Société Micro 21 Gafsa

Conception et réalisation d'un enregistreur de données

Réalisé par: Alibi Elmehdi
Jawadi Sami

Rapport de projet de fin d'études

En vue d'obtention du diplôme :
Licence appliquée en Sciences et technique d'information et de communications (LASTIC)
Université Virtuelle de Tunis

Encadré par :Ferchichi Ahmed

Année universitaire :2010/2011

و قل ربي زدني علما

طه 114

Table des matières

Liste des Figures.....5
Liste des Tables.....6
Remerciements.....1
Introduction.....4
I- L'enregistreur de données.....5
1. Définition : 5
2. Applications.....5
II- Composition du Système 7
III- Les microcontrôleurs 8
1. Généralités:.....8
2. Les avantages du microcontrôleur :.....8
3. Contenu d'un microcontrôleur : 8
IV- Les PICs 10
1. Structure d'un PIC :.....10
2. Structure minimale d'un PIC : 10
3. Identification d'un PIC :.....11
4. Le choix du microcontrôleur :.....12
5. Le microcontrôleur PIC 18F452 :.....12
5.1 .Caractéristiques générales:.....12
5.2. Schéma fonctionnel:.....13
5.3. Brochage du 18F452 :.....14
5.4. Organisation du 18F452 :.....14
5.5. Les ports entrée/sortie :.....15
5.6. Le convertisseur :.....16
5.7. L'oscillateur *:* 16
5.8. MCLR *:* 16
V- Afficheur à cristaux liquides (LCD).....17
1- Introduction :.....17
2- Schéma fonctionnel : 17
3- Brochage :.....18
4- La mémoire 19
5- Commande d'un afficheur LCD 19
VI- Module horloge temps réel.....22

1- Caractéristiques électriques : 22
VII- Bus I2C....23
1- Présentation....23
2- Principe d'un échange de données....24
3- Formats de transmission 26
VIII- Carte SD....27
1. Définition: 27
2. Le bus SPI 28
IX- Etude des différentes parties du système 31
1- Implantation du circuit horloge....31
2- Implantation de l'afficheur LCD....31
3- Implantation des entrées analogiques et numériques 33
4- Implantation de la carte mémoire....34
5- Circuit d'alimentation....35
X- Simulation 37
XI- Routage et création du circuit imprimé....39
XII- Partie Logicielle....42
1- Introduction :....42
1.1- Avantages du C....42
1.2- Inconvénients du C 42
2- Présentation du MikroC :....43
XIII- Logiciel embarquée dans le PIC 44
1- Organigramme du logiciel : 44
2- Initialisation : 45
3- Gestion de l'horloge :....45
4- Ecriture dans l'horloge:....46
5- Gestion des fichiers : 47
6- Les interruptions....49
7- Les entrées analogiques : 51
XIV- Description du fonctionnement de l'enregistreur de données....52
XV- Logiciel de visualisation : 55
XVI- L'enregistreur de données en pratique 56
CONCLUSION....57
Annexes 59
Liste de composants 60

Liste des Figures

Figure 1:Schéma synoptique de la carte.....7
Figure 2 : Structure interne d'un microcontrôleur.....9
Figure 3 : Schéma fonctionnel d'un PIC.....13
Figure 4 : Brochage du 18F452.....14
Figure 5 : schéma fonctionnel d'un LCD.....17
Figure 6: Photo d'un LCD et son brochage.....18
Figure 7: organisation interne du PCF 8583.....22
Figure 8:Schéma interne interface I2C.....24
Figure 9: principe fondamental d'un transfert Bus I2C.....24
Figure 10: forme de signal.....25
Figure 11: le contenu du premier octet d'un Bus I2C.....26
Figure 12:Différents types de carte mémoire.....27
Figure 13: Représentation de différentes broches d'une carte SD.....27
Figure 14: Bus SPI 1 maitre 1 esclave.....29
Figure 15: BUS SPI 1 maitre N esclaves.....29
Figure 16:Schéma d'implantation du circuit d'horloge......31
Figure 17: schéma d'implantation de l'afficheur LCD.....32
Figure 18:Schéma d'implantation des entrées.....33
Figure 19: schéma de la carte mémoire.....34
Figure 20:Schéma de l'alimentation.....35
Figure 21:Schéma de l'enregistreur de données.....36
Figure 22:Fenêtre du logiciel de simulation Proteus ISIS.....37
Figure 23:Le logiciel ISIS en mode simulation du projet.....38
Figure 24: carte principale coté cuivre.....39
Figure 25:carte principale coté composants.....40
Figure 26: carte alimentation coté cuivre.....41
Figure 27: carte alimentation coté composants.....41
Figure 28:Interface du logiciel MikroC.....43
Figure 29:Module LCD.....52
Figure 30: Mode affichage.....53
Figure 31: Mode Trigger externe.....53
Figure 32: Mode Trigger externe avec 4 données.....53
Figure 33: Mode échantillonnage sur demande.....54
Figure 34: Mode enregistrement sur intervalle.....54
Figure 35: Choix de nombre de chaines.....54
Figure 36 : Copie écran du logiciel d'exploitation des données.....55
Figure 37 : Exemple d'une courbe.....55
Figure 38 : Carte principale et carte alimentation.....56
Figure 39 : Boitier.....56

Liste des Tables

Tableau 1: Caractéristiques du PIC 18F452 12
Tableau 2: Brochage d'un Afficheur LCD 18
Tableau 3 : Brochage d'une carte mémoire 28

Remerciements

En premier lieu, nous tenons à remercier et a exprimer nos gratitudes à notre encadreur Mr **Ferchichi Ahmed** qui nous a orienté et nous a soutenu durant notre projet de fin d'études.

Nous témoignons reconnaissance et gratitudes aux aimables personnes, quelque soit de prés ou de loin, qui nous ont soutenu, acceptent et orientant durant notre projet de fin d'étude.

Nous Remercions Mr **Abdelwaheb Essoud**, qui nous a infiniment aidé dans notre travail.

Enfin, nous remercions vivement les membres de jury.

Cahier des Charges

La société Micro 21 nous a chargé d'étudier et de concevoir un enregistreur de données, qui permet de prendre des mesures de façon automatique et qui enregistre les données sur un support magnétique.

La société Micro 21 utilise ce type de dispositif dans la surveillance de différents types de systèmes électriques.

Les consignes pour cette étude sont :

- Affichage sur écran LCD des données en temps réel.
- Réglage des paramètres du dispositif pour permettre de choisir les intervalles de mesure.
- Enregistrement sur carte mémoire (MMC).
- Données horodatées : enregistrement des données avec leurs dates et heures.
- Prise d'échantillon de données sur demande.
- Un minimum de deux chaines d'échantillonnage est requis.

Nous allons travailler sur cette base, pour pouvoir satisfaire ces conditions.

Etude Théorique

Introduction

Dans les milieux industriels, il est nécessaire d'améliorer et optimiser les processus de production. Ainsi pour atteindre cet objectif on doit surveiller continuellement les différents paramètres du système étudié et analyser les données recueillis.

Un système de surveillance de données dans un milieu industriel doit pouvoir proposer la possibilité d'enregistrer une large gamme de données.

Dans ce cadre se place notre travail qui consiste à concevoir et réaliser un enregistreur de données.

La société Micro21 est une petite société, sise à Gafsa, qui réalise des études dans le domaine informatique et électronique en essayant de proposer des solutions adéquates pour les besoins de ses clients.

Nous allons travailler selon une méthode de travail bien déterminée :

1- Etude théorique des différents systèmes qui composent notre enregistreur de données.
2- Conception et simulation du système.
3- Réalisation pratique et diagnostique.

Ce rapport reflète et décrit le travail effectué.

I- L'enregistreur de données

1. Définition :

Un **enregistreur de données** est un système électronique qui permet d'enregistrer des valeurs de mesure individuelles et des séries de mesure sur une longue période.

Les grandeurs sont mesurées de façon totalement automatique, et numérisées et enregistrées sur un support souvent numérique.

Les données peuvent être affichées sur un afficheur .et peuvent être transmises à un ordinateur, où il existe un logiciel qui permet de visualiser les données et de les analyser.

Selon le type d'enregistreur de données, les grandeurs mesurées peuvent être : Température, humidité, tension, intensité, vitesse, pression, position GPS …etc..

Le plus souvent ces enregistreurs de données sont utilisés dans les milieux industriels pour pouvoir contrôler les systèmes et pouvoir les diagnostiquer.

Selon l'application d'un enregistreur de données, il peut être autonome ou alimenté par batterie ou par panneaux solaires (exemple : système de mesure en milieu externe).

Souvent ces enregistreurs sont menu d'un système d'affichage de type LCD qui permet d'accéder à un menu de configuration, et d'afficher les valeurs mesurées.

Un tel système doit disposer d'un dispositif de stockage qui permet d'enregistrer les mesures effectuées, ce dispositif peut être une mémoire interne ou une mémoire externe (carte mémoire, flash disk , disquette …).

La connexion de l'enregistreur au PC se fait par interface série (RS 232) ,USB, LAN, Bluetooth, etc.…

2. Applications

Voici une liste non limitative d'applications:

- En laboratoires, en production (procédés de fabrication) ;
- Capteur météorologique ;
- Diagnostic médical (holter cardiaque...) ;
- Comptage du trafic routier ;
- Sur véhicules (diagnostic, développement (prototypes)) ; « boîte noire » pour certaines automobiles ;
- Surveillance d'ouvrages (ponts, barrages...), comme Jauge de déformation ;

- Recherche sur la faune ;
- Surveillance de postes électriques ;

II- Composition du Système

Commençons d'abord par donner le schéma synoptique de notre enregistreur de données

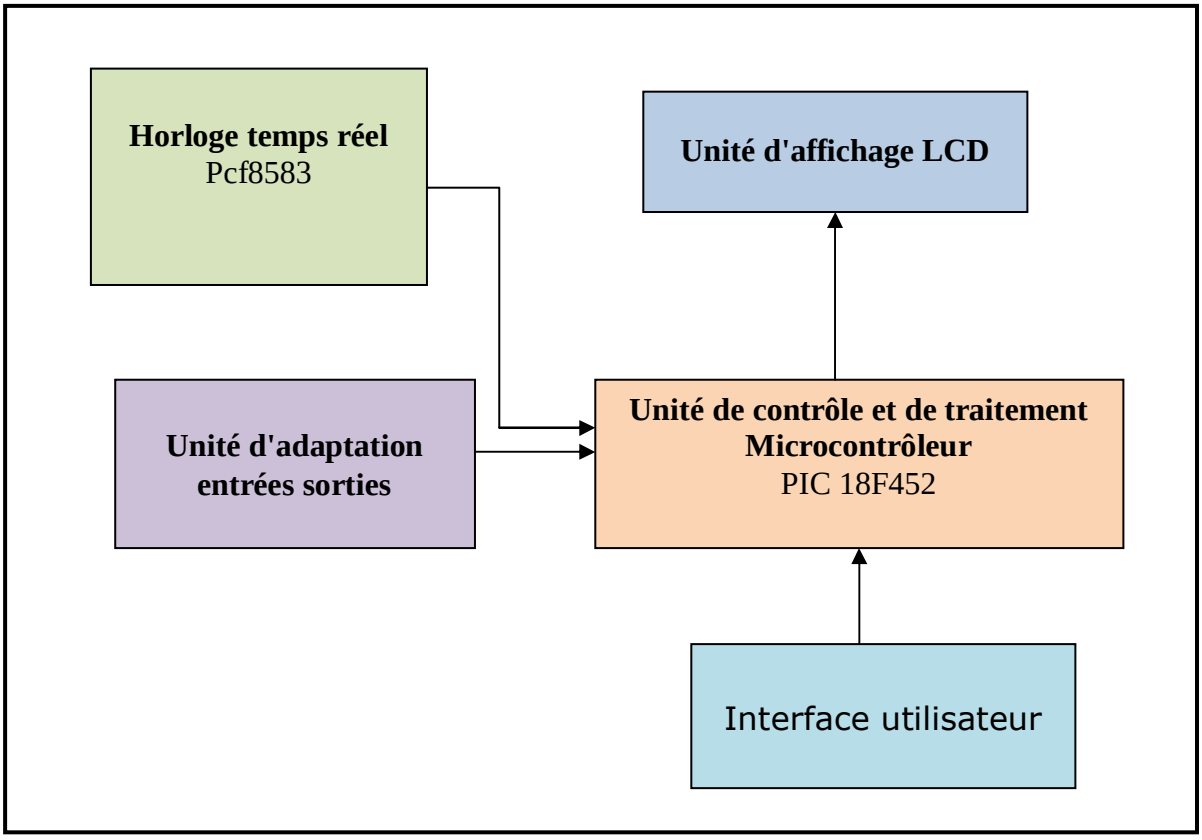


Figure 1:*Schéma synoptique de la carte*

En effet, on trouve :

- l'unité de traitement et de contrôle des données qui est le microcontrôleur (PIC 18F452).
- Un afficheur LCD est lié directement à la carte pour afficher les données.
- Un clavier composé de trois touches de commande pour le défilement des instructions contenues dans l'afficheur.
- Une horloge temps réel qui joue le rôle d'un vrai calendrier.

III- Les microcontrôleurs

1. Généralités:

Un microcontrôleur est une unité de traitement de l'information de type microprocesseur contenant tous les composants d'un système informatique, à savoir microprocesseur, des mémoires et des périphériques (ports, timers, convertisseurs…). Chaque fabricant a sa ou ses familles de microcontrôleur. Une famille se caractérise par un noyau commun (le microprocesseur, le jeu d'instruction…).

2. Les avantages du microcontrôleur :

L'utilisation des microcontrôleurs pour les circuits programmables à plusieurs points forts. Il suffit pour s'en persuader, d'examiner la spectaculaire évolution de l'offre des fabricants de circuits intégrés en ce domaine depuis quelques années.

Nous allons voir que le nombre d'entre eux découle du simple sens.

- Tout d'abord, un microcontrôleur intègre dans un seul et même boîtier ce qui, avant nécessitait une dizaine d'éléments séparés. Il résulte donc une diminution évidente de l'encombrement de matériel et de circuit imprimé.
- Cette intégration a aussi comme conséquence immédiate de simplifier le tracé du circuit imprimé puisqu'il n'est plus nécessaire de véhiculer des bus d'adresses et de donnée d'un composant à un autre.
- L'augmentation de la fiabilité du système puisque, le nombre des composants diminuant, le nombre des connexions composants/supports ou composants/circuits imprimer diminue.
- Le microcontrôleur contribue à réduire les coûts à plusieurs niveaux :
  - Moins cher que les autres composants qu'il remplace.
  - Diminuer les coûts de main d'œuvre.
- Réalisation des applications non réalisables avec d'autres composants.

3. Contenu d'un microcontrôleur :

Un circuit microcontrôleur doit contenir dans un seul boîtier tous Les éléments de bases qu'on verra par la suite. En effet, pour l'analyse des divers systèmes réalisés avant l'avènement des microcontrôleurs, les fabricants des circuits intégrés a affinées un peu la définition de ce qu'il fallait intégrer pour arriver à un schéma type analogue à la figure suivante :

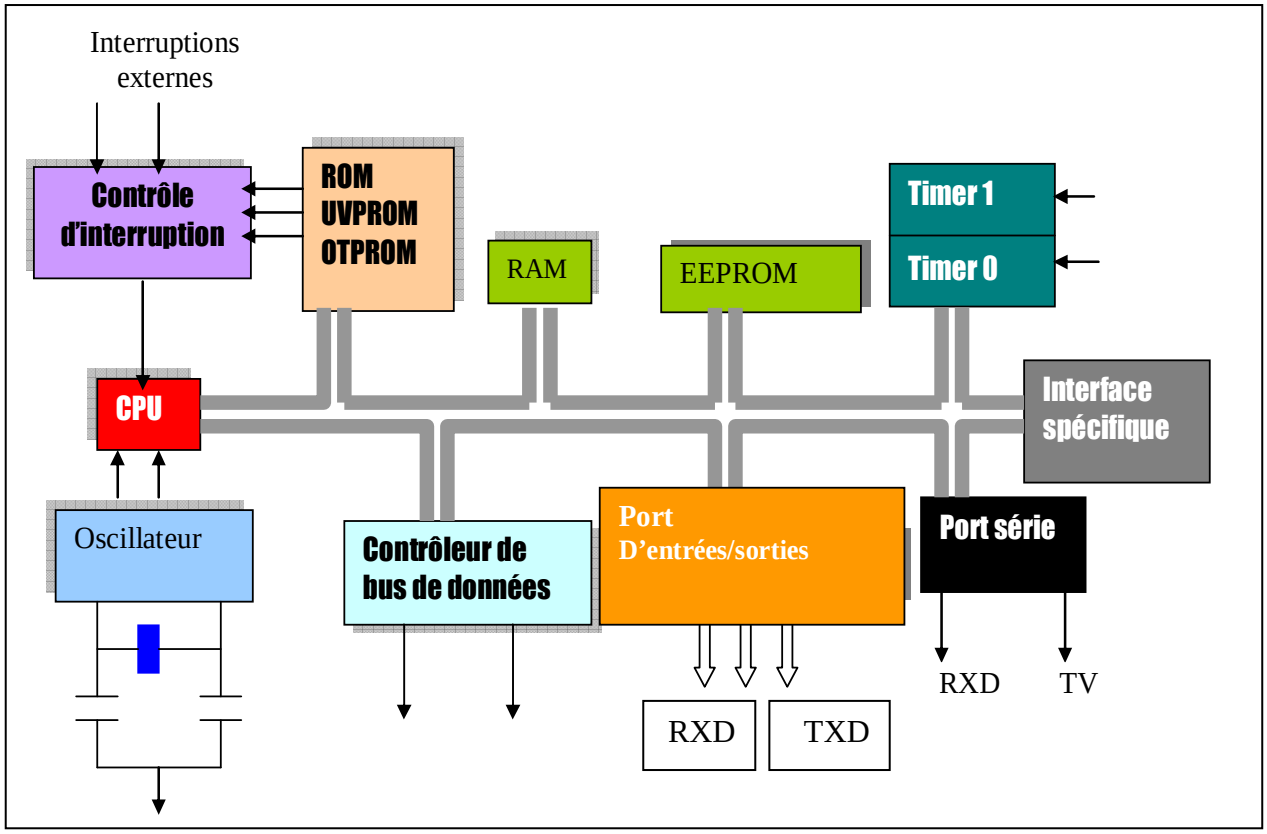


Figure 2 : *Structure interne d'un microcontrôleur*

IV- Les PICs

1. Structure d'un PIC :

Les PIC, au même titre que les microprocesseurs, sont composés essentiellement de registres ayant chacun une fonction bien définie. Les PIC possèdent également des périphériques intégrés, tels qu'une mémoire EEPROM, un timer, des ports d'entrées/ sorties ou bien encore un convertisseur analogique/numérique.

Selon le type de PIC utilisé, on retrouvera en interne un certain nombre de registres et périphériques possédant des caractéristiques différentes. Les différences de caractéristiques selon le PIC utilisé sont :

- La taille de la RAM interne ;
- La mémoire EEPROM intégrée ;
- Le type de mémoire programme : FLASH, EPROM ou OTP et la taille de celle-ci.
- Le timer intégré ;
- Les convertisseurs analogique/numérique intégrés.

2. Structure minimale d'un PIC :

La structure minimale d'un PIC est constituée des éléments ci-dessous :

- Une mémoire de programme contient le code binaire correspondant aux instructions que doit exécuter le microcontrôleur. La capacité de cette mémoire est variable selon les PIC
- Une mémoire RAM sauvegarde temporairement des données. Sa capacité est aussi variable selon les PIC
- Une Unité Arithmétique et Logique (UAL ou ALU en anglais) est chargée d'effectuer toutes les opérations arithmétiques de base (addition, soustraction, etc.) ainsi que les opérations logiques de base (ET, OU logique, etc.).
- Des ports d'entrées/sorties permettent de dialoguer avec l'extérieur du microcontrôleur, par exemple pour prendre en compte l'état d'un interrupteur (entrée logique), ou encore pour commander un relais (sortie logique).
- Un registre compteur de programme (CP ou PC en anglais), est chargé de pointer l'adresse mémoire courante contenant l'instruction à réaliser par le microcontrôleur. Le contenu du registre PC évolue selon le pas de programme.

- Un registre pointeur de pile (PP ou SP en anglais) est essentiellement utilisé lorsque l'on réalise un sous-programme. Le pointeur de pile est chargé de mémoriser l'adresse courante que contient le compteur de programme avant le saut à l'adresse du sous-programme. Lorsque le sous-programme est terminé, le pointeur restitue l'adresse sauvegardée vers le compteur de programme.
- Un registre d'instruction contient tous les codes binaires correspondant aux instructions à réaliser par le microcontrôleur. Le PIC 18F452 comporte 35 instructions.
- Un registre d'état est en relation avec l'UAL et permet de tester le résultat de la dernière opération effectuée par le microcontrôleur. Selon la dernière opération effectuée, des bits sont positionnés dans le registre d'état et ceux-ci peuvent être testés à l'aide d'une instruction de branchement pour effectuer des sauts conditionnels.
- Une horloge système permet de cadencer tous les échanges internes ou externes au microcontrôleur.

La famille des PICs est subdivisée en 3 grandes familles : La famille Base-Line, qui utilise des mots d'instructions de 12 bits, la famille Mid-Range, qui utilise des mots de 14 bits, et la famille High-End, qui utilise des mots de 16 bits.

3. Identification d'un PIC :

Pour identifier un PIC, nous utilisons simplement son numéro : les 2 premiers chiffres indiquent la catégorie du PIC, **16** indique un PIC Mid-Range, 18 indique PIC high-Range.

Vient ensuite parfois une lettre **L** : Celle-ci indique que le PIC peut fonctionner avec une plage de tension beaucoup plus tolérante. Ensuite, nous trouvons : **C** indique que la mémoire programme est une EPROM ou plus rarement une EEPROM, **CR** pour indiquer une mémoire de type ROM ou **F** pour indiquer une mémoire de type *FLASH*.

À ce niveau, on rappelle que seule une mémoire FLASH ou EEPROM est susceptible d'être effacée.

Finalement nous trouvons sur les boîtiers le suffixe « -XX » dans lequel XX représente la fréquence d'horloge maximale que le PIC peut recevoir. Par exemple –04 pour un 4MHz.

Notons dès à présent que les PICs sont des composants STATIQUES, c'est à dire que la fréquence d'horloge peut être abaissée jusqu'à l'arrêt complet sans perte de données et sans dysfonctionnement

4. Le choix du microcontrôleur :

Le choix d'un microcontrôleur est primordial car c'est de lui que dépendent en grande partie les performances, la taille, la facilité d'utilisation et le prix du montage. En fait ce choix est imposé dans le cahier de charge.

Le PIC 18F452 possède en plus des instructions très puissantes donc un programme à développer réduit, une programmation simple grâce au mode série. En fait la cause principale du choix du 18F452 est qu'il dispose de l'option du convertisseur A/D pour satisfaire côté acquisition.

5. Le microcontrôleur PIC 18F452 :

Nous allons maintenant s'intéresser à la structure interne du PIC 18F452, avec lequel nous avons travaillé.

Le 18F452 est un microcontrôleur de MICROCHIP, fait partie intégrante de la famille des High Range (18) dont la mémoire programme est de type flash (F) de type 452 et capable d'accepter une fréquence d'horloge maximale de 40Mhz.

5.1 .Caractéristiques générales:

PIC	FLASH	RAM	EEPROM	I/O	A/D	PORT //	Port série
18F452	32K	1536	256	32	10bits	NON	USART/ MSSP

Tableau 1: *Caractéristiques du PIC 18F452*

5.2. Schéma fonctionnel:

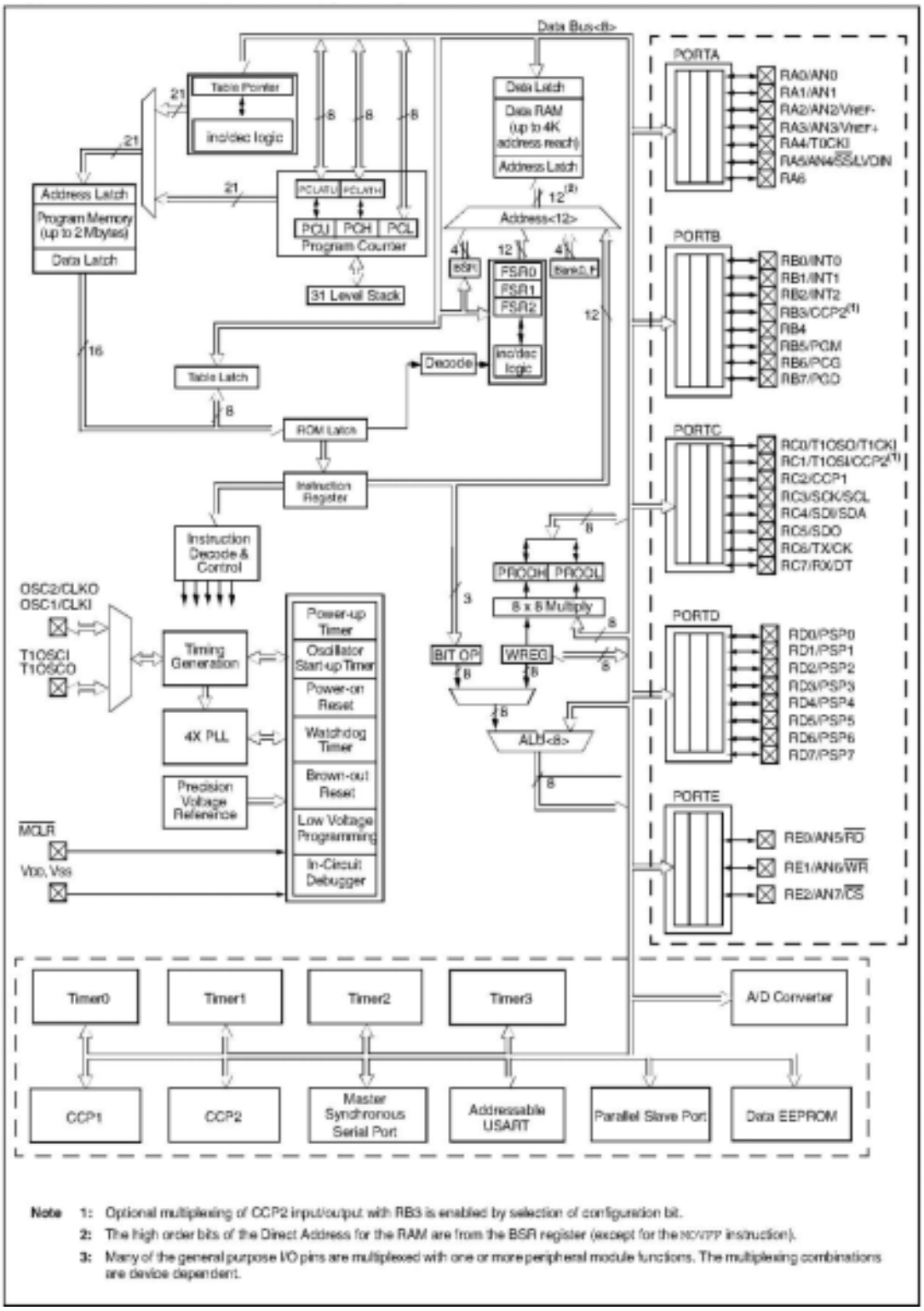


Figure 3 : *Schéma fonctionnel d'un PIC*

5.3. Brochage du 18F452 :

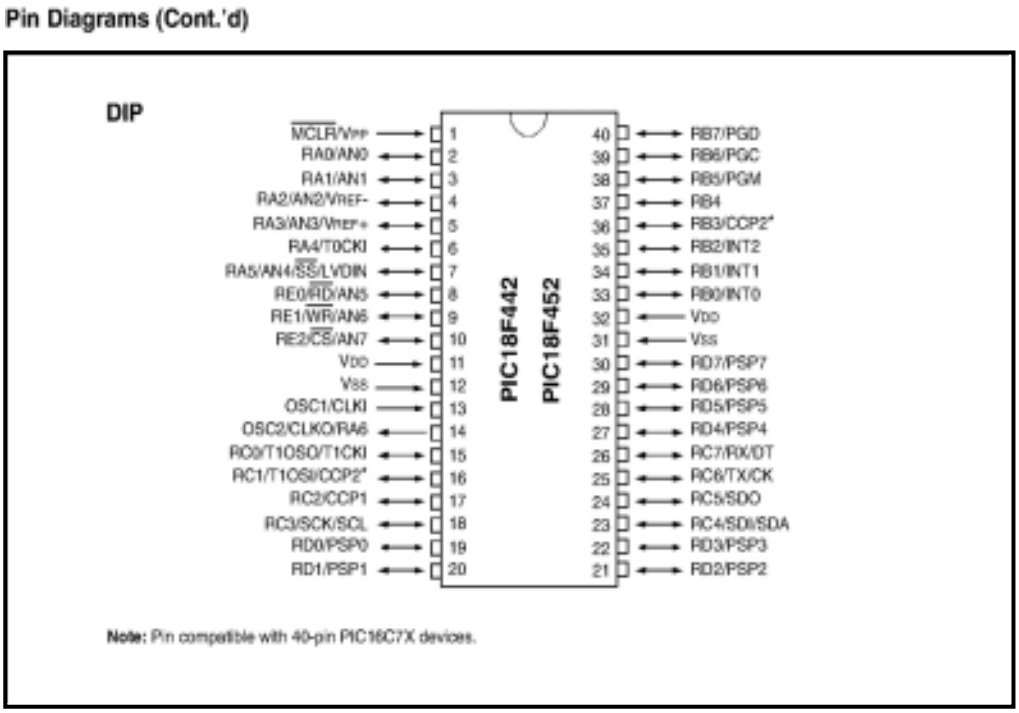


Figure 4 : *Brochage du 18F452*

5.4. Organisation du 18F452 :

La mémoire du 18F452 est divisée en 3 parties Pour mieux le connaître regardons de près son Schéma fonctionnel.

- **La mémoire programme :**

La mémoire programme est constituée de 32Kbits. C'est dans cette zone que nous allons écrire notre programme.

- **La mémoire EEPROM :**

La mémoire EEPROM (Electrical Erasable Programmable Read Only Memory), est constituée de 256 octets que nous pouvons lire et écrire depuis notre programme. Ces octets sont conservés après une coupure de courant et sont très utiles pour conserver des paramètres semi permanents.

- **La mémoire Ram et organisation :**

La mémoire RAM est celle que nous allons sans cesse utiliser. Toutes les données qui y sont stockées sont perdues lors d'une coupure de courant.

La mémoire RAM disponible du 18F452 est de 1536 octets.

- **Watchdog :**

Cette fonction est capable de surveiller le bon fonctionnement du programme que le micro contrôleur exécute. Le rôle du Watchdog (ou chien de garde) est de "reseter" le micro contrôleur si 1 'on ne remet pas à zéro périodiquement (à intervalle définissable) un registre interne grâce à 1 'instruction clrwdt (clear watchdog), si le programme tourne par exemple dans une boucle sans fin (qui est un bug dans le programme) la fonction de watchdog va permettre de remettre à 0 le micro contrôleur afin de relancer le programme.

- **Le TIMER :**

Un timer est un registre interne au micro contrôleur, celui-ci s'incrémente au grès d'une horloge, ce registre peut servir par exemple pour réaliser des temporisations, ou bien encore pour faire du comptage (par l'intermédiaire d'une broche spécifique : RA4/TOKI). Le PIC 18F452 possède 4 timers configurable par logiciel.

5.5. Les ports entrée/sortie :

On dispose de 32 broches d'entrées/sorties, chacune configurables soit en entrée soit en sortie (PORTA, PORTB, PORTC, PORTD).

Un registre interne au PIC, nommé TRIS, permet de définir le sens de chaque broche d'un port d'entrées/sorties. En règle générale, un bit positionné à « 0 » dans le registre TRIS donnera une configuration en sortie pour la broche concernée ; si ce bit est positionné à « 1 », ce sera une broche d'entrée.

5.5 .1. Particularité du port A :

Le 18F452 dispose de 4 canaux d'entrée analogique. Nous pouvons donc échantillonner successivement jusque 4 signaux différents avec ce composant. Les pins utilisés sont les pins AN0 à AN3.

5.5.2. Particularités du port B :

Hors de sa fonction principale autant que ports d'entrées /sorties, on note les pins RB0, RB1, RB2 qui, en configuration d'entrée, est de type « trigger de Schmitt » quand elle est utilisée en mode interruption « INT » ; La lecture simple de ces entrées se fait, de façon tout à fait classique, en entrée de type TTL. Encore il y a (RB3-RB6-RB7) qui peuvent servir dans la programmation (en mode LVP).

5.5.3. Particularités du port C :

C'est un port d'entrée sortie classique, avec deux pins qu'on utilisera plus tard dans la communication série avec le PC à travers (TX et RX) (RC6 et RC7).

5.5.4. Particularités du port D :

C'est un port d'entrée sortie classique.

5.6. Le convertisseur :

Le CAN est un périphérique intégré destiné à mesurer une tension et la convertir en nombre binaire qui pourra être utilisé par un programme.

Notre 18F452 travaille avec un convertisseur analogique/numérique qui permet un échantillonnage sur 10 bits. Le signal numérique peut donc prendre 1024 valeurs possibles. On sait que pour pouvoir numériser une grandeur, nous devons connaître la valeur minimale qu'elle peut prendre, ainsi que sa valeur maximale, Les pics considèrent par défaut que la valeur minimale correspond à leur Vss d'alimentation, tandis que la valeur maximale correspond à la tension positive d'alimentation Vdd.

5.7. L'oscillateur *:*

L'horloge système peut être réalisée soit avec un quartz , soit avec une horloge extérieure, soit avec un circuit RC. Dans ce dernier cas, la stabilité du montage est limitée.

La fréquence maximale d'utilisation va dépendre du microcontrôleur utilisé. Le suffixe indiqué sur le boîtier donne la nature de l'horloge à utiliser et sa fréquence maximale.

Notre PIC utilisé est limité à 40Mhz.

5.8. MCLR *:*

La broche MCLR permet de réaliser un Reset du circuit quand elle est placée à 0V

V- Afficheur à cristaux liquides (LCD)

1- Introduction :

Les afficheurs à cristaux liquides, 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 5 mA).

Plusieurs afficheurs sont disponibles sur le marché et diffèrent les uns des autres, par leurs dimensions, (de 1 à 4 lignes de 6 à 80 caractères), et aussi par leurs caractéristiques techniques et leur tension de service. Certains sont dotés d'un rétro-éclairage. Cette fonction fait appel à des LED montées derrière l'écran du module.

2- Schéma fonctionnel :

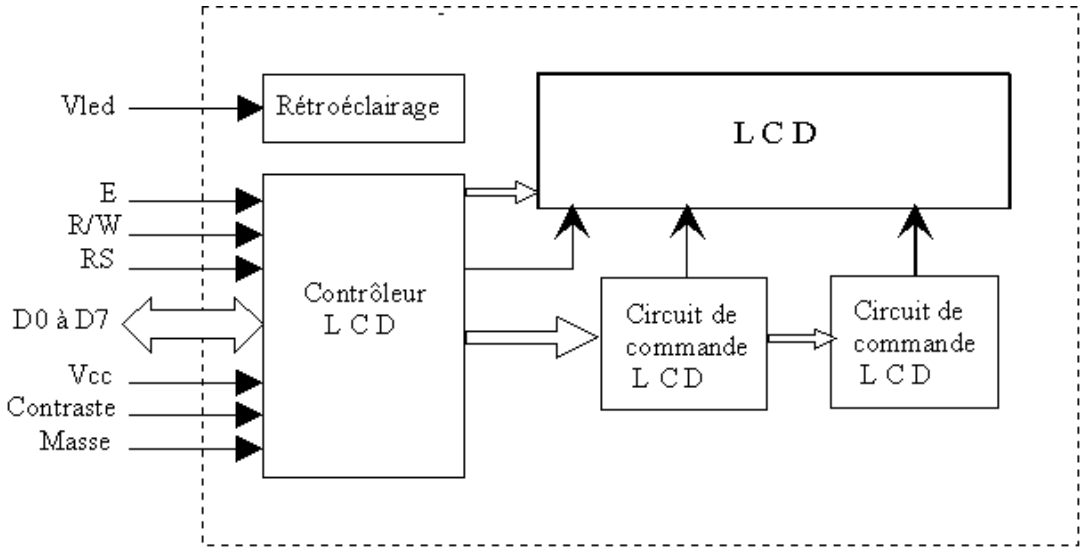


Figure 5 : *schéma fonctionnel d'un LCD*

Comme il le montre le schéma fonctionnel, l'affichage comporte d'autres composants que l'afficheur à cristaux liquides (LCD) seul. Un circuit intégré de commande spécialisé, le LCD-Controller, est chargé de la gestion du module. Le "contrôleur" remplit une double fonction: d'une part il commande l'affichage et de l'autre se charge de la communication avec l'extérieur.

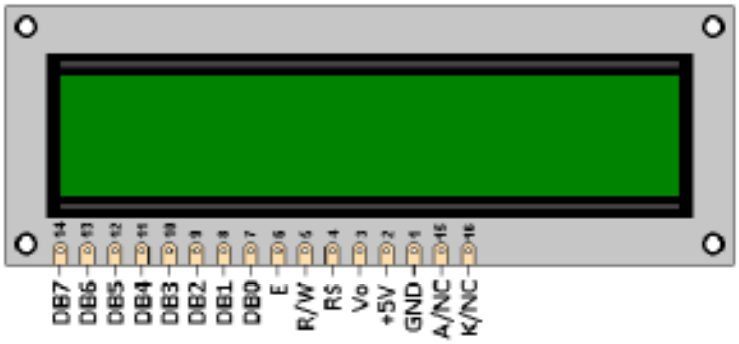


Figure 6: *Photo d'un LCD et son brochage*

3- Brochage :

Broche	Nom	Niveau	Fonction
1	Vss	-	Masse
2	Vdd	-	Alimentation positive +5V
3	Vo	0 - 5V	Cette tension permet, en la faisant varier entre 0 et +5V, le réglage du contraste de l'afficheur.
4	RS	TTL	Sélection du registre (Register Select) Grâce à cette broche, l'afficheur est capable de faire la différence entre une commande et une donnée. Un niveau bas indique une commande et un niveau haut indique une donnée.
5	R/W	TTL	Lecture ou écriture (Read/Write) L : Écriture/H : Lecture
6	E	TTL	Entrée de validation (Enable) active sur front descendant. Le niveau haut doit être maintenue pendant au moins 450 ns à l'état haut.
7	D0	TTL	
8	D1	TTL	
9	D2	TTL	
10	D3	TTL	D0 –> D7 Bus de données bidirectionnel 3 états (haute impédance lorsque E=0)
11	D4	TTL	
12	D5	TTL	
13	D6	TTL	
14	D7	TTL	
15	A	-	Anode rétro éclairage (+5V)
16	K	-	Cathode rétro éclairage (masse)

Tableau 2: *Brochage d'un Afficheur LCD*

Les broches 15 et 16 ne sont présentes que sur les afficheurs LCD avec retro-éclairage.

4- La mémoire

L'afficheur possède deux type de mémoire, la DD RAM et la CG RAM. La DD RAM est la mémoire d'affichage et la CG RAM est la mémoire du générateur de caractères.

La mémoire d'affichage (DD RAM) :

La DD RAM est la mémoire qui stocke les caractères actuellement affiché à l'écran..

La mémoire du générateur de caractères (CG RAM) :

Le générateur de caractère est quelque chose de très utile. Il permet la création d'un maximum de 8 caractères ou symboles 5x7. Une fois les nouveaux caractères chargés en mémoire, il est possible d'y accéder comme s'il s'agissait de caractères classiques stockés en ROM.

5- Commande d'un afficheur LCD

Deux modes de fonctionnement de l'afficheur sont disponibles, le mode 4 bits et le mode 8 bits, modes que l'on choisira à l'initialisation de l'afficheur (voir plus bas).

- **Mode 8 bits :**

Dans ce mode 8 bits, les données sont envoyées à l'afficheur sur les broches D0 à D7. On place la ligne RS à 0 ou à 1 selon que l'on désire transmettre une commande ou une donnée. Il faut aussi placer la ligne R/W à 0 pour indiquer à l'afficheur que l'on désire effectuer une écriture. Il reste à envoyer une impulsion d'au moins 450 ns sur l'entrée E, pour indiquer que des données valides sont présentes sur les broches D0 à D7. L'afficheur lira la donnée sur le front descendant de cette entrée.

Si on désire au contraire effectuer une lecture, la procédure est identique, mais on place cette fois la ligne R/W à 1 pour demander une lecture. Les données seront valide sur les lignes D0 à D7 lors de l'état haut de la ligne E.

- **Mode 4 bits :**

Il peut, dans certains cas, être nécessaire de diminuer le nombre de fils utilisés pour commander l'afficheur, comme, par exemple lorsqu'on dispose de très peu de broches d'entrées sorties disponibles sur un microcontrôleur. Dans ce cas, on peut utiliser le mode quatre bits de l'afficheur LCD. Dans ce mode, seuls les 4 bits de poids fort (D4 à D7) de l'afficheur sont utilisées pour transmettre les données et les lire. Les 4 bits de poids faible (D0 à D3) sont alors connectés à la masse. On a donc besoin, hors alimentation de sept fils pour commander l'afficheur. Les données sont alors écrites ou lues en envoyant séquentiellement les quatre bits de poids fort suivi des

quatre bits de poids faible. Une impulsion positive d'au moins 450 ns doit être envoyée sur la ligne E pour valider chaque demi-octet ou nibble.

Dans les deux modes, on peut, après chaque action sur l'afficheur, vérifier que celui-ci est en mesure de traiter l'information suivrante. Pour cela, il faut demander une lecture en mode commande, et tester le flag Busy BF. Lorsque BF=0, l'afficheur est près à recevoir une nouvelle commande ou donnée.

Il se peut qu'on dispose encore de moins de broches disponibles dans l'application envisagée. Dans ce cas, on peut alors relier la ligne R/W à la masse de façon à forcer l'afficheur en écriture. On a alors besoin, hors alimentation de seulement six fils en mode 4 bits, et dix fils en mode 8 bits, pour commander l'afficheur, mais on ne peut alors plus relire l'afficheur. Ceci n'est pas gênant dans la mesure où on sait ce qu'on a écrit sur l'afficheur, mais on ne peut alors plus relire le flag Busy. Il faut alors utiliser des temporisations après chaque écriture sur l'afficheur. On perd alors un peu en temps d'affichage, mais on gagne une broche d'entrée sortie.

◆ Initialisation

En mode 8 bits

À la mise sous tension de l'afficheur, la ligne supérieur devrait être totalement sombre, celle du bas complètement claire. Si tel n'était pas le cas, il faudra régler le contraste de l'afficheur en jouant sur la tension de la broche Vo.

Avant de pouvoir utiliser l'afficheur, il faut tout d'abord l'initialiser. Pour cela, la première commande à envoyer, est le commande permettant de définir le mode de dialogue avec l'afficheur (DL), et le nombre de lignes sélectionnées (N), et l'envoyer plusieurs fois de façon à ce que celle ci soit comprise, que le mode de départ soit quatre ou huit bits. On peut ensuite paramétrer l'afficheur, puis l'effacer.

Voici donc les différentes commandes (RS=0) à envoyer à l'afficheur LCD. Entre chaque valeur, il faut envoyer une impulsion d'au moins 450 ns sur la ligne E :

33h, 33h, 33h : on force le LCD en mode 8 bits (3xh)

38h : mode 8 bits, 2 lignes, caractères 5x7

0Ch : affichage en fonction, pas de curseur

06h : le curseur se déplace vers la gauche

01h : on efface l'afficheur

En mode 4 bits

Pour l'initialisation d'un afficheur en mode quatre bits, on commence par forcer celui-ci dans le mode huit bits, puis quand on est sûr que celui-ci est valide, on bascule en mode quatre bits. Comme on ne sait pas au début de l'initialisation si l'afficheur est positionné en quatre ou huit bits, il est nécessaire d'envoyer la commande de passage en mode huit bits plusieurs fois de façon à ce que celle-ci soit comprise, que le mode de départ soit quatre ou huit bits. Les données sont écrites ou lues en envoyant séquentiellement les quatres bits de poids fort suivi des quatres bits de poids faible, séparés par une impulsion positive d'au moins 450 ns sur la ligne E.

En résumé, voici sur quatre bits, les commandes (RS=0) à envoyer à l'afficheur LCD. Entre chaque valeur, il faut envoyer une impulsion positive sur la ligne E.

0h, 1h, 0h, 0h, 1h : on commence par effacer l'afficheur (01h)

3h, 3h, 3h : on force le LCD en mode 8 bits (3xh)

2h : on passe en mode 4 bits (20h)

2h, 8h : mode 4 bits, 2 lignes, caractères 5x7 (28h)

0h, Ch : affichage en fonction, pas de curseur (0Ch)

0h, 6h : le curseur se déplace vers la gauche (06h)

0h, 1h : on efface l'afficheur

VI- Module horloge temps réel

Puisque le facteur temps est primordial dans le fonctionnement de notre système nous avons pensé à l'utilisation d'un circuit intégré de la famille PCF qui servira d'horloge temps réel.

Dans la famille des circuits intégrés compatibles au bus I2C, le PCF8583 satisfait à cette fonction. Le PCF8583 fonctionne en véritable horloge calendrier c'est-à-dire en mode de 24 heures et sur une période de vingt quatre ans. Il possède une sortie d'interruption et de la RAM qui possède 240 octets disponibles en plus de ceux de sa propre fonction.

1- Caractéristiques électriques :

- Consommation faible de courant.
- Garantie de la fonction d'horloge et de rétention de mémoire sous 1v (et 2µA) ce qui permet de le secourir facilement par une batterie.

Le bloc diagramme de notre circuit nous aide à mieux comprendre les choses :

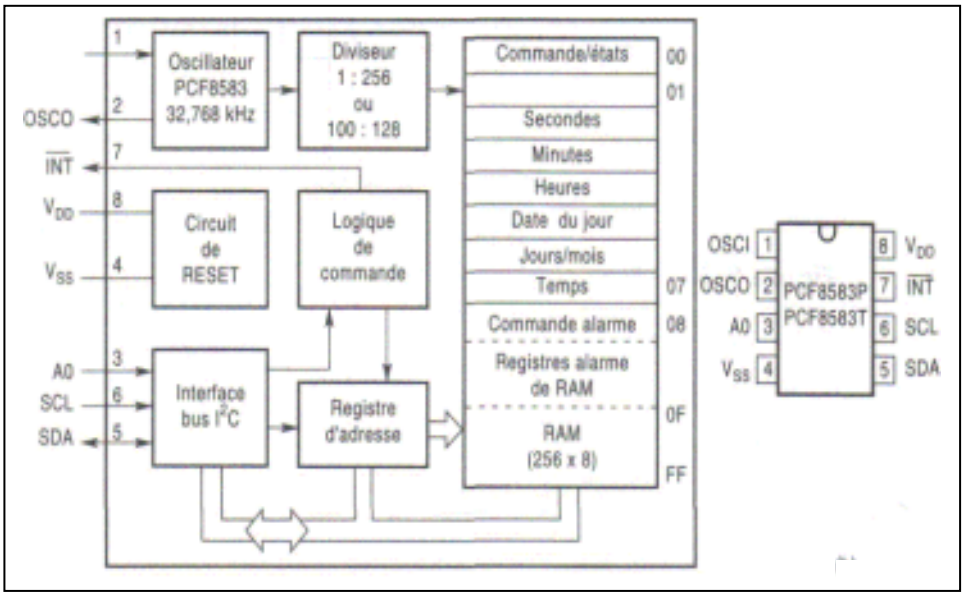


Figure 7: *organisation interne du PCF 8583*

VII-Bus I2C

Le bus I2C, dont le sigle signifie Inter Integrated Circuit ce qui donne IIC et par contraction I2C.Le protocole est initialement proposé par Philips mais adopté de nos jours par de très nombreux fabricants. C'est un bus de communication de type série.

1- Présentation

Le bus I2C qui n'utilise que deux lignes de signal permet à un certain nombre d'appareils d'échanger des informations sous forme série avec un débit pouvant atteindre 100 Kbps ou 400 Kbps pour les versions les plus récentes.

Ceci étant précisé, voici quels sont les points forts du bus I2C :

- C'est un bus série bifilaire utilisant une ligne de données appelée SDA (Serial Data) et une ligne d'horloge appelée SCL (Serial Clock) ;
- Les données peuvent être échangées dans les deux sens sans restriction ;
- Le bus est multi-maîtres ;
- Chaque abonné dispose d'une adresse codée sur 7 bits. On peut donc connecter simultanément 128 abonnés d'adresses différentes sur le même bus, sous réserve de ne pas le surcharger électriquement ;
- Un acquittement est généré pour chaque octet de donnée transféré ;
- Le bus peut travailler à une vitesse maximum de 100 Kbps (ou 400 Kbps) le protocole permet de ralentir automatiquement l'équipement le plus rapide pour s'adapter à la vitesse de l'élément le plus lent, lors d'un transfert ;
- Le nombre maximum d'abonnés n'est limité que par la charge capacitive maximale du bus qui peut être de 400 pF.
- Les niveaux électriques permettent l'utilisation de circuits en technologies CMOS, NMOS ou TTL.

2- Principe d'un échange de données

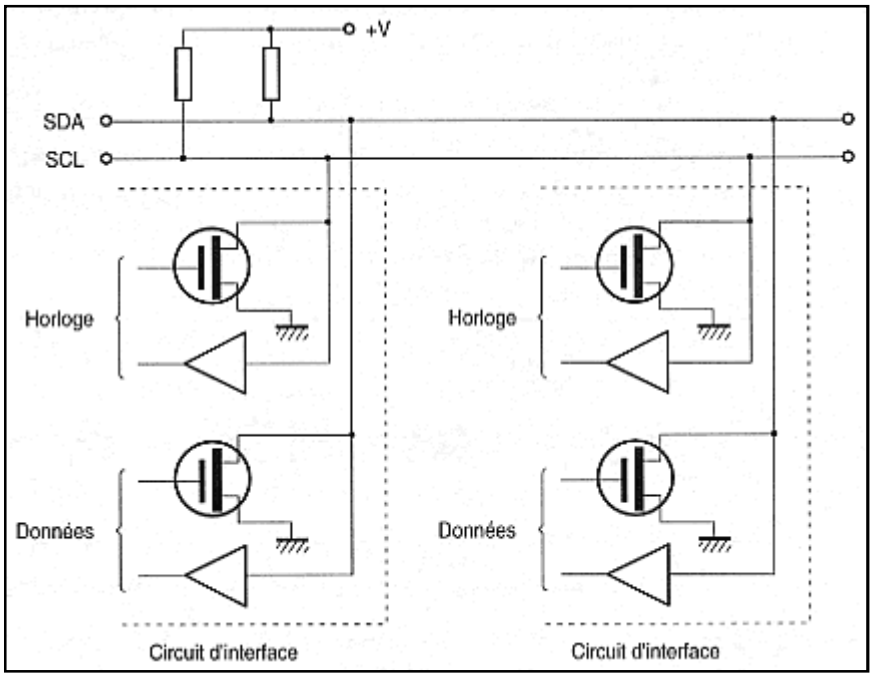


Figure 8:*Schéma interne interface I2C*

Cette figure montre le principe adopté au niveau des étages d'entrée/sortie des circuits d'interface au bus I2C.

Aucune charge n'étant prévue dans ces derniers, une résistance de rappel à une tension positive doit être mise en place. Le niveau électrique n'est pas précisé pour l'instant car il dépend de cette tension. Nous parlerons donc de niveaux logiques hauts ou « 1 » ou bien encore de niveaux logiques bas ou « 0 » étant entendu que l'on travaille en logique positive c'est-à-dire qu'un niveau haut correspond à une tension plus élevée qu'un niveau bas.

Compte tenu de ce mode de connexion en ET câblé, lorsqu'aucun abonné n'émet sur le bus, les lignes SDA et SCL sont au niveau haut qui est leur état de repos.

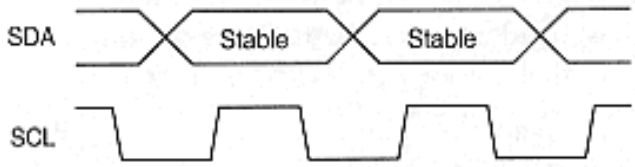


Figure 9: *principe fondamental d'un transfert Bus I2C*

Cette figure résume le principe fondamental d'un transfert à savoir : une donnée n'est considérée comme valide sur le bus que lorsque le signal SCL est à l'état haut. L'émetteur doit donc positionner la donnée à émettre lorsque SCL est à l'état bas et la maintenir tant que SCL reste à l'état haut .

Comme la transmission s'effectue sous forme série, une information de début et de fin doit être prévue. L'information de début s'appelle ici condition de départ et l'information de fin condition d'arrêt.

Une condition de départ est réalisée lorsque la ligne SDA passe du niveau haut au niveau bas alors que SCL est au niveau haut. Réciproquement, une condition d'arrêt est réalisée lorsque SDA passe du niveau bas au niveau haut alors que SCL est au niveau haut.

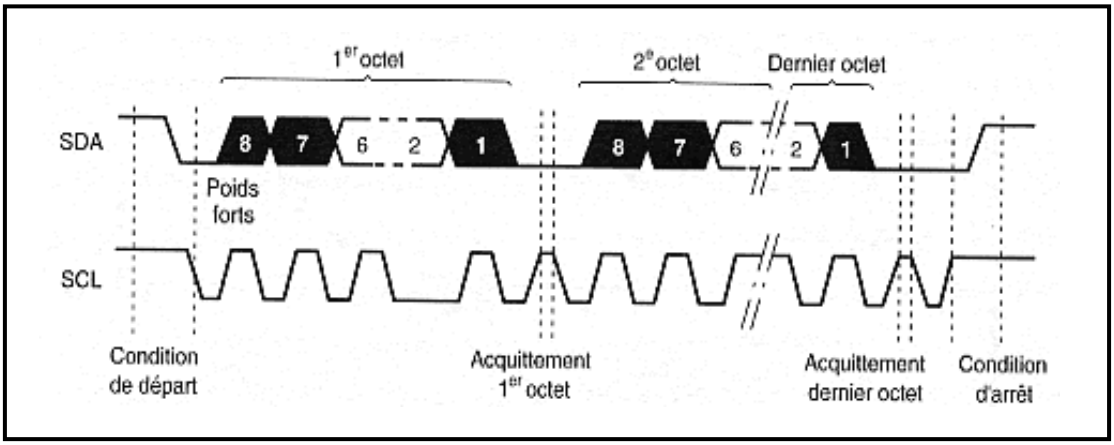


Figure 10: *forme de signal*

Bien que nous soyons en présence d'un bus série, les données sont envoyées par paquets de huit, même si un octet regroupe en fait huit bits indépendants. Le bit de poids fort est envoyé le premier.

Chaque octet est suivi par un bit d'acquittement de la part du destinataire et l'ensemble du processus fonctionne comme indiqué sur la figure 11.

Tout d'abord, sachez que lors d'un échange de ce type, la ligne SCL est pilotée par l'initiateur de l'échange ou maître, quitte à ce que l'esclave agisse également dessus dans certains cas particuliers.

La figure ci-dessus montre tout d'abord une condition de départ, générée par le maître du bus à cet instant. Elle est suivie par le premier octet de données, poids forts en tête. Après le huitième bit, l'émetteur qui est aussi le maître dans ce cas, met sa ligne SDA au niveau haut c'est-à-dire au

repos mais continue à générer l'horloge sur SCL. Pour acquitter l'octet, le récepteur doit alors forcer la ligne SDA au niveau bas pendant l'état haut de SCL qui correspond à cet acquittement, prenant en quelque sorte la place d'un neuvième bit.

Le processus peut alors continuer avec l'octet suivant et se répéter autant de fois que nécessaire pour réaliser un échange d'informations complet. Lorsque cet échange est terminé, le maître génère une condition d'arrêt.

3- Formats de transmission

Nous savons maintenant comment se déroulent les échanges ; il nous reste à examiner le format des données transmises afin de comprendre comment fonctionne l'adressage, mais aussi la définition du sens de transferts des données.

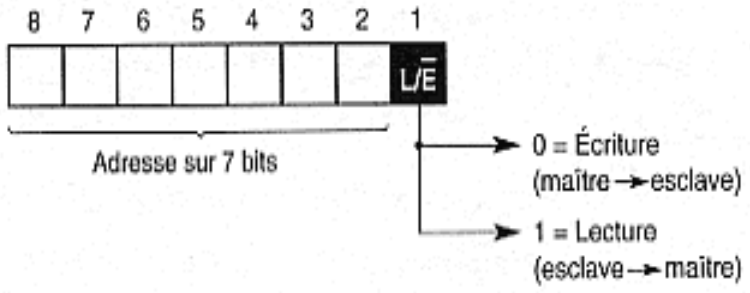


Figure 11: *le contenu du premier octet d'un Bus I2C*

Cette figure montre le contenu du premier octet qui est toujours présent en début d'échange. Ses sept bits de poids forts contiennent l'adresse du destinataire du message ce qui autorise 128 combinaisons différentes

Le bit de poids faible indique si le maître va réaliser une lecture ou une écriture. Si ce bit est à zéro le maître va écrire dans l'esclave ou lui envoyer des données. S'il est à un, le maître va lire dans l'esclave c'est-à-dire que le maître va recevoir des données de l'esclave.

Lorsqu'un maître désire effectuer plusieurs échanges à destination d'esclaves d'adresses différentes, il n'est pas obligé de terminer le premier échange par une condition d'arrêt mais peut les enchaîner en générant une condition de départ dès la fin d'un échange.

Enfin, il existe une procédure dite d'appel général où l'adresse envoyée par le maître, c'est-à-dire rappelons-le, les sept bits de poids forts du premier octet, est nulle. Tous les circuits connectés sur le bus capables de répondre à un tel appel général doivent alors le faire et prendre en compte les données qui suivent. Leur attitude dépend du bit de lecture/écriture de ce premier octet.

VIII- Carte SD

1. Définition:

Une carte SD (SD étant le sigle de l'anglais Secure Digital) est une carte mémoire amovible de stockage de données numériques créée en janvier 2000 par une alliance formée entre les industriels Panasonic, SanDisk et Toshiba.

Elle est essentiellement utilisée pour le stockage des fichiers dans les appareils photo et caméscopes numériques, les systèmes de navigation par GPS et les consoles de jeux. C'est au premier semestre 2010 la carte mémoire la plus répandue, offrant une capacité maximale de 64 Go (les capacités 2, 4,8 et 16 Go étant actuellement les plus courantes).

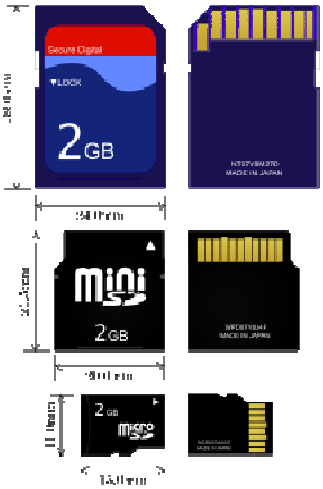


Figure 12: *Différents types de carte mémoire*

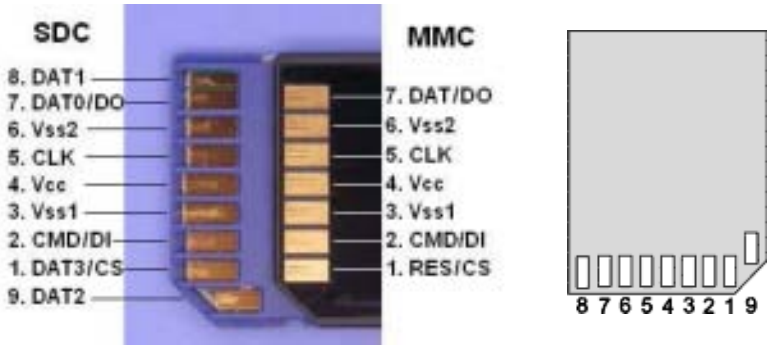


Figure 13: *Représentation de différentes broches d'une carte SD*

Ci dessous la description de chaque broche d'une carte SD

	SD Mode			SPI Mode		
PIN	**Name**	**Type**	**Description**	**Nom**	**Type**	**Description**
1	CD/DAT3	I/O/PP	Détection de la carte/ Ligne de données du connecteur 3	CS	I	Sélection de la puce à l'état bas
2	CMD	PP	Ligne de commande/Réponse	DI	I	Données d'entrée
3	Vss1	S	Tension d'alimentation (masse)	VSS	S	Tension d'alimentation
4	Vdd	S	Alimentation	VDD	S	Alimentation
5	CLK	I	Horloge	SCLK	I	Horloge
6	Vss2	S	Tension d'alimentation	VSS2	S	Tension d'alimentation
7	DAT0	I/O/PP	Ligne de donnée du connecteur 0	DO	O/PP	Données de sortie
8	DAT1	I/O/PP	Ligne de donnée du connecteur 1	RSV		
9	DAT2	I/O/PP	Ligne de donnée du connecteur 2	RSV		

Tableau 3 : *Brochage d'une carte mémoire*

Avec le PIC nous allons utiliser notre carte SD en mode SPI

2. Le bus SPI

Une liaison SPI (pour Serial Peripheral Interface) est un bus de donnée série synchrone baptisé ainsi par Motorola, et qui opère en Full Duplex. Les circuits communiquent selon un schéma maître-esclaves, où le maître s'occupe totalement de la communication. Plusieurs esclaves peuvent coexister sur un bus, la sélection du destinataire se fait par une ligne dédiée entre le maître et l'esclave appelée chip select.

Le bus SPI contient 4 signaux logiques :

- **SCLK —** Horloge (généré par le maître)

- **MOSI** — Master Output, Slave Input (généré par le maître)
- **MISO** — Master Input, Slave Output (généré par l'esclave)
- **SS** — Slave Select, Actif à l'état bas, (généré par le maître)

Il existe d'autres noms qui sont souvent utilisés.

- **SCK** — Horloge (généré par le maître)
- **SDI, DI, SI** — Serial Data IN
- **SDO, DO, SO** — Serial Data OUT
- **nCS, CS, nSS, STE** — SS

Dans le cas de la convention de nommage SDI/SDO, le SDO du maître doit-être relié au SDI de l'esclave et vice-versa. Pour éviter les confusions au moment du câblage, il est donc souvent recommandé d'utiliser les dénominations MISO-MOSI qui évitent une certaine ambiguïté.

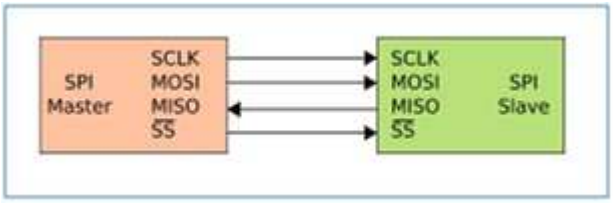


Figure 14: *Bus SPI 1 maitre 1 esclave*

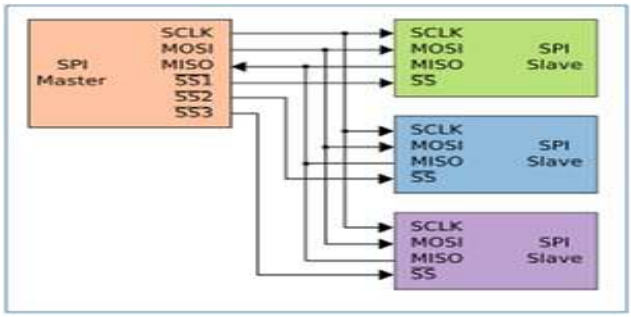


Figure 15: *BUS SPI 1 maitre N esclaves*

Etude Pratique

IX- Etude des différentes parties du système

1- Implantation du circuit horloge

Le système doit intégrer un circuit d'horloge qui se charge de la gestion de l'heure et de la date, pour cela on a adopté le circuit PCF8583, nous avons fait une implantation standard du circuit selon le schéma ci-dessous

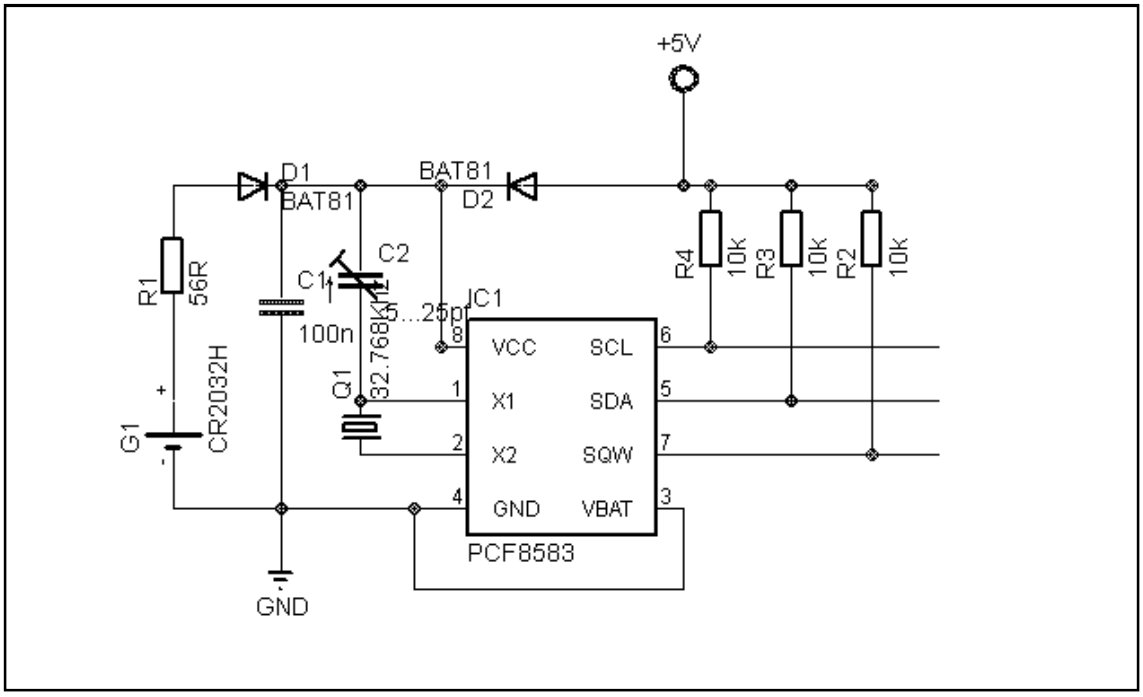


Figure 16:*Schéma d'implantation du circuit d'horloge***.**

Le circuit d'horloge est configuré pour générer un signal carré de fréquence 1HZ sur la broche 7 (SQW,INT) pour commander les interruptions du PIC.

Nous avons ajoutés une pile de sauvegarde et un système de commutation rapide (via diodes schottky) et ce pour permettre l'alimentation du circuit via cette pile en cas d'absence d'alimentation principale.

2- Implantation de l'afficheur LCD

Nous avons optés pour le mode 4bits, et nous avons utilisé un schéma classique pour l'implantation.

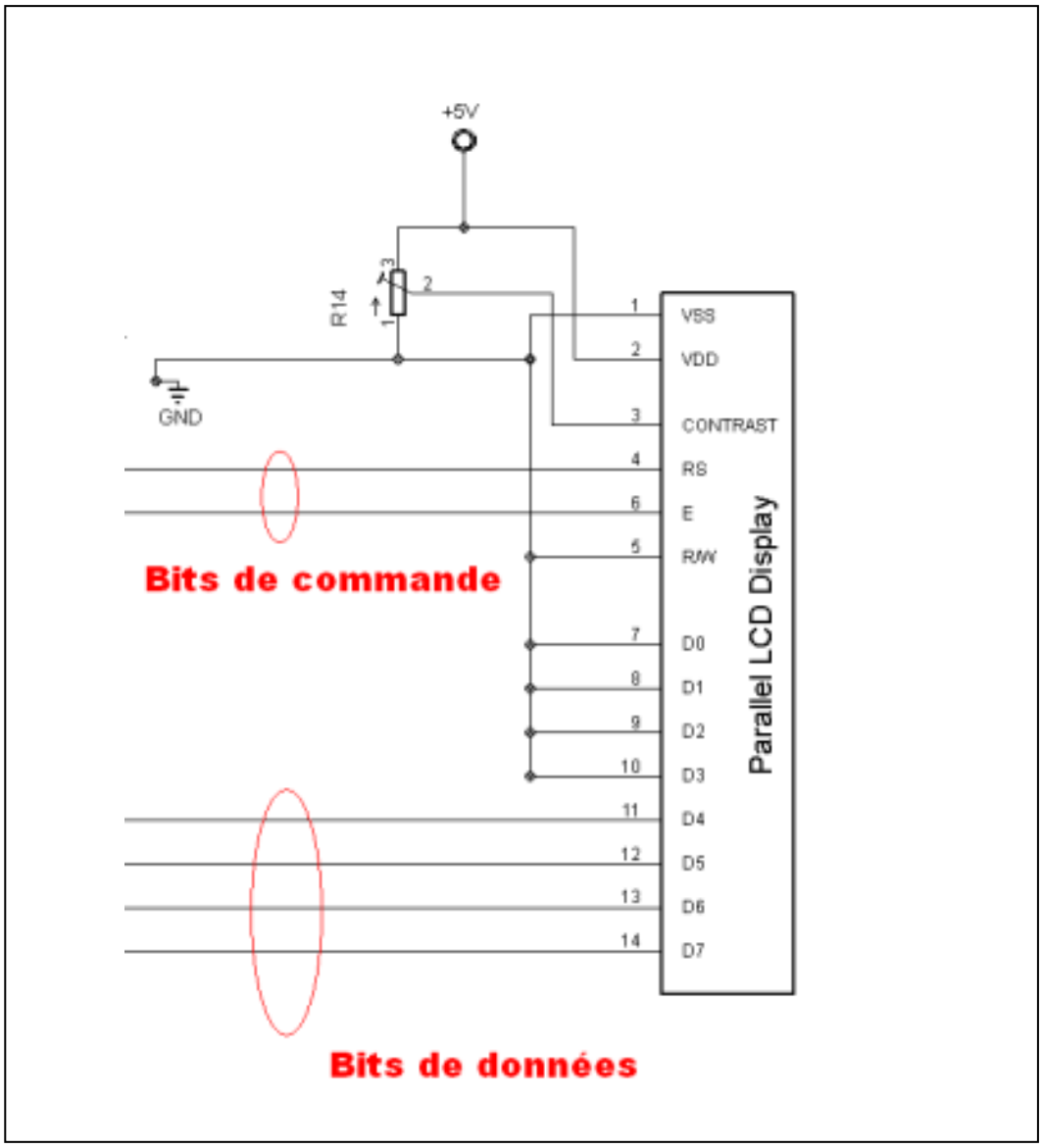


Figure 17: *schéma d'implantation de l'afficheur LCD*

3- Implantation des entrées analogiques et numériques

Nous avons utilisés 04 entrées analogiques et une entrée numérique. Les entrées analogiques sont protégées des surtensions par des diodes Zener. Bien que cette protection est basique mais elle suffit amplement pour notre système. Une protection plus perfectionnée peut être mise en place en assurant l'isolation galvanique du circuit.

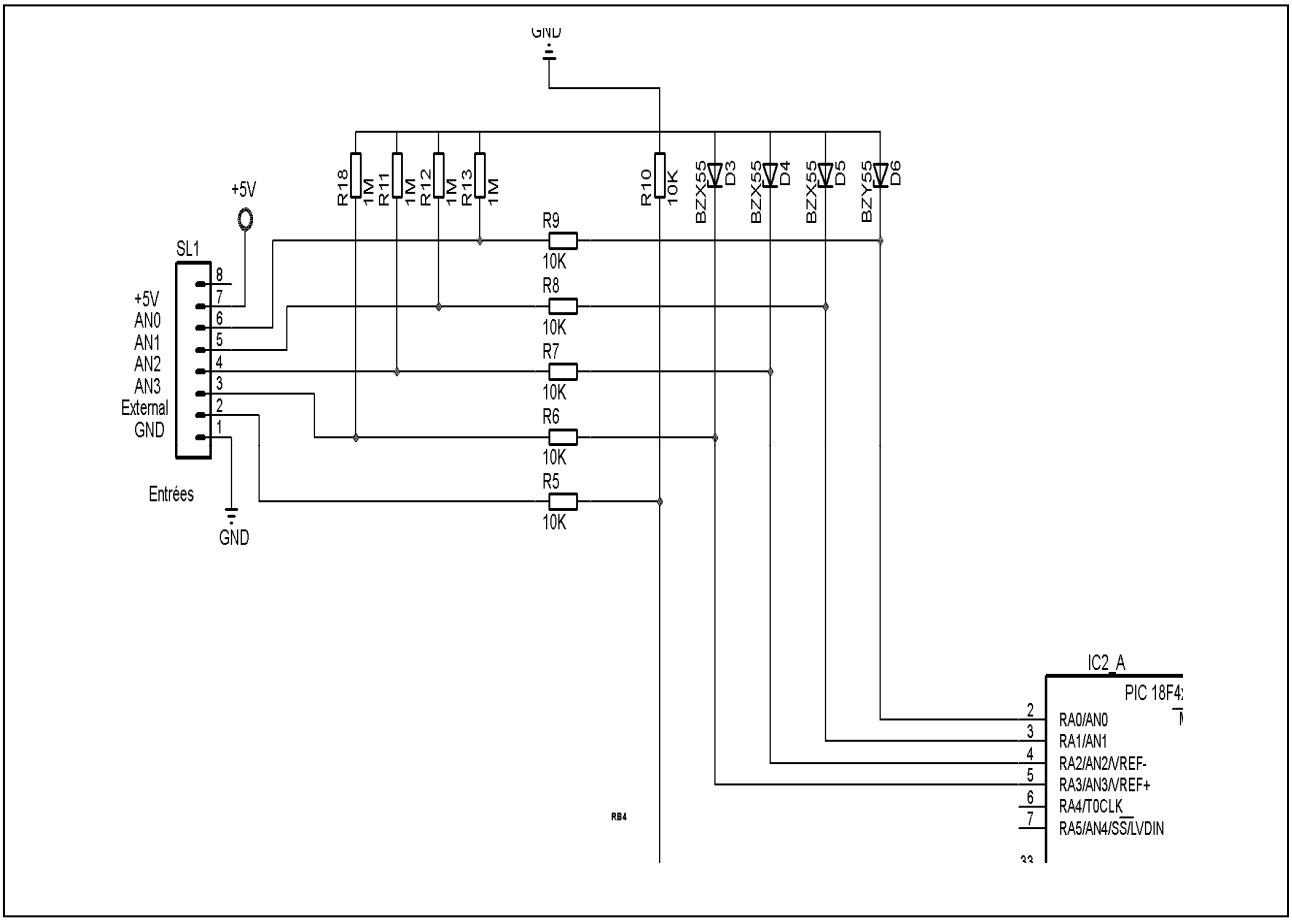


Figure 18:*Schéma d'implantation des entrées*

Les entrées analogiques sont reliées à la masse par des résistances qui font qu'en l'absence de données sur une entrées, elle sera ramenée à zéro.

4- Implantation de la carte mémoire

La carte mémoire est reliée au port C sur les broches RC2/RC3/RC4/RC5 qui sont configurées en mode SPI donc ils fonctionnement successivement : CCPI/SCL/SDA/SD0 .

Comme c'est indiqué dans la partie théorique nous avons respectés le brochage de la carte mémoire et on l'a reliée au PIC via le port indiqué ci-dessus.

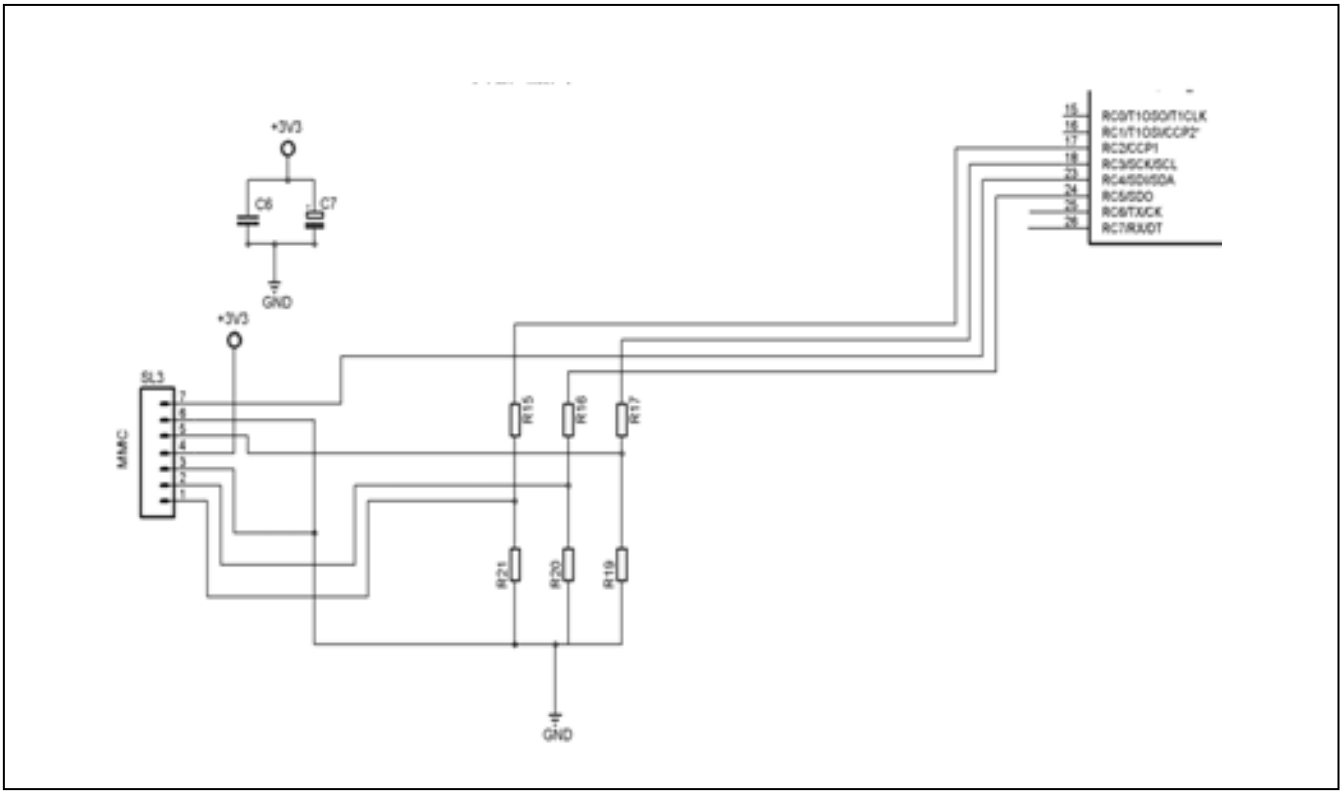


Figure 19: *schéma de la carte mémoire*

Notons que la carte mémoire nécessite une tension de 3.3V pour fonctionner.

5- Circuit d'alimentation

Tout montage électronique nécessite une alimentation pour fonctionner. Notre montage nécessite une alimentation 5V pour alimenter le PIC et ces périphériques, mais aussi une tension de 3.3V pour alimenter la carte mémoire.

Autre contrainte présente dans ce circuit est qu'il peut être, dans certains cas, alimenté par une batterie dans des zones qui ne sont pas couverts par le réseau électrique .

Pour ces raisons nous avons optés pour une alimentation externe, qui sera régulé et stabilisé vers les tensions nécessaires.

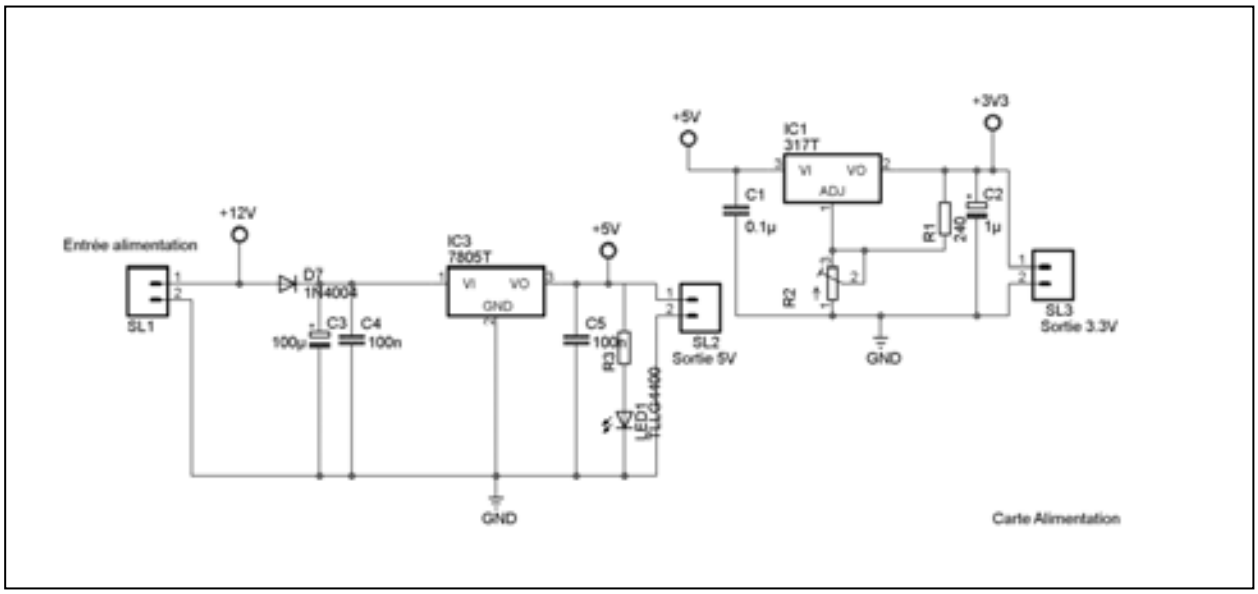


Figure 20:*Schéma de l'alimentation*

Le circuit d'alimentation reste un classique dans le genre, un régulateur de tension de type LM7805 nous fournit une tension positive de 5V avec un courant maximum de 1A.

Pour générer la tension de 3.3V nécessaire à la carte mémoire, nous avons optés pour un régulateur de type LM317, qui est un régulateur de tension positive qui peut varier entre 1.2V et 37V. Le régulateur est couplé à une résistance variable qui permet de choisir la tension qu'on désire.

Notons que nous avons pu opté pour un régulateur de tension fixe qui fournit 3.3V mais nous avons trouvé un problème d'approvisionnement de ce type de composants, donc nous avons choisit d'utiliser un composant courant facile à trouver.

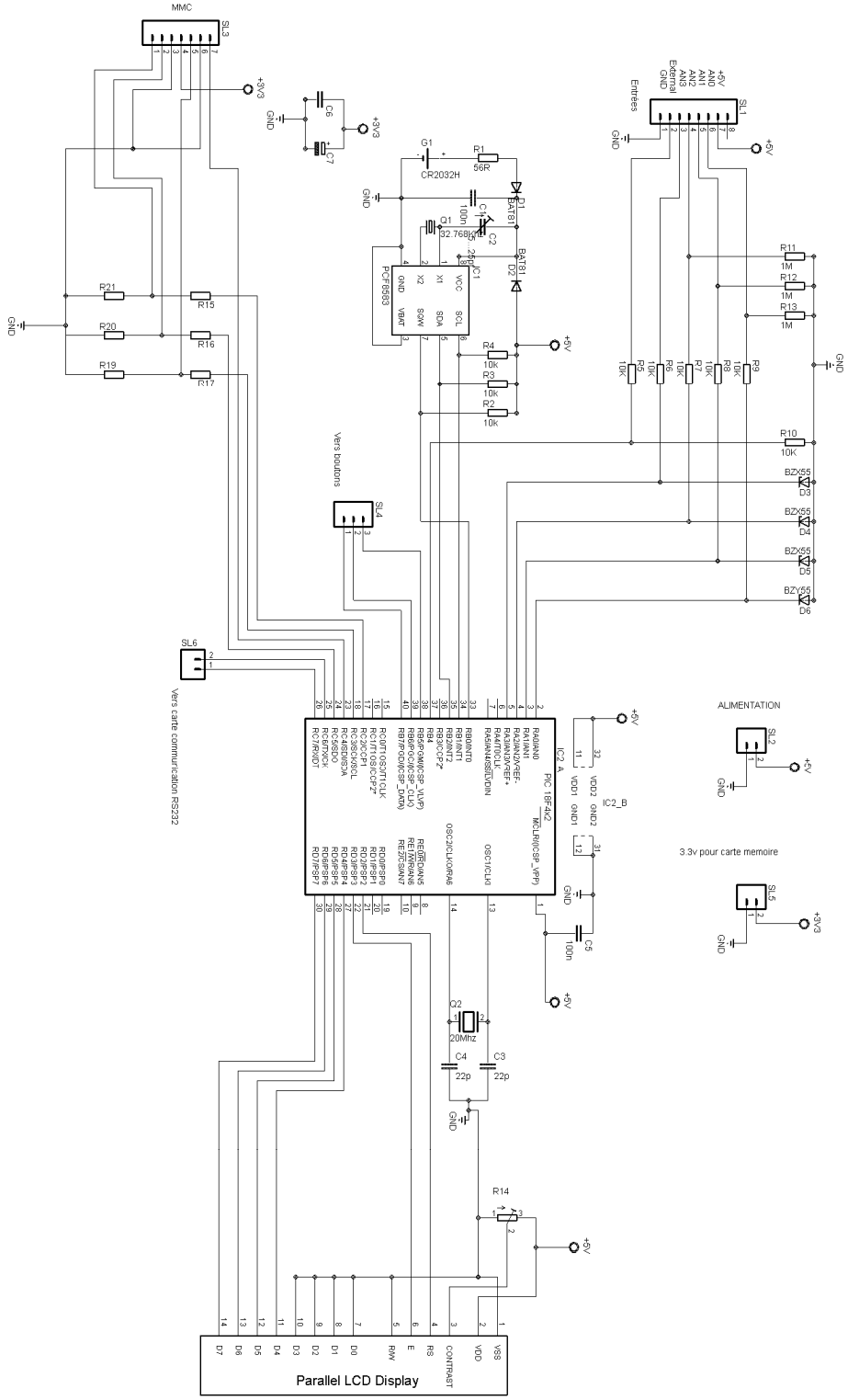


Figure 21: *Schéma de l'enregistreur de données*

X- Simulation

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 utilisé le logiciel ISIS qui est un très bon logiciel de simulation en électronique.

Isis est un éditeur de schémas qui intègre un simulateur analogique, logique ou mixte. Toutes les opérations se passent dans cet environnement, aussi bien la configuration des différentes sources que le placement des sondes et le tracé des courbes.

La simulation permet d'ajuster et de modifier le circuit comme si on manipulait un montage réel. Ceci permet d'accélérer le prototypage et de réduire son coût.

Il faut toujours prendre en considération que les résultats obtenus de la simulation sont un peu différents de celles du monde réel, et ce dépend de la précision des modèles SPICE[1] des composants et de la complication des montages.

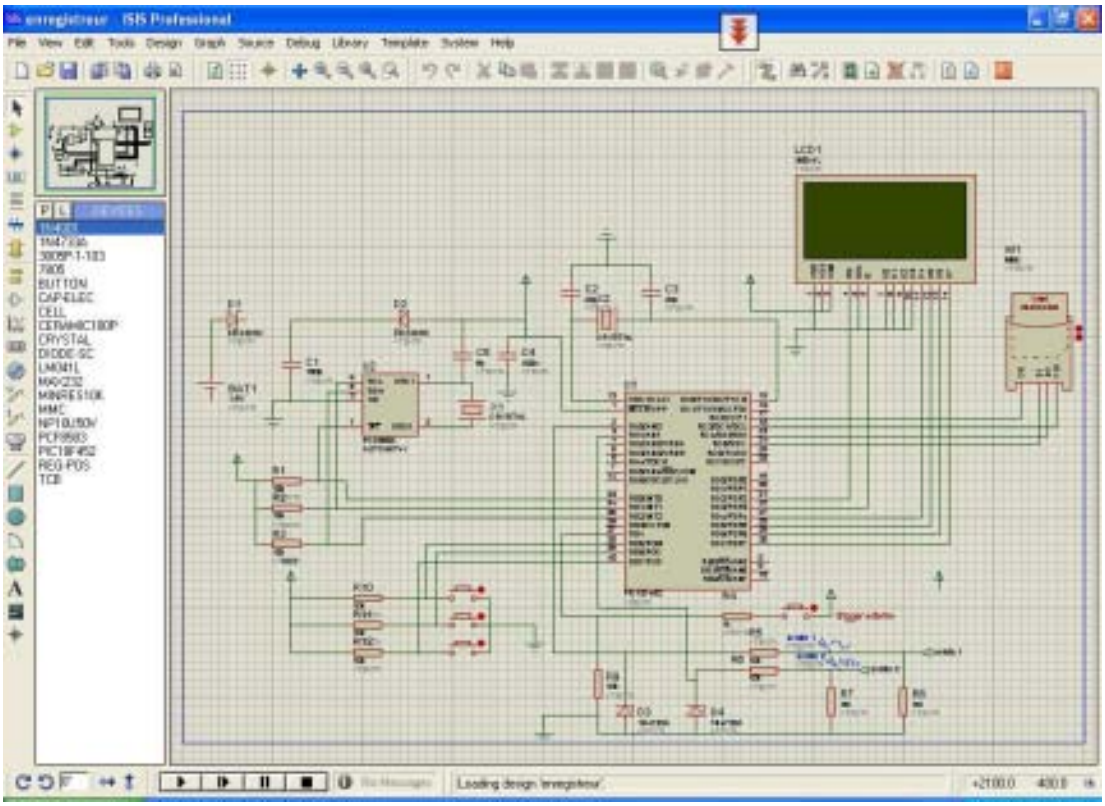


Figure 22:*Fenêtre du logiciel de simulation Proteus ISIS*

[1] **SPICE** (***S**imulation **P**rogram with **I**ntegrated **C**ircuit **E**mphasis*) est un logiciel de simulation généraliste de circuits électroniques analogiques. Il permet la simulation au niveau du composant (résistances, condensateurs, transistors) en utilisant différents types d'analyses

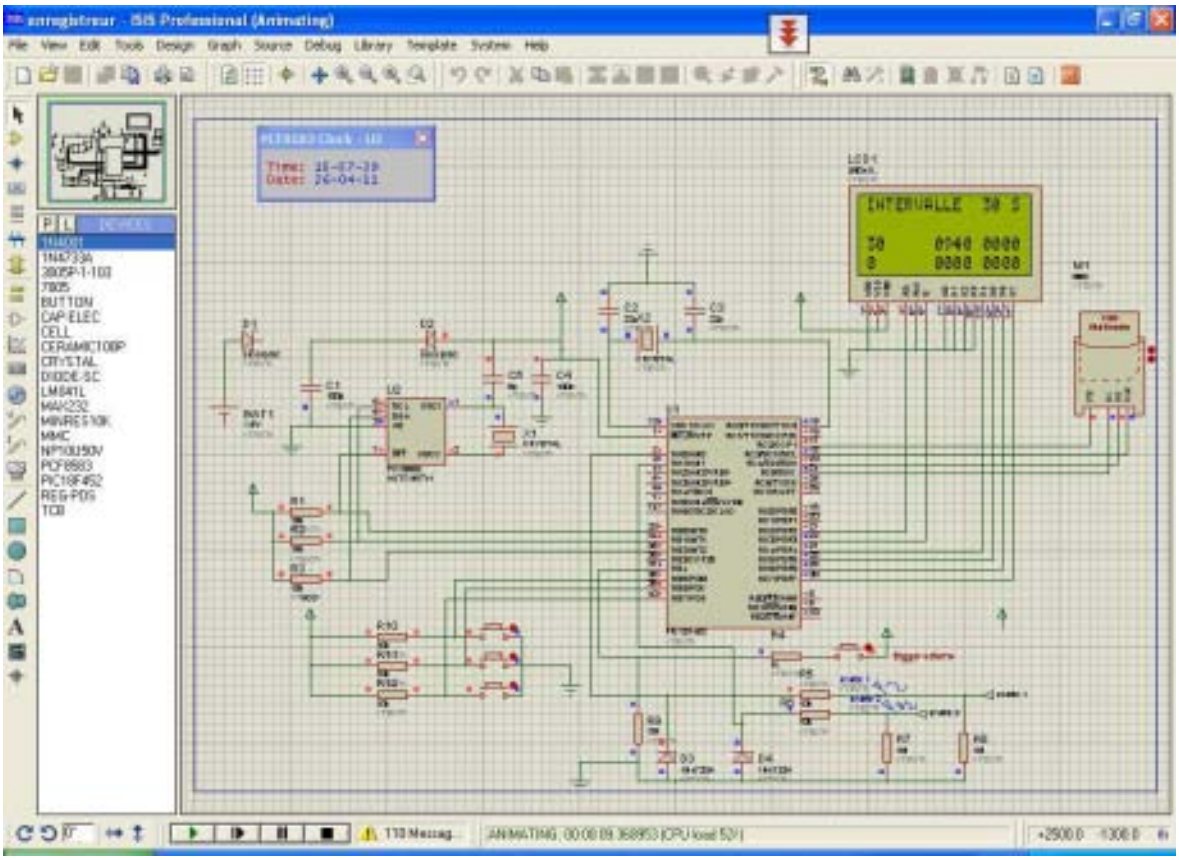


Figure 23: *Le logiciel ISIS en mode simulation du projet*

XI- Routage et création du circuit imprimé

Nous avons utilisé pour le routage du circuit imprimé le logiciel EAGLE de Cadsoft, qui permet de créer des cartes simple face et multi faces.

Nous avons crées deux cartes : carte alimentation et carte d'acquisition, les cartes sont de type simple face.

Notons que nous avons optés pour les cartes simple face à causes des contraintes techniques, car nous n'avons pas le matériel adéquat pour crée des cartes double face, qui aurait nous permit de créer un système plus compact.

Après identification des différents composants électronique de notre système, on peut choisir les différents empreintes de ces composants , car chaque composant peut avoir plusieurs types de boitiers.

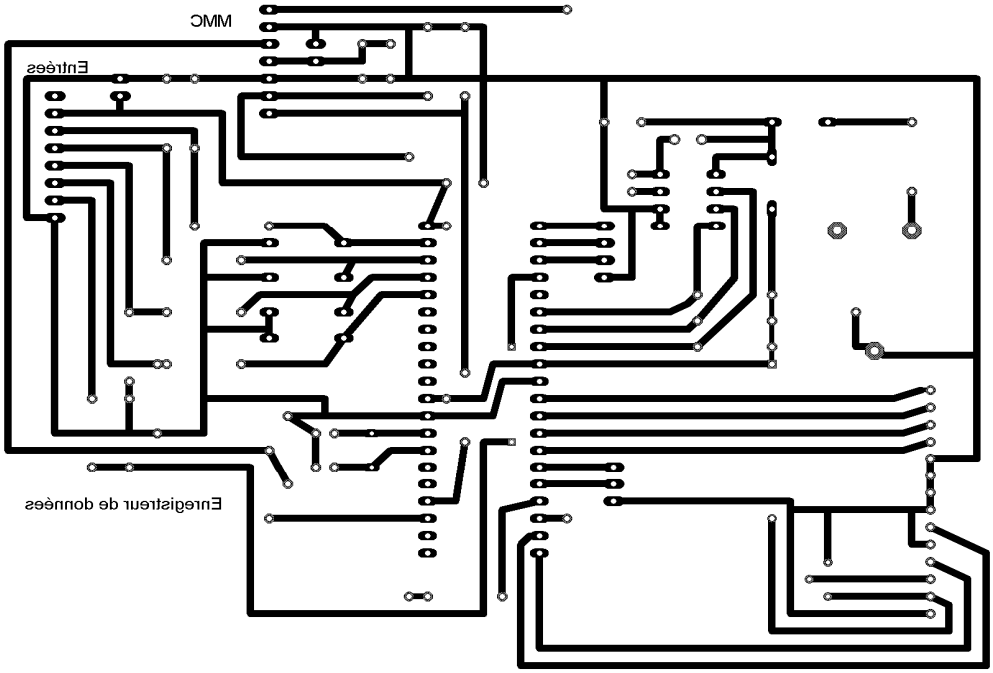


Figure 24: *carte principale coté cuivre*

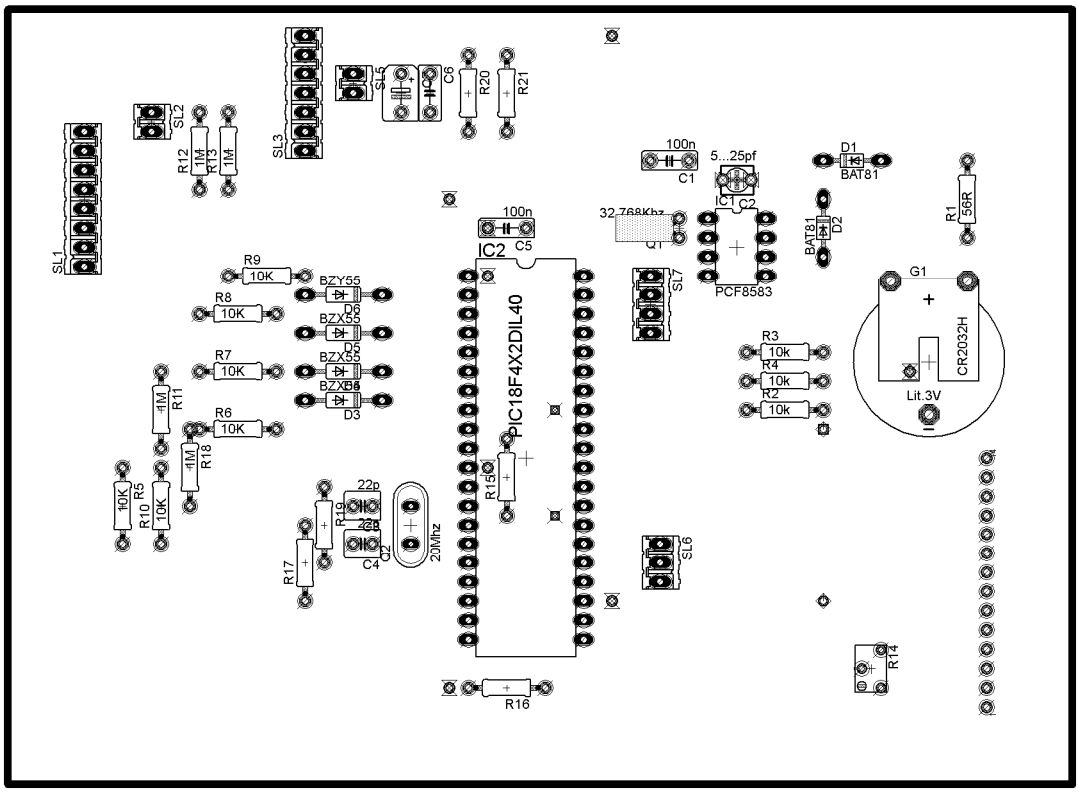


Figure 25:*carte principale coté composants*

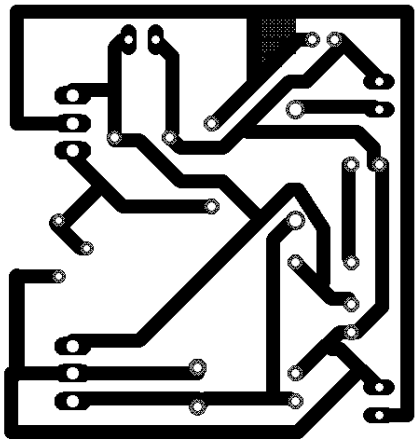


Figure 26: *carte alimentation coté cuivre*

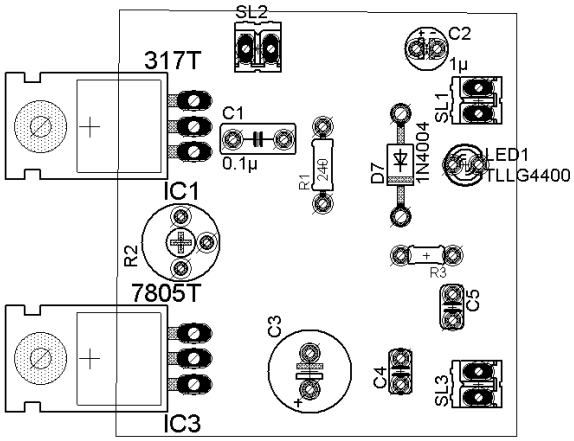


Figure 27: *carte alimentation coté composants*

XII- Partie Logicielle

1- Introduction :

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 naitre de nouveaux compilateurs qui permettent de programmer avec les langages haut niveau telsque 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 de l'outil que nous allons utiliser pour programmer notre PIC , nous avons optés pour le langage C. 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.

1.1- Avantages du C

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

1.2- Inconvénients du C

Le C n'est pas le langage naturel du microcontrôleur. Il permet de programmer plus intuitivement. Les logiciels de programmation en C transforment alors les lignes en C en lignes assembleurs directement compréhensibles par le microcontrôleur. Pour programmer efficacement, il est souvent nécessaire d'aller voir le code assembleur, il est donc conseillé d'avoir des bases solides en assembleur.

Nous avons fait le choix d'utiliser l'environnement de développement MikroC de Mikroelektronika.

2- Présentation du MikroC :

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

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

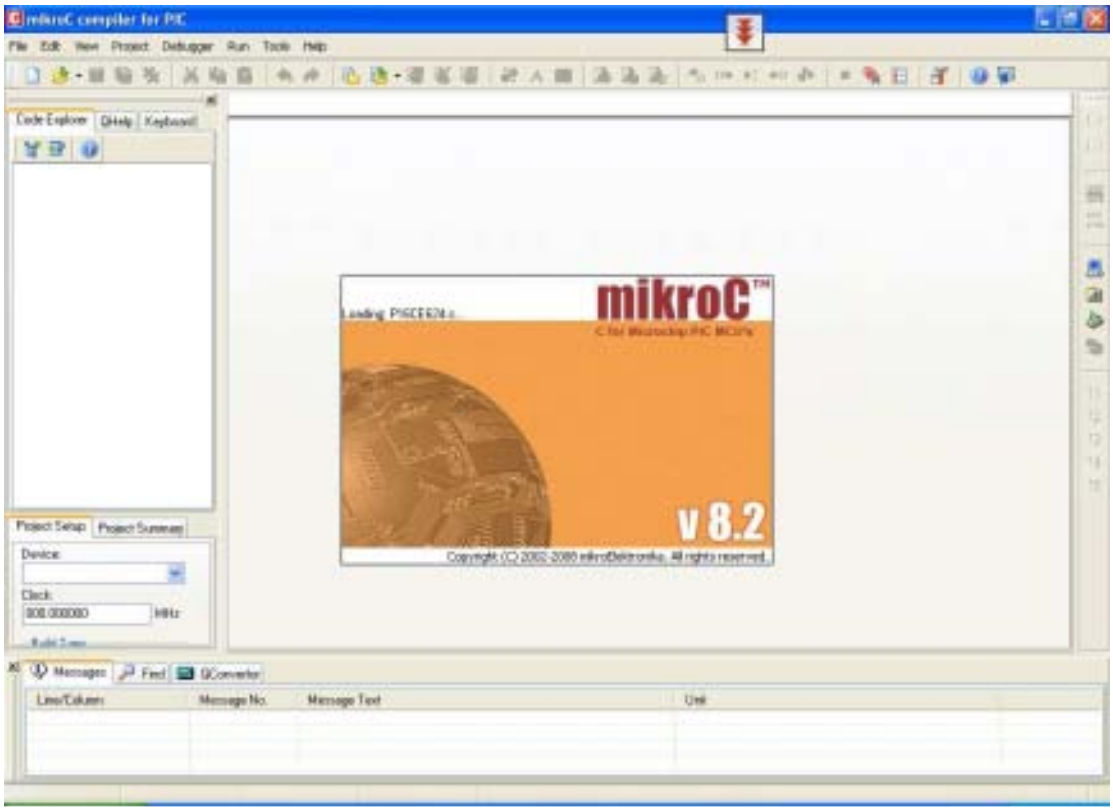
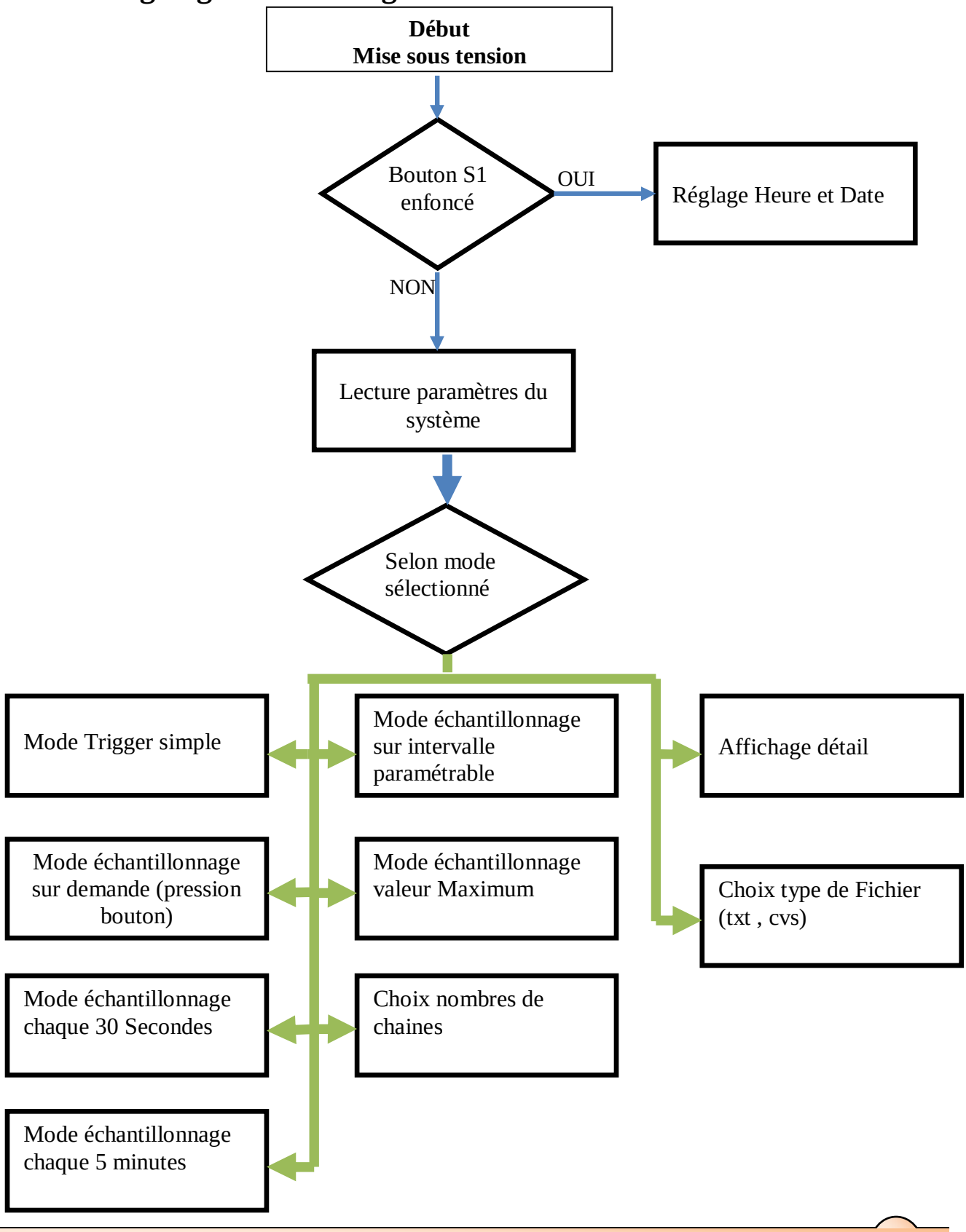


Figure 28: *Interface du logiciel MikroC*

Le compilateur MikroC nous permet de développer rapidement des applications complexes.

XIII- Logiciel embarquée dans le PIC

1- Organigramme du logiciel :



2- Initialisation :

Pour pouvoir utiliser les différents ports du PIC , MikroC exige qu'on lui indique comment est connecté chaque port .

Pour l'afficheur LCD qui est connecté sur le port D :

```
Lcd_Init(&PORTD); //port LCD
```

Pour l'horloge qui communique en protocole I2C et connecté sur le port B:

```
Soft_I2C_Config(&PORTB,2,1); //Port Soft i2c
```

Notons que pour la communication en mode I2C nous avons adoptés le mode soft I2C qui consiste à simuler par logiciel le fonctionnement du protocole I2C.

Bien que MikroC gère le soft I2C, il ne peut pas gérer ce mode en même temps que les interruptions, or nous avons besoins des interruptions pour exécuter quelques routines. La solution étant d'utiliser une bibliothèque externe qui gère Soft I2C et interruptions. Nous avons utilisés une bibliothèque crée par Michael Pearce de l'université de Canterbury (new zeland).

3- Gestion de l'horloge :

L'horloge étant installé sur le port B (B0,B1,B2) , nous allons utiliser le protocole I2C pour la gérer , deux procédures principales sont présentent RTC_Read (Real Time clock read:lire l'horloge) et RTC_Setup (Real Time clock setup:écrire dans l'horloge)

Lecture de l'horloge:

```
//Lecture de la date et de l'heure
// Read time & date
void RTC_Read(void)
{
     // Lire données a partir RTC /Read data from RTC.
     i2cstart();
     i2csendbyte(0xa0); // Addresse  RTC.
     i2cgetack();
     i2csendbyte(0x02);
     i2cgetack();
     I2C_Receive(0xa0,&Time.sec_bcd,5);
 //lecture de l'année
     if (((Time.day_bcd>>6)&0x03) != (Year&0x03))
```

```
    {
  //ecriture de l'année dans eeprom
          Year++;
          Eeprom_Write(EEPROM_ANNEE,Year);
          Delay_ms(30);
    }
    Time.day_bcd &= 0x3f; // mask of year bits.
    Time.month_bcd &= 0x1f; // mask of weekdays
    Time.year_bcd = Dec2Bcd(Year);
    // We need time in binary format.
    Time.sec = bcd_2_bin(Time.sec_bcd);
    Time.min = bcd_2_bin(Time.min_bcd);
    Time.hour = bcd_2_bin(Time.hour_bcd);
    Time.day = bcd_2_bin(Time.day_bcd);
    Time.month = bcd_2_bin(Time.month_bcd);
    Time.year = Year + 2000; // s i la machine va être
commercialisé en l'an 2100 alors il ya un bug
}
```

4- Ecriture dans l'horloge:

L'écriture sur l'horloge se fait à travers la fonction RTC_setup qui permet de régler l'heure et la date et l'enregistrer sur le circuit.

La principale fonction qui envoie les données sur l'horloge est *I2C_Send* qui envoie une valeur BCD (Binary Coded Decimal) .

```
bcd[0] = 0; // adresse depart /start address
bcd[1] = 0x80; // stopper /stop
bcd[2] = 0; // cents
I2C_Send(0xa0,bcd,9);
bcd[1] = 0; // demarrage /start
I2C_Send(0xa0,bcd,2); // Update ctrl_stat register
```

2. **Le PCF8583 a pour adresse 0Xa0 sur le bus I2C .**
3. **Le PCF8583 renvoi et reçoit toujours des valeurs BCD.**

5- Gestion des fichiers :

Notre enregistreur de données écrit sur la carte mémoire, nous devons au début initialiser le système de fichiers pour cela on utilise la procédure suivante:

```
//Initialisation du fichier sur MMC
//Init file on mmc
void Fichier_Init(void)
{
     Lcd_Out(1,1,"INIT Memoire     ");
     Spi_Init_Advanced(MASTER_OSC_DIV64,DATA_SAMPLE_MIDDLE,CLK_IDL
E_LOW,LOW_2_HIGH);
     if (!Mmc_Fat_Init(&PORTC,2))
     {
    //Quand la carte MMC est détectée on reinitialise le bus SPI
à une fréquence superieure
    // Once a card is detected the SPI bus may be reinitialised
at a higher speed.

     Spi_Init_Advanced(MASTER_OSC_DIV16,DATA_SAMPLE_MIDDLE,CLK_IDL
E_LOW,LOW_2_HIGH);
          Lcd_Out_Cp("  OK");
     }
     else
     {
          Lcd_Out_Cp("Erreur           ");
          Lcd_Out(2,1,"SYSTEME ARRETE  ");
          while(1);
     }
}
```

1. **Il faut toujours initialiser la carte mémoire à une vitesse minimale, ensuite on peut l'augmenter.**
2. **Il faut toujours tester si l'opération s'est bien déroulée ou non.**
3. **Le système de fichiers doit être de type FAT16**

Pour créer un fichier on utilise la fonction suivante:

```
//Creation nouveau fichier
//Create a new file
void Fichier_Nouveau(void)
{
     int nr;

     Lcd_Cmd(LCD_CLEAR);
     Lcd_Out(1,1,"Nouveau fichier.");

     if (Csv==0) strcpy(Filename,"    0000.TXT");
     else strcpy(Filename,"    0000.CSV");
     nr = 1;
     while (nr<10000)
     {
          Int2Str(nr,Filename+2,2); // Now Filename = "
xxxx\0TXT"
          Filename[8]='.'; // replace zero by dot -> Filename = "
xxxx.TXT"
          if (!Mmc_Fat_Assign(Filename,1)) // Try to open file for
reading.
          {
               // File does not exist so we can create it.
               Mmc_Fat_Assign(Filename,0xA0);
               Mmc_Fat_Rewrite();

     Mmc_Fat_Set_File_Date(Time.year,Time.month,Time.day,Time.hour
,Time.min,Time.sec);
               Mmc_Fat_Write(CRLF,2);
               break;
          }
          nr += 1;
     }

     if (nr<10000)
     {
          // Un nouveau fichier est crée /New file was created.
          Fichier_voir_nom(Filename);
     }
     else
     {
          // On a une erreur de creation de fichier /New file was
NOT created.
          Fichier_voir_nom("ERREUR!");
     }

     Delay_ms(500);
}
```

Cette procédure permet de créer des noms de fichiers séquentiellement.

1- **On ouvre un fichier avec la commande *Mmc_Fat_Assign* .**
2- **On crée un nouveau fichier avec les commandes *Mmc_Fat_assign* ,*Mmc_Fat_Rewrite* et *Mmc_Fat_write***

Pour écrire des données dans un fichier on a eu recours à la découpe des fonctionnalités en sous ensembles pour mieux gérer le code et simplifier l'utilisation:

La procédure *Fichier_Ecrire_val* , permet d'écrire des entiers dans le fichier .

La procédure *Fichier_Ecrire_temp_date* , permet d'écrire l'heure et la date dans le fichier.

La procédure *Fichier_Ecrire_Donnees* , permet d'écrire les données récoltés des entrées analogiques du PIC.

1. **la fonction *Mmc_Fat_Write* permet d'écrire des données dans un fichier.**
2. **Pour lire les entrées Analogiques du convertisseur ADC on utilise la fonction *Adc_read***

6- Les interruptions

Une interruption est une action, déclenchée par un événement précis, qui va arrêter le programme principal, s'exécuter, puis reprendre le programme principal à l'endroit où il s'était arrêté. Les interruptions peuvent être déclenchées par:

- Les timers
- Les pins interruptibles (ex : RB0)
- Les interfaces de communication
- Le convertisseur analogique-numérique

Pour utiliser les interruptions, il faut tout d'abord les activer. Il faut pour cela fixer les valeurs des registres suivant :

** INTCON qui contient :

- GIE (bit 7) : activation générale des interruptions
- PEIE (bit 6) : activation des interruptions des périphériques

- bit5-bit3 (RBIE,TMR0IE..): activation individuelle de interruptions principales
- bit2-bit0 (RBIF,TMR0IF..): Lorsqu'un événement déclencheur se produit, il est notifié dans un "flag" (drapeau) qui permettra par la suite d'identifier la cause de l'interruption
- INTCON2,INTCON3 contiennent des options supplémentaires
- PIR1,PIR2,PIR3 contiennent les flags des différentes sources d'interruptions des périphériques
- PIE1,PIE2,PIE3 contiennent l'activation des différentes sources d'interruptions des périphériques

L'exécution des interruptions se fait à partir de l'adresse programme 0x08. En effet, lorsqu'une interruption se produit, le programme principal s'arrête et exécute l'instruction à l'adresse 0x08. La routine d'interruption est structurée de la façon suivante:

- Sauvegarder les registres
- Identifier la source de l'interruption
- Effectuer l'action associée à cette interruption
- Restaurer les registres
- Revenir au programme principal

Pour notre cas nous avons crées trois fonctions de gestion d'interruptions à savoir : *Interrupt* , *Interrupt_Enable* et *Interrupt_Disable*.
La fonction *Interrupt* est un mot réservé de MikroC qui permet d'initialiser les interruptions.

La fonction *Interrupt_Enable* permet d'activer les interruptions. Dans cette fonction on utilise les commandes suivantes :

```
INTCON.PEIE = 1;// Enable peripheral interrupts.
INTCON.GIE = 1;// Globally enable interrupts.
INTCON.INT0IF = 0;// Clear interrupt flag.
INTCON.INT0IE = 1;// Enable INT0 interrupts.
```

L'horloge étant configurée sur le port INT0 et qui envoie un signal carré de fréquence 1Hz donc active les interruptions chaque seconde.

La fonction Interrupt_Disable permet de désactiver les interruptions et ce via la commande :

```
INTCON.GIE = 0; // Globally disable interrupts.
```

1. **Pour activer les interruptions on utilise la commande :** `INTCON.GIE=1`
2. **Pour désactiver les interruptions on utilise la commande:** `INTCON.GIE=0`
3. **Pour utiliser les interruptions on doit <u>toujours</u> utiliser la fonction réservée** `Interrupt`**.**

Nous avons utilisés l'EEPROM du PIC pour sauvegarder les paramètres globaux du système. La mémoire EEPROM est une mémoire non volatile, donc on y va sauvegarder les données. En MikroC la fonction *Eeprom_Write* permet d'écrire dans l'EEPROM.

Pour lire une variable de l'EEPROM nous utilisons la fonction *Eeprom_Read*.

1. **L'EEPROM est une mémoire non volatile.**
2. **Pour ecrire sur l'EEPROM on utilise la fonction *Eeprom_Write***
3. **Pour lire de l'EEPROM on utilise la fonction *Eeprom_Read***

7- Les entrées analogiques :

Nous avons utilisés le convertisseur analogique numérique (CAN) intégré au PIC, ce CAN se trouve sur le port A, on doit l'initialiser en entrées analogiques et pour cela on utilise la fonction suivante :

```
ADCON1 = 0; //le registre ADCON1 configuré en analogique
TRISA = 0xff; //le port A configuré en entrée
```

Ensuite pour lire les données on utilise la fonction *Adc_read*, dont ci-dessous un exemple d'utilisation :

```
Samples[0] = Adc_read(0); // lecture Port A0
Samples[1] = Adc_read(1); // lecture Port A1
Samples[2] = Adc_read(2); // lecture Port A2
Samples[3] = Adc_read(3); // lecture Port A3
```

XIV-Description du fonctionnement de l'enregistreur de données

L'enregistreur de données se présente avec 3 boutons de commande et un afficheur, l'interaction avec l'utilisateur se fait à travers cette interface, certes minimaliste mais amplement suffisante.

Les boutons sont nommées "bouton reglage", "bouton +", "bouton –", chacun de ces boutons à une fonction principale et une fonction secondaire.

Au début, pour avancer dans le menu du système il faut utiliser le bouton réglage qui va naviguer dans les options du menu, pour changer les valeurs présents dans le menu utiliser les deux boutons + et - .

Le système permet d'afficher sur l'afficheur LCD le mode de fonctionnement actuel et les valeurs en acquisition.

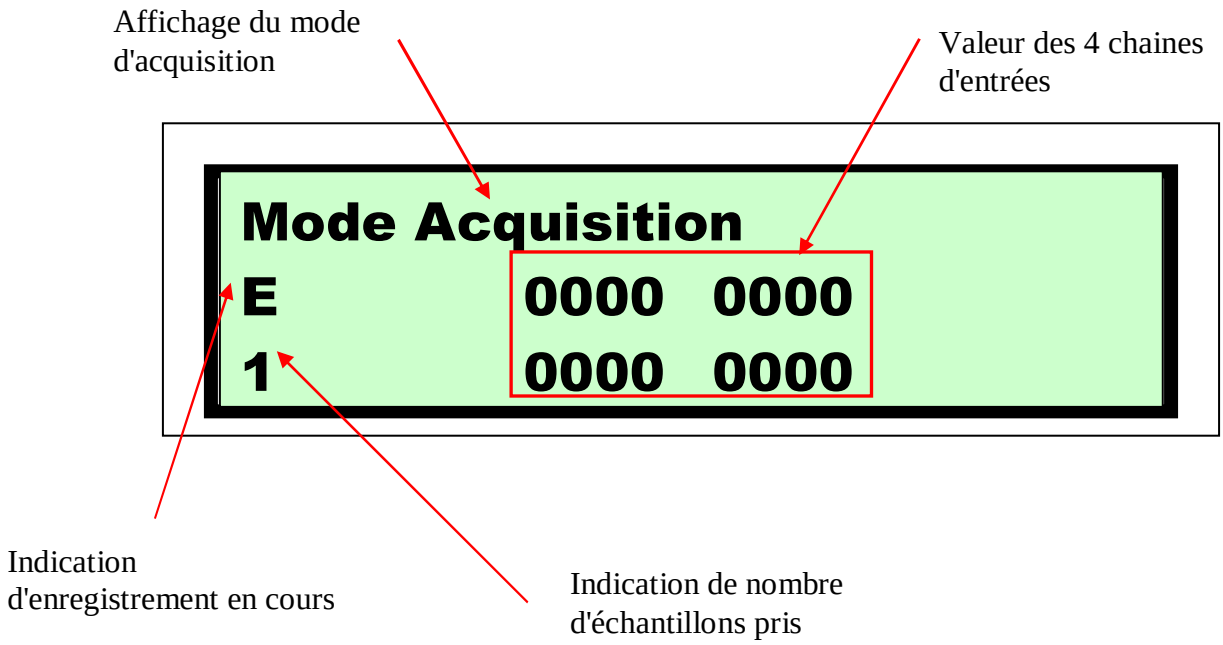


Figure 29:*Module LCD*

Un mode spécial du système permet de visualiser sans enregistrer les données:

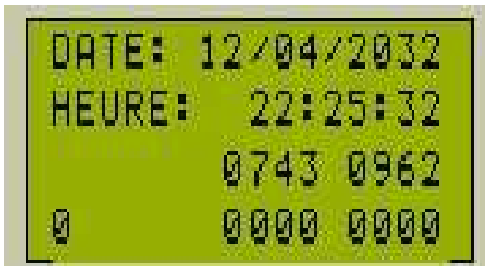


Figure 30: *Mode affichage*

Le mode par défaut du système est le mode Trigger externe qui permet d'enregistrer les données sur une demande externe

Figure 31: *Mode Trigger externe*

L'écran suivant illustre le cas de figure où on a réalisé 4 prises de données :

Nombre d'échantillons pris

Figure 32: *Mode Trigger externe avec 4 données*

Le mode suivant est le mode où on prend les échantillons par demande de l'utilisateur en appuyant sur le bouton +

Figure 33: *Mode échantillonnage sur demande*

La prise automatique de données dans des intervalles que l'utilisateur définit est possible en mode INTERVALLE, l'utilisateur peut changer l'intervalle en appuyant sur les boutons + et - :

Figure 34: *Mode enregistrement sur intervalle*

L'utilisateur peut choisir le nombre de chaines échantillonnés, il peut choisir une valeur entre 1 et 4 chaines.

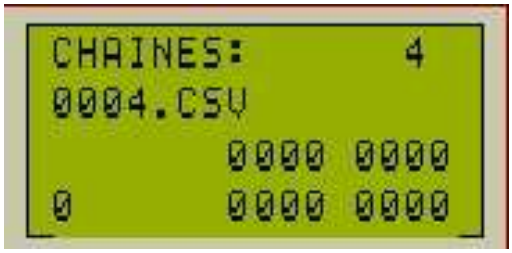


Figure 35: *Choix de nombre de chaines*

XV- Logiciel de visualisation :

Pour exploiter efficacement les données enregistrés on a crée un petit logiciel qui lit les fichiers de données et les transforme en courbes exploitable.

Le logiciel crée exploite une seule forme d'échantillonnage à savoir : prise de données sur intervalle régulier. Bien sûr un logiciel plus développé selon les besoins spécifiques du domaine d'application de l'enregistreur, doit être mis en place ; nous avons voulu démontrer une application concrète de l'enregistreur.

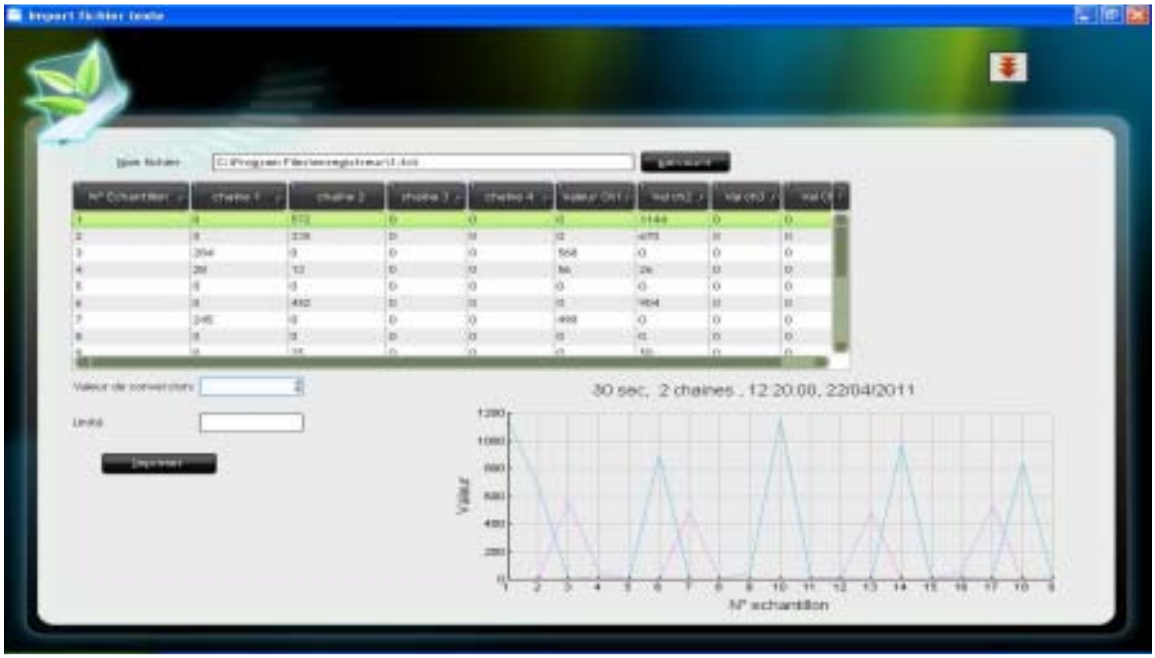


Figure 36 : *Copie écran du logiciel d'exploitation des données*

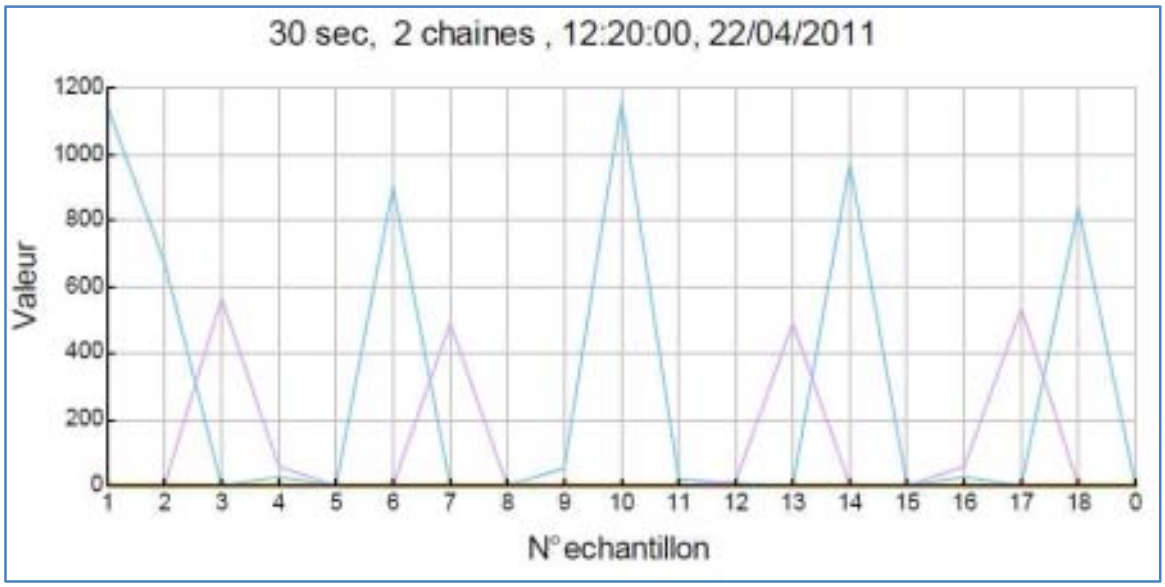


Figure 37 : *Exemple d'une courbe*

XVI- L'enregistreur de données en pratique

Figure 38 : *Carte principale et carte alimentation*

Figure 39 : *Boitier*

CONCLUSION

La réalisation de ce projet nous a énormément appris, autant au niveau de l'électronique, de la programmation des microcontrôleurs (programmation embarquée). Nous avons aussi appris des nouvelles connaissances au niveau de la gestion du temps et des équipes.

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

- ✓ Créer un logiciel sous PC pour lire les données et les transférer vers des formats standards.
- ✓ Améliorer la protection des entrées analogiques.
- ✓ Permettre à l'enregistreur de communiquer en temps réel avec un PC via interface série (RS232) ou USB.
- ✓ Créer des cartes d'entrées spécifiques à chaque application.

Les améliorations qu'on peut apporter à notre travail sont énormes et peuvent varier selon le type d'application qu'on souhaite.

BIBLIOGRAPHIE

[1] SD card projects using the PIC microcontroller –Dogan Ibrahim-Newnes 2010

[2] Advanced PIC microcontroller projects in C-Dogan Ibrahim-newnes

[3] PIC Microcontroller project book –JOHN IOVINE- McGrawhill

[4] PIC MICRO Pocket reference-Myke predko-McGrawhill

[5] http://www.platea.123.fr/datasheet/cs.pdf

[6] http://fr.wikipedia.org/wiki/SPICE_%28logiciel%29

[7] http://www.allpinouts.org/index.php/Secure_Digital_(SD)_Card

Annexes

Liste de composants

Carte principale :

C1,C5 100n
C2 5...25pf
C3 ,C4 22p
D1 schottkly BAT81
D3,D4,D5,D6: Zener 5.5V BZX55
G1 pile CR2032H
IC1 PCF8583
IC2 PIC18F452DIL40
Q1 crystal 32.768Khz
Q2 crystal 20Mhz
R1 56ohms
R2 ,R3,R4,R5,R6,R7,R8,R9 ,R10,R22,R23,R24 : 10k
R11,R12,R13,R18 1M
R15,R16,R17 2.2k
R19 ,R20,R21 3.3k
U1 afficheur LCD 4*16

Carte alimentation

C1 0.1µ
C2 1µ
C3
C4
C5
D7 1N4004
IC1 LM317T
IC3 LM7805T
LED1 Diode led
R1 240 ohm
R2
R3