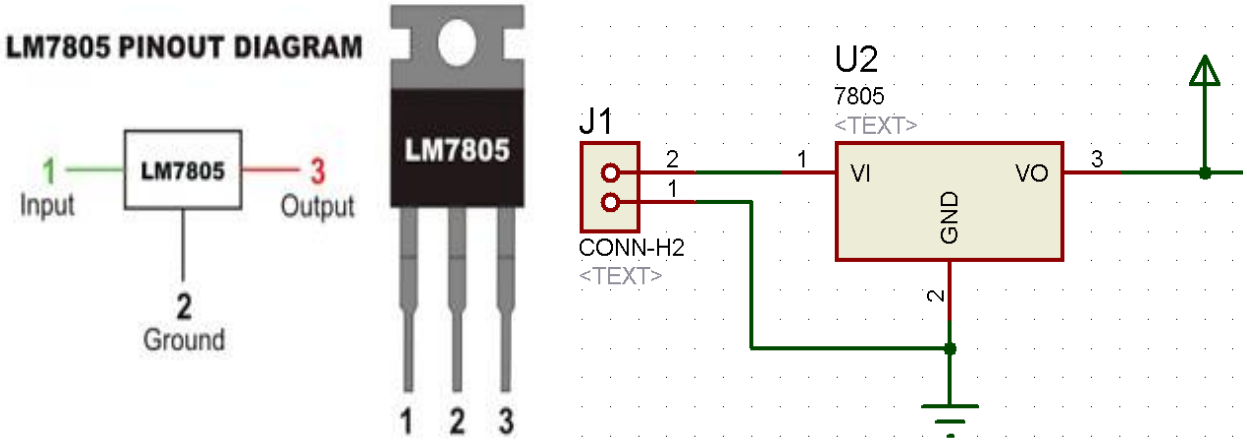


Figure III.2 : regulateur LM7805

❖ **Unité de contrôle et traitement :**

De nos jours, les mini projets ont besoin d'un circuit intégré a pour rôle le traitement de l'information, rapide et à un prix moins chère alors que la solution c'est le pic, avec cette Condition on a utilisé le pic 16F84A dans notre projet.

➢ **Alimentation :**

L'alimentation est assurée par les pattes **VDD** et **VSS.** Elles permettent à l'ensemble des composants électroniques du **PIC** de fonctionner. Pour cela on relie **V_{SS} (patte 5)** à la masse **(0 Volt)** et **VDD (patte 14)** à la borne positive de l'alimentation qui doit délivrer une tension continue comprise entre 3 et 6 Volts.

➢ **Cadencement du PIC 16F84A :**

Le PIC 16F84A peut fonctionner en 4 modes d'oscillateur.

- ✓ **LP** : Low Power crystal : quartz à faible puissance.
- ✓ **XT** : Crystal / Resonator : quartz/résonateur en céramique.
- ✓ **HS** : High Speed crystal/resonator : quartz à haute fréquence/résonateur en céramique HF.
- ✓ **RC** : Circuit RC (oscillateur externe)

On peut utiliser un quartz allant jusqu'à 20Mhz relié avec deux condensateurs de découplage, du fait de la fréquence importante du quartz utilisé. Quelque soit l'oscillateur utilisé, l'horloge système dite aussi horloge instruction est obtenue en divisant la fréquence par 4.

Dans notre cas on a utilisé un quartz de 4 MHz, on obtient une horloge instruction de 1 MHz, soit le temps pour exécuter une instruction de 1µs. Cette horloge doit être stabilisée de manière externe au moyen d'un cristal de quartz connecté aux pattes OSC1/CLKIN **(patte 16)** et OSC2/CLKOUT **(patte 15).**

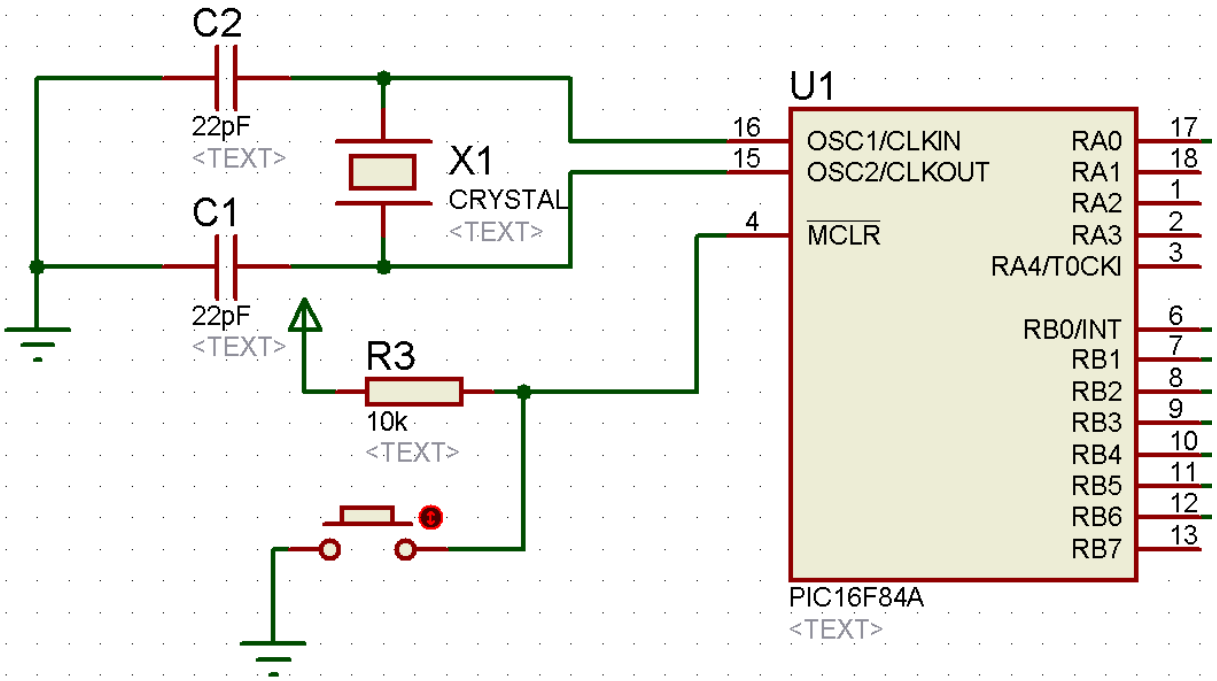


Figure III.3 : cadencement du pic 16F84A.

- **Circuit Reset MCLR :**

La broche MCLR (Master Clear) a pour effet de provoquer la réinitialisation du microprocesseur lorsqu'elle est connectée à 0.

Lorsque on branche la patte 4 à 0 v, le signal de "RESET" est activé, tous les registres sont initialisés et le compteur programme se place à une adresse spécifique appelée "Vecteur de RESET".

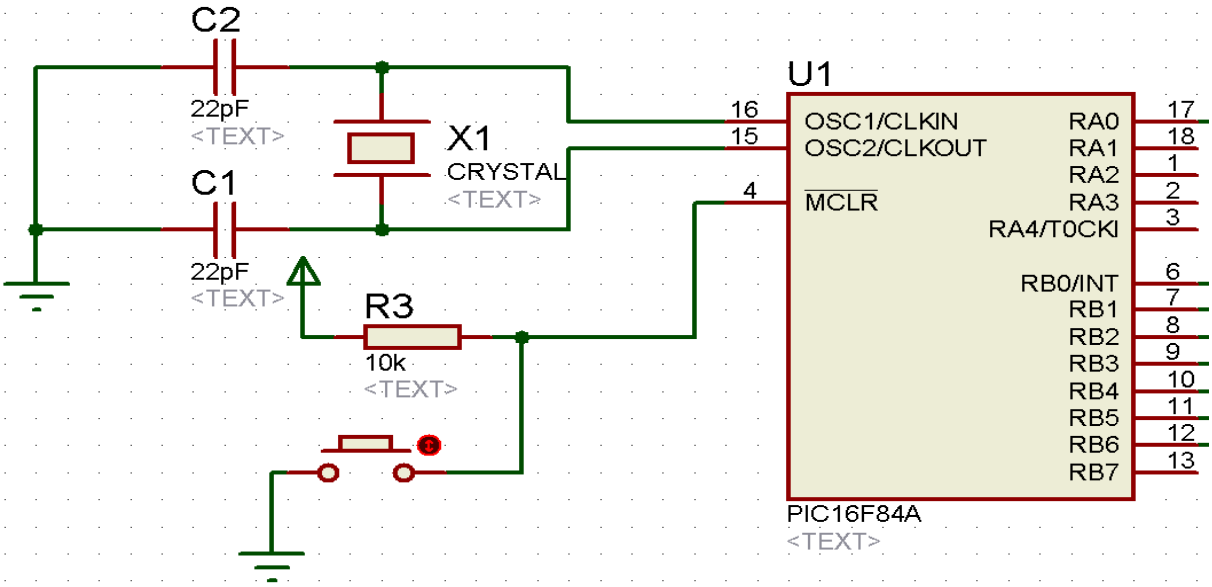


Figure III.4 : circuit Reset MCLR.

- **Branchement des ports sorties / entrées :**

Il est intéressant de souligner que notre choix s'est porté sur un microcontrôleur de la firme *MICROCHIP*. Ce microcontrôleur est le plus répandu dédié pour des applications de la robotique. Son prix compétitif, la disponibilité de la documentation, ses performances et la distribution gratuite des logiciels pour le programmer et le déboguer, font de lui un bon candidat pour différentes applications industrielles.

Le 16F84A est un circuit intégré de 18 broches, possède 13 broches configurables, réparties sur deux ports : le port A et le port B:

- **Le port A :** possède **5** broches (nommées RA0 à RA4), nous avons configuré la broche RA0 comme une entrée pour pouvoir régler l'heure .

- **Le port B :** possède **8** broches (nommées RB0 à RB7) ; la broche **RB0** peut également servir comme interruption éventuelle afin de régler l'heure (un peu comme un garde sur un évènement).
  - ✓ La broche **RB1** ⟶ sortie destinée à commander une entrée de contrôle **RS** (Register Select)
  - ✓ La broche **RB2** ⟶ sortie destinée à la validation **(EN)**, qui est utilisée pour initier le transfert de commandes ou de données entre le module et le microcontrôleur.
  - ✓ Les broches **RB3** a **RB6** sont des lignes de bus de données (**D4** a **D7**) .pour communiquer avec l'écran LCD. Les données peuvent être transférées entre le microcontrôleur et le module LCD.

- **Programmation du pic 16F84A :**

La programmation des PIC se fait par le langage assembleur qui est un langage de bas niveau qui représente le langage machine sous une forme lisible par un humain. Les combinaisons de bits du langage machine sont représentées par des symboles dits « mnémoniques » (du grec mnêmonikos, relatif à la mémoire), c'est-à-dire faciles à retenir.
Le programme assembleur convertit ces mnémoniques en langage machine en vue de créer par exemple un fichier exécutable. le développement des environnements de programmation , nous a permis de voir naître de nouveaux compilateurs qui permettent de programmer avec les langages haut niveau tels que le C , PASCAL,BASIC etc… .

Ces environnements comportent aussi des bibliothèques qui permettent de faciliter le développement. Il existe plusieurs outils de développement, les uns sont gratuits, les autres sont payants.

Dans notre recherche, nous avons opté pour le langage C pour la programmation du pic . Ce choix est à la fois un choix personnel et un choix technologique.

D'une part le langage C est utilisé dans différents systèmes et domaines de développement, ce qui nous permettra une évolution future, d'autre part le langage C est l'un des langages les plus puissants.

- ❖ **Programme principal :**

Le programme principal est constitué d'un sous-programme destiné à gérer le processus du réglage de l'heure et jour, l'organigramme (figure III.5) ci-dessous indique une représentation générale du programme de régulation.

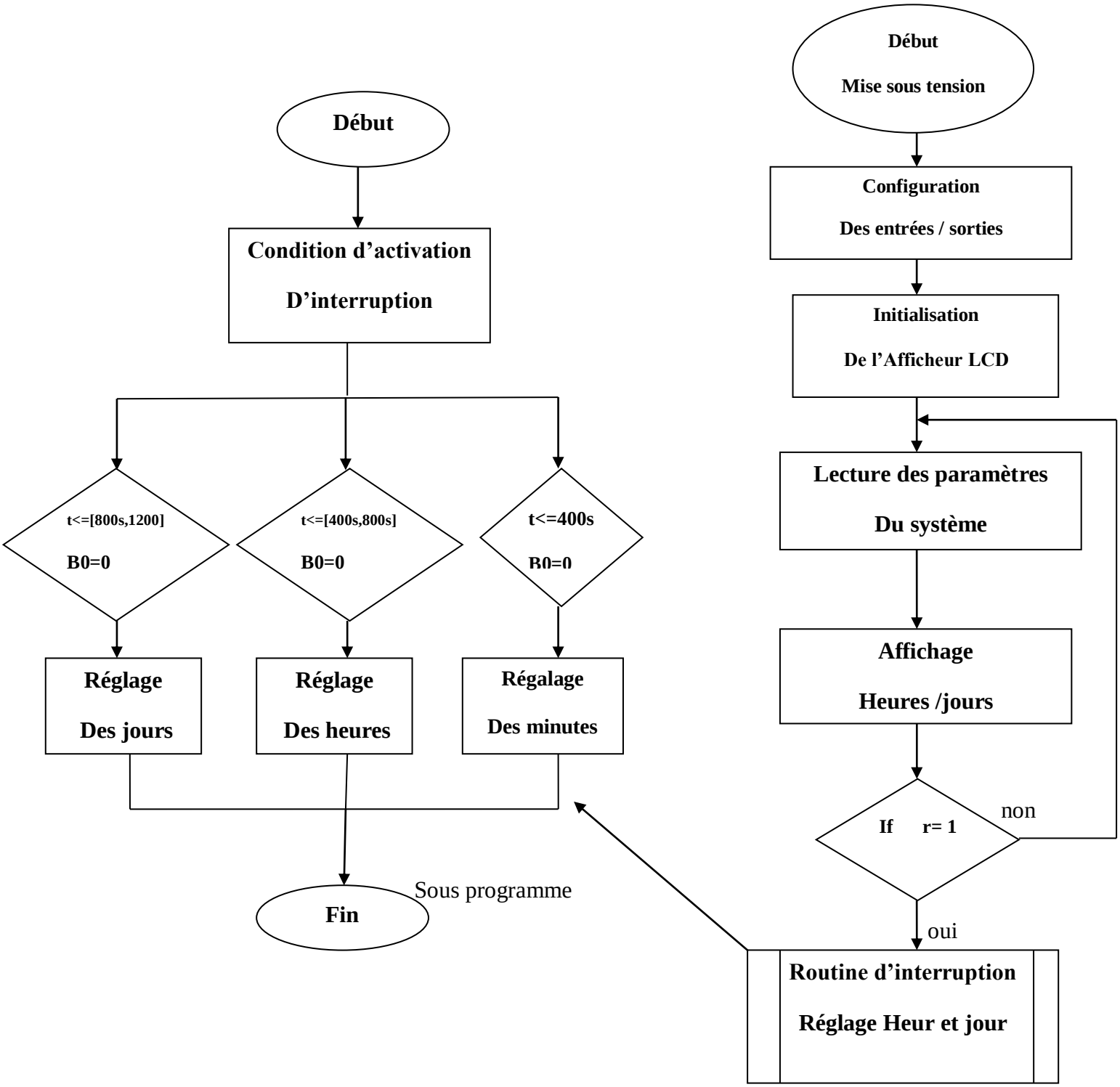


Figure III.5:organigramme principal

❖ **Unité d'Afficheur (Afficheur LCD):**

La programmation en assembleur d'un écran LCD en mode parallèle ou en série est généralement une tâche complexe et nécessite une bonne compréhension du fonctionnement interne de l'écran LCD. La langue MicroC fournit des fonctions pour les deux modes de programmation basée sur le texte déjà préparé, ce qui simplifie l'utilisation d'écrans LCD.

➢ **Le contrôleur LCD :**

Selon le modèle, l'écran LCD est fabriqué avec 16 pattes pour l'interface. Le tableau suivant présente la configuration des pattes et les fonctions des pattes d'un LCD à 16 pattes.

Broche	Nom	Description
1	Vss	Masse
2	Vdd	Alimentation 5v
3	CO	Variables de 0 à 5v permet de modifier le contraste de l'afficheur
4	RS	Indique une commande ou une donnée à afficher (0: Commande / 1: Donnée)
5	R/W	Indique une écriture ou une lecture (0: Écriture / 1: Lecture)
6	E	Indique une validation (Le niveau Haut doit être maintenu 500µs)
7	D0	Bus de données bidirectionnel.
8	D1	
9	D2	
10	D3	
11	D4	
12	D5	
13	D6	
14	D7	
15	A	Anode rétroéclairage (+5V)
16	K	Cathode rétroéclairage (masse)

Tableau III.1: le brochage d'un afficheur LCD.

❖ **La patte VSS** est reliée à **0**V.

❖ **La patte VDD** doit être connectée à la borne positive d'alimentation. Bien que les fabricants spécifient une alimentation **5V** DC.

❖ **La patte 3 VEE** est désignée pour le réglage du contraste de l'affichage et doit être reliée à une alimentation en courant continu.

- **La patte 4** est le registre de sélection **(RS)** et lorsque cette patte à **0** V, les données sont transférées à l'affichage. lorsque **RS** est a **+5** V, les données de caractères peuvent être transférées à partir du module LCD.
- **La patte 5** est le registre de sélection de lecture / écriture (**R** / **W**). Cette patte est reliée avec la masse (état logique bas) afin d'écrire des données de caractères au module LCD.
- **La patte 6** est la validation **(EN)**, qui est utilisée pour initier le transfert de commandes ou de données entre le module et le microcontrôleur.
- **Les pattes 7 à 14** sont les huit lignes de bus de données (**D0** à **D7**). Les données peuvent être transférées entre le microcontrôleur et le module LCD à l'aide soit d'un seul octet de 8 bits soit de deux 4-bits. Dans notre cas, seules les quatre lignes de données (**D4** à **D7**) sont utilisées. Le 4-bits mode a l'avantage de nécessiter moins de lignes d'E/ S pour communiquer avec l'écran LCD, comme il est indiqué dans ce schéma :

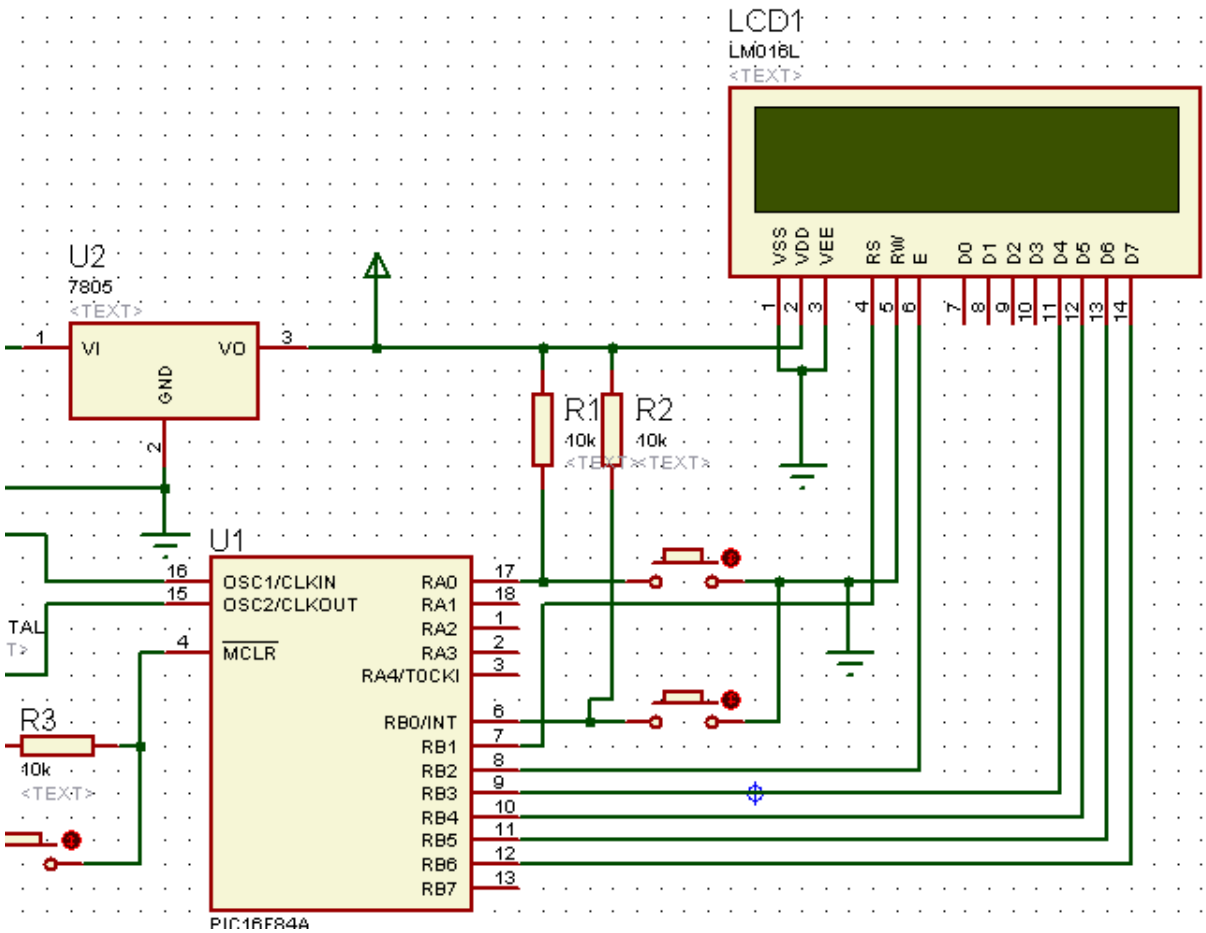


Figure III.6 : branchement afficheur LCD

III .3 Simulation du montage :

Avant de passer à la réalisation pratique de notre système nous avons eu recours à la simulation des différentes parties du système. Pour cela on a utilisé le logiciel ISIS Proteus qui est un très bon logiciel de simulation en électronique.

Proteus Professional est une suite logicielle destinée à l'électronique. Développé par la société **Labcenter Electronics**, les logiciels incluent dans **Proteus Professional** permettent la CAO (Construction Assistée par Ordinateur) dans le domaine électronique. Deux logiciels principaux composent cette suite logicielle: (ISIS, ARES, PROSPICE) et VSM.

III.3.1 : Schéma électronique :

Pour la création du schéma électronique de notre carte on a utilisé le logiciel ISIS Proteus. Afin de le saisir, il faut créer un nouveau projet sur l'écran, puis placer les composants qui doivent être sélectionnés à partir de la bibliothèque des composants sur la zone de travail.

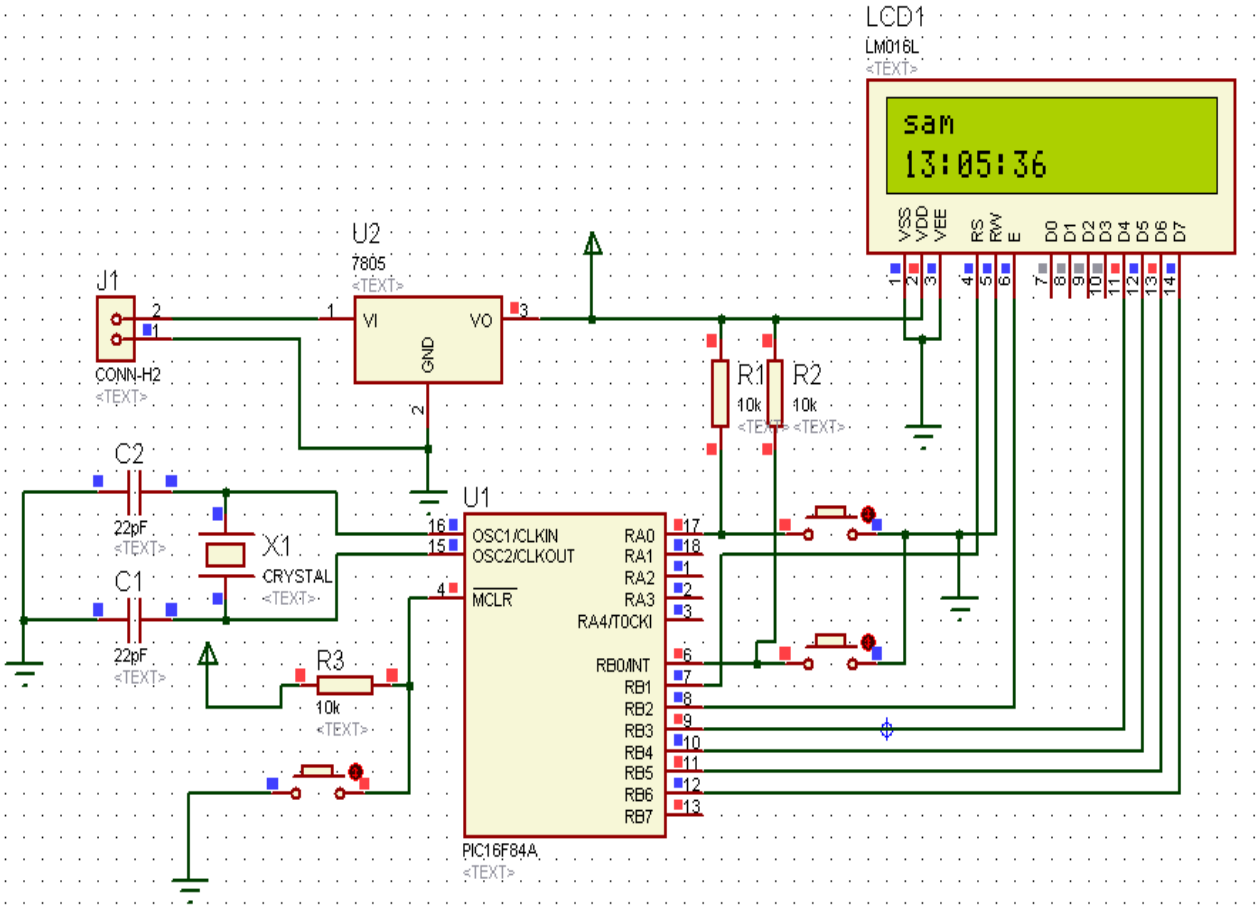


Figure III.7: schéma électronique d'une montre digitale.

III.3.2 : Programmation du pic 16F84A :

La programmation des microcontrôleurs est basée sur le code machine, qui est connu comme code assembleur, ce code contient les instructions du programme, le code assembleur est bien détaillé et difficile à écrire. Le programme en code assembleur est très long et difficile à comprendre.la création des compilateurs de haut niveau a rendu facile la création et l'édition de programmes, bien entendu les microcontrôleurs ne font pas exception. Dans le commerce, il y a plusieurs variétés de compilateurs des différents fabricants et avec différents langages de haut niveau.

Dans notre cas on a utilisé le compilateur MikroC pro, Pour créer un nouveau projet, saisir et compiler un programme on suive les étapes suivantes :

- ✓ Lancement du programme Mikro PRO for PIC: double clique sur l'icône de logiciel.
- ✓ Création d'un nouveau projet : on cliquant « New Project ».

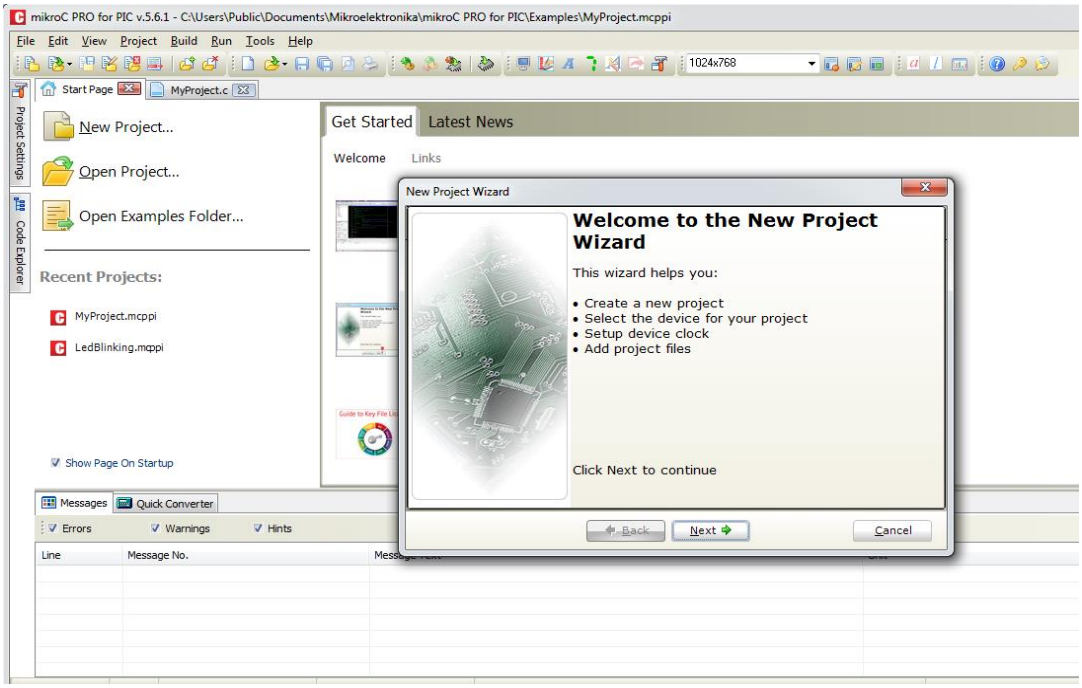


Figure III.8: création d'un nouveau projet.

La prochaine action est de cliquer sur le bouton *Next*, à cette étape de l'assistant affiche une case pour sélectionner la référence de PIC, qu'on souhaite utiliser. Dans ce champ, on sélectionne leP16F84A.

L'étape suivante est de définir la fréquence d'oscillation avec laquelle travaillera le PIC ; dans cet exemple on sélectionne 4.000000 MHz.

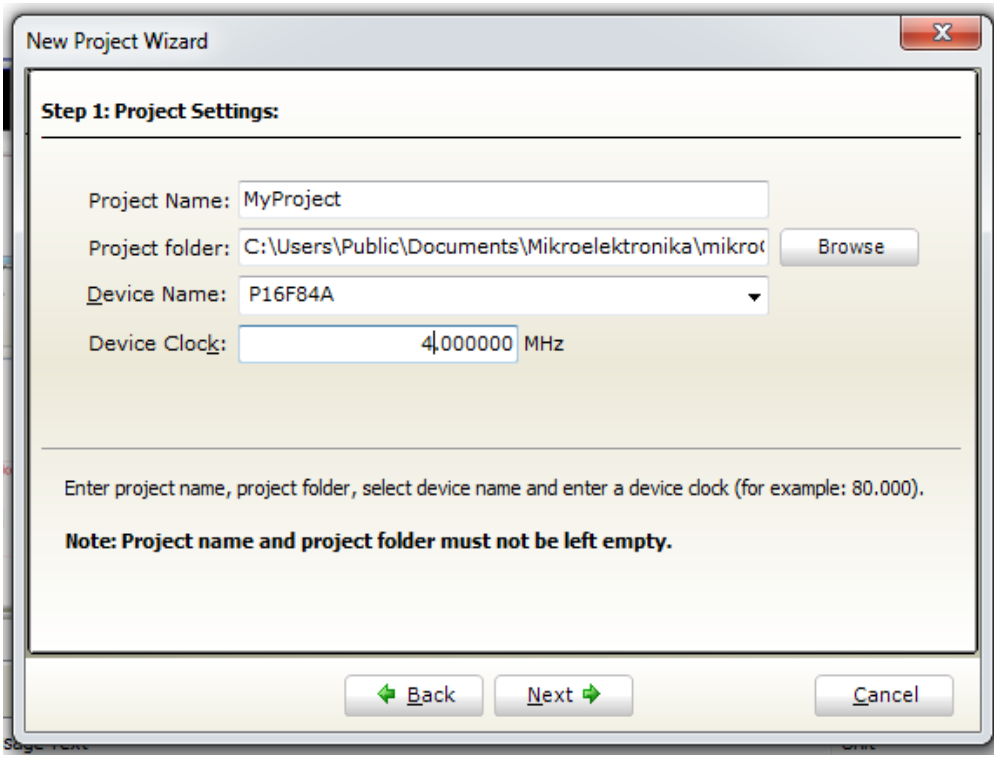


Figure III.9: choix de fréquence et le pic.

L'option suivante permet de définir le répertoire où le développeur enregistrera le projet, dans ce répertoire le programme enregistrera tous les fichiers nécessaires, parmi lesquels le code source qui sera archivé avec l'extension *.c*, et l'exécutable du PIC avec l'extension (*hex)*.

Enfin, la configuration est terminée et le projet est créé, à la fin la fenêtre doit apparaître ou on peut écrire le programme tout simplement comme suit :

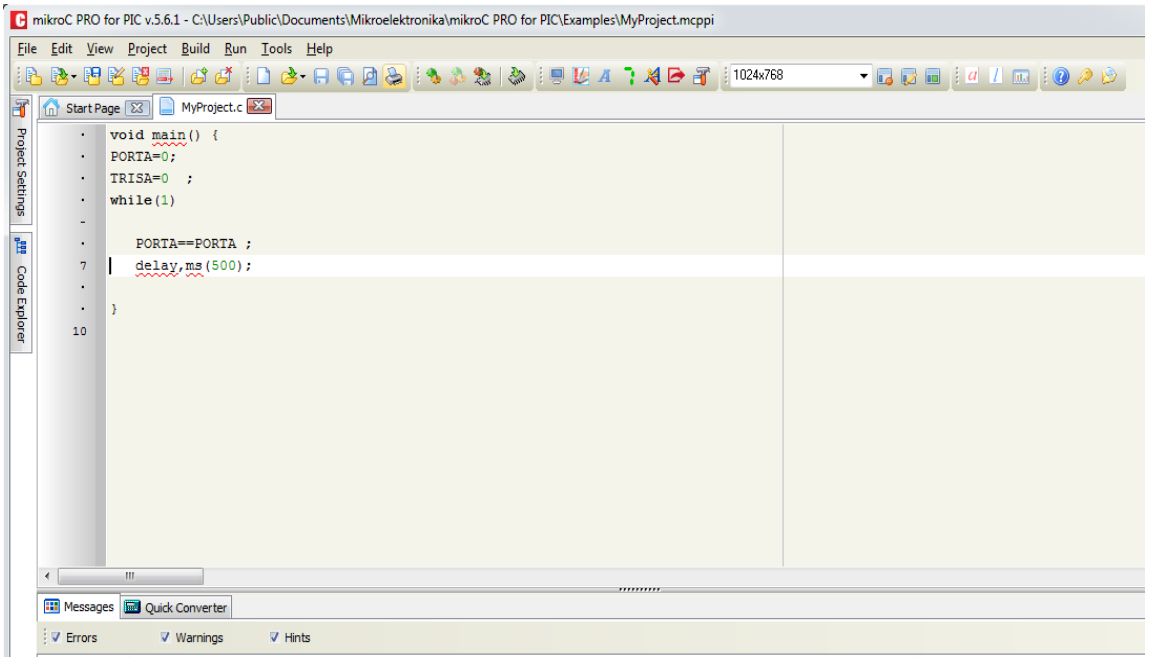


Figure III.10 : exemple d'un programme.

Après la création du programme, on doit associer ce programme au pic en passant par les étapes suivantes :

- On revient à la zone de travail sur l'écran ISIS.
- Double clique sur le pic 16F84A.

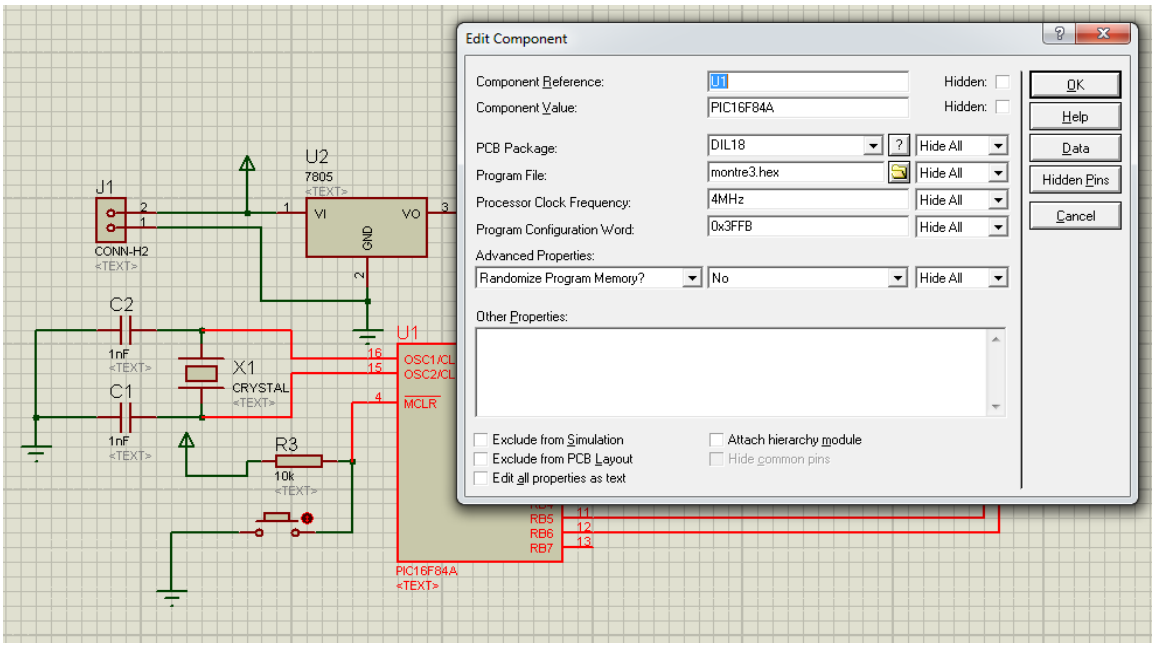


Figure III.11 : Saisie d'un programme sur ISIS Proteus.

- Entrez sur (program file) et apportez le fichier (file.HEX) et appuyez sur ok
- Le programme est chargé dans le PIC16F84.
- Cliquez sur le bouton (play) pour commencer la simulation

III.3.3 : Routage et création du circuit imprimé :

Afin de réaliser notre schéma de routage, on a utilisé le logiciel ISIS (ARES).

Le logiciel ARES est un outil d'édition et de routage qui complète parfaitement ISIS. Un schéma électrique réalisé sur ISIS peut alors être importé facilement sur ARES pour réaliser le PCB (**Printed circuit board)** de la carte électronique. Bien que l'édition d'un circuit imprimé soit plus efficiente lorsqu'elle est réalisée manuellement, ce logiciel permet de placer automatiquement les composants et de réaliser le routage automatiquement.

- L'aspect Général de ARES et très approchant de celui de ISIS. En exécutant notre logiciel on obtient la fenêtre principale, dans laquelle on va créer notre routage.

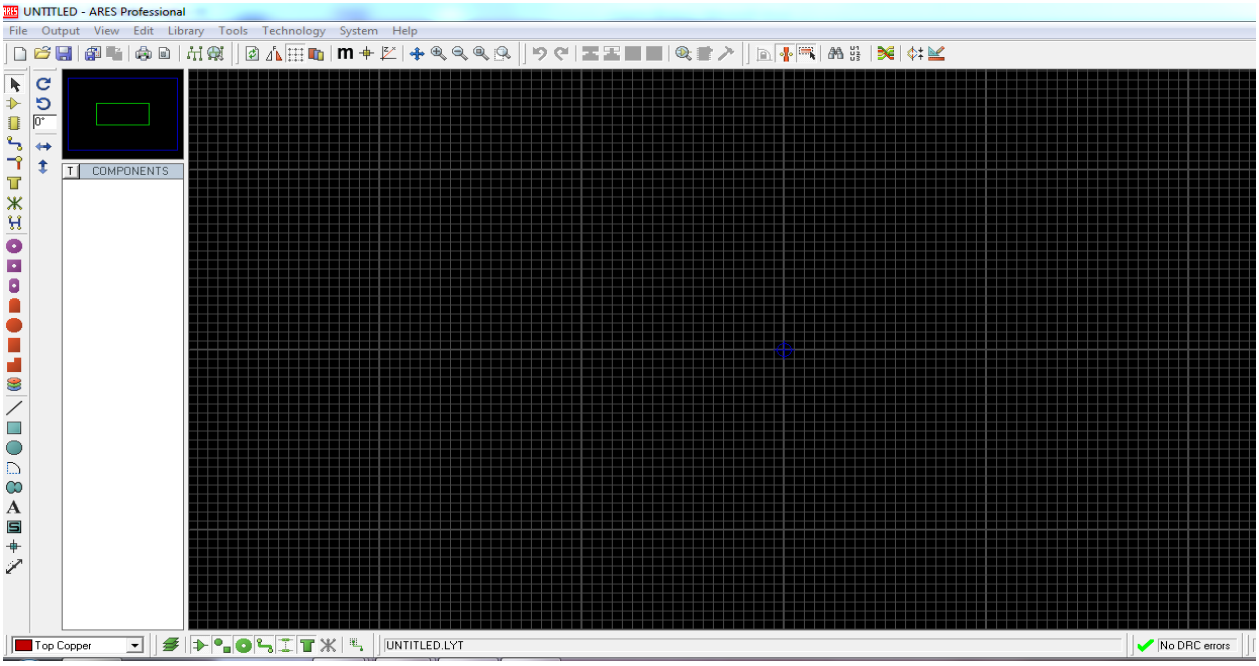


Figure III.12: Ecran d'accueil d'Ares.

- La première chose à faire avant de commencer un routage est de définir une carte aux dimensions de projet. Après avoir défini notre carte, on peut placer les composants en mode automatique ou manuelle.
- Après la validation, les composants seront positionnés sur la carte et on obtiendra notre schéma de routage suivant :

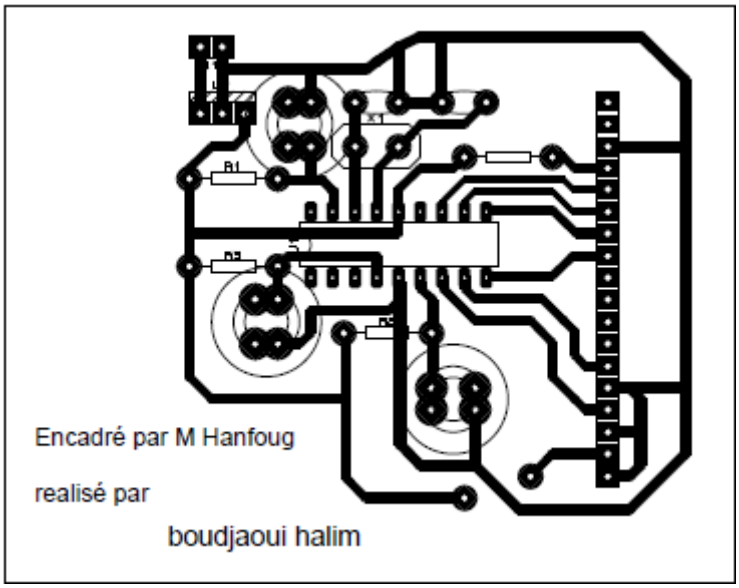


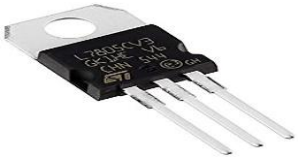
Figure III.13 : schéma de routage

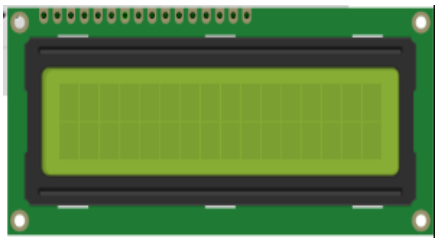
III .4 Réalisation pratique :

III.4.1 Liste des composantes :

✓ **Circuits intégrés :**

Un microcontrôleur PIC16F84A Régulateur de tension 7805 Afficheur LCD 2*16 caractères





✓ **Resistances /condensateurs :**

- **03 Resistances (10K) :**

- **02 condensateurs (22 pF):**

✓ **Alimentation / Oscillateurs** :

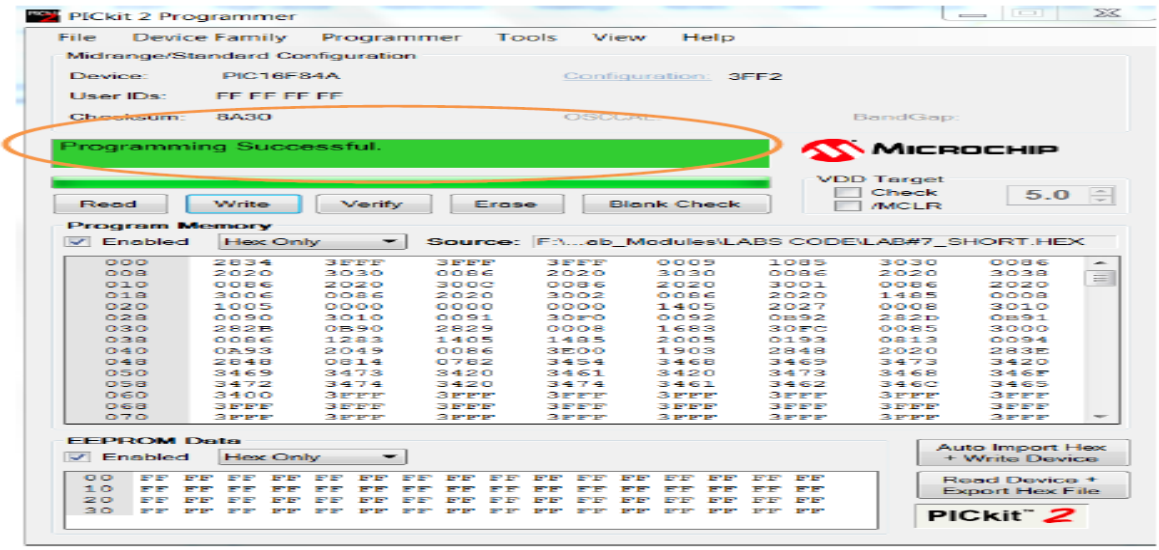
- **Batterie DC (9 v) :**

- **Quartz XTAL : f= 4MHz.**

✓ **Boutons poussoirs (03) :**

III.4.2 Gravure du programme sur le pic :

Après les tests du programme par simulation, nous pouvons transférer le programme vers le microcontrôleur pour procéduraux essais pratiques, pour cela nous avons utilisé un logiciel PICkit2 (figure). Ce programme permet en particulier de transférer le fichier compilé (hex) vers le PIC, après avoir été détecté par le logiciel, nous lui transférant le fichier concerné (hex) et enfin nous procédons à une vérification (verify) qui nous permet de bien confirmer le transfert complet du programme vers le PIC.



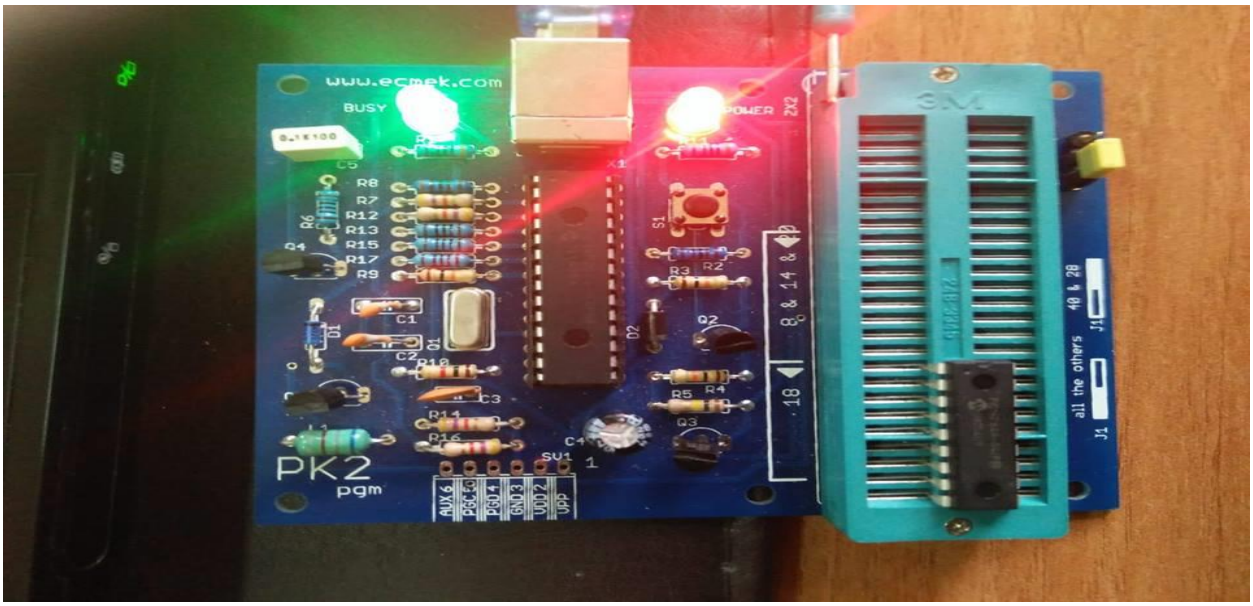


Figure III.14 : Gravure du programme sur le pic 16F84A

III.4.3 Implantation des composantes :

- **Circuit imprimé :**

A l'aide du logiciel ISIS (ARES), on obtiendra le schéma de circuit qu'on va implanter sur la carte présentée dans la figure….. Il reste alors à réaliser une photocopie laser de bonne qualité sur un transparent. On peut également passer certains transparents spéciaux directement à l'imprimante (laser ou jet d'encre).

En prévoyant une bonne petite marge, Attention, le typon est présenté côté composants, cela signifie que sa face encrée sera plaquée contre le cuivre. Après passage par l'insoleuse puis dans le perchlorure de fer, le résultat du tirage doit impérativement être impeccable, sans microcoupures ni restant de cuivre entre les pistes :

Pour terminer, on peut éventuellement étamer la carte (à froid ou à chaud) ; puis vient l'étape du perçage, avec les différents diamètres de mèches selon les composants : comme ordre d'idée, disons généralement 0,8 mm pour les résistances, les condensateurs, les supports, les circuits intégrés,…etc.

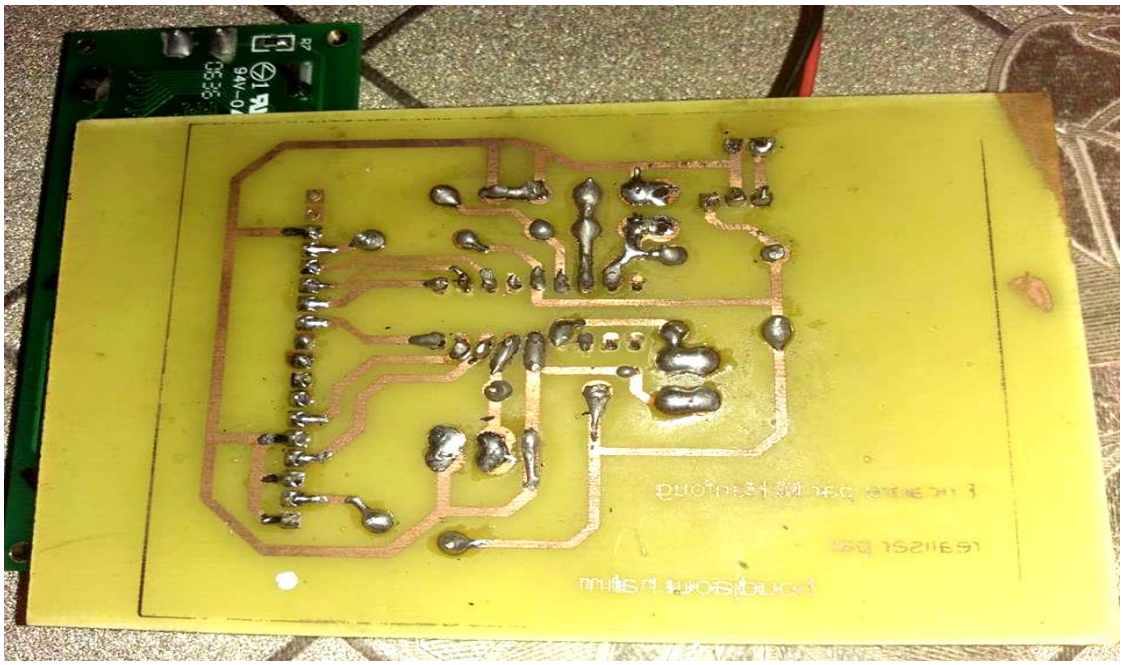


Figure III.15 : image réelle du circuit imprimé.

- **Soudure et implémentation des composantes :**

On insère les circuits intégrés sur leurs supports. Le PIC16F84A aura été préalablement programmé, Connectez l'afficheur LCD, le clip de pile (le fil rouge est le "plus", le fil noir le "moins").

On assure une ultime fois que les précautions habituelles ont été prises :

- ✓ Les composants polarisés on été soudés dans le bon sens (résistances, condensateurs).
- ✓ Les circuits intégrés ont également été insérés dans le bon sens, sans broche tordue.

- ✓ Les valeurs des composants ont bien été respectées.
- ✓ Toutes les soudures ont été faites, aucune ne "bave" sur une piste voisine ou sur une autre soudure. Pas de microcoupures au niveau des pistes, ni de restant de cuivre court-circuitant deux pistes.
- ✓ On vérifiera avec une attention particulière les pistes fines passant entre deux broches des circuits intégrés (le PIC16F84A, l'afficheur).
- ✓ Alimentez les différents composants du circuit

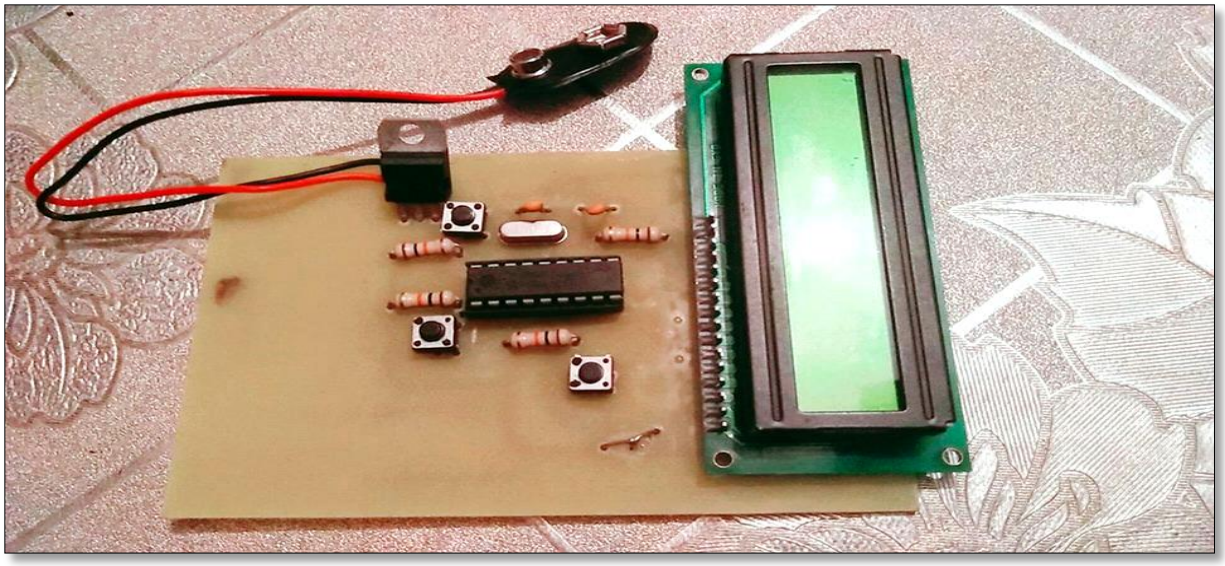


Figure III.16 : image réel de la carte.

III.5 Conclusion :

Nous avons décrit dans ce chapitre les différentes étapes de conception et réalisation des différentes parties de notre carte. Nous avons aussi présenté l'organigramme pour bien comprendre le principe de fonctionnement de notre système. En outre, on a pu programmer le PIC16F84A à l'aide d'un compilateur microC et le convertisseur pickit2.

Conclusion Générale :

Les microcontrôleurs permettent de réaliser des applications diverses qui peuvent servir dans différents domaines industriels ou pour les projets de fin d'étude.

Ce projet nous a beaucoup appris sur les microcontrôleurs et leurs programmations. Il a été mené d'une recherche bibliographique bien détaillée.

L'élaboration de ce travail, nous a permis d'approfondir nos connaissances théoriques et d'acquérir une bonne expérience au niveau de la réalisation pratique.

Lors de cette manipulation, nous avons réalisé une montre digitale à base d'un microcontrôleur PIC16F84A. Ce projet nous a permis d'enrichir nos connaissances dans le domaine électronique :

- La conception et l'assimilation des cartes et des circuits imprimés.
- La réalisation des cartes d'interfaces.
- La programmation des microcontrôleurs PIC en langage C.
- La mise en pratique des connaissances théoriques en électronique.
- La manipulation des différents modules.

Nous avons aussi appris des nouvelles connaissances au niveau de l'importance de la mesure et la gestion du temps et des équipements.

Ce travail reste, comme toute œuvre humaine, incomplète et perfectible, nous recommandons d'en améliorer la conception et pour cela nous proposons ci-dessous des améliorations pour les futurs développements :

- Synchronisation.
- Ajouter la date et l'année, en utilisant un PIC plus puissant.

Bibliographie

[1] - N.NASRI, Affichage sur cristaux liquide LCD **:** https://www.blogmatlab.blogspot.com, Consulté le 16 mars 2018.

[2] - PIC16F84 de Microchip, https://fr.wikipedia.org/wiki/, consulté le 12 mars 2018.

[3]- M.Toure Mehamed Lamine, cours de Proteus professionnel, www.magoe.net/, Consulté le 12 mars 2018.

[**4] - V. TOURTCHINE**. Programmation en mikroC. Application pour les microcontrôleurs de la famille PIC, Université m'hamed bougara-boumerdes 2012.

[5]- G BERTHOME, Organisation fonctionnelle d'un système à microcontrôleur, http://gilles.berthome.free.fr.consulté ,consulté le 16 mars 2016

[6]- Alexandre Galodé ; Le PIC 16F84 : http://diablotronic.free.fr ,consulté le 16 mars 2018.

Annex I : Pic 16F84A datasheet:

MICROCHIP

PIC16F84A

18-pin *Enhanced* FLASH/EEPROM 8-Bit Microcontroller

High Performance RISC CPU Features:

- Only 35 single word instructions to learn
- All instructions single-cycle except for program branches which are two-cycle
- Operating speed: DC - 20 MHz clock input
  DC - 200 ns instruction cycle
- 1024 words of program memory
- 68 bytes of Data RAM
- 64 bytes of Data EEPROM
- 14-bit wide instruction words
- 8-bit wide data bytes
- 15 Special Function Hardware registers
- Eight-level deep hardware stack
- Direct, Indirect and relative addressing modes
- Four interrupt sources:
  - External RB0/INT pin
  - TMR0 timer overflow
  - PORTB<7:4> interrupt-on-change
  - Data EEPROM write complete

Peripheral Features:

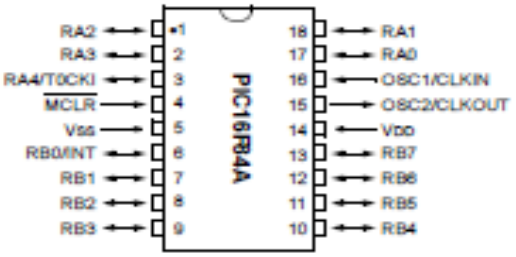
- 13 I/O pins with individual direction control
- High current sink/source for direct LED drive
  - 25 mA sink max. per pin
  - 25 mA source max. per pin
- TMR0: 8-bit timer/counter with 8-bit programmable prescaler

Special Microcontroller Features:

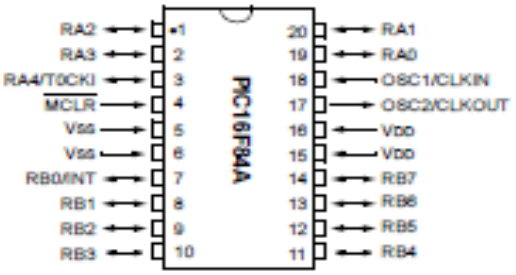
- 10,000 erase/write cycles *Enhanced* FLASH Program memory typical
- 10,000,000 typical erase/write cycles EEPROM Data memory typical
- EEPROM Data Retention > 40 years
- In-Circuit Serial Programming™ (ICSP™) - via two pins
- Power-on Reset (POR), Power-up Timer (PWRT), Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own On-Chip RC Oscillator for reliable operation
- Code protection
- Power saving SLEEP mode
- Selectable oscillator options

Pin Diagrams

PDIP, SOIC



SSOP



CMOS Enhanced FLASH/EEPROM Technology:

- Low power, high speed technology
- Fully static design
- Wide operating voltage range:
  - Commercial: 2.0V to 5.5V
  - Industrial: 2.0V to 5.5V
- Low power consumption:
  - < 2 mA typical @ 5V, 4 MHz
  - 15 µA typical @ 2V, 32 kHz
  - < 0.5 µA typical standby current @ 2V

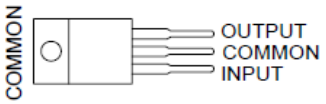
Annexe II: Régulateur 7805 datasheet :

µA7800 SERIES
POSITIVE-VOLTAGE REGULATORS

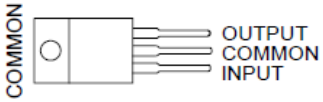
SLVS056J – MAY 1976 – REVISED MAY 2003

- 3-Terminal Regulators
- Output Current up to 1.5 A
- Internal Thermal-Overload Protection
- High Power-Dissipation Capability
- Internal Short-Circuit Current Limiting
- Output Transistor Safe-Area Compensation

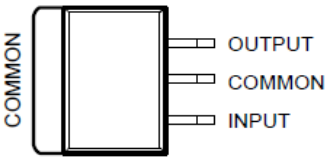
KC (TO-220) PACKAGE
(TOP VIEW)



KCS (TO-220) PACKAGE
(TOP VIEW)



KTE PACKAGE
(TOP VIEW)



description/ordering information

This series of fixed-voltage integrated-circuit voltage regulators is designed for a wide range of applications. These applications include on-card regulation for elimination of noise and distribution problems associated with single-point regulation. Each of these regulators can deliver up to 1.5 A of output current. The internal current-limiting and thermal-shutdown features of these regulators essentially make them immune to overload. In addition to use as fixed-voltage regulators, these devices can be used with external components to obtain adjustable output voltages and currents, and also can be used as the power-pass element in precision regulators.

ORDERING INFORMATION

T_J	$V_{O(NOM)}$ (V)	PACKAGE†		ORDERABLE PART NUMBER	TOP-SIDE MARKING
0°C to 125°C	5	POWER-FLEX (KTE)	Reel of 2000	µA7805CKTER	µA7805C
		TO-220 (KC)	Tube of 50	µA7805CKC	µA7805C
		TO-220, short shoulder (KCS)	Tube of 20	µA7805CKCS	
	8	POWER-FLEX (KTE)	Reel of 2000	µA7808CKTER	µA7808C
		TO-220 (KC)	Tube of 50	µA7808CKC	µA7808C
		TO-220, short shoulder (KCS)	Tube of 20	µA7808CKCS	
	10	POWER-FLEX (KTE)	Reel of 2000	µA7810CKTER	µA7810C
		TO-220 (KC)	Tube of 50	µA7810CKC	µA7810C
	12	POWER-FLEX (KTE)	Reel of 2000	µA7812CKTER	µA7812C
		TO-220 (KC)	Tube of 50	µA7812CKC	µA7812C
		TO-220, short shoulder (KCS)	Tube of 20	µA7812CKCS	
	15	POWER-FLEX (KTE)	Reel of 2000	µA7815CKTER	µA7815C
		TO-220 (KC)	Tube of 50	µA7815CKC	µA7815C
		TO-220, short shoulder (KCS)	Tube of 20	µA7815CKCS	
	24	POWER-FLEX (KTE)	Reel of 2000	µA7824CKTER	µA7824C
		TO-220 (KC)	Tube of 50	µA7824CKC	µA7824C

† Package drawings, standard packing quantities, thermal data, symbolization, and PCB design guidelines are available at www.ti.com/sc/package.

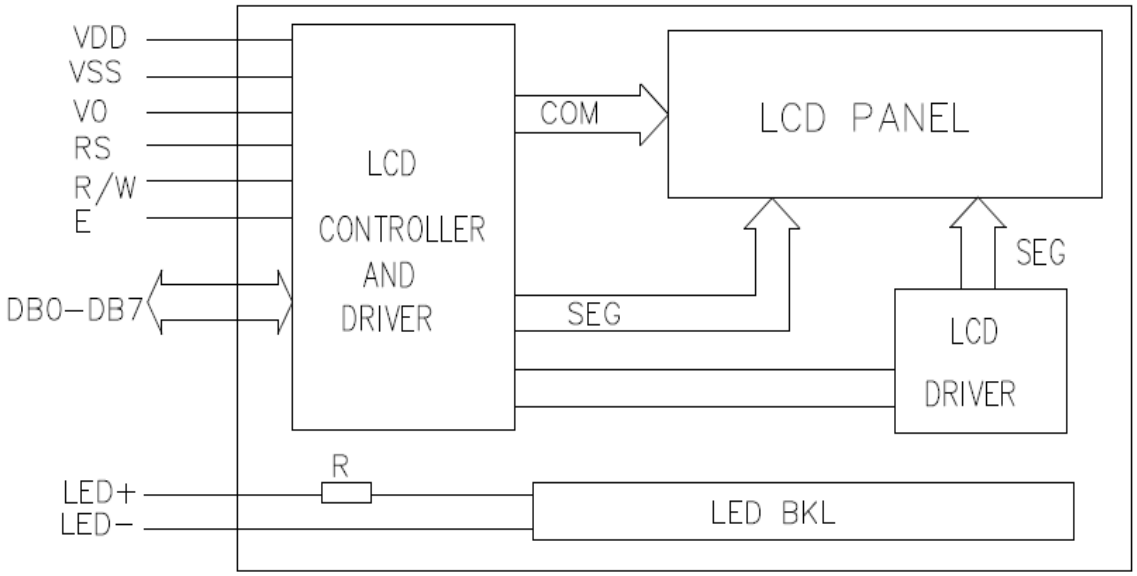
TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265

Annexe III : Afficheur LCD datasheet :

4. Absolute maximum ratings

Item	Symbol	Standard			Unit
Power voltage	V_{DD}-V_{SS}	0	-	7.0	V
Input voltage	V_{IN}	VSS	-	VDD	
Operating temperature range	V_{OP}	0	-	+50	℃
Storage temperature range	V_{ST}	-10	-	+60	

5. Block diagram



A

6. Interface pin description

Pin no.	Symbol	External connection	Function
1	Vss	Power supply	Signal ground for LCM
2	V_{DD}		Power supply for logic for LCM
3	V_0		Contrast adjust
4	RS	MPU	Register select signal
5	R/W	MPU	Read/write select signal
6	E	MPU	Operation (data read/write) enable signal
7~10	DB0~DB3	MPU	Four low order bi-directional three-state data bus lines. Used for data transfer between the MPU and the LCM. These four are not used during 4-bit operation.
11~14	DB4~DB7	MPU	Four high order bi-directional three-state data bus lines. Used for data transfer between the MPU
15	LED+	LED BKL power supply	Power supply for BKL
16	LED-		Power supply for BKL

Annexe IV : code source du microcontrôleur pic 16F84A

```
//Module de connexion de l'Afficheur LCD//
sbit LCD_RS at  RB1_bit;
sbit LCD_EN at  RB2_bit;
sbit LCD_D4 at  RB3_bit;
sbit LCD_D5 at  RB4_bit;
sbit LCD_D6 at  RB5_bit;
sbit LCD_D7 at  RB6_bit;
sbit LCD_RS_Direction at TRISB1_bit;
sbit LCD_EN_Direction at TRISB2_bit;
sbit LCD_D4_Direction at TRISB3_bit;
sbit LCD_D5_Direction at TRISB4_bit;
sbit LCD_D6_Direction at TRISB5_bit;
sbit LCD_D7_Direction at TRISB6_bit;
// Fin module de connexion de l'Afficheur LCD//
// Déclarations des variables
char h1,h2,m1,m2,s1,s2,j,d,t;
bit r;
// Affichage initial de l'heur et jour //
Void affichage()
{
Lcd_chr(2,1,h1); Lcd_chr(2,2,h2);Lcd_chr(2,3,d);Lcd_chr(2,4,m1);Lcd_chr(2,5,m2);
Lcd_chr(2,6,d); Lcd_chr(2,7,s1);Lcd_chr(2,8,s2);if(j==0){ Lcd_out(1,1,"sam");}
}
void affichagej()
{
if(j==0) Lcd_out(1,1,"sam");
```

```
if(j==1) Lcd_out(1,1,"dim");
if(j==2) Lcd_out(1,1,"lun");
if(j==3) Lcd_out(1,1,"mar");
if(j==4) Lcd_out(1,1,"mer");
if(j==5) Lcd_out(1,1,"jeu");
if(j==6) Lcd_out(1,1,"ve");
if(j==7) j=0;
}
// Sous programme du Timer0//
void interrupt()
{
if(INTCON.T0IF)
{ t++;
if(t==15)
{s2++;t=0;
if(s2==58){s1++;s2=48;}
if(s1==54){m2++;s1=s2=48;}
if(m2==57){m1++;s2=s1=m2=48;}
if(m1==54){h2++;s1=s2=m2=m1=48;}
if(h2==57){h1++;h2=48;}
if(h1==50&& h2==51){if(m1==53&& m2==57){if(s1==53&&
s2==57){s2=s1=m2=m1=h2=h1=48;j++;}}}
}
INTCON.T0IF=0;
}
//Routine d'interruption //
if(INTCON.INTF)
{r=~r;}
INTCON.INTF=0;
}
```

```
// programme principal //
void main()
//déclaration et initialisation des variables //
{ h1=h2=m1=m2=s1=s2=48;
j=r=t=0;d=58;
//déclaration des registres  //
trisa=1;
option_reg=0b01010111;
INTCON=0b10110000;
TMR0=0;
// Initialisation de l'Afficheur LCD//
Lcd_Init();
Lcd_cmd(_LCD_CLEAR);
Lcd_cmd(_LCD_CURSOR_OFF);
affichage();
for(;;)
{
// programme d'affichage //
affichagej();
affichage();
while(r==1)
{
Lcd_cmd(_LCD_CLEAR);delay_ms(400);affichage();delay_ms(400);
Lcd_cmd(_LCD_CLEAR);delay_ms(400);affichage();
while(porta.B0==0)
{
delay_ms(1000);m2++;
if(m2==58){m2=48;m1++;}
if(m1==54){m2=m1=48;}
affichage();
```

```
}
delay_ms(2000);
Lcd_cmd(_LCD_CLEAR);delay_ms(400);affichage();delay_ms(400);
Lcd_cmd(_LCD_CLEAR);delay_ms(400);affichage();
while(porta.B0==0)
{
delay_ms(1000);h2++;
if(h2==58){h2=48;h1++;}
if(h1==50&&h2==52){h2=h1=48;}
affichage();
}
delay_ms(2000);
Lcd_cmd(_LCD_CLEAR);delay_ms(400);affichage();delay_ms(400);
Lcd_cmd(_LCD_CLEAR);delay_ms(400);affichage();
while(porta.B0==0)
{     j++;
affichagej();
affichage();
delay_ms(1000);
}
r=0;
}
}
}
```

Résumé :

Avec les progrès de la technologie, plus particulièrement de l'électronique, il n'est plus rare de nos jours de voir les appareils usuels remplacés par des appareils électroniques. Parmi eux, on a pensé aux montres digitales.

Le travail présenté dans ce mémoire est une application autour d'un microcontrôleur. Il s'agit d'une horloge et d'un calendrier numérique avec affichage sur un module LCD alphanumérique à base d'un microcontrôleur 16F84A programmé en langage C a l'aide d'un compilateur MicroC et le logiciel PICkit2m qui permet le fichier compilé (hex) vers le PIC.

Mots clés : PIC 16F84A, Afficheur LCD, Langage c, compilateur microC, logiciel PICkit 2, logiciel Isis Proteus, programme, Organigramme, Interruption**,** Registre.

Abstract:

With the progress of the technology, more particularly the electronics, it is not rare anymore nowadays to see usual devices replaced by electronic devices. Among them, we thought to the digital watches.

The work presented in this report is an application around a microcontroller. It is a clock and a digital calendar with display on a module alphanumeric LCD with a microcontroller 16F84A, scheduled in language C has the help of a compiler MicroC and the PICkit2 software, which allows of injected the compiled file (hex) on the Pic.

Keywords: PIC 16F84A, LCD display, c language, microC compiler, PICkit 2 software, Isis Proteus software, program, flowchart, interrupt, register.