REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

Université A/MIRA Bejaïa

Faculté de Technologie

Département : Génie Electrique

Mémoire de fin d'études

En vue de l'obtention du diplôme Master en Electrotechnique

Option : Commande des systèmes électriques

Thème

Conception et simulation d'un multi-afficheur pour une ligne de production

Présenté par : AFTIS Fatah
AKKOUCHE Merzouk

Encadré par : GHEDAMSI Kaci
CHALABI Yanis

Promotion : 2014-2015

Remeciements

Tout d’abord, Nous tiens à remercier DIEU le miséricordieux de nos avoir donné la possibilité de réaliser notre projet, d’arriver à notre souhaits et d’atteindre notre objectifs.

Nous aimerons dans ces quelques lignes remercier toutes les personnes qui d’une manière ou d’une autre, ont contribué au bon déroulement de notre travail, tout au niveau humain qu’au niveau scientifique.

Nous tenons tout d’abord à remercier notre encadreur **GHEDAMSI Kaci**, on a pu bénéficier à la fois de ses compétences scientifiques, et de sa grande disponibilité, tant pour résoudre les difficultés rencontrées lors de notre réalisation, de répandre à nos questions.

Nos remerciements vont également à tout le personnel de l’entreprise SPA Générale Emballage D’Akbou pour leur hospitalité surtout à Mr CHALABI Yanis.

Et à toute personne ayant contribué de près ou de loin à notre soutien moral.

DEDICACE

Au nom du tout puissant

Je dédie ce modeste travail à :

Mes chers parents. Ce travail est le fruit de leur soutien depuis toujours, leur présence à mes cotes et leur affection.
Mes sœurs.
A toute ma famille
A toute la promo de l'électrotechnique 2014/2015
A mes copains de chambre
A mes amis
A mon binôme Merzouk

fatah

DEDICACE

Au nom du tout puissant

Je dédie ce modeste travail :

A mes très chers parents et ma grand-mère. Ce travail est le fruit de leurs soutiens depuis toujours, leurs présences à mes cotes et leurs affections.
A mes frères : jugurta,tarik,yuba,fayçal,koussayla et islem.
A ma sœur tinhinane .
A Liza.
A mes copains de chambre : karim,toufik et malak
A mes amis : aiteur,billal,tahar ,zohir, ...
A mon binôme fatah

MERZOUK

Sommaire

Sommaire

Chapitre II : Les différents outils pour la conception de l'afficheur

Chapitre III Simulation des données de l'afficheur

Chapitre IV : Conception Virtuel

Liste de Figures

Liste de figures

Liste des tableaux

Liste de tableaux

Introduction générale

Introduction générale

Notre projet de fin d'étude consiste à étudier et à concevoir un afficheur des données, qui permet d'afficher des mesures de façon automatique au profit de la SARL Général Emballage, conformément au cahier des charges établi au préalable.

La SARL Général Emballage est un leader confirmé au niveau national de l'industrie du carton ondulé, fondée en Aout 2000 à la zone industrielle TAHARACHT Akbou, véritable carrefour Économique de Bejaia.

Depuis 1969, l'informatique industriel (au sens du traitement automatique de l'information), grâce à sa flexibilité et à la variété des applications, a profondément modifié l'organisation des sociétés industrielles. Traditionnellement réservée au calcul scientifique et à la gestion.

L'automatisation des systèmes de production a été une des réponses à l'évolution du contexte industriel et à la compétitivité. Cette automatisation visait un double objectif : l'augmentation de la productivité du système technique (réduction des coûts, fiabilité, disponibilité, qualité) et l'amélioration de la sécurité directe des opérateurs, dans la mesure où la majorité d'entre eux est éloignée et protégée du lieu de transformation du produit.

Les architectures des système automatisée ont aussi très fortement évolué. L'intégration dans les produits et systèmes des nouvelles technologies de l'information et de la communication se concrétise par l'apparition des nouvelles générations d'équipements. De plus enplus de traitements sont inclus au niveau des capteurs et des actionneurs et la décentralisation des entrées/sorties et des périphériques de dialogue homme/machine .

Le développement de la technologie et la complexité des schémas électrique a permis de remplacer les blocs de fonctionnement par des microcontrôleurs, ils sont une révolution dans les mondes des circuits intégrés car ils sont adapté à accomplir de nombreuse fonction leur utilisation ne limite plus pour des commandes d'entrée/sortie, mais aussi des applications beaucoup plus complexes.

Pour réalisation de notre système, sur une ligne de production, nous allons opter pour le plan de travail suivant :

- la présentation des généralités sur les systèmes automatisés, des généralités sur les microcontrôleurs et les outils de programmation.
- La présentation les différents outils pour la conception de l'afficheur.
- la programmation et la compilation dans PIC C Compiler, la simulation du programme avec logiciel d'ISIS PROTEUS.

- La conception virtuelle, c'est un domaine en plein progrès. Elle offre de nouveaux outils pour de nombreuses applications dans des divers domaines.

Présentation de l’entreprise

Présentation de l'entreprise

Général Emballage est une entreprise algérienne spécialisée dans la fabrication et la transformation de carton ondulé.

Actuellement, elle entre en production sur trois sites industriels (akbou, oran et setif).

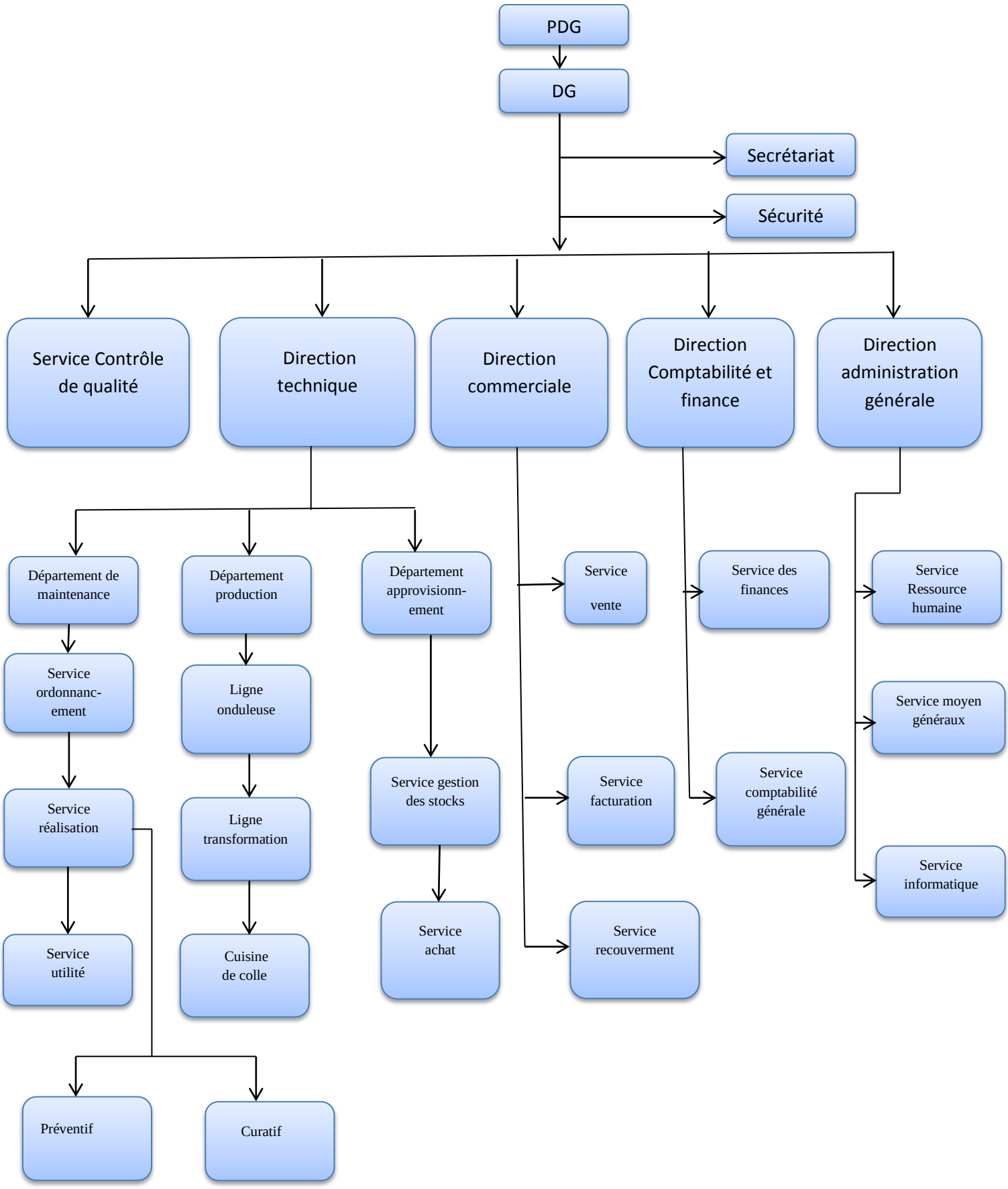
Elle est entrée en exploitation en 2002 et a obtenu le trophée de la production (Euro-devloppemment PME) en 2007. En raison de son développement .Générale emballage débute son exportation vers Tunis en 2008.

Générale Emballage réalise depuis quelques années déjà une croissance à deux chiffres. Les indicateurs de performance de l'entreprise n'ont jamais été aussi mondiaux de certification en Algérie .Dans le cadre de partenariat, l'entreprise signe un accord professionnel avec l'université de Bejaia en juillet 2012.

Générale emballage a su au cours des années mettre son expérience au service du client grâce a une recherche permanente de l'efficience qui s'est traduite en 2013 par la classification ISO 9001 : 2008 de son système de management de la qualité.

L'organigramme ci-après représente la réparation de l'entreprise et ses différents services.

Organigramme de l'entreprise



Chapitre I

Généralités sur l'Automatisation et microcontrôleur

I.1 Introduction

Aujourd'hui, dans un monde où la concurrence règne, automatisé son unité de production n'est plus un choix, mais une nécessite pour toute entreprise voulant rester en activité. L'automatisation améliore la qualité du produit, facilite la tâche de l'opérateur et du maintenancier.

L'évolution dans le domaine de l'électronique a pu élaborer des contrôleurs intelligents appelé microcontrôleurs, ces derniers caractérisés par leur taille et leur architecteur interne qui leur confie souplesse et vitesse, parmi ces microcontrôleurs on distingue les PICs présenté par l'entreprise américaine MICROCHIP.

I.2 Les systèmes automatisés

Un système est dit automatisé s'il exécute toujours le même cycle de travail après avoir reçu les consignes d'un opérateur. L'automatisation d'un procédé (un équipement industriel) consiste à assurer la conduite par un dispositif technologique. Fonction globale d'un système automatisé est de conférer une valeur ajoutée, à un ensemble de matières d'œuvre dans un contexte donné. De plus, un système de production est dit industriel si l'obtention de cette valeur ajoutée pour un ensemble de matières œuvre donne a un caractère reproductible et peut être exprimée et quantifiée en termes économiques.

I.3 Objectif de l'automatisation

Les objectifs d'un système automatisé sont :

- Accroitre la productivité du système et améliore la qualité de produit.
- Augmenter la sécurité du personnel.
- Contrôler et protèger les installations et les machines.
- Améliorer la flexibilité de production.
- Eliminer les taches complexes, dangereuses, pénibles ou indésirables en les faisant exécuter par la machine **[1]**.

I 4. Les différentes approches de l'automatisme

I.4.1. Les systèmes logiques

On désire étudier le comportement global du système automatisé, et ainsi, décrire l'ordre dans lequel le système effectue les différentes tâches.

Alors, toutes les chaînes d'actions et d'acquisition (sorties et entrées) sont représentées par des variables de type logique (signal binaire 0 ou 1).

Chaque grandeur ne connaît que deux états différents, on ne tient pas compte des régimes transitoires : allumé/éteint, ouvrir/fermer, présent/absent, à l'arrêt/en mouvement…

I.4.2 Les systèmes continus

On désire étudier le comportement temporel d'une seule chaîne d'action, on prend en compte les régimes transitoires. Les grandeurs d'entrées et de sortie ne sont plus binaires (elles sont analogiques).

I.5 Structure d'un système automatisé

Tout système automatisé est composé de trois parties principales :

I.5.1 La partie opérative

C'est la partie visible du système, elle effectue les opérations en exécutant les ordres qui lui sont donnés par la partie commande. Elle comporte les éléments suivant :

- Des prés actionneurs, lesquels reçoivent des ordres de la partie commande.
- Des actionneurs qui ont pour rôle d'exécuter ces ordres, ils transforment l'énergie pneumatique, hydraulique ou électrique en énergie mécanique.
- D'une détection (capteurs) qui informe la partie commande de l'exécution du travail.

I.5.2 La partie commande

C'est cette partie qui élabore les ordres nécessaires à l'exécution du processus**,** en fonction des informations qu'elle reçoit de la partie opérative, elle les restitue vers cette partie en direction des prés actionneurs.

La partie commande se présente sous forme de :

- Carte électronique.
- Automates programmables API.
- Calculateur (microordinateur).

I.5.3 La partie dialogue

Sa complexité et sa taille dépend de l'importance du système. Cette partie regroupe les différentes commandes nécessaires au bon fonctionnement de procédé : marche-arrêt, arrêt d'urgence, marche automatique **[1]**.

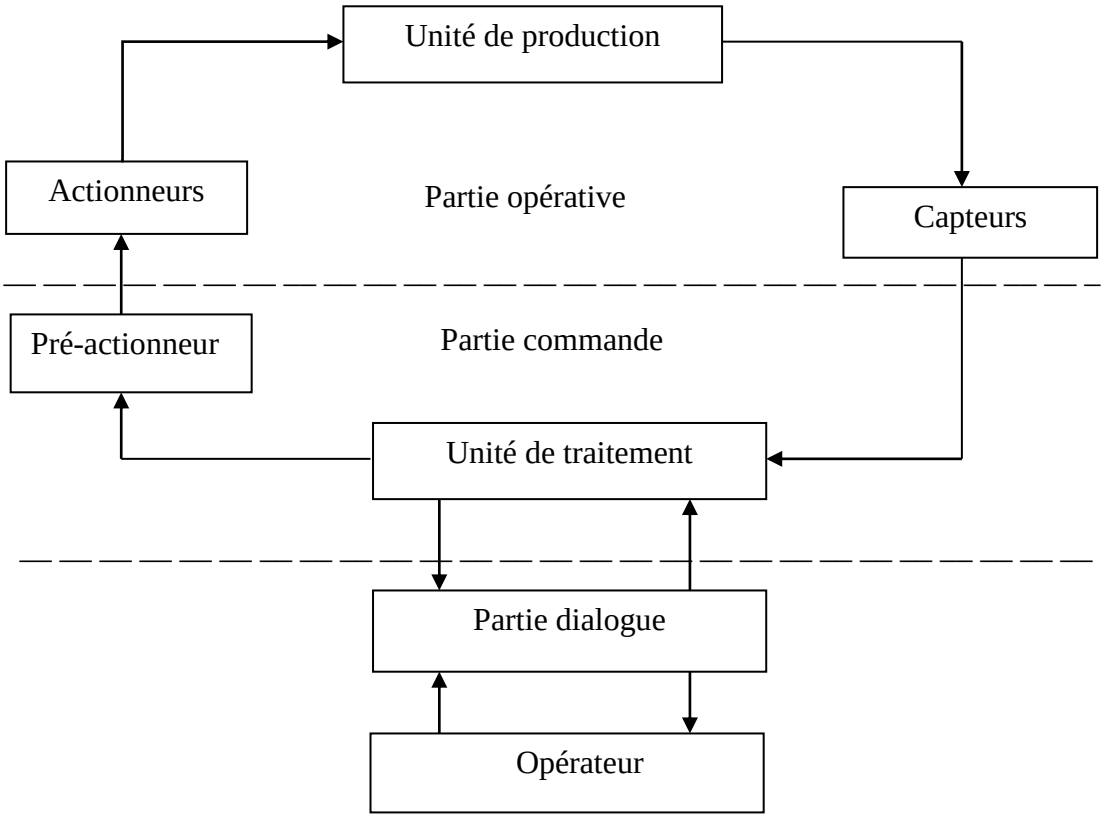


Figure I.1 Structure détaillée d'un automatisme [2].

I.6 Les choix technologiques

Lorsque l'automaticien a décrit le fonctionnement de l'automatisme en utilisant le GRAFCET ou L'ORGANIGRAMME, il a le choix pour réaliser la partie commande de l'automatisme, entre divers solutions technologiques qui sont récapitulées dans le tableau ci-dessous. En fait, trois grandes familles technologiques sont à la disposition des automaticiens : les constituants électromagnétiques, électroniques et pneumatiques. A l'intérieur de chacune des familles on doit prendre en considération la nature du traitement réalisé : logique câblée seule pour l'électromagnétique et la pneumatique, logique programmée pour l'électronique. Une troisième caractéristique concerne le type d'utilisation lié à la nature du système mis en œuvre : universel pour les systèmes unitaires ou semblable, spécifique pour les systèmes répétitifs **[2]**

Tableau I.1 les principales solutions technologiques en automatisme

<table>
<tr><td colspan="4">Les principales solutions technologiques en automatisme</td></tr>
<tr><td colspan="2">Famille technologique de base</td><td>Universel</td><td>Spécifique</td></tr>
<tr><td colspan="2" rowspan="2">Electromagnétique</td><td>Relais</td><td rowspan="2"></td></tr>
<tr><td>Contacteurs auxiliaires</td></tr>
<tr><td rowspan="5">Electronique</td><td rowspan="2">Câblée</td><td>Blocs ou cartes</td><td></td></tr>
<tr><td></td><td>Cartes spécifiques</td></tr>
<tr><td rowspan="3">programmée</td><td>Mini ou
Micro-ordinateur</td><td rowspan="3"></td></tr>
<tr><td>Automates programmables</td></tr>
<tr><td>Microsystème universels</td></tr>
<tr><td colspan="2"></td><td>Microsystèmes spécifiques</td></tr>
<tr><td colspan="2">Pneumatique</td><td>Logique à clapets</td><td></td></tr>
</table>

I.7 Le microcontrôleur

C'est un circuit intégré qui rassemble les éléments essentiels d'un ordinateur : processeur, mémoires (mémoire morte pour le programme, mémoire vive pour les données), unités 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 (de quelques mégahertz jusqu'à plus d'un gigahertz) 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'électro-ménager, les jouets, la téléphonie mobile, etc.

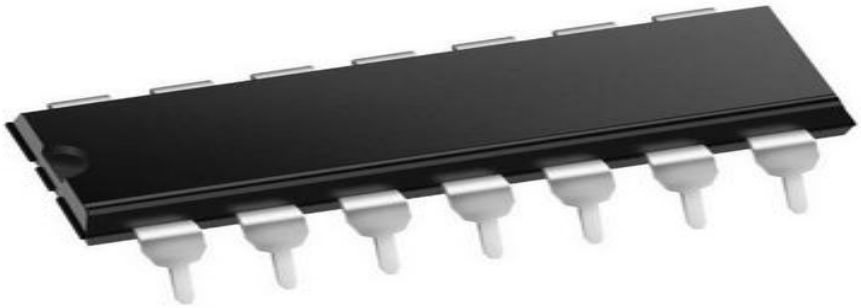


Figure I.2 : Le microcontrôleur

I.7.1 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.

- 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.
- Réalisation des applications non réalisables avec d'autres composants.

I.7.2 Les défauts des microcontrôleurs

- le microcontrôleur est souvent surdimensionné devant les besoins de l'application.
- Investissement dans les outils de développement.
- Écrire les programmes, les tester et tester leurs mise en place sur le matériel qui entoure le microcontrôleur.
- Incompatibilité possible des outils de développement pour des microcontrôleurs de même marque.

- Les microcontrôleurs les plus intégrés et les moins coûteux sont ceux disposant de :
    - ROM programmables par masque.
    - Fabrication uniquement en grande série >1000.
    - Défaut relatif car il existe maintenant des version OTPROM un peu plus chère.

I.7.3 Utilisation d'un microcontrôleur

Toutes les solutions à base de composants programmables ont pour but de réduire le nombre de composants sur le circuit électronique et donc fiabiliser le circuit.

Le microcontrôleur est en concurrence avec d'autres technologies suivants les applications :

Il y a 3 types de technologies

- Logique câblée :

très rapide, fonctions réalisées par une voie matérielle, non programmable, peu économique quand l'application est complexe, peu de souplesse (durée d'étude prohibitif et circuit difficilement modifiable).

- Réseaux de logique programmables (PAL, LCA,..) :

rapide, adapté au traitement de signaux complexes, prix élevé et langage de programmation non standard

- Les µprocesseurs :

grande souplesse : Les fonctions sont réalisées par voie logicielle
puissance de calcul, langage évolué, nombre important de composant à réunir, solution onéreuse à retenir :

- si la fonction à réaliser est simple applique une logique câblée
- si le nombre d'unités à réaliser est très important applique circuits intégrés dédié en logique câblée pour les fonctions simples

Une réalisation logicielle est toujours plus lente qu'une réalisation en logique câblée , le microprocesseur exécute une seule instruction à la fois.

(Les µcontroleurs ils ont des avantage des µprocesseurs mais limités aux applications ne nécessitant pas trop de puissance de calcul, nombre de composant très réduit, mais souvent surdimensionnement devant les besoins de l'application)

I.7.4 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 ont affiné un peu la définition de ce qu'il fallait intégrer pour arriver à un schéma type analogue à la figure I.3 :

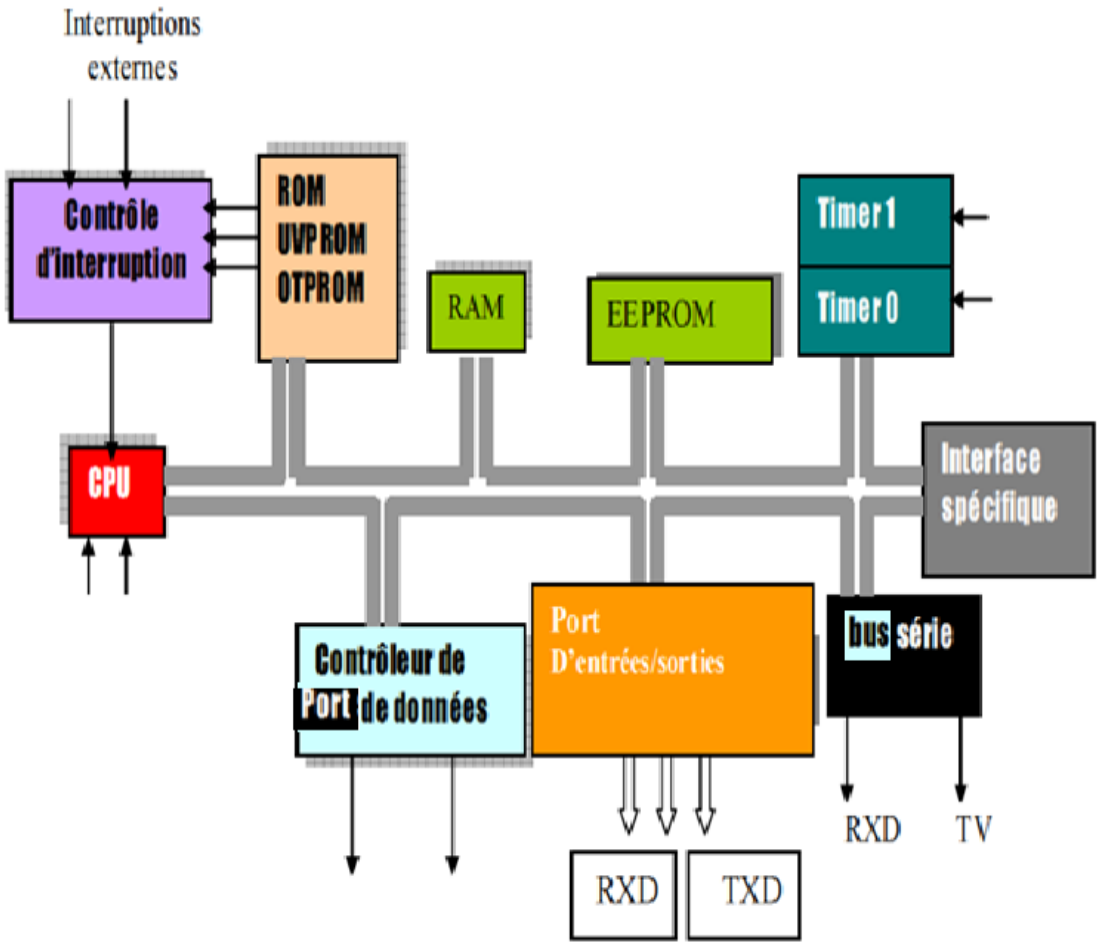


Figure I.3 : Structure interne d'un microcontrôleur

I.8 Le microcontrôleur PIC

I.8.1 Définition d'un PIC

Un PIC est un microcontrôleur, c'est à dire une unité de traitement de l'information de type microprocesseur, ses caractéristiques principales sont :

- Séparation des mémoires de programme et de données (architecture Harvard), on obtient une meilleure bande passante et des instructions et des données pas forcément codées sur le même nombre de bits.
- Communication avec l'extérieur seulement par des ports, il ne possède pas de bus d'adresses, de données et de contrôle comme la plupart des microprocesseurs.
- Utilisation d'un jeu d'instructions réduit, le nom de son architecture RISC (Reduce Instructions Set Construction). Les instructions sont ainsi codées sur un nombre réduit de bits, ce qui accélère l'exécution (1 cycle machine par instruction sauf pour les sauts qui requirent 2 cycles). En revanche, leur nombre limité oblige à se restreindre à des instructions basiques, contrairement aux systèmes d'architecture CISC (Complex Instructions Set Construction) qui proposent plus d'instructions donc codées sur plus de bits mais réalisant des traitements plus complexes**.[5]**

I.8.2 Les différentes familles des PICs

MICROCHIP propose une très grande variété de PIC élargissant ainsi l'étendu des applications possibles, et offrant la possibilité d'adapter le PIC à l'application à réaliser. Les PICs sont réparti en trois familles principales, chaque famille possède une architecture de base bien déterminée, et utilise un mot d'instruction qui est différent pour chaque famille **[4]**.
Les trois familles sont :

- **les pics base-line** : qui utilisent des mots d'instructions de 12bits.
- **les pics mid-range** : qui utilisent des mots d'instructions de 14bits.
- **les pics high-end** : qui utilisent des mots d'instructions de 16 bits et plus.

I.8.3 Identification d'un PIC

les PICs sont identifiés par une étiquette de ce genre : xx(L)XXyy-zz
Xx : indique la famille.
L : indique qu'il fonctionne avec une plage de tension beaucoup plus tolérante.
XX : c'est le type de mémoire de programme.
exp : C : pour indiquer qu'il s'agit d'une mémoire EPROM ou EEPROM.
CR : pour dire qu'elle est de type ROM ou (O.T.p).
F : pour préciser le genre FLASH.
Yy : identifie le PIC lui-même.
Zz : la vitesse max du quartz **[4]**.

I.8.4 Classification des microcontrôleurs [2]

On peut classer les microcontrôleurs selon deux critères importants :

- Selon l'organisation des mémoires.
- Selon l'unité centrale de traitement.

A. Selon l'organisation des mémoires

On distingue deux architectures souvent utilisées en conception des microcontrôleurs :

- **Architecture de Von Neumann**

Cette architecture est très utilisée dans la majorité des microprocesseurs et des microcontrôleurs, elle est commune à celles des ordinateurs, où la mémoire contient à la fois les instructions à exécuter et les données à manipuler, et elle ne dispose que d'un seul bus de données qui relie la mémoire à l'unité centrale.

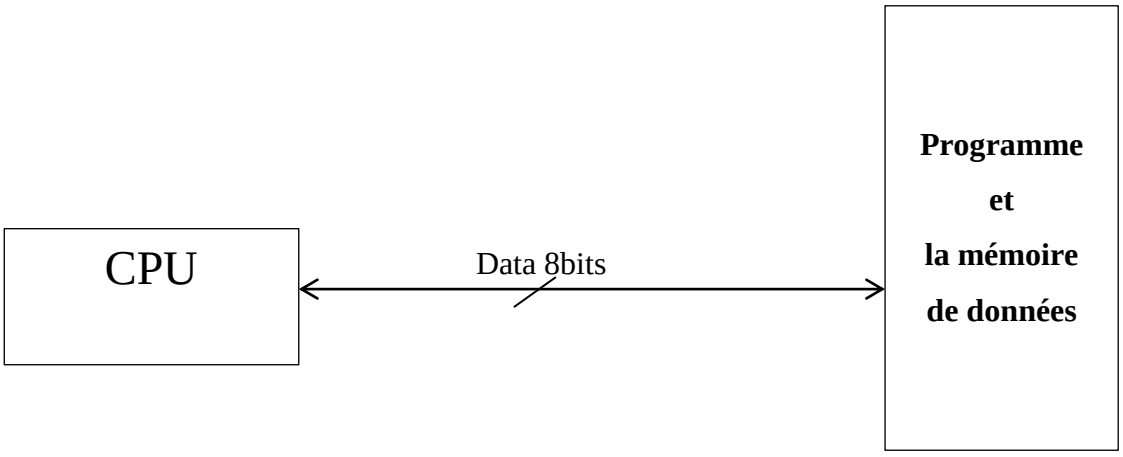


Figure I.4 Structure de Von Neumann

- **Architecture Harvard**

Contrairement à l'architecture Von Neumann, ce type d'architecture contient des mémoires, l'une propre au programme et l'autre aux données, elles sont séparées et câblées à l'unité centrale via des bus différents ce qui offre une possibilité d'avoir simultanément la donnée et l'instruction.

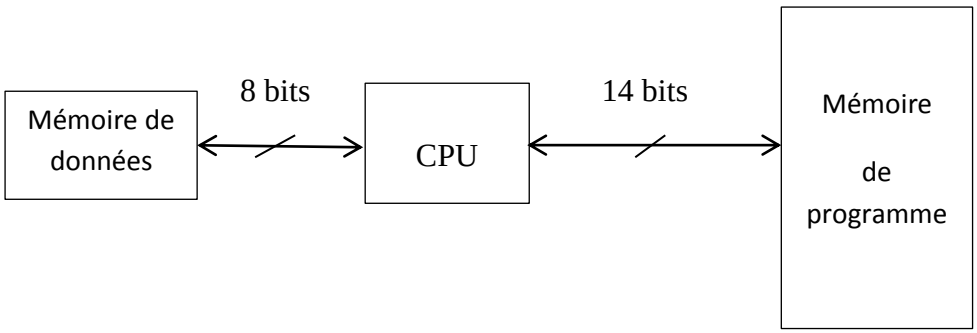


Figure I.5 L'architecture de Harvard

B. Selon l'unité centrale de traitement

Au niveau de l'unité centrale de traitement, on trouve deux types :

- **RISC** : qui veut dire microprocesseur à jeu réduit (reduced instruction-set computer).

Les PICs à unité RISC utilisent des instructions sur 12 à 14 bits ou sur mot ce qui se traduit par deux avantages : tous les emplacements de la mémoire contiennent une instruction, et un seul cycle machine suffit pour lire le code de l'instruction complet. En plus ils utilisent une structure de type Pipe-line.

- **CISC :** qui veut dire microprocesseur à jeu d'instruction complexe (complexe instruction-set computer). Comme son nom l'indique cette unité utilise un jeu d'instruction compliqué et encombré, difficile à maîtriser mais permet d'autre part de réaliser des traitements très précieux **[3]**.

I.8.5 Familles de PIC disponibles sur le marché [4]

Les principales familles de PIC sont :

10Fxxx/12Fxxx : sont des composants récents. Ils ont comme particularités d'être extrêmement petits, simples et économiques.

16Cxxx/16Fxxx : est un composant haute de gamme, il constitue la famille la plus fournie et la plus utilisée.

17Cxxx : est une gamme intermédiaire entre 16Cxxx et 18Fxxx. Cette gamme n'est enrichie par MICROCHIP. Elle supporte la compilation en C.

18Cxxx/18Fxxx : est une famille qui a un jeu d'instruction plus complet puisqu'il comprend l'ordre de 75 instructions. Cette gamme d'instruction étendue lui permet de faire fonctionner du code en langage C compilé de manière nettement plus efficace que les familles précédentes. On peut les utiliser avec un quartz oscillant jusqu'à 48MHz.

24Fxxx : sortie en 2004, c'est la famille plus récente. Elle est programmable en langage C comme tous les autres PICs.

PIC32 : sortie en novembre 2007, les PICs 32 sont des microcontrôleurs 32 bits.

dsPIC30/dsPIC33 : le dsPIC (digital signal PIC) est le premier microcontrôleur qui a une architecture de 16 bits (les autres étant à 8bits) de la société microchop.il es adapté aux applications de traitement de signal **[5]**.

I.8.6 Caractéristique générale de quelque famille des PICs

PIC	FLASH	RAM	VITESSE	EEPROM
12F675	//	64 bytes	20 MHz	128 bytes
16F877A	8 kbytes	368 bytes	20 MHz	256 bytes
18F452	8 ko	1536 bytes	40 MHz	256 bytes
PIC32	32-512 kbytes	8-32 KB	80 MHz	//

Les déférences fondamentales entre ces PICs sont généralement dans la capacité de la mémoire, la vitesse d'horloge et le nombre de pins

I.8.7 Choix d'un PIC [5]

Le choix d'un PIC est lié à l'application envisagée, en se basant sur les directives suivantes :

- Il faut dans un premier temps déterminer le nombre d'entrées-sorties nécessaires pour l'application. Ce nombre nous donne la famille du PIC.
- Il faut déterminer si l'application nécessite un convertisseur analogique numérique. Ce qui va centrer un peu plus vers le choix d'un PIC.
- La rapidité d'exécution est un élément important, il faut consulter les databook pour vérifier la compatibilité entre la vitesse du PIC choisi et la vitesse maximale nécessaire au montage.
- La taille de la RAM et la présence ou non d'une EEPROM pour mémoriser les données sont également importantes pour l'application souhaitée.
- La longueur du programme de l'application détermine la taille de la mémoire du PIC recherchée.

Selon l'application, il appartient à l'utilisateur de choisir la famille qui répond au mieux à son cahier de charge.

I.9 Les différents modèles de programmation

La programmation peut se faire par deux méthodes, soit en langage assembleur ou langage évolué tel que : pascal, basic, C,

Le choix d'un langage dépend essentiellement de l'occupation en mémoire. Si en utilise un langage évolué on aura d'une part des mots puissants et directe, donc le programme comportera peu de mots et automatiquement peu d'erreurs, d'autre part il nécessite un logiciel précisément conçu pour la programmation des PICs, ce dernier est généralement couteux et exige plus d'espace en mémoire, car une instruction de ce langage peut se traduire par plusieurs lignes en langage machine

I.10 PIC C Compiler

PIC C Compiler est un outil de développement spécifique aux microcontrôleurs PIC, conçu et fourni gratuitement par MICROCHIP, il permet d'écrire, mettre au point et optimiser les programmes, sous PICs. En effet, en plus d'un éditeur et d'un assembleur, il inclut un ensemble d'outils permettant non seulement de fabriquer le code objet d'une application, mais aussi de simuler le programme, c'est à dire le voir dérouler à l'écran.

Il offre beaucoup de flexibilité aux développeurs, notamment grâce aux nombreuses fenêtres pouvant être ouvertes à tout moment lors d'une mise au point, permettant de voir le contenu d'un quelconque emplacement mémoire et d'un quelconque registre**.[3]**

I.11 Les outils de développement MICROCHIP [3]

Ces outils de développement peuvent être classés en trois catégories principales

- Les outils de génération de code, incluant évidement l'assembleur mais aussi divers compilateurs pour les langages C ou logique floue notamment.
- Les programmes dans le microcontrôleur permettent de programmer les différents mémoires.
- L'émulateur qui représente évidemment l'offre haut de gamme ou matière d'outils de développement.

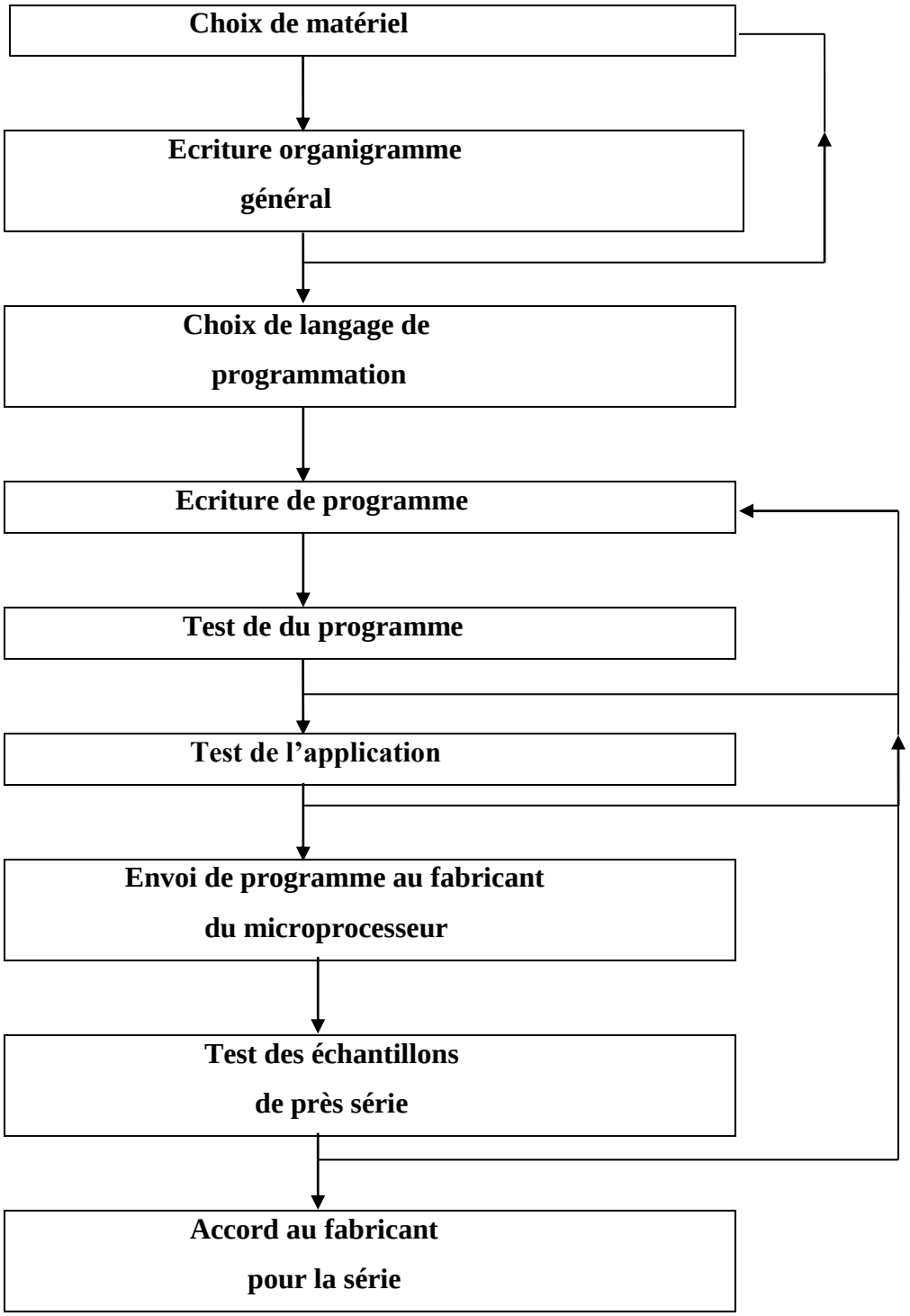


Figure I.6 Organigramme de principes de développement d'une application [3]

I.12 Conclusion

Dans ce chapitre nous avons présenté en général, les systèmes automatisés et comment choisir une technologie d'automatisation, les microcontrôleurs, les différentes familles des PICs ainsi que leurs programmations. Dans le deuxième point nous allons présenter les différents outils de conception d'un afficheur, dans le troisième point on s'est intéresser à la simulation des données de l'afficheur. Le quatrième point aborde le sujet de la conception Virtuel, quand désigne comme un processus itératif au cours duquel un objet est conçu et modifié afin qu'il puisse remplir des fonctions bien définies et se conformer à un ensemble de contraintes. Puis le cinquième point, conception matériel, qui est une synthèse sur la conception et la réalisation de notre afficheur avec la maté c'est le système final, doit être conforme aux exigences de fonctionnement établies dans le cahier des charges et on terminera avec une conclusion générale sur notre travail.

Chapitre II

les différents outils pour la conception de l’afficheur

II.1 Introduction

Ce chapitre est consacré à la présentation de la carte PICPLC16 v6 et microcontrôleur PIC18f4520.

On aura à définir les différentes parties de la carte PICPLC16 v6, elles permettent aux PICs d'être connectés avec des circuits externes.

On aura aussi à définir l'unité de contrôle de pic 18f4520 et tout ce qui l'englobe de part et d'autre de sa structure. La partie la plus importante sera celle qui traite les différents ports d'entrée/sortie, ainsi que les Timers.

II.2 Les diffèrent composants de la carte PICPLC16 v6

La figure II.1 montre les diffèrent composants de la carte PICPLC16 v6 :

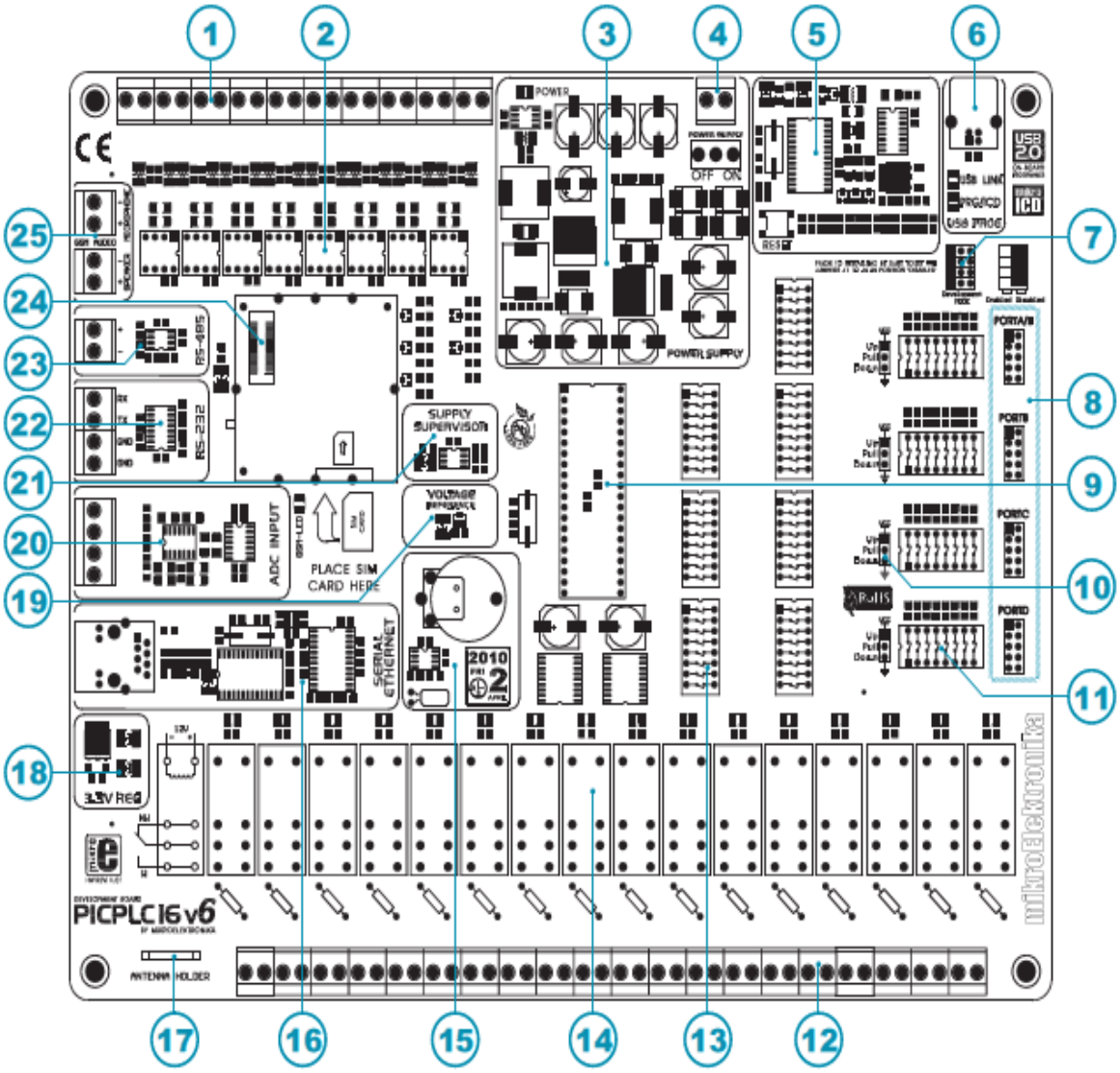


Figure II.1 La carte de développement PICPLC16 v6 [7].

- 1. connecteur pour optocoupleur
- 2. un optocoupleur
- 3. un régulateur de tension d'alimentation
- 4. connecteur d'alimentationCN1
- 5. programmateur PIC flash USB 2.0
- 6. connecteur USB
- 7. le programmateur sous la carte PICPLC16 v6.
- 8. portes entrées/sorties
- 9. microcontrôleur
- 10 .jumper de selection pull-up/pull-down
- 11. DIP Switch pour connecter les résistances à chaque broche
- 12. Connecteurs pour relier des périphériques externes avec relais
- 13. Commutateurs DIP à Active / désactive des modules
- 14. relais
- 15. horloge temps réel
- 16. un module Ethernet
- 17. Support d'antenne
- 18. régulateur de tension 3.3 v
- 19. la source de tension de référence
- 20. A /D entrée de test de convertisseur
- 21. Alimentation contrôle de la tension
- 22. interface de communication RS-232
- 23. communication RS-485
- 24. le connecteur GSM
- 25. connecteurs pour haut-parleur et microphone

II.3 Le microcontrôleur

La carte de développement PICPLC16 V6 comporte un PIC18f4520 de 40 pins dans le paquet DIP40 comme il est montre sur la figure II.2 :

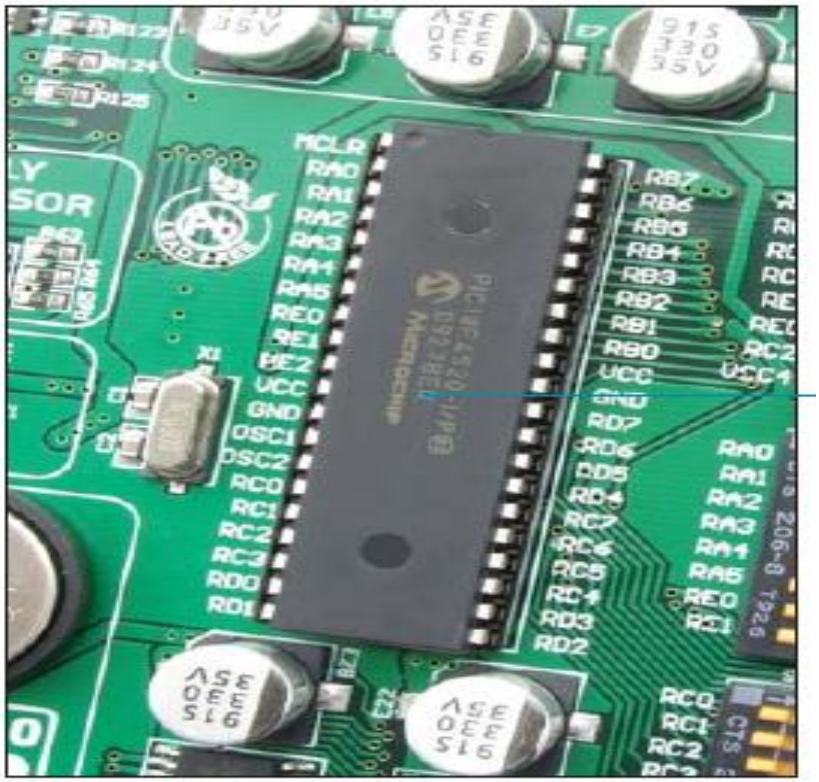


Figure II.2 Le PIC18f4520 en DIP40 [7]

II.4 Le programmateur PIC flash USB 2.0

Un programmateur est un outil nécessaire quand on travaille avec microcontrôleur. Dispositif PICPLC16 v6 a un flash de pic à bord avec programmateur qui fournit une interface entre le microcontrôleur et le PC. Le programme flash pic est utilisé pour le chargement du fichier (HEX) dans le microcontrôleur la figure **II.3** montre la connexion entre le compilateur, le programmateur et le microcontrôleur.

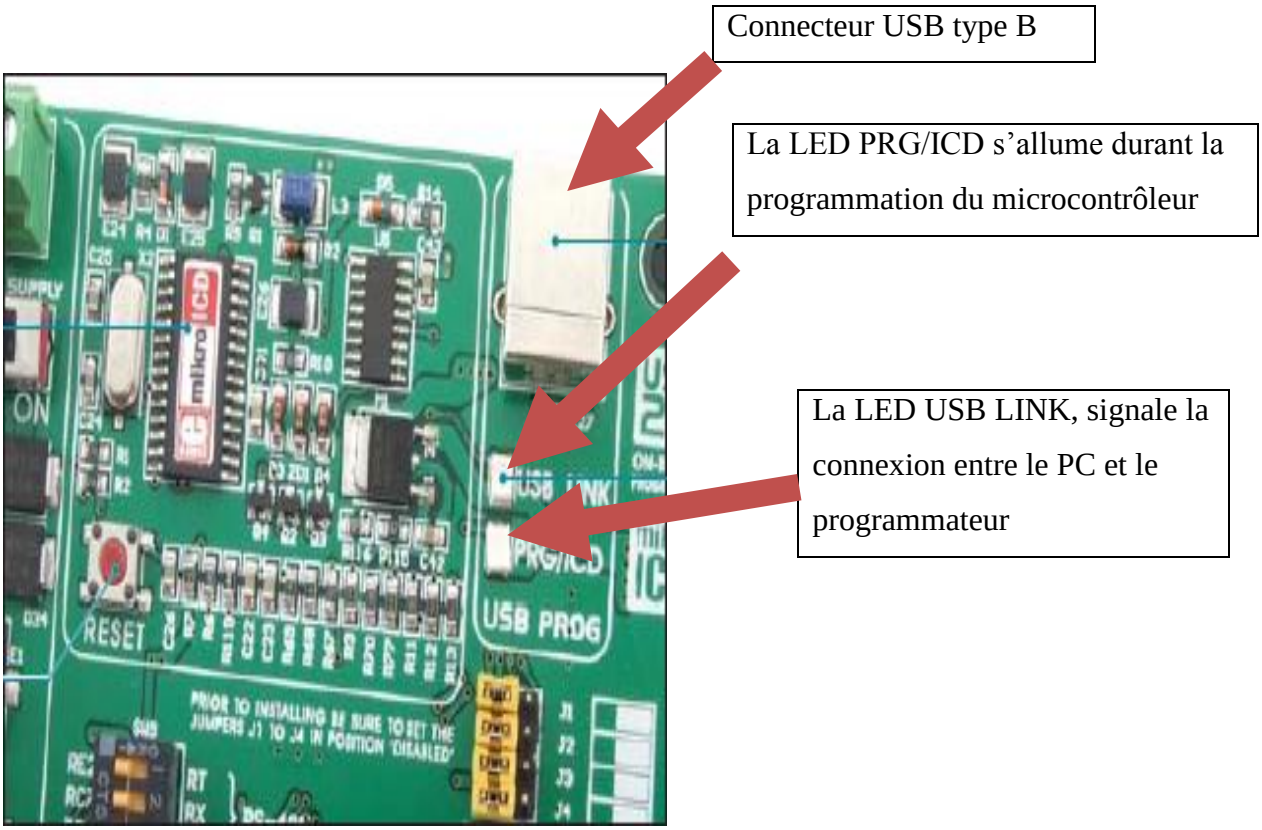


Figure II.3 Le programmateur PICflash [7].

II.5 Alimentation d'énergie

Le système de développement de la carte PICPLC16v6 est connecté à la source d'alimentation d'énergie par l'intermédiaire du connecteur CN1 .La tension d'alimentation électrique peut être soit continu ou alternatif .Un DC tension d'alimentation peut être de l'ordre de16V à 30V, alors que l'alimentation AC la tension peut varier entre12et 22v. Avoir à l'esprit que le programmateur à bord ne peut pas fonctionner sans être connecté à la source d'alimentation mais il est connecté à un PC avec le câble USB.

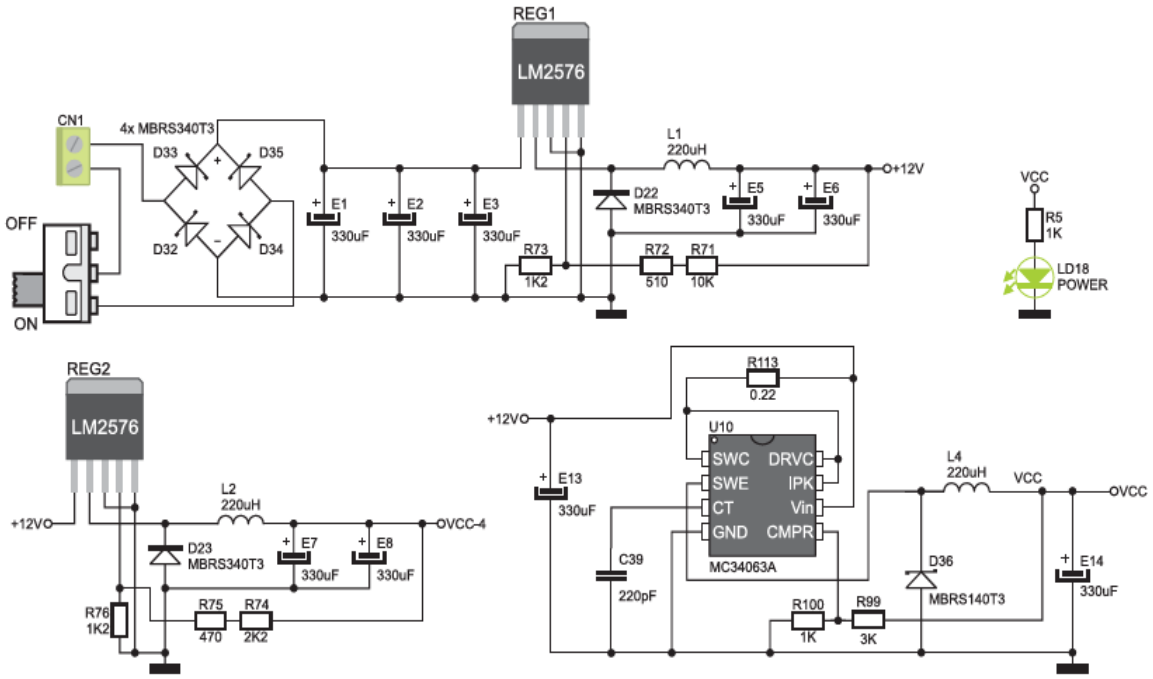


Figure II.4 Schéma d'alimentation d'énergie [7].

II.6 Interface de communication RS-232

USART est l'une des façons les plus courantes de l'échange de données entre le PC et des unités périphériques. La communication série RS-232 est effectuée avec le connecteur CN4 et CN5 et le module USART du microcontrôleur. Le port RS-232 sur la carte PICPLC16 v6 utilise le commutateur marqué comme RX232 et TX232 sur le SW11 de commutateur DIP pour permettre à ce port .Les broches du microcontrôleur utilisé pour la communication RS-232 sont marques comme suit:

- RX Ligne de réception de données
- TX émission taux

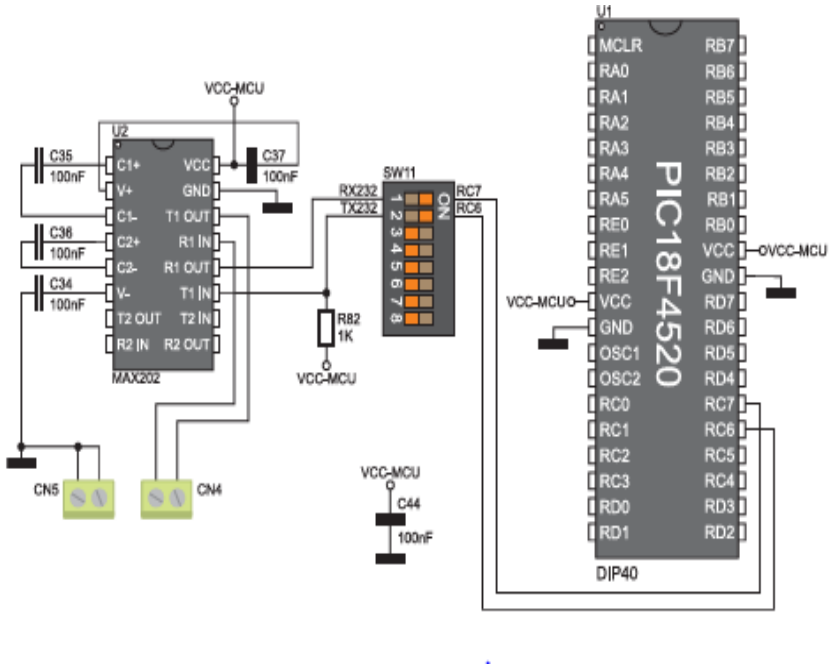


Figure II.5 Schéma d'interface de communication RS-232 [7].

II.7 La communication RS-485

La communication RS-485 permet le transfert des données point par point et point à multipoint. Elle est utilisée généralement pour le transfert des données entre plusieurs microcontrôleurs.

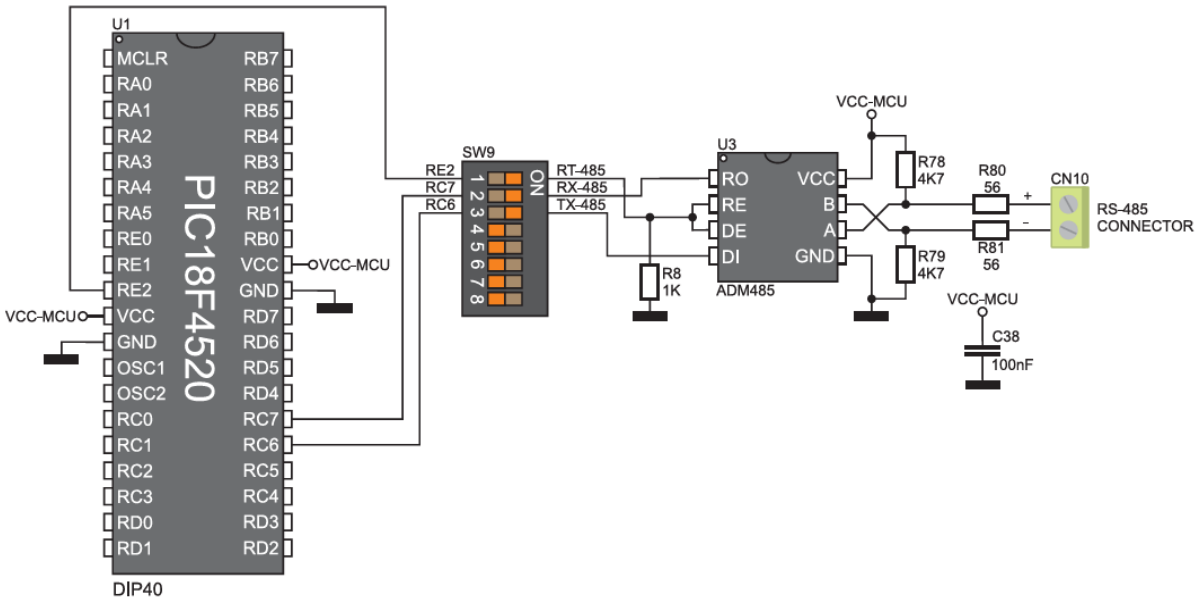


Figure II.6 Schéma de module RS-485 [7].

II.8 Module Ethernet

La fonction de système de développement picplc16 v6 sur le module Ethernet fournissant une interface entre le microcontrôleur et le LAN (réseau local). Le contrôleur ENC28J60 autonome permet la communication Ethernet sur la carte. La tension 3,3V est requise pour le fonctionnement de ce contrôleur. pour permettre de transférer les données au microcontrôleur alimenté avec la tension d'alimentation 5 V, il est nécessaire de régler ces niveaux de tension au moyen d'émetteurs/récepteurs et 74LVCC3245 74LVC1T45. Pour permettre la connexion entre le module Ethernet et le microcontrôleur, commutateurs 1,2 et 3 sur la sw10 DIP SWITCH, ainsi que les commutateurs 4, 5 et 6 sur le DIP SW9 Switch doit être réglé sur la position ON.

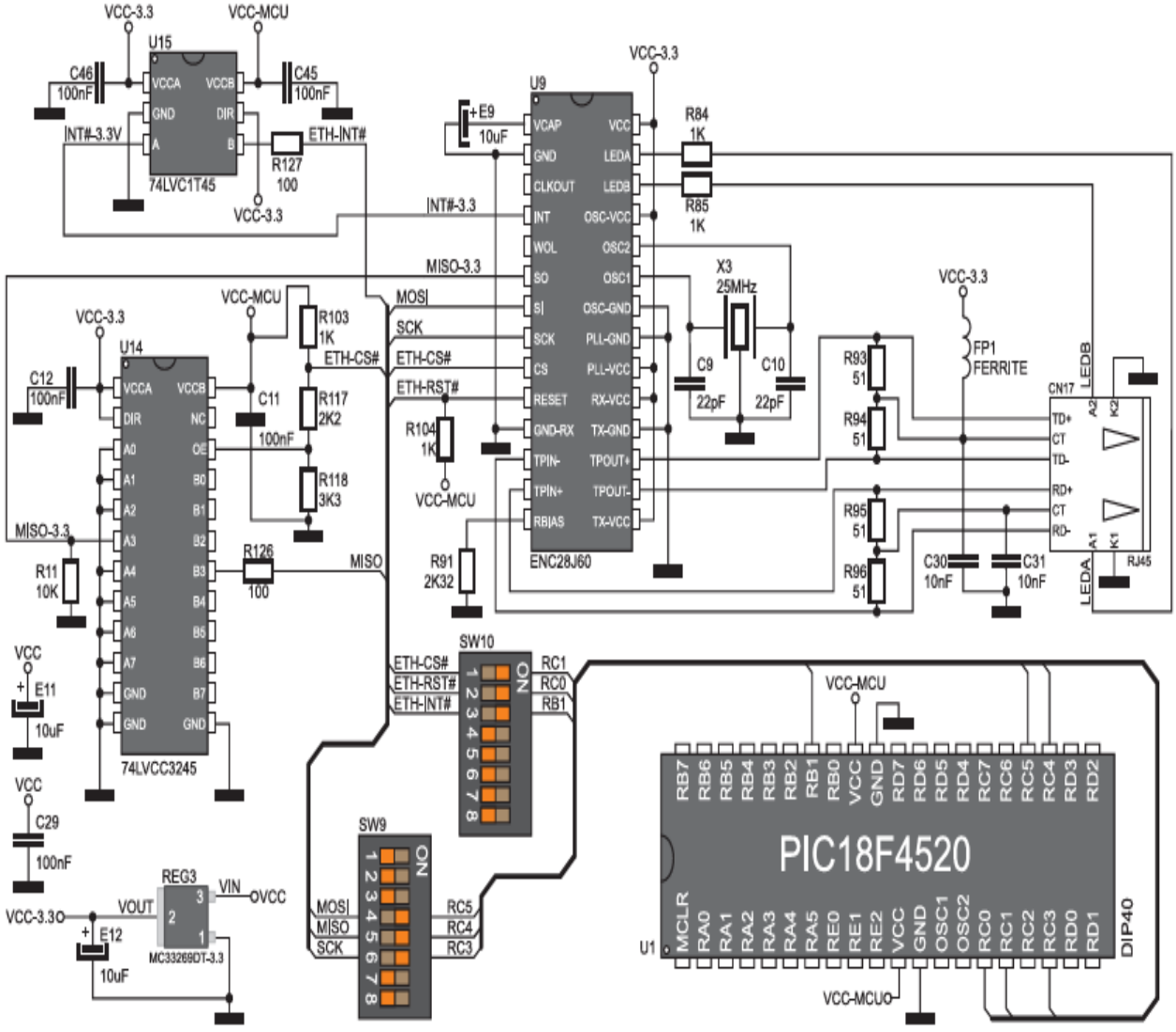


Figure II.7 Schéma du module Ethernet [7].

II.9 Le connecteur GSM

Grâce à un connecteur intégré pour le module GSM, le système de développement de PICPLC16v6 est capable de communiquer avec le monde extérieur en utilisant le réseau GSM. Un module GM862 de TELIT GSM/GPRS-GSM862 peut être commandé avec la carte PICPLC16 v6. Ce module comprend une fente pour placer une carte SIM et un connecteur pour une antenne externe. Pour le module GSM pour être connecté au microcontrôleur, il est nécessaire de régler les commutateurs3-8sur le DIP Switch SW11sur la position ON. Dans le cas où le module GSM est utilisé pour la communication audio, il est nécessaire de brancher un haut-parleur et un microphone dans des connecteurs appropriés, comme il est représenté sur la figure II.8 :

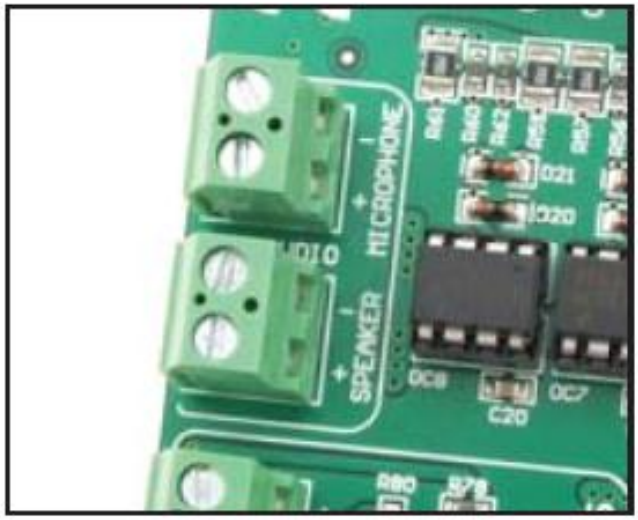


Figure II.8 Connecteur audio [7].

En plus de la transmission de signaux audio, le module GSM peut être utilisé pour envoyer des données conformément à la norme GPRS utilisés dans l'application mobile.

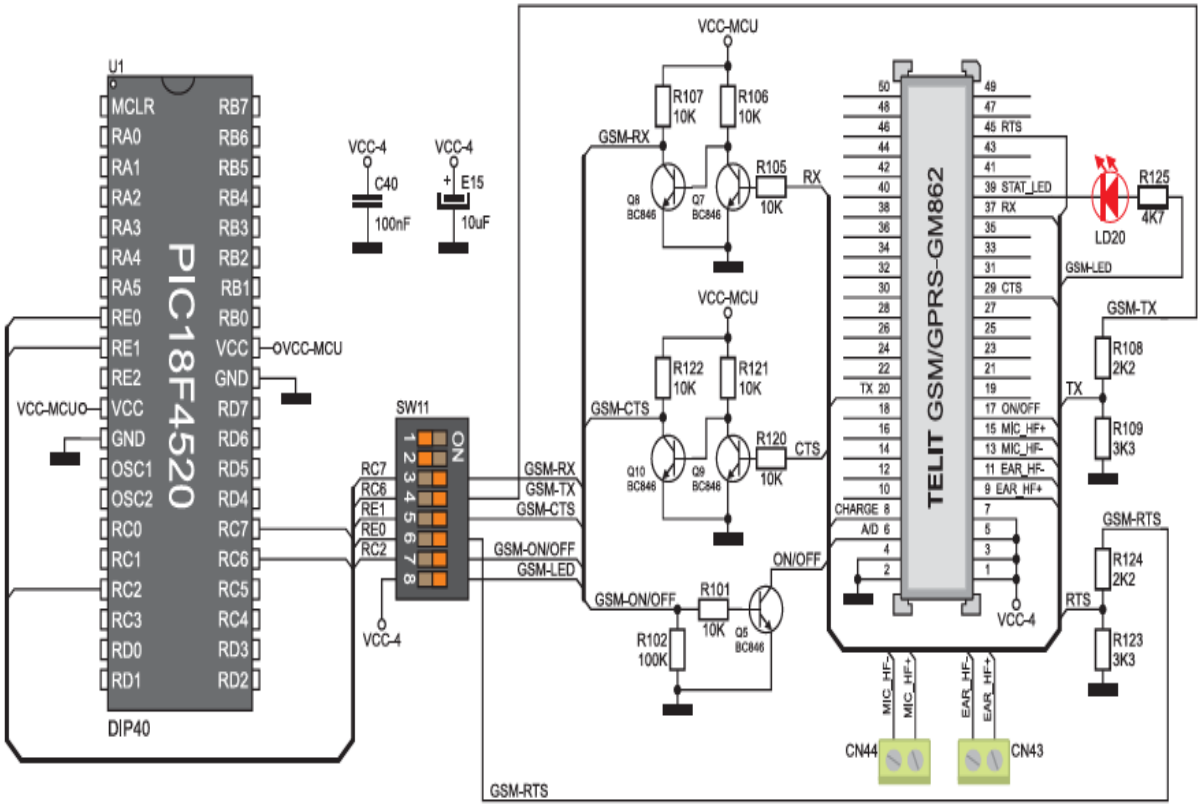


Figure II.9 Schéma de microcontrôleur et GSM connecteur [7].

II.10 Convertisseur d'essai (le module A/N)

Un convertisseur A/N est utilisé pour convertir un signal analogique en une valeur numérique appropriée. Un convertisseur A/N est linéairement dépendant de la valeur de tension d'entrée. Le circuit MCP3204 est utilisé comme un convertisseur A / N sur la carte PICPLC16 v6. Le système tension à convertir il a 12 bits est présenté aux broches d'entrée du convertisseur A / N au moyen de l'amplificateur opérationnel MCP6284 .le résultat de conversion est transféré au microcontrôleur. pour le signal numérique à transférer au microcontrôleur, il est nécessaire de mettre en Switch 4,5,6 et 7 sur le plongeur Switch SW9 en position de marche.

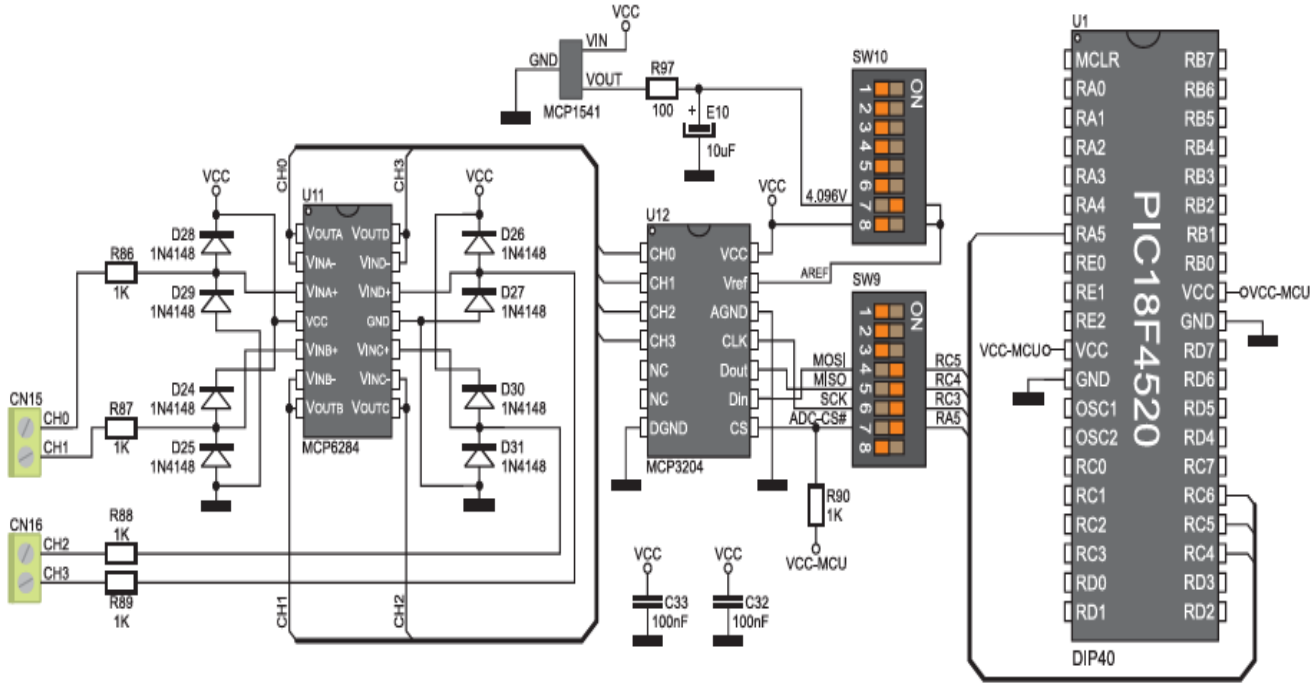


Figure II.10 Schéma de module A/D [7].

II.11 Horloge en temps réel (RTC)

Le système de développement de picplc16v6 a un circuit intégré DS1307, est capable de maintenir le temps réel. les principales caractéristiques de l'horloge temps réel sont les suivantes:

- Fournir des informations sur les secondes, minutes, heures et jours de la semaine.
- Détection automatique des coupures d'alimentation.
- La consommation d'énergie inférieure à 500 nA.

L'horloge en temps réel est largement utilisé dans les dispositifs d'alarme, les contrôleurs industriels, les appareils grand public, etc .L'horloge en temps réel fournis sur le système de développement de la carte PICPLC16 est utilisé pour générer une interruption .afin d'établir la connexion entre le microcontrôleur et l'horloge en temps réel, il est nécessaire de régler les commutateurs4,5 et 6 sur le Switch plongeur SW10 sur la position ON.

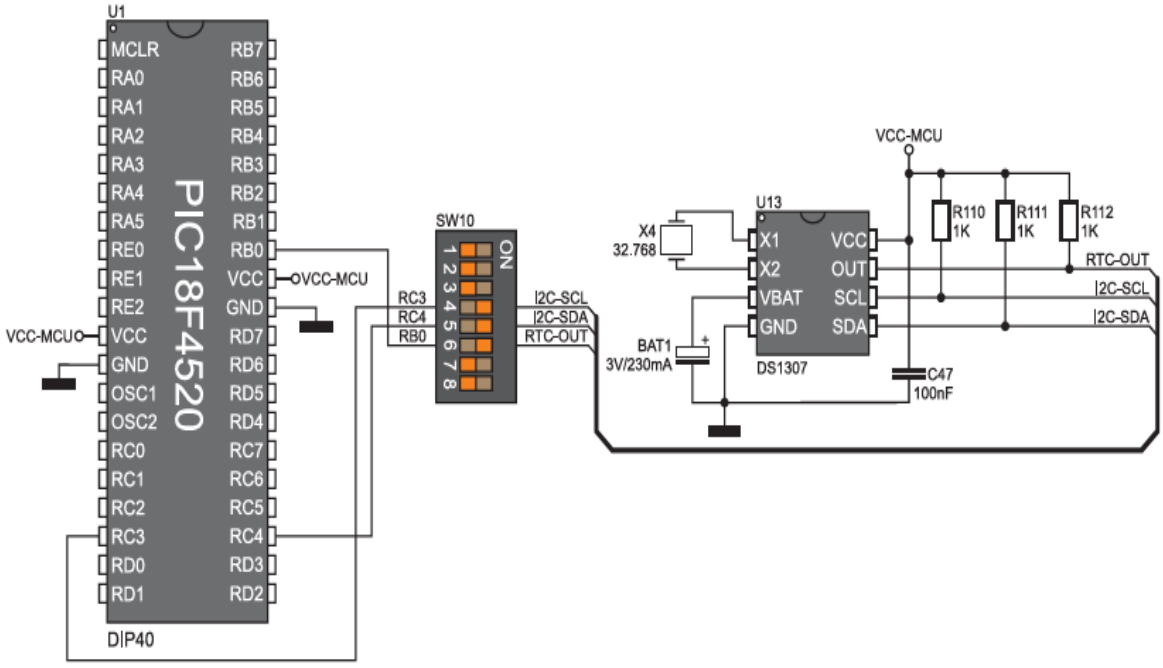


Figure II.11 Schéma d'horloge en temps réel [7].

II.12 Les optocoupleurs

La carte PICPLC16 v6 à 16 entrées optocoupleurs .il est employée couramment dans des applications industrielles ou des entrées qui doivent être électriquement isolé dans le reste de la carte de développement. Il sert à protéger le microcontrôleur contre les impulsions électrique qui pourraient se produire sur des lignes d'entrées.

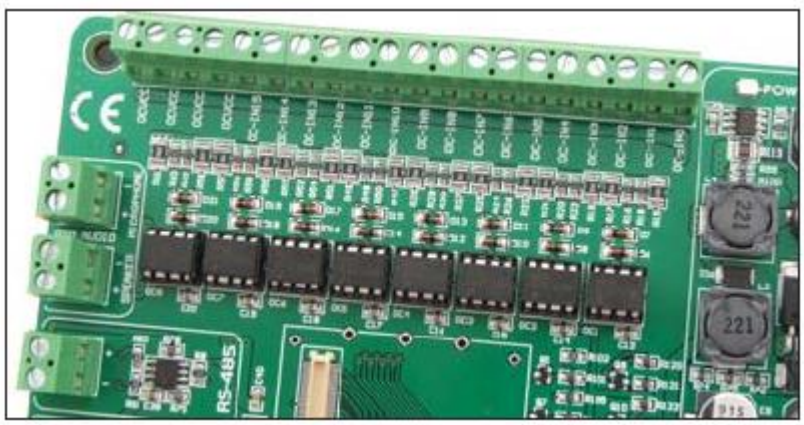


Figure II.12 Les optocoupleurs [7].

II.13 Les relais

Les dispositifs industriels utilisent habituellement des puissances plus fortes que microcontrôleur peut fournir par l'intermédiaire de ses ports d'entrée sortie .Pour permettre au microcontrôleur d'être relié à des tels dispositifs, le système de développement est équipé de 16 relais qui peuvent fournir jusqu' a 250V

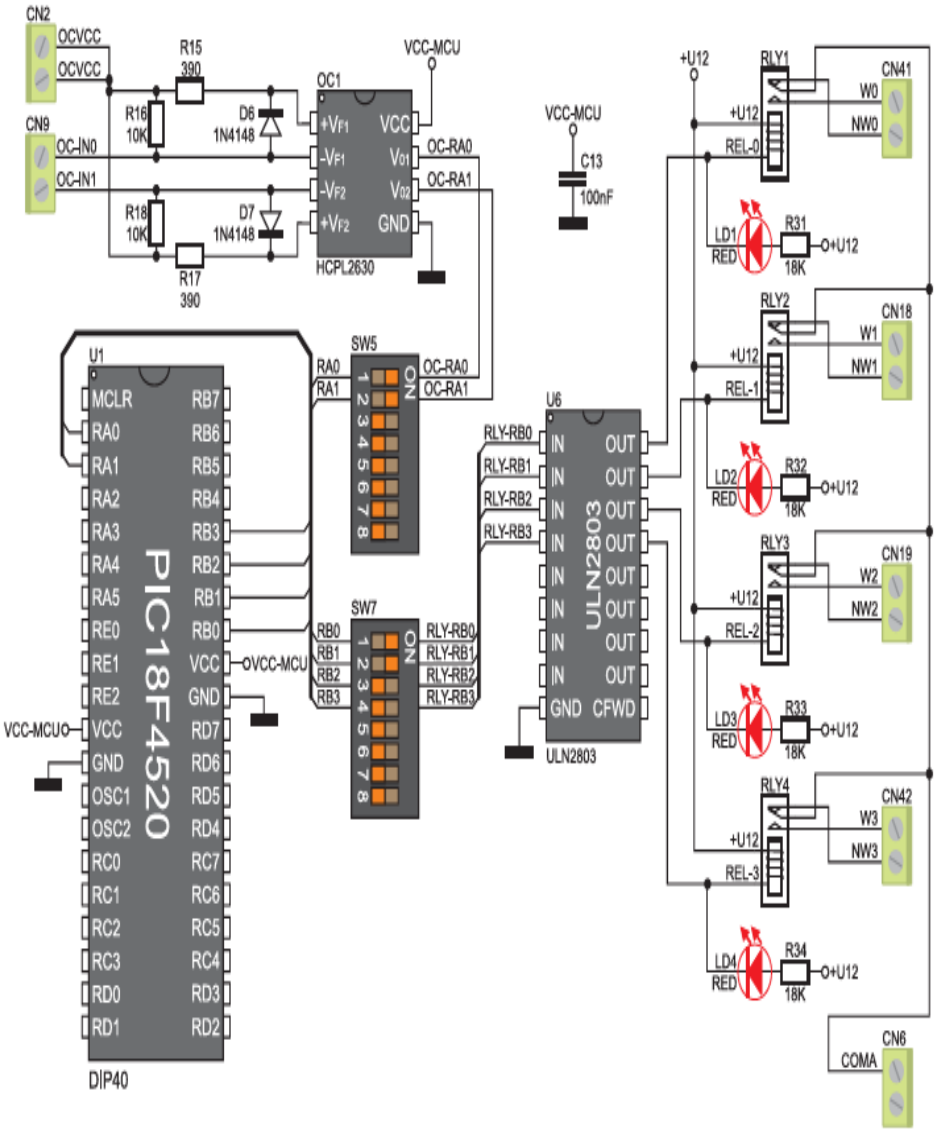


Figure II.13 Schéma des relais et les optocoupleurs [7].

II.14 Les portes entrée /sortie

La carte PICPLC16 v6 contient 4 connecteur males de 10 branches relies aux ports d' E/S du microcontrôleur .Les branches du microcontrôleur ne sont pas directement reliées au connecteur, mais par l'intermédiaire des jumpers les interrupteur a position multiples SW1-SW4 servent a relier les branche aux résistances de tirage pull-up / pull-down. La position des jumpers j5-18 détermine si les portes utilisant une résistance de pull-up ou pull-down.

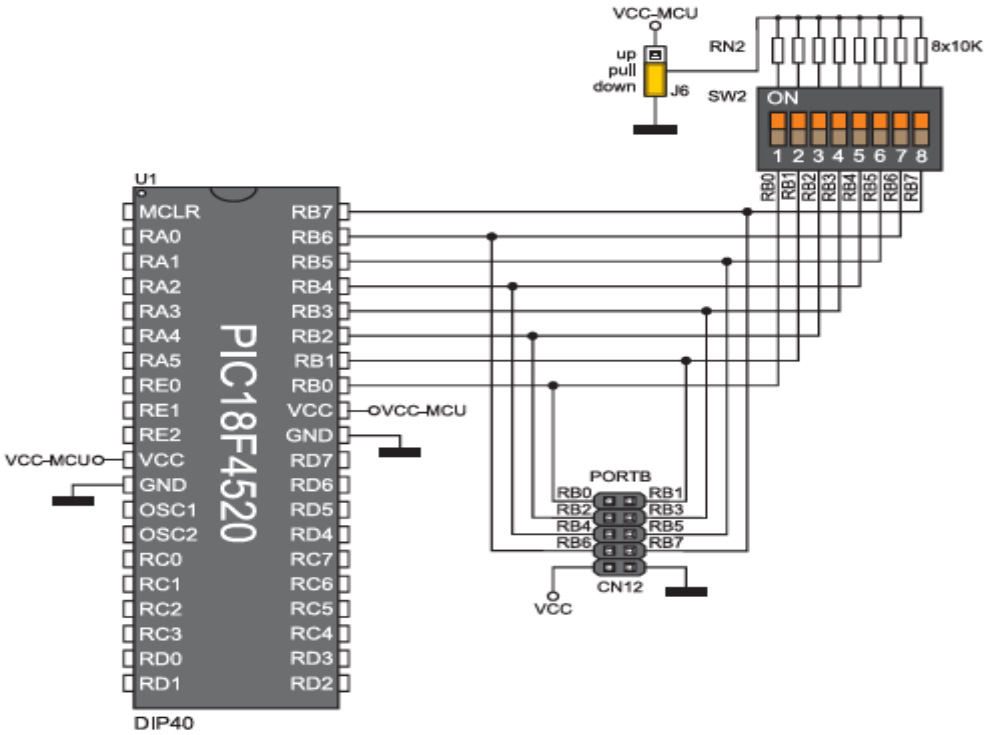


Figure II.14 Schéma des portes entrées /sortie [7].

II.15 Présentation du PIC18f4520

Le schéma ci-dessous représente clairement l'architecture externe du PIC utilisé :

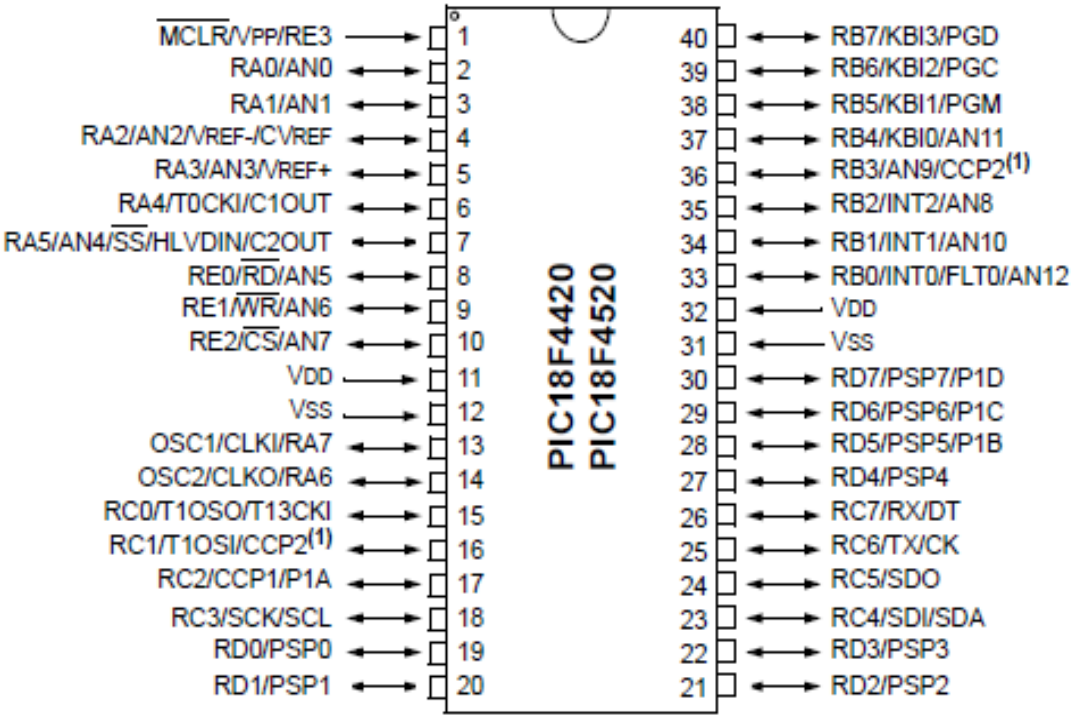


Figure II.15 Architecture externe du pic 18f4520.[7]

On peut distinguer sur ce schéma :

- L'alimentation : V_{DD} (+5V) et Vss (0V).

-Les bornes du quartz (oscillateur a quartz) : OSC1 et OSC2.

- Entrée RESET (MCLR: Master Clear).

- Les différents ports d'Entrées/Sorties : PORTA, PORTB, PORTC, PORTD, PORTE.

II.16 Architecture interne du PIC18f4520

Le schéma ci-dessous représente clairement l'architecture interne du PIC :

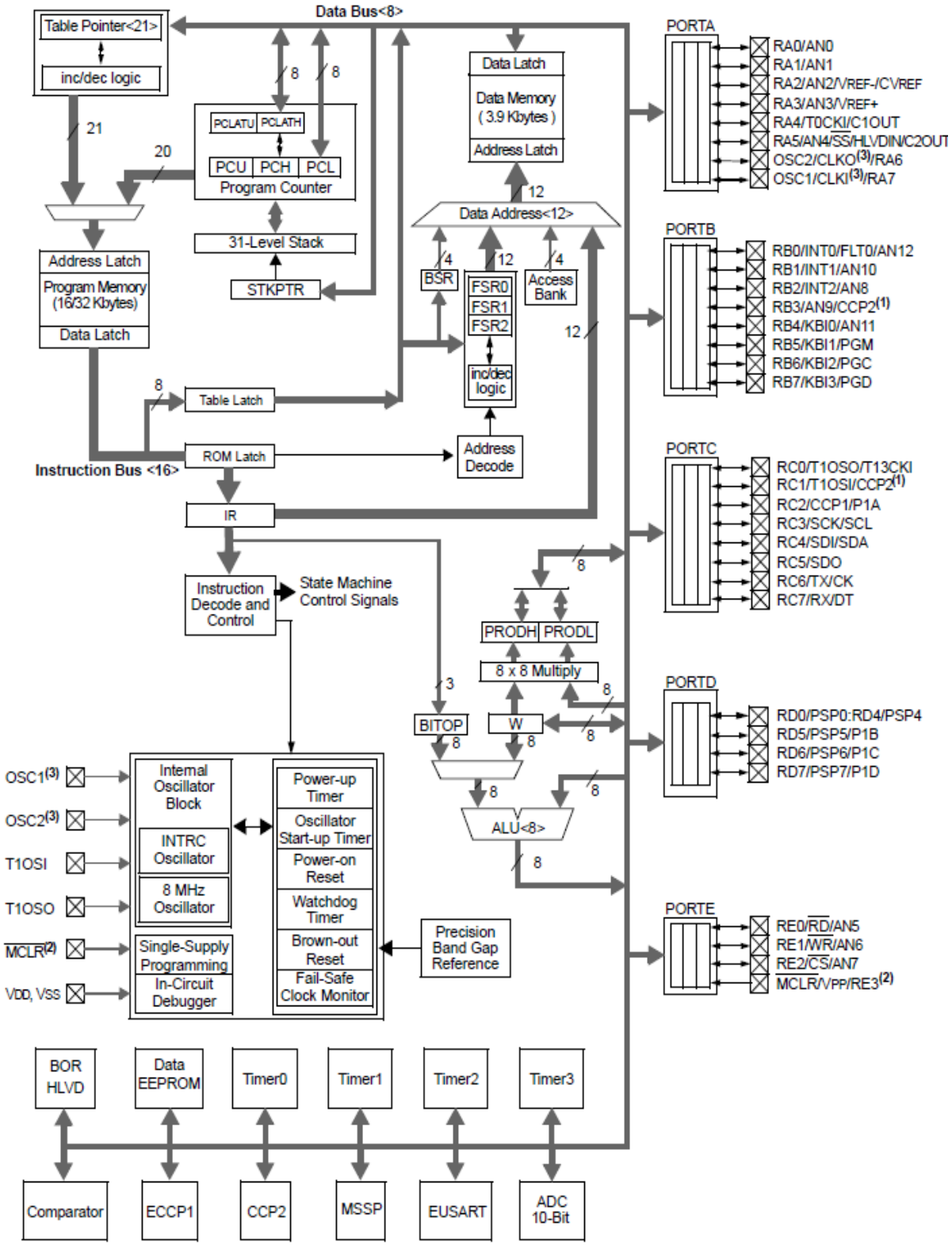


Figure II.17 Architecture interne du pic 18f4520.[7]

II.17 Caractéristiques du PIC18F4520

Le PIC18f4520 est un microcontrôleur fournir par l'entreprise Américaine MICROCHIP technologie, ses caractéristiques sont [7] :

- Vitesse maximal de 40 MHz.
- Mémoire de programme 32 KB.
- Mémoire RAM 1664B.
- 4 Timers (Timer0 sur 8- bits et 3 Timers sur 16 bits).
- Jusqu'à 20 sources d'interruption.
- EEPROM 256B.
- Un convertisseur analogique-numérique (CAN) 10 bits
- Deux modules de génération d'impulsion à période réglable (PWM)
- Un module de communication série synchrone (MSSP)
- Une USART
- Un module de communication en ≪ port parallèle ≫

II.18 Classification des éléments

On peut classer ces éléments en trois classes : le noyau, les périphériques et les fonctions spéciales.

II.18.1 Le noyau :

C'est la partie commune à tout les PICs, elle contient les éléments de base pour le fonctionnement du microcontrôleur, parmi eux on trouve :

- L'unité Arithmétique et Logique(UAL).
- Les mémoires : mémoire de programme, mémoire RAM et mémoire EEPROM.
- Les RESETs: MCLR, POR, BOR, PER.
- L'oscillateur.

II.18.2 Les périphériques :

Cette partie est propre au PIC, c'est elle qui les diffère des microcontrôleurs, elle permet de communiquer avec l'environnement extérieur, elle contient :

- Les ports d'entrées/sorties.
- Le convertisseur analogique/numérique(CAN).
- Le port parallèle esclave PSP.

- Les CCP1/2(compare/capture/PWM).
- L'USART.
- Le SSP.

II.18.3 Les fonctions spéciales :

Elles permettent d'améliorer la fiabilité du système et parmi ces fonctions on trouve :

- Les différents RESETs.
- Les Timers et chien de garde WDT.
- L'oscillateur interne RC.

II.19 Organisation de la mémoire

Il y a trois zones de mémoire dans l'architecture; mémoire de programme, mémoire de données et la mémoire EEPROM, la structure Harvard des PICs fournit un accès séparé à chacune.

II.19.1 Mémoire de programme (Flash) :

C'est une mémoire réinscriptible qui conserve ses données lorsque le PIC n'est pas alimenté. Elle est utilisée pour stocker le programme.

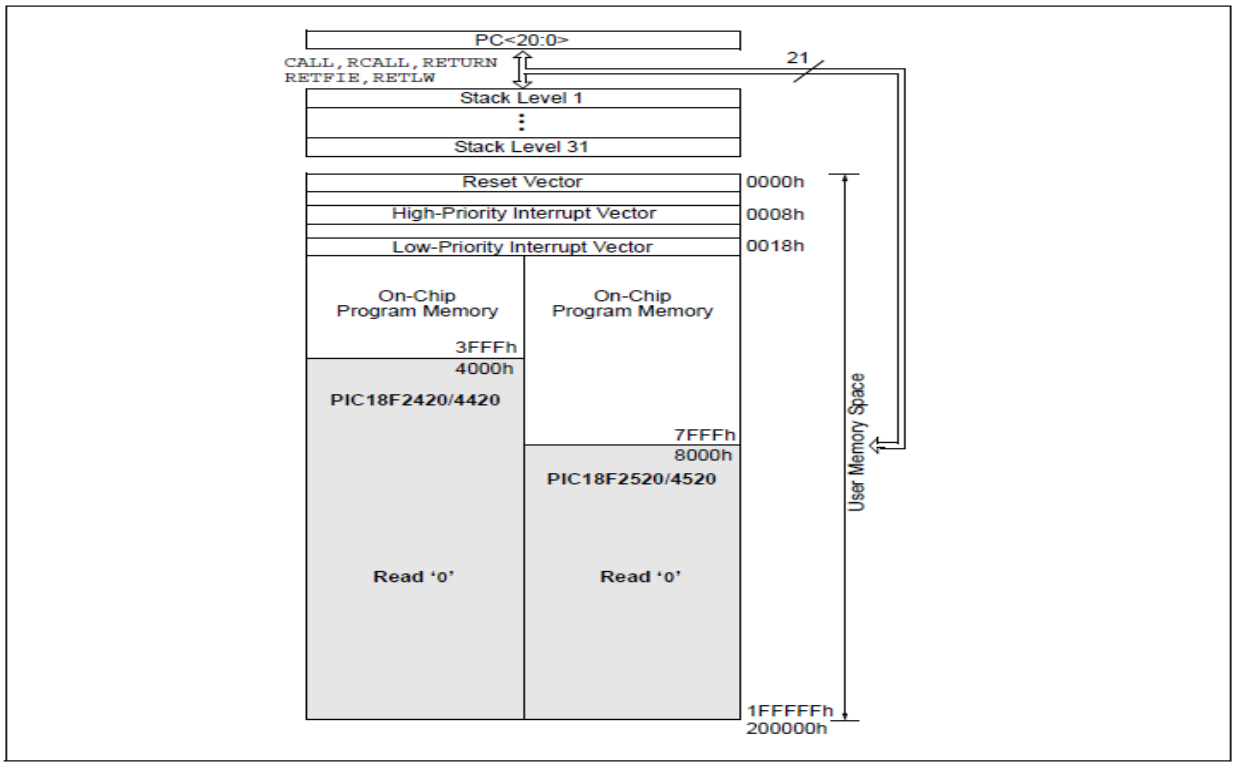


Figure II.18 Organisation de la mémoire et de la pile.[7]

Comme le montre la figure II.18, la mémoire s'étale linéairement de 0000h à 1FFFFFh. Le vecteur de RESET est placé à l'adresse 0000h, le vecteur d'interruption prioritaire élevé est à l'adresse 0008h et le vecteur faible priorité d'interruption est à 0018h, la pile est à 31 niveaux de 21 bits chacun, elle est utilisée lors d'un appel de sous programme ou de service d'interruption pour sauvegarder le contenue de compteur ordinale PC avant l'exécution de ce dernier afin de permettre au PIC de savoir qu'elle est l'instruction suivante à exécuter.

II.19.2 Mémoire RAM :

C'est une mémoire volatile c'est- à-dire qu'elle s'efface quand le PIC n'est plus alimenté. Les variables utilisées au programme sont stockées à cet endroit.

II.19.3 EEPROM :

La particularité de ce genre de mémoire qui se comporte comme une mémoire vive et de conserver son contenue pendant une durée indéterminée, même en absence d'alimentation. Elle est d'une taille de 256 bytes. Pour accéder à cette mémoire on utilise les registres suivants :

- EEDATA : contient la donnée à enregistrer ou celle issue de la mémoire.
- EEADR : contient l'adresse de la case mémoire dont on écrit ou on lit.
- EECON1 et EECON2 : ces registres permettent de définir le mode de fonctionnement de la mémoire ou de contrôler son accès.

II.20 Les ports d'entrée/sortie (I/O)

Les ports d'entrée/sortie font partie des ressources les plus utilisées sur les microcontrôleurs, ils sont très variable en nombre, ces derniers sont liés aux dimensions du boitier donc aux nombres de pattes disponible.

Le PIC18f4520 contient cinq ports d'entrée/sorties: A, B, C, D et E qui permettent la communication entre le processeur et le milieu extérieur. Ils peuvent servir d'E/S numériques standards ou d'E/S de périphériques internes lorsqu'il s'agit d'une fonction spéciale telle que liaison série ou parallèle, convertisseur A/N.

Ces ports sont bidirectionnels, leurs configurations se fait par des registres spécifiques.

II.21 Le reset

Le reset est définit comme étant l'évènement qui redémarre le programme. Pour le PIC18F4520 il a plusieurs sources de reset :

- POR: Power-on Reset.
- MCLR: Master Clear Reset Pin.
- WDT: Watchdog Timer.
- BOR: Brown-out Reset.
- Reset Instruction**.**
- Stack Full Reset.
- Stack Under flow Reset.

II.22 Timers

D'une façon générale la fonction Timers se charge de la gestion du temps. Le Timer peut être cadencé par une horloge externe ou interne.

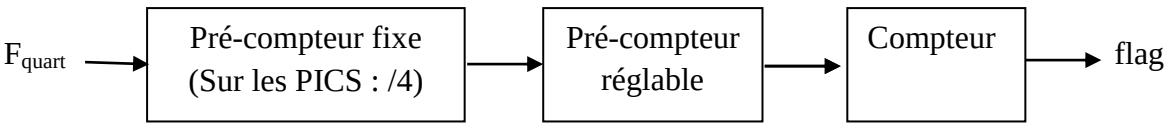
Tous les microcontrôleurs ont un ou plusieurs Timers pour gérer le temps. Le pic 18f4520 possède 4 Timers (timer0….timer3).

Un Timer est un circuit périphérique (inclut dans le pic) lancé par le programme et qui compte un nombre prédéfini d'impulsion du quartz. Ces impulsions peuvent provenir du :

- Quartz cadençant le microcontrôleur, on parle alors d'un fonctionnement en compteur de temps.
- D'une entrée TOR du pic (exemple : RA4 pour timer0, RC0 pour timer1), on parle alors d'un fonctionnement en compteur d'évènement extérieur. Quand ce nombre d'impulsion est atteint, un drapeau (flag en anglais) se lève.une interruption peut être également générée.

Les Timers ont presque tous le même fonctionnement, ils ont tous un compteur, un pré-compteur (prescaler en anglais).

L'horloge du microcontrôleur est pré-divisée pour obtenir une fréquence plus lente.dans le cas des PICs, l'horloge du quartz n'est pas directement connecté au pré-compteur mais passe par un diviseur par 4.



On règle le compteur à une valeur initiale et on attend qu'il atteigne une valeur finale. Lorsque le compteur atteint cette valeur finale, un indicateur (flag en anglais) devient actif.

Le temps mis par le compteur pour faire lever le flag est égal à :

$$T = \frac{1}{\frac{\text{F}}{\text{Valeur du précompteur fixe} \times \text{Valeur du précepteur réglable} \times \text{nombre d'impulsion}}}$$

II.22.1 Timer0

Le Timer0 peut être utilisé en mode 8 bits ou en mode 16 bits pour obtenir un temps plus long. Le Timer0 possède des sources d'horloges :

- Soit l'horloge du pic (l'horloge à quartz) divisée par4.
- Soit avec une horloge externe connectée sur la broche RA4/T0CKI.

Dans le cas de Timer0 en mode 8 bits, la formule du temps devient :

$$T = \frac{1}{\frac{\text{F}}{4 \times \text{Valeur du précompteur réglable} \times (256 - \text{valeur initiale})}}$$

Dans le cas de Timer1 et Timer0 en mode 16 bits, la formule du temps devient :

$$T = \frac{1}{\frac{\text{F}}{4 \times \text{Valeur du précompteur réglable} \times (65536 - \text{valeur initiale})}}$$

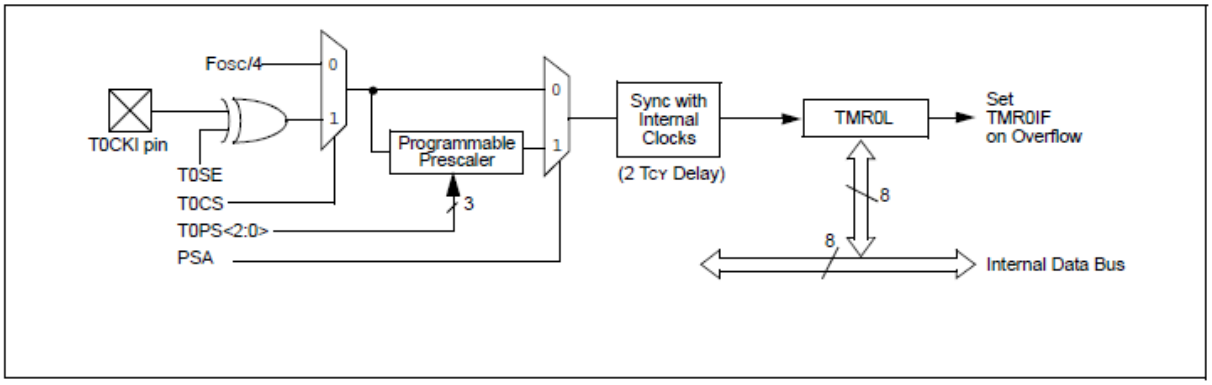


Figure II.19 Bloque diagramme du Timer0 (mode 8 bits).[7]

II.22.2 Timer1

Le timer1 fonctionne sur le même principe que le timer0 mais il est plus moderne dans sa conception. C'est un compteur à 16 bits divisé en 2 registres de 8 bits : TMR1L pour les poids faibles et TMR1H pour les poids forts. Son fonctionnement est résumé dans la figure ci-dessous.

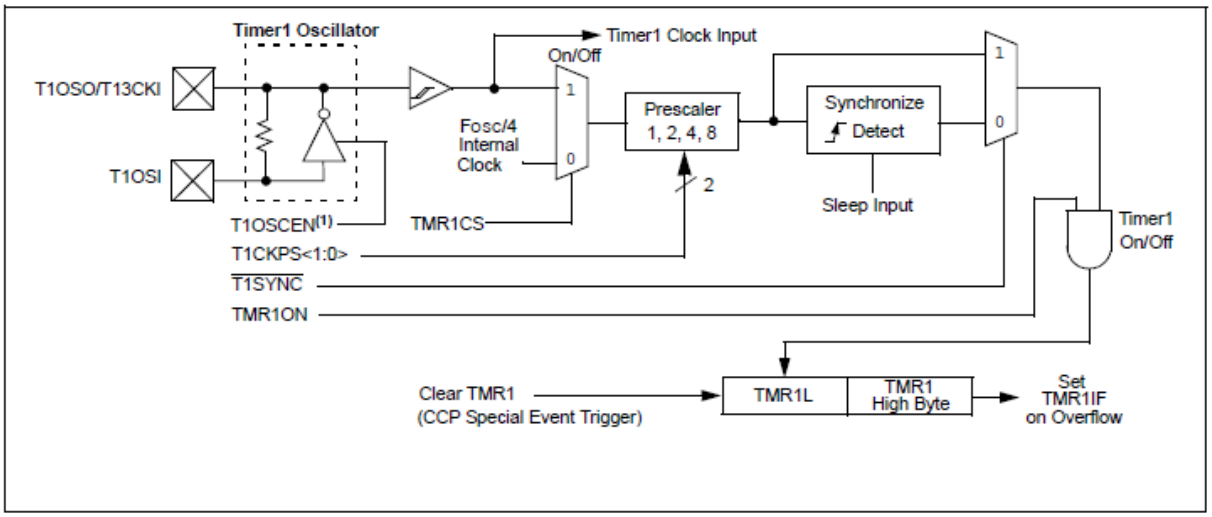


Figure II.20 Bloque diagramme du Timer1. [7]

Le timer1 possède deux sources d'horloge :

- Soit l'horloge du PIC(en général un quartz) divisée par 4.
- Soit avec une horloge externe connectée sur la broche RC0/T13CKI ou la broche RC1.

Le choix entre ces deux horloges se fait à l'aide du bit TMR1CS.

II.22.3 Timer2

Il est légèrement différent des Timer0, Timer1 et Timer3, puisque le début de comptage est obligatoirement 0x00 et que la fin de comptage est la valeur à entrer. C'est un compteur 8 bits, son horloge est divisée par 4(Fosc/4). Il est composé d'un registre de 8 bits TMR2, d'un pré-diviseur qui peut être paramétré avec l'une de ces trois valeurs : 1, 4 et 16, d'un post diviseur qui peut prendre des valeurs de 1 à 256, ainsi que d'un registre dit de période appelé PR2.

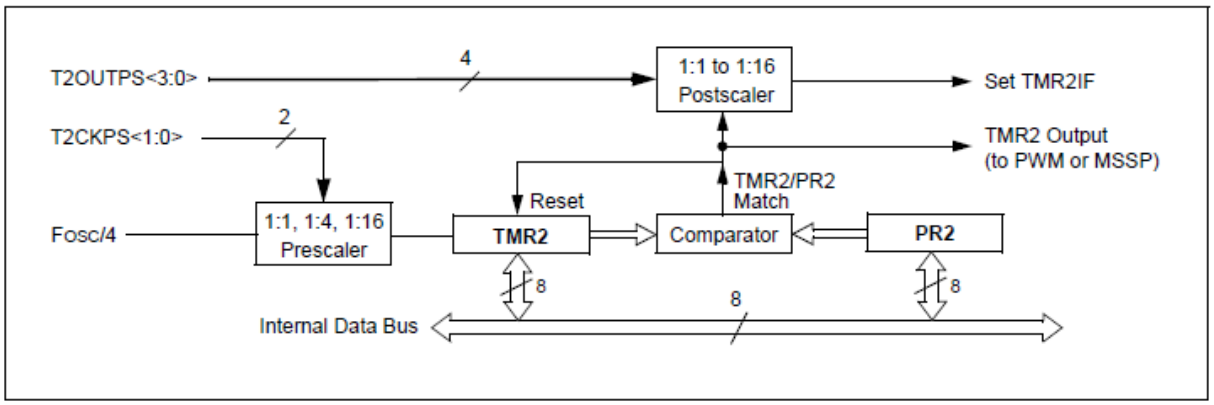


Figure II.21 Bloque diagramme du Timer2.[7]

II.22.4 Timer3

Il fonctionne comme le Timer0 et le Timer1.c'est un Timer de 16 bits, il est composé d'un registre de 8 bits TMR3, d'un pré-diviseur qui peut être paramétré avec l'une de ces valeurs : 1, 2, 4 et 8.

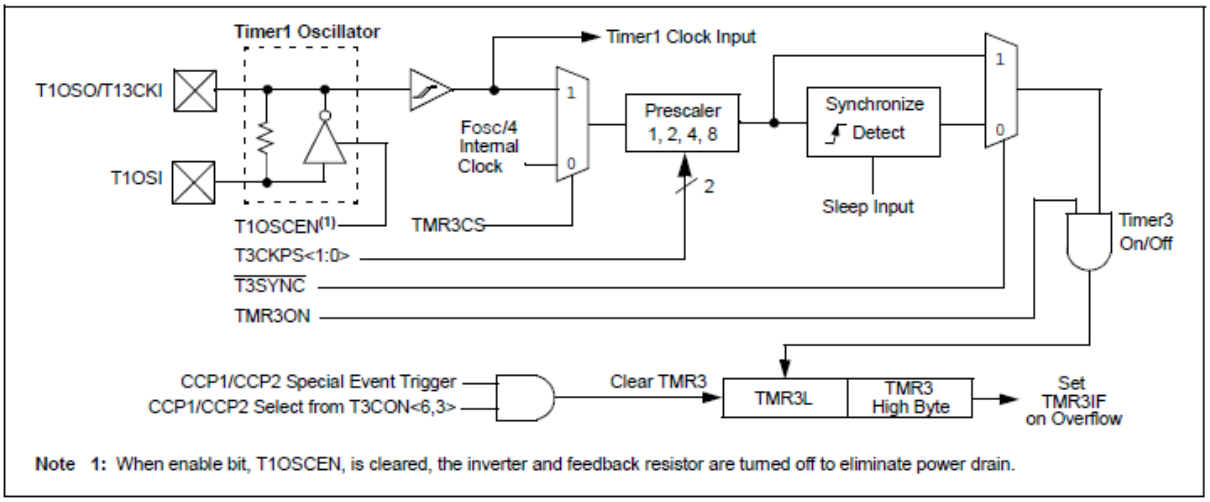


Figure II.22 Bloque diagramme du Timer3.[7]

II.23 Interruption

Une interruption est un arrêt temporaire de l'exécution normale du programme principal par le microprocesseur afin d'exécuté un autre programme appelé routine d'interruption ou programme d'interruption. Ce programme d'interruption va permettre de gérer l'événement qui a provoqué cette interruption.

Les interruptions sont, en général, contrôlées par trois bits :

- Un bit de flag : Indique qu'une interruption a été déclenchée et indique la source.
- Un bit de validation : permet à l'utilisateur d'activer ou une interruption.
- Un bit de priorité : permet de sélectionner la priorité (haute/basse) de l'interruption.

Le PIC 18F4520 dispose de 20 sources d'interruptions externes et internes, parmi ces sources :

- Débordement des Timers.
- Fin de la conversion A/N.
- Interruption externe sur INT0/RB0.
- Ecriture dans la mémoire EEPROM.
- Réception ou fin d'émission d'une information sur la liaison série.

Les bits d'interruptions sont accessibles dans les registres : RCON, INTCON, INTCON2, INTCON3, PIR1, PIR2, PIE1, PIE2, IPR1, IPR2. Le premier est le contrôle global des interruptions, son contenu est précisé dans le tableau suivant :

GIE	PEIE	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF

Registre INTCON

- RBIF : indique les changements d'état des lignes de RB7 s'il est mis à 1.
- INT0IF : indique un front montant sur l'entrée INT/RB0 (interruption externe).
- TMR0IF : indique un débordement duTimer0.
- RBIE : autorise l'interruption provoquée pas les changements d'états des lignes RB4 à RB7.
- INT0IE : autorise les interruptions provoquées par la ligne externe INT/RB0.
- TMR0IE : autorise les interruptions dues au débordement du Timer0.
- PEIE : autorise les interruptions périphériques.
- GIE : permet de valider toute les interruptions ; c'est une validation générale.

Il existe encore d'autres registres qui interviennent dans le contrôle des interruptions, il s'agit : des registres d'autorisation d'interruptions périphériques PIE1 et PIE2 et des registres des indicateurs d'évènements PIR et PIR2.

- **La séquence de déroulement d'une interruption**

Quand l'interruption se produit :

- Le microcontrôleur finit l'instruction en cours.
- Sauvegarde l'adresse suivante du programme principal.
- Sauvegarde le contexte
- Exécuté le programme lié au périphérique demandant l'interruption.
- Restitue le contexte.
- Retourne au programme principal à l'adresse suivante.

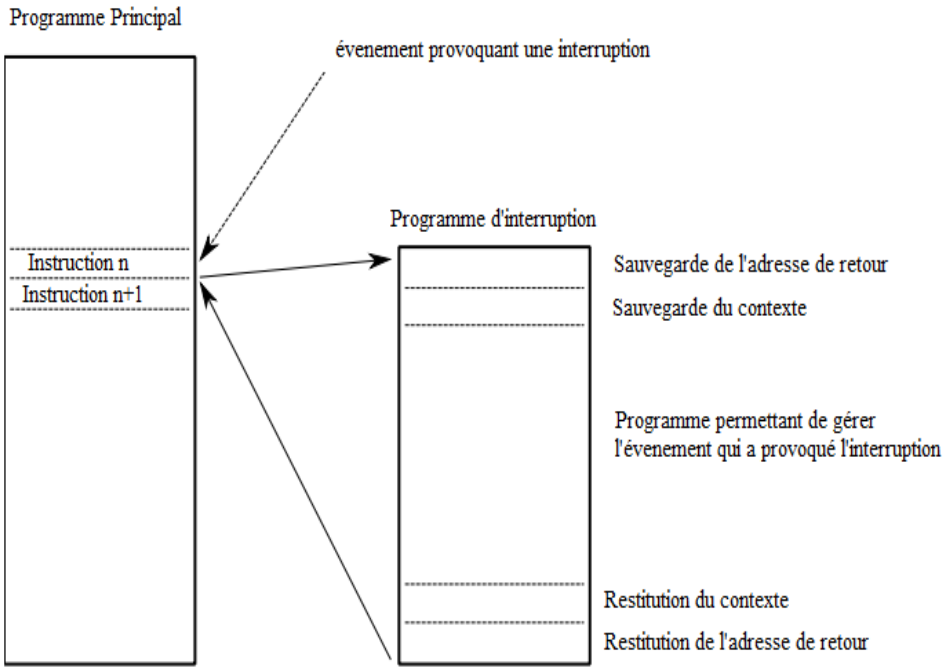


Figure II.23 Déroulement d'un programme lors d'une interruption.

II.24 Watch dog (chien de garde)

C'est un système de protection contre un blocage du programme. Par exemple, si le programme attend le résultat d'un système extérieur (conversion analogique numérique) et qu'il n'y a pas de réponse, il peut rester bloquer. Pour en sortir on utilise un chien de garde. Il s'agit d'un compteur qui, lorsqu'il arrive enfin de comptage, permet de redémarrer le programme. Il est lancé au début du programme. En fonctionnement normal, il est remis à

zéro régulièrement dans une branche du programme qui s'exécute régulièrement. Si le programme est bloqué, il ne passe plus dans la branche de remise à zéro et le comptage va jusqu'au bout, déclenche le chien de garde qui relance le programme.

II.25 Mode sommeil (mode SLEEP)

La fonction Sleep arrête l'activité du système et réduit sa consommation en énergie en arrêtant l'activité da la PLL. Dans ce mode, la consommation est très basse, permettant l'alimentation par batterie. Le fonctionnement normal peut être repris (sortie du mode sleep) s'y il a une interruption, un reset ou expiration du temps du watchdog.

II.26 Les périphériques

Les modules CCPM possède trois modes de fonctionnement :

II.26.1 Mode Capture

Le mode capture déclenche une action si un évènement prédéterminé apparait (ex : changement d'état sur une broche). Utilisé avec Timers, ce mode peut compter les temps d'arrivées.

II.26.2 Mode Compare

Le mode compare effectue une comparaison permanente entre le contenu d'un Timer et une valeur donnée pour déclencher une action si ces contenus sont égaux.

II.26.3 Mode PMW

Le mode PMW génère un signal rectangulaire de fréquence et de rapport cyclique choisis par l'utilisateur.

II.27 Les comparateurs

Les comparateurs permettent de comparer le signal analogique présent sur une broche du microcontrôleur à une valeur de référence.

II.28 Module de conversion analogique/numérique

Le module de conversion analogique/numérique permet de convertir le signal analogique présent sur une broche du microcontrôleur en un signal numérique. Ce module

dispose de 13 canaux d'entrées analogiques sur les pins AN0 À AN12 du port A. il permet un échantillonnage ou bien une résolution sur 10 bits.

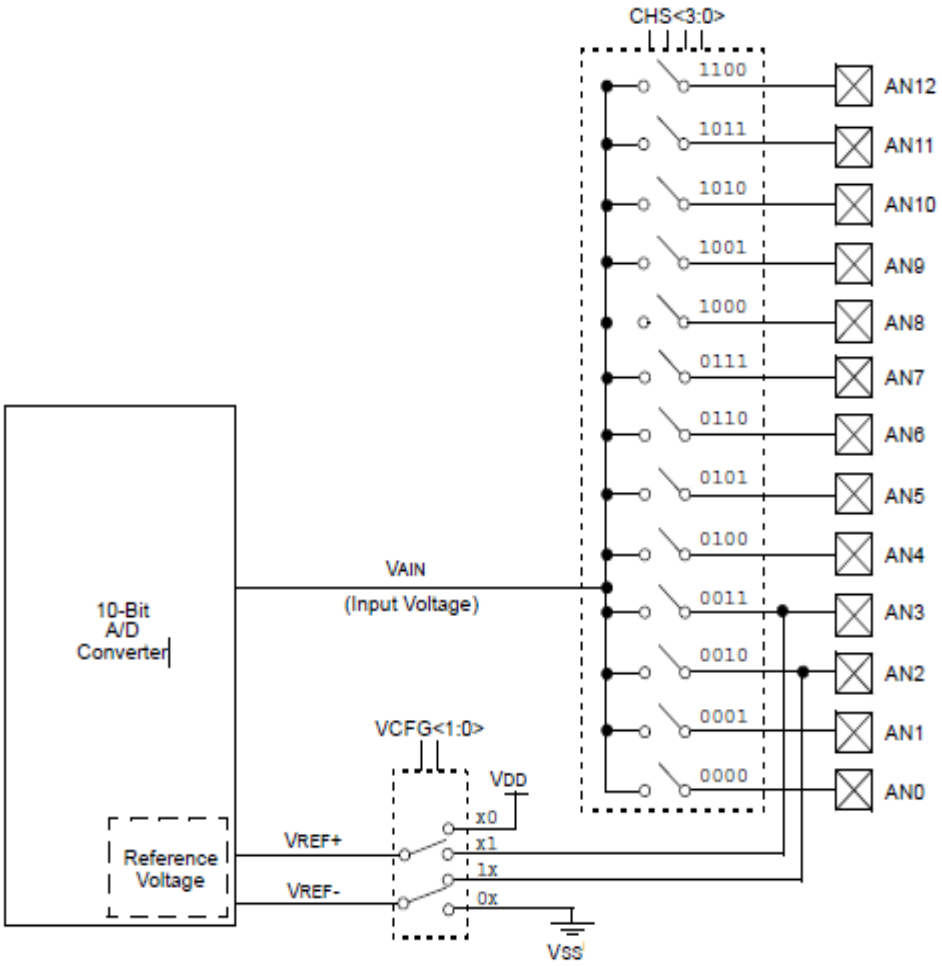


Figure II.24 Module de conversion analogique/numérique.[7]

II.29 L'interface série

Ce type d'interface permet au microcontrôleur de communiquer avec d'autres systèmes par liaison série où les bits d'informations sont envoyés ou reçus successivement, soit à des intervalles réguliers (transmission synchrone) soit à des intervalles aléatoires (transmission asynchrone).

Elle utilise les pins RC6/TX et RC7/RX du port C comme lignes de communication.

II.30 Conclusion

En conclusion dans ce chapitre nous pouvons dire que le microcontrôleur 18f4520 peut bien jouer le rôle d'une unité de contrôle pour notre système. Et après que nous avons fait l'étude de la carte de développement on peut dire que c'est un outil important pour le développement des microcontrôleurs, vu le nombre important de modules intégrés et prêt à l'emploi et la capacité de la mémoire du microcontrôleur la rend les utile dans le domaine de l'industrie. Maintenant, nous pouvons passer à la programmation et la compilation dans PIC C Compiler, la simulation du programme avec logiciel ISIS PROTEUS.

.

Chapitre III

Simulation des données de l'afficheur

III .1 Introduction

La programmation du PIC nécessite la maitrise du logiciel de programmation dans le cas présent c'est le «PIC C Compiler », alors dans ce chapitre on va présenter le logiciel, puis on va introduire le programme pour notre afficheur et la simulation de programme

III .2 Les systèmes de développement

Pour utiliser un microcontrôleur dans un système plusieurs étapes sont nécessaires :

- Le choix de microcontrôleur de la carte de développement. Il est déterminé par le nombre de ports nécessaires, les fonctions à réaliser et la vitesse souhaitée.
- L'écriture du logiciel aboutissant à une liste d'octets qui devront être implantés dans la ROM programme.
- Le test de ce logiciel dans des conditions aussi proches que possible de la réalité dans Les quelles travaillera le microcontrôleur.
- La gravure définitive de la PROM qui sera implantée sur la carte finale.

III.2.1 Base d'un système de développement

III.2.1.a Côté logiciel:

Un système de développement comporte en premier lieu un assembleur et parfois un ou des compilateurs adaptés au langage évolué que l'on souhaite utiliser pour programmer. L'assembleur traduit les instructions écrites en utilisant du langage machine en code binaire exécutable par le microcontrôleur.

Le compilateur quant à lui traduit les instructions écrites en langage évolué (BASIC, PIC C Compiler, PASCAL, MIKRO C…) qui constituent aussi ce que nous appelons le listing ou code source, en code binaire exécutable par le microcontrôleur qui constitue le code objet.

Dans un système de développement bien conçu, les deux programmes, assembleur et compilateur, peuvent coexister et être utilisables l'un et l'autre sans difficulté. Ces deux Programmes, assembleur et compilateur doivent nécessairement fonctionner sur une machine appelée machine hôte. Cette machine peut être un système spécifique du fabricant des microcontrôleurs, une station de travail, un calculateur ou un PC.

Une fois le programme de l'application écrit et assemblé ou compilé sur la machine hôte, nous sommes en possession d'un binaire exécutable.

III.2.1.b Côté matériel:

Pour implanter le binaire exécutable, nous avons besoin d'un programmateur (PIC C Compiler) et appareil est en fait un système qui va transférer de la machine hôte, le code objet du programme dans la mémoire du microcontrôleur.

III.2.2 Organisation du développement (matériel et programmation):

Pour cette solution, on utilise la carte du développement PICPLC6 v6 qui contient le microcontrôleur.

Pour utiliser la carte du développement et le logiciel associé, il faut passer par:

- Association des deux parties matérielle et logicielle.
- Utilisation d'un émulateur pour tester cet ensemble.

La figure.III.1 montre comment marche l'organisation du développement.

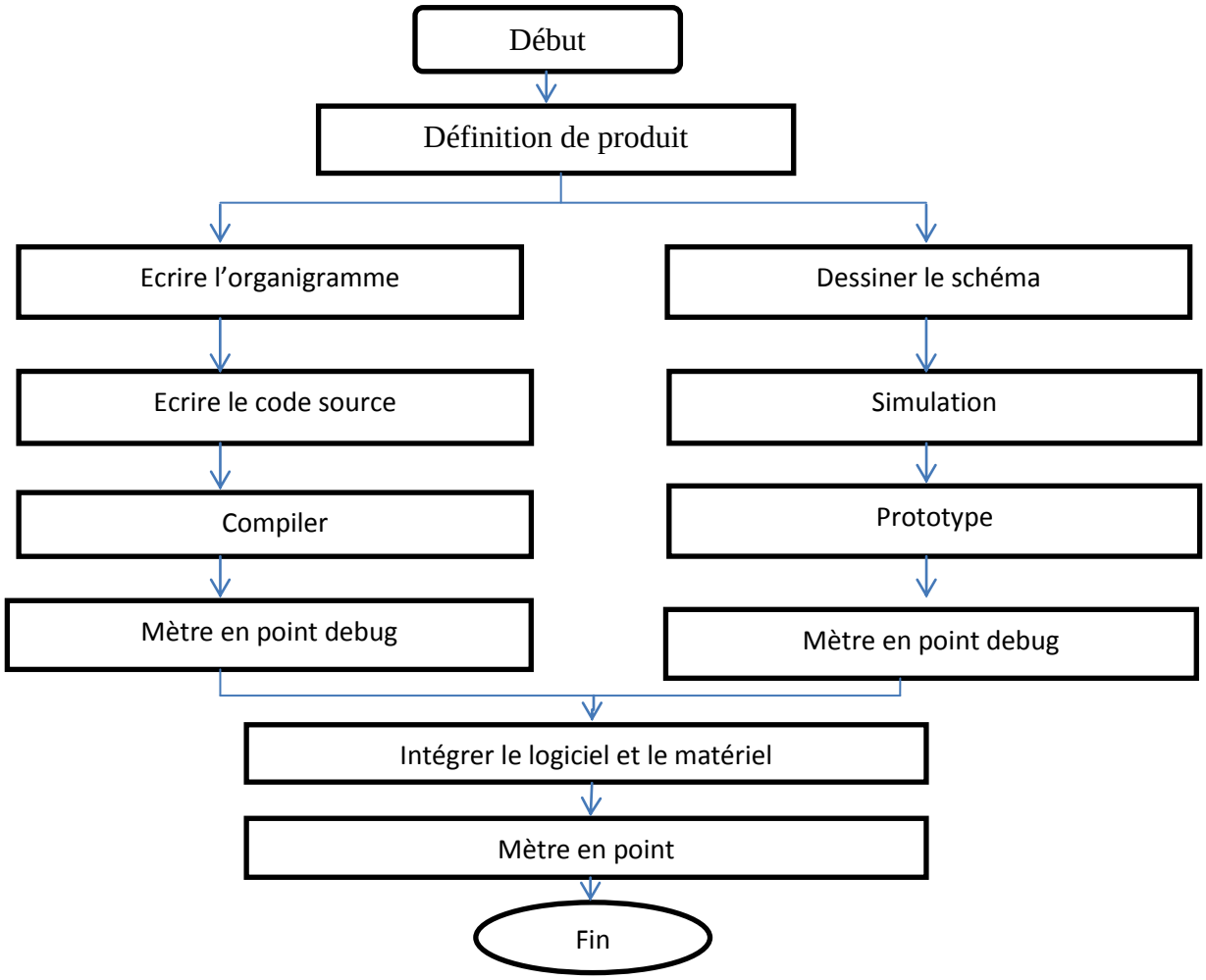


Figure III.1 Organisation du développement

On ne peut parler de microcontrôleur sans aborder les logiciels de programmation et les matériels permettant de développer le composant. Pour réaliser les programmes exécutables, on utilisera l'assembleur ou le compilateur PIC C ou le Mikro C …etc.

La figure III.2 montre comment marche l'organisation du développement coté programmation.

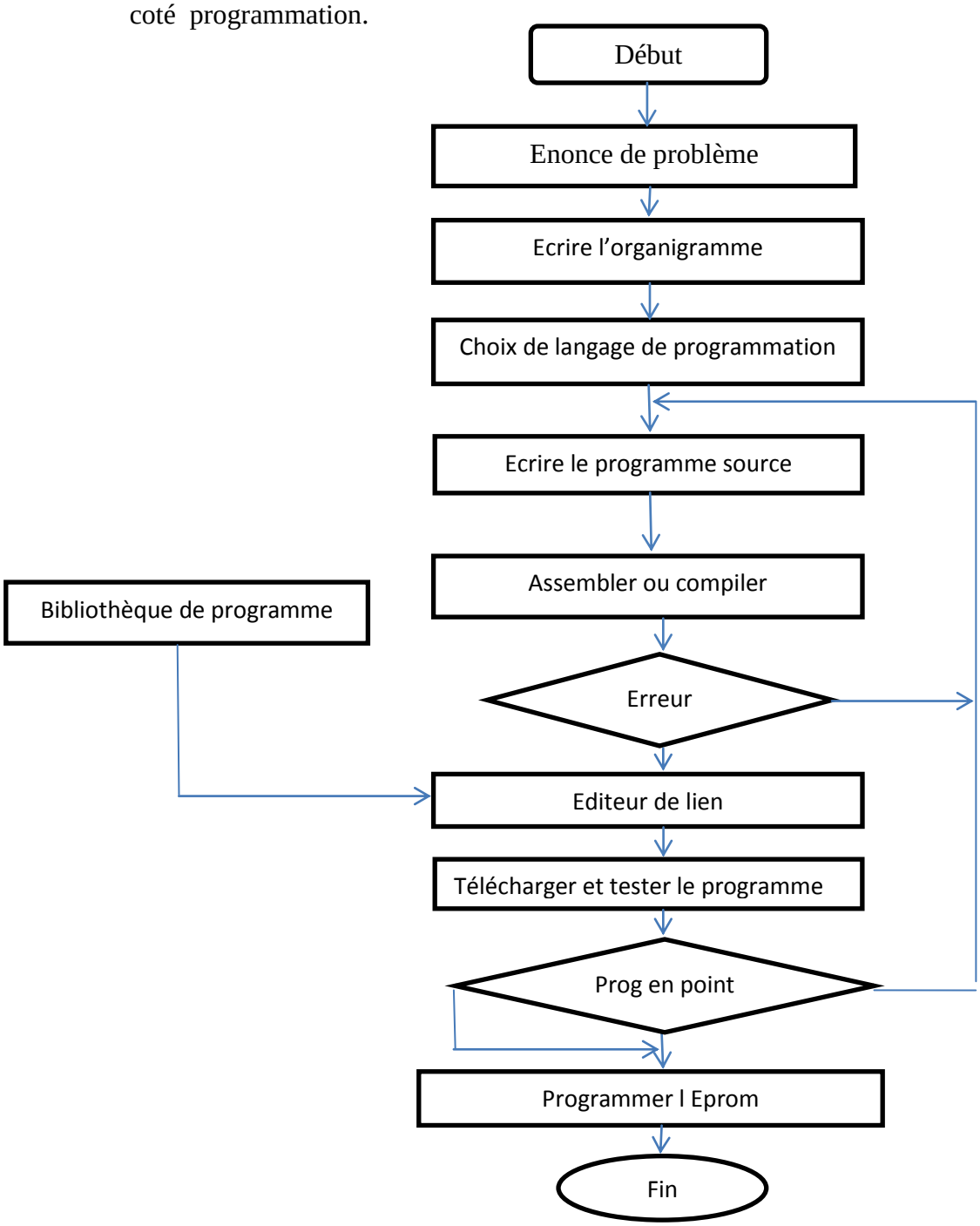


Figure III.2 Organisation du développement coté programmation

III.3 Présentation du logiciel

Le PIC C Compiler pour PIC est un plein de fonctionnalités compilateur C ainsi pour les appareils PIC de MICROCHIP. Il est la meilleure solution pour le développement du code pour les appareils PIC.

Nous allons présenter dans les pages suivantes l'utilisation de l'environnement intégré du développement en langage C pour PIC de l'éditeur CCS. La référence complète du langage C pour PIC est disponible sur le réseau dans le répertoire « P:\Docs PIC ». Une aide en ligne est aussi intégrée au logiciel. Pour lancer PIC C Compiler, cliquez sur l'icône 

« PIC C Compiler ». Au lancement, vous obtenez la fenêtre suivante :

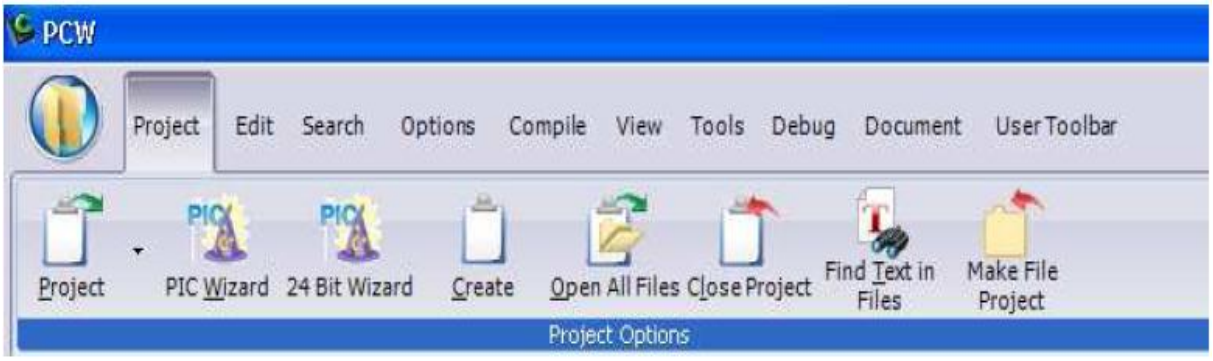


Figure III.3 Création d'un nouveau projet

Vous remarquez que ce logiciel fonctionne avec un système d'onglet permettant d'accéder aux différentes fonctions de l'environnement de développement.

Cliquez sur la première icône qui représente un dossier ouvert. Puis choisissez « New » puis « Source File ». Vous obtiendrez une fenêtre qui vous demande d'enregistrer un nouveau fichier. Vérifiez que vous êtes bien dans votre répertoire personnel puis tapez le nom de votre programme dans la zone « Nom du fichier » (par ex: test .c) puis validez avec le bouton « Enregistrer ».

Vous obtenez alors la fenêtre suivante :

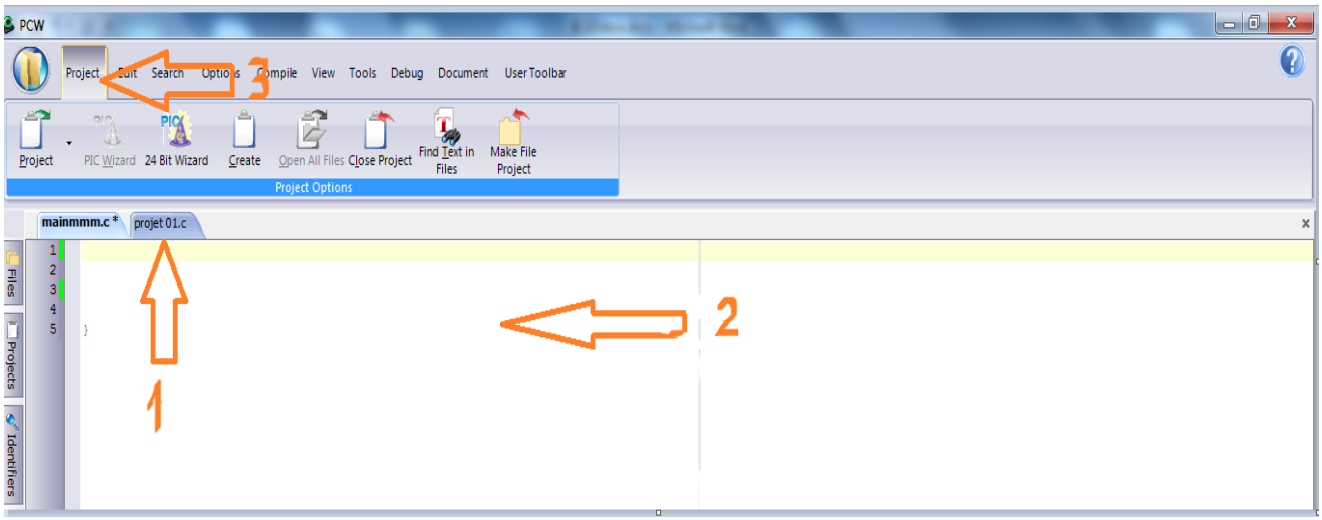


Figure III.4 Création d'un projet avec succès

- (1) Vérifiez que le nom de votre fichier source apparaît bien (ici : projet .c).
- (2) Vous pouvez alors commencer à taper votre code source C dans la zone d'édition. Vous remarquez que les lignes sont numérotées automatiquement.
- (3) A la fin de la saisie, sauvegardez votre fichier en cliquant sur « Save ».

Il faut ensuite compiler votre programme, c'est-à-dire une suite d'instructions compréhensible par le µC PIC : convertir le fichier texte en langage C.

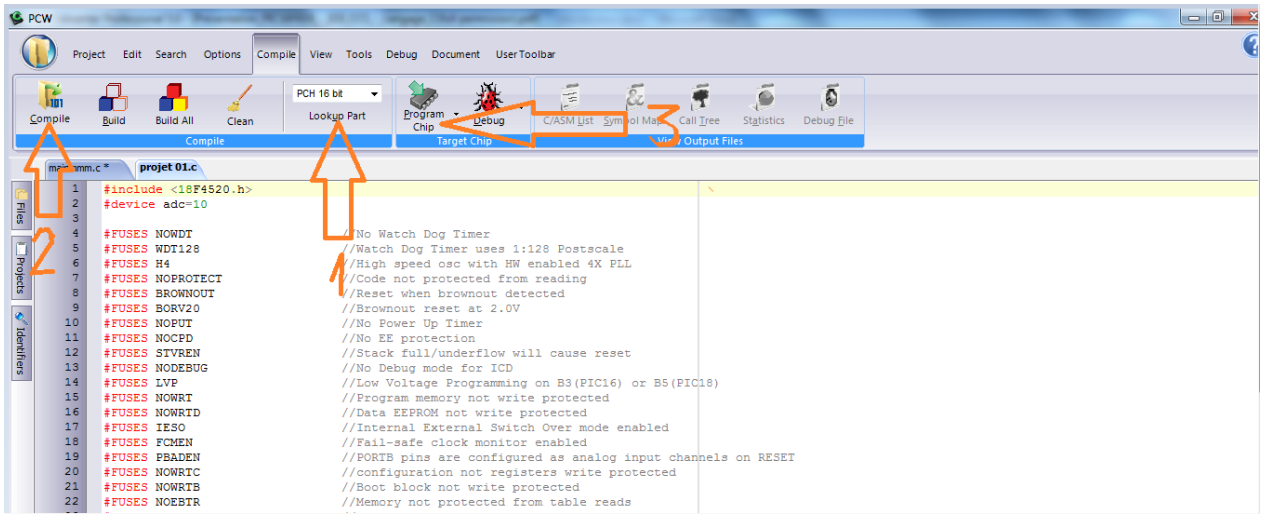


Figure III.5 Ecriture de programme

Vous cliquez alors sur l'onglet « Compile » puis :

- (1) Vous vérifiez que vous compilez votre code source pour un PIC 18F.

- (2) Vous lancez la compilation en cliquant sur l'icône « Compile »,
- (3) Si la compilation s'est déroulée correctement (sans erreur donc), vous pouvez programmer votre PIC en cliquant sur l'icône « Program Chip ». et en sélectionnant « ICD ».

Si la compilation de votre code source se passe correctement vous obtiendrez alors la fenêtre ci-dessous :

- (1) Ce rectangle indique le nombre d'erreurs (errors) et d'avertissements (warnings)
- (2) Ce rectangle rappelle le nom de votre fichier et son emplacement.

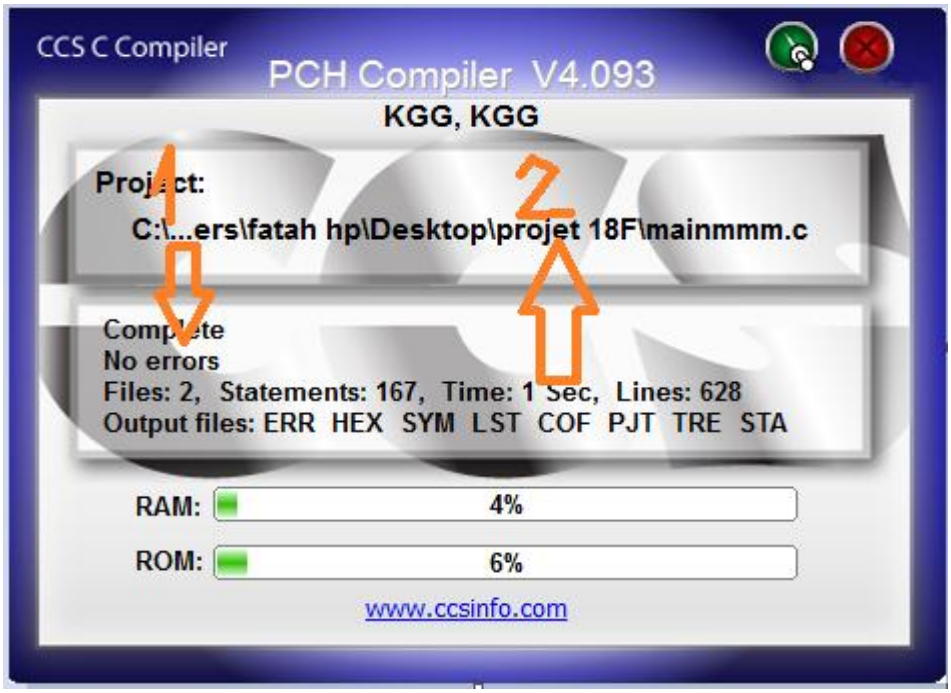


Figure III.6 La compilation de programme

Dans le cas contraire vous obtiendrez la fenêtre suivante (figure III.7) avec un message d'erreur qui apparaitra en bas de la fenêtre (flèche 1), la ligne contenant l'erreur apparaitra d'elle-même (flèche 2):

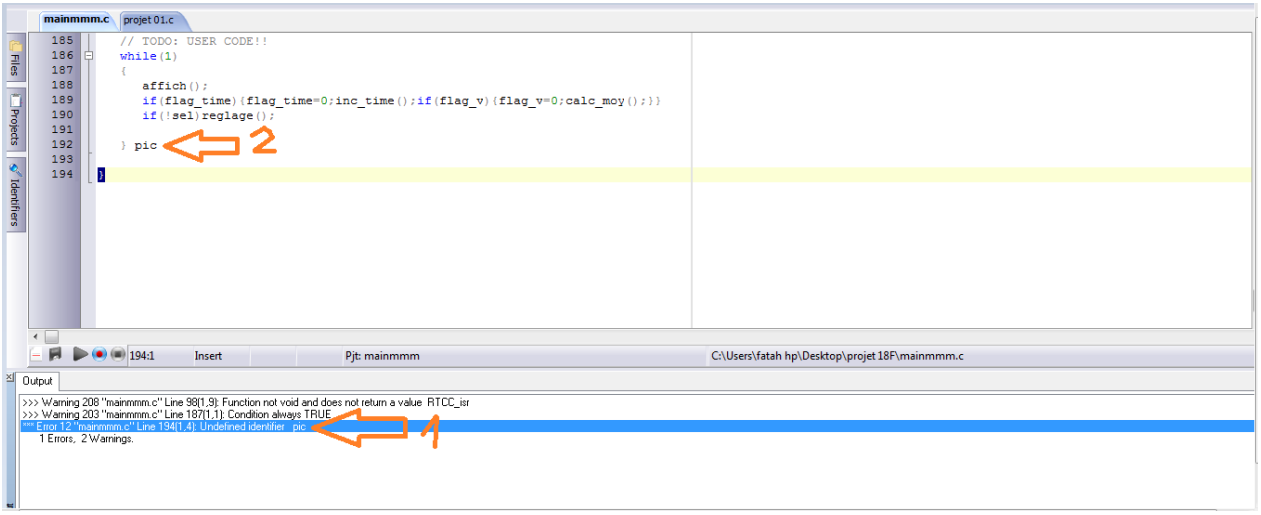


Figure III.7 Démonstration de l'erreur

La programmation dans le PIC C Compiler

Les étapes nécessaires pour avoir un programme qui s'exécute sur un PIC sont :

L'écriture de programme en langage C et le sauvegarder avec l'extension .C

Apres l'écriture du programme on passe à la compilation la figure 5 montre le succès de la compilation et un fichier d'extension (.hex) est généré à partir du code source. C'est ce dernier fichier qu'on va injecter au microcontrôleur.

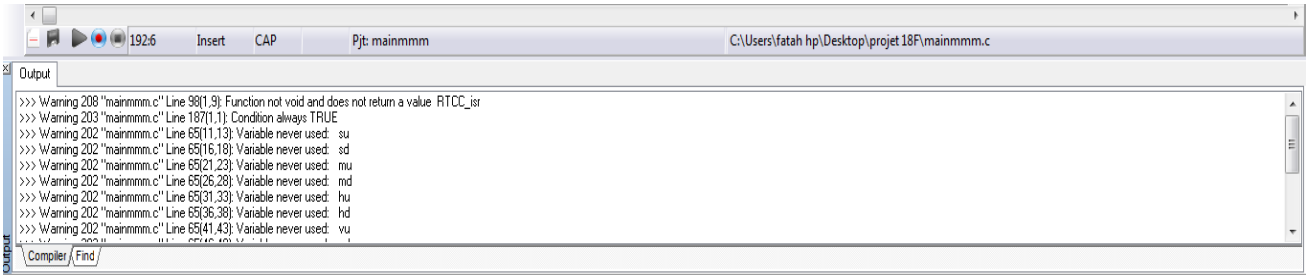


Figure III.8 Compilation du programme.

Le programme de commande va afficher l'horloge, la vitesse de la machine en temps réel et la vitesse moyenne. On transfère le programme au microcontrôleur 18F4520 par la carte de développement PICPLC16 v6.

.

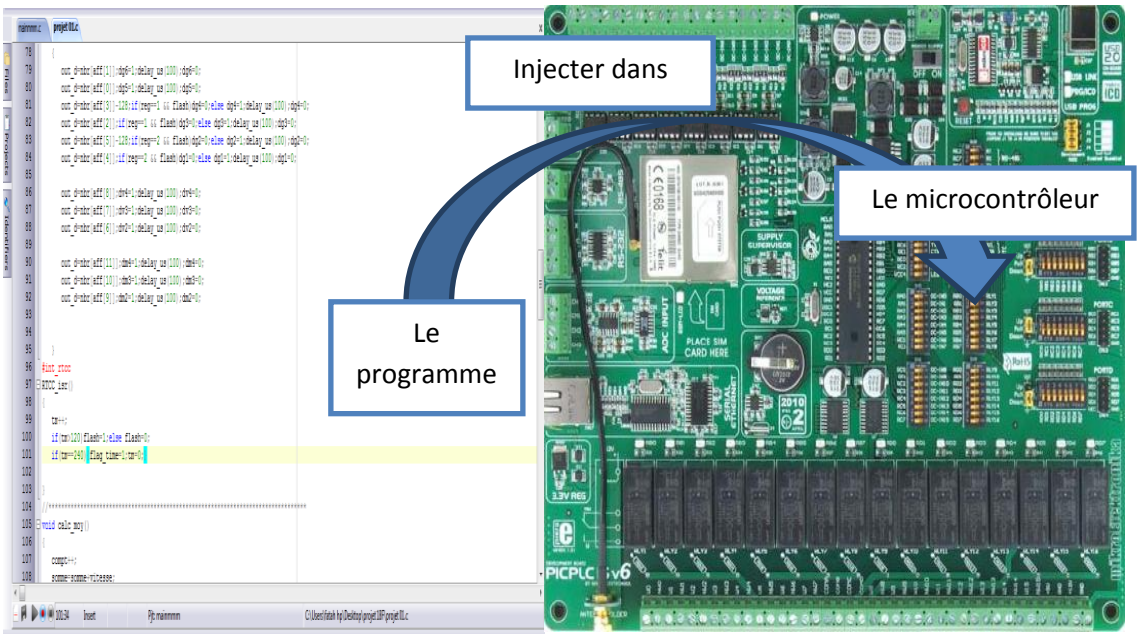


Figure III.9 Injection du programme dans le microcontrôleur

Le programme doit pouvoir être lancé à partir d'un ordre venant du PC. Pour rendre le programme fiable, nous implémenterons un mécanisme qui s'assurera du bon fonctionnement du logiciel de contrôle, situé sur le PC. Toutes les 50 millisecondes, une commande est envoyée au microcontrôleur pour lui signaler que tout fonctionne bien. Si le PC plante, la commande n'est plus envoyée et au bout de 200 millisecondes le microcontrôleur 18F4520 s'initialise et s'arrête. Pour chaque commande reçue, le microcontrôleur réagit en interrogeant les périphériques et en les commandons.

III.4 Conclusion

Dans ce chapitre nous avons fait une description, puis on a établis un programme selon le cahier de charge après avoir dressé un logigramme général de fonctionnement de notre système.

1233

Chapitre IV

Conception Virtual

IV.1 Introduction

La réalité virtuelle est un domaine en plein progrès. Elle offre de nouveaux outils pour de nombreuses applications dans des divers domaines, comme par exemple la conception avec le ISIS PROTEUS. L'avantage décisif de cette technologie est d'offrir un bon degré de contrôle sur la simulation ainsi la planification de tous les évènements simulés, ce qui est d'habitude impossible dans la réalité.

Ce chapitre est consacré aux conceptions virtuelles de notre afficheur et la présentation de l'afficheur.

IV.2 PROTEUS

PROTEUS est une suite logicielle permettant la conception assistée par ordinateur éditée par la société l'ABSENTER ELECTRONIQUES. Il est compose de deux logiciels principaux :

- ISIS**,** permettant entre autres la création de schémas et la simulation électrique,
- ARES dédié à la création de circuit imprimés

Grace à des modules additionnels, ISIS est également capable de simuler le comportement d'un microcontrôleur (PIC, atmel, 8051, ARM, HC11……) et son interaction avec les composants qui l'entourent. C'est ce dernier atout qui nous a convaincu de le choisir pour concevoir notre projet.

IV.2.1 ISIS

Le logiciel ISIS de PROTEUS 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 des étapes de conception .Indirectement les circuits électrique conçus grâces a ce logiciel peuvent être utilise dans des documentations car le logiciel permet de contrôler la majorité de l'aspect graphique des circuits.

IV.2.2 ARES

Le logiciel ARES est un outil d'édition et de routage qui complimente parfaitement ISIS. Un schéma électrique réalisé sur ISIS peut alors être importé facilement sur ARES pour réaliser le circuit imprimé.

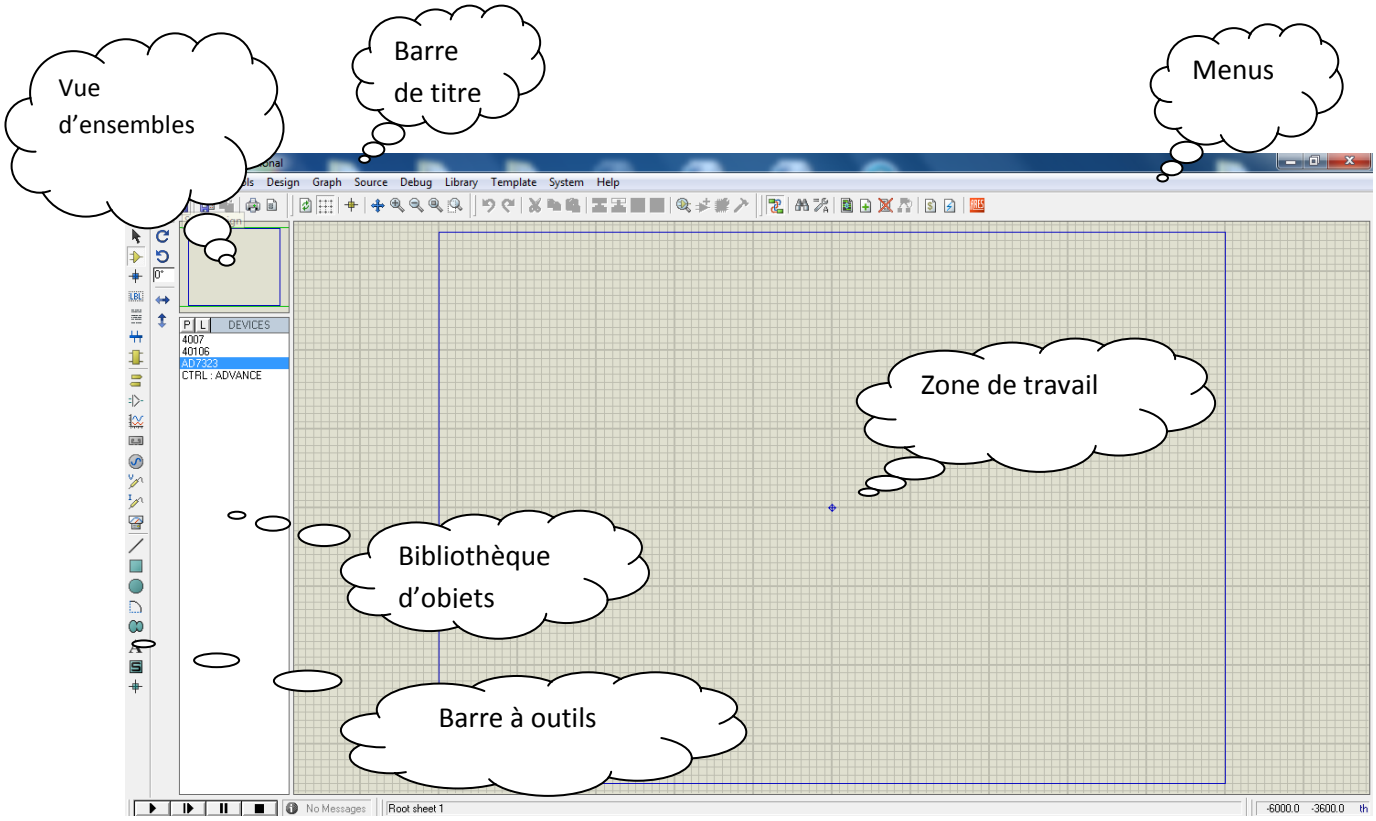


Figure IV.1 Fenêtre principale de travail sur ISIS

IV. 3 Sélection des composants à utiliser

Pour faire la sélection des éléments qu'on veut utiliser

Un clique sur l'icône (component Mode) puis sur bouton parcourir la bibliothèque

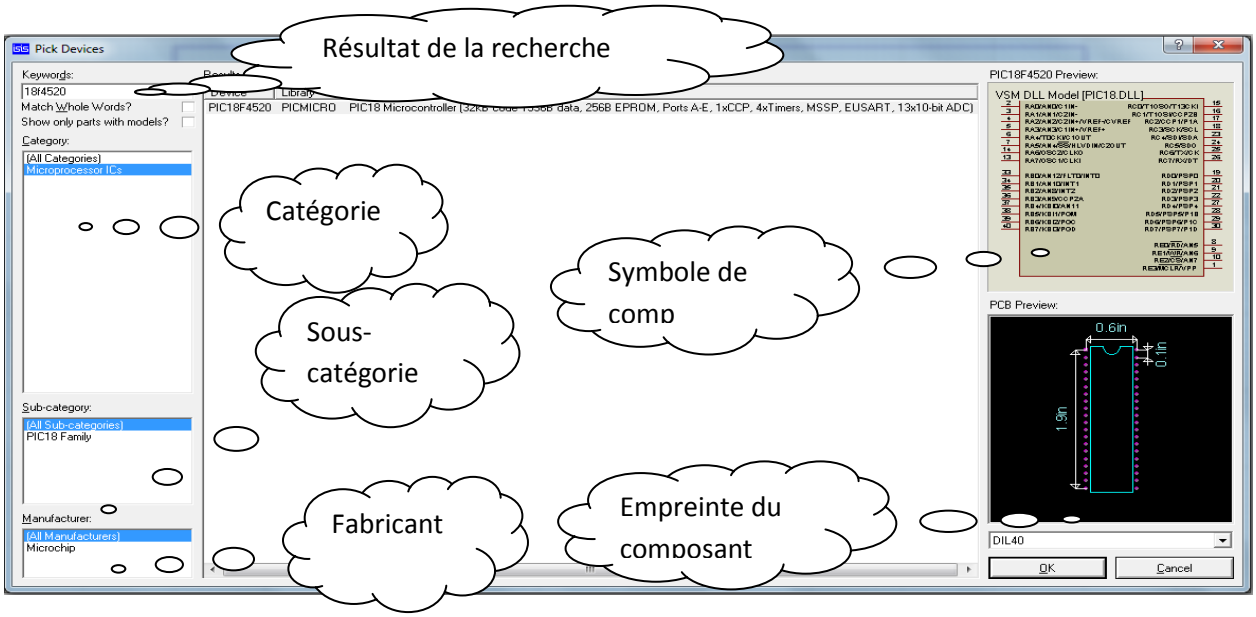


Figure IV.2 Bibliothèque ISIS

IV.4 Les éléments constituants l'afficheur

IV.4.1. Microcontrôleur

C'est le composant principal de l'afficheur, on a opté pour le pic 18F4520 illustré en figure IV .3, car il nous convient sur tous les plans. Il est chargé d'acquérir les données, les afficher puis les transmettre via le port RS232. Les instructions de son programme sont contenues dans sa mémoire morte ROM, pour injecter le programme dans sa ROM, il suffit de parcourir le code sources sous l'extension (.hex) dans les propriétés du microcontrôleur, catégorie programme file. Ce dernier fichier est généré par l'environnement PIC C Compiler. Le PIC saura alors gérer la suite lors de la simulation.

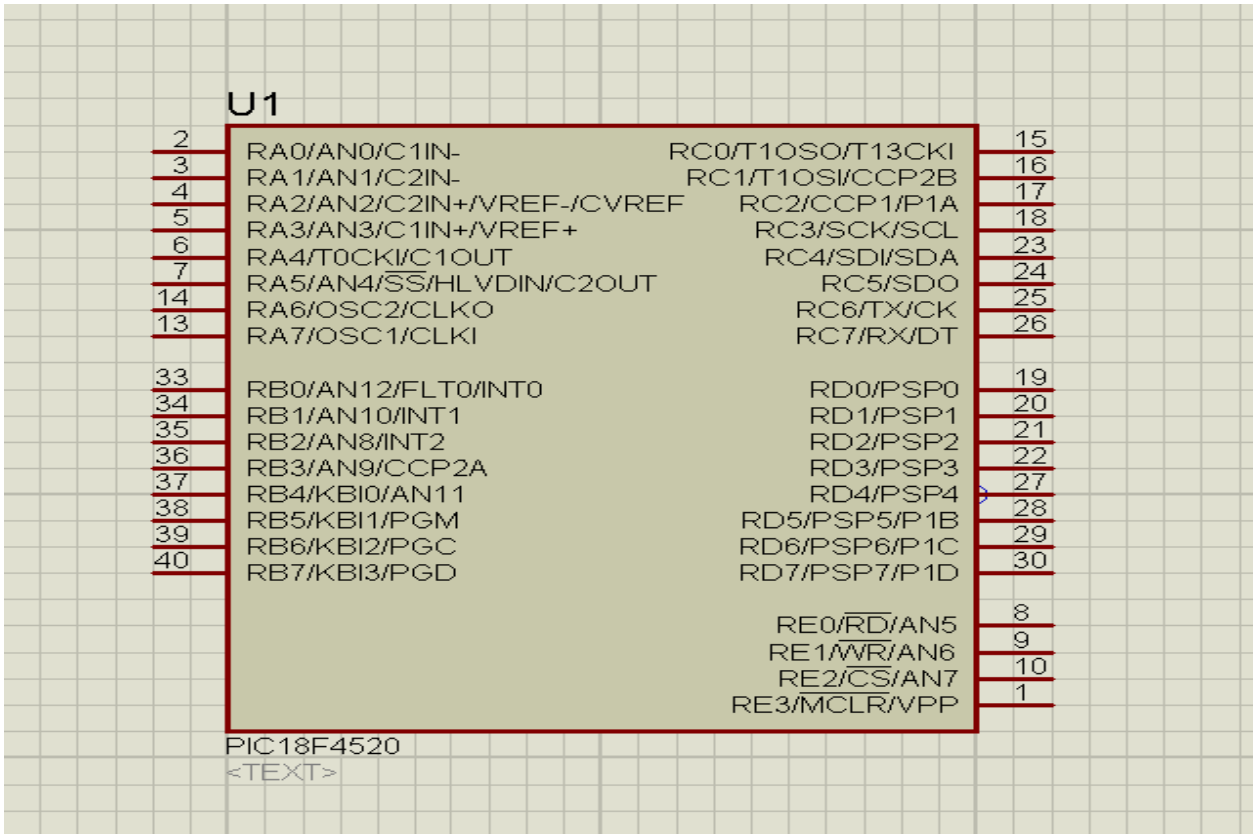


Figure IV.3 Image du PIC18f4520 sous ISIS

IV.4.2 L'affichage de la vitesse

Les capteurs de détections de la vitesse n'existe pas dans la bibliothèque de ISIS, c'est pour cela qu'on était contraint d'utiliser des résistances variables (voire la figure IV.4), elles nous permettent d'exploiter la tension délivrer a leurs bornes comme sortie d'un capteur réel.

Une adaptation doit être faite au niveau du programme du microcontrôleur pour avoir la plage de mesure souhaite.

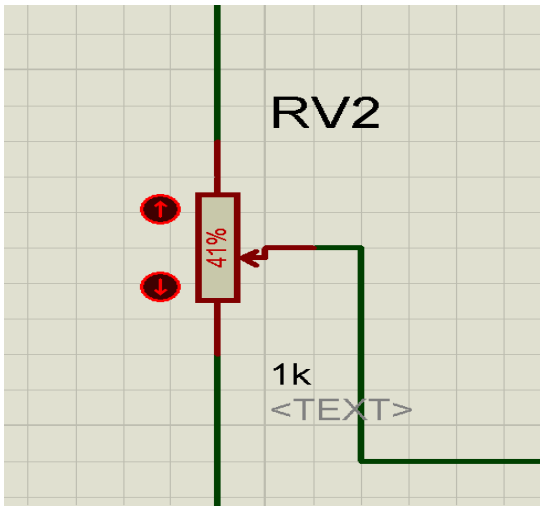


Figure IV.4 Résistance variable sous ISIS

IV.4.3 Un afficheur 7 segment

La fonction affichage permet à l'utilisateur d'un système ou d'un appareil, de lire, dans un code usuel, des grandeurs caractéristiques du fonctionnement de l'appareil.

Chaque segment est éclairé par une LED. Un afficheur 7 segments possède 8 LED, une de plus pour l'affichage du point.

Il existe trois types d'afficheurs :

- A cathode commune, avec une polarisation à 5V des entrées anode.
- A anode commune, avec une polarisation à 0V des entrées cathode.
- Universel.

On distingue les deux types d'afficheurs suivant la position du point décimal et du brochage du boîtier de l'afficheur.

IV.5 Simulation par ISIS

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

Il faut toujours prendre en considération que les résultats obtenus de la simulation sont un peu différents de ceux du monde réel, et ça dépend de la précision des modèles, des composants et de la complication des montages.

IV.6 Simulation numérique :

Il faut bien évidemment commencer par la saisie du schéma qu'on souhaite réaliser : Pour saisir le schéma, il faut créer un nouveau projet puis placer les composants qui doivent être sélectionné à partir de la bibliothèque des composants :

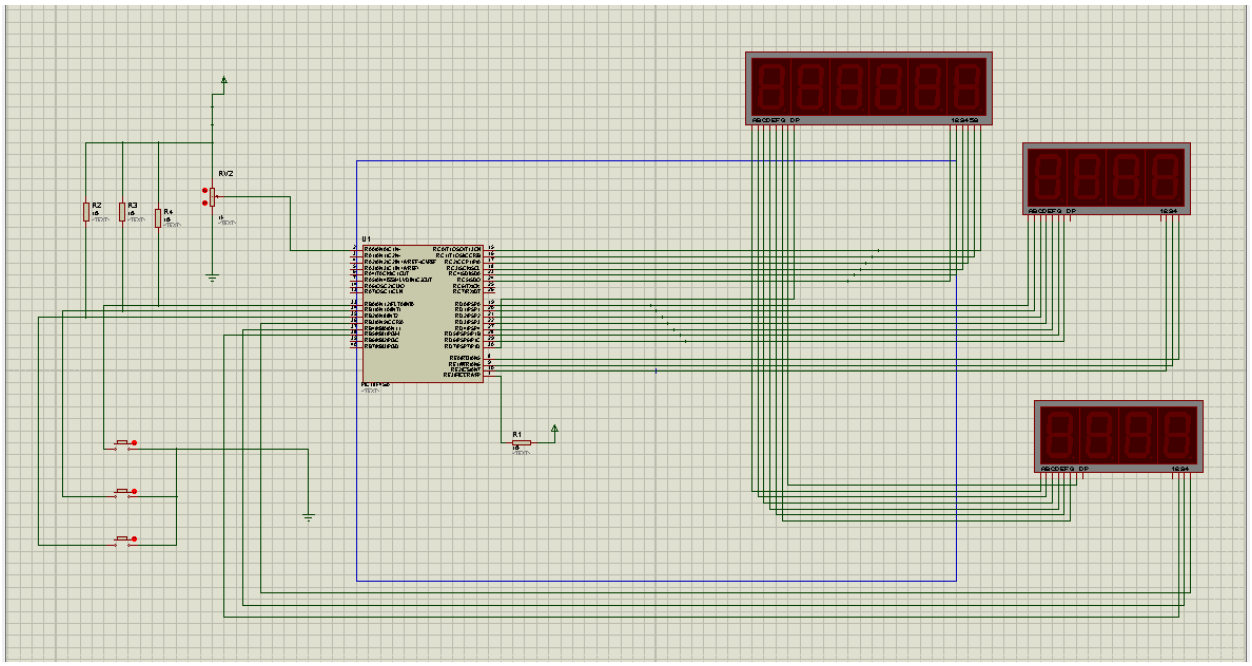


Figure IV.5 Montage de l'afficheur sous ISIS

Dans le menu « source », sélectionnez la commande « Ajout/suppression fichiers source » puis sélectionnez le fichier ASM du programme à utiliser dans la simulation. Puis choisissez « Outil de génération de code» pour générer un fichier HEX. Éditez le PIC18F4520 et ajoutez le fichier HEX dans le champ « programme file ».Puis on lance la simulation.

Appuyez sur le bouton « play»  . La barre des messages doit indiquer le temps écoulé depuis le lancement de l'animation.

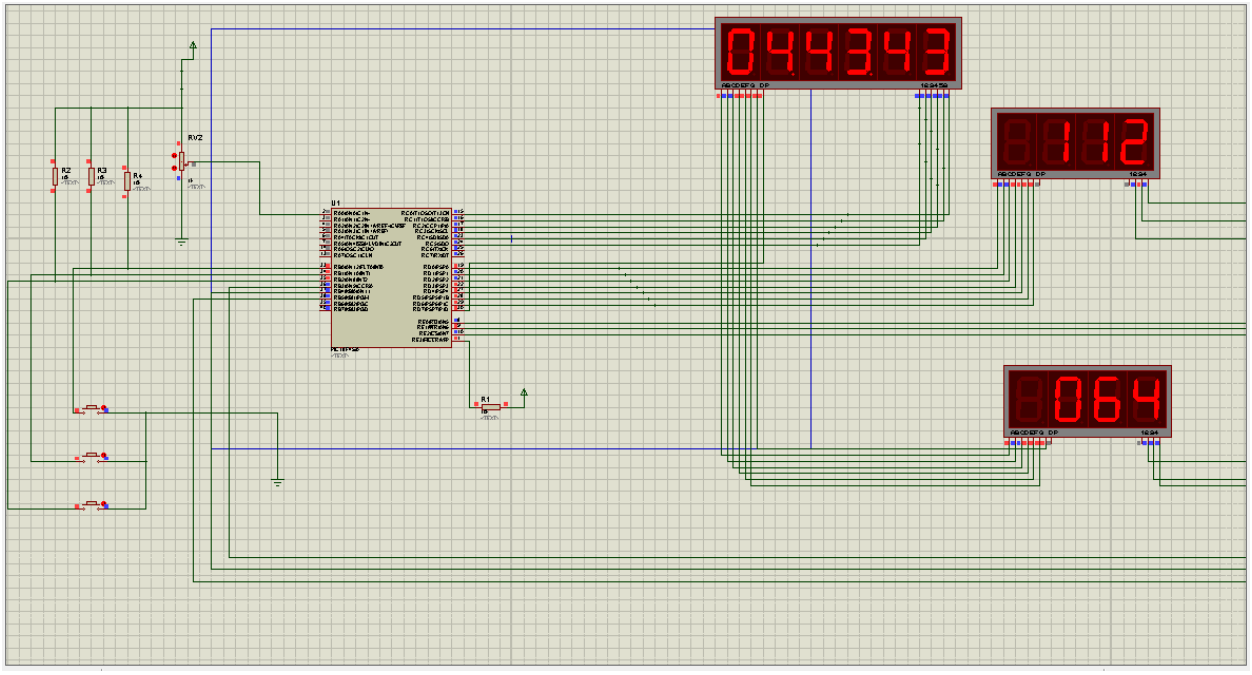


Figure IV.6 Montage de l'afficheur lors de la simulation

IV.7 Conclusion

Dans ce chapitre on a pu faire le schéma d'implantation dans le logiciel ISIS de tout les composants électriques de l'afficheur, où on a pu voir le comportement de l'afficheur lors de la simulation de l'horloge et la vitesse, la vitesse moyenne et tout cela piloté par le microcontrôleur comme s'il s'agissait d'un modèle réel. La simulation donnait des résultats satisfaisants.

Conclusion générale

Conclusion générale

L'objectif de notre travail était d'établir un programme pour un afficheur des données à base de microcontrôleur PIC 18F4520 et l'étude de la carte PICPLC16 V6.

L'élaboration de ce projet nous a apporté énormément de connaissances dans le domaine de l'électronique et de la programmation sur les microcontrôleurs.

Afin de mettre au clair le fonctionnement de notre système, nous avons dressé des organigrammes qui représentent le déroulement de toutes les étapes du processus. Le programme est réalisé à l'aide du logiciel PIC C Compiler avec le langage C et après la simulation les résultats obtenus sont satisfaisants.

Nous avons effectué des simulations du circuit avec logiciel ISIS, ce qui nous a aidé à étalonner et tester le conditionnement ainsi que la mise en œuvre du programme de gestion du microcontrôleur.

Ces trois mois de stage passé à SARL général emballage nous ont apporté beaucoup plus que de connaissances techniques. Nous n'avons jamais eu l'occasion de mettre à profit nos connaissances dans le monde professionnel. On a également remarque qu'avoir de bonnes connaissances techniques n'est pas suffisant pour la réalisation d'un projet. Cette expérience nous a été bénéfique sur plusieurs plans, elle nous a permis de nous familiariser avec les PICs et nous initier au logiciel de programmation.

On a constaté que la qualité de l'enseignement nous a permis de nous adapter rapidement au milieu professionnel et aux différentes méthodes de travail.

Des améliorations peuvent être apportées à ce travail, comme :

- L'affichage de la date et les jours de la semaine.
- L'affichage de la température.
- Réglage de la date avec une télécommande.
- Exploiter d'autre type de protocole de communication entre l'afficheur et la salle contrôle de la machine (wifi par exemple).

Dans tout les systèmes de productions automatisés le bon choix de l'unité est fondamentale vue son rôle très important dans la gestion des entrées / sorties du processus industriel.

Références bibliographique

Bibliographie

[1] D.MERAT ET J.C.BOSSY {AUTOMATISME APPLEQUE}, Edition ANDRE

CASTEILLA, paris 1985

[2] A.SIMON, {AUTOMATES PROGRAMMABLES programmation automatisme et logique programmée} Edition l'ELAN 1983

[3] C.TRVENIER {Les microcontrôleurs PIC, description et mise en œuvre} Edition DUNOD, pris 2002, Edition 2004 et Edition 2005

[4] BIGONOFF {LA PROGRAMMTION DES PICs}

[5] M. KACIMI Mohand Akli et M. KACI Khaled {le microcontrôleur PIC 16f877 étude et application}, université de Bejaia, promotion 2008

[6] https://www.techniques-ingenieur.fr

[7] www.Microchip.com

Annexe

Résumé

L’objectif principal de ce travail est la réalisation d’un afficheur des données au sien de l’entreprise Général Emballage. Pour atteindre notre objectif on a opté pour l’utilisation d’une carte du développement PICPLC16 v6 a base du PIC18F4520.

Le programme réalisé à l’aide du logiciel PIC C Compiler avec le langage C, la simulation dans l’Isis Proteus confirme le bon fonctionnement du programme.

annexe

```
#include <18F4520.h>
#device adc=10
#FUSES NOWDT                 //No Watch Dog Timer
#FUSES WDT128                //Watch Dog Timer uses 1:128 Postscale
#FUSES H4                   //High speed osc with HW enabled 4X PLL
#FUSES NOPROTECT             //Code not protected from reading
#FUSES BROWNOUT              //Reset when brownout detected
#FUSES BORV20                //Brownout reset at 2.0V
#FUSES NOPUT                 //No Power Up Timer
#FUSES NOCPD                 //No EE protection
#FUSES STVREN                //Stack full/underflow will cause reset
#FUSES NODEBUG               //No Debug mode for ICD
#FUSES LVP                  //Low Voltage Programming on B3(PIC16) or B5(PIC18)
#FUSES NOWRT                 //Program memory not write protected
#FUSES NOWRTD                //Data EEPROM not write protected
#FUSES IESO                 //Internal External Switch Over mode enabled
#FUSES FCMEN                 //Fail-safe clock monitor enabled
#FUSES PBADEN                //PORTB pins are configured as analog input channels on RESET
#FUSES NOWRTC                //configuration not registers write protected
#FUSES NOWRTB                //Boot block not write protected
#FUSES NOEBTR                //Memory not protected from table reads
#FUSES NOEBTRB               //Boot block not protected from table reads
#FUSES NOCPB                 //No Boot Block code protection
#FUSES LPT1OSC               //Timer1 configured for low-power operation
#FUSES MCLR                 //Master Clear pin enabled
#FUSES NOXINST               //Extended set extension and Indexed Addressing mode disabled (Legacy
mode)
```

```
#use delay(clock=40000000)
#byte out_b= 0xf81
#byte out_c= 0xf82
#byte out_d= 0xf83
#byte out_e= 0xf84
#bit dg1=out_c.5
#bit dg2=out_c.4
#bit dg3=out_c.3
#bit dg4=out_c.2
#bit dg5=out_c.1
#bit dg6=out_c.0
#bit dv2=out_e.2
#bit dv3=out_e.1
#bit dv4=out_e.0
#bit dm2=out_b.5
#bit dm3=out_b.4
#bit dm4=out_b.3
#bit up=out_b.0
#bit sel=out_b.1
#bit dn=out_b.2
#bit pnt=out_d.7
#define hr_transit1  5    // 1ere heure d'initalisation
#define hr_transit2  13   // 2eme heure d'initalisation
#define hr_transit3  21   // 3eme heure d'initalisation
const char nbr[]={0xc0,0xf9,0xa4,0xb0,0x99,0x92,0x82,0xf8,0x80,0x90};
int aff[12];
```

```
int reg=0,i_ch,sec=0,min=0,hor=0;
int16 vitesse,vit_moy,tm,compt;
int32 somme;
int1 flag_transit=0,flash,flag_time,flag_v=0;

//*****************************************************************************
void charger(int ch)
{
  aff[i_ch]=ch-48;
  i_ch++;
}
//*****************************************************************************
void affich()
{
    out_d=nbr[aff[1]];dg6=1;delay_us(100);dg6=0;
    out_d=nbr[aff[0]];dg5=1;delay_us(100);dg5=0;
    out_d=nbr[aff[3]]-128;if(reg==1 && flash)dg4=0;else dg4=1;delay_us(100);dg4=0;
    out_d=nbr[aff[2]];if(reg==1 && flash)dg3=0;else dg3=1;delay_us(100);dg3=0;
    out_d=nbr[aff[5]]-128;if(reg==2 && flash)dg2=0;else dg2=1;delay_us(100);dg2=0;
    out_d=nbr[aff[4]];if(reg==2 && flash)dg1=0;else dg1=1;delay_us(100);dg1=0;
    out_d=nbr[aff[8]];dv4=1;delay_us(100);dv4=0;
    out_d=nbr[aff[7]];dv3=1;delay_us(100);dv3=0;
    out_d=nbr[aff[6]];dv2=1;delay_us(100);dv2=0;
    out_d=nbr[aff[11]];dm4=1;delay_us(100);dm4=0;
    out_d=nbr[aff[10]];dm3=1;delay_us(100);dm3=0;
    out_d=nbr[aff[9]];dm2=1;delay_us(100);dm2=0;
}
```

```
//*****************************************************************************
#int_rtcc
RTCC_isr()
{
  tm++;
  if(tm>120)flash=1;else flash=0;
  if(tm==50){flag_time=1;tm=0;}
}
//*****************************************************************************
void calc_moy()
{
  compt++;
  somme=somme+vitesse;
  vit_moy=somme/compt;
}
//*****************************************************************************
void load()
{
  i_ch=0;
  printf(charger,"%02u%02u%02u%03lu%03lu",sec,min,hor,vitesse,vit_moy);
}
//*****************************************************************************
void inc_time()
{
  sec++;if(sec>59)
  {
    sec=0;flag_v=1;min++;
```

```
    if(min>59)
      {
        min=0;hor++;
        if(hor>23)hor=0;
        if(hor==hr_transit1 || hor==hr_transit2 || hor==hr_transit3)flag_transit=1; // activer le flag
d'initialisation
      }
  }
  vitesse=(read_adc()+1) * 13 /38;
  load();
}
//*****************************************************************************
void init()
{
 sec=0;min=0;hor=0;
  vitesse=0;
  vit_moy=0;
  compt=0;
  somme=0;
}
//*****************************************************************************
void reglage()
{
  reg=1;
  while(!sel)affich();
  while(reg)
  {
    if(!sel || !up || !dn)
```

```
    {
      if(!sel){reg++;if(reg>2)reg=0;}
      if(reg==1 && !up){min++;if(min>59)min=0;sec=0;load();}
      if(reg==1 && !dn){min--;if(min>59)min=59;sec=0;load();}
      if(reg==2 && !up){hor++;if(hor>23)hor=0;sec=0;load();}
      if(reg==2 && !dn){hor--;if(hor>23)hor=23;sec=0;load();}
      while(!sel || !up || !dn)affich();
    }
    affich();
  }
}
//*****************************************************************************
void main()
{
 setup_adc_ports(AN0|VSS_VDD);
  setup_adc(ADC_CLOCK_INTERNAL );
 // setup_adc(ADC_OFF|ADC_TAD_MUL_0);
  setup_psp(PSP_DISABLED);
  setup_spi(FALSE);
  setup_wdt(WDT_OFF);
  setup_timer_0(RTCC_INTERNAL|RTCC_DIV_128|RTCC_8_bit);
  setup_timer_1(T1_DISABLED);
  setup_timer_2(T2_DISABLED,0,1);
  setup_timer_3(T3_DISABLED|T3_DIV_BY_1);
  setup_comparator(NC_NC_NC_NC);
  setup_vref(FALSE);
  setup_oscillator(False);
```

```
  enable_interrupts(int_rtcc);
  enable_interrupts(global);
  set_tris_b(0x07);
  set_tris_c(0x00);
  set_tris_d(0x00);
  set_tris_e(0x00);
   init();
  // TODO: USER CODE!!
  while(1)
  {
    affich();
    if(flag_time){flag_time=0;inc_time();if(flag_v){flag_v=0;calc_moy();}}
    if(!sel)reglage();
    if(flag_transit)//tester le flag d'initalisation
    {flag_transit=0;vit_moy=0;compt=0;somme=0;load();}// initialisaton
     }
}
```